applied sciences

MDPI

Article

Anomaly Detection of CAN Bus Messages Using a Deep Neural Network for Autonomous Vehicles

Aiguo Zhou [1], Zhenyu Li [1,*] and Yong Shen [2]

[1] School of Mechanical Engineering, Tongji University, Shanghai 201804, China
[2] School of Automotive Studies, Tongji University, Shanghai 201804, China
* Correspondence: zhenyu.li@tongji.edu.cn; Tel.: +86-151-5331-2859

Received: 17 June 2019; Accepted: 1 August 2019; Published: 4 August 2019

Abstract: The in-vehicle controller area network (CAN) bus is one of the essential components for autonomous vehicles, and its safety will be one of the greatest challenges in the field of intelligent vehicles in the future. In this paper, we propose a novel system that uses a deep neural network (DNN) to detect anomalous CAN bus messages. We treat anomaly detection as a cross-domain modelling problem, in which three CAN bus data packets as a group are directly imported into the DNN architecture for parallel training with shared weights. After that, three data packets are represented as three independent feature vectors, which corresponds to three different types of data sequences, namely anchor, positive and negative. The proposed DNN architecture is an embedded triplet loss network that optimizes the distance between the anchor example and the positive example, makes it smaller than the distance between the anchor example and the negative example, and realizes the similarity calculation of samples, which were originally used in face detection. Compared to traditional anomaly detection methods, the proposed method to learn the parameters with shared-weight could improve detection efficiency and detection accuracy. The whole detection system is composed of the front-end and the back-end, which correspond to deep network and triplet loss network, respectively, and are trainable in an end-to-end fashion. Experimental results demonstrate that the proposed technology can make real-time responses to anomalies and attacks to the CAN bus, and significantly improve the detection ratio. To the best of our knowledge, the proposed method is the first used for anomaly detection in the in-vehicle CAN bus.

Keywords: CAN bus; anomaly detection; DNN; autonomous vehicle; triplet loss network

1. Introduction

With the development of automotive electronic technology, lots of advanced electronic components are installed on intelligent vehicles, which leads to more complex information shuttling on the vehicle bus. On the other hand, intelligent vehicles are developing toward the direction of unmanned and networked. How to guarantee the security of information flowing among many components will be one of the focuses of intelligent vehicles in the future. The in-vehicle controller area network (CAN) bus is a standardized serial communication protocol widely used in automobile internal control systems [1]. Because of its good reliability, real-time performance and easy wiring, it has been favored by major automobile manufacturers. The CAN bus collects branch data flow from various core control systems of automobiles, such as the engine, transmission system, body system and other electrical equipment, as shown in Figure 1. All branch data are transmitted to the CAN bus in the form of broadcasting, and any node can send messages to the network at any time. This also indicates that any node in the vehicle CAN network may be attacked by malicious internal or external information at any time [2]. Nowadays, automobiles are equipped with a large number of electronic devices. In addition to basic electronic control and media systems, there are intelligent, advanced driving aids, such as automatic start–stop, parking, Accessory system, and information entertainment systems that can be connected with smart devices such as mobile phones. These systems will obtain data from the CAN bus network on the vehicle. From the perspective of intelligent development, it is inevitable for automobiles to connect to the Internet, and these electronic devices and intelligent information systems may become a way for hackers to intrude into the automobile network system. Once hackers invade these systems and successfully connect to the car CAN bus network, the driver may lose control of the vehicle [3]. At the Black Hat conference in 2014, researchers released a report on the network security of more than 20 models on the market, assessing the ability of different car manufacturers to withstand malicious attacks on different models [4]. In addition, the (Identity Document) ID in the CAN bus protocol only represents the priority of the message, and there is no original address information in the protocol. The receiving electronic control unit (ECU) cannot confirm whether the received data is the original data or not; that is, the authenticity of the received message cannot be confirmed under the existing mechanism, which easily leads to forgery and tampering of the CAN bus message by injecting false information. What makes this worse is that the priority-based arbitration mechanism in the CAN bus protocol makes it possible for hackers to carry out denial of service attacks on bus messages. An attacker can replay or flood the vehicle bus by means of sniffing or listening, which could cause the ECU to fail to send and receive messages normally. Imagine the serious consequences of an intelligent vehicle full of passengers driving automatically when the steering control node is maliciously attacked. For another example, a malicious attacker will set an attack in a target frame of the CAN bus, which will cause the driver to lose control of the throttle position and thus prevent the car from moving. Although these will not necessarily be dangerous, a money-oriented attacker will take advantage of the loopholes in the car's entertainment system to stop the car and display messages on the entertainment system screen. The owner will have to pay a ransom in order to regain control of the car. Therefore, the establishment of a real-time vehicle information security detection system is very important for intelligent vehicles [5–7].

