HistCAN: A real-time CAN IDS with enhanced historical traffic learning capability

Shuguo Zhuo[†‡], Nuo Li[†§], Kui Ren[†§]
[†]The State Key Laboratory of Blockchain and Data Security, Zhejiang University
[‡] ZJU-Hangzhou Global Scientific and Technological Innovation Center, Hangzhou, China
[§] School of Cyber Science and Technology, Zhejiang University
{shuguo.zhuo, nuo_li, kuiren}@zju.edu.cn

Abstract—**Due to the absence of encryption and authentication mechanisms, the Controller Area Network (CAN) protocol, widely employed in in-vehicle networks, is susceptible to various cyber attacks. In safeguarding in-vehicle networks against cyber threats, numerous Machine Learning-based (ML) and Deep Learning-based (DL) anomaly detection methods have been proposed, demonstrating high accuracy and proficiency in capturing intricate data patterns. However, the majority of these methods are supervised and heavily reliant on labeled training datasets with known attack types, posing limitations in real-world scenarios where acquiring labeled attack data is challenging. In this paper, we present HistCAN, a lightweight and self-supervised Intrusion Detection System (IDS) designed to confront cyber attacks using solely benign training data. HistCAN employs a hybrid encoder capable of simultaneously learning spatial and temporal features of the input data, exhibiting robust pattern-capturing capabilities with a relatively compact parameter set. Additionally, a historical information fusion module is integrated into HistCAN, facilitating the capture of long-term dependencies and trends within the CAN ID series. Extensive experimental results demonstrate that HistCAN generally outperforms the compared baseline methods, achieving a high F1 score of 0.9954 in a purely self-supervised manner while satisfying real-time requirements.**

I. Introduction

With the rapid development of Intelligent Transportation Systems (ITS), wireless communication technologies, such as Cellular Vehicle-to-Everything (C-V2X) [12] and Dedicated Short Range Communication (DSRC) [10], are increasingly employed to enhance interactions among vehicles, pedestrians, and infrastructure. Through the exchange of information among road users, ITS can prevent potential traffic accidents and substantially improve traffic efficiency and safety. However, the extensive use of remote connections exposes smart vehicles to the insecure internet, making them susceptible to remote intrusion and hijacking, as reported in [2], [20].

The Controller Area Network (CAN) protocol serves as the de facto standard for communication within in-vehicle networks (IVNs), connecting numerous electronic control units (ECUs). These ECUs manage various vehicle functions and collaborate by exchanging CAN messages. However, the absence of an authentication procedure makes CAN vulnerable to cyber attacks, lacking essential security features. Specifically, through remote connectivity channels like C-V2X, a potential adversary can exploit vulnerabilities to hijack an onboard ECU and sniff CAN traffic. Moreover, the collected traffic data can be analyzed through reverse engineering analysis techniques, enabling adversaries to execute injection attacks on the CAN bus and posing potential risks to the safety of the target vehicle, pedestrians, and road infrastructures.

To address the limitations of the CAN protocol, various types of CAN Intrusion Detection Systems (IDS) have been proposed to monitor broadcast messages on the CAN bus and verify the presence of injected attacks. Among these methods, deep learning-based approaches have garnered significant attention in recent years due to their enhanced effectiveness in identifying anomalies in high-dimensional or non-linear time series data. However, many existing supervised and semi-supervised deep learning methods, such as [1], [3] and [19], encounter significant challenges when dealing with limited labeled data and dynamic anomalies that have not been observed before - a common scenario in in-vehicle anomaly traffic detection. Consequently, self-supervised methods are well-received, as they do not impose strict requirements on elaborately and precisely labeled data. These methods can be broadly categorized into one-class classification-based, distance-based, and reconstruction-based methods [22].

