J. Highw. Transp. Res. Dev. 2025, 19(1): 13−17

Journal of Highway and Transportation Research and Development

Home page: www.sciopen.com/journal/2095-6215

Lightweight framework for misbehavior detection in internet of vehicles

Yujing Gong*, Bin-Jie Hu**

School of Electronic and Information Engineering, South China University of Technology, Guangzhou, Guangdong 510641, China

Abstract: The proliferation of connected vehicles in the Internet of Vehicles (IoV) ecosystem has introduced new security challenges, particularly in the context of internal network attacks. Traditional public key infrastructure (PKI) technologies are no longer sufficient to ensure secure communication within a network that experiences dynamic topology changes and high vehicle density. In response, there is a growing need for a lightweight misbehavior detection framework that offers fast computation and minimal space complexity. This paper presents a novel approach using continuous-time recurrent neural networks for detecting misbehavior in the IoV and assesses their performance against the Vehicular Reference Misbehavior (VeReMi) extension dataset. We compare two recently introduced models—the liquid time-constant (LTC) network and the closed-form continuous-time (CFC) neural network—with the established convolutional neural network-long short-term memory (CNN-LSTM) model. The results indicate that continuous-time neural networks marginally outperform CNN-LSTM on evaluation metrics. Despite LTC and CFC having significantly fewer parameters, making them less complex and more space-efficient than CNN-LSTM, the latter proves to be more time-efficient. Therefore, a careful balance between runtime cost and space complexity must be considered when deploying lightweight neural networks in practical applications.

Keywords: Intelligent transport; Internet of vehicles; Closed-form continuous-time neural network; VeReMi extension dataset; Misbehavior detection

1 Introduction

IoV is an emerging cross-domain technology covering fields such as information science, communication technologies, automobile, transportation, etc. It connects vehicles to "everything" such as vehicles (V2V), infrastructures (V2I), pedestrians (V2P) and clouds (V2C) empowered by either the IEEE 802.11p protocols or the cellular technologies, providing low-latency, high-reliability and high-throughput communications for applications in the network. In China [1], taking advantages of the world's largest cellular network, C-V2X is superior in multiple aspects over the Dedicated Short-Range Communications (DSRC): a low deployment cost due to the reusage of the widely-deployed 4G networks, a continuous technology support with an initiation of long-term evolution V2X (LTE-V2X) complemented by an evolution of new radio-V2X (NR-V2X), and more. Despite its dominance, life-threatening risks still exist in the network and methods against intruders are critical. Being one of the main businesses in IoV, misbehavior detection schemes are exclusively designed for protecting the network from internal attackers, who are authenticated by the public key infrastructure-based mechanisms but are sending malicious messages in the network to misguide the traffic. Common approaches include machine learning-based schemes [2], for instance the SVM and the KNN [3], as well as the deep learning-based schemes, for instance the CNN-LSTM [4] and the Attention-based mechanisms [5]. The orientation of this paper is to compare the misbehavior detection performance of the recently proposed continuous-time recurrent neural networks, i.e., the LTC and the CFC, with that of the discrete-time neural network, i.e., the CNN-LSTM on the public dataset VeReMi extension. The comparisons are to attest to the fact that the CFC and the LTC, being lightweight in space consumption, trade time for space while the CNN-LSTM, a time-efficient approach, trades space for time. As far as we know, this is the first work to look into the tradeoffs of continuous-time neural networks versus their discrete counterparts.

1.1 Continuous-time neural networks

Discrete residual learning framework Resnet [6] given by Eq. (1):

$$h(t+1) = h(t) + f[h(t), \theta], \tag{1}$$

where, f is the residual block, and $h(t)$ is the output (or the hidden state in Recurrent Neural Network, RNN), can be extended to continuous-time recurrent neural network represented by neural ordinary differential equation (ODE) [7]. Rather than directly parameterizing the hidden state of each time point, the first-order derivative of the hidden state is modeled as Eq. (2):

$$\frac{\mathrm{d}h(t)}{\mathrm{d}t} = f[h(t), \theta], \tag{2}$$

where f is the neural network determining the rate of change of the hidden state with weights θ. Given the initial value $h(0)$, solving the equation for any given time t of the corresponding hidden state can be implemented as a Euler approximation [8].

