SCIMI: Semantic-Aware Collaborative Intrusion Detection for Multi-domain In-Vehicle Networks

Chang Zhu[1,2], Xiaohang Wang[1,2,*], Abla Smahi[1,2], Kaiwei Wu[1,2], Yinhe Shen[1,2], Yingtao Jiang[3], Amit Kumar Singh[4], and Kui Ren[1,2]
[1]State Key Laboratory of Blockchain and Data Security, Zhejiang University, Hangzhou, China
[2]Hangzhou High-Tech Zone (Binjiang), Institute of Blockchain and Data Security, China
[3]University of Nevada, Las Vegas, NV, USA
[4]University of Essex, Colchester, Essex, UK

Abstract

Modern vehicles integrate multiple in-vehicle networks (IVNs), such as Controller Area Network (CAN), Automotive Ethernet, and host systems, to support advanced automation and connectivity. While these networks enable new functionalities, they also expand attack surface across multiple domains. Sophisticated multi-stage attacks exploit these surfaces to propagate covertly, and existing vehicular IDS systems cannot detect them as these attacks only cause minor changes in IVNs. This paper presents SCIMI, a Semantic-aware Collaborative Intrusion detection system for Multi-domain In-vehicle networks, which learns inter-domain causal relationships to identify subtle, coordinated attacks that evade single domain intrusion detection systems. SCIMI aligns asynchronous traffic patterns through a dual window temporal model, extracts semantic features via domain adaptive encoders, and applies cross domain attention to correlate suspicious behaviors. A dataset is also released for collaborative IDS. Experiments show that SCIMI achieves over 99% F1-score with a 17% improvement over state-of-the-art methods, while maintaining low false positive alarms and real time inference efficiency, underscoring its applicability to real time automotive cyber-security.

Keywords

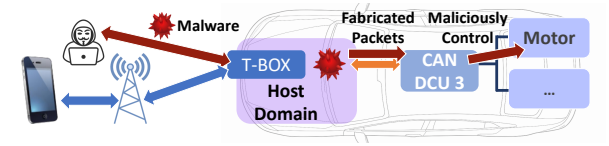
In-vehicle Network, Intrusion Detection, Multi-stage Attacks

ACM Reference Format:
Chang Zhu[1,2], Xiaohang Wang[1,2,*], Abla Smahi[1,2], Kaiwei Wu[1,2], Yinhe Shen[1,2], Yingtao Jiang[3], Amit Kumar Singh[4], and Kui Ren[1,2]. 2026. SCIMI: Semantic-Aware Collaborative Intrusion Detection for Multi-domain In-Vehicle Networks. In *63rd ACM/IEEE Design Automation Conference (DAC '26), July 26–29, 2026, Long Beach, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3770743.3803979

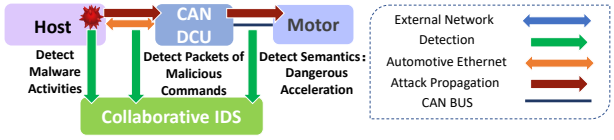
1 Introduction

The rapid rise of automation, connectivity, and user-centric services in modern vehicles has led to increasingly complex in-vehicle networks (IVNs), such as the Controller Area Network (CAN), Automotive Ethernet, and host systems. In-Vehicle Infotainment (IVI) systems further extend connectivity to external networks such as WiFi and Bluetooth, providing advanced functionality but also exposing vehicles to new cyber-security risks. This growing interconnectivity has resulted in broader and more diverse attack surface, as evidenced by recent incidents such as the Hyundai Ioniq hack [36] and the Subaru hack [5], where attackers remotely accessed vehicle host and executed malicious operations across other IVN domains. Figure 1a illustrates a schematic of the Hyundai Ioniq hack [36]. In this attack, the attacker hacks into the vehicle host via a wireless interface and implants malware that excuses to launch malicious control operations across multiple IVN domains.

The ISO/SAE 21434 standard [16] specifies the capabilities required for vehicles to defend against attacks traversing multiple internal networks. Multi-stage attacks, however, can achieve their goals by injecting minimal traffic into IVN subdomains. As shown in Figure 2, such minor injections barely alter CAN traffic, with a Euclidean similarity of 99.4% between the original and injected sequences. Most existing IVN Intrusion Detection Systems (IDSs) [11–13, 22–24, 33, 34, 38, 43] focus on traffic-pattern anomalies and thus fail to detect these subtle multi-stage attacks.



(a) Multi-Stage Attack Example



(b) Basic Idea of Collaborative IDS for multiple IVNs

Figure 1: Multi-Stage Attack Example and Defense Idea

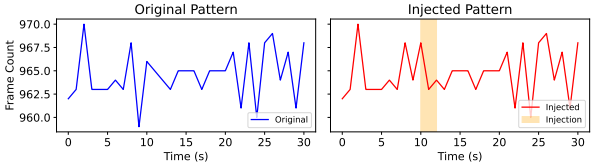


Figure 2: A Comparison of Normal CAN Traffic Pattern and Injected Pattern

To address this limitation, we make two key observations. First, semantic awareness can reveal suspicious IVN activities. For instance, if a vehicle is moving at 120 km/h while the motor continues receiving acceleration commands via CAN frames, this behavior is suspicious. Although these injections cause almost no change in traffic patterns, they alter IVN semantics derived from ISO 26262 safety specifications [15] as listed in [44]. Second, cross-domain

* Xiaohang Wang is the corresponding author.

This work is licensed under a Creative Commons Attribution 4.0 International License.
DAC '26, Long Beach, CA, USA
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2254-7/2026/07
https://doi.org/10.1145/3770743.3803979

correlation can help trace the information flow that attributes an intrusion, thereby distinguishing true attacks from benign anomalies. For example, if the abnormal acceleration frames originate from the host via Ethernet, they are identified as attacks, since normal acceleration frames should come from the accelerator pedal. These observations indicate that multi-stage attacks, though indistinguishable from normal traffic patterns, disrupt the semantic consistency and cross-domain information flow within IVNs. Leveraging these learnable semantic and inter-domain dependencies, a collaborative, semantics-driven IDS across multiple IVN domains can be developed, as illustrated in Figure 1b.