Reconstruction-based models, such as those presented in [5], [6], [11] and [21], are designed to reconstruct normal samples. Instances failing to be reconstructed by the model are deemed anomalies, expressed concretely through a significant reconstruction error. This approach is gaining rapid momentum due to its capability to handle complex data by integrating with various models to capture data patterns. However, when applied to the CAN stream, most current methods singularly focus on either the time-domain [7] or spatial-domain [16]. Consequently, to enhance the comprehensive pattern-capturing capabilities, these methods often incorporate large parameters or layers, a process that is time-consuming and may be unsuitable for real-time systems. Furthermore, nearly all methods concentrate only on the CAN sequence at the current time step, neglecting the historical association between CAN sequences at different time steps [16] [19]. This oversight leads to a failure to detect traffic patterns related to long-term dependencies in CAN bus traffic.

Symposium on Vehicles Security and Privacy (VehicleSec) 2024
26 February 2024, San Diego, CA, USA
ISBN 979-8-9894372-7-6
https://dx.doi.org/10.14722/vehiclesec.2024.23028
www.ndss-symposium.org

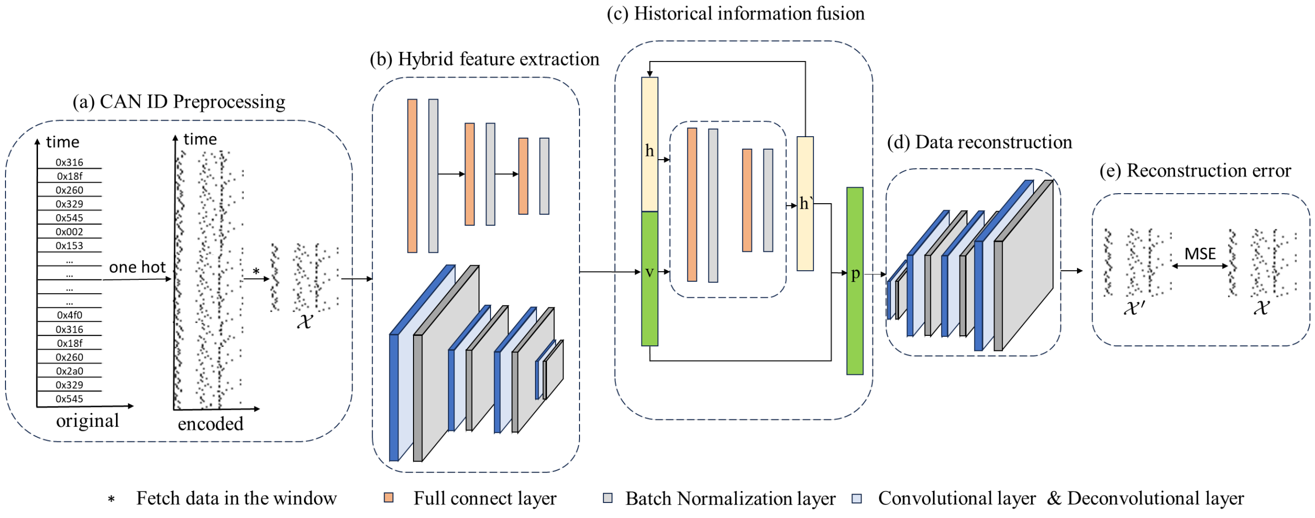

Fig. 1. The overview of the framework of HistCAN, which comprises a CNN-MLP hybrid encoder, a historical information fusion module and a Deconvolutional based decoder.

To overcome the limitations of previous approaches, we introduce HistCAN, a self-supervised method for detecting anomalies in CAN bus traffic within a reconstruction-based framework. HistCAN utilizes a hybrid encoder and a historical information fusion module to learn latent features. The hybrid encoder integrates components of Convolutional Neural Networks (CNNs) and Multi-Layer Perceptrons (MLPs) to simultaneously extract spatial and temporal features from the input data. Concurrently, the historical information fusion module aids in capturing long-term dependencies across different CAN series. We demonstrate that the hybrid encoder exhibits robust pattern-capturing capabilities, even with a reduced training dataset, and maintains a relatively compact parameter set for efficient real-time detection. Our contributions can be summarized as follows:

- We propose a hybrid autoencoder (AE) architecture that incorporates CNN and MLP components into its encoder to simultaneously capture spatial and temporal patterns, countering both known and unknown attacks.
- We introduce a historical information fusion module into the proposed model that can learn long-term dependencies from CAN series, further enhancing the model's detection accuracy.
- The proposed HistCAN generally outperforms the compared baseline methods, achieving a high F1 score of 0.9954. Meanwhile, HistCAN maintains a relatively compact parameter set and is proven to be suitable for real-time deployment, as indicated by experimental results.

The rest of the paper is organized as follows: in Section II, we introduce the proposed self-supervised learning-based anomaly detection method. Section III demonstrates the experimental setup and provides a detailed analysis of the experimental results. Finally, we offer a summary of the paper.

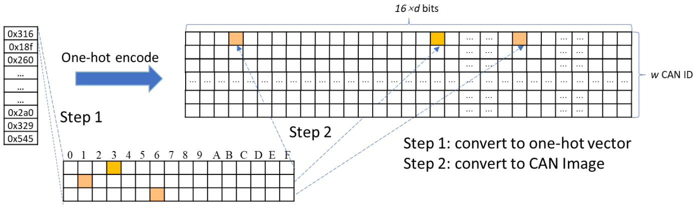
16 ×d bits
0x316
0x18f
0x260
...
...
...
0x2a0
0x329
0x545
One-hot encode
w CAN ID
Step 1
Step 2
0 1 2 3 4 5 6 7 8 9 A B C D E F
Step 1: convert to one-hot vector
Step 2: convert to CAN Image

Fig. 2. Demonstration of CAN IDs preprocess procedure. Step 1: convert CAN IDs to one-hot vector, Step 2: stack one-hot vectors to CAN ID images.

II. Proposed Framework: HistCAN

In this section, we present our proposed model, HistCAN (as illustrated in Fig 1), which incorporates a CNN-MLP hybrid encoder and a historical information fusion module for CAN anomaly detection, distinguishing it from other reconstruction-based methods. The primary components of the model are outlined as follows.

A. Input Preprocess

The packet preprocessing methodology adheres to a similar approach as described in [16]. The CAN stream is segmented into individual windows, each comprising w CAN messages. Each CAN message contains a CAN ID consisting of d hexadecimal numbers. For model data preparation, one-hot encoding is employed for each CAN ID, leading to a binary representation. As illustrated in Fig 2, these encoded CAN IDs are subsequently stacked to construct a CAN image matrix, denoted as M, with dimensions $(w, 16 \times d)$. In this matrix, the column size corresponds to the window size w, and each row represents a one-hot encoded hexadecimal value.

B. Encoder with Hybrid Feature Extraction

As shown in Figure 1 (b), Convolutional Neural Networks (CNNs) and Multi-Layer Perceptrons (MLPs) are combined to extract features from input data, forming a hybrid encoder. For the reshaped sequential CAN message data, i.e., matrix

$M \in \mathbb{R}^{w \times 16 \times d}$, the latent presentation vector v is obtained as follows:

$$v = concat(CNN(M), MLP(expand(M)))$$

where CNN is a simple Convolutional Neural Network, MLP is a shallow Multi-Layer Perceptrons and $expand$ is a function expanding input CAN image into one dimension vector in sequential order. The 2-dimensional CAN images (i.e., M) encapsulate bit-level spatial correlation information among different CAN IDs, directing them to CNNs adept at capturing local spatial image patterns. Simultaneously, the 1-dimensional ordered CAN ID sequences (i.e., $expand(M)$) convey the temporal information of the CAN ID stream over time, guiding them to MLPs aimed at learning global temporal relationships. This fusion facilitates the capture of both spatial and temporal features in the data, enhancing the model's comprehension of behaviors within CAN ID sequences.