In 2018, Hasani et al. proposed a universal approximator [9],

Received: 29 April 2024; **Revised:** 6 August 2024; **Accepted:** 30 December 2024
* *E-mail address*: yujing_gong@qq.com; ** *Corresponding author*: eebjiehu@scut.edu.cn
DOI: 10.26599/HTRD.2025.9480044

termed as liquid time-constant recurrent neural network, a subclass of continuous-time recurrent neural network. The network is proved capable of modeling any finite trajectory of certain dimensions with a minor number of computational units (including hidden units and output units). The hidden state dynamics of the network is given by Eq. (3) and Eq. (4):

$$\frac{\mathrm{d}h_i(t)}{\mathrm{d}t} = -\left\{\frac{1}{\tau_i} + \frac{\omega_{ij}}{C_{m_i}}\sigma_i\left[h_j(t)\right]\right\}h_i(t) + \left\{\frac{h_{\mathrm{leak}_i}}{\tau_i} + \frac{\omega_{ij}}{C_{m_i}}\sigma_i\left[h_j(t)\right]E_{ij}\right\}, \quad (3)$$

$$\sigma_i\left[h_j(t)\right] = \frac{1}{1+\mathrm{e}^{-\gamma_{ij}\left[h_j(t)+\mu_{ij}\right]}}, \quad (4)$$

where, (μ_{ij}, γ_{ij}), a function of $h_j(t)$, is a sigmoidal nonlinearity modeling the chemical synaptic transmission from neuron j to neuron i with ω_{ij} being its maximum weight; C_{m_i} and h_{leak_i} are neuronal parameters; E_{ij} indicates if the synapse transmits either an activation or a suppression to the succeeding neuron; τ_i is the time constant manifesting the speed and the coupling sensitivity of the ODE.

Later in 2020, Lechner et al. built a compact network named neural circuit policies (NCP) [10] with LTC neurons as its cornerstone. The network was inspired by the wiring diagram of a small species and was utilized for the lane keeping task of an autonomous driving vehicle. The network consists of 19 neurons and is constructed into 4 layers. Each neuron in the network is simply a LTC cell inside of which is a semi-implicit ODE solver employed as a black-box to approach the real dynamics of the system. Surprisingly, the proposed work has achieved a distinguishing performance than some advanced networks such as CNN and LSTM in certain aspects of an ideal autonomous controller [10].

Note that the compactness of the network is at the cost of a computationally intensive ODE solver which prevents it from scaling up to comparably sophisticated scenarios. Thus in 2022, Hasani et al. put forward a continuous-time neural network, known as the closed-form continuous-time neural network [11]. The authors started with a closed-form solution given by Eq. (5):

$$h(t) \approx (h_0 - A)\mathrm{e}^{-\left\{\frac{1}{\tau}+f[I(t),h(t);\theta]\right\}t} f[-I(t),-h(t);\theta] + A \quad (5)$$

to the LTC network simplified from Eq. (3), given below in Eq. (6):

$$\frac{\mathrm{d}h(t)}{\mathrm{d}t} = -\left\{\frac{1}{\tau} + f[h(t),I(t);\theta]\right\}h(t) + f[h(t),I(t);\theta]A, \quad (6)$$

where, A is a bias vector and f is the modeled continuous-time recurrent neural network. A more general CFC network is then derived as Eq. (7):

$$h(t) = \sigma\left[-t\cdot f\left(h(t),I(t);\theta_f\right)\right]g\left(h(t),I(t);\theta_g\right) + \left\{1-\sigma\left[-t\cdot f\left(h(t),I(t);\theta_f\right)\right]\right\}r(h(t),I(t);\theta_r), \quad (7)$$

where, f, g and r are neural networks to be learnt; σ is for the gating mechanism mentioned in Ref. [11].

