This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LNET.2021.3058292, IEEE Networking Letters

Securing the Internet of Vehicles: A Deep Learning based Classification Framework

Tejasvi Alladi, Varun Kohli, Vinay Chamola, *Senior Member, IEEE*, F. Richard Yu, *Fellow, IEEE*

***Abstract*—Along with the various technological advancements, the next generation vehicular networks such as the Internet of Vehicles (IoV) also bring in various cybersecurity challenges. To effectively address these challenges, in addition to the existing authentication techniques, there is also a need for identification of the misbehaving entities in the network. This letter proposes a deep learning-based classification framework to identify potential misbehaving vehicles before the communication requests from the On Board Units (OBUs) of the vehicles can be entertained by the network infrastructure such as the Road Side Units (RSUs). The evaluated metrics demonstrate the performance of the proposed classification approaches.**

***Index Terms*—Internet of Vehicles (IoV), deep learning, intrusion detection, edge computing.**

I. Introduction

The Internet of Vehicles (IoV) is a new paradigm of vehicular networks inspired by the adoption of the Internet of Things (IoT) in Vehicular Ad-hoc Networks (VANETs). IoV is expected to usher in an era of connected vehicles, which use their On Board Units (OBUs) to communicate with each other and with the road-side infrastructure generally called Road Side Units (RSUs) [1].

Nevertheless, the growing number of Vehicle to Vehicle (V2V) and Vehicle to Infrastructure (V2I) communication links in these next-generation vehicular networks also increases the potential attack surfaces in the networks. And with the increasing number of attack surfaces, there is a higher possibility of cybersecurity attacks [2, 3]. Thus, there is a greater need for developing security solutions for these next-generation vehicular networks such as IoV networks. Traditional solutions such as cryptography-based authentication techniques have been extensively proposed in the existing literature [4, 5]. However, more robust solutions for identifying misbehaving and malicious vehicles are needed. Some intrusion identification and detection techniques inspired by artificial intelligence have been proposed in the literature. The authors of [6] proposed a misbehavior detection technique using machine learning in vehicular communication networks, but their approach detects only internal attacks in the network. However, since the number of connected vehicles in the IoV scenario is exponentially higher compared to traditional vehicular networks, the data generated is also much higher. The classical intrusion detection approaches based on statistics or even machine learning may not be sufficient to address such high data applications. Hence, there is a need for deep learning-based classification approaches to identify potential attack scenarios. Existing literature also has works based on some deep learning techniques. Van Wyk *et al.* proposed a sensor anomaly detection technique using Convolutional Neural Networks (CNNs) [7]. Another intrusion detection technique using deep learning techniques was proposed by Loukas *et al.* [8]. However, both approaches consider only a limited number of attacks in their works. Thus, along with using a deep learning-based classification approach, there is also a need to consider the growing number of possible attack scenarios.

Another aspect of consideration is regarding the deployment of the proposed classification approach. The architecture proposed by Loukas *et al.* is deployed on the cloud servers, however, in the literature, cloud deployment has been shown to be both cost and computation-intensive. Instead edge computing as an alternative to cloud computing has been widely discussed. Even in vehicular networks, task offloading to edge computing is quite promising as shown in recent works [9]. An intelligent path planning scheme using the concepts of edge computing has been proposed by researchers in [10]. Another work [11] combined deep reinforcement learning with edge servers in vehicular networks.

This letter combines the concepts of deep learning and edge computing with vehicular networks. The following are the major contributions of this letter:

i. We propose a deep learning framework for securing the IoV networks backed by edge computing devices.
ii. We present two classification approaches inspired by deep learning techniques for classifying the misbehaving vehicles in the network.
iii. We demonstrate the performance of both the approaches by simulating various deep learning models for both the classification approaches.

This letter is organized as follows: we describe the network model in Section II. The classification approaches taken and the deep learning models considered are described as part of the proposed framework in Section III. The dataset used, the simulation environment, and the evaluation results are discussed in Section IV. Section V concludes the letter.