C. Historical Information Fusion Module

In each window, a CNN-MLP model is utilized to extract a feature vector v. However, the encoder of a AE model often learns localized representations, neglecting long-term dependencies between different windows. To preserve the temporal context of the CAN ID series, a historical information vector h is introduced and integrated with the latent presentation vector v. As illustrated in Figure 1 (c), this integration is achieved by employing an additional MLP network, generating a new vector h'. Subsequently, h' is concatenated with v to create the final and comprehensive latent presentation vector p, which is then input to a decoder for further processing. At the conclusion of each window, h' is looped back for the subsequent round of historical information extraction. Algorithm 1 describes the fusion procedure, where v_t, p_t, h_t and M_t are vectors, matrix mentioned before at the t-th timestep and $fuse$ is a function fusing historical information and the current latent presentation vector.

Algorithm 1: Historical Information Fusion Procedure

```
Input: previous historical feature $h_{t-1}$, current
       window series $W_t$
Output: hybrid feature $p_t$
Initialization: initial historical feature $h_0 = \mathbf{0}$;
For t in series_windows_counts do
    v$_t = concat(CNN(W_t), MLP(W_t))$
    p$_t = concat(v_t, h_{t-1})$
    h$_t = fuse(v_t, h_{t-1})$
end for
return $p_t$;
```

The main objective of this approach is to maintain a specific level of historical information throughout the feature extraction process. Through the incorporation of the historical information vector h and subsequent fusion, this methodology allows for the simultaneous consideration of both current and past information at each time step. This assists the model in gaining a more comprehensive understanding of the long-term dependencies within the CAN ID series. By updating h at the end of each window, historical information is propagated across various windows, enhancing the ability to capture enduring patterns and evolving trends within the sequences.

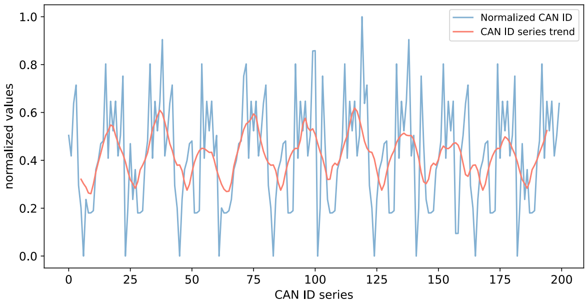

Fig. 3. Variation and trend of normalized CAN ID values

D. Reconstruction Module

As shown in Figure 1 (d), we employ a four-layered deconvolutional neural network architecture to reconstruct the one-dimensional feature vectors (i.e., the input data $\mathcal{X}$) back to their original size, denoted as $\mathcal{X}'$. The training objective aims to minimize the mean squared error between $\mathcal{X}$ and $\mathcal{X}'$ as a $reconstruction\,error$, serving as a measure of dissimilarity.

$$reconstruction\,error = MSE(\mathcal{X}, \mathcal{X}')$$

By doing so, we aim to train the model to capture the underlying patterns within normal data while demonstrating sensitivity to anomalous patterns. Specifically, when abnormal data is encountered, characterized by distinct patterns compared to normal data which is not learned in the encoder and proposed historical information module, a substantial dissimilarity between input $\mathcal{X}$ and output $\mathcal{X}'$ arises.

E. Anomaly Score

The Mean-Square-Error between the inputs and their reconstructions is employed to predict the anomaly status of the CAN Frame sequence. The sequence is classified as anomalous if the reconstruction error exceeds a predetermined threshold, chosen based on the loss observed during the training phase. Thus, we consider the following formulated labeling criteria:

$$\mathcal{Y} = \begin{cases} 1 \text{ (anomaly) } \text{ if } AnomalyScore(\mathcal{X}) \geq \phi \\ 0 \text{ (normal) } \text{ if } AnomalyScore(\mathcal{X}) < \phi \end{cases}$$

where $\mathcal{Y}$ is the detection result, $\mathcal{X}$ is the CAN ID sequence, $AnomalyScore$ is the reconstruction error function, and ϕ is the selected threshold.