1.2 Misbehavior detection

To secure the IoV, cryptographic methodologies such as the public key infrastructure (PKI) [12] can defend against the external attackers while the internal attackers are mainly thwarted by misbehavior detection mechanisms. Faulty nodes and malicious nodes are two typical misbehavior entities in the vehicular network, where a faulty node malfunctions usually due to its broken sensors and a malicious node generates erroneous messages to mislead its neighbors. In this subsection, we will survey works applying deep learning models on both faulty and malicious misbehavior detection.

Alladi et al. [13] proposed an artificial intelligence-based intrusion detection architecture and deployed it on the multi-access edge computing (MEC) servers. Raw data are firstly fed into a sequence generating preprocessing engine to form the time series. The sequences are later run through either the time-sequence classification deep learning engine (DLE) or the image generation engine followed by the sequence-image classification DLE. Considering the attacks Denial-of-Service (DoS), Data replay, Disruptive and Sybil, the best results were from the sequence-image classification DLE using Convolutional Neural Network (CNN). Immediately after the work got published, Alladi et al. [14] came up with another model that considered both the attacks and the faulty types.

In Ref. [14], only normal data were trained on the neural networks, i.e., the CNN-LSTM and the stacked-LSTM, which implies that any divergence from the normal sequence will be classified as misbehavior. The author started with collecting position and speed coordinates from normal vehicles. The position-speed sequences later went through a sequence reconstruction pipeline to develop into genuine position sequences. Next, the loss function, the Mean Absolute Error (MAE) of each input position sequence and its reconstructed position sequence was used for backpropagation. Finally, a threshold, which was used as an indicator whether the sequence would be classified as anomalous or not, was calculated as specified in the thresholding algorithm. Results show that the one-dimensional convolutional layer (with 20 filters) connected to a 4-layer LSTM (with 256 units for each layer) outperforms the stacked-LSTM of equivalent sizes.

Further at the end of the year 2021, Alladi et al. [15] put forward a comprehensive deep neural network (DNN) framework taking into account the combinations of the following schemes: CNN, LSTM, GRU and attention mechanisms. The models were trained again only on the genuine data sequences and multiple scenarios (fault only scenario, attack only scenario, and mixed scenario) were tested against the sequence reconstruction unit. The thresholding algorithm was also utilized in the framework. Evaluations metrics were computed for each of the 6 models and the CNN-LSTM model turned out to be the best.

In 2022, Alladi et al. [16] submitted a fourth work on the deep learning-based misbehavior classification scheme to the Digital Communications and Networks. They introduced an idea of multi-stage classification, where data were firstly fed to a coarse-grained classifier and then followed by a fine-grained one. 3-LSTM, 4-LSTM, 5-LSTM, 1-CNN-1-LSTM, 2-CNN-1-LSTM and 2-CNN-2-LSTM were taken for performance evaluations of the 3 DLCEs and results show that the single-stage classification wins.

2 Experimental

2.1 Models

The network architecture is given in Fig. 1, where Vehicle-to-Vehicle (V2V) connection is adopted for beacon message broadcasting and Vehicle-to-Infrastructure (V2I) communication is used for misbehavior detection. Thus, the misbehavior classifier will be deployed on the RSUs, collecting beacons from the vehicular

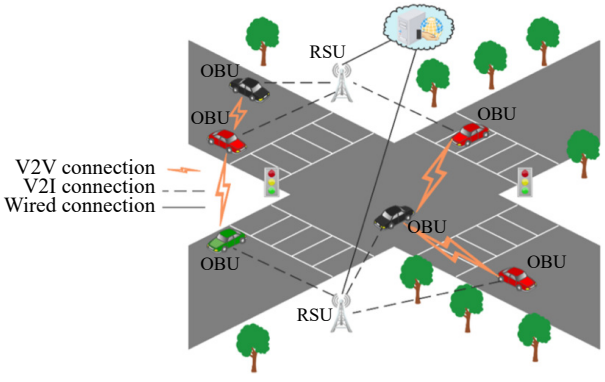

Fig. 1 System model.