Manuscript received October 13, 2020; revised January 22, 2021; accepted February 05, 2021. The associate editor coordinating the review of this article and approving it for publication was M. Dong (Corresponding author: F. Richard Yu). This work was partially supported by the SICI SICRG grant received by Dr. Vinay Chamola and Prof. Richard Yu for the project Artificial Intelligence Enabled Security Provisioning and Vehicular Vision innovations for Autonomous Vehicles, and also through the Government of Canada's National Crime Prevention Strategy and Natural Sciences and Engineering Research Council of Canada (NSERC) CREATE program for Building Trust in Connected and Autonomous Vehicles (TrustCAV).

T. Alladi and F. Richard Yu are with School of Information Technology, Carleton University, Ottawa, ON K1S 5B6, Canada. (e-mail: talladi.carleton@gmail.com, richard.yu@carleton.ca).

V. Kohli and V. Chamola are with the Department of Electrical and Electronics Engineering & APPCAIR, BITS-Pilani, Pilani Campus, 333031, India. (e-mail: {f20170374, vinay.chamola}@pilani.bits-pilani.ac.in).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LNET.2021.3058292, IEEE Networking Letters

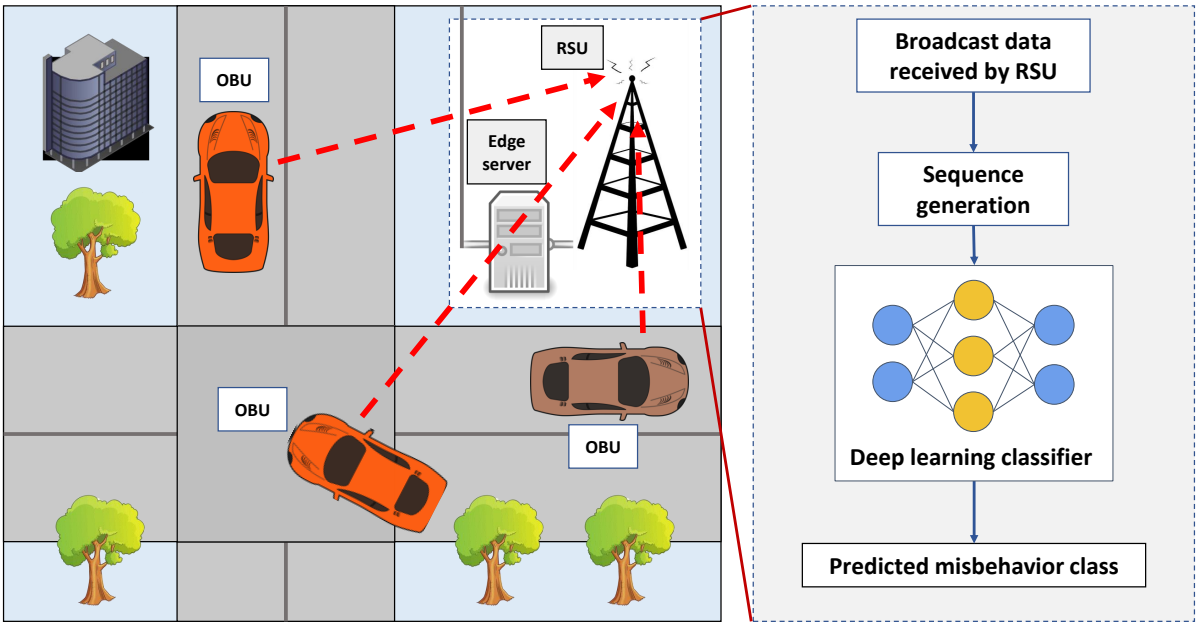

Fig. 1: Network model.

