Mobile Edge Assisted Literal Multi-Dimensional Anomaly Detection of In-Vehicle Network Using LSTM

Konglin Zhu, Zhicheng Chen, Yuyang Peng, and Lin Zhang

***Abstract*—The development of Internet of Vehicles (IoVs) have introduced more intrusions to the vulnerabilities of in-vehicle network. It is important to detect in-vehicle network anomaly for the purpose of driving safety. The previous studies either focus on one-dimension anomaly behavior of in-vehicle network or looks into the semantics of in-vehicle network, which is neither effective nor efficient for vehicle pilot. In this paper, we propose a literal multi-dimensional anomaly detection approach using the distributed long-short-term-memory (LSTM) framework for in-vehicle network, especially the Control Area Network (CAN). The proposed approach only needs the literal binary CAN message instead of revealing the semantics of CAN message. To enhance the accuracy and efficiency of detection, it detects anomaly from both time and data dimension simultaneously by exploiting multi-task LSTM neural network on mobile edge. The extensive evaluation results show that the proposed anomaly detection achieves a satisfying accuracy of 90%. The detection speed is as fast as 0.61 ms on mobile edge.**

***Index Terms*—CAN, anomaly detection, LSTM, multi-task, edge computing.**

I. Introduction

With the development of Internet of Vehicles (IoVs), people are increasingly demanding information exchange between automobiles and devices by connected vehicles. The in-vehicle network not only is connected to smart devices such as mobile phones, but also share information with the traffic management system and other vehicles passing by in the form of vehicle to everything (V2X). Indeed, all driving actions are operated by Electronic Control Units (ECU) [1] which are connected by buses composing the in-vehicle network for the vehicle control. Such in-vehicle network manifests intrinsic security vulnerabilities the same as other kinds of networks. Attackers can hack the in-vehicle network from outside the vehicle by networking connections. Especially in the case where an automobile is connected to the cloud through the communication network, each computing unit, controlling unit, sensing unit, and connection path are vulnerable for hackers to attack or even control the automobile. As an important part of the public transportation system, automobiles intruded by hackers will not only cause the leakage of drivers' personal information and privacy, but also incur direct damage to personal security and property safety.

Many previous studies focus on the anomaly detection of vehicle intelligence, such as anomaly detection on image recognition [2]–[4] and speech language identification [5], [6]. Besides, the IoV circumstances render the in-vehicle network vulnerable to be hacked by malicious parties [7], which drives researchers to work on anomaly detection for in-vehicle network. In a vehicle, Controller Area Network (CAN) bus is the most frequently used in-vehicle network, and the majority of attacks is introduced to the in-vehicle network via CAN bus. Therefore, detecting malicious behaviors in CAN bus is the bottom line of defense [8]. Hackers may send abnormal commands via the CAN bus to disturb or control the vehicle. Previous research put much effort on CAN bus anomaly detection. For instance, Song *et al.* [9] put forward a compact detection model based on the time intervals between CAN messages, Cho *et al.* [10] chose the clock offset of timestamp in in-vehicle messages as the detected feature and used clock skew to authenticate ECUs, Kang *et al.* [11] used compact Deep Neural Networks (DNN) to classify CAN message packages, Taylor *et al.* [8] developed a neural network with Long Short-Term Memory (LSTM) [12] layer by using the regularity of binary data of CAN bus to predict the future messages and judge anomaly. However, these methods either only detect periodic CAN bus messages [9], [10], or merely detect limited types of anomaly [8], [11]. Notwithstanding, some further studies exploit machine learning to parse the semantics of the CAN bus for CAN bus anomaly detection, and the researchers intending to apply such a progress usually has to obtain the permission of vehicle manufacturers [13]. Whereas, the semantics of the CAN bus in different vehicles are parsed by different means the majority of vehicle manufacturers are not willing to open the

Manuscript received September 30, 2018; revised January 9, 2019 and March 6, 2019; accepted March 18, 2019. Date of publication March 25, 2019; date of current version May 28, 2019. This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB0100902, in part by the National Natural Science Foundation of China under Grant 61502045, and in part by the Fundamental Research Funds for the Central Universities. The review of this paper was coordinated by the Guest Editors of Special Section on Machine Learning-Based Internet of Vehicles. *(Corresponding author: Konglin Zhu.)*

K. Zhu is with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China, and Purple Mountain Laboratories, Nanjing 210008, China (e-mail: klzhu@bupt.edu.cn).

Z. Chen and L. Zhang are with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: chenzhicheng@bupt.edu.cn; zhanglin@bupt.edu.cn).

Y. Peng is with the Faculty of Information Technology, Macau University of Science and Technology, Macau, China (e-mail: yypeng@must.edu.mo).

Digital Object Identifier 10.1109/TVT.2019.2907269

protocol out of security reasons. Nevertheless, there is a need for a compact and comprehensive anomaly detection for CAN bus to enhance the driving safety of vehicles under the IoV environment.

To overcome the limits of anomaly CAN message detection which either focuses on one dimension features of CAN messages or investigate the semantics of CAN message, we propose a literal multi-dimensional anomaly detection for CAN using a distributed deep learning framework. We enrich the CAN message features from both the dimension of time and the dimension of data so that more features from anomaly CAN message can be discovered. To detect the anomaly in CAN messages, we exploit a LSTM neural network from both the dimension of time and the dimension of data without parsing the semantics of CAN messages. Since LSTM is a neural networking framework for time-serial events prediction [14]. It can be well adapted to anomaly detection without utilizing the semantics of CAN messages. Furthermore, the anomaly detection makes parallel computing possible both in the local end and the mobile edge using multi-task LSTM, thus effectively reducing the computation limitations of in-vehicle mobile devices and enhancing the efficiency of anomaly detection significantly. The evaluation results show that the proposed anomaly detection methods reach 90% of accuracy, and the detection can be finished on the mobile edge within 0.61 milliseconds.

The contributions of the paper are summarized as follows:

- LSTM neural network is exploited to detection anomaly of CAN messages from both the dimension of time and the dimension of data without parsing the semantics of CAN messages. As a result, the multi-dimensional anomaly detection enhances the accuracy of detection.
- The multi-task LSTM framework is used for anomaly detection with the assistance of the mobile edge, thus surpassing the limits of computing capacity of on-board vehicle devices. Via a V2X connection to the mobile edge, it detects anomaly of CAN bus for a crowd of vehicles and shortens the computation time used by on-board vehicle devices to milliseconds.
- The extensive evaluation results show that the proposed method detects anomaly with the accuracy of 90%. The detection can be accomplished in 0.61 milliseconds.

The remainder of the paper is organized as follows. Section II reviews the related studies on CAN bus anomaly detection. Section III discusses the preliminaries of the work. Section IV illustrates the basic approach based on LSTM neural network used for CAN bus anomaly detection. Section V discusses the mobile edge assisted multi-task LSTM iin both the dimension of time and the dimension of data. Section VI shows the performance of the proposed anomaly detection method. The paper is concluded in Section VII.

II. Related Works

Previous research on in-vehicle CAN bus network anomaly detection schemes mainly based on features of CAN messages [15]. They firstly analyze the characteristics of normal CAN bus data in a certain aspect, and then to examine the extent to which the abnormal data does not conform to the analyzed characteristics. Based on the feature they used for detection, these schemes are divided into three categories: time-interval based anomaly detection, data based anomaly detection and semantics based anomaly detection.