OBUs.

In this paper, we consider 19 misbehaviors [17] launched at the OBUs in the system model, where each of the misbehavior is listed with its type and a concise description:

Type 1, Constant Position: The position is fixed.

Type 2, Constant Position Offset: A constant offset is added to the real position.

Type 3, Random Position: The position is random.

Type 4, Random Position Offset: A random offset is added to the real position.

Type 5, Constant Speed: The speed is fixed.

Type 6, Constant Speed Offset: A constant offset is added to the real speed.

Type 7, Random Speed: The speed is random.

Type 8, Random Speed Offset: A random offset is added to the real speed.

Type 9, Eventual Stop: A sudden stop with the position becoming constant and the speed and acceleration set to zero.

Type 10, Disruptive: Retransmit messages received from neighbors choosing randomly, with the attacker's pseudonym.

Type 11, Data Replay: Retransmit messages from a specific neighbor, with the attacker's pseudonym

Type 12, Delayed Messages: The real messages are delayed sending for Δt.

Type 13, DoS: The frequency of messages sent surpasses the limit in the standards.

Type 14, DoS Random: A DoS attack with message contents all set to random values.

Type 15, DoS Disruptive: A DoS attack retransmitting messages received from random neighbors.

Type 16, Grid Sybil: Creating a grid of fake vehicles with distinct pseudonyms in 1s.

Type 17, Data Replay Sybil: A Data Replay attack with a new pseudonym for each chosen neighbor.

Type 18, DoS Random Sybil: A DoS Random attack with each message sent with a different pseudonym.

Type 19, DoS Disruptive Sybil: A DoS Disruptive attack with each message sent with a different pseudonym.

2.2 *Framework*

Before diving deep into the test framework, we will first elaborate on the feature extraction processes. The features in work [18] done by Joseph et al. relied mostly on the local detectors which check for the plausibility and consistency of the raw data. Besides, other features describing the behavior of each vehicle are available in [19], for instance, the difference between the calculated average velocity and the predicted one, etc. In this paper, we extract 7 features from VeReMi extension and the features can be broken down into three categories. The first one is the distance between the sender and the receiver, implemented as the difference between the X-Y coordinates of the sender's position and the receiver's position [20], given by Eqs. (8) and (9):

$$d_x = \left| p_{s_x} - p_{r_x} \right|, \tag{8}$$

$$d_y = \left| p_{s_y} - p_{r_y} \right|, \tag{9}$$

where p is the position field in the beacon and the subscript x and y are the coordinates of the positions of the sender and the receiver. The 2nd feature is the counting feature, which deals with the frequencies of message transmissions. The last feature is the behavioral dynamics with reference to [19] that an absolute value of the difference between the calculated velocity/acceleration and the averaged velocity/acceleration is obtained, given as Eqs. (10) to (13):

$$\Delta v_x = \left| \frac{p_{t_x} - p_{t-1_x}}{1} - \frac{s_{t_x} + s_{t-1_x}}{2} \right|, \tag{10}$$

$$\Delta v_y = \left| \frac{p_{t_y} - p_{t-1_y}}{1} - \frac{s_{t_y} + s_{t-1_y}}{2} \right|, \tag{11}$$

$$\Delta a_x = \left| \frac{s_{t_x} - s_{t-1_x}}{1} - \frac{a_{t_x} + a_{t-1_x}}{2} \right|, \tag{12}$$

$$\Delta a_y = \left| \frac{s_{t_y} - s_{t-1_y}}{1} - \frac{a_{t_y} + a_{t-1_y}}{2} \right|, \tag{13}$$

where s is the speed field and a is the acceleration field in the beacon.