II. Network Model

In this study, the considered IoV network model is a network of interconnected vehicles that communicate with each other (V2V communication) and with the RSUs (V2I communication) which are deployed at major road intersections. Deep learning classifiers are deployed on the edge servers that are co-located with these RSUs. The data broadcasted by the vehicles is received by the nearest RSU, which then passes it on to the edge server to check for possible intrusion in the network. This network model is shown in Fig. 1. A moving vehicle can exhibit either normal behavior or misbehavior where misbehavior could be a faulty data transmission or a possible cybersecurity attack. As shown in the figure, the broadcasted messages received by the edge server from each vehicle is converted into time sequences and then passed through the deep learning classifiers to predict it as one of the classes defined. Based on the granularity required for classification of the data received, classes are defined in the proposed framework as discussed in the next section.

III. Proposed Framework

In this section, we discuss the proposed classification approaches and the various deep learning models considered for carrying out the classification tasks.

A. Classification Approaches

We propose two different deep learning-based classification approaches for intrusion detection in the considered network.

1) Coarse-grained classification method: This approach is a two-class coarse-grained classification method (CGCM) that distinguishes normal vehicle data from misbehavior data. In this approach, the faults and attacks are grouped into a single misbehavior (Faults + Attacks) class. When the input data is passed through the CGCM classifier, every possible fault and attack type data is classified into the misbehavior class, while the normal vehicle data is classified into the normal class.

2) Fine-grained classification method: The second approach is a fine-grained classification method (FGCM) with three predicted classes based on normal vehicle behavior, faulty behavior, or attack behavior. Thus, compared to the first approach this is a more fine-grained classification approach. When the input data is passed through the FGCM classifier, normal vehicle data is classified into normal class, the fault type data into the fault class and the attack type data into the attack class.

B. Deep Learning Models

Two types of deep neural networks called Long Short-term Memory (LSTM) and CNN are considered in this work which are the building blocks for the deep learning models which go into the proposed classifiers. LSTMs are a type of Recurrent Neural Networks (RNNs) which are found to be quite efficient in classifying time series data due to the feedback loops present in their architecture that can remember temporal data. Thus they find their applications in temporal sequence classification problems such as intrusion detection and speech recognition. CNNs are another type of deep neural networks which work best on visual images. Using an architecture of sliding filters and convolutional input layers, they can identify complex patterns in the input data.

In this letter, we use two deep learning architectures namely stacked LSTM and CNN-LSTM. Stacked LSTMs are created using multiple layers of LSTMs stacked one after the other, while CNN-LSTMs are intelligent combinations of CNN and LSTM layers. Four different models based on these architectures are used here. The CNN-LSTM model (**Model 1**)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LNET.2021.3058292, IEEE Networking Letters

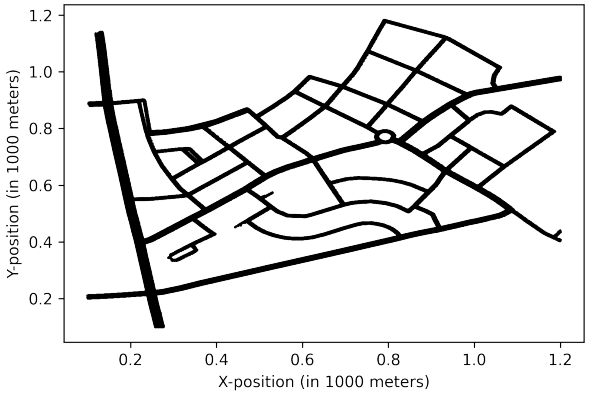

Fig. 2: Road map.

considered consists of two 1D CNN layers with 1024 and 512 filters each, followed by a max-pooling layer, an LSTM layer of 512 units, and an output dense layer of 1 unit. The CNN and dense layers are activated by rectified linear unit (ReLU) activation functions. Three different stacked LSTM models are considered in this study, namely, 3-LSTM model (**Model 2**) with three stacked LSTM layers, 4-LSTM model (**Model 3**) with four stacked LSTM layers, and 5-LSTM model (**Model 4**) with five stacked LSTM layers, followed by a single neuron output dense layer which is activated by a ReLU activation function. Each LSTM layer in the stacked LSTM models consists of 256 units. The final LSTM layer in the stacked LSTM models has its return sequences parameter set to false.

