Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/MBE

MBE, 19(7): 6996–7018.
DOI: 10.3934/mbe.2022330
Received: 13 March 2022
Revised: 12 April 2022
Accepted: 25 April 2022
Published: 11 May 2022

Research article

GAN model using field fuzz mutation for in-vehicle CAN bus intrusion detection

Zhongwei Li[1], Wenqi Jiang[1], Xiaosheng Liu[1,*], Kai Tan[2], Xianji Jin[1] and Ming Yang[1]

[1] School of Electrical Engineering and Automation, Harbin Institute of Technology, Harbin 150001, China
[2] School of Cyberspace Science, Harbin Institute of Technology, Harbin 150001, China

* **Correspondence:** Email: liuxsh@hit.edu.cn.

Abstract: Controller area network (CAN) are widely used in smart vehicles to realize information interactions between electronic control units and other devices in vehicles. Owing to an increase in external communication interfaces, the cybersecurity of in-vehicle CAN bus networks is threatened. In-vehicle CAN intrusion detection systems with high detection rates and low false-negative rates have become important security protection measures for automotive networks. The boundary of the current machine learning-based in-vehicle CAN bus intrusion detection algorithm to determine the anomalous behavior triggered by CAN messages is unclear, and a validity check is required after the intrusion detection algorithm is designed. To solve the low coverage rate problem in the process of validating intrusion detection algorithms, an in-vehicle CAN fuzz-testing message generation model, the field-associative mutation generation adversarial network (FAMGAN), is proposed. To improve the defects of high randomness in generating messages in traditional fuzz-testing algorithms, FAMGAN adopts field division based on a conditional random field and the field association method based on the Apriori algorithm. Experiments were conducted on a real car using a code-built intrusion detection algorithm. The results demonstrate that FAMGAN can efficiently generate anomalous CAN messages and evaluate the performance of an in-vehicle CAN intrusion detection algorithm.

Keywords: controller area network bus; in-vehicle network; cyber security; generation adversarial network; fuzz testing; intrusion detection; FAMGAN

1. Introduction

Artificial intelligence, cloud computing, and other advanced technologies are widely used in various fields, and intelligent connected vehicle technology has become the key development direction of the automotive industry [1]. An intelligent connected vehicle can analyze and make decisions to realize powerful functions through an onboard sensing system and information interaction in a car network [2]. However, the development of intelligent networked vehicles causes serious cyber security threats, and the increase in interfaces outside of the car leads to a significant increase in the attackable path [3,4]. The basic electric facilities connected to intelligent connected vehicles, such as charging piles and smart grids, have a higher probability of attacks starting from intelligent connected vehicles and faults, and the prognosis of the above facilities has also become a popular research content [5].

Taking electronic control units (ECUs) as an example, most of the ECUs communicate with each other through the CAN bus, and the CAN bus is not designed considering cyber security protection mechanisms. The security vulnerability of in-vehicle CAN buses leads to greater challenges for automotive cybersecurity [6]. Intrusion detection algorithms for in-vehicle CAN buses have gradually become a research focus in the field of cyber security [7–9]. Currently, most common intrusion detection algorithms use message datasets for training, and the message datasets collected from real vehicles cannot meet the coverage requirements of an algorithm for detecting anomalous messages. The effectiveness of the detection algorithms cannot be verified effectively [10,11].

Therefore, an in-vehicle CAN fuzz-testing message generation model, namely, the field-associated mutation generation adversarial network (FAMGAN), was designed in this study. This model can generate messages conforming to the in-vehicle CAN protocol specification, which can be applied to the training set or test set of the intrusion detection algorithm to solve the problems of low coverage and poor specificity of traditional intrusion detection algorithms. To generate test messages that conform to the real message protocol specification, FAMGAN uses an in-vehicle CAN message bit transition rate to divide the data field and select conditional random fields (CRFs) to validate the field division result. FAMGAN uses the Apriori algorithm to perform association mining on in-vehicle CAN message fields to extract the relationship between message fields; this improves the similarity between the generated and tested messages. The generators and discriminators in FAMGAN are trained to generate a large number of in-vehicle CAN bus messages that can be used to test the intrusion detection algorithm.

The main contributions of this study are as follows.

1) The Wasserstein generative adversarial network gradient penalty (WGAN-GP)-based fuzz-testing method for in-vehicle CAN messages was employed. The WGAN-GP model was used to learn the protocol format of the CAN bus messages and induce mutations to generate test cases similar to the original messages.

2) To improve the specificity of the WGAN-GP model for in-vehicle CAN messages, an in-vehicle CAN protocol was analyzed, and a field association variant model was developed to optimize the WGAN-GP model by analyzing the in-vehicle CAN message field association.

3) A comparison of experimental results obtained using a real car and a code-built intrusion detection algorithm revealed that the proposed FAMGAN has good in-vehicle CAN anomaly message generation efficiency.

The remainder of this paper is organized as follows. Section 2 describes the current research status of in-vehicle CAN bus intrusion detection and CAN message fuzz-testing techniques, as well as the principles, advantages, and disadvantages of the relevant models. Section 3 introduces FAMGAN, including the mathematical model and implementation process. Section 4 analyzes the performance metrics of the proposed model based on its implementation in a real car, using intrusion detection algorithms. Finally, Section 5 concludes the paper and discusses future research directions.

2. Related research

Numerous security studies on in-vehicle CAN bus communication networks have been conducted in recent years. In 2013, Valasek and Miller controlled the Ford Impala and Toyota Prius to achieve steering and braking actions through the OBD-II interface [12]. In 2015, they used a laptop to control a Jeep Cherokee remotely [13]. In 2016, they launched an attack experiment on a Jeep Cherokee, used a laptop to connect to the onboard diagnostics (OBD) interface of the vehicle, and successfully controlled the steering and braking mechanisms [14]. In 2018, three security researchers designed an in-vehicle CAN bus security test for an attack traffic generation tool that enables the automatic generation of attack loads for multiple attack modes [15]. It is clear that automotive cyber security issues are continuously being exposed, and researchers have proposed a large number of in-vehicle CAN bus message intrusion detection algorithms and testing tools. In this section, the principles of in-vehicle CAN bus message intrusion detection and related studies on in-vehicle CAN message fuzz testing are reviewed.

2.1. In-vehicle CAN bus intrusion detection algorithm

Intrusion detection for an in-vehicle CAN bus can be divided into rule-based, feature-based, and machine learning-based intrusion detection. The rule-based intrusion detection algorithm judges anomalous messages by establishing a rule database. This class of methods can detect simple signal injection attacks, and alarms occur when messages that do not conform to the protocol rules are detected. Olufowobi et al. used message timing and worst-case response time to develop specifications and extract CAN bus real-time model parameters, and they designed a rule-based intrusion detection system, SAIDUcANT, which has a high detection capability for data injection attacks [16].

A feature-based intrusion detection algorithm detects whether anomalies occur in messages by collecting sensitive features, such as the message sending frequency and ECU fingerprint, and comparing them with the subsequent in-vehicle CAN bus transmission features based on the formation of feature fingerprints. Zhou et al. proposed an in-vehicle CAN bus intrusion detection algorithm based on temperature fingerprints and observed that the quartz crystal used in the ECU is more sensitive to temperature. They proposed the concept of an ECU temperature-sensitive fingerprint based on the characteristics of ECU clock deviation with temperature change [17]. Li et al. conducted their research on this basis to demonstrate a temperature-varying voltage fingerprinting scheme that can be used to construct a fingerprint database for each ECU based on the voltage signals emitted at different temperatures and the current driving status of the car to perform intrusion detection [18]. In other research, a message-periodicity-based adaptive intrusion detection algorithm for an in-vehicle CAN bus was developed. The communication load of the vehicle CAN bus was analyzed, and the period fluctuation and intrusion detection accuracy were mined based on CAN message priority and a message transmission waiting mechanism [19]. Intrusion detection algorithm based on attack features has also received attention recently [20,21].

A machine learning-based intrusion detection algorithm continuously improves the ability to identify anomalous messages from normal messages by learning the message characteristics. Because the data features of messages sent by spoofing attacks do not conform to the normal variation pattern, the machine learning-based intrusion detection algorithm can detect spoofing attacks. In a previous study, we designed a linear-chain conditional random stochastic-based intrusion detection algorithm for an in-vehicle CAN bus that can identify the relevance of CAN message context content by constructing a dual-profile model of a linear CRF normal model and intrusion model [22]. He et al. proposed the hybrid similar neighborhood robust factorization machine (HSNRFM) intrusion detection model for detecting anomalies in CAN bus messages. This model outperforms several classical machine learning models in several evaluation metrics [23]. Xie et al. proposed an enhanced deep learning generative adversarial network (GAN) model with well-designed CAN message blocks and enhanced GAN discriminators for intrusion detection in various attacks and threats [24].

2.2. In-vehicle CAN bus fuzz-testing algorithm

Fuzz testing of in-vehicle CAN bus messages can be divided into protocol-based and machine learning-based fuzz testing. Protocol-based fuzz testing generates fuzz-testing cases using mutations. Such algorithms require knowledge of CAN message application layer protocol specification. Lee et al. performed fuzz testing using an open-source tool. They used a mutation-based fuzz-testing method by sending test messages to the vehicle CAN bus after blinking the indicator lights of the vehicle dynamic control system and the OBD indicator light alarm [25]. Fowler et al. developed a fuzz-testing tool for a CAN bus communication network. It was applied to the ECU of an experimental vehicle and could effectively investigate errors in the vehicle ECU software and defects in the design of the vehicle communication network. To solve the problem of exploding test case combinations, fuzz testing was performed on each of the 64 bits of the message data field. However, a fuzz-testing algorithm that focused on the CAN bus application layer protocol was not considered [26]. Protocol-based fuzz testing has two main drawbacks. First, if the CAN bus data field contains a checksum field, such algorithms can generate invalid message test cases that are ignored by the ECU receiving the message, thereby affecting the fuzz-testing results. Second, the generated data often cannot cover all possible inputs. Because the message data field has, at most, 8 bytes of data, 2^{64} test cases are required to generate all the message data, which can lead to a combinatorial explosion.

