FLEXE: Investigating Federated Learning in Connected Autonomous Vehicle Simulations

Wellington Lobato*, Joahannes B. D. da Costa*†, Allan M. de Souza*, Denis Rosário‡,
Christoph Sommer†, Leandro A. Villas*
*University of Campinas (UNICAMP), Institute of Computing (IC), Brazil
‡Federal University of Pará (UFPA), Institute of Natural Sciences (ICEN), Brazil
†TU Dresden, Faculty of Computer Science, Germany
{wellington.lobato, joahannes.costa, allan.souza, leandro}@ic.unicamp.br, denis@ufpa.br, cms-labs.org/people/sommer

***Abstract*—Due to the increased computational capacity of Connected and Autonomous Vehicles (CAVs) and worries about transferring private information, it is becoming more and more appealing to store data locally and move network computing to the edge. This trend also extends to Machine Learning (ML) where Federated learning (FL) has emerged as an attractive solution for preserving privacy. Today, to evaluate the implemented vehicular FL mechanisms for ML training, researchers often disregard the impact of CAV mobility, network topology dynamics, or communication patterns, all of which have a large impact on the final system performance. To address this, this work presents FLEXE, an Open Source extension to Veins that offers researchers a simulation environment to run FL experiments in realistic scenarios. FLEXE combines the popular Veins framework with the OpenCV library. Using the example of traffic sign recognition, we demonstrate how FLEXE can support investigations of FL techniques in a vehicular environment.**

I. Introduction

Connected and Autonomous Vehicles (CAVs) are considered one of the modern distributed networks generating a wealth of data daily [1]. CAVs rely on Machine Learning (ML) models capable of performing autonomous decision tasks, such as, dynamically adjusting the vehicle's speed, braking, steering task based on their surroundings, and other [2]. Nonetheless, CAVs's data expose sensitive information about vehicle, driver, and passengers, where a malicious parties could intercepted and misused the data.

In this context, it is becoming more and more appealing to store and process data locally by moving network computing to the edge [3]. This is possible by the increased computational capacity of CAVs and network edge nodes, as well as the worries about transferring private and sensitive information to the cloud over the network [4]. For this purpose, a privacy-preserving property makes Federated learning (FL) appealing an attractive solution for privacy-preserving ML-based application for CAVs [4].

FL allows multiple parties to train a Machine Learning (ML) model locally collaboratively using a given ML architecture without send its data and send only trained ML models to the server. FL ensures that the its performance is comparable to a model trained using a centralized approach, while protect the privacy of each data owner [3]. Therefore, CAVs could rely on FL to share their model parameters rather than their CAVs's data, and the models are aggregated at cloud servers to produce an accurate global model [5].

However, FL applied in CAVs is subject to several challenges related to data, mobility, and communication resources that impacts the performance of FL. For instance, CAVs mobility and communication channel varies dynamically in the vehicular environment, resulting in frequent drop-outs and hand-overs [6]. In this sense, the transmission delay caused by FL parameter drop-outs and hand-overs can be much longer than the time devices take to train their local ML models, which influences the convergence time of the global model. In addition, there is a lack of communication bandwidth and vulnerability to malicious CAVs [6], where it is necessary to design communication-efficient FL mechanisms that can significantly improve the global model's accuracy and convergence speed, allowing it to be used to train large-scale ML models. To the best of our knowledge several approaches [7]–[9] failed to account for CAV mobility, intermittent communication, or network dynamics, putting the evaluation of aggregated models and FL schemes at risk.

In this paper, we investigate FL under a realistic CAV scenario. To this end, we propose FLEXE[1], an extension to the well known and widely used Veins simulation framework [10]. FLEXE introduces several enhanced capabilities, enabling realistic studies of vehicular FL and ML applications. Specifically, FLEXE relies on the capabilities of Veins to simulate both the communication among vehicles as well as their mobility within the road network. In addition, we integrated the Veins framework with OpenCV (Open Source Computer Vision Library) to develop feed-forward artificial neural networks and the Federated Averaging (FEDAvg) model aggregation algorithm. In brief, the key contributions of this paper are: i) We present FLEXE, an Open source under the terms of a GPL license simulation framework for evaluating vehicular FL applications and communication aspects. ii) We demonstrate the flexibility and applicability of simulation studies about FL in CAVs via IEEE 802.11p.