As shown in Fig. 2, six components constitute the test framework. In the first step, a hybrid data of normal, faulty and attack behaviors is preprocessed into time sequences per vehicle, where each vehicular sequence is "reflectedly" padded to be an integral multiple of a hundred and each consecutive pair of sequences is offset by 10. Next, finer-grained sub-sequences of 10 by 7 are constructed for each vehicular sequence, where 10 timepoints are identified for each sub-sequence with a 7-dimensional feature vector spanning all the features mentioned above. Apart from the vehicular sequences, the sub-sequences maintain recurrent relations in the time dimension and each consecutive pair of sub-sequence is offset by 5. Thirdly, two continuous-time RNN models are utilized for experimentation, i.e., the liquid time-constant network (model 1) and the closed-form continuous-time neural network (model 2). Finally, a fully-connected feed-forward neural network is responsible for classifying the data, followed by an output layer.

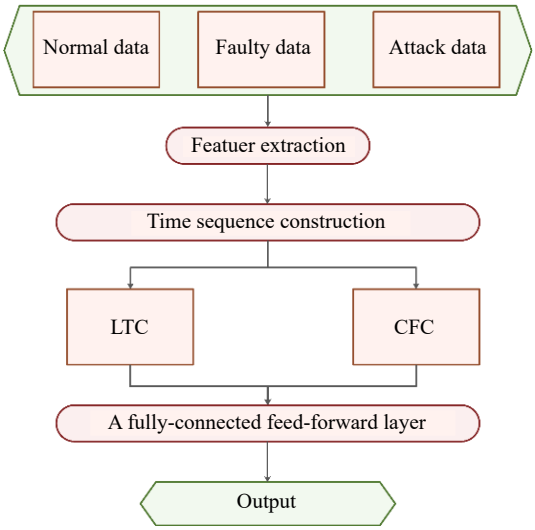

Fig. 2 Test framework.

2.3 *Datasets, settings and metrics*

VeReMi extension dataset [17] has been a public dataset available for numerous works [21-24] to reproduce results and validate effectiveness of their proposals. The dataset was extended by Kamel et al. from the original dataset [25] with three simulations, i.e., the high-density traffic (from 7 to 9 in the morning), the low-density traffic (from 2 to 4 in the afternoon) and the whole-day testbench (from 0 to 24), and 19 misbehaviors shown in the attack model. In each simulation, there are ground-truth files, which are the true messages to be sent as if all the vehicles were normal behaving, and trace-log files that record the messages received by each vehicle in the network with faulty and attack behaviors. In this study, we are mainly concerned with the first two simulations, i.e., the high-density and the low-density simulations and with fields including receive time, sender pseudonym, and the *X*-*Y* coordinates of position, speed and acceleration in the vehicular trace-log files.

Supported by NVIDIA GeForce RTX 3 090 and 25G memory, our test framework is deployed with Pytorch 1.13.0, Python 3.9.13

and Cuda 11.7. In both model 1 and model 2, an NCP wiring of 10 inter-neurons, 10 command neurons and 8 motor neurons is selected as the hidden units [10]. For each of the 19 misbehaviors, a thread is initialized with a set of parameters containing the model, the optimizer, the loss function, the scheduler, the training and inference data, etc. A complete list of parameters is given in Table 1.

Table 1 Propellant masses vs. transfer time using orbital averaging and guidance scheme.

Attributes	Model 1	Attributes	Model 2
Training set: Test set	80%:20%	Training set: Test set	80%:20%
Epoch	10	Epoch	10
Learning rate	0.01	Learning rate	0.01
Optimizer (weigh decay)	AdamW (0.001)	Optimizer (weigh decay)	AdamW (0.001)
Scheduler (gamma)	Exponential (0.6)	Scheduler (gamma)	Exponential (0.6)
Loss function	Cross entropy loss	Loss function	Cross entropy loss
Batch size (low density: high density)	64:128	Batch size (low density: high density)	64:128
Vehicular sequence length	100	Vehicular sequence length	100
Sub-sequence length	10	Sub-sequqence length	10
Ode folder number	6	Activation function	Relu

Before introducing the performance evaluation metrics, the confusion matrix shall be visited, given by Table 2.