Machine learning-based fuzz testing does not require reverse analysis of the CAN bus message protocol. It enables the machine learning model to generate data in the same format as the original data, but with different content after the messages collected from the CAN bus are preprocessed and input to the machine learning model for training. An intrusion detection algorithm that uses a deep learning model to generate an adversarial network can detect unknown attacks using only normal data. An anomalous message dataset was constructed using fuzz testing for model training and evaluation experiments. This intrusion detection algorithm has high detection accuracy for unknown types of malicious attack [27]. Owing to the lack of knowledge of the KWP2000 protocol of the body network, Zhang et al. proposed a deep learning-based generative adversarial neural network for the body network KWP2000 protocol vulnerability mining algorithm. The forward feedback network was chosen as the generator and the support vector machine as the discriminator. The output data obtained were used as semi-valid data for fuzz testing [28].

Protocol-based fuzz testing requires access to the content of the protocol specification. In the absence of a protocol specification, a protocol inversion is required. Therefore, protocol-based fuzz-testing algorithms are more demanding for security testers. In addition, it is difficult to guarantee that the protocol field format will be correctly parsed when performing a protocol reversal analysis. It is necessary to ensure that the test case data are in the correct format and to ensure that there is some variability in the data content. Therefore, an appropriate machine learning model can be designed to learn the CAN bus protocol format and generate a large number of test cases for in-vehicle CAN bus fuzz testing. When researchers use such methods to construct test cases, they improve testing efficiency [29,30]. The generalizability of the CAN bus fuzz-testing algorithm is also improved because CAN private protocols vary from vehicle to vehicle.

3. FAMGAN model construction

The proposed FAMGAN model takes advantage of protocol-based fuzz testing and machine learning-based fuzz testing. FAMGAN mainly uses WGAN-GP to generate the original in-vehicle CAN bus fuzz-testing messages, adjusts parameters by creating a competing environment between the generator and discriminator, and generates in-vehicle CAN fuzz-testing messages that match a real in-vehicle environment. In addition, FAMGAN selects linear CRF to parse and divide in-vehicle CAN bus messages. The Apriori algorithm is used to realize in-vehicle CAN bus message field association mining, and the extracted field information is mapped onto the GAN generator and discriminator. Compared with the original WGAN-GP and related improved models, FAMGAN improves the specificity of the fuzz-testing message generation.

The basic functional flow of FAMGAN model is as follows: 1) Obtain SAE J1939 message dataset and protocol specification, generate field verification model of vehicle CAN messages by Linear-CRF, and derive real vehicle field division results by combining with vehicle CAN message field division algorithm. 2) Analyze field correlation after field division by Apriori algorithm, and generate test messages with test message constraints. 3) Input real vehicle CAN messages, test message constraints and random noise into the generative adversarial network to realize test message generation for fuzz testing.

Section 3.1 describes how WGAN-GP generates CAN fuzz-testing messages, as well as its implementation process. Section 3.2 analyzes the CAN bus message division characteristics and presents the linear-CRF-based CAN message division method. Finally, Section 3.3 explores the application of association information to the fuzz-testing message generation process of FAMGAN by mining the information of in-vehicle CAN message division using the Apriori algorithm. Furthermore, the overall implementation process of the FAMGAN model is described.

3.1. Generating testing message using WGAN-GP model

In-vehicle CAN bus fuzz testing is an algorithm for discovering CAN bus network security vulnerabilities by sending a large number of unintended messages to the in-vehicle CAN bus network and monitoring abnormal car operation status. Because the attack behavior characteristics of the in-vehicle CAN bus network differ greatly from those of the Internet, the fuzz-testing algorithms applied to the Internet protocol cannot be directly applied to the in-vehicle CAN bus network. The generated in-vehicle CAN bus test messages should ensure a high similarity to real messages generated by actual vehicles, and the overall in-vehicle CAN bus test messages should have a high coverage rate.

The FAMGAN model is based on the GAN model, which is a deep learning algorithm consisting of a generator and discriminator. Generators and discriminators generally adopt a neural network structure. The data generated by the generator gradually approach the real data by gaming.

The learning process of GAN must consider the dynamic optimization of the two neural networks at the same time, continuously adjusting the parameters of the two models so that they can be optimized simultaneously. The objective function optimized by the GAN is given by

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[\log D(x)] + E_{z\sim p_z(z)}[\log(1-D(G(z)))] \tag{1}$$

where D represents the discriminator, $D(x)$ represents the probability that x is the true data, G represents the generator, $G(z)$ is the probability distribution of the sample data z, P_{data} is the real data distribution, and P_z is the generated data distribution. The overall goal is to optimize V (D, G) gradually using adversarial training. For a further understanding of Equation (1), when training the discriminator, the optimization goal is to maximize the probability of correctly classifying the real and generated data:

$$\max_D V(D,G) = E_{x\sim p_{data}(x)}[\log D(x)] + E_{z\sim p_z(z)}[\log(1-D(G(z)))] \tag{2}$$

The optimization goal of training the generator is to minimize the probability that the generated data will be correctly identified by the discriminator:

$$\min_G V(D,G) = E_{z\sim p_z(z)}[\log(1-D(G(z)))] \tag{3}$$

However, the GAN itself also suffers from problems, such as model nonconvergence, pattern collapse, gradient disappearance, and gradient explosion. To address these problems, Arjovsky et al. proposed the Wasserstein generative adversarial network (WGAN) model [31]. In 2017, Gulrajani et al. proposed an improved WGAN [32], the WGAN-GP. It was pointed out that using weight clipping in the WGAN leads to the aforementioned adverse effects on training. The WGAN-GP removes the weight clipping term. To ensure that the discriminator still satisfies Lipschitz continuity, the WGAN-GP adds a gradient penalty term for random samples to the objective function of the discriminator, given as

$$-E_{x\sim p_{data}(x)}[D(x)] + E_{z\sim p_z(z)}[D(G(z))] + \lambda \underset{\tilde{x}\sim P_{\tilde{x}}}{E}[(\|\nabla_{\tilde{x}} D(\tilde{x})\|_2 - 1)^2] \tag{4}$$

Figure 1 shows the process of the WGAN-GP-based in-vehicle CAN bus message fuzz-testing method. The specific implementation steps are as follows.

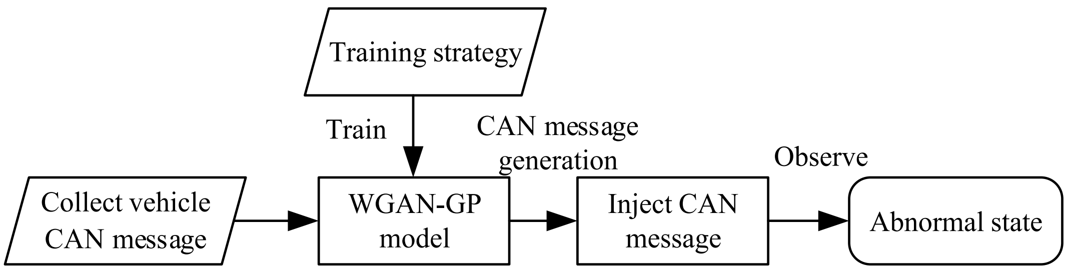


Figure 1. Fuzz-testing process based on WGAN-GP.

1) Dataset construction: Collected CAN messages are usually displayed in hexadecimal format, and preprocessing of the message data is required to construct the dataset for training the GAN. In this approach, redundant and repetitive data are removed from the message dataset, and then the hexadecimal data are converted to decimal data. The processed data constitute the dataset used for training the model.

2) Model construction: There are two requirements. It is necessary to generate test cases of CAN bus messages in the correct format and ensure the differentiation of the message data content to cause abnormal states of the car. Meanwhile, the model must be constructed and test messages generated in a manner that minimizes the time overhead.

3) Input test: Test messages are generated using the GAN model and injected into the automotive CAN bus network. Then, the operating state of the car is observed.

3.2. Field division of in-vehicle CAN bus data field

Because only the average noise *z* is input to the generator and discriminator in the WGAN-GP model, the generated in-vehicle CAN messages have a significantly random nature. For CAN bus fuzz testing, the required message test cases must be generated based on the protocol format. By setting appropriate parameters in the FAMGAN model, the field format of the collected CAN bus messages is learned, and its applicability is enhanced. The in-vehicle CAN bus protocol can be considered for parsing before generating test messages; therefore, it is important to divide the message data field.

The SAE J1939 protocol is considered as an example. In SAE J1939, various data types are represented by SPNs, a group of suspect parameter numbers (SPNs) with a high correlation, and the same sending frequency is indexed by parameter group numbers (PGNs). The type, length, range, and resolution of the data are defined in the data field. Referring to the SAE J1939 protocol, the same ID of the message data field should contain various vehicle status parameters for the in-vehicle CAN bus private application layer protocol. The data field of the in-vehicle CAN bus private application layer protocol contains signal fields indicating the vehicle status parameters (abbreviated as signal fields) and a number of other types of field.

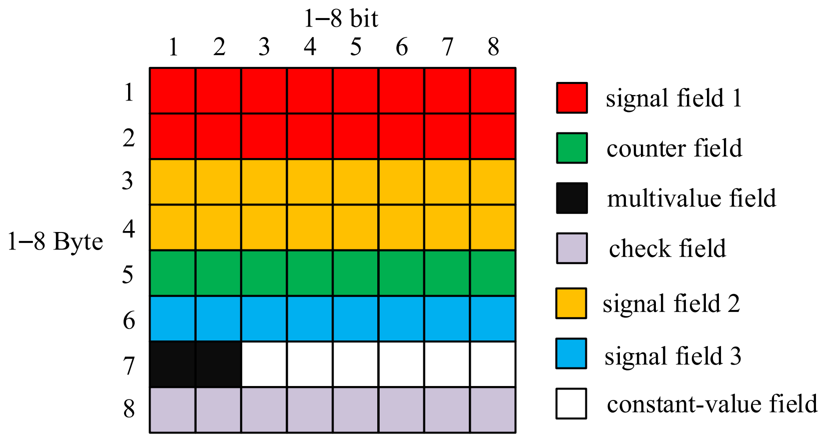


Figure 2. In-vehicle CAN message data field format division.