The remainder of this paper is structured as follows. Section II introduces the related works in FL simulations. Section III presents architecture and components used as a

[1]http://www.lrc.ic.unicamp.br/~wellington/

IEEE 96th Vehicular Technology Conference (VTC-Fall 2022)

guideline for the FLEXE. Section IV introduces the simulation scenario of the FLEXE. Finally, Section V describes this paper's conclusion and presents some future work directions.

II. Related Work

Beutel *et al.* [7] presented a unified approach to FL analytics and evaluation, called Flower. This framework allows large-scale FL experiments to consider richly heterogeneous FL device scenarios. The framework has an Application Programming Interface (API) for using different ML platforms, such as TensorFlow and PyTorch. TensorFlow Federated (TFF)[2] is an open source library for ML focused on decentralized data. In summary, TFF is a robust and extensible framework for conducting FL surveys by simulating federated calculations on realistic proxy datasets. TFF hosts multiple datasets that are representative of the characteristics of real-world problems that can be solved with FL. TFF can also simulate attacks targeting FL systems and differentiated privacy-based defenses. Besides, TFF provides some base classes for the FedAvg, federated stochastic gradient descent (FedSGD) algorithms, and a simple implementation of the federated evaluation. However, both works do not consider the inherent vehicular characteristics of mobility and internet connectivity.

Schettler *et al.* [11] proposed Veins-GYM to implement Reinforcement Learning (RL) in the CAVs environments. The authors combine the Veins framework with OpenAI Gym[3], which allows them to efficiently formulate and test ML/RL solutions in the domain of Intelligent Transportation Systems. The framework was evaluated using a problem for selecting the optimal communication technology in a heterogeneous communication scenario and presented different hand-crafted, learning-based, and hybrid approaches. On the other hand, Veins-GYM does not consider an FL architecture and its application scenario, instead focusing on RL techniques.

Li *et al.* [8] developed a framework for efficiently building simulators for FL. Unlike Ad Hoc simulators, FLSim[4] is envisioned as an open repository of simulator building blocks. Developers can create different simulators by combining the selected components, allowing researchers to focus on the studied problems. Dimitriadis *et al.* [9] introduced "Federated Learning Utilities and Tools for Experimentation" (FLUTE)[5], an open source platform for FL research and offline simulations. FLUTE enables the prototyping and simulation of new FL algorithms at scale, including novel optimization, privacy, and communications strategies. However, both works focus on the development and behavior presented by the applications that use federated learning. In addition, the works do not consider intermittent communication and the impacts on the federated model accuracy.

By analyzing the related works, we argue that a realistic network simulation is critical to analyze FL approaches, since FL heavily rely on communication to exchange the models parameters. Most of existing approaches [7], [8] did not take into account CAV mobility, intermittent communication, or network dynamics, which could jeopardize the evaluation of aggregated models and FL schemes. Furthermore, some approaches [7], [9] require a more controlled and specific environment to function properly.

[2]https://www.tensorflow.org/federated
[3]https://www.gymlibrary.ml/
[4]https://github.com/facebookresearch/FLSim
[5]https://github.com/microsoft/msrflute

III. FLEXE Structure

We developed FLEXE to make it possible to implement and develop vehicular FL applications within the context of CAVs. It further simplifies the process of modeling specific ML and FL applications into environments suitable for CAVs. FLEXE integrates the Veins framework [10] with OpenCV [12] to implement feed-forward artificial neural networks. Specifically, we developed FLEXE on top of the veins network simulator to simulate the dynamics of communication between vehicles. In addition, FLEXE relies on OpenCV, which is a well-known open source computer vision and ML library to allow data to be formatted locally before being transmitted to others CAVs or Road Side Units (RSUs) inside the communication range. Figure 1 depicts the schematic overview of the extended simulation framework.

We implemented the model of training and transmission on top of Veins framework, version 5.2. We use OpenCV version 4.5.5 to implement the model training and testing. For the simulation of traffic and vehicle mobility, we employed SUMO version 1.8.0. This allows us to reproduce the desired vehicle movements with random cruise speed and Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) interactions according to empirical data.