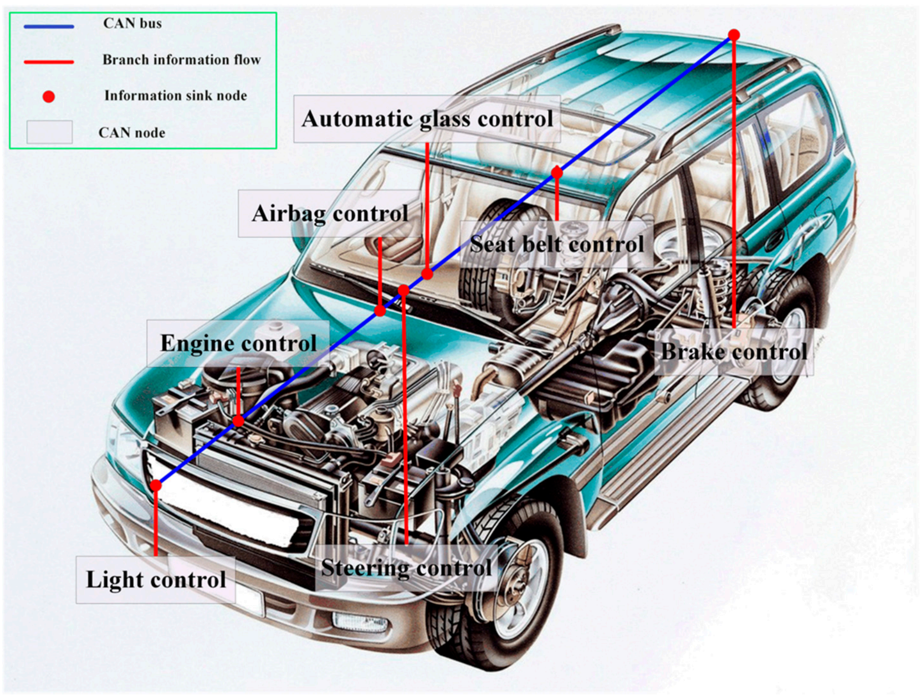

Figure 1. Controller area network (CAN) bus in an intelligent vehicle. Date information from control electronic units is collected on the CAN bus through CAN nodes.

We introduced the importance of in-vehicle CAN security and the need to deal with safety problems in Section 1. The rest of the paper is organized as follows: Section 2 presents the recent research regarding in-vehicle CAN bus messages. A novel anomaly detection method is presented in Section 3 for intelligent vehicles. In Section 4, we show and discuss the results of experiments performed on a real vehicle. Finally, we summarize the paper based on the analysis of the experimental results.

2. Related Work

2.1. Anomaly Detection Based on Traditional Methods

The future of intelligent cars must be toward the direction of unmanned information and network connection. The in-vehicle information must be the combination of the multi-node convergence of multiple sensors. However, in-vehicle information security is the barrier for the normal running of self-driving vehicles. Therefore, the establishment of an in-vehicle information anomaly detection system is extremely urgent. In the literature [8], a novel entropy-based attack detection method for in-vehicle networks is presented, which utilized entropy to describe a measure for the uncertainty of a collection of data items. The collected CAN bus messages are pre-treated, and then the information entropy calculation and relative distance measurement are compared to the baseline sample library established in the calibration stage to detect whether there are abnormalities. If there are abnormalities, the alarm will sound. If not, the CAN bus network will not be threatened by attacks during this period. A structured approach [9] is introduced to detect anomaly messages for in-vehicle networks, which allows the recognition of attacks during the operation of the vehicle without causing false positives. A further study [10] proposed an actual attack model using a malicious smartphone app in the connected car environment and demonstrated it through practical experiments, and at the same time designed a security protocol that could be applied to the car environment. Some further studies [11–13] present a set of broadcast authentication protocols, which can utilize the simplest one-way functions that are computationally efficient while authentication does not depend on disclosure delays as in the case of protocols based on one-way chains and time synchronization. A new intrusion detection algorithm is presented in [14], which is designed to identify malicious CAN messages injected by attackers in the modern vehicle CAN bus. This algorithm can recognize the anomalies in the message sequence of the CAN bus and has the characteristics of small memory and small computation. It is suitable for current bus ECUs. Aiming at the popularization of external interfaces of the vehicle control network, such as the (On-Board Diagnostic) OBD port and auxiliary media port, the technology of vehicle-to-vehicle/vehicle-to-vehicle infrastructure will be realized in the near future. The increasing attack area of vehicles exposes the control network to life-threatening attacks. A method [15] is proposed to detect the abnormal flow pattern by detecting the abnormal refresh rate of some commands, so as to meet the need of protecting the CAN bus.

2.2. Anomaly Detection Based on Deep Learning Architecture

Recently, the development of machine learning has aroused more attention in regard to its application to anomaly detection for the security of in-vehicle information. Due to its ability to automatically extract data features, real-time learning parameters and high prediction accuracy, the methods of deep learning are used in a variety of anomaly detection tasks. In the literature [16], a novel intrusion detection system is proposed, which utilizes a deep neural network (DNN) to train the weight and extract the feature vector from original CAN data packets. When real-time data packets are extracted, DNN provides the probability of each classification to identify normal and attack packets, so sensors can identify any malicious attacks on vehicles. In addition, in the literature [17,18] a DNN architecture is established, which is trained on normal data to learn the higher level features, is then used to predict future values.