Time-interval based anomaly detection schemes [9], [10], [16] exploit the periodicity of the CAN messages to find the abnormal CAN that does not follows the periodicity. Song *et al.* [9] put forward a compact detection model based on the analysis of time intervals. Cho *et al.* [10] chose the clock offset of the timestamp of in-vehicle messages as the detected future and used clock skew to authenticate ECUs. Taylor *et al.* [16] proposed an anomaly detection algorithm with a sliding window to measure interval time. Michael *et al.* [17] provided a data-driven anomaly detection algorithm relays on regularly and redundantly. All these schemes are based on time information and rely on the periodicity of CAN messages. Ji *et al.* [18] takes into clock drift into consideration for anomaly detection to find time patterns in CAN bus messages. However, aperiodic messages in CAN bus are also very common which cannot be disposed by these schemes. To overcome this issue, Marchetti *et al.* [19] built a model of the normal behavior of CAN bus based on a particular feature of CAN ID sequence. Muter *et al.* [20] used packet message entropy related to CAN ID sequence to detect insertion attacks. Indeed, ID sequence is related to time interval among message. Therefore, these two schemes still used time-interval feature for anomaly detection for CAN bus.

Data based anomaly detection schemes [8] uses the regularity of binary data of CAN bus to predict the next bits. If the CAN bus message is the same as the predicted one, it is considered as normal message. Otherwise, it is considered as an anomaly. A neural network with Long Short-Term Memory (LSTM) layer is developed to predict the future message and judge anomaly in [8]. It used LSTM to predict the possible value of next bits and compare with the CAN bus message to determine the correctness of CAN bus information. This scheme has good performance on those CAN message that is predictable. However, it does not have good performance on time-interval featured CAN message as they are normally not predictable.

Several recent studies started to dig into the semantics of CAN bus messages and attempted to find some features from semantics for anomaly detection. Kang and Li also used machine learning in CAN bus anomaly detection [11], [21]. Li *et al.* [21] employed a classification deep neural network using GPS speed, wheel speed *et al.* as the input data. Kang *et al.* [11] used filtered data in a package of CAN messages to train a classification deep neural network. However, both schemes need the protocol of CAN bus to parse the semantics information, whereas these protocols vary from different brand of vehicles and majority of vehicle manufacturers do not open the protocol of CAN bus for the information security.

Our work differs from the previous studies by incorporating both time-interval feature and data feature by multi-task LSTM neural network, which allows the distributed framework on the mobile edge to enhance the computation speed. The proposed scheme does not need to parse the semantics of CAN bus thus does not need to corporate with vehicle manufacturer tightly.

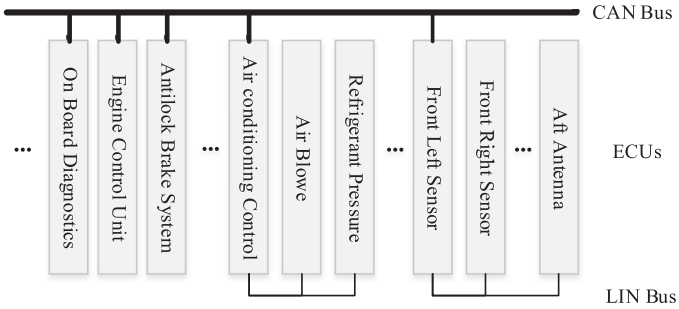


Fig. 1. Basic elements of In-Vehicle Network.

SOF (1 bit)	ID (11 bit)	RTR (1 bit)	IDE (1 bit)	RB0 (1 bit)	DLC (4 bit)	Data (64 bit)	CRC (16 bit)	ACK (2 bit)	EOF (7 bit)

Fig. 2. Structure of CAN bus message.

III. Preliminaries

In this section, we first introduce the in-vehicle network and CAN bus message. Then we discuss the attack model to CAN bus message.

A. In-Vehicle Network Data

The most common in-vehicle network is composed by ECU, CAN bus, Local Interconnect Network (LIN) bus, and FlexRay bus. ECU is the control unit that takes in charge of driving actions, which is inter-connected by these buses for data communication in vehicle. Fig. 1 shows the basic elements of in-vehicle network. CAN bus is the main network connections connecting different kinds of ECUs. LIN bus is used as a messenger to connect CAN bus to ECU. Each ECU is identified by a unique ID. The in-vehicle message is sent in the form of in-vehicle broadcast by ECU through CAN bus. When the corresponding vehicle component received the message, it parsed the message and conduct the corresponding vehicle control command. In contrast, FlexRay bus has relatively high cost. It has larger transfer rate and higher security than CAN bus but has not been widely used. Until now, most common buses used for in-vehicle network is still CAN bus. Therefore, we mainly study the anomaly detection for CAN bus.

A common structure of CAN bus message is shown in Fig. 2. The message is composed by 108 bits, containing ECU ID, timestamp, status of vehicle, data verification, data entity and etc. The most important two parts of this structure are the ID and Data entity. The ID part represents the ECU who sent this message, while the Data part comprises information from this ECU. There are 64 binary digits in the Data part, and how the 64 bits are encoded is determined by automobile manufacturers. Therefore, although many anomaly detection methods exploit the semantics of CAN bus messages provided by automobile manufacturers, it lacks the generality as different automobile manufacturers encode the Data in different manners. To enhance the generality of the anomaly detection methods, we only get the information such as ECU ID, timestamp, the Data is in a binary form from a CAN bus message. The anomaly detection is to find the abnormal behavior on ID, timestamp, Data and etc.

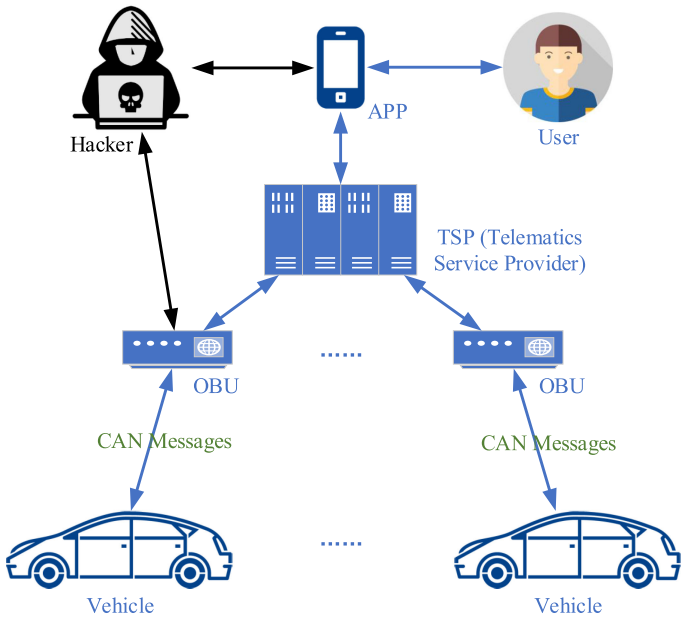


Fig. 3. Attack to Internet of Vehicles.