1) Constant number field: This is a value that does not change over time. This field may not be used, e.g., the data in the field are all 0. It can also represent a fixed value.

2) Multivalue field: This is a field in which only a few fixed values appear over time. The data type in this field may be a state type, which is used to represent several states of the signal of a car.

3) Counter field: This prevents replay attacks on the CAN bus. This type of field is typically used for critical messages.

4) Checksum field: In addition to the CRC field at the end of each CAN bus message, the data field contains other checksum fields, which are usually located at the end of the data field.

Information about the individual fields in the data field of a CAN bus message for a certain ID is shown in Figure 2. There are different types of field in the 8 bytes of the data field.

To distinguish the fields of the in-vehicle CAN bus messages, the FAMGAN model uses the field bit transition rate feature to achieve different types of field classification. The bit transition rate of each bit of the data field is defined. Assuming that there are *n* messages data named *m* of a certain ID in a period of time *t*, the number of transitions of the *j*-th bit in time *t* is defined as

$$BTN_j = \sum\nolimits_{i=2}^{n} 1, \forall j \in [1, 64] \cap m_{i,j} \neq m_{i-1,j} \tag{5}$$

Then, the *j*-th bit transition rate of the message during that time is

$$BTA_j = BFN_j / n \tag{6}$$

The bit transition rate characterizes how each bit in the message data field changes over time *t*. The larger the transition rate, the more it changes over time *t*.

In this model, the fields are divided according to the bit transition rate characteristics of the messages. Therefore, it is necessary to calculate the bit transition rate of the data fields of messages with different IDs over a period of time. The specific approach is to read the message data stream on the CAN bus and save the data. The message is then divided into several substreams according to the message IDs. Subsequently, the bit transition rate of the message is calculated. After the concept of bit transition rate is introduced, the bit transition rate characteristics of each field can be analyzed.

1) Signal field: Owing to the limitation of the cycle time, the transition rate between two consecutive bits in the physical signal field is less different, and two bits with a larger difference in the bit transition rate can be used as the basis for boundary division.

2) Counter field: Every time a message with the same ID is sent, the value of the counter field is increased by 1. Therefore, the bit transition rate of the lowest bit in the counter field is the highest. Each subsequent bit is twice as low as the bit transition rate of the previous bit.

3) Constant and multivalue fields: The bit transition rate of the constant field is 0. The bit transition rate of the multivalue field may be related to the event, while its bit transition rate is more random.

4) Check field: The bit transition rate of the check field obeys a normal probability distribution centered at 0.5 [33]. In general, the check field is located at the end of the data field. Therefore, the bit transition rate of the field can be tested for normality. If it conforms to a normal distribution, it can be considered a check field. The pseudo-code of the CAN bus data field division algorithm used to divide signal field is given below.

```
Algorithm signalfieldDivided (messageflow, Threshold)
01: bitTransition = array (len(messageflow [])) , divide = List() , divide_Flag = 0
02: for i in range (0, len(messageflow []) ):
03:     for j in range (1, len(messageflow) ):
04:         if messageflow [j][i] != messageflow[j - 1][i] :
05:             bitTransition [i] ++
06:     bitTransition [i] = bitTransition [i] / len(messageflow [])
07: for k in range(1, len(messageflow []):
08:     if bitTransition [k] / bitTransition [k - 1] < Threshold:
09:         divide.add(divide_Flag, k - 1) , divide_Flag = k
10: divide.add (divide_Flag, len(messageflow [] - 1)
11: return divide
```

As a result analyzing the field bit transition rate characteristics, a CAN bus data field division algorithm based on bit transition rate characteristics is proposed. The algorithm is divided into three stages. 1) In the variable region extraction stage, the bit transition rate of the message is calculated, and the entire data field is divided into variable and invariant regions. 2) In the initial signal field division stage, the change in the bit transition rate of two adjacent bits is examined, and an initial physical signal boundary is determined according to the predefined bit transition rate change threshold. 3) In the final field division stage, the field division of the previous stage is analyzed again. The specific field boundaries are determined according to the bit transition rate characteristics of the different types of field. Through these three stages, a simple field division of the in-vehicle CAN bus message data field can be realized. To verify the accuracy of the field division results, linear CRF, an undirected graph model, is used as the field verification model. It can make full use of the bit transition rate of adjacent bits as features for training and inference [34]. A specific field-partitioning process based on CRFs is used in this study, as shown in Figure 3.

The observation sequence X represents the field bit transition rate sequence, and the state sequence (format sequence) Y represents the corresponding attribute tag sequence. Here, X can be obtained by calculating the bit transition rate of each bit of the in-vehicle CAN message data field, and Y is obtained by prior knowledge, i.e., the known SAE J1939 protocol specification. For $\forall Y_i \in Y$, $Y_i = y \in \{0,1\}$. If $\exists Y' \in Y$ belongs to the field boundary and is the first end of the field, then $Y_i' = 1$; vice versa, $Y_i' = 0$.

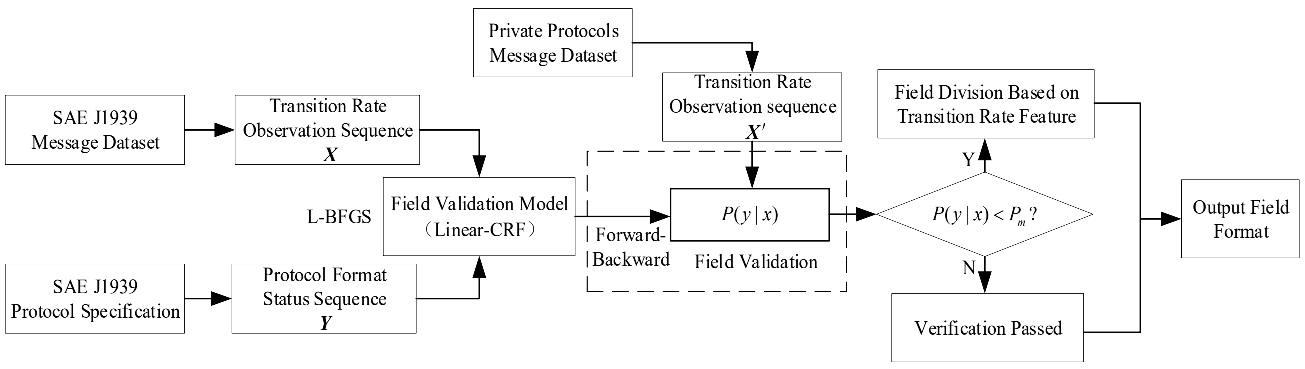

Private Protocols Message Dataset
Transition Rate Observation sequence X'
Field Division Based on Transition Rate Feature
SAE J1939 Message Dataset
Transition Rate Observation Sequence X
L-BFGS
Field Validation Model (Linear-CRF)
P(y|x)
Forward-Backward
Field Validation
Y
P(y|x) < P_m ?
N
Output Field Format
SAE J1939 Protocol Specification
Protocol Format Status Sequence Y
Verification Passed

Figure 3. Specific implementation process based on linear-CRF field division algorithm.

Here, X and Y are sequences generated by random variables—in the case of a given sequence of observed CAN bus message bit transition rate, the output field boundary labeling attribute sequence Y completes the division of the protocol field. In addition, $P(Y \mid X)$ is a linear chain random field. Then, conditional probability $P(y \mid x)$, where X takes the value x and Y takes the value y, is given by

$$\frac{1}{Z(x)}\exp(\sum_{i,k}\lambda_k t_k(y_{i-1},y_i,x,i)+\sum_{i,1}\mu_l s_l(y_i,x,i)) \tag{7}$$

where t_k and s_l are the transfer and state feature functions, respectively, and k and l are the number of feature functions. Feature functions t_k and s_l represent the constraint relationships among and within the fields, λ_k and μ_l are the corresponding weights, and $Z(x)$ is the normalization factor. According to the obtained conditional probability formula of linear CRF, verification of the fields can be achieved by providing a sequence of private protocol transition rates X' and finding the conditional probability of X'. Private protocol refers to the CAN bus application layer protocol. For in-vehicle CAN bus messages, the application layer protocol specification for the Data field is non-public. The protocol format specification is proprietary to the original vehicle manufacturer and varies by brand, model and year. It is usually stored in a .dbc file format. To improve the convergence speed, the construction process of the linear-CRF model is based on the limited-memory Broyden-Fletcher-Goldfarb-Shanno learning algorithm.

In the field verification phase, observation sequence X is a bit transition rate sequence generated by a private protocol message, and state sequence Y is obtained from the SAE J1939 protocol. The validation process of the in-vehicle CAN message field involves calculating the conditional probability of the observation sequence X for the state sequence Y. If the conditional probability is greater than a certain threshold value, it means that the field validation is passed, which is in accordance with the original SAE J1939 field format. The field validation can be achieved by calculating the conditional probability through the forward–backward algorithm. For each state observation marker $i=1,2\cdots,n+1$, the forward vector $\alpha_i(x)$ and backward vector $\beta_i(x)$ are defined, and the conditional probability $P(Y_i=y_i \mid x)$, $P(Y_{i-1}=y_{i-1},Y_i=y_i \mid x)$ can be calculated by

$$P(Y_i=y_i \mid x)=\alpha_i^T(y_i \mid x)\beta_i(y_i \mid x)/Z(x) \tag{8}$$

$$P(Y_{i-1}=y_{i-1},Y_i=y_i \mid x)=\alpha_{i-1}^T(y_{i-1} \mid x)M_i(y_i,y_{i-1} \mid x)\beta_i(y_i \mid x)/Z(x) \tag{9}$$

Then, the conditional probability of the observation series is given by

$$P(y \mid x)=P(Y_1=y_1 \mid x)\prod_{i=2}^{n}P(Y_{i-1}=y_{i-1},Y_i=y_i \mid x) \tag{10}$$

The threshold P_{m} is set. If $P(y \mid x)<P_{\mathrm{m}}$, the observed sequence has a low probability of correctness of field division and does not pass the field validation.