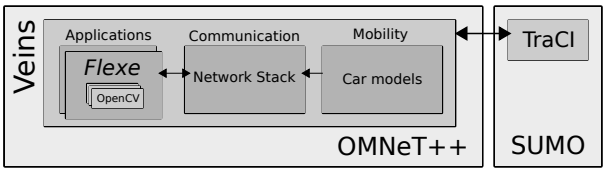


Fig. 1. FLEXE schematic structure

A. CAVs Network Dynamics Simulator

Veins is an OMNeT++ network simulator that includes the complete vehicular communication stack based on IEEE 802.11p and tailored channel models. It also models a realistic node mobility using the Simulation of Urban MObility (SUMO) road traffic simulator, SUMO exposes the TraCI interface, which connects the net- work and traffic simulation frameworks. The Veins framework updates the mobility model every time a vehicle moves in order to replicate the movement in the corresponding OMNeT++ node.

FLEXE considers the network's dynamic changes to evaluate the behavior of CAVs with greater realism, which is critical for developing new mechanisms and FL applications in the vehicular environment. Veins collects statistics on received packets and failures, as well as the internals of the state machine, which are methods for thoroughly evaluating the network behavior of the FL applications.

Veins concentrated on the communication layer, which is critical for actual channel access and packet transmission. Each OMNeT++ node is linked to a network stack that includes an IEEE 802.11p wireless network interface, a beaconing protocol, and one or more applications. This network stack (Communication Layer) is directly connected to FLEXE (Application Layer). The communication layer notifies the FLEXE layer of several events, such as successful or unsuccessful packet reception, channel busyness or idleness, and incorrect packet decoding. Veins also considers the impact of obstacles on using a packet reception model. Assuming that each obstacle is a polygon, the received power is reduced based on the number of edges intersected by the signal and the distance covered within polygons.

B. Machine Learning for CAVs Research

FLEXE relies on OpenCV library to develop the Multi-Layer Perceptron (MLP) and to simulate ML applications in the vehicular scenario. OpenCV provides a common infrastructure for computer vision applications and speed up machine perception's incorporation. A high-level interface was provided for processing, capturing, and presenting image data. The OpenCV library includes algorithms based on artificial neural networks, support vector machines, the K-nearest neighbor algorithm, and many more.

The most common type of neural network is MLP. MLPs are typically composed of at least three layers, the input layer, output layer, and one or more hidden layers. The first layer containing a neuron for each input feature of the data and the last layer containing a node for each class label. The layer in between is referred to as the hidden layer. Each layer of MLP contains one or more neurons that are linked to neurons in the preceding and following layers. We use the CV::ML::ANN_MLP class from the OpenCV library to perform local training of the model in each vehicle.

First, using the non-default constructor, an MLP with the specified topology is created. The initial weights are all set to zero. The network is trained with a set of input and output vectors, we use the interface of training data CV::PTR<CV::ML::TRAINDATA> to load the dataset. Unsupervised learning algorithms use training data with no response to learn the structure of the supplied data based on distances between different samples. In supervised learning algorithms, which learn the function mapping samples to responses, training data with a response is used. Typically, the responses are scalar values, either ordered or categorical. The training procedure can be repeated multiple times, with the weights being adjusted based on new training data.

C. Implementing FL Capabilities in Veins

We designed the FLEXE sequence diagram involving local training and testing of the model, and also the aggregation and update of the global model, as depicted in Figure 2. We assume that each vehicle (client) gets a random subset of the data. This data is then distributed to various clients in order to train collaboratively, and the communication round begins. Each communication round consist in 4 steps, and these steps comprise the vehicular FL. The procedure entails sending the current global model state to participating clients, training local models to generate a set of potential model updates, and then aggregating these local updates into a single global update and applying it to the global model. In this sense, we demonstrate the functioning and dynamics considered in an FL scenario simulated by FLEXE.