B. Attack Model

IoVs communicates with V2X applications by an embedded on-board unit (OBU). The OBU also connect to the CAN bus to enable the communication between in-vehicle network and outside network. As connecting to the outside network, it communicates with the cloud or infrastructure for remote driving service, and communicates with other vehicles by vehicle-to-vehicle (V2V) interface. Among all these services and applications, the hacker can attack the in-vehicle network either directly from OBU or from applications that connected to OBU. A typical attack process to in-vehicle network is illustrated in Fig. 3. The typical attack models to CAN bus including spoof, replay, flood, drop and tamper [17], [22]. The detail of these attack models are illustrated as follows:

Spoof: The attacker sends forged information to interfere with the normal in-vehicle communication.

Replay: The attacker intercepts and captures the data segment, and then re-send the captured data into network in a future time point, trying to reproduce the corresponding function, so as to interfere with normal driving.

Flood: The attacker sends a large amount of random information within a short time, leading to dysfunction of the in-vehicle network.

Drop: The attacker controls the ECU or triggering communication isolation, so as to paralyze the sending of normal information.

Tamper: The attacker tampers with the message content, so as to interfere with the normal function.

All these attack models will lead to the change of CAN bus message. For instance, the spoof attack sends additional CAN bus message to the in-vehicle network, leading to the shorten

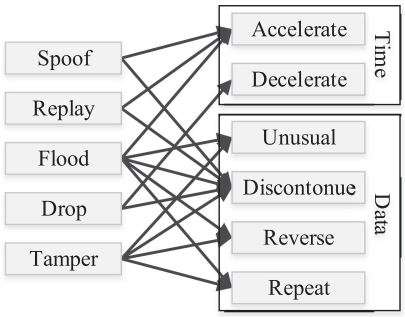


Fig. 4. Relationships between attacks and abnormal data patterns.

of time interval between pairwise of CAN bus message; similar attacks leading to the shorten of time interval also include replay and flood. In contrast, drop attack results in the delay of time interval between pairwise of CAN bus message as it may block and drop certain CAN bus message. Furthermore, flood and tamper has ability to generate different patterns of miscellaneous message. Overall, these attacks result in a series of abnormal data patterns. We show six typical abnormal patterns as follows: (1) accelerate: the time interval before and after the message becomes shorter, which may occur by the addition of messages or the flapping of messages; (2) decelerate: the time interval before and after the message becomes longer, which may occurs by the drop of messages or the flapping of messages; (3) unusual: some fixed bits (always 1 or 0) of message is changed; (4) discontinue: includes both the loss of intermediate message and the addition of message; (5) reverse: the data information is reversed in a chronological order; (6) repeat: repeated data information. The relationship between these 6 types of abnormal data patterns and 5 kinds of attacks is shown in Fig. 4. It shows that one sort of attacks usually leads to several abnormal data patterns on CAN bus message.

IV. Anomaly Detection Using LSTM

In this section, we discuss the proposed approach for anomaly detection using LSTM combined time dimension and data dimension simultaneously.

A. Basic Approach

As CAN bus message appears in a series of bits with patterns, similar as image pattern abstraction [23], [24], it allows us to find the specific patterns from different perspectives of the message. For a CAN bus message, the most salient pattern features are data and time. In time perspective, the time interval between two CAN messages is relevant to the periodicity of CAN messages, so the periodicity is often used as features. In data perspective, 64-bit Data in CAN bus messages always has certain features that allow the classification of CAN bus messages. In spite that both two types of features can be used for anomaly detection, each of them only takes effect on the portion of messages owning the feature. To enhance the accuracy of anomaly detection, we focus on the scheme using both data and time interval feature to explore CAN bus message behavior.

Among the prediction and anomaly detection on series of data, LSTM [25] neural network is considered as the one of the powerful tool to find the patterns of CAN bus message. LSTM shows good performance on time sequence data with the structure of cumulative memory, and meanwhile the CAN bus messages are exactly time sequence data that changed by driving conditions and driver behavior. Such consistency allows us to use LSTM on CAN bus message anomaly detection. Due to different characteristics of Data and time interval, we exploit a multi-dimension LSTM framework for anomaly detection. As shown in Fig. 5, the CAN bus message containing Data and time interval features come to the LSTM framework. The framework contains a prediction process and a detection process. The prediction process is to predict the coming CAN bus message by investigating the feature and pattern of CAN bus Data and time interval. The detection is to decide whether arrived message is anomaly or not. In the following, we will explain how the features is combined to contain both Data and time interval feature and how to construct the LSTM neural network for anomaly detection.

B. Combination of CAN Features

As mentioned-above, each CAN bus message is in a form of 108 bits, and two CAN bus messages comes within milliseconds. Therefore, the feature of CAN bus message is presented by the feature of bit-stream and the feature of time interval between a pairwise of messages.

In the 108-bit CAN bus message, the 64-bit Data is with most significant feature. Different groups of CAN bus messages have various Data features. To show the feature of Data, we use a vector $D \in \mathbb{R}^{64}$, written as:

$$D = \{d_1, d_2, \ldots, d_{64}\}, \tag{1}$$

where d_i represents each bit in data part. The anomaly occurs if unexpected bits or bit patterns are detected. In time aspect, the time interval between two CAN bus message varies from several milliseconds to hundreds of milliseconds depending on different kinds of CAN bus messages. Certain types of CAN bus message has certain time interval, so anomaly occurs if unexpected time interval appears. Mathematically, the time interval ΔT is an integer presenting the number of milliseconds between a pairwise of CAN bus messages.

As Data and time interval are from different dimension of message, the naive approach is using two separated LSTM neural networks to predict the corresponding value in data and time dimension respectively. However, such an operation will result in the higher computation complexity. In order to enhance the efficiency of computation, we combine the Data and time interval feature into one dimension so that one LSTM neural network can be sufficient. However, the direct combination of data and time results in the expand of vectors to a huge limit range with low performance. To reduce the complexity of combined data and time and make it fit for the LSTM neural network, we conduct the normalization to time interval to [0,1]. Then the Data D which is a vector of binary bits with normalized time interval T constitutes the combined feature from a CAN message, as shown in the left part of Fig. 5 The combined feature is represented as