3.3. Improving generating message specificity with date field association mining

After the in-vehicle CAN bus messages have been divided, the results are input to the generator as a constraint to obtain a message instance that is more in line with the CAN communication protocol. However, although the message instances generated by this method are similar to real messages in format, the data content is not considered reasonable during the generation process. However, this also improves the authenticity of the generated message data content.

The Apriori algorithm is a classical association rule mining algorithm that uses iterative layer-by-layer search to find the relationship of item sets in the dataset to form rules. The process consists of concatenation and pruning. Although the Apriori algorithm performs a global search on the dataset, leading to a high consumption of computational resources, the purpose of using this algorithm in this study was to discover field relationships and constrain the message instance generation conditions of FAMGAN.

For association mining among message fields, the objects for association mining of message fields required by the FAMGAN model are divided into value type and change type, according to the characteristics of in-vehicle CAN bus messages.

1) The value type of the association mining object indicates the association analysis of values corresponding to the fields in the area of interest within a single in-vehicle CAN bus message. For example, if a certain ID of an in-vehicle CAN bus message contains three fields (door switch status, window switch status, and door indicator status), it is easy to find that there must be some association rules between the two fields of door switch status and door indicator status; the two fields of door switch status and window switch status are not directly related. The association rules of these fields can be inferred quickly and accurately. However, more association rules between fields must be mined through algorithms, and field values must be analyzed as association mining objects.

2) When the association mining object is the change type, it indicates the association analysis of the changes in the corresponding values of the fields in the area of interest within the continuous sequence of the in-vehicle CAN bus messages. Real in-vehicle CAN bus messages are continuously distributed over time, and the value changes of some fields in the message data field may have such characteristics as consistency, so using the change type as the association mining object is required.

The field association mining process is similar for value and change types. However, there are some differences in the data preprocessing. Because the fields of in-vehicle CAN message data contain a large number of continuous-type variables, they must be discretized. For value-type field association mining, field values can be classified and tagged according to the value range and granularity requirements, the value tag distribution of all fields can be counted, and the degree of support and confidence among fields can be calculated. For change-type field association mining, messages should be sorted according to the time sequence. For each message, the difference between the field value of the message and the corresponding field value of its previous message is calculated. By marking the positive and negative of the difference, the degree of support and confidence among fields can be calculated. Figure 4 shows the process of calculating support and confidence using the Apriori algorithm after processing the variation-type fields.

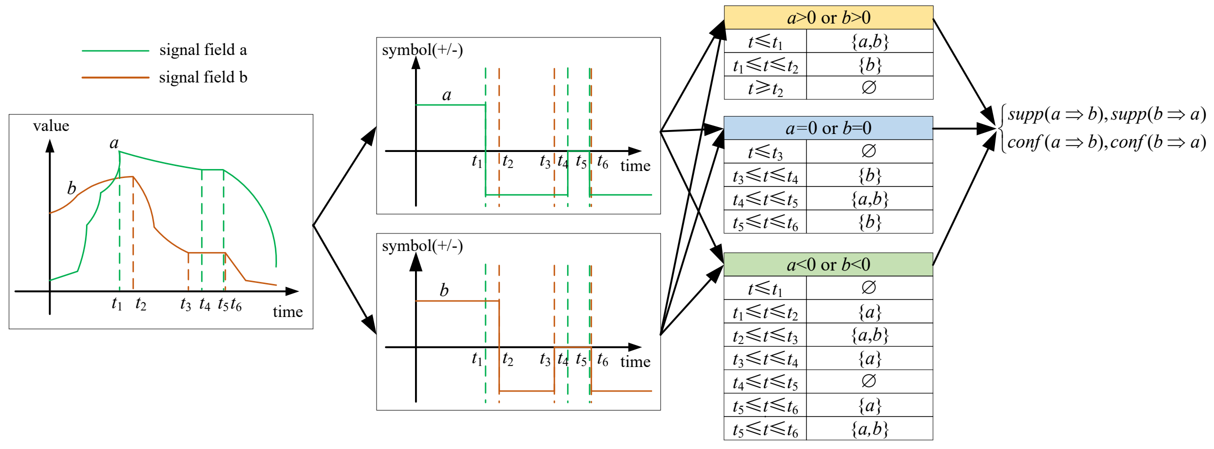

signal field a
signal field b
value
a
b
t_1 t_2 t_3 t_4 t_5 t_6
time
symbol(+/-)
a>0 or b>0
$t \leq t_1$ {a,b}
$t_1 \leq t \leq t_2$ {b}
$t \geq t_2$ ∅
a=0 or b=0
$t \leq t_3$ ∅
$t_3 \leq t \leq t_4$ {b}
$t_4 \leq t \leq t_5$ {a,b}
$t_5 \leq t \leq t_6$ {b}
a<0 or b<0
$t \leq t_1$ ∅
$t_1 \leq t \leq t_2$ {a}
$t_2 \leq t \leq t_3$ {a,b}
$t_3 \leq t \leq t_4$ {a}
$t_4 \leq t \leq t_5$ ∅
$t_5 \leq t \leq t_6$ {a}
$t_5 \leq t \leq t_6$ {a,b}
$supp(a \Rightarrow b), supp(b \Rightarrow a)$
$conf(a \Rightarrow b), conf(b \Rightarrow a)$

Figure 4. Apriori-based calculation of support and confidence among fields of CAN messages.

After the field association rules for in-vehicle CAN data fields have been determined, the field constraints formed by the abstraction of the association rules can be entered into the generator and discriminator of the FAMGAN model. The overall structure of FAMGAN is shown in Figure 5.

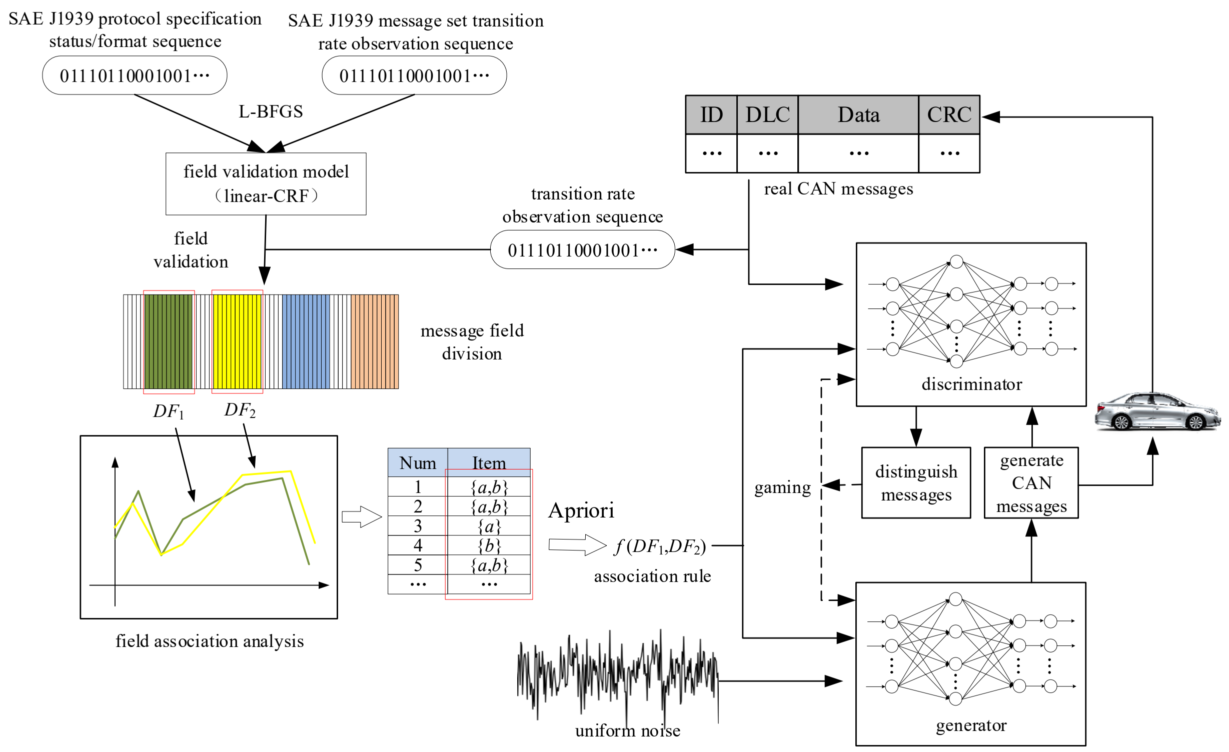


Figure 5. Structure of FAMGAN.