The initial step (*i.e.*, step a) of the vehicular FL occurs when the global model is disseminated to the clients by the remote server. In each client, the model is created using the OpenCV library. After that, the second step (*i.e.*, step b) is initiated in each CAV separated, where it conducts the local model training based on local data. We use the OpenCV's training data interface to split the data for each client and train the local model with the MLP class. Local model training may take different times for different clients, depending on local training data. Once the local training is completed, the generated local model is uploaded to the server, where it is encapsulated and sent using the veins module. It is important to note that the transmitted packet may be lost at this point due to the network dynamics depicted in the vehicular scenario. Finally, the server generates the new global model based on aggregating the collected local models (*i.e.*, step c) and sends the updated global model (*i.e.*, step d).

In the FLEXE architecture, we implemented the widely and predominant used FL algorithm for aggregation called FedAvg [13]. The aggregation is made in step c, as shown in Figure 2, where the server receives the minimum number of models from different clients and averages the local models to compute the updated global model. FedAvg assumes that all clients are willing to join each communication round for FL training.

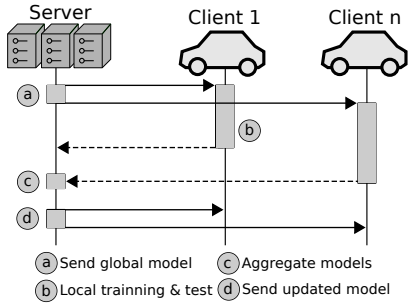
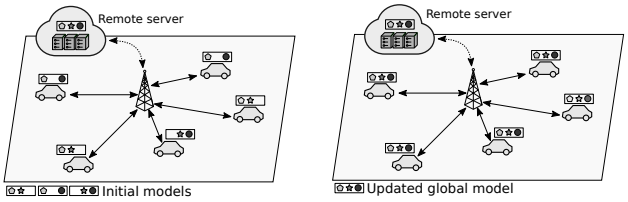


Fig. 2. FLEXE Framework Sequence Diagram

IV. PROOF OF CONCEPT

To demonstrate the possibilities opened up by FLEXE, we present a showcase of Traffic Sign Classification in the vehicular FL environment. This section also details the methodology and metrics used to evaluate the accuracy of the aggregated models in two different communication scenarios: with or without considering that buildings act as obstacles to the propagation of wireless transmissions. Subsequently, we



(a) Initial model (b) Updated model

Fig. 3. FLEXE simulation scenario with main components

evaluate the impact of the different numbers of vehicles in the aggregated model.

A. Scenario description

In order to evaluate the impacts of vehicular mobility in the FL models, we conducted simulations in a Manhattan Grid scenario. The scenario is a 1 km^2 fragment of Manhattan, USA, with several blocks and two-way streets so the vehicles can move in opposite directions. The vehicle densities varied from 20 to 200 vehicles/km^2. Vehicles in the simulation have the same dimensions, mean speed, and standard deviation speed. Furthermore, we use a random mobility model to generate the vehicles' routes, so that different paths are generated for each vehicle for each replication and density. We conducted 10 simulation runs with different randomly generated seeds.

As shown in Figure 3, a server node with a single global model and a hundred participating vehicles comprise the simulation scenario. We also consider putting an RSU in the middle of the scenario to cover all vehicles. It is assumed that all participating vehicles are willing to train the FL model using local data. Each CAV has one hidden layer, which contains 50 neurons with Symmetrical Sigmoid activation functions. Each communication round, the CAVs train the local model with 75 epochs.

We used the Chinese Traffic Sign Database [14] as the training dataset for evaluating the proposed scheme to evaluate the behavior of FL applications in the vehicular context. There are 6164 traffic sign images and 58 sign classes in the Traffic Sign Recognition Dataset. Each vehicle was allocated samples randomly from 58 classes. Training samples were changed in the whole simulation. The entire database is divided into two sub-databases: training and testing. There are 4170 images in the training database and 1994 images in the testing database. All images are labeled with the four sign coordinates and the category coordinates. All the main simulation parameters are summarized in Table I.