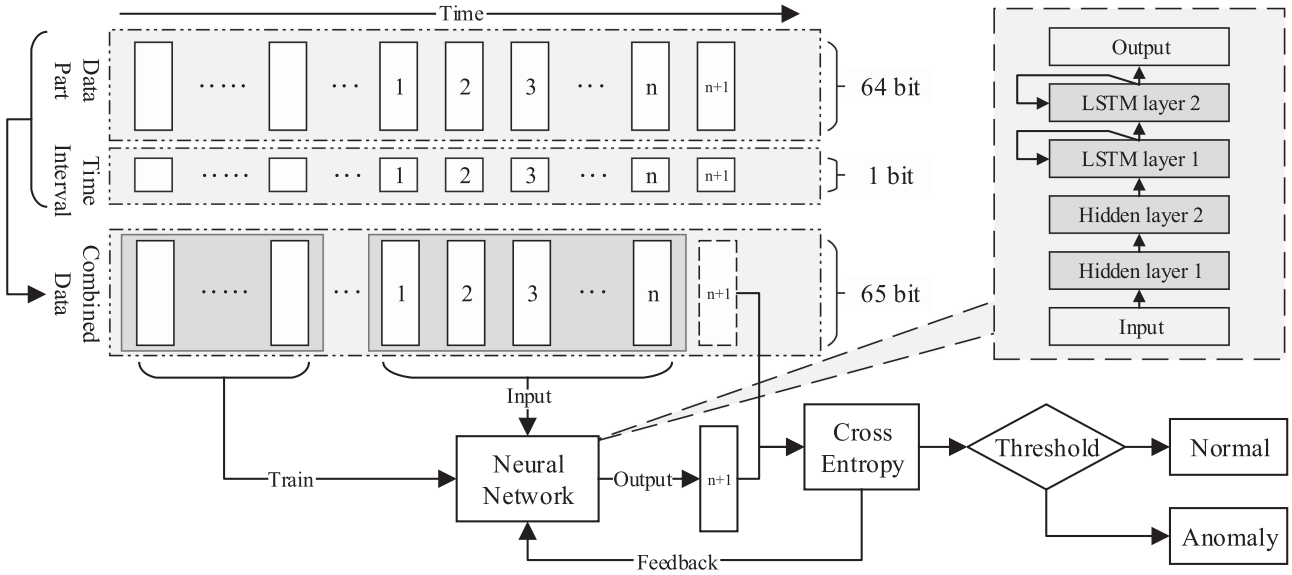


Fig. 5. The multi-dimension LSTM for CAN bus message anomaly detection.

vector $M \in \mathbb{R}^{65}$, written as:

$$F_m = \{D_m, \Delta T_{(m,m-1)}\} \tag{2}$$

where F_m represents the combined feature of CAN message m. It is a combination of Data feature D_m and time interval feature $\Delta T_{(m,m-1)}$.

C. LSTM Framework

Once the combined feature of CAN bus message is determined, we employ the multi-dimension LSTM framework to predict the CAN bus message for anomaly detection. The main idea is to predict information of next CAN message and then compare the predicted information with the real next one to determine normal or anomaly. As shown in the bottom part of Fig. 5, the LSTM framework is divided to two parts: CAN prediction and attack detection. In CAN prediction, a LSTM-based neural network is trained with real CAN bus messages to achieve precise prediction. In attack detection, cross entropy based decision function is used to determine normal or anomaly.

1) CAN Prediction: The CAN prediction is a trained LSTM-based neural network that employs the combined features of the latest n CAN bus messages as input to generate predicted features of next CAN message as output. Suppose that n CAN bus messages come in order with subtitle from 1 to n, the input of the prediction is represented by a vector $F_{m_1:m_n} \in \mathbb{R}^{65\times n}$, written as:

$$F_{m_1:m_n} = \{F_{m_1}, F_{m_2}, \ldots, F_{m_n}\}, \tag{3}$$

where F_{m_n} represents the combined feature of CAN message m_n. The output vector is the predicted combined feature of next CAN message, written as $F_{m_{n+1}}$. The core function of the predictor is to calculate $F_{m_{n+1}}$ from $F_{m_1:m_n}$ with LSTM neural networks.

In our proposed LSTM framework, the prediction contains four layers of neural networks, as shown in right upper part of Fig. 5, which are two hidden layers and two LSTM layers. In hidden layers, the input vector $F_{m_1:m_n}$ contains the combined feature from n CAN messages at different time, and the same weight and bias are used for every CAN message m_i to reduce the size of neural network. Then a Rectified Linear Unit (ReLU) activation function is used after the first hidden layer to enhance nonlinearity. These two hidden layers with ReLU function is expressed as:

$$O_h = \{\mathcal{F}_1(F_{m_i}), \forall F_{m_i} \in F_{m_1:m_n}\} \tag{4}$$

where O_h is the output of hidden layer two, F_{m_i} is combined feature of CAN message m_i in $F_{m_1:m_n}$, and $\mathcal{F}_1(\cdot)$ represents the two hidden layers with ReLU function. Then the output of hidden layer two O_h is directly used as the input to the LSTM layer.

The LSTM layer is composed of LSTM units. To process the time sequence iteratively, at each time step, two values are generated. One is the output at this time step, and the other is the memory that carries information of all previous time steps. Then the data of this time step and the memory value are used as the input of the process of next time step. Thus a data with m time steps as input on LSTM unit will be processed m times and it will generate a m-time-step data as output, in which the last mth time step is the only one that accumulates all time steps of input data. The output of LSTM layer two is a vector sized of $65 \times n$, where n is the total step. The last time step whose size is 65×1 is selected as the final output because it accumulates the information of total n time steps. These two LSTM layers is expressed as:

$$O_l = [\mathcal{F}_2(O_h)](n) \tag{5}$$

where O_l is the final output of LSTM layer two, $\mathcal{F}_2(O_h)$ is origin output of two LSTM layers that sized of $65 \times n$, and n represents the nth time step of the output. Thus O_l sized of 65×1 is the output of the prediction process that represents the predicted information of next CAN message. Given such

an overall prediction process, we discuss the most significant component loss function, which is used in the LSTM neural network that determines the value of input and output of each time step.

The loss function based on cross entropy is employed to evaluate the difference between the information of the real next $(n+1)$th CAN message and the predicted one. The loss is divided into two parts: the time interval ΔT with predicted time interval $\widehat{T}$ and 64-bit Data D with predicted Data $\widehat{D}$. For the time interval , the loss function $\mathcal{L}_t(w,u)$ is expressed as:

$$\mathcal{L}_t(w,u) = T\log_2\left(\widehat{T}(w,u)+\xi\right) + (1-T)\log_2\left(1-\widehat{T}(w,u)+\xi\right) \tag{6}$$

where w represents the parameter set in neural networks including weight and bias in hidden layers and so on, u represents a group of input and output CAN messages' information that sizes of $65\times(n+1)$. T is the real time interval, while $\widehat{T}$ is predicted time interval that depends on the input part of u and the network parameter w. In log function, ξ represents a small offset to avoid infinity, and an empirical value 1×10^{-10} is used. The cross-entropy based loss grows exponentially with the difference between T and $\widehat{T}$. As for the 64-bit Data, the loss $\mathcal{L}_d$ is similar to $\mathcal{L}_t$, written as:

$$\mathcal{L}_d(w,u) = \sum_{d\in D,\widehat{d}\in\widehat{D}} d\log_2\left(\widehat{d}(w,u)+\xi\right) + (1-d)\log_2\left(1-\widehat{d}(w,u)+\xi\right) \tag{7}$$

where w and u are the same as Eq. 6, d is the real one bit in 64-bit, and $\widehat{d}$ is the predicted one. Thus, the loss function of 64-bit Data part D is sum of cross-entropy of every bit. The loss of the information of a CAN message is written as:

$$\mathcal{L}_m(w,u) = \mathcal{L}_d(w,u) + c\cdot\mathcal{L}_t(w,u) \tag{8}$$

where c is a coefficient to balance the time and data parts. As the neural network is trained by batches of groups of input and output data, the actual effect is the loss of a batch $\mathcal{L}_b$, written as:

$$\mathcal{L}_b(w,v) = \sum_{M\in\mathcal{L}_b}\mathcal{L}_m(w,u) \tag{9}$$

where v represents a batch of groups of input and output CAN messages' information that sizes of $65*(n+1)*|\mathcal{L}_b|$. Therefore, the $\mathcal{L}_b(w,v)$ is the final loss function as well as the optimization object of the training. The optimization target is to minimize the difference between the predicted and the real information of the next CAN message, that is to obtain a optimal parameter set w^* to achieve minimization of the batch loss $\mathcal{L}_b(w,v)$, written as:

$$w^* = \arg\min\mathcal{L}_b(w,v). \tag{10}$$

The optimization can be achieved by the Adam optimization algorithm [26]. This neural network with learned parameters w^* has investigated the feature and pattern of CAN bus data and has ability to predict the coming CAN messages.