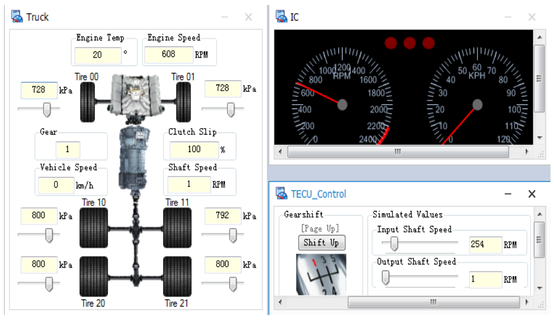
4. Simulation and experiment

4.1. Simulation environment

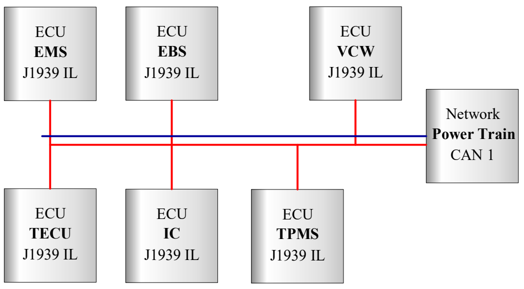
To verify the accuracy of the data field division in FAMGAN, a simulation platform was established using CANoe software. CANoe 12.0 SP4 simulation software was used to establish the in-vehicle CAN bus network communication simulation platform and simulate the nodes on the CAN network to send and receive messages. The message protocol is based on the J1939 protocol and exports CAN bus messages to generate the datasets.

1) Software environment

The simulated in-vehicle CAN communication network uses the CANoe J1939 System Demo, which simulates the working state of each part of the CAN communication network and the message sending situation when a vehicle is running. The Power_train network inside the demo simulates the power CAN bus of the vehicle, including the engine, gears, brakes, and other major control units. Figure 6 shows the in-vehicle CAN network CANoe simulation platform and topology of the two simulated CAN bus subnets Power_train.



(a) CAN network simulation control interface



(b) Power_train bus topology

Figure 6. In-vehicle CAN network CANoe simulation platform.

For the two simulated CAN bus subnets, the Power_train topology shown in Figure 6(b) contains seven ECUs and 29 different messages. Some of the simulated network messages are listed in Table 1.

Table 1. Examples of messages used in the experiment.

Seq	PGN	Description	Seq	PGN	Description
1	61442	ETC1_TECU	5	61445	ETC2_TECU
2	61443	EEC2_EMS	6	61449	VSC2_EBS
3	61444	EEC2_EMS	7	64964	EBC5_EBS
4	61441	EEC1_EBS	…	…	…

2) Acquisition of datasets

CAN bus messages are generated by simulating the driving process of a car in the simulation control interface. The SAE J1939 application layer protocol specification of the messages is stored in the powertrain.dbc file. Part of the CAN bus message dataset is generated using the SAE J1939 protocol specification. At the same time, to differ from the original protocol specification to verify the validity of FAMGAN, some of the contents in the powertrain.dbc file are modified to generate a private message protocol based on the J1939 protocol.

4.2. Message field division and association verification

In this study, the message with the parameter group number PGN 614,444 (Engine Controller for

Electronics) was selected for verification. According to the definition of the protocol specification in SAE J1939, the field format of PGN 614,444 is defined as shown in Table 2. This table is then introduced as prior knowledge.

Table 2. Format definition of PGN 61444.

Field position (bytes)	Meaning	SPN	Field position (bytes)	Meaning	SPN
1.1–1.4	Engine torque mode	899	4.1–5.8	Engine speed	190
2.1–2.8	Demand engine-torque percentage	512	6.1–6.8	Control device source address	1483
3.1–3.8	Actual engine-torque percentage	513	7.1–7.4	Engine starter mode	1675

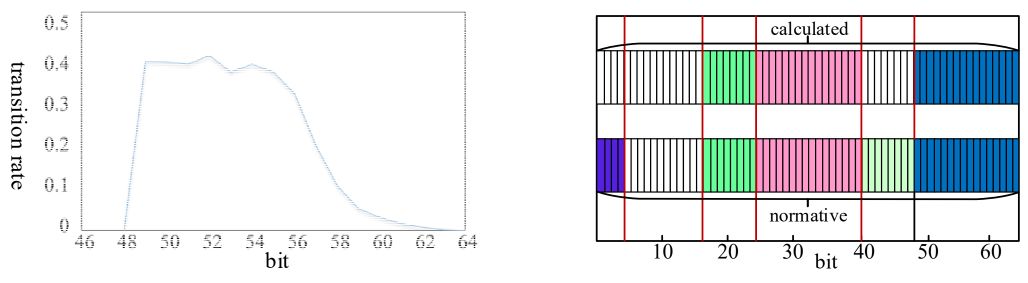
PGN 61,444 is defined and arranged in the powertrain.dbc file, and each suspect parameter SPN is also defined according to the SAE J1939 protocol specification. To verify the results of the field division, the positions of the brake, gear, and throttle are continuously adjusted randomly to generate the message data and calculate the transition rate of each bit of the PGN 61,444 messages. By continuously repeating the above steps, a dataset of the 64-bit transition rate of PGN 61,444 messages is finally generated. The linear-CRF field verification model is constructed using the transition rate as the observed variable and the division of signal parameters in the powertrain.dbc file as the state variable. The bit transition rate under the SPN 513 field is between 0 and 0.45, and the bit transition rate under the SPN 190 field is between 0.01 and 0.5. When training the field validation model, the bit transition rate observation sequence is divided into 11 levels as $t_0, t_{0.05}, \cdots, t_{0.5}$. For example, if the bit transition rate is 0.256, this is marked as $t_{0.25}$. At the end of the training phase, the conditional probability of field verification is obtained from 0.721 to 0.987 by substituting the data in the training. Therefore, the threshold value of field verification is set to 0.7. When it is lower than this threshold value, field verification fails.

The powertrain.dbc file is remodified, the definitions of SPN 512 and SPN 1675 are deleted, the signal of wheel speed (SPN 84) is added in the last byte, the factor value of SPN 191 is changed to 0.13, and it is offset to 50. The modified powertrain.dbc file is applied to generate message data and calculate the transition rate of each bit of the PGN 61,444 message.

Table 3. SPN 513,190-field validation results.

Name	Number of Bits	Observed Sequence Description	State Sequence	Conditional Probability	Name	Number of Bits	Observed Sequence Description	State Sequence	Conditional Probability
SPN 513	16–19	$t_0 / t_{0.4} / t_{0.25} / t_{0.2}$	0/1/0/0	0.875	SPN 190	24–27	$t_0 / t_{0.45} / t_{0.4} / t_{0.4}$	0/1/0/0	0.841
	17–20	$t_{0.25} / t_{0.2} / t_{0.1} / t_{0.1}$	0/0/0/0	0.898		25–28	$t_{0.45} / t_{0.4} / t_{0.4} / t_{0.4}$	1/0/0/0	0.963
	18–21	$t_{0.2} / t_{0.1} / t_{0.1} / t_{0.05}$	0/0/0/0	0.938		26–29	$t_{0.4} / t_{0.4} / t_{0.4} / t_{0.4}$	0/0/0/0	0.806
	20–23	$t_{0.1} / t_{0.05} / t_{0.05} / t_{0.05}$	0/0/0/0	0.861		29–32	$t_{0.4} / t_{0.45} / t_{0.45} / t_{0.4}$	0/0/0/0	0.913
	21–24	$t_{0.05} / t_{0.05} / t_{0.05} / t_0$	0/0/0/0	0.912		32–35	$t_{0.4} / t_{0.35} / t_{0.3} / t_{0.2}$	0/0/0/0	0.948
	23–26	$t_{0.05} / t_0 / t_{0.45} / t_{0.4}$	0/0/1/0	0.804		36–39	$t_{0.25} / t_{0.2} / t_{0.1} / t_{0.05}$	0/0/0/0	0.904
						37–40	$t_{0.2} / t_{0.1} / t_{0.05} / t_{0.05}$	0/0/0/0	0.914
						38–41	$t_{0.1} / t_{0.05} / t_{0.05} / t_{0.4}$	0/0/0/1	0.793

The SPN parameters in the first, second, and sixth bytes did not change during message acquisition. Only the third byte (SPN 513), the fourth and fifth bytes (SPN 190), and last two bytes (SPN 84) were analyzed. SPN 513 is located in the third byte of the data field and is distributed in bits 17–24. SPN 190 is located in the fourth and fifth bytes of the data field and is distributed in bits 25–40. Field validation is performed using CRFs for the SPN 513 and SPN 190 fields. Choosing the bit transition rate of every 4 bits in the field as the observed sequence, conditional probabilities can be obtained under the CRF model, as shown in Table 3

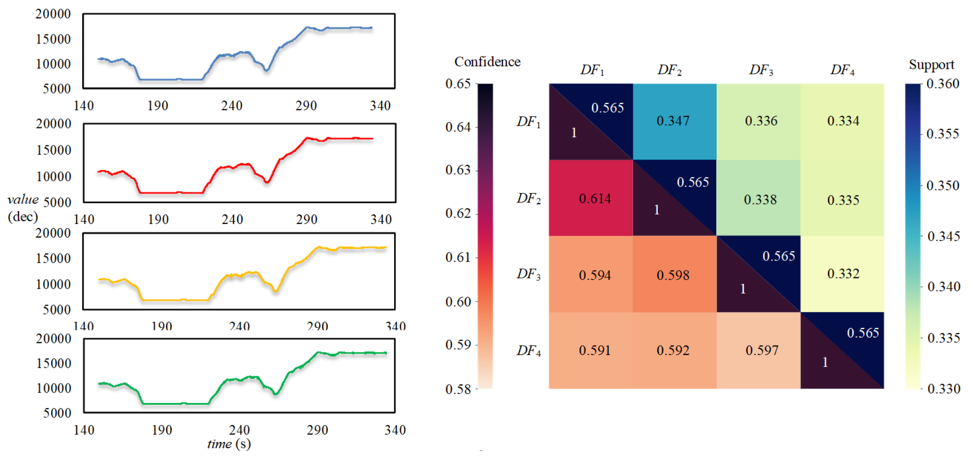


(a) Message SPN 84 field transition rate (b) Comparison of division results

Figure 7. PGN 61,444 message division verification.

As shown in Table 3, by setting the boundary state values of the SPN 513 and SPN 190 fields, the conditional probability of the bit transition rate observation sequence for every 4 bits under the CRF model is above the threshold. Therefore, the validation of the fields is passed, indicating that the modified protocol specification also contains SPN 513 and SPN 190 fields.