2.3. Triplet Loss Network

The triplet loss is a loss function in deep learning which is used to train samples with small differences, such as handwriting, faces, etc. The feed data packet includes positive and negative anchor samples. The similarity calculation of samples can be achieved by optimizing the distance between anchor samples and positive samples to be less than that between anchor samples and negative samples. Nowadays, the triplet loss network is widely used in face recognition and object detection. For example, the literature [19] presents the use of triplet networks to solve the problem of local image descriptor learning. For the challenge of text-independent speaker verification against short utterances, the literature [20] presents an end-to-end system that directly maps learning speech features to compact fixed-length speech discrimination embedding. The system uses Euclidean distance to measure the similarity between experiments. In order to learn feature mapping, an improved priori network with residual blocks is proposed to optimize the triple loss function. In this paper, an advanced CAN bus anomaly detection system is presented for intelligent vehicles, which integrates DNN technology and a triple loss network. Firstly, the system extracts data features as a set of vectors through the deep network, and then calculates the similarity between two real-time extracted CAN data sequences and another calibrated data sequence from three data sequences using the triple loss to find out the abnormal data.

3. Proposed Method

In this section, we present the Siamese structure of the proposed system, as well as describing the DNN architecture used in this paper. In addition, we also describe the process of network training and data feature extraction. Finally, a triplet loss network is used to calculate the similarity between two randomly extracted data and annotated data, respectively.

3.1. The Overall Framework

Figure 2 presents the whole structure of CAN bus messages anomaly detection. We define the annotated CAN bus messages as an anchor, which is pre-processed off-line. The negative and positive represent two collected real-time CAN bus messages. The first three CAN bus messages make up a batch. As time goes by, we will collect a lot of CAN data sequences, and the two sequences collected in each adjacent time will form a batch with the annotated sequence, so that there will be a lot of batch data sequences. In short, the aim of this paper is to find out these abnormal sequences from lots of batches. Similar to a convolution neural network (CNN), the function of a DNN is to extract data features. Three data sequences from the same batch are imported into three networks, which are structurally consistent and share weights with each other. After that, we represent the extracted three sets of data features as three independent feature vectors. Finally, referring to the similarity calculation of a handwritten image [21,22] and face recognition [23,24], the triplet loss function is used to calculate the similarity between random data sequences and the anchor, which aims to close the distance between the normal data sequence and the annotated data sequence, and at the same time push the distance between the abnormal data sequence and the annotated data sequence.

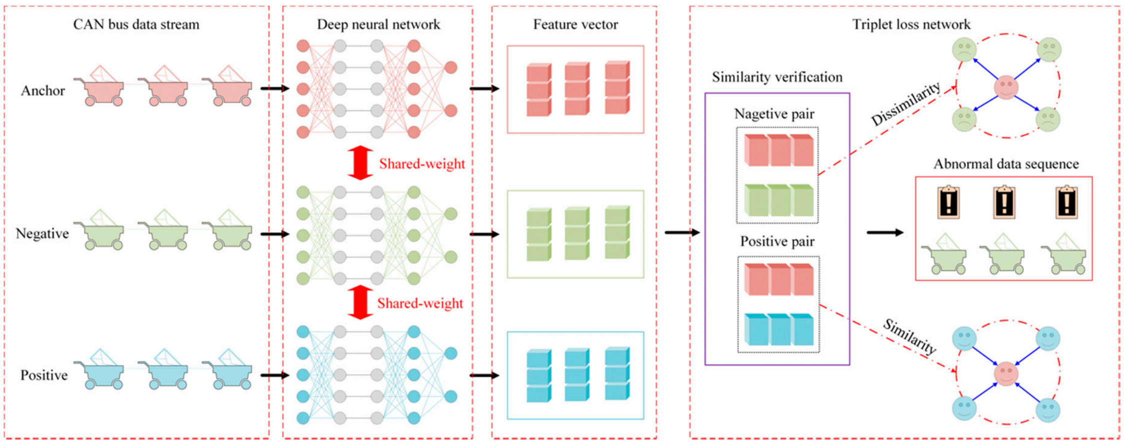

Figure 2. The full pipeline of the in-vehicle CAN anomaly detection system. The anchor, negative and positive represents annotated data, abnormal data and normal data, respectively. The aim of the proposed system is to find out the positive data.

3.2. The Shared-Weight DNN Module

In recent years, DNN algorithms have been a hot topic in the field of machine learning [25–27]. Among various detection tasks, it improves the recognition rate by a significant level. A standard CAN communication system consists of four types of data frames: data frames, remote frames, error frames and overload frames. These frames have different lengths and play different roles in the CAN communication system. However, among these frames, only data frames make meaningful operations in the process of CAN communication. The syntax of a standard CAN packet is shown in Figure 3. The whole data frame consists of different seven-bit fields: the start of the frame, arbitration field, control field, data field, cyclic redundancy check (CRC) field, (Acknowledgement Character) ACK field and the end of the frame. The start of the frame indicates the beginning of the data frame and the remote frame and consists of only one "dominant" bit. The arbitration field consists of an identifier and RTR (Radio Teletype Receiver) bits. The standard frame format is different from the extended frame format in the arbitration field format. The control field consists of six bits, including two reserved bits (r0 and r1 are the same as the CAN bus protocol extension) and four bits of data length code, and the allowable data length value is zero to eight bytes. The data field sends buffer according to the length code, indicating the length. The same is true for the received data. It can be zero to eight bytes; each byte contains eight bits, the first of which is (the Most Significant Bit) MSB. The CRC code field consists of a CRC field (15 bits) and a CRC boundary character (an implicit bit). In CRC computation, the divided polynomials include the initial domain, arbitration domain, control domain, data domain and the de-filling bit stream with 15 bits of zero. The ACK field is composed of two recessive bits sent by the sender (reply gap and reply definition), and all nodes that receive the correct CRC sequence will change the recessive bit sent into dominant bits in the reply gap of the sending node. A standard CAN bus packet consists of several parts, each of which has different functions. Therefore, each part has its own unique characteristics.