2) Attack Detection: After the neural network is trained and CAN bus message feature is predicted, the detection process uses the real CAN message and the predicted one as input to classify this real message is normal or anomaly. The main purpose of detection is to evaluate the difference between the predicted and the real information of the latest CAN message, so it is similar to the loss function in last subsection. The evaluation of difference is expressed as $\mathcal{E}$, it is also divided into two parts: the time interval and Data. The expressions are the same as $\mathcal{L}_t(w,u)$ and $\mathcal{L}_d(w,u)$ as shown in Eq. 6 and Eq. 7, but the value w is with the optimized w^*. A bigger value of $\mathcal{E}_t$ or $\mathcal{E}_d$ shows more different between the predicted and the real one. Then two thresholds of two parts are used to decide whether the real latest CAN message is normal or anomaly. To reduce the unbalance from the difference between two thresholds, the form proportion of is used to calculate the final difference evaluation $\mathcal{E}$ as follows:

$$\mathcal{E}(w^*,u) = \frac{\mathcal{L}_t(w^*,u)-\Gamma_t}{\Gamma_t} + \frac{\mathcal{L}_d(w^*,u)-\Gamma_d}{\Gamma_d} \tag{11}$$

where Γ_t and Γ_d are the thresholds of time interval and data cross entropy. If $\mathcal{E}$ is more than zero, the real latest CAN message is determined as abnormal data. Otherwise, it is determined as normal data.

V. Mobile Edge Assisted Multi-Task LSTM

Since LSTM neural network owns many layers and needs to process different types of features in one network, the computation time is too long to discover the anomaly in CAN bus message as CAN bus stream usually takes very short time to flow in the in-vehicle network. In spite that the computation task can be offloaded to the cloud, it introduces significant amount of delay which does not adapt to the driving condition. Therefore, we propose a mobile edge-assisted multi-task LSTM.

Due to different characteristics of time interval feature and Data feature, the multi-dimensional anomaly detection has the inherent characteristics to exploit multi-task LSTM framework as its input feature is a combination of time interval feature and Data feature. In the multi-task LSTM, the combination of features can be processed in parallel. Different from two single LSTM neural networks in which time-based feature and data-based feature are processed independently without sharing any common computation. Multi-task LSTM processes the common computation in the same process thus reduce the duplication of computation. After the common part, each task in the LSTM can be processed in parallel. Compared with one LSTM neural network, multi-task LSTM framework has twofold of benefits. First, the multi-task LSTM allow the parallel computation of the neural network to multiple computing servers. Second, it reduces the computation cost by put the common computation task into one procedure.

Mobile Edge Computing (MEC) [27] is a model enabling business-oriented cloud computing platform within the radio access network at the close proximity of mobile subscribers to serve delay sensitive, context aware applications [28]. Fig. 6 shows a MEC structure employed in Internet of Vehicles. A group of vehicles with close proximity are connected to the same MEC server for advanced driving services. MEC servers

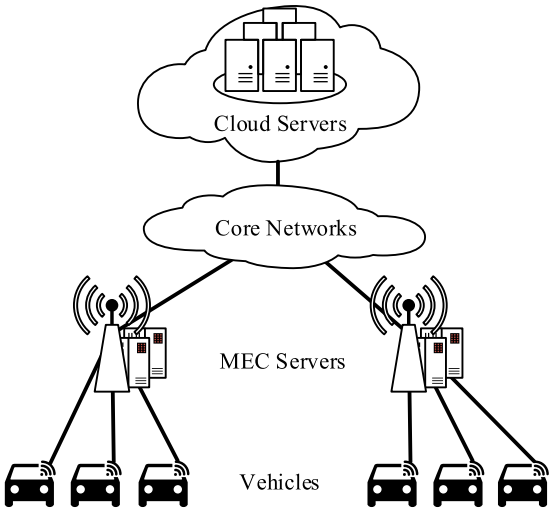


Fig. 6. Basic architecture of mobile edge computing.

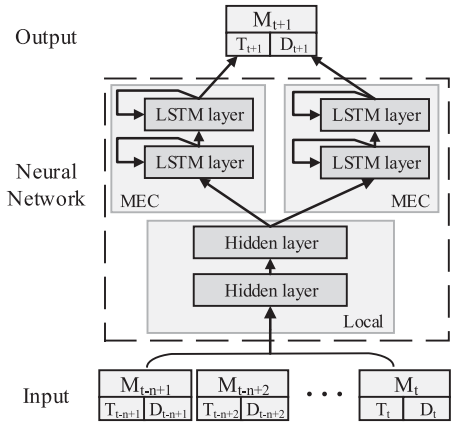


Fig. 7. Multi-Task LSTM with MEC server.

are further connected to the backhaul infrastructures for communication. The optimization of deep learning algorithms under the MEC framework is mainly focused on partitioning [29] and Right-Sizing [30] of existing models. For a M-layered network, the partitioning method looks for a cutting point N, realizing a result that Layer 1 to Layer N of the deep learning network run locally, while Layer $N+1$ to Layer M running on the edge. With the latter result on the edge, the overall time consumption or power consumption can be minimized. For the right-sizing, the network is trained as a multi-layered architecture, enabling the base layer to perform tasks with lower precision with the precision layers spare and ready to be used.

A multi-task network is considered capable of extracting a low-dimensional representation of the input data that is shared among a set of multiple related tasks. It sheds light on the optimization of neural network on MEC. By employing the multi-task architecture of LSTM neural network, the anomaly detection can be allocated to different mobile edges so that each mobile edge can achieve different tasks and make the LSTM in parallel. In the context of MEC, the multi-task LSTM neural network for anomaly detection can use MEC server for computation with close proximity. On the one hand, MEC server is capable of complex computing such as LSTM neural network with low computation latency; on the other hand, as MEC server are close to the end users, the vehicle does not need too much time for data transmission.

Combining the advantage of multi-task LSTM and MEC, a part of workload in multi-task LSTM can be offloaded to MEC for computation speedup. In practice, for the multi-task LSTM anomaly detection, we apply one computation party for the common hidden layer computation. After two tiers of hidden layers, the network further branches into sub-networks that predict the data and time respectively. Specifically, each sub-network first passes through a tier of hidden layers in order to adjust the data dimensions, and then two tiers of LSTM layers in order to complete the core task of time series prediction. Fig. 7 shows a typical structure to deploy our multi-task LSTM to MEC. The initial input of the multi-task LSTM is still the combined Data feature and time interval feature. The common hidden layer can be executed by on-board device. The time-based feature and data-based feature can be processed in parallel in MEC for prediction. Afterwards, the framework combines prediction results to detect anomaly. The multi-task LSTM certainly can be further optimized to reduce the time consumption by well optimized workload on MEC. As our work mainly focus on the deployment architecture on MEC server, a more complete optimization of multitasking networks under the MEC structure remains to be studied and is beyond the scope of this paper.