B. Simulation Results

Figure 4 presents the learning accuracy results for the application of Traffic Sign Recognition with different numbers of vehicles. Different vehicle densities and mobility were evaluated to analyze the impact on the accuracy of the trained and aggregated models. It is worth noting that the vehicles had the same training conditions, varying only in the training data. We can see, in Figure 4(a) and Figure 4(b), that a small

TABLE I
SIMULATION PARAMETERS

	Parameter	Value
communication	Simulation area	1 km^2
	Number of road segments	9
	Scenario	Grid
	Number of vehicles	{20, 40, 60, 80, 100, 200}
	Vehicles speed	13.84 m/s (50 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
FL configuration	Minimum number of clients	2
	Learning rate	0.000001
	Number of epochs per round	75
	Activation function	Symmetrical Sigmoid
	α & β	0.0 & 0.0
	Hidden layer formation	2150 × 50 × 58

number of vehicles results in a lower accuracy of the aggregate models. This is due to the fact that there are fewer customers available to contribute to the global model. The same result can be observed with the total number of communications rounds, the results being less than 10 rounds in total.

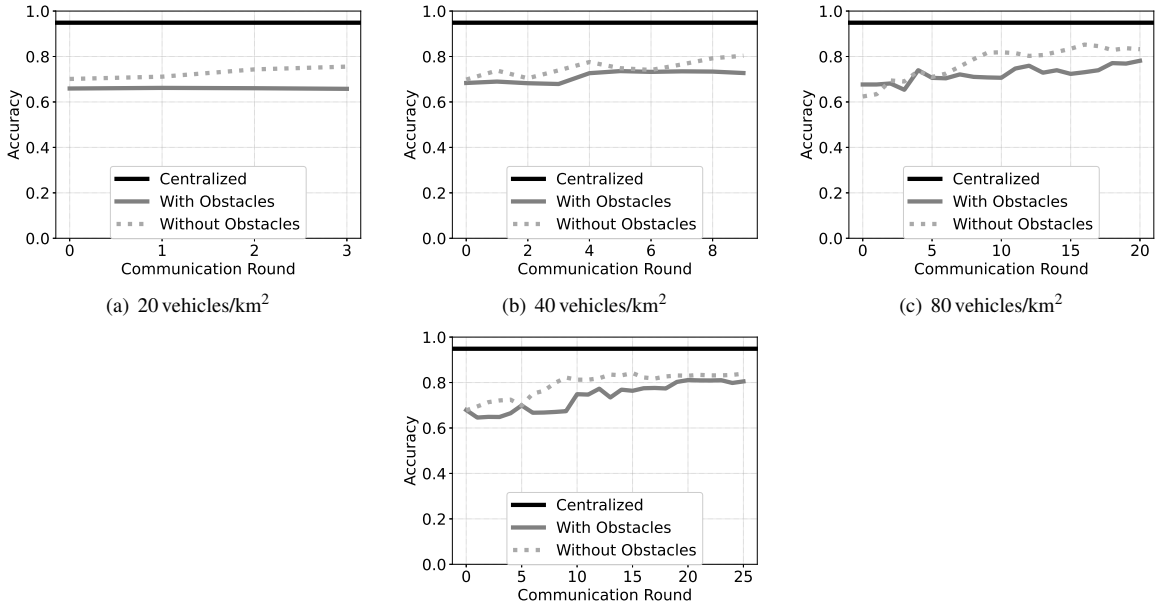
All vehicles in the scenario also transmit beacons with a frequency of 1 Hz to represent a concurrent application. It is possible to observe in Figure 4(c) that the impact of the evaluated scenarios on the accuracy values, we can observe that the accuracy values for the scenario without obstacles were 10% higher compared to the accuracy values in the scenario with obstacles. However, this difference tends to be reduced with a greater number of communication rounds. In the simulation scenarios, everyone had the same period of 200 simulation times to perform the message exchanges.

With a higher density of vehicles, the difference between the models tends to be reduced, as shown in Figure 4(d). This is due to the fact that there are more clients and, consequently, more communication rounds available to converge the global model, which lessens the impacts caused by block transmissions. The highest values observed in the accuracy occurred with the density of 100 vehicles, as can be seen in Figure 4(d), with a value of 84% in the scenario with obstacles and 87% in the scenario without obstacles. In addition, we simulated the centralized scheme with 94% accuracy versus the FL scheme.