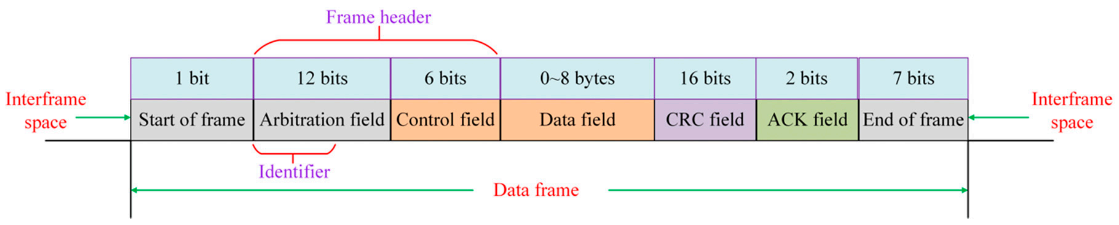

Figure 3. The syntax structure of a standard CAN packet. CRC = cyclic redundancy check, ACK = Acknowledgement Character.

In our work, we extract CAN data features by training a DNN architecture. When a CAN bus is invaded, its internal data structure will change, and the related features extracted through the deep network will be inconsistent with standard CAN features. A complete CAN data packet can be seen as a bit-stream, in which the frequency of occurrence of each bit-symbol is fixed. In our work, we define X^a, X^n, X^p as the annotated data packet, anomaly data packet and positive packet, respectively. Then:

$$\begin{cases} X^a = (P_1{}^a, P_2{}^a, \ldots, P_N{}^a) \\ X^n = (P_1{}^n, P_2{}^n, \ldots, P_N{}^n) \\ X^p = (P_1{}^p, P_2{}^p, \ldots, P_N{}^p) \end{cases}, N \in R \tag{1}$$

where p_N^a, p_N^n, p_N^p are the frequencies of bit-symbol "N" occurring in the anchor, negative and positive data sequences. Furthermore, we take them as the input of the deep network. Therefore, each layer output in the deep network can be calculated as a formula:

$$Y(X_i; w_i, b_i) = w_i X_i + b_i,\ w, b \in R \tag{2}$$

where w_i is the weight of the i^{th} layer and b_i is the bias vector of the i^{th} layer. The final output of the whole network is shown as Equation (3):

$$\begin{cases} Y = WX + B \\ W = [w_1, w_2, \ldots, w_n]^T \\ B = [b_1, b_2, \ldots, b_n]^T \end{cases} \tag{3}$$

In the process of network training, we utilize the strategy of shared-weights to ensure that the three data sequences (anchor, positive and negative) in a batch have the same training parameters, which greatly improves the efficiency of the batch process. We know that the input from each hidden layer node to another hidden layer node will be updated with the weight. The updating formula of any weight parameter is:

$$w \leftarrow w + \Delta w \tag{4}$$

where Δw is the modified value of the weight, which is realized by gradient descent. In order to calculate the updating weight of each layer, we can first calculate the modified value of the weight:

$$\Delta w = \eta \frac{\partial E}{\partial w} \tag{5}$$

where E is mean square error and η is the learning rate.

3.3. The Triplet Loss Network

The triple loss consists of a randomly selected sample from the training dataset, which is called Anchor, and then a sample of the same class as the anchor and a different sample randomly selected from Anchor. The corresponding samples are called Positive and Negative. We define $Y = f(x) \in R^d$ as the extracted data feature vector from original CAN data sequences, and then the input of the triplet loss network can be represented as $\left(f\left(X_i^a\right), f\left(X_i^n\right) f\left(X_i^p\right)\right)$. Referring to the idea of learning to rank, in this paper we utilize triplet loss to rank the similarity score of a batch data feature vector. As shown in Figure 4, the aim of the triplet loss is to close the distance between the anchor and positive and extend the distance between the anchor and negative. Unlike face recognition, we utilize a triplet loss network to find abnormal data sequences while face recognition uses it to find images belonging to the same face.