The bit transition rate of the seventh byte in the message is 0, and the bit transition rate from bits 49 to 64 is not 0. The bit transition rate of the last 2 bytes is shown in Figure 7(a). The field format shown in Figure 7(b) is obtained according to the field division algorithm in FAMGAN. The results show that the final field division results conform to the field format after modification to the SAE J1939 protocol.



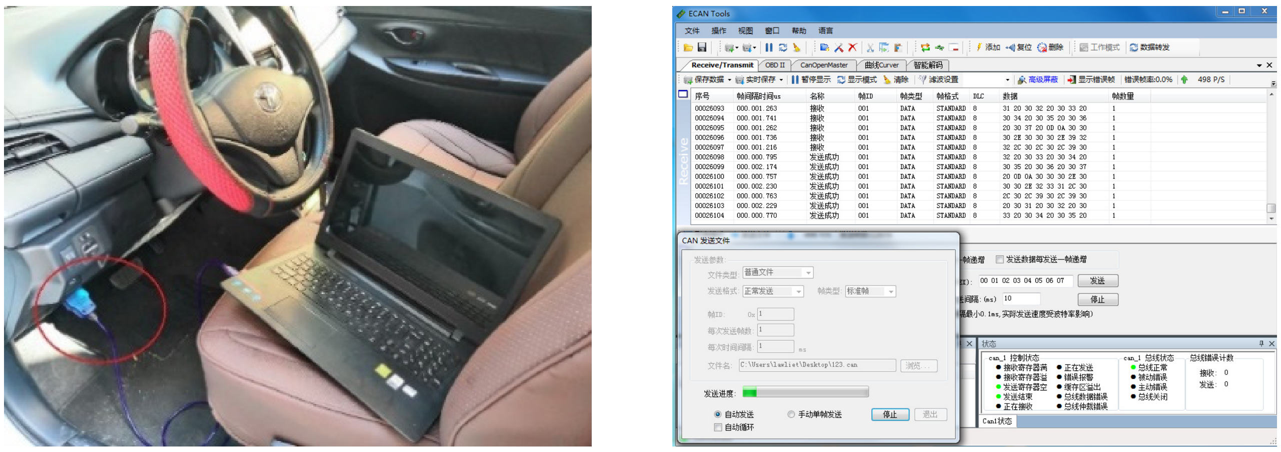
(a) 150–330-s signal field data change (b) 150–330-s signal field data change correlation

Figure 8. Correlation analysis of the change in the value of the signal field with message ID 0xAA.

To analyze the correlation of the data fields of in-vehicle CAN messages, messages with ID 0xAA were taken as an example for the simulation. The test host hardware for the test case generation method was an Intel Core i7-4720HQ with a 3.00-GHz main frequency. The operating system was a Windows 7 64 bit. The compiler was Python 3.9. After analyzing the field division and meaning of the data field of the in-vehicle CAN message with ID 0xAA, it was found that its 8-byte data field consisted of four 2-byte signal fields, and their meanings were related to the speed of the four wheels of the car. The field extraction of the in-vehicle CAN message with ID 0xAA and acquisition time of 150–330 s yielded the four signal field data changes with time, as shown in Figure 8(a). The correlation analysis of the field data changes yielded the correlation results of the field changes, as shown in Figure 8(b), and the algorithm running time is 11.33 s. The change trend of the four signal fields had a more obvious consistency. The FAMGAN generator generated the message instance with ID 0xAA. The changes in the four signal fields should have a high consistency if the test message sequence with time relation is generated.

4.3. Model-based generating message execution

The USBCAN-OBD device and OBD-II interface of the vehicle were used to obtain the CAN bus messages. The messages were recorded and exported by the host software ECanTools, and the message dataset was generated for test case generation. Each data record in the dataset included the message reception interval time, message ID, DLC, and data in the data field. The test cases were then generated using the FAMGAN model and converted in order to batch the .can files for storage. The CAN bus messages were sent to the in-vehicle CAN bus for fuzz testing through the USBCAN-OBD device using ECanTools. The process of fuzz testing of the vehicle through the OBD interface is shown in Figure 9.


ECAN Tools
文件 操作 视图 窗口 帮助 语言
添加 复位 删除 工作模式 数据转发
Receive/Transmit OBD II CanOpenMaster 曲线Curver 智能解码
保存数据 实时保存 暂停显示 显示模式 清除 滤波设置 高级屏蔽 显示错误帧
Receive
CAN 发送文件
发送 停止 浏览... 停止 退出
状态
Can1状态

错误帧率:0.0% 498 P/S

序号	帧间隔时间us	名称	帧ID	帧类型	帧格式	DLC	数据	帧数量
00026093	000.001.263	接收	001	DATA	STANDARD	8	31 20 30 32 20 30 33 20	1
00026094	000.001.741	接收	001	DATA	STANDARD	8	30 34 20 30 35 20 30 36	1
00026095	000.001.262	接收	001	DATA	STANDARD	8	20 30 37 20 0D 0A 30 30	1
00026096	000.001.736	接收	001	DATA	STANDARD	8	30 2E 30 30 30 2E 39 32	1
00026097	000.001.216	接收	001	DATA	STANDARD	8	32 2C 30 2C 30 2C 39 30	1
00026098	000.000.795	发送成功	001	DATA	STANDARD	8	32 20 30 33 20 30 34 20	1
00026099	000.002.174	发送成功	001	DATA	STANDARD	8	30 35 20 30 36 20 30 37	1
00026100	000.000.757	发送成功	001	DATA	STANDARD	8	20 0D 0A 30 30 30 2E 30	1
00026101	000.002.230	发送成功	001	DATA	STANDARD	8	30 30 2E 32 33 31 2C 30	1
00026102	000.000.763	发送成功	001	DATA	STANDARD	8	2C 30 2C 39 30 2C 39 30	1
00026103	000.002.229	发送成功	001	DATA	STANDARD	8	20 30 31 20 30 32 20 30	1
00026104	000.000.770	发送成功	001	DATA	STANDARD	8	33 20 30 34 20 30 35 20	1

发送参数:
文件类型: 普通文件
发送格式: 正常发送　帧类型: 标准帧
帧ID: 0x 1
每次发送帧数: 1
每次时间间隔: 1 ms
文件名: C:\Users\lawliet\Desktop\123.can
发送进度:
[✓ 自动发送]　手动单帧发送
自动循环

帧递增　发送数据每发送一帧递增
EX): 00 01 02 03 04 05 06 07
送间隔: (ms) 10
隔最小0.1ms, 实际发送速度受波特率影响)

can_1 控制状态: ● 接收寄存器满 ● 接收寄存器溢 ● 发送寄存器空 ● 发送结束 ● 正在接收 ● 正在发送 ● 错误报警 ● 缓存区溢出 ● 总线数据错误 ● 总线仲裁错误
can_1 总线状态: ● 总线正常 ● 被动错误 ● 主动错误 ● 总线关闭
总线错误计数: 接收: 0 发送: 0

(a) USBCAN-OBD device　　(b) Fuzz testing with batch .can file

Figure 9. CAN bus fuzz testing through OBD interface.

The deep learning framework TensorFlow was used to build the FAMGAN. Model-generated message execution experiments were performed on a live vehicle to analyze the message execution results. Uniform noise and field constraints were input to the generator, while real message data and field constraints were input to the discriminator for FAMGAN model training. The curve of the training process parameter *L* with the number of training is shown in Figure 10. It can be seen from the figure that when the learning rate is 0.001, the parameter L converges faster and its final value is smaller. Therefore, the learning rate is set to 0.001 in this paper.

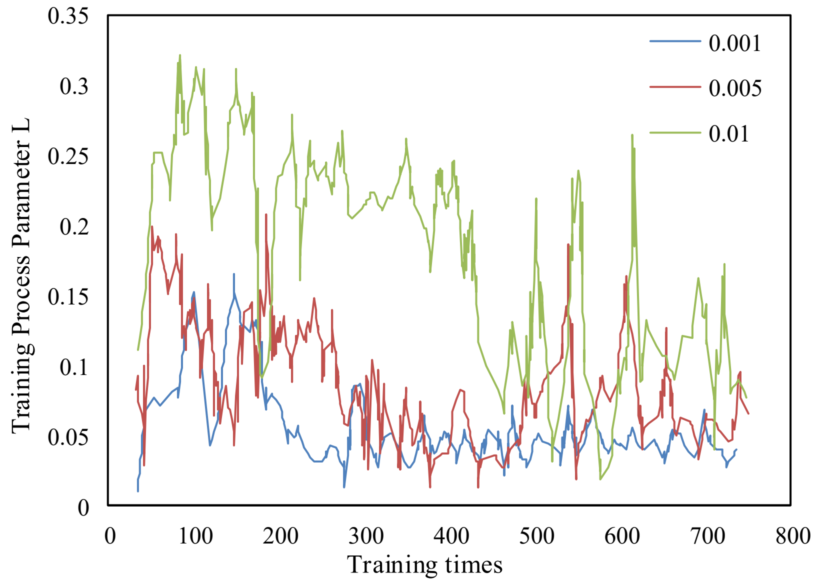


Figure 10. The training process parameter L varies with the number of training times.

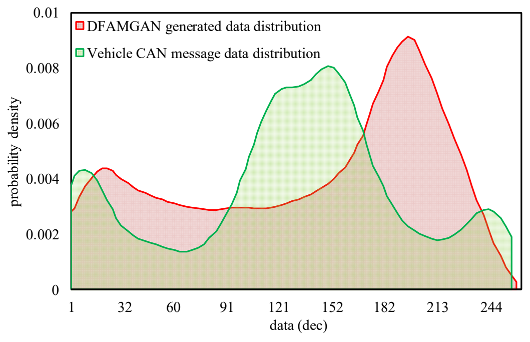
To analyze the distribution of FAMGAN-generated data, the probability density distribution of the second byte of ID 0x1□4 message data is shown in Figure 11, the average training time of FAMGAN algorithm is 3.97 s. The distribution of the generated data is closer to the distribution of the real data as the number of trainings increases.