However, developing such a collaborative IDS faces two major challenges. The first is the enormous search space for cross-domain event correlation. Multi-stage attacks manifest as suspicious behaviors in distinct semantic features across multiple IVN domains, making it difficult to identify causal relations among events in the huge search space. Correlating Ethernet or host events with their corresponding CAN frames is particularly complex, as each second may involve over 1,000 CAN frames and 5,000 Ethernet packets. The second challenge is temporal disparity. Data from different domains are generated at vastly different rates; host events may occur once per second, whereas CAN and Ethernet transmit thousands of packets in the same interval.

To address those challenges, we propose a Semantic-Aware Collaborative Intrusion Detection System for Multi-domain In-vehicle networks (SCIMI). SCIMI adopts a dual sliding window design to align heterogeneous temporal dynamics. Domain adaptive encoders compress high frequency information from CAN and Ethernet while preserving low frequency host data, producing unified short window embeddings. These embeddings align time scales and avoid brute force per packet matching. A cross-domain attention mechanism prioritizes correlated embeddings, enabling efficient causal dependency learning and reducing the event correlation search space.

This paper makes the following key contributions:

(1) We propose SCIMI, a semantic-aware collaborative IDS that correlates multiple IVN domains for threat attribution of being a multi-stage attack.

(2) We construct and release a labeled dataset collected from a production vehicle containing multi stage attacks propagating across CAN, Ethernet, and host domains [44].

(3) We design a dual sliding window and cross domain attention framework that aligns temporal dynamics and learns inter-domain dependencies for efficient multi domain intrusion detection.

(4) We demonstrate through real vehicle experiments that SCIMI achieves over 99% F1 score across all attack types, a 17% improvement over state of the art methods, while maintaining real time efficiency on commercial automotive platforms.

2 Related Work

Within a vehicle, there are IDSs for each domain IVN. CAN IDSs can be classified into the following types. Traditional machine learning models, such as support vector machines (SVM), decision trees, and k-nearest neighbors (KNN), have been widely applied to CAN intrusion detection [2, 3, 27, 29]. Deep learning methods for CAN intrusion detection include image-based Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and large language models (LLMs) [11, 13, 22, 23, 33, 34, 43]. In addition, unsupervised methods—particularly autoencoders—have been adopted to reduce reliance on labeled data [12, 24, 38]. Some studies explore signature-based non-ML methods that leverage physical characteristics of in-vehicle networks, such as transmission timing, voltage levels, or data semantics [9, 21, 26, 39–41].

Recent studies have applied deep learning to Automotive Ethernet intrusion detection, including CNN-based models, multistage frameworks combining classical classifiers with neural networks, and autoencoder-based unsupervised approaches [1, 6, 10, 18].

Vehicle host systems serve as central hubs for connectivity and computation, but their complex software stacks introduce extensive attack surfaces. Provenance graph–based methods analyze system audit logs for anomaly detection [4, 30, 37], which are effective for long-term threats but less applicable to the rapid, concentrated nature of IVN attacks and often require heavy computation. Alternative approaches use system metrics such as CPU or memory usage to detect abnormal behaviors [14, 20, 31].

Only a few studies explore collaborative approaches for IVN intrusion detection [28, 42]. They focus on the collaboration between the IVN and external networks, such as cloud platforms, instead of intra-IVN cross-domain collaboration.

Limitations of Existing IDSs: Existing IDSs work for a single IVN domain, detecting intrusions from IVN traffic patterns without awareness of semantics and causality between events in multiple IVN domains. Thus, they are unable to detect the aforementioned multi-stage attacks.

3 Threat Model

3.1 IVN architecture

As shown in Figure 3, an IVN in a smart vehicle (e.g., Leapmotor C10 or Geely Galaxy L6) includes three domains: CAN, Ethernet, and host. CAN links Electronic Control Units (ECUs) via Domain Control Units (DCUs), while automotive Ethernet enables inter-domain communication. The host integrates IVI and autonomous-driving modules and connects to DCUs via Ethernet, which then relay commands to ECUs over CAN.

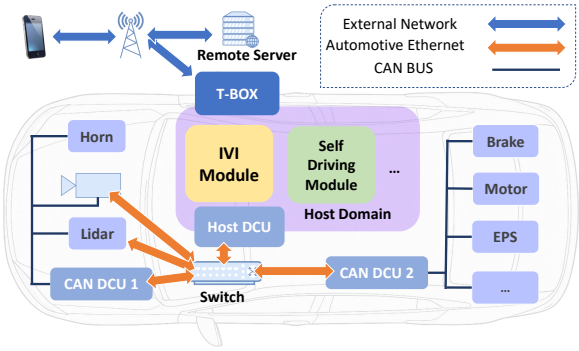


Figure 3: IVN Architecture

3.2 Attack Procedure

The attacker exploits various vulnerabilities to implant malware into the vehicle and gain potential control. For example, an attacker can use a Man-In-The-Middle (MITM) attack [8] to disguise malware as legitimate software and implant it into the host. Subsequently, by exploiting vulnerabilities [25] in the vehicle host system, the attacker can execute arbitrary code.

In the second phase, the attacker activates the implanted malware on the host to perform malicious operations, such as issuing unauthorized commands, injecting forged data, or disrupting network functionality. These actions may propagate across different IVN domains, forming cross-domain attack paths [19]. Enumerating all possible attack chains is impractical and beyond the scope

of this work. Based on typical IVN structures and known attacks, we focus on three common types of cross-domain attack paths, as shown in Figure 4:

Path 1: The attacker sends forged control commands from the host over Ethernet, which are received by a DCU, and translates them into CAN frames that directly affect in-vehicle actuators.