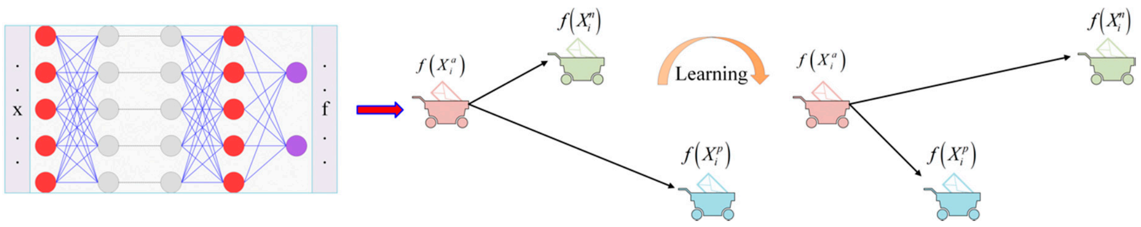

Figure 4. Learning machine of triplet loss network. The triplet loss minimizes the distance between an anchor and a positive, both of which have a normal data sequence, and maximizes the distance between the anchor and a negative of an abnormal data sequence. Where $f(x_i^a), f(x_i^p), f(x_i^n)$ are anchor data, positive data and negative data, respectively.

The embedding is represented by $f(x) \in R^d$; it embeds a data sequence into a d-dimensional Euclidean space. Furthermore, we constrain this embedding to live on the d-dimensional hypersphere [14], i.e., $\|f(x)\|_2 = 1$. The function of triplet loss is to ensure that a data sequence X_i^a (anchor) of a special CAN bus data sequence is more similar to X_i^p (positive) and dissimilarity to X_i^n (negative). Then the formula becomes:

$$\|f(x_i^a) - f(x_i^p)\|_2^2 + \alpha < \|f(x_i^a) - f(x_i^n)\|_2^2 \tag{6}$$

$$\forall(f(x_i^a), f(x_i^p), f(x_i^n)) \in \Gamma \tag{7}$$

where α is a margin that is enforced between the positive and negative pair. Γ is the set of triplets in the training data sequences. Then we minimize the loss by utilizing the formula:

$$L = \sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha\right]_+ \tag{8}$$

The purpose of the loss function is to make the reduction of loss in the training iteration as small as possible. In other words, it should be as close as possible to positive as the anchor, and as far as possible between the anchor and negative. Based on the above, we analyze the value of the margin next. The smaller the margin value, the easier it is for loss to approach zero. Therefore, both the anchor and the positive need not be pulled too close, and the anchor and the negative need not be pulled too far, so that loss can quickly approach zero. As a result of such training, it cannot distinguish similar data well.

When the margin is larger, it is necessary for the network parameters to narrow the distance between the positive and the anchor and pull the distance between the negative and the anchor. If the margin value is set too high, it is likely that the value of loss will be large, which makes it difficult to approach zero. Therefore, it is critical to set a reasonable margin value, which is an important indicator of similarity.

In fact, to generate all possible triples will result in many easily satisfied triples, which will not affect the operation of training, and lead to a slower convergence rate. The key of the triplet loss network is to choose hard triplets, which are positive, and can therefore help improve the model. However, the online generation and use of large mini-batches in the order of a few thousand exemplars only compute the $\text{argmax}_{x_i^n}\|f(x_i^a) - f(x_i^n)\|_2^2$ and $\text{argmax}_i^p\|f(x_i^a) - f(x_i^p)\|_2^2$ within a mini-batch. In addition, to have a meaningful representation of anchors, it is necessary to ensure that a minimum number of exemplars of any one identifier appear in each mini-batch. In our experiment, we sampled the training data so that in each mini-batch, about 45 CAN bus sequences are selected for each identity. In addition, randomly sampled negative data sequences are added to each mini-batch.

4. Experimental Results

In this section, we introduce the establishment of the experiment and describe the experimental results. In order to demonstrate the superiority of our method, we compare our method with the other two methods in terms of performance. In addition, we also evaluate the time consumption of the three methods.

4.1. Datasets

In this section, we simulate the scene of in-vehicle network communicating. The CAN bus data packets are created by CANoe. The used dataset is composed of three parts: the training set, validating set and test set. The number of the generated packets is about 200,000. Of this total number, 150,000 of them are normal CAN bus data and the rest of them are abnormal data. In order to avoid the over-fitting problem found in [8], 70% of normal packets are used to train and 30% of normal packets are used to validate. All data features have a special bit position in the data field. The proposed network architecture can regard mode information and value information as semantics, in which the mode information and the value information represent a special message corresponding to an ECU and the value of the mode, respectively. For example, the special command message from mode information can control the wheels, and the command message from value information can control wheel angle or the speed, as shown in Figure 5. The modal information is constant in a short time, and the value information may change with the change of noise. In our work, the valuable information is only used in the training stage, and the mode information is used in the detection stage. In the CAN syntax, the data field contains 64-bit positions (eight bytes), and the probability distribution of each kind of bit-symbol forms a special feature vector.