IV. Simulation and Results

In this section, we discuss the test, validation, and train dataset split-up, testing and training approaches, and the performance evaluation of the four models considered.

A. Dataset

We used the popular VeReMi Extension dataset [12] in this research. It covers several misbehavior types including both faulty transmissions (henceforth called faults), and cybersecurity attacks (henceforth called attacks) [13]. Faults in this dataset pertain to incorrect position and velocity values. There is data corresponding to eight such faults in this dataset namely constant position/velocity, constant position/velocity offset, random position/velocity, and random position/velocity offset. The dataset also consists of the following nine attacks namely Denial of Service (DoS), data replay, disruptive, DoS random, DoS disruptive, data replay sybil, traffic congestion sybil, DoS random sybil, and DoS disruptive sybil. The data available in this dataset is in a rudimentary form, comprising of individual messages of every single simulated vehicle.

The messages from each vehicle were used to create our dataset comprising of time sequences for each vehicle. Each of these generated sequences is 20x7 in size containing 20 messages per sequence and seven data fields, namely X, Y position coordinates of the vehicle, X, Y velocity coordinates of the vehicle, timestamp at which the message was broadcast, the pseudo-identity of the vehicle, and label for the vehicle's class type. The road map used in the simulation environment is plotted in Fig. 2 where the original X and Y position coordinates (in meters) are shown to be scaled down by a factor of 1000. For creating a dataset with all data fields on a similar scale, we scaled down the position coordinates accordingly. We discuss the dataset split up for both the classifiers below.

1) Dataset for the CGCM classifier: The train set generated for the CGCM classifier has a total sample size of 85000 sequences: 42500 each for normal behavior and misbehavior, with 2500 sequences for each of the 17 misbehavior types considered. The validation and test sets comprise 13600 sequences (6800 each for normal and misbehavior) and 6800 sequences (3400 each for normal and misbehavior) respectively.

2) Dataset for FGCM classifier: The train set for the FGCM classifier consists of 59998 input sequences, nearly 20000 for each class. There are 20000 normal behavior sequences, 2500 sequences for each of the 8 fault types to obtain 20000 sequences of faults, and 2222 sequences each for the 9 attack types, making a total of 19998 attack sequences. This division is done to have an equal amount of data for each class. The validation set and test set are also created similarly with a total size of 9595 sequences (3200, 3200, 3195 sequences for normal, faults and attacks) and 4793 sequences (1600, 1600, 1593 sequences for normal, faults and attacks) each.

B. Training and Testing

Keras, a Python library for deep learning was used for the development of the models, and all the experimental training was carried out on the Google Colaboratory environment [14]. We used Adam optimizer with a learning rate of 0.0003. The four models discussed (Model 1-4), were trained on mean absolute error ($\mathcal{MAE}$) loss which is defined in the following equation where x is the input value, x_p is the predicted value and n is the total number of data-points over which the $\mathcal{MAE}$ is calculated. Testing was conducted on Intel i5 7^{th} Generation workstation using the Jupyter Notebook environment.

$$\mathcal{MAE} = \frac{1}{n}\sum_{i=1}^{n}|x - x_p| \tag{1}$$

C. Evaluation

In this section, we evaluate both the classifiers in terms of the four popular evaluation metrics after passing the test dataset through these classifiers. The four metrics accuracy, precision, recall, and F1-score denoted by $\mathcal{A}, \mathcal{P}, \mathcal{R}, \mathcal{F}1$ respectively are defined below. Here $\mathcal{TP}, \mathcal{TN}, \mathcal{FP}, \mathcal{FN}$ refer to True Positive, True Negative, False Positive, and False Negative respectively.