F. Real-time Discussion

As studied in [23], their findings show that a simple model, such as MLP, outperforms intricate models like transformer-based methods in scenarios where time series data exhibit clear trends and periodicity. A time series trend reflects the persistent movement of the series over a specific duration, offering valuable insights into its long-term behaviors. In Fig. 3, the variation and trend of normalized CAN ID values, extracted from the Car-Hacking dataset, are illustrated. The

TABLE I. DETECTION PERFORMANCE COMPARED WITH OTHER METHODS

	DoS			Fuzzy			gear			RPM			Average		
	F1	recall	Precision	F1	recall	Precision	F1	recall	Precision	F1	recall	Precision	F1	recall	Precision
DCNN(supervised) [19]	0.9995	0.9989	1.0	0.9980	0.9965	0.9995	0.9994	0.9989	0.9999	0.9996	0.9994	0.9999	0.9996	0.9994	0.9999
GIDS [16]	0.9818	**0.9960**	0.9680	0.9839	0.9950	0.9730	0.9865	0.9900	0.9830	0.9729	0.9650	0.9810	0.9813	0.9865	0.9763
SSAD-NP [18]	0.9833	0.9916	0.9751	0.8861	0.8345	0.9445	0.9261	0.8803	0.9768	0.9850	0.9983	0.9720	0.9451	0.9262	0.9671
Hybrid-AE(ours)	0.9871	0.9810	0.9933	0.9913	0.9853	0.9974	**0.9919**	**0.9905**	**0.9934**	0.9939	0.9923	0.9954	0.9911	0.9873	0.9949
HistCAN(ours)	**0.9972**	0.9952	**0.9992**	**0.9973**	**0.9960**	**0.9985**	0.9907	0.9895	0.9919	**0.9964**	**0.9951**	**0.9976**	**0.9954**	**0.9940**	**0.9968**

CAN ID stream manifests a highly cyclical pattern. Therefore, instead of employing a complex deep neural network with a large number of parameters and layers for powerful feature extraction, we opt for a lightweight CNN-MLP hybrid encoder. This choice allows us to achieve comparable performance with a relatively compact parameter set. By reducing the model parameter size, HistCAN attains improved detection speed, aiming to meet the real-time requirements of real-world vehicle applications (see real-time analysis in the following section).

III. EXPERIMENTAL EVALUATION

A. Implementation

The proposed HistCAN and its variant are implemented using PyTorch [13] and trained with the Adam optimizer [9] employing a learning rate of 0.0001, and a fixed batch size of 128. The reconstruction threshold ϕ is set $\mu + 3\sigma = 0.006$ where μ is the mean reconstruction loss on normal data and σ is the empirical standard deviation.

For the details of the model, we first define Conv2(k, s, c, p) to denote a 2D convolution layer, where k, s, c and p are the kernel size, stride size and the number of channels respectively. L(i, o) is defined to denote a linear layer, in which i and o are input and output dimensions. For Car-hacking dataset, we implement the encoder using four convolution layers: Conv2(5, 2, 128, 1)-Conv2(5, 2, 256, 1)-Conv2(5, 2, 256, 1)-Conv2(5, 2, 128, 1) and three linear layers: L(48×48, 1024)-L(1024, 256)-L(256, 48). The decoder is implemented as Dconv2(4, 2, 256, 1)-Dconv2(4, 2, 256, 1)-Dconv2(4, 2, 128, 1)-Dconv2(4, 2, 1, 1), where Dconv2 denotes the 2D deconvolution layer. Historical information and current representations are fused by an two linear layers: L(432, 48)-L(48, 48). Except for the last Dconv2, each layer is followed by a batch-normalization (BN) [8] and a ReLU activation. We utilize a sliding window to obtain a set of sub-series [17], with a fixed window size of 48. All experiments are conducted on a NVIDIA GeForce RTX 3060 12GB GPU and a 11th Gen Intel(R) Core(TM) i7-11700F @ 2.50GHz 16GB CPU.