The partial message data generated by the FAMGAN model with ID 0x 0x1□4 are listed in Table 4. The values of the third, fourth, and sixth bytes of the message data field with ID 0x 0x1□4 remain unchanged, while the values of the remaining fields are mutated only.

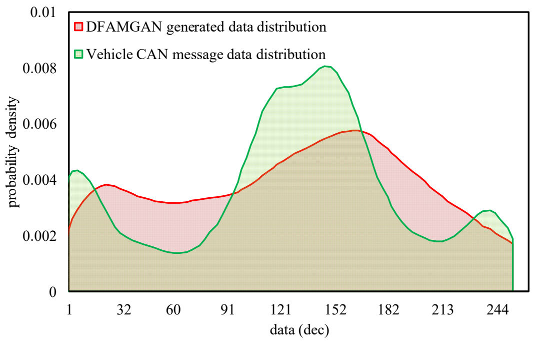
Table 4. Partial ID 0x1□4 messages generated by FAMGAN-based fuzz-testing method.

Byte	1	2	3	4	5	6	7	8
	05	04	1B	08	30	00	C6	F7
	04	FC	1B	08	30	00	F0	C5
	04	FA	1B	08	30	00	C7	12
	05	00	1B	08	30	00	E7	CB
	05	06	1B	08	30	00	C8	76

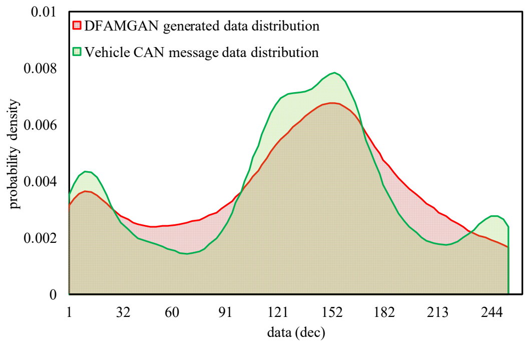
The fuzz-testing cases were sent to the CAN bus of the vehicle through the OBD interface to monitor the abnormal status of the vehicle and record abnormal messages. Figure 12 shows the abnormal status of the vehicle during the execution of fuzz-testing cases. Two fuzz-testing methods were used to generate 10,000 test cases for each message ID, and the training time of FAMGAN model is about 2380 s. The messages generated by both fuzz-testing methods affect the anomalous state of the vehicle. The fuzz-testing methods based on field weights and the FAMGAN model triggered nine and eleven anomalous car states respectively, and the inference times are 0.161 and 5.247 s respectively.



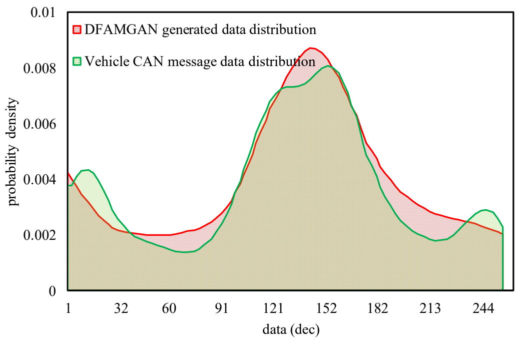
(a) 100th training



(b) 200th training

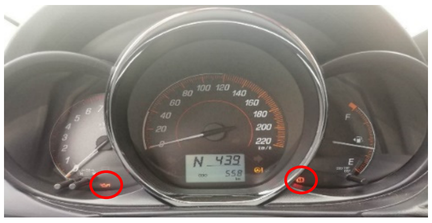


(c) 500th training

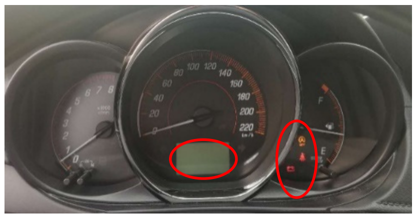


(d) 600th training

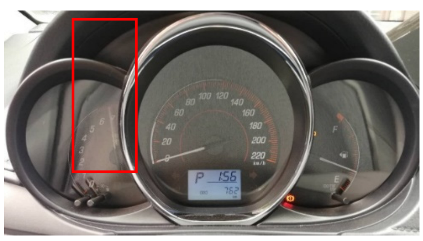
Figure 11. Comparison of the probability density distribution of the second byte of ID 0x1□4.



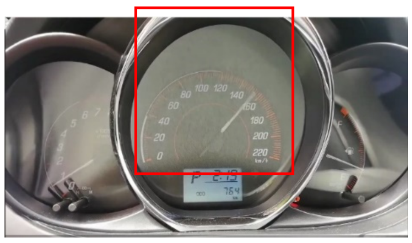
(a) Fault indication abnormal situation 1



(b) Fault indication abnormal situation 2



(c) Abnormal display of RPM dashboard



(d) Abnormal display of speed dashboard

Figure 12. Automotive equipment abnormal display.

As shown in Figure 12(a), after sending the test case with messages with IDs 0x□2 and 0x□4, the low oil pressure warning light, electric steering system warning light, and brake system light on the instrument panel remained constant. This means that the telegram triggered a false alarm, whereas the vehicle did not actually malfunction. As shown in Figure 12(b), after the messages with IDs 0x□3 and 0x1□F were sent, the smart start–stop canceled the indicator light on the instrument panel, the charging system warning light was always on, the front seat belt reminder light flashed, and the digital display on the instrument panel disappeared.

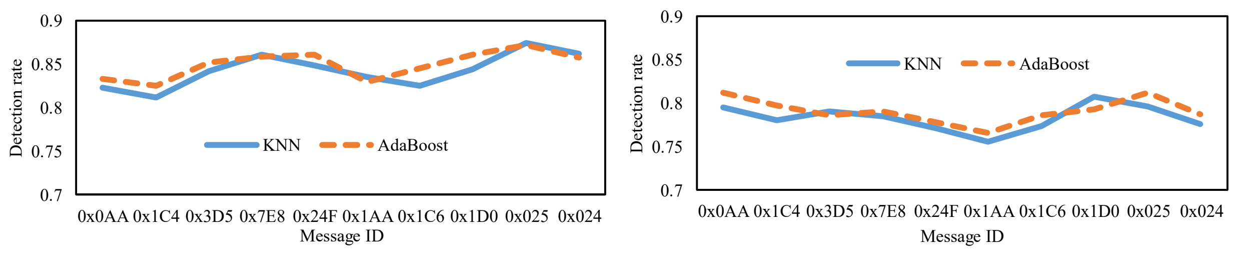
As shown in Figures 12(c),(d), after the test cases with message IDs 0x□4 and 0x□C4 were sent, respectively, the RPM and vehicle speed on the instrument panel changed. The RPM reached 7000 r/min, and the driving speed reached 156 km/h. The car was not actually in a driving condition.

4.4. In-vehicle CAN bus intrusion detection experiment

The performance analysis of the intrusion detection algorithm based on fuzz testing is divided into the following four steps: message data collection, data preprocessing, classifier training, and performance verification of the classifier based on fuzz testing. Normal and abnormal CAN bus message data were used to extract the features of the message data. The message data were trained using a machine learning classifier. Then, test cases were generated using the fuzz-testing algorithm and input to the machine learning classifier for intrusion detection. Subsequently, the performance of the intrusion detection algorithm was analyzed. During the training phase of the intrusion detection classifier, different intrusion detection algorithms were used to train the CAN bus message dataset to build an intrusion detection classifier. In this study, the k-nearest neighbors (KNN) and AdaBoost algorithms were used to construct the classifier.

The 250,000 real vehicle messages collected were used as the normal message dataset for testing, and the messages were classified and saved according to their IDs. Anomalous messages were generated for training the classifier of the intrusion detection algorithm by manual random synthesis. The experimental environment was as follows. The host hardware was an Intel Core i7-4720HQ with a main frequency of 3.00 GHz. The operating system used was a Windows 7 64-bit system.

During the experiment, 3000 messages for each ID were used as the training set, including 2000 normal messages and 1000 randomly generated abnormal messages. The classifiers based on the KNN and AdaBoost algorithms were implemented by programming software and related toolboxes, and the detection rate of the classifiers was then calculated by fuzz testing.



(a) Fuzz testing based on field weights (b) Fuzz testing based on FAMGAN.

Figure 13. Detection rate results of KNN and AdaBoost using different fuzz-testing algorithms.

Ten of these CAN messages with different IDs were selected for training the classifier. CAN message test cases were generated by performing fuzz testing to calculate the detection rate of the intrusion detection algorithm. Figure 13 shows the comparison results of the detection rates based on the KNN and AdaBoost algorithms while using different fuzz-testing algorithms.

The results of the detection rate calculation of the intrusion detection algorithm using the field weight-based fuzz-testing method reveal that the average detection rate of the KNN algorithm was 84.31% and that of the AdaBoost algorithm was 85.06%. The results of the detection rate calculation of the intrusion detection algorithms using the FAMGAN-based fuzz-testing method show that the average detection rate of the KNN algorithm was 78.30% and that of the AdaBoost algorithm was 79.83%. For CAN bus messages with different IDs, in general, both classifiers have lower detection rates for test cases generated by the fuzz-testing method based on the FAMGAN model generation. This is because this type of test method generates messages that are similar to the original message dataset, lowering the detection rate. In terms of the detection rate, AdaBoost is more effective in detecting anomalous messages.