Table 2 Confusion matrix.

Labels	Misbehavior	Normal
Predicted Misbehavior	True Positive (P_T)	False Positive (P_F)
Predicted Normal	False Negative (N_F)	True Negative (N_T)

The confusion matrix draws the connection between real labels and their predicted ones. Based on the confusion matrix, accuracy (A), precision(P), recall (R) and f1 score (F_1) can be defined as Eqs. (14) to (17):

$$A = \frac{P_T + N_T}{P_T + P_F + N_F + N_T}, \tag{14}$$

$$P = \frac{P_T}{P_T + P_F}, \tag{15}$$

$$R = \frac{P_T}{P_T + N_F}, \tag{16}$$

$$F_1 = \frac{2PR}{P + R}. \tag{17}$$

3 Results and Discussion

3.1 Performance comparison

For comparison purposes, the results of the CNN-LSTM model, which is a CNN (20 filters for each convolution layer) concatenated by a bidirectional LSTM (each hidden layer being of 256 units), will be displayed as a third option (model 3) [15]. In Table 3, the averages of accuracy, precision, recall and f1 score are tabulated for model 1 and model 2, in contrast to those of model 3 [15].

Table 3 Performance comparisons.

Models	Accuracy	Precision	Recall	F1 score
Model 1	0.992	0.997	0.992	0.994
Model 2	0.996	0.998	0.996	0.997
Model 3 [15]	0.980	0.996	0.956	0.976

The table indicates a rank of Model 2, Model 1 followed by Model 3 according to all the metrics defined in the previous section, which favors the continuous-time neural network over the CNN-LSTM.

3.2 Computation time

Fig. 3 renders the classification time cost of each model over 19 misbehaviors, with the x-coordinates being different types of misbehavior and the y-axis being the average time cost per prediction.

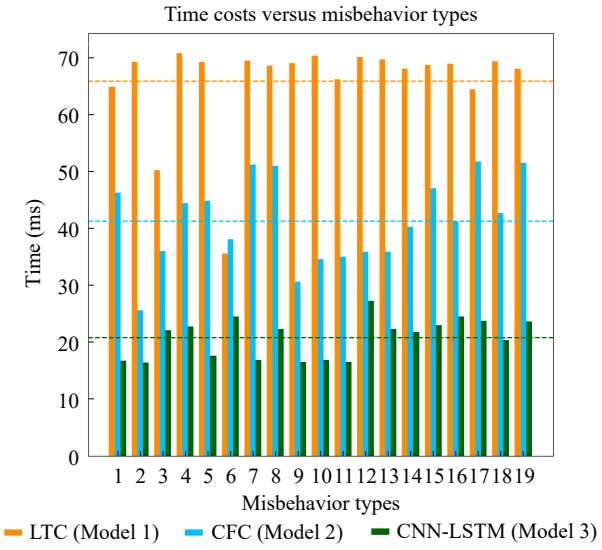

Fig. 3 Time cost.

Fig. 3 demonstrates a superiority of CNN-LSTM, with a runtime average of 21 ms, over LTC and CFC, with runtime averages of 66 ms and 41 ms respectively. The reason for the time cost of LTC being triple that of CNN-LSTM is that LTC depends heavily on its time-consuming operator, the ODE solver, which computes the state transition of the system by Euler approximation. Meanwhile, CFC doubles the runtime cost comparing to that of CNN-LSTM because it expands the state transition integral into nonlinear operators using Taylor expansion. Either the Euler approximation in LTC or the Taylor expansion in CFC takes more time than the matrix additions and multiplications in CNN-LSMT. Nonetheless, the drawbacks of the continuous-time neural networks are balanced out in the following subsection which analyzes the model complexity in terms of the number of parameters (or space-complexity).