VI. Performance Evaluation

In this section, we first evaluate the performance of the proposed mechanism in terms of anomaly detection accuracy. Then we build up a MEC assisted environment to show the benefits of multi-task LSTM on time consumption.

A. Experiment Setup

We collect a total of 4.14 million CAN bus messages from vehicles as the data trace, which contains the time of the CAN bus message generated, and the binary form of the CAN message. The data is collected in various driving conditions including parking, acceleration, deceleration, and steering, driving and etc. to show the ubiquity. We employ the collected data to evaluate the performance of the proposed mechanism and compare it with baseline mechanisms. In particular, we compare the proposed basic multi-dimension LSTM method (MD-LSTM) and mobile edge assisted multi-task LSTM mechanism (MT-LSTM) with four baseline algorithms which represent different types of CAN message anomaly detection methods for the evaluation comparison: (1) in time dimension, a time based mechanism [10] using clock drift as anomaly metric (Clock-Drift), (2) a LSTM based on time feature (T-LSTM), (3) in data dimension, a LSTM based on data feature (D-LSTM) [8], and (4) in both data and time dimension but with Support Vector Machine (MD-SVM). In the experiment, we empirically set the value of n as 10, average Γ_t as 0.63, average Γ_d as 38 and learning rate as 0.0002. For the convergence of results, we run each experiments 1000 times,

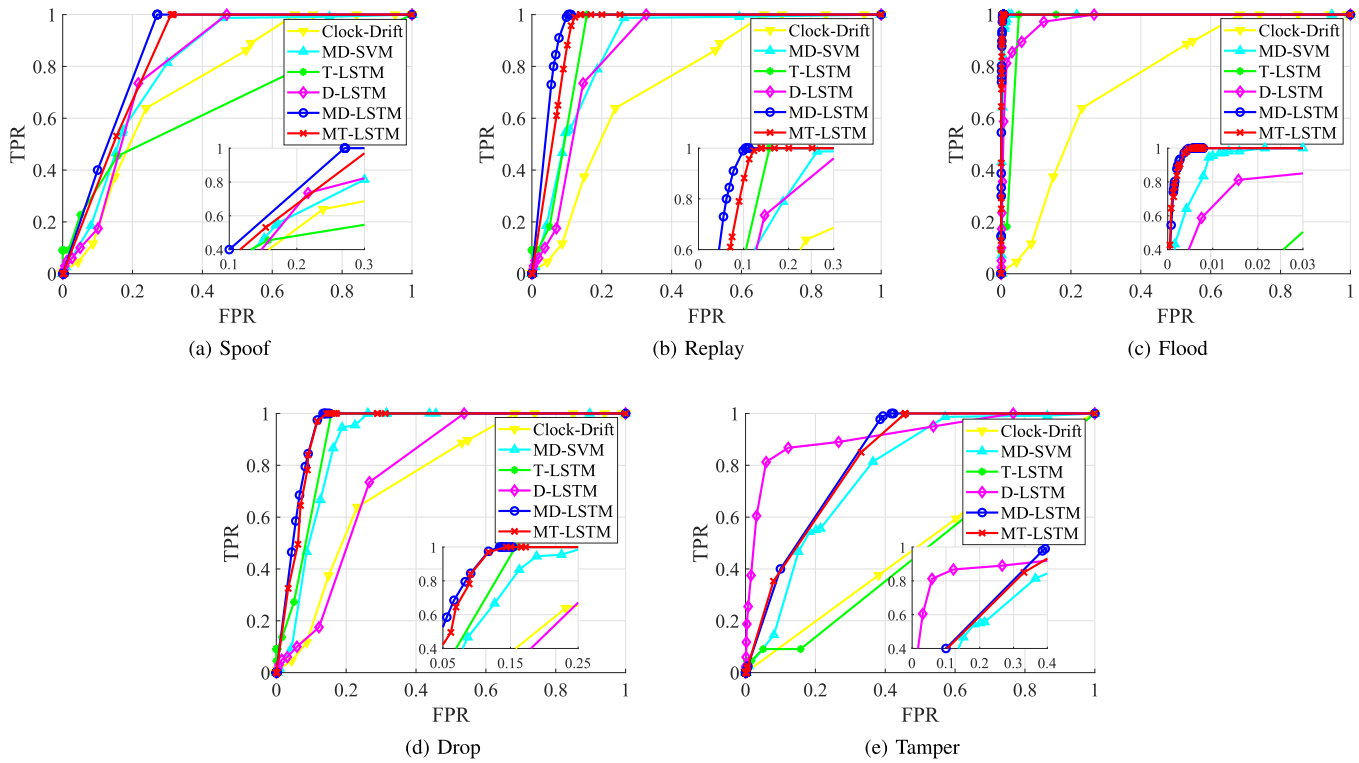


Fig. 8. ROC curves of different anomaly detection mechanisms under five kinds of attacks.

TABLE I
PERFORMANCE OF DIFFERENT MECHANISMS

	Recall	Precision	Accuracy	F_1 Score
Clock-Drift	0.596	0.687	0.658	0.638
MD-SVM	0.674	0.712	0.742	0.675
T-LSTM	0.791	0.782	0.792	0.778
D-LSTM	0.838	0.826	0.826	0.829
MD-LSTM	0.984	0.856	0.901	0.913
MT-LSTM	0.962	0.853	0.893	0.902

and for the measurement of accuracy, we use 70% of collected normal data as the training data and 30% as the testing data, half of which are anomaly data of five attacks that generates from related abnormal data patterns.

B. *Effectiveness Evaluation*

We first evaluate the performance of anomaly detection mechanisms by typical metrics such as precision, accuracy and F_1 Score. Furthermore, we measure the performance of different mechanisms for detecting certain types of attacks using the Receiver Operating Characteristic (ROC) [31] metric, which outlines the relationship between True Positive Rate (TPR) and False Positive Rate (FPR). Generally speaking, the higher TPR and the lower FPR, the anomaly detection mechanism is better.

The performance of different mechanisms is shown in Table. I. It shows that the recall of MD-LSTM and MT-LSTM reaches 0.984 and 0.962. MD-LSTM has a slightly better performance than MT-LSTM. Compared with baseline approaches, both MD-LSTM and MT-LSTM have about 40% higher recall than the Clock-Drift approach. They also have 30%, 20% and 15% higher recall compared with MD-SVM, T-LSTM and D-LSTM. With respect to the precision, both MD-LSTM and MT-LSTM have more than 85% of precision, outperforming baseline approaches up to 16%. Furthermore, MD-LSTM and MT-LSTM has about 90% of accuracy, with up to 24% of higher accuracy compared with baseline algorithms. Additionally, both MD-LSTM and MT-LSTM have over 90% of F_1 Score, which is also with up to 26% more than that of baseline approaches. Compared with baseline algorithms, both MT-LSTM and MD-LSTM are with better performance by considering data and time dimension features thus can detect anomaly CAN messages from both dimensions. Although MD-SVM also consider both dimensions, due to the inefficiency of SVM, it cannot reach as good performance as MD-LSTM and MT-LSTM.