$$\mathcal{A} = \frac{\mathcal{TP} + \mathcal{TN}}{\mathcal{TP} + \mathcal{FP} + \mathcal{TN} + \mathcal{FN}} \tag{2}$$

$$\mathcal{P} = \frac{\mathcal{TP}}{\mathcal{TP} + \mathcal{FP}} \tag{3}$$

$$\mathcal{R} = \frac{\mathcal{TP}}{\mathcal{TP} + \mathcal{FN}} \tag{4}$$

$$\mathcal{F}1 = \frac{2\mathcal{PR}}{\mathcal{P} + \mathcal{R}} \tag{5}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LNET.2021.3058292, IEEE Networking Letters

	Model 1				Model 2				Model 3				Model 4			
Class	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}1$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}1$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}1$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}1$
Normal	0.933	0.893	0.984	0.936	0.95	0.926	0.977	0.951	0.973	0.951	0.998	0.974	0.965	0.936	0.999	0.966
Faults + Attacks	0.933	0.982	0.882	0.93	0.95	0.976	0.922	0.948	0.973	0.998	0.948	0.974	0.965	0.999	0.932	0.964
Average	0.933	0.9375	0.933	0.933	0.95	0.951	0.9495	0.9495	**0.973**	**0.9745**	**0.973**	**0.974**	0.965	0.9675	0.9655	0.965

TABLE I: Evaluation metrics for the course-grained classification method (CGCM)

	Model 1				Model 2				Model 3				Model 4			
Class	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}1$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}1$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}1$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}1$
Normal	0.924	0.829	0.973	0.895	0.939	0.861	0.977	0.915	0.982	0.958	0.989	0.973	0.934	0.852	0.971	0.908
Faults	0.926	0.976	0.799	0.879	0.938	0.975	0.839	0.902	0.981	0.987	0.957	0.972	0.935	0.969	0.834	0.896
Attacks	0.992	0.985	0.992	0.988	0.991	0.984	0.989	0.987	0.994	0.992	0.991	0.991	0.994	0.991	0.991	0.991
Average	0.947	0.93	0.921	0.921	0.956	0.94	0.935	0.935	**0.986**	**0.979**	**0.979**	**0.979**	0.954	0.937	0.932	0.932

TABLE II: Evaluation metrics for the fine-grained classification method (FGCM)

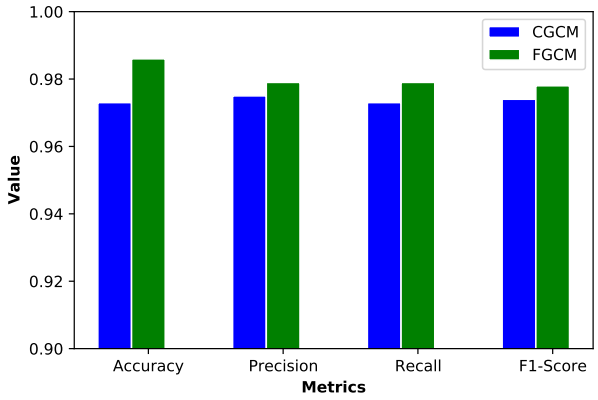

Fig. 3: Comparison of the best average evaluation metrics for both the classification approaches.

Table I lists the evaluation metrics of all the four models for CGCM classifier, while Table II lists the evaluation metrics of the models for FGCM classifier. It can be observed in both the tables that the performance of all the stacked LSTM models across all the four metrics is higher compared to the CNN-LSTM model. Further, the 4-LSTM model performs the best across all the stacked LSTM models considered. Among both the classifiers, the FGCM classifier seems to be performing better where the models can better classify faults and attacks individually. The best results of both the classifiers are highlighted in bold in both the tables. These results are also plotted in Fig. 3 showing a comparison of the best average evaluation metrics for both the classifiers. Thus, a fine-grained classification based on the FGCM classifier seems to be a better approach since the performance is higher and at the same time, we can classify attacks and faults separately. Applications requiring identification of attacks and faults as different classes can benefit from this approach.