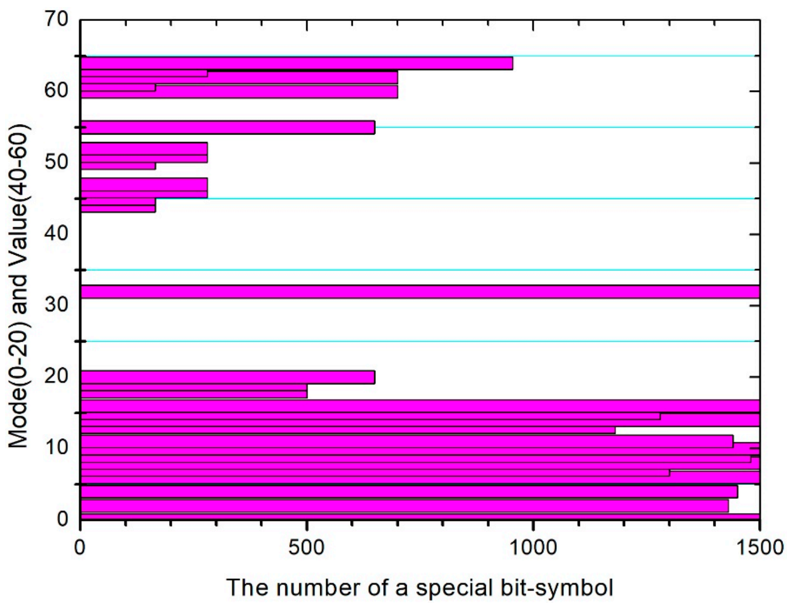

Figure 5. The number of a special bit-symbol in the data field of 8 bytes.

4.2. Performance Evaluation

At the same time, to demonstrate the better performance of proposed method, we compare the proposed method with two other methods: DNN + support vector machine (SVM) and DNN + Softmax. The former uses a deep network to train the original data, and then classifies it by embedding SVM as the back-end. The basic principle of SVM is to find the best separating hyperplane in feature space to maximize the interval between positive and negative samples in a training set. SVM is a supervised learning algorithm to solve binary classification problems. After introducing the kernel method, SVM can also be used to solve non-linear problems. The latter also uses a deep network to train the original data, and then classifies the data by embedding the Softmax function as the back-end. Softmax is a very common and important function in deep learning, especially in the field of classification. It maps some inputs to real numbers between zero and one, and guarantees the normalization sum to one, so the sum of probability of classification is just one. The anomaly detection stage is completed at the back-end. In this stage, the similarity between any two sets of CAN bus sequences and the calibrated sequence is compared to determine which group of sequences is normal and which group of sequences is invaded. In our experiment, all comparative experiments are performed under the same hardware and software conditions. To evaluate the performance of the proposed method, the front-end of the three methods proposed in this paper all adopt the same deep network structure; that is, they all have the same number of layers and neurons. In addition, the training parameters are shared with each other. Among the three methods, the learning rate of the network training stage is 0.001.

Figure 6 shows the performance changes of the three methods mentioned in this paper as the number of hidden layers increases. It can be seen that with the increase in the number of layers, the recognition accuracy of the three methods is improved, but when the number of layers increases to a certain level, the improvement gradually tapered off. In terms of performance, under the same number of hidden layers, the method that uses the triplet loss function as the back-end is the best, followed by the method using SVM as the back-end. The worst performance was from the method that used Softmax as the back-end.

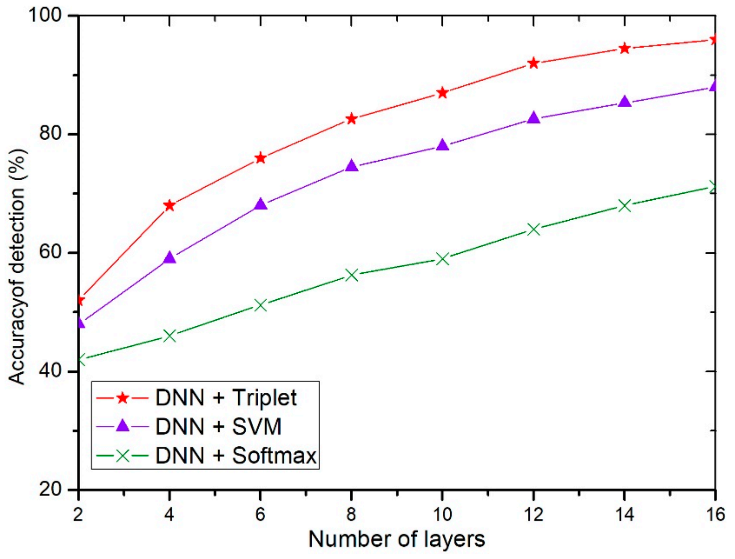

Figure 6. Performance changes with the increase of neural network layers. DNN = deep neural network; SVM = support vector machine.

In order to compare the performance of the three methods more intuitively, Figure 7 shows a box chart that provides a statistical graph of data distribution, in which the three different color boxes represent the performance of the three methods. It can also be seen that the performance rankings of the three methods are DNN + Triplet, DNN + SVM and DNN + Softmax. Specifically speaking, breaking through the limitation of hidden layer number, the lower limit and upper limit of detection accuracy using DNN + Triplet architecture are higher than the other two architectures. In addition, from the median detection accuracy of the three methods, the architecture of DNN + Triplet is also significantly higher than the other two architectures.