5. Conclusions

An efficient generation model, FAMGAN, was proposed for in-vehicle CAN messages. In this model, the bit transition rate and CRF model are used to divide the message field, whereas the Apriori algorithm is used to analyze the CAN message field relationships. The algorithms mentioned above improve the compliance and reasonableness of the generated messages using FAMGAN. WGAN-GP was used to achieve large-scale generation of in-vehicle CAN bus messages, input noise, real messages, and field constraints to the generator and discriminator to obtain messages with in-vehicle CAN network specificities. Comprehensive simulations and experiments were conducted to compare intrusion detection algorithms on real vehicles with a code-built intrusion detection algorithm. The results show that the FAMGAN model can generate CAN messages that trigger vehicle anomalies in in-vehicle CAN networks. FAMGAN can also be applied to machine learning-based intrusion detection algorithm evaluation. However, FAMGAN has some limitations. Methods of generating a series of CAN messages with time-dependent characteristics in a batch and improving the efficiency of anomalous message generation should be the focus of future research.

Acknowledgments

This study is supported by "Project for the Development and Application of Safety Testing and Verification Platform for Industrial Robots" under Grant TC200H01Q and Heilongjiang Touyan Innovation Team Program under Grant AUEA5640202520-09.

Conflict of interest

The authors declare there is no conflict of interests.

References

1. A. Neffati, A. Marzouki, Local energy management in hybrid electrical vehicle via fuzzy rules system, *AIMS Energy*, **8** (2020), 421–437. https://doi.org/10.3934/energy.2020.3.421

2. Y. Ma, Z. Wang, H. Yang, Artificial intelligence applications in the development of autonomous vehicles: A survey, *IEEE/CAA J. Autom. Sin.*, **7** (2020), 315–329. https://doi.org/1109/JAS.2020.1003021
3. Z. Feng, M. He, B. Li, Research on car information security attack and protection technology, *J. Cyber Secur.*, **2** (2017), 1–14. https://doi.org/10.19363/j.cnki.cn10-1380/tn.2017.04.001
4. H. Kong, T. Kim, M. Hong, A security risk assessment framework for smart car, in *International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*, (2016), 102–108, https://doi.org/10.1109/IMIS.2016.42
5. R. Solaiman, T. Kherbek, A. Ahmad, Defining a new method to set certainty factors to improve power systems prognosis with fuzzy petri nets, *AIMS Energy*, **8** (2020), 686–700. https://doi.org/10.3934/energy.2020.4.686
6. K. Nohara, K. Asahi, M. Yoshikawa, Study of threat for automotive embedded system by Trojan virus, in *2014 IEEE 3rd Global Conference on Consumer*, (2014), 405–406, https://doi.org/10.1109/GCCE.2014.7031151
7. S. Abbott-McCune, L. A. Shay, Intrusion prevention system of automotive network CAN bus, in *2016 IEEE International Carnahan Conference (ICCST)*, (2016), 1–8, https://doi.org/10.1109/CC ST.2016.7815711
8. B. Marco, Design and implementation of an intrusion detection system (IDS) for in-vehicle networks, Master thesis, University of Gothenburg, 2017.
9. L. Kang, H. Shen, Abnormal message detection for CAN bus based on message transmission behaviors, in *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, (2020), 432–441. https://doi.org/10.1109/ICDCS47774.2020.00041
10. H. Markus, S. Thilo, D. Katharina, U. Holger, CANet: An unsupervised intrusion detection system for high dimensional CAN bus data, *IEEE Access*, **8** (2020), 58194–58205. https://doi.org/10.1109/ACCESS.2020.2982544
11. S. Lokman, A. Othman, M. Abu-Bakar, Intrusion detection system for automotive Controller Area Network (CAN) bus system: a review, *EURASIP J. Wirel. Commun. Netw.*, **1** (2019), 184–200. https://doi.org/10.1186/s13638-019-1484-3
12. C. Miller, C. Valasek, Adventures in automotive networks and control units, 2013. Available from: https://defcon.org/html/defcon-21/dc-21-speakers.html.
13. C. Valasek, C. Miller, Who's behind the wheel? Exposing the vulnerabilities and risks of high tech vehicles, 2015. Available from: https://icitech.org/wp-content/uploads/2015/09/ICIT-Brief_Whos-Behind-the-Wheel_Car-Hacking1.pdf.
14. A. Greenberg, The jeep hackers are back to prove car hacking can get much worse, 2016. Available from: https://www.wired.com/2016/08/jeep-hackers-return-high-speed-steering-acceleration-hacks/.
15. T. Huang, J. Zhou, A. Bytes, ATG: An attack traffic generation tool for security testing of in-vehicle CAN bus, *ACM Int. Conf. Proc. Ser.*, (2018), 1–6, https://doi.org/10.1145/3230833.3230843
16. H. Olufowobi, C. Young, J. Zambreno, G. Bloom, SAIDuCANT: Specification-based automotive intrusion detection using Controller Area Network (CAN) timing, *IEEE Trans. Veh. Technol.*, **69** (2020), 1484–1494, https://doi.org/10.1109/TVT.2019.2961344

17. X. Zhou, R. Jiang, M. Tian, H. Qu, H. Zhang, Temperature-sensitive Fingerprinting on ECU Clock Offset for CAN Intrusion Detection and Source Identification, in *Proceedings of the ACM Turing Celebration Conference-China*, (2020), 89–94, https://doi.org/10.1145/3393527.3393543
18. D. Li, M. Tian, R. Jiang, K. Yang, Exploiting temperature-varied voltage fingerprints for in-vehicle CAN intrusion detection, in *ACM Turing Award Celebration Conference-China (ACM TURC 2021)*, (2021), 116–120, https://doi.org/10.1145/3472634.3472662
19. W. Jiang, Z. Li, K. Tan, An adaptive intrusion detection algorithm for in-vehicle CAN bus based on periodicity of message, *J. Phys. Conf. Ser.*, **1748** (2021), 1–9, https://doi.org/10.1088/1742-6596/1748/3/032023
20. R. Islam, M. K. Devnath, M. D. Samad, S. M. Kadry, GGNB: Graph-based Gaussian naive Bayes intrusion detection system for CAN bus, *Veh. Commun.*, **33** (2021), 69–79. https://doi.org/10.1016/j.vehcom.2021.100442
21. R. Islam, R. U. D. Refat, S. M. Yerram, H. Malik, Graph-based intrusion detection system for Controller Area Networks, *IEEE Trans. Intell. Transp. Syst. (T-ITS)*, **23** (2022), 1727–1736, https://doi.org/10.1109/TITS.2020.3025685
22. K. Tan, Z. Li, W. Jiang Y. Guan, W. Tong, In-vehicle CAN bus anomaly detection algorithm based on linear chain condition random field, in *2019 IEEE 19th International Conference on Communication Technology (ICCT)*, (2019), 1153–1159, https://doi.org/10.1109/ICCT46805.2019.8947020
23. Y. He, Z. Jia, M. Hu, C. Cui, Y. Cheng, Y. Yang, The hybrid similar neighborhood robust factorization machine model for can bus intrusion detection in the in-vehicle network, *IEEE Trans. Intell. Transp. Syst.(T-ITS)*, **22** (2021), 1–9, https://doi.org/10.1109/TITS.2021.3113638
24. G. Xie, L. T. Yang, Y. Yang, H. Luo, R. Li, M. Alazab, Threat analysis for automotive CAN networks: A GAN model-based intrusion detection technique, *IEEE Trans. Intell. Transp. Syst.(T-ITS)*, **22** (2021), 4467–4477, https://doi.org/10.1109/TITS.2021.3055351
25. H. Lee, K. Choi, K. Chung, J. Kim, K. Yim, Fuzzing CAN packets into automobiles, in *IEEE International Conference on Advanced Information Networking & Applications (AINA)*, (2015), 817–821, https://doi.org/10.1109/AINA.2015.274
26. D. S. Fowler, J. Bryans, M. Cheah, P. Wooderson, S. A. Shaikh, A method for constructing automotive cybersecurity tests, a CAN fuzz testing example, in *IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, (2019), 1–8, https://doi.org/10.1109/QRS-C.2019.00015
27. E. Seo，H. Song，H. Kim, GIDS: GAN based intrusion detection system for in-vehicle network, in *2018 16th Annual Conference on Privacy, Security and Trust (PST)*, (2018), 1–6, https://doi.org/10.1109/ PST.2018.8514157
28. C. Zhang, H. Zhao, Z. Cao. The vulnerability mining method for KWP2000 protocol based on deep learning and fuzzing, *J. Shand. Univ.*, **32** (2018), 17–22, https://doi.org/10.6040/j.issn.1672-3961.0.2018.340
29. D. S. Fowler, J. Bryans, S. A. Shaikh, P. Wooderson, Fuzz testing for automotive cyber-security, in *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, (2018), 239–246, https://doi.org/10.1109/DSN-W.2018.00070
30. H. Lee, S. H. Jeong, H. K. Kim, OTIDS: A novel intrusion detection system for in-vehicle network by using remote frame, in *2017 15th Annual Conference on Privacy, Security and Trust (PST)*, (2017), 57–5709, https://doi.org/10.1109/PST.2017.00017

31. M. Arjovsky, S. Chintala, L. Bottou. Wasserstein GAN, preprint, arXiv:1701.07875.
32. I. Gulrajani, F. Ahmed, M. Arjovsky, Improved training of wasserstein GANs, *Adv. Neural Inf. Proc. Syst.*, (2017), 5767–5777, https://doi.org/10.48550/arXiv.1704.00028.
33. J. Lafferty, A. Mccallum, F. Pereira, Conditional random fields: Probabilistic models for segmenting and labeling sequence data, in *Proceedings of the Eighteenth International Conference on Machine Learning*, (2001), 282–289, Available from: https://www.seas.upenn.edu/~strctlrn/bib/PDF/crf.pdf.
34. M. Marchetti, D. Stabili, READ: Reverse engineering of automotive data frames, *IEEE Trans. Inf. Forensics Secur.*, **14** (2019), 1083–1097, https://doi.org/10.1109/TIFS.2018.2870826