V. Conclusion

This letter proposed deep learning-based classification approaches for identifying and classifying misbehaving vehicles in the IoV networks. Two classification approaches were considered where one is a coarse-grained classification of normal vehicle data and all possible misbehavior types, the other is a more fine-grained classification to classify the misbehavior types into faults and attacks. Experimental results show that the fine-grained classification performs better across all the metrics and can be a better classification approach to different possible faults and attacks.

References

[1] B. Ji, X. Zhang, S. Mumtaz, C. Han, C. Li, H. Wen, and D. Wang, "Survey on the internet of vehicles: Network architectures and applications," *IEEE Communications Standards Magazine*, vol. 4, no. 1, pp. 34–41, 2020.

[2] Z. Lu, G. Qu, and Z. Liu, "A survey on recent advances in vehicular network security, trust, and privacy," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 2, pp. 760–776, 2018.

[3] T. Alladi, V. Chamola, B. Sikdar, and K.-K. R. Choo, "Consumer iot: Security vulnerability case studies and solutions," *IEEE Consumer Electronics Magazine*, vol. 9, no. 2, pp. 17–25, 2020.

[4] A. K. Sutrala, P. Bagga, A. K. Das, N. Kumar, J. J. Rodrigues, and P. Lorenz, "On the design of conditional privacy preserving batch verification-based authentication scheme for internet of vehicles deployment," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 5535–5548, 2020.

[5] G. Bansal, N. Naren, V. Chamola, B. Sikdar, N. Kumar, and M. Guizani, "Lightweight mutual authentication protocol for v2g using physical unclonable function," *IEEE Transactions on Vehicular Technology*, 2020.

[6] S. Gyawali, Y. Qian, and R. Q. Hu, "Machine learning and reputation based misbehavior detection in vehicular communication networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 8871–8885, 2020.

[7] F. Van Wyk, Y. Wang, A. Khojandi, and N. Masoud, "Real-time sensor anomaly detection and identification in automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1264–1276, 2019.

[8] G. Loukas, T. Vuong, R. Heartfield, G. Sakellari, Y. Yoon, and D. Gan, "Cloud-based cyber-physical intrusion detection for vehicles using deep learning," *IEEE Access*, vol. 6, pp. 3491–3508, 2017.

[9] T. Alladi, S. Chakravarty, V. Chamola, and M. Guizani, "A lightweight authentication and attestation scheme for in-transit vehicles in iov scenario," *IEEE Transactions on Vehicular Technology*, 2020.

[10] C. Chen, J. Jiang, N. Lv, and S. Li, "An intelligent path planning scheme of autonomous vehicles platoon using deep reinforcement learning on network edge," *IEEE Access*, vol. 8, pp. 99 059–99 069, 2020.

[11] Z. Ning, P. Dong, X. Wang, L. Guo, J. J. Rodrigues, X. Kong, J. Huang, and R. Y. Kwok, "Deep reinforcement learning for intelligent internet of vehicles: An energy-efficient computational offloading scheme," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 4, pp. 1060–1072, 2019.

[12] "VeReMi Extension," https://github.com/josephkamel/VeReMi-Dataset, 2020, [Online; accessed 20-September-2020].

[13] J. Kamel, M. Wolf, R. W. van der Hei, A. Kaiser, P. Urien, and F. Kargl, "Veremi extension: A dataset for comparable evaluation of misbehavior detection in vanets," in *Proc. IEEE (ICC)*, June 2020, pp. 1–6.

[14] "Google Colaboratory," https://colab.research.google.com/notebooks/intro.ipynb, 2020, [Online; accessed 10-October-2020].