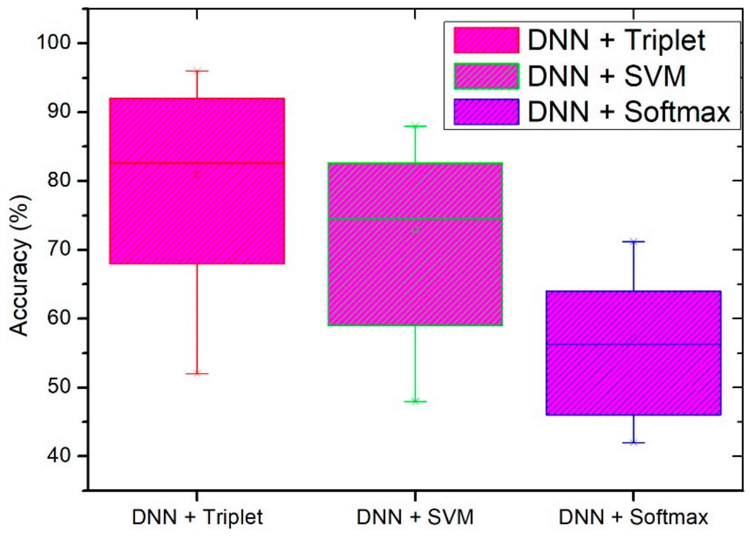

Figure 7. Statistical graph of data distribution.

We also show the time cost to perform the detection task, which depends on the different number of hidden layers as shown in Figure 8. It can be seen that the time cost increases with the increasing number of hidden layers, and the whole detection phase is about 2–19 s, in which the training phase takes up almost the entire phase of time consumption. Training time represents the measurement time, which is required to train the DNN structure in the training phase. The time cost of feature extraction represents the process consumption of transforming data features into feature vectors. The time cost of detection represents the process consumption of similarity calculation in terms of the triplet loss network.

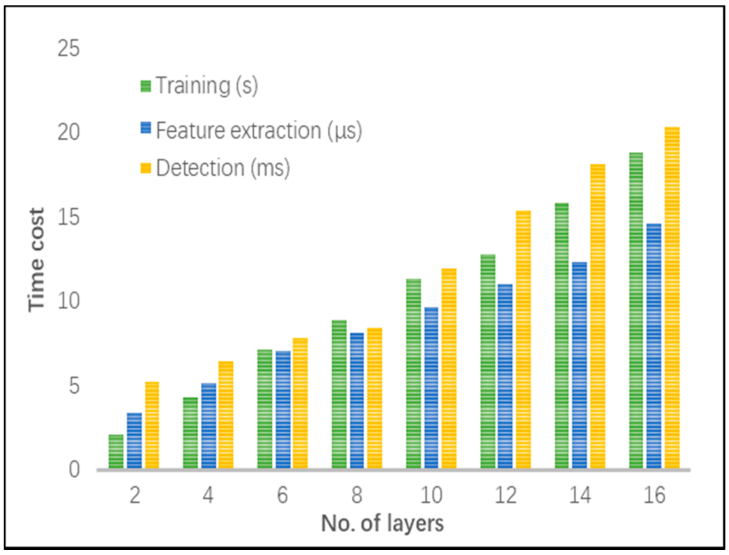

Figure 8. Time complexity in a different number of layers.

5. Conclusions

We propose a novel approach with the fusion of a deep neural network and a triplet loss network for CAN bus message anomaly detection, as well as evaluate the performance by comparing the proposed method with two other similar methods. The proposed method first utilizes DNN to train CAN bus sequences and extract feature vectors that correspond to special CAN bus sequences. Most importantly, the weight parameters in this stage of training are shared. Then, three random data sequences from the CAN bus database are imported into the triplet loss network for similarity calculations. The normal CAN message is more similar to the labeled data, otherwise it is considered to be an abnormal message. Experimental results demonstrate that the performance of the proposed method is excellent. In addition, we show the performance of our proposed method under different hidden layers. The results show that the performance of the proposed method is greatly improved with the increase of the number of hidden layers. However, when we compare the time consumption under different hidden layers, we can see that the time consumption increases with the increase of layers.

Author Contributions: Z.L. conceived and designed the experiments, as well as writing the manuscript. A.Z. and Y.S. helped to make suggestions and revise this manuscript. Z.L. performed the experiments and analyzed the results.

Funding: This research was funded by National key research and development program, grant number 2016YFB0100902.