B. Dataset

We evaluate the proposed HistCAN on the Car-Hacking dataset [4] [16] which is currently the most widely used dataset in the literature for evaluating CAN-based IDSs [14]. The dataset was collected from a real vehicle *Hyundai YF Sonata* with injected attacks. It comprises 5 subsets, including 500 seconds of benign data (collected during normal driving) and four attack types: DoS, Fuzzing, and two Spoofing attacks (RPM and gear). Each attack subset involves 300 instances of individual intrusions (message injections). For additional details, we direct readers to [4].

C. Main Results

The proposed model is trained in self-supervised paradigm, with only normal data collected in Car-Hacking dataset. It is subsequently tested on four types of attack data, combining both normal and abnormal traffic.

We compare the proposed model with several conventional and deep learning based methods for CAN bus anomaly detection as baselines. These include DCNN, a supervised method [19], and two self-supervised methods: GIDS [16] and SSAD-NP (***S**elf-**S**upervised **A**nomaly **D**etection Using **N**oised **P**seudo Normal Data*) [18].

DCNN is a supervised model with high accuracy. We select it as our baseline with the goal of approximating the performance of the best-supervised model using our self-supervised approach. GIDS is a self-supervised model based on GAN, demonstrating an ability to identify unknown attacks with an accuracy of nearly 98% using only benign data. SSAD-NP is another self-supervised model employing noised pseudo-normal data to delineate the boundary between normal and abnormal CAN traffic, achieving significant accuracy. To show the importance of the major component, we also conduct comparisons with a variant of HistCAN which is without the historical information fusion module and uses only a CNN-MLP hybrid encoder (denoted as Hybrid-AE).

We utilize F1-score, recall, precision, and AUC (Area Under Curve) as evaluation metrics to assess HistCAN's performance against message injection attacks. Precision gauges the detector's accuracy in predicting anomaly traffic, while recall measures its ability to detect all anomaly traffic. The F1-score offers a comprehensive evaluation of the detector's overall performance by harmonizing the aforementioned metrics. Finally, AUC is determined by calculating the area under the Receiver Operating Characteristic (ROC) curve with varying thresholds. A higher AUC value indicates superior performance in detecting anomaly traffic and reducing the false alarm rate.

Quantitative results are given in Table I. We can see that the proposed HistCAN model and its variant Hybrid-AE generally outperforms the compared methods. Thanks to the CNN-MLP hybrid encoder extracting both spatial and temporal patterns, Hybrid-AE generally exhibits superior performance compared to GIDS and SSAD-NP, achieving a high precision of 0.9949 and an impressive F1-score of 0.9911 on average. Moreover, the incorporation of the historical information fusion module into HistCAN helps to acquire extra knowledge between individual sequences and enhances the performance of Hybrid-AE, further boosting the average F1 score, recall, and precision to 0.9954, 0.9940, and 0.9968, respectively. Note

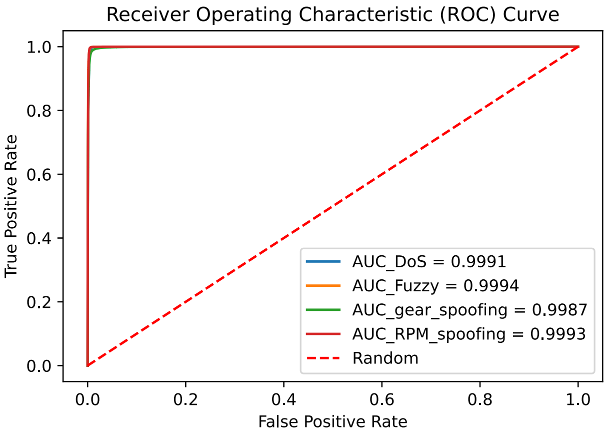

Fig. 4. AUC scores of HistCAN across all attack types.

that DCNN attains the highest performance metrics. However, DCNN requires training with traced attack data, whereas our proposed HistCAN model achieves comparable performance using only benign data, which is more practical for real-world applications. In addition, Fig 4 shows the AUC scores of HistCAN across all attack types. We can see that HistCAN attains AUC scores surpassing 0.9980 for every attack type, indicating the high performance of the proposed model.