3.3 Model complexity

Table 4 illustrates the profiling including input and output size, hidden layer size and network parameters for each of the models. The input size and output size are determined by the task and its requirements, which is of the form "(batch size in low-density traffic/batch size in high-density traffic, time points, dimension of each time point)". The hidden units are pertaining to the recurrent neural network used and the network parameters describe the number of arguments of the overall architecture.

Table 4 Profiling.

Models	Input size	Output size	Hidden units	Network Parameters
Model 1	(64/128,10,7)	(64/128,10,2)	28	5 032
Model 2	(64/128,10,7)	(64/128,10,2)	28	2 700
Model 3 [15]	(1 800,20,7)	(1 800,20,2)	(20+256)*3	1 697 760

From Table 4, it is not difficult to find that the continuous-time neural network (Model 1 and 2) is of a much simpler network with very few hidden units and network parameters, which is a representative characteristic for misbehavior detection schemes to be deployed into the Internet of vehicles, comparing to the CNN-LSTM.

4 Conclusion

In this work, we systematically applied two recently proposed

continuous-time recurrent neural networks to misbehavior detection in Internet of Vehicles. Results demonstrate that, at a negligible sacrifice of runtime cost, the continuous-time neural networks with remarkably fewer parameters perform slightly better than the most popular scheme composed of CNN and LSTM, measured by Accuracy, Precision, Recall and F1 score. Furthermore, one will most likely choose CFC (model 2) among three models due to a balance struck between the computation time and the model complexity. Nevertheless, this work might overlook presumptions in practice, for instance, the privacy issues of beacon messages etc., which we plan to tackle in the upcoming work. In a nutshell, we hope this work would shed light on the application of continuous models into realities.

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

Data Availability Statement

The VeReMi extension dataset can be found at https://github.com/josephkamel/VeReMi-Dataset.

Acknowledgements

This work is supported in part by the National Natural Science Foundation of China under grant 61 871 193, in part by the Guangzhou Key Field R&D Program under grant 202 206 030 005 and in part by the R&D Program of key science and technology fields in Guangdong province under grant 2019B090912001.

References

[1] Chen SZ, Hu JL, Shi Y, et al. A vision of C-V2X: Technologies, field testing, and challenges with chinese development. IEEE Internet of Things Journal, 2020, 7(5): 3872−3881.10.1109/JIOT.2020.2974823.

[2] Boualouache A, Engel T. A survey on machine learning-based misbehavior detection systems for 5g and beyond vehicular networks. IEEE Communications Surveys & Tutorials, 2023, 25(2): 1128−1172. 10.1109/COMST.2023.3236448.

[3] Le A, Maple C. Shadows don't lie: N-sequence trajectory inspection for misbehaviour detection and classification in VANETs. 2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall). Honolulu. IEEE, 2019: 1−6.

[4] Hsu HY, Cheng NH, Tsai CW. A deep learning-based integrated algorithm for misbehavior detection system in VANETs. Proceedings of the 2021 ACM International Conference on Intelligent Computing and Its Emerging Applications. Jinan China. ACM, 2021: 53−58.

[5] GoyaL A, Bhatia A, Yadav A, et al. Misbehavior detection in cooperative intelligent transportation systems using temporal fusion transformer. Proceedings of the 24th International Conference on Distributed Computing and Networking. Kharagpur, India. ACM, 2023: 431−437.

[6] He KM, Zhang XY, Ren SQ, et al. Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas. IEEE, 2016: 770−778.

[7] Chen RTQ, Rubanova Y, Bettencourt J, et al. Neural ordinary differential equations. Proceedings of the 32nd International Conference on Neural Information Processing Systems. Montréal, Canada. ACM, 2018: 6572−6583.

[8] Wang X, Blum EK. Discrete-time versus continuous-time models of neural networks. Journal of Computer and System Sciences, 1992, 45(1): 1−19.