Path 2: The attacker modifies or injects sensor data (e.g., from radar or cameras) transmitted over Ethernet. This manipulated data is received by perception or decision modules, potentially misleading the vehicle's autonomous or assisted driving algorithm, thereby causing anomalous behaviors in the CAN domain.

Path 3: In path 3, malicious traffic is sent from the host to critical Ethernet components such as switches or gateways, aiming to affect the availability and stability of in-vehicle communication.

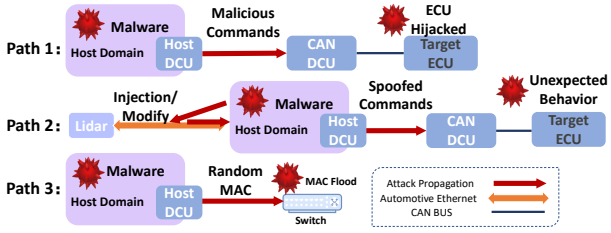


Figure 4: Attack Path in IVN

3.3 Attack Types

Type 1. Fuzzing-Based Probing. Without knowledge of IVN semantics (e.g., DBC mappings or Ethernet formats), the attacker sends random messages to infer protocols and find exploitable patterns [33] through path 1 or 3.

Type 2. Unauthorized Vehicle Control. The attacker issues unauthorized host commands via path 1, relayed over Ethernet and executed by CAN ECUs, potentially causing unsafe behaviors such as unintended acceleration or braking [36].

Type 3. Replay Attack. By recording and replaying CAN or Ethernet traffic, the attacker triggers hazardous vehicle actions without reverse engineering [33] via path 1.

Type 4. Sensor Signal Spoofing. The attacker injects falsified sensor data into Ethernet (path 2), misleading the autonomous driving perception module, e.g., fabricating obstacles to induce unnecessary braking [10].

Type 5. MAC Flooding. Exploiting path 3, the attacker floods the network with frames using spoofed MAC addresses [10], saturating the switch MAC table and disrupting time-critical communications .

4 The Proposed SCIMI Method

4.1 Overview

Figure 5 illustrates the overall architecture of our proposed method, SCIMI, which consists of three stages: preprocessing, intra-domain representation learning, and inter-domain correlation. In preprocessing, domain-specific feature extractors are used to convert raw CAN, Ethernet, and host data into semantic time series. In the second stage, the temporal disparity challenge in Section 1 is solved by a dual-window structure. In short windows, domain-adaptive encoders are used to learn local temporal semantics and perform temporal compression. This produces unified embeddings in the time dimension across domains. In long windows, a self-attention mechanism is applied to capture long-term dependencies. Finally, cross-domain attention is used to learn dependencies between domains by highlighting embeddings of strongly correlated events in multiple domains, thereby mitigating the search space challenge in Section 1. The attention-weighted representation is classified by a Multilayer Perceptron (MLP) to identify intrusions, reducing correlation search complexity.

4.2 Preprocessing

SCIMI utilizes domain-specific feature extractors to preprocess IVN raw data into semantic sequences. The purpose of this semantic extraction is to detect suspicious behaviors.

4.2.1 CAN Semantic Extraction. SCIMI utilizes payload context and inter-signal-value delays from CAN traffic. Payload bytes in CAN frames can be decoded into payload signals, such as vehicle speed or brake status, which can be achieved either using a DBC file or through reverse engineering [7]. The CAN semantic sequence is $\mathbf{x}^{can}[i],\ i \in [1, 2, \ldots, n]$. Each vector has two sub-vectors:

$$\mathbf{x}^{can}[i] = \langle\langle v_1[i], v_2[i], \ldots, v_{m_{sig}}[i]\rangle, \langle \Delta\tau_1^{can}[i], \Delta\tau_2^{can}[i], \ldots, \Delta\tau_{m_{sig}}^{can}[i]\rangle\rangle \quad (1)$$

where $v_j[i]$ is the ith value of the payload signal j, and m_{sig} is the total number of signals decoded from all CAN buses. $\Delta\tau_j^{can}[i] = \tau_j^{can}[i] - \tau_j^{can}[i-1]$ is the inter-signal-value delay, where $\tau_j^{can}[i]$ and $\tau_j^{can}[i-1]$ are the timestamps of two consecutive values from the signal j.

4.2.2 Ethernet Semantic Extraction. SCIMI extracts transmission semantics from Automotive Ethernet to detect possible injection. The Ethernet semantic sequence is $\mathbf{x}^{eth}[i],\ i \in [1, 2, \ldots, n]$. Each semantic vector is extracted from an Ethernet packet:

$$\mathbf{x}^{eth}[i] = \langle e_{src}^{mac}[i], e_{dst}^{mac}[i], e_{src}^{ip}[i], e_{dst}^{ip}[i], e_{src}^{port}[i], e_{dst}^{port}[i], e^{proto}[i], \Delta\tau^{eth}[i]\rangle \quad (2)$$

where $e^{mac}[i]$, $e^{ip}[i]$, $e^{port}[i]$, and $e^{proto}[i]$ are MAC addresses, IP addresses, port numbers, and the protocol identifier mapped by a hash table. $\Delta\tau^{eth}[i] = \tau^{eth}[i] - \tau^{eth}[i-1]$ is the inter-packet delay, where $\tau^{eth}[i]$ and $\tau^{eth}[i-1]$ are the timestamps of two adjacent packets.

4.2.3 Host Semantic Extraction. SCIMI extracts system and process semantics from the host to detect the activities of malware. The host semantic sequence is $\mathbf{x}^{host}[i],\ i \in [1, 2, \ldots, n]$. Each semantic vector is extracted from the system monitor information at time step i:

$$\mathbf{x}^{host}[i] = \langle h^{cpu}[i], h^{mem}[i], h^{apc}[i], h^{tcu}[i], h^{pce}[i], h^{usm}[i], h^{usv}[i], h^{sc}[i], h^{mtc}[i], h^{atc}[i], h^{npc}[i], h^{tpc}[i]\rangle \quad (3)$$

where the elements in the vector are CPU usage, memory consumption, network I/O load, active process count, top-5 process CPU usage, process CPU usage entropy, user/system ratio mean & variance, state counts, maximum & average thread count, and new & terminated process count, respectively. Malicious processes induce abnormal resource usage and activity on the host, which can be detected using the semantic features described above.

4.3 Intra-domain Representation Learning

SCIMI performs intra-domain representation learning on the semantic sequences of each domain.

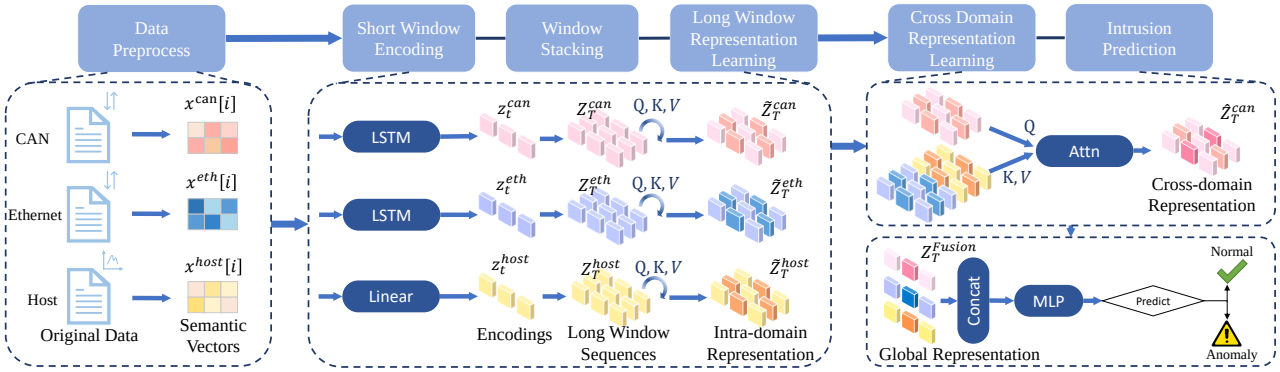


Figure 5: Overview of SCIMI

4.3.1 Dual Sliding Window Mechanism. A dual sliding window mechanism is used to solve the event timing gap problem. With short windows, SCIMI uses domain-specific models to encode semantic sequences into compressed representation embeddings, which align different domains with varying temporal scales and make suspicious behaviors distinguishable in the embedding space. With long windows, a self-attention mechanism is applied to capture long-term dependencies, making temporally correlated embedding segments distinguishable and focusing on the most relevant windows to narrow down the search space of the temporal dimension.

4.3.2 Short Window Semantic Encoding. For the semantic sequences extracted in the preprocessing stage, SCIMI employs domain-specific encoders to distinguish suspicious behaviors in an embedding space. Besides, such encoding can compress sequences from high-frequency domains (CAN/Ethernet) into embedding vectors, addressing the inter-domain timing gap problem and thereby reducing the search space.

Firstly, SCIMI employs the short sliding window for semantic sequences of each domain. $X_t^d = \mathbf{x}^d[i],\ i \in t$ is the sub-sequence of domain $d \in \{can, eth, host\}$ in the time window t. The length of t is one second, aligned with the host metrics' fastest sampling rate of 1 Hz. CAN and Ethernet traffic are dense within short windows and exhibit temporally correlated patterns; therefore, SCIMI uses LSTM-based short window encoders to capture short-term dependencies and dynamic variations, $\mathbf{z}_t^d = LSTM(X_t^d)$, where $d \in \{can, eth\}$ notes the domains. In contrast, Host-domain data produce one observation per short window, so a linear projection is used to align its short window representation with other domains, while temporal dependencies for host are modeled across long windows rather than within short windows: $\mathbf{z}_t^{(host)} = Linear(X_t^{(host)})$.

4.3.3 Intra-domain Attention Mechanism in Long Windows. SCIMI applies intra-domain self-attention over long temporal windows to capture meaningful semantic patterns within each domain. Some anomalous behaviors may not be evident in short windows but induce long-term deviations in CAN signals—for example, a malicious acceleration command that causes sustained abnormal variations in subsequent CAN messages. By assigning higher attention weights to embeddings that preserve such long-range temporal dependencies, the model emphasizes relevant semantic patterns, making normal and abnormal segments more distinguishable.

SCIMI builds the long window embedding sequence by stacking short window embedding vectors: $Z_T^d = [\mathbf{z}_t^d, \mathbf{z}_{t+1}^d, \ldots, \mathbf{z}_{t+L-1}^d]$, where T is the long window time step and L is its length, and $d \in \{can, eth, host\}$. We adopt the following generic attention formulation, where Q, K, and V denote the query, key, and value projections of the inputs:

$$\mathrm{Attn}(Q, K, V) = \mathrm{softmax}\Big((QK^\top)/\sqrt{D}\Big)\,V. \tag{4}$$

In each domain, the embedding sequence Z_T^d is linearly projected into query, key, and value matrices using learnable parameter matrices $W_Q^{d,self}$, $W_K^{d,self}$, and $W_Q^{d,self}$, forming Q_{self}^d, K_{self}^d, V_{self}^d for the multi-head self-attention layer. By applying self-attention to Z_T^d, the model captures the temporal dependencies within each domain, producing the embedding sequences refined by self-attention:

$$\tilde{Z}_T^d = \mathrm{Attn}(Q_{self}^d, K_{self}^d, V_{self}^d). \tag{5}$$

4.4 Inter-domain Correlation

In this stage, SCIMI fuses information from multiple IVN domains to capture abnormal inter-domain information flows that reflect multi-stage attack chains.

4.4.1 Cross-domain Attention Mechanism. Multi-stage attacks induce suspicious behaviors that are dispersed across domains, creating a large search space when attempting to correlate semantic features across IVN domains. To detect abnormal events caused by such attacks, SCIMI applies cross-domain attention on the refined embedding sequences, enhancing the numerical separation between normal and abnormal segments in the embedding space. It classifies sequences accurately without performing exhaustive searches across all domain features. To achieve this, we perform cross-attention on the embedding sequences $\tilde{Z}_T^d$ refined by self-attention. For each domain d, we compute the query $Q_{cross}^d = \tilde{Z}_T^d\, W_Q^{d,cross}$ from its own embedding sequence. The key K_{cross}^d and value V_{cross}^d are computed from the embedding sequences $\langle \tilde{Z}_T^{d'}, \tilde{Z}_T^{d''} \rangle$ of the other two domains d' and d'', where $d, d', d'' \in \{can, eth, host\}$. Then we perform the cross-attention computation:

$$\hat{Z}_T^d = \mathrm{Attn}(Q_{cross}^d, K_{cross}^d, V_{cross}^d). \tag{6}$$

It produces embedding sequences refined by cross-domain attention $\hat{Z}_T^d$, where correlations caused by multi-stage attacks are assigned higher weights, making them distinguishable from normal sequences. As shown in Figure 6, higher attention scores are assigned to CAN embeddings when correlated events are triggered in host or Ethernet domains.

4.4.2 Inference. Since the correlations of anomalous behaviors in multiple domains are already encoded by the embeddings, a lightweight multi-layer perceptron (MLP) can classify them as normal

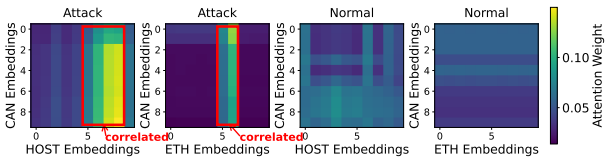


Figure 6: The Cross Attention Score Heat Domain Comparison between Normal and Attack Samples

or attacks. This fused representation $Z_T^{\text{fusion}} = \langle \hat{Z}_T^{can} ; \hat{Z}_T^{eth} ; \hat{Z}_T^{host} \rangle$ is passed through a MLP to produce an anomaly score at each short time step t: $s_t = \sigma\Big(\text{MLP}\big(Z_T^{\text{fusion}}(t)\big)\Big)$, where $s_t \in \mathbb{R}$ is the likelihood of intrusion. The anomaly score is mapped to a $(0, 1)$ interval and compared to a threshold to yield the final binary decision indicating the presence or absence of an intrusion.

4.5 Model Training

We adopt a supervised learning paradigm to train SCIMI. By using data labeled with suspicious behaviors caused by intrusions, it's trained to distinguish between normal and abnormal sequences. The model is optimized using the Adam optimizer. We employ the binary cross-entropy with logits loss function and incorporate an early stopping strategy to mitigate overfitting.

5 Evaluation

5.1 Setups

We implemented our work in Python and used PyTorch for the neural network components. Our model is trained on a machine equipped with an NVIDIA H100. The inference is performed on a Raspberry Pi 4B (BCM2711 CPU, 4 GB RAM) and a Leapmotor C10 test vehicle (Qualcomm 8295). The dataset is divided into training, validation, and test sets in a 6:2:2 ratio. Training hyper-parameters are listed in Table 1.

Table 1: Experiment Setup

Parameter	Value	Parameter	Value
Epochs	80	LSTM hidden dim	64
Early stop patience	5	LSTM layers	1
Learning rate	0.0001	Host Linear dim	32
Batch size	64	MLP dim	96
Attention Heads	4	Attention dim	32

5.2 Dataset and Attack Implementation

Existing public datasets [17, 32] are limited to single-domain scenarios and lack multi-stage cross-domain attacks, making them unsuitable for evaluating our method. We therefore collected a one-hour dataset on an unmanned vehicle with cloud-point Radar (Teemo Autobot W1 [35]), implementing six multi-stage attacks as described in Section 3. Malware was implanted into the host through a MITM attack, injecting malicious data into Automotive Ethernet and CAN, while Radar attacks were implemented via fake frame injections to the Ethernet gateway. Experiments covered various vehicle behaviors, including parking, driving, and reversing. Enough normal data is included to balance the dataset. CAN dumps, Ethernet tcpdump traffic, and host Atop metrics are recorded. Details are summarized in Table 2, and the dataset has been released in [44].

Table 2: Details of Dataset

Attack Case	Scenario	Attack Path	Attack Type	Length
AC1	Vehicle Control	Path 1	Type 2	10 min
AC2	Vehicle Control*	Path 1	Type 2	10 min
AC3	Replay	Path 1	Type 3	15 min
AC4	Fuzzing	Path 1&3	Type 1	10 min
AC5	Sensor Spoof	Path 2	Type 4	10 min
AC6	MAC Flood	Path 3	Type 5	5 min

* The scenarios of AC1 and AC2 differ in the types of ECUs they control.

5.3 Baseline IDS methods

To evaluate our work, we reproduced several existing methods as baselines on our dataset. As shown in Table 3, some of these baselines are SOTA approaches in their respective domains, while others are representative of different methodological categories. To the best of our knowledge, no IDS has been specifically designed for in-vehicle host domains.

Table 3: Baseline IDS Methods Summary

Method	Domain	Approach	Reason
Hydl [23]	CAN	CNN + LSTM	SOTA
DCNN [34]	CAN	Deep CNN	Representative
ACGAN [43]	CAN	AC-GAN	Representative
TOW [10]	Ethernet	Wavelet + CNN	SOTA
Multi [6]	Ethernet	Pruned CNN	Representative

5.4 Performance Evaluation of SCIMI

Table 4 presents the overall comparison results. Compared with the baseline models, SCIMI achieves over 99% F1-score in every attack case. Compared to Hydl, the best IDS model that only uses CAN domain data, SCIMI achieves improvements of 5.4%, 27.9%, and 17.2% in precision, recall, and F1-score on average. Although still outperformed by SCIMI, Ethernet-based IDS models demonstrate slightly better results. SCIMI achieves an improvement of 8.3%, 20.0%, and 17.0% in precision, recall, and F1-score compared to the best Ethernet-based method TOW. As shown in Table 7, SCIMI has the lowest false positive rate, decreased by 73% compared to the best existing method Hydl. Existing methods rely on traffic patterns without correlating global IVN semantic information, and thus perform poorly in detecting sophisticated multi-stage attacks. Table 6 shows that SCIMI achieves performance comparable to the SOTA methods on public datasets.

5.5 Ablation Analysis

To assess the contribution of each component in SCIMI, we performed ablation studies as shown in Table 5. First, the sensitivity of domains was examined with CAN-ONLY, ETH-ONLY, and HOST-ONLY models, where the semantic sequences of non-selected domains were replaced with zero tensors. In Table 5, w/o(without)-CAN, w/o-ETH, and w/o-HOST are methods without CAN, Ethernet, or Host domain semantic sequences, respectively. Second, the importance of attention mechanisms was evaluated by bypassing specific modules, yielding w/o-SA (self-attention), w/o-CA (cross-attention), and w/o-BA (both) models to evaluate the contributions of semantic extraction and cross-domain correlation.

In Table 5, one can see that: (1) Cross-domain semantic correlation brings substantial improvements. Correlating semantics from two domains instead of one increases precision, recall, and F1 by 16.5%, 16.8%, and 21.8% on average, while incorporating all three domains yields further gains of 8.1%, 4.1%, and 6.6%. These results

Table 4: Performance of Different IDS Methods on Various Attack Scenarios

Method	AC 1			AC 2			AC 3			AC 4			AC 5			AC 6		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
Hydl	0.923	0.709	0.817	0.944	0.758	0.822	0.914	0.639	0.752	0.997	0.985	0.991	0.964	0.817	0.884	-	-	-
DCNN	0.889	0.569	0.694	0.865	0.509	0.641	0.830	0.538	0.653	0.876	0.524	0.656	0.877	0.682	0.767	-	-	-
ACGAN	0.295	0.626	0.401	0.295	0.578	0.387	0.277	0.632	0.385	0.272	0.500	0.352	0.306	0.622	0.410	-	-	-
TOW	0.751	0.654	0.699	0.873	0.898	0.885	0.915	0.921	0.918	0.991	0.926	0.957	0.890	0.341	0.493	0.979	0.966	0.972
Multi	0.802	0.789	0.796	0.899	0.964	0.930	0.936	0.970	0.953	0.990	0.968	0.979	0.923	0.335	0.492	0.983	0.970	0.977
SCIMI	**1.000**	**1.000**	**1.000**	**0.997**	**0.998**	**0.998**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**0.997**	**0.999**	**0.998**

Table 5: Performance of Different Ablation Models on Various Attack Scenarios

Method	AC 1			AC 2			AC 3			AC 4			AC 5			AC 6		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
CAN-ONLY	0.813	0.848	0.830	0.840	0.421	0.560	0.972	0.957	0.965	0.916	0.744	0.821	0.844	0.486	0.617	0.738	0.951	0.831
ETH-ONLY	0.495	1.000	0.662	0.551	1.000	0.710	0.449	1.000	0.620	0.877	1.000	0.935	0.736	1.000	0.848	0.623	0.990	0.765
HOST-ONLY	0.930	0.910	0.920	0.898	0.718	0.798	0.865	0.863	0.864	0.844	0.703	0.768	0.918	0.234	0.372	0.965	0.974	0.969
w/o-CAN	0.784	1.000	0.879	0.684	1.000	0.812	0.861	0.967	0.911	0.902	1.000	0.948	0.975	1.000	0.987	0.848	0.992	0.914
w/o-ETH	1.000	0.885	0.939	0.966	0.990	0.978	0.983	0.990	0.986	1.000	0.754	0.860	0.726	0.844	0.780	0.933	0.902	0.917
w/o-HOST	1.000	0.967	0.983	0.989	1.000	0.994	1.000	1.000	1.000	1.000	1.000	1.000	0.997	1.000	0.998	0.985	0.991	0.988
w/o-BA	1.000	0.909	0.952	0.981	0.989	0.985	1.000	0.990	0.995	0.946	0.996	0.970	0.896	0.953	0.924	0.913	0.966	0.939
w/o-SA	0.918	0.998	0.956	0.994	1.000	0.997	0.981	0.988	0.984	1.000	0.999	0.999	1.000	1.000	1.000	0.978	0.993	0.985
w/o-CA	1.000	0.960	0.980	0.990	0.998	0.994	1.000	0.996	0.998	1.000	0.930	0.964	0.994	0.786	0.878	0.995	0.936	0.965
SCIMI	**1.000**	**1.000**	**1.000**	**0.997**	**0.998**	**0.998**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**0.997**	**0.999**	**0.998**

Table 6: Performance Comparison on Two Public Datasets.

AEID [17]				Car-Hacking [32]			
Methods	SCIMI	TOW	Multi	SCIMI	Hydl	DCNN	ACGAN
Prec.	0.9461	0.9625	0.9767	0.9994	0.9364	0.9980	0.9678
Rec.	0.9731	0.8999	0.9349	0.9941	0.9808	0.9995	0.9671

Table 7: False Positive Rate and False Negative Rate Performance

Method	SCIMI	Hydl	DCNN	ACGAN	TOW	Multi
FPR.	0.003	0.011	0.029	0.492	0.037	0.031
FNR.	0.001	0.218	0.436	0.408	0.216	0.167

indicate that integrating semantic information from multiple domains enhances the model's ability to capture coordinated attack behaviors. (2) ETH-only model has high recall but poor precision, as it tends to flag broad suspicious traffic; confirming actual attacks, however, requires semantic cues from CAN. (3) Attention mechanisms further reinforce cross-domain collaboration by explicitly modeling inter-domain dependencies and emphasizing correlated cues, contributing an additional 3.8% improvement in F1 score.

Table 8: The Average Inference Time to Process IVN Data Generated in 1 second

Method	SCIMI	Hydl	DCNN	ACGAN	TOW	Multi
Time (s)	0.755	0.655	0.739	0.443	18.327	17.052

5.6 Overhead Analysis

To evaluate the overhead of our approach, we measured the runtime of all methods on a Raspberry Pi CPU. Table 8 shows the average inference time for each method to process IVN data generated in 1 second. SCIMI is much faster than existing Ethernet-based methods, while slightly slower than CAN-based methods. Although ACGAN runs faster, its detection performance is very low as shown in Table 4. A method is considered deployable in real time if its inference time is shorter than the data generation interval (1 second).

We further ported our method to the host system of a Leapmotor C10 test vehicle with Qualcomm 8295 MCU. The operating system in this test car provides APIs for CAN/Ethernet packets reading. On the C10, CAN and Ethernet messages arrive every 1 ms and 0.2 ms on average. SCIMI achieves inference times of 0.8 ms and 0.12 ms, satisfying real-time requirements, with peak CPU and memory usage of 12.6% and 8.7%, respectively. Running SCIMI increases host power consumption only slightly, from 37.8 W (idle) to 38.9 W, remaining lightweight.

6 Conclusion

This work introduces SCIMI, a semantic-aware collaborative intrusion detection system for multi-domain in-vehicle networks. By capturing temporal and causal relations across IVN domains, SCIMI delivers robust, interpretable anomaly detection with low computational overhead. Real-vehicle experiments validate its practicality and over 99% precision. Future work will extend SCIMI toward unsupervised, adaptive learning for detecting emerging attacks.

Acknowledgments

This work was supported in part by the key R&D programme of Zhejiang Province No. 2024C01012, in part by the National Natural Science Foundation of China under Grants 62374146 and 92373205, in part by the Key Technologies R&D Program of Jiangsu (Prospective and Key Technologies for Industry) under Grant BE2023005-2, in part by the National Key Research and Development Program of China No. 2023YFB4404404, in part by CIE-Smartchip research fund No. 2023-004, and in part by the Ascend AI technology stack.

References

[1] Natasha Alkhatib, Maria Mushtaq, Hadi Ghauch, and Jean-Luc Danger. 2022. Unsupervised network intrusion detection system for AVTP in Automotive Ethernet networks. In *IEEE Intelligent Vehicles Symposium*. 1731–1738.

[2] Ahmad Aloqaily, Emad E Abdallah, Hiba AbuZaid, Alaa E Abdallah, and Malak Al-hassan. 2025. Supervised Machine Learning for Real-Time Intrusion Attack Detection in Connected and Autonomous Vehicles: A Security Paradigm Shift. *Informatics* 12, 1 (2025).

[3] Bifta Sama Bari, Kumar Yelamarthi, and Sheikh Ghafoor. 2023. Intrusion Detection in Vehicle Controller Area Network (CAN) Bus Using Machine Learning: A Comparative Performance Study. *Sensors* 23, 7 (2023).

[4] Zijun Cheng, Qiujian Lv, Jinyuan Liang, Yan Wang, Degang Sun, Thomas Pasquier, and Xueyuan Han. 2024. Kairos: Practical intrusion detection and investigation using whole-system provenance. In *IEEE Security and Privacy*. 3533–3551.

[5] Sam Curry. 2025. *Hacking Subaru: Tracking and Controlling Cars via the STARLINK Admin Panel.* https://samcurry.net/hacking-subaru

[6] Luigi F Marques da Luz, Paulo Freitas de Araujo-Filho, and Divanilson R Campelo. 2024. Multi-stage deep learning-based intrusion detection system for Automotive Ethernet networks. *Ad Hoc Networks* 162 (2024).

[7] Daniel Frassinelli, Sohyeon Park, and Stefan Nürnberger. 2020. I know where you parked last summer: Automated reverse engineering and privacy analysis of modern cars. In *IEEE Security and Privacy*. 1401–1415.

[8] Rahul Gothwal, Gaurav Dharmani, Ramnik Singh Reen, and Eslam G. AbdAllah. 2023. Evaluation of Man-in-the-Middle Attacks and Countermeasures on Autonomous Vehicles. In *International Dependable Systems and Their Applications*. 502–509.

[9] Yoshihiro Hamada, Masayuki Inoue, Hiroshi Ueda, Yukihiro Miyashita, and Yoichi Hata. 2018. Anomaly-based intrusion detection using the density estimation of reception cycle periods for in-vehicle networks. *SAE Transportation Cybersecurity and Privacy* 1 (2018), 39–56.

[10] Mee Lan Han, Byung Il Kwak, and Huy Kang Kim. 2022. TOW-IDS: intrusion detection system based on three overlapped wavelets for Automotive Ethernet. *IEEE Information Forensics and Security* 18 (2022), 411–422.

[11] Markus Hanselmann, Thilo Strauss, Katharina Dormann, and Holger Ulmer. 2020. CANet: An unsupervised intrusion detection system for high dimensional CAN bus data. *Ieee Access* 8 (2020), 58194–58205.

[12] Thien-Nu Hoang and Daehee Kim. 2022. Detecting in-vehicle intrusion via semi-supervised learning-based convolutional adversarial autoencoders. *Vehicular Communications* 38 (2022).

[13] Md Delwar Hossain, Hiroyuki Inoue, Hideya Ochiai, Doudou Fall, and Youki Kadobayashi. 2020. LSTM-based intrusion detection system for in-vehicle CAN bus communications. *Ieee Access* 8 (2020), 185489–185502.

[14] Ren-Hung Hwang, Chieh-Lun Lee, Ying-Dar Lin, Po-Chin Lin, Hsiao-Kuang Wu, Yuan-Cheng Lai, and CK Chen. 2023. Host-based intrusion detection with multi-datasource and deep learning. *Journal of Information Security and Applications* 78 (2023), 103625.

[15] ISO/SAE International. 2018. *ISO/SAE 26262:2018 - Road vehicles – Functional safety.* International Organization for Standardization and SAE International.

[16] ISO/SAE International. 2021. *ISO/SAE 21434:2021 - Road Vehicles — Cybersecurity Engineering.* International Organization for Standardization and SAE International.

[17] Seonghoon Jeong, Boosun Jeon, Boheung Chung, and Huy Kang Kim. 2021. Convolutional neural network-based intrusion detection system for AVTP streams in automotive Ethernet-based networks. *Vehicular Communications* 29 (2021).

[18] Seonghoon Jeong, Huy Kang Kim, Mee Lan Han, and Byung Il Kwak. 2023. AERO: Automotive Ethernet real-time observer for anomaly detection in in-vehicle networks. *IEEE Industrial Informatics* 20, 3 (2023), 4651–4662.

[19] Pengfei Jing, Zhiqiang Cai, Yingjie Cao, Le Yu, Yuefeng Du, Wenkai Zhang, Chenxiong Qian, Xiapu Luo, Sen Nie, and Shi Wu. 2024. Revisiting automotive attack surfaces: A practitioners' perspective. In *2024 IEEE Security and Privacy*. 2348–2365.

[20] Cheryl Lee, Tianyi Yang, Zhuangbin Chen, Yuxin Su, Yongqiang Yang, and Michael R Lyu. 2023. Heterogeneous anomaly detection for software systems via semi-supervised cross-modal attention. In *IEEE/ACM Software Engineering*. 1724–1736.

[21] Seyoung Lee, Wonsuk Choi, Hyo Jin Jo, and Dong Hoon Lee. 2023. ErrIDS: An enhanced cumulative timing error-based automotive intrusion detection system. *IEEE Intelligent Transportation Systems* 24, 11 (2023), 12406–12421.

[22] Sifan Li, Yue Cao, Hassan Jalil Hadi, Feng Hao, Faisal Bashir Hussain, and Luan Chen. 2024. ECF-IDS: an enhanced cuckoo filter-based intrusion detection system for in-vehicle network. *IEEE Network and Service Management* (2024).

[23] Wei Lo, Hamed Alqahtani, Kutub Thakur, Ahmad Almadhor, Subhash Chander, and Gulshan Kumar. 2022. A hybrid deep learning based intrusion detection system using spatial-temporal representation of in-vehicle network traffic. *Vehicular Communications* 35 (2022), 100471.

[24] Stefano Longari, Daniel Humberto Nova Valcarcel, Mattia Zago, Michele Carminati, and Stefano Zanero. 2020. CANnolo: An anomaly detection system based on LSTM autoencoders for controller area network. *IEEE Network and Service Management* 18, 2 (2020), 1913–1924.

[25] MITRE Corporation. 2024. *CVE-2024-8358: Visteon Infotainment UPDATES_ExtractFile Command Injection Remote Code Execution Vulnerability.* https://www.cve.org/CVERecord?id=CVE-2024-8358

[26] Michael R Moore, Robert A Bridges, Frank L Combs, Michael S Starr, and Stacy J Prowell. 2017. Modeling inter-signal arrival times for accurate detection of CAN bus signal injection attacks: a data-driven approach to in-vehicle intrusion detection. In *Cyber and Information Security Research*. 1–4.

[27] Tarek Moulahi, Salah Zidi, Abdulatif Alabdulatif, and Mohammed Atiquzzaman. 2021. Comparative Performance Evaluation of Intrusion Detection Based on Machine Learning in In-Vehicle Controller Area Network Bus. *IEEE Access* 9 (2021), 99595–99605.

[28] Junman Qin, Yijie Xun, and Jiajia Liu. 2023. CVMIDS: Cloud–vehicle collaborative intrusion detection system for internet of vehicles. *IEEE Internet of Things Journal* 11, 1 (2023), 321–332.

[29] Rafi Ud Daula Refat, Abdulrahman Abu Elkhail, Azeem Hafeez, and Hafiz Malik. 2022. Detecting CAN bus intrusion by applying machine learning method to graph based features. In *Intelligent Systems Conference*. 730–748.

[30] Mati Ur Rehman, Hadi Ahmadi, and Wajih Ul Hassan. 2024. FLASH: A comprehensive approach to intrusion detection via provenance graph representation learning. In *IEEE Security and Privacy*. 3552–3570.

[31] Jose Ribeiro, Firooz B Saghezchi, Georgios Mantas, Jonathan Rodriguez, and Raed A Abd-Alhameed. 2020. Hidroid: prototyping a behavioral host-based intrusion detection and prevention system for android. *IEEE Access* 8 (2020), 23154–23168.

[32] Eunbi Seo, Hyun Min Song, and Huy Kang Kim. 2018. *Car-Hacking Dataset.* https://ocslab.hksecurity.net/Datasets/car-hacking-dataset

[33] Eunbi Seo, Hyun Min Song, and Huy Kang Kim. 2018. GIDS: GAN based intrusion detection system for in-vehicle network. In *privacy, security and trust*. 1–6.

[34] Hyun Min Song, Jiyoung Woo, and Huy Kang Kim. 2020. In-vehicle network intrusion detection using deep convolutional neural network. *Vehicular Communications* 21