TABLE II. F1 SCORES WITH DISTINCT TRAINING DATA SIZES

train_data size	DoS	Fuzzy	gear	RPM	Average
10,000	0.9935	0.9884	0.9406	0.9654	0.9720
20,601	0.9972	0.9973	0.9907	0.9964	0.9954

To assess the model's robustness under limited training resources, we conducted an additional experiment wherein HistCAN was trained with a dataset reduced by half. Table II presents the F1 scores of HistCAN for the two distinct training dataset sizes. Notably, even with a halved training dataset, HistCAN can learn meaningful patterns and attains performance comparable to the full dataset.

TABLE III. DETECTION SPEED OF HISTCAN

	DoS	Fuzzy	gear	RPM	Average
frame per seconds(GPU)	9145	9312	8998	8305	8940
frame per seconds(CPU)	2834	2905	2524	2504	2692

It is imperative for an IDS to satisfy the latency demands of real-world applications, particularly for resource-limited vehicles. Fig 5 illustrates the Kernel Density Estimation (KDE) [15] of the normal data collected from a real *Hyundai YF Sonata* vehicle in the Car-Hacking dataset, providing insight into the underlying distribution of CAN frames per second on the CAN bus. As depicted in Fig 5, the concentration of CAN frames per second is predominantly below 2000 frames per second. Meanwhile, we assessed the detection speed of HistCAN on both CPU[1] and GPU[2], as detailed in Table III. The average detection speed on GPU reached 8940 frames per second, whereas on CPU, it was 2692 frames per second. Consequently, the proposed HistCAN detector demonstrates the capability to achieve real-time monitoring even with the use of inexpensive, generic CPUs.

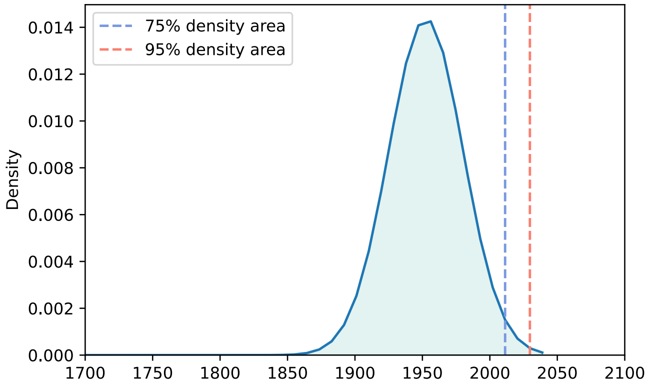

Fig. 5. Kernel Density Estimation (KDE) of the normal data collected from the Car-Hacking dataset.

IV. CONCLUSION

This paper introduces a novel IDS named HistCAN designed for CAN bus anomaly detection. In HistCAN, we employ a hybrid CNN-MLP structure to acquire a spatial-temporal representation. This representation captures multidimensional information from CAN sequences, encompassing spatial and temporal perspectives. Additionally, HistCAN integrates a historical information fusion module to grasp long-term dependencies across CAN ID series, thereby enhancing detection accuracy and overall performance. Evaluation results demonstrate that, in comparison to various state-of-the-art methods, HistCAN attains superior performance on the Car-Hacking dataset in a self-supervised manner, without the need for labeled attack data.

REFERENCES

[1] F. Amato, L. Coppolino, F. Mercaldo, F. Moscato, R. Nardone, and A. Santone, "Can-bus attack detection with deep learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 8, pp. 5081–5090, 2021.

[2] Andy Greenberg, "Hackers Remotely Kill a Jeep on the Highway—With Me in It," https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/, 2015.