We conclude from the simulation results that FLEXE provides a realistic network simulation for vehicular FL applications, which is important when analyzing FL approaches because they rely heavily on communications to function.

V. CONCLUSIONS AND FUTURE WORK

We described the structure and aggregate model of FLEXE, a new framework for simulation of FL in CAV. Its adaptable design allows for the implementation of a variety of FL schemes, including horizontal, vertical, and Federated Transfer Learning. FLEXE is free to download and use, built for customization, and allows for the realistic simulation of wireless



(a) 20 vehicles/km^2 (b) 40 vehicles/km^2 (c) 80 vehicles/km^2

(d) 100 vehicles/km^2

Fig. 4. Impact of vehicle density on the accuracy

networking and vehicle dynamics. The paper demonstrates the FLEXE simulator's potential by focusing on one application, Traffic Sign Classification. FLEXE, we believe, can be a useful research tool for large-scale analysis and comparison of vehicular FL systems prior to their actual deployment. Following the tradition of Veins, our contributions are available for the research community[6].

As future works we aim to provide support for other ML platforms such as TensorFlow[7] and Darknet[8]. In addition, we aim to include more complex aggregation operations and additional parameters to allow for even more personalization when accounting for non-i.i.d data in FL.

ACKNOWLEDGMENT

The authors would like to thank the São Paulo Research Foundation (FAPESP), grants #2015/24494-8, #2019/19105-3, #2018/16703-4, and #2021/13780-0.

REFERENCES

[1] "Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles," SAE, Standard J3016_201806, 2018.

[2] M. Ambrosin *et al.*, "Object-level perception sharing among connected vehicles," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, Oct. 2019, pp. 1566–1573.

[3] Q. Yang, S. Fu, H. Wang, and H. Fang, "Machine-Learning-Enabled Cooperative Perception for Connected Autonomous Vehicles: Challenges and Opportunities," *IEEE Network*, vol. 35, 2021.

[4] A. Du, Y. Shen, L. Tseng, T. Higuchi, S. Ucar, and O. Altintas, "Enabling Pervasive Federated Learning using Vehicular Virtual Edge Servers," in *2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, IEEE, Mar. 2021, pp. 324–327.

[6]http://www.lrc.ic.unicamp.br/~wellington/

[7]https://www.tensorflow.org/

[8]https://github.com/pjreddie/darknet

[5] S. AbdulRahman, H. Tout, A. Mourad, and C. Talhi, "FedMCCS: Multicriteria Client Selection Model for Optimal IoT Federated Learning," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4723–4735, Mar. 2021.

[6] N. Cha *et al.*, "Fuzzy Logic Based Client Selection for Federated Learning in Vehicular Networks," *IEEE Open Journal of the Computer Society*, vol. 3, pp. 39–50, 2022.

[7] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, T. Parcollet, and N. D. Lane, "Flower: A Friendly Federated Learning Research Framework," *arXiv preprint arXiv:2007.14390*, 2020.

[8] L. Li, J. Wang, and C. Xu, "FLSim: An Extensible and Reusable Simulation Framework for Federated Learning," in *International Conference on Simulation Tools and Techniques*, Springer, 2020, pp. 350–369.

[9] D. Dimitriadis, M. H. Garcia, D. M. Diaz, A. Manoel, and R. Sim, "Flute: A scalable, extensible framework for high-performance federated learning simulations," *arXiv preprint arXiv:2203.13789*, 2022.

[10] C. Sommer, R. German, and F. Dressler, "Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis," *IEEE Transactions on Mobile Computing*, vol. 10, 2011.

[11] M. Schettler, D. S. Buse, A. Zubow, and F. Dressler, "How to Train your ITS? Integrating Machine Learning with Vehicular Network Simulation," in *2020 IEEE Vehicular Networking Conference (VNC)*, IEEE, Dec. 2020.

[12] G. Bradski, "The openCV library.," *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, vol. 25, no. 11, pp. 120–123, 2000.

[13] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, PMLR, 2017, pp. 1273–1282.

[14] H. Zhang, B. Wang, Z. Zheng, and Y. Dai, "A novel detection and recognition system for Chinese traffic signs," in *Proceedings of the 32nd Chinese Control Conference*, IEEE, 2013.