To show the performance of different mechanisms for different types of anomaly detection, we run the anomaly detection under different attacks, as shown in Fig. 8. In particular, Fig. 8a shows the ROC curves of different mechanisms for spoof attack detection. It presents that the proposed MD-LSTM and MT-LSTM are with higher TPR and lower FPR than baseline approaches. In particular, as the increasing of FPR, TRP becomes larger until it reaches to 1 for all mechanisms. The TPR of MD-LSTM and MT-LSTM reach to 1 when their FPR are 0.25 and 0.3 respectively. In contrast, the TPR of baseline approaches converge much more slowly. The results are in line with the observed accuracy, precision and F_1 score of the entire data trace. As spoof attack affects the time dimension features in forms of

TABLE II
AVERAGE RUN TIME USED FOR EACH CAN DETECTION

Mechanism	Platform	Mode	Run Time[ms]
MD-LSTM	MEC server	non-parallel	1.31
MD-LSTM	OBU only	non-parallel	144
MT-LSTM	MEC server	non-parallel	1.42
MT-LSTM	OBU only	non-parallel	159
MT-LSTM	OBU+MEC	parallel	$44 + 0.92$
MT-LSTM	MEC server	parallel	0.61

acceleration, the performance of clock drift and T-LSTM are significantly degraded compared with others. Meanwhile, as spoof attack also affects data dimension features in forms of discontinuation, resulting in the performance degradation of D-LSTM as well. Similarly, in the case of attacks such as replay, flood and drop, as shown in Fig. 8b, 8c and 8d, the proposed MD-LSTM and MT-LSTM outperform baseline algorithms in the ROC curve, which shows the advantages of combined multi-dimensional features as these attacks affect data and time dimension features in different aspects. Even in the case of tamper attack, as shown in Fig. 8e, where the data and time dimension features are difficult to distinguish, the ROC curves of MD-LSTM and MT-LSTM outperforms that of T-LSTM, MD-SVM and Clock-Drift, but have slightly lower than that of D-LSTM. This attributes to the fact that the undistinguishable data and time features may introduce noisy features misleading the decision of the proposed MD-LSTM and MT-LSTM approaches since they combine both data and time dimensional features. However, since the tempered CAN bus frame cannot convey any useful vehicle control meanings, irregular temper attacks are not common in the in-vehicle network.

C. Efficiency Evaluation

We further evaluate the efficiency of the proposed mobile edge assisted multi-task LSTM with respect to its efficiency. We measure the efficiency of different mechanisms by the run time.

In the experiment, we use a mobile edge computing (MEC) server with 3.3GHz dual core Intel i5 CPU. Its computation capability measured by Floating-point Operations Per Second (FLOPS) [32] is 50G FLOPS. For the On-Board Unit (OBU) in the vehicle, we employ an embedded device with an ARM CPU, and its computation capability is 50M PLOPS. Taking the advantage of the paralleling of multi-task LSTM, the computation of LSTM can be divided to many pieces of computation components. Three main components in multi-task LSTM is hidden layer component, time task and data task, where hidden layer is the foundation of the time task and data task. We evaluate the MD-LSTM and MT-LSTM on different device combinations and correspondingly different modes of mechanisms. The run time of different combinations is shown in Table II. We measure the run time of MD-LSTM on OBU and MEC server respectively. In the case that MD-LSTM only runs on OBU, its run time is 144 milliseconds. In contrast, if the MD-LSTM only runs on MEC server, its run time is shortened to 1.31 milliseconds, which is 100 times faster than that on OBU. However, as MD-LSTM uses only one LSTM neural network, indicating it runs on either MEC server or OBU without parallel. Compared with MD-LSTM, MT-LSTM has more flexible parallel configuration on both OBU and MEC server. In the case that MT-LSTM running on only OBU or MEC server, the run time of MT-LSTM is 159 milliseconds and 1.42 milliseconds, respectively. The run time is even long than that of MD-LSTM. This may attribute to the common computation in multi-task LSTM neural network. However, taking advantage of multi-task, it allows two LSTM neural network to run in parallel. Two different device combinations are tested. The first one is running the common and fundamental hidden layer on the OBU while running the upper data domain and time domain detection tasks parallel on MEC server. Its run time is 44.92 milliseconds. OBU takes 44 milliseconds and MEC server takes 0.92 milliseconds. If a completely parallel on MEC server, the run time of MD-LSTM can be as short as 0.61 millisecond. It is much shorter than that of MD-LSTM running on MEC server. Applying MEC server introduces network delay needed to deliver CAN bus message to the MEC server. However, as the proliferation of 5G and C-V2X [33] technology, which allows the delay to be short as 10 milliseconds or even 1 millisecond, which is sufficient for the detection. Overall, taking advantage of multi-task, the detection mechanism is more flexible and easily deployed to MEC servers. Different device combination can be employed in practice depending on the condition of the detection.

VII. CONCLUSION

In this paper, we proposed a multi-task LSTM mechanism assisted by the mobile edge to cope with the anomaly detection challenges for in-vehicle network in IoVs. It incorporates features of both the dimension of time and the dimension of data to enhance the detection accuracy of anomaly pattern. Besides, to overcome the computing capacity limit of on-board devices while enriching the training data trace, the proposed mechanism employed the multi-task LSTM framework with the assistance of the mobile edge. Furthermore, the proposed mechanism can be applied to different kinds of vehicles without digging into the semantics of CAN bus messages. The evaluation results show that the proposed mechanism achieves 90% of accuracy. With the assistance of mobile edge, the anomaly detection for one CAN message can be finished within 0.61 milliseconds on the average. For the future work, we will look into the optimization of multi-task LSTM computation on MEC servers.

REFERENCES

[1] K. A. Bayindir, M. A. Gzkk, and A. Teke, "A comprehensive overview of hybrid electric vehicle: Powertrain configurations, powertrain control techniques, and electronic control units," *Energy Convers. Manage.*, vol. 52, no. 2, pp. 1305–1313, 2011.

[2] D. Zhang, D. Meng, and J. Han, "Co-saliency detection via a self-paced multiple-instance learning framework," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 5, pp. 865–878, May 2017.

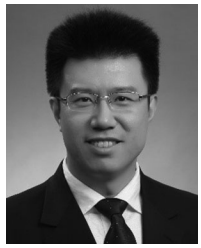
[3] J. Han, D. Zhang, X. Hu, L. Guo, J. Ren, and F. Wu, "Background prior-based salient object detection via deep reconstruction residual," *IEEE Trans. Circuit Syst. Video Technol.*, vol. 25, no. 8, pp. 1309–1321, Aug. 2015.