[3] R. Gundu and M. Maleki, "Securing can bus in connected and autonomous vehicles using supervised machine learning approaches," in *2022 IEEE International Conference on Electro Information Technology (eIT)*. IEEE, 2022, pp. 042–046.

[4] Hacking and Countermeasure Research Lab, "Car-Hacking Dataset for the Intrusion Detection," https://ocslab.hksecurity.net/Datasets/CAN-intrusion-dataset, 2020, : 202181.

[5] M. Hanselmann, T. Strauss, K. Dormann, and H. Ulmer, "Canet: An unsupervised intrusion detection system for high dimensional can bus data," *IEEE Access*, vol. 8, pp. 58 194–58 205, 2020.

[6] T.-N. Hoang and D. Kim, "Detecting in-vehicle intrusion via semi-supervised learning-based convolutional adversarial autoencoders," *Vehicular Communications*, vol. 38, p. 100520, 2022.

[1] 11th Gen Intel(R) Core(TM) i7-11700F @ 2.50GHz

[2] NVIDIA GeForce RTX 3060

[7] M. D. Hossain, H. Inoue, H. Ochiai, D. Fall, and Y. Kadobayashi, "Lstm-based intrusion detection system for in-vehicle can bus communications," *IEEE Access*, vol. 8, pp. 185 489–185 502, 2020.

[8] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*. pmlr, 2015, pp. 448–456.

[9] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA, 2015.

[10] Y. Li, "An overview of the dsrc/wave technology." Springer, 2012, pp. 544–558.

[11] Y. Lin, C. Chen, F. Xiao, O. Avatefipour, K. Alsubhi, and A. Yunianta, "An evolutionary deep learning anomaly detection framework for in-vehicle networks-can bus," *IEEE Transactions on Industry Applications*, 2020.

[12] V. Mannoni, V. Berg, S. Sesia, and E. Perraud, "A comparison of the v2x communication systems: Its-g5 and c-v2x," in *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*. IEEE, 2019, pp. 1–5.

[13] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[14] S. Rajapaksha, H. Kalutarage, M. O. Al-Kadri, A. Petrovski, G. Madzudzo, and M. Cheah, "Ai-based intrusion detection systems for in-vehicle networks: A survey," *ACM Computing Surveys*, vol. 55, no. 11, pp. 1–40, 2023.

[15] M. Rosenblatt, "Remarks on some nonparametric estimates of a density function," *The annals of mathematical statistics*, pp. 832–837, 1956.

[16] E. Seo, H. M. Song, and H. K. Kim, "Gids: Gan based intrusion detection system for in-vehicle network," in *2018 16th Annual Conference on Privacy, Security and Trust (PST)*. IEEE, 2018, pp. 1–6.

[17] L. Shen, Z. Li, and J. Kwok, "Timeseries anomaly detection using temporal hierarchical one-class network," *Advances in Neural Information Processing Systems*, vol. 33, pp. 13 016–13 026, 2020.

[18] H. M. Song and H. K. Kim, "Self-supervised anomaly detection for in-vehicle network using noised pseudo normal data," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 2, pp. 1098–1108, 2021.

[19] H. M. Song, J. Woo, and H. K. Kim, "In-vehicle network intrusion detection using deep convolutional neural network," *Vehicular Communications*, vol. 21, p. 100198, 2020.

[20] Tencent Security Keen Lab, "Experimental Security Assessment of Mercedes-Benz-Cars," https://keenlab.tencent.com/en/.

[21] Y. Wang, Y. Lai, Y. Chen, J. Wei, and Z. Zhang, "Transfer learning-based self-learning intrusion detection system for in-vehicle networks," *Neural Computing and Applications*, pp. 1–17, 2023.

[22] Y. y. Yang, C. Zhang, T. Zhou, Q. Wen, and L. Sun, "DCdetector: Dual attention contrastive representation learning for time series anomaly detection," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3033–3045. [Online]. Available: http://arxiv.org/abs/2306.10347

[23] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, no. 9, 2023, pp. 11 121–11 128.