[9] Hasani RM, Lechner M, Amini A, et al. Liquid time-constant recurrent neural networks as universal approximators. arXiv preprint arXiv: 1811.00321, 2018. https://doi.org/10.48550/arxiv.1811.00321.

[10] Lechner M, Hasani R, Amini A, et al. Neural circuit policies enabling auditable autonomy. Nature Machine Intelligence, 2020, 2(10): 642−652.

[11] Hasani R, Lechner M, Amini A, et al. Closed-form continuous-time neural networks. Nature Machine Intelligence, 2022, 4(11): 992−1003.

[12] Diffie W, Hellman M. New directions in cryptography. IEEE Transactions on Information Theory, 1976, 22(6): 644−654. https://doi.org/10.1109/TIT.1976.1055638.

[13] Alladi T, Kohli V, Chamola V, et al. Artificial intelligence (AI)-empowered intrusion detection architecture for the internet of vehicles [J]. IEEE Wireless Communications, 2021, 28(3): 144−149.10.1109/MWC.001.2000428.

[14] Alladi T, Agrawal A, Gera B, et al. Deep neural networks for securing IoT enabled vehicular Ad-Hoc networks. ICC 2021 - IEEE International Conference on Communications. Montreal. IEEE, 2021: 1−6.

[15] Alladi T, Gera B, Agrawal A, et al. DeepADV: a deep neural network framework for anomaly detection in VANETs. IEEE Transactions on Vehicular Technology, 2021, 70(11): 12013−12023. 10.1109/TVT.2021.3113807

[16] Alladi T, Kohli V, Chamola V, et al. A deep learning based misbehavior classification scheme for intrusion detection in cooperative intelligent transportation systems[J]. Digital Communications and Networks, 2023, 9(5): 1113–1122.

[17] Kamel J, Wolf M, Van der Hei RW, et al. VeReMi Extension: a dataset for comparable evaluation of misbehavior detection in VANETs. ICC 2020 - 2020 IEEE International Conference on Communications (ICC). Dublin. IEEE, 2020: 1−6.

[18] Kamel J, Haidar F, Ben Jemaa I, et al. A misbehavior authority system for sybil attack detection in C-ITS. 2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON). New York. IEEE, 2019: 1117−1123.

[19] So S, Sharma P, Petit J. Integrating plausibility checks and machine learning for misbehavior detection in VANET. 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA). Orlando. IEEE, 2018: 564−571.

[20] Ercan S, Ayaida M, Messai N. New features for position falsification detection in VANETs using machine learning . ICC 2021 - IEEE International Conference on Communications. Montreal. IEEE, 2021: 1−6.

[21] Sedar R, Kalalas C, Dini P, et al. Misbehavior detection in vehicular networks: an ensemble learning approach. GLOBECOM 2022 - 2022 IEEE Global Communications Conference. Rio de Janeiro. IEEE, 2022: 1850−1855.

[22] Sedar R, Kalalas C, Vázquez-gallego F, et al. Reinforcement learning based misbehavior detection in vehicular networks. ICC 2022 - IEEE International Conference on Communications. Seoul. IEEE, 2022: 3550−3555.

[23] Kohli V, Chougule A, Chamola V, et al. MbRE IDS: an AI and edge computing empowered framework for securing intelligent transportation systems. IEEE INFOCOM 2022 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). New York. IEEE, 2022: 1−6.

[24] Kushardianto NC, El Hillali Y, Tatkeu C. 2−step prediction for detecting attacker in vehicle to vehicle communication. 2021 IEEE 94th Vehicular Technology Conference (VTC2021−Fall). Norman. IEEE, 2021: 1−5.

[25] Van der Heijden RW, Lukaseder T, Kargl F. VeReMi: a dataset for comparable evaluation of misbehavior detection in VANETs. arXiv preprint arXiv: 1804.06701, 2018. https://doi.org/10.48550/arxiv.1804.06701.