[4] D. Zhang, J. Han, C. Li, J. Wang, and X. Li, "Detection of co-salient objects by looking deep and wide," *Int. J. Comput. Vision*, vol. 120, no. 2, pp. 215–232, 2016.
[5] I. Bisio, C. Garibotto, A. Grattarola, F. Lavagetto, and A. Sciarrone, "Smart and robust speaker recognition for context-aware in-vehicle applications," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8808–8821, Sep. 2018.
[6] Z. Ma, H. Yu, W. Chen, and J. Guo, "Short utterance based speech language identification in intelligent vehicles with time-scale modifications and deep bottleneck features," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 121–128, Jan. 2019.
[7] K. S. L. of Tencent, Car Hacking Research: Remote Attack Tesla Motors. (2016). [Online]. Available: http://keenlab.tencent.com/en/2016/09/19/Keen-Security-Lab-of-Tencent-Car-Hacking-Research-Remote-Attack-to-Tesla-Cars/
[8] A. Taylor, S. Leblanc, and N. Japkowicz, "Anomaly detection in automobile control network data with long short-term memory networks," in *Proc. IEEE Int. Conf. Data Sci. Adv. Analytics*, 2016, pp. 130–139.
[9] H. M. Song, H. R. Kim, and H. K. Kim, "Intrusion detection system based on the analysis of time intervals of can messages for in-vehicle network," in *Proc. Int. Conf. Inf. Netw.*, 2016, pp. 63–68.
[10] K.-T. Cho and K. G. Shin, "Fingerprinting electronic control units for vehicle intrusion detection," in *Proc. 25th USENIX Secur. Symp. (USENIX Security 16)*. Austin, TX, USA: USENIX Association, 2016, pp. 911–927. [Online]. Available: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/cho
[11] M. J. Kang and J. W. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *Plos One*, vol. 11, no. 6, p. e0155781, 2016.
[12] A. Graves, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[13] O. Pfeiffer, A. Ayre, and C. Keydel, *Embedded Networking With CAN and CANopen*. Greenfield, Massachusetts, USA: Copperhill Media Corporation, 2008.
[14] Q. Gong *et al.*, "Deepscan: Exploiting deep learning for malicious account detection in location-based social networks," *IEEE Commun. Mag.*, vol. 56, no. 11, pp. 21–27, Nov. 2018.
[15] H. Ji, Y. Wang, H. Qin, Y. Wang, and H. Li, "Comparative performance evaluation of intrusion detection methods for in-vehicle networks," *IEEE Access*, vol. 6, pp. 37 523–37 532, 2018.
[16] A. Taylor, N. Japkowicz, and S. Leblanc, "Frequency-based anomaly detection for the automotive can bus," in *Proc. Ind. Control Syst. Secur.*, 2016, pp. 45–49.
[17] M. R. Moore, R. A. Bridges, F. L. Combs, M. S. Starr, and S. J. Prowell, "Modeling inter-signal arrival times for accurate detection of can bus signal injection attacks: A data-driven approach to in-vehicle intrusion detection," in *Proc. Conf. Cyber Inf. Secur. Res.*, 2017, p. 11.
[18] H. Ji, Y. Wang, H. Qin, X. Wu, and G. Yu, "Investigating the effects of attack detection for in-vehicle networks based on clock drift of ecus," *IEEE Access*, vol. 6, pp. 49 375–49 384, 2018.
[19] M. Marchetti and D. Stabili, "Anomaly detection of can bus messages through analysis of id sequences," in *Proc. Intell. Vehicles Symp.*, 2017, pp. 1577–1583.
[20] M. Muter and N. Asaj, "Entropy-based anomaly detection for in-vehicle networks," in *Proc. IEEE Intell. Vehicles Symp.*, 2011, pp. 1110–1115.
[21] H. Li, L. Zhao, M. Juliato, S. Ahmed, M. R. Sastry, and L. L. Yang, "Poster: Intrusion detection system for in-vehicle networks using sensor correlation and integration," in *Proc. ACM Sigsac Conf.*, 2017, pp. 2531–2533.
[22] P. S. Murvay and B. Groza, "Source identification using signal characteristics in controller area networks," *IEEE Signal Process. Lett.*, vol. 21, no. 4, pp. 395–399, Apr. 2014.
[23] J. Han *et al.*, "Representing and retrieving video shots in human-centric brain imaging space," *IEEE Trans. Image Process.*, vol. 22, no. 7, pp. 2723–2736, Jul. 2013.
[24] Z. Ma, Y. Lai, W. B. Kleijn, Y.-Z. Song, L. Wang, and J. Guo, "Variational Bayesian learning for dirichlet process mixture of inverted Dirichlet distributions in Non-Gaussian image feature modeling," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 2, pp. 449–463, Feb. 2019.
[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, arXiv:1412.6980.
[27] X. Chen, W. Li, S. Lu, Z. Zhou, and X. Fu, "Efficient resource allocation for on-demand mobile-edge cloud computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8769–8780, Sep. 2018.
[28] A. Ahmed and E. Ahmed, "A survey on mobile edge computing," in *Proc. Int. Conf. Intell. Syst. Control*, 2016, pp. 1–8.
[29] Y. Kang *et al.*, "Neurosurgeon: Collaborative intelligence between the cloud and mobile edge," *ACM Sigplan Notices*, vol. 52, no. 4, pp. 615–629, 2017.
[30] E. Li, Z. Zhou, and X. Chen, "Edge intelligence: On-demand deep learning model co-inference with device-edge synergy," in *Proc. MECOMM@SIGCOMM*, 2018, pp. 31–36.
[31] S. Pundir and R. Amala, "Parametric receiver operating characteristic modeling for continuous data: A glance," *MASA*, vol. 9, pp. 121–135, 2014.
[32] S. F. Oberman and M. J. Flynn, "Design issues in division and other floating-point operations," *IEEE Trans. Comput.*, vol. 46, no. 2, pp. 154–161, Feb. 1997.
[33] A. Nabil, V. Marojevic, K. Kaur, and C. B. Dietrich, "Performance analysis of sensing-based semi-persistent scheduling in C-V2X networks," in *Proc. IEEE 88th Vehicular Technol. Conf. (VTC-Fall)*, 2018.

Konglin Zhu received the master's degree in computer Science from the University of California, Los Angeles, CA, USA, and the Ph.D. degree from the University of Goettingen, Goettingen, Germany, in 2009 and 2014, respectively. He is now an Associate Professor with the Beijing University of Posts and Telecommunications, Beijing, China. His research interests include Internet of Vehicles and information security.

Zhicheng Chen received the bachelors' degree from Shandong University, Jinan, China, in 2017. He is currently working toward the masters' degree at the Advanced Networking Technology Laboratory, Beijing University of Posts and Telecommunications, Beijing, China. His current research interests include information security in Internet of Vehicles.

Yuyang Peng received the M.S. and the Ph.D. degrees in electrical and electronic engineering from Chonbuk National University, Jeonju, South Korea, in 2011 and 2014, respectively. From 2014 to 2018, he has been a Postdoctoral Research Fellow with the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea. He is currently an Assistant Professor with the Faculty of Information Technology at Macau University of Science and Technology (MUST), Taipa, Macau. His current research interests include cooperative communications, energy optimization, and cloud computing.



Lin Zhang received the B.S. and the Ph.D. degrees in 1996 and 2001, both from the Beijing University of Posts and Telecommunications, Beijing, China. From 2000 to 2004, he was a Postdoctoral Researcher with Information and Communications University, Daejeon, South Korea, and Nanyang Technological University, Singapore, respectively. He joined Beijing University of Posts and Telecommunications in 2004, where he has been a Professor since 2011. His current research interests include mobile cloud computing and Internet of Things.