Acknowledgments: The authors would like to thank the anonymous referee for the helpful comments. The research was supported by the National Key Research and Development Program (no. 2016YFB0100902).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Müter, M.; Asaj, N. Entropy-based anomaly detection for in-vehicle networks. In Proceedings of the IEEE Intelligent Vehicles Symposium, Baden-Baden, Germany, 5–9 June 2011.
2. Nilsson, D.; Larson, U.E. Simulated attacks on CAN buses: vehicle virus. In Proceedings of the 5th IASTED International Conference on Communication Systems and Networks, Palma de Mallorca, Spain, 1–3 September 2008.
3. Miller, C.; Valasek, C. Adventures in automotive networks and control units. *Def Con.* **2013**, *21*, 260–264.
4. Miller, C.; Valasek, C. A survey of remote automotive attack surfaces. In Proceedings of the Black Hat, Las Vegas, NV, USA, 2–7 August 2014; p. 94.
5. Othmane, L.B.; Weffers, H.; Mohamad, M.M.; Wolf, M. A survey of security and privacy in connected vehicles. In *Wireless Sensor and Mobile Ad-Hoc Networks*; Springer: New York, NY, USA, 2015; pp. 217–247.
6. Markovitz, M.; Wool, A. Field classification, modeling and anomaly detection in unknown can bus networks. *Veh. Commun.* **2017**, *9*, 43–52. [CrossRef]
7. Wang, J.; Ma, H. Humanoid force information detection system based on can bus. *J. Huazhong Univ. Sci. Technol.* **2004**, *32*, 164–166.
8. Müter, M.; Groll, A.; Freiling, F.C. A structured approach to anomaly detection for in-vehicle networks. In Proceedings of the Sixth IEEE International Conference on Information Assurance and Security, Atlanta, GA, USA, 23–25 August 2010; pp. 92–98.
9. Woo, S.; Jo, H.J.; Lee, D.H. A Practical Wireless Attack on the Connected Car and Security Protocol for In-Vehicle CAN. *IEEE Trans. Intell. Transp. Syst.* **2014**, *16*, 1–14. [CrossRef]
10. Groza, B.; Murvay, S. Efficient Protocols for Secure Broadcast in Controller Area Networks. *IEEE Trans. Ind. Inform.* **2013**, *9*, 2034–2042. [CrossRef]
11. Groza, B.; Murvay, P.S. Broadcast Authentication in a Low Speed Controller Area Network. In Proceedings of the International Conference on E-Business and Telecommunications, Rome, Italy, 24–17 July 2012; Springer: Berlin/Heidelberg, Germany.
12. Groza, B.; Murvay, S.; Herrewege, A.V.; Verbauwhede, I. LiBrA-CAN: Lightweight Broadcast Authentication for Controller Area Networks. *ACM Trans. Embed. Comput. Sys.* **2017**, *16*, 1–28. [CrossRef]
13. Marchetti, M.; Stabili, D. Anomaly detection of CAN bus messages through analysis of ID sequences. In Proceedings of the IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 11–14 June 2017.
14. Kang, M.J.; Kang, J.W. Intrusion detection system using deep neural network for in-vehicle network security. *PLoS ONE* **2016**, *11*, e0155781. [CrossRef] [PubMed]
15. Moore, M.R.; Bridges, R.A.; Combs, F.L.; Starr, M.S.; Prowell, S.J. Modeling inter-signal arrival times for accurate detection of can bus signal injection attacks: a data-driven approach to in-vehicle intrusion detection. In Proceedings of the 12th Annual Conference on Cyber and Information Security Research, Oak Ridge, TN, USA, 4–6 April 2017; p. 11.

16. Zang, D.; Liu, J.; Wang, H. Markov Chain-Based Feature Extraction for Anomaly Detection in Time Series and Its Industrial Application. In Proceedings of the Chinese Control and Decision Conference (CCDC), Shenyang, China, 9–11 June 2018.
17. Wang, X.; Zhou, Q.; Harer, J.; Brown, G.; Chin, P. Deep learning-based classification and anomaly detection of side-channel signals. *Cyber Sens.* **2018**. [CrossRef]
18. Nawaz, S.; Calefati, A.; Ahmed, N.; Gallo, I. Handwritten Characters Recognition via Deep Metric Learning. In Proceedings of the 13th IAPR International Workshop on Document Analysis Systems (DAS), Vienna, Austria, 24–27 April 2018; pp. 417–422.
19. Kumar, B.G.; Carneiro, G.; Reid, I. Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 5385–5394.
20. Zhang, C.; Koishida, K. End-to-End Text-Independent Speaker Verification with Triplet Loss on Short Utterances. *Interspeech* **2017**, 1487–1491. [CrossRef]
21. Hoffer, E.; Ailon, N. Deep metric learning using triplet network. In Proceedings of the International Workshop on Similarity-Based Pattern Recognition, Copenhagen, Denmark, 12–14 October 2015; Springer: Cham, Germany; pp. 84–92.
22. Cheng, D.; Gong, Y.; Zhou, S.; Wang, J.; Zheng, N. Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1335–1344.
23. Schroff, F.; Kalenichenko, D.; Philbin, J. Facenet: A Unified Embedding for Face Recognition and Clustering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 815–823.
24. Hermans, A.; Beyer, L.; Leibe, B. In defense of the triplet loss for person re-identification. *arXiv preprint* **2017**, arXiv:1703.07737.
25. Li, K.; Mao, S.; Li, X.; Wu, Z.; Meng, H. Automatic lexical stress and pitch accent detection for L2 English speech using multi-distribution deep neural networks. *Speech Commun.* **2018**, *96*, 28–36. [CrossRef]
26. Tuttle, A.H.; Molinaro, M.J.; Jethwa, J.F.; Sotocinal, S.G.; Prieto, J.C.; Styner, M.A.; Zylka, M.J. A deep neural network to assess spontaneous pain from mouse facial expressions. *Mol. Pain* **2018**, *14*, 1744806918763658. [CrossRef] [PubMed]
27. Hannun, A.Y.; Rajpurkar, P.; Haghpanahi, M.; Tison, G.H.; Bourn, C.; Turakhia, M.P.; Ng, A.Y. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nat. Med.* **2019**, *25*, 65. [CrossRef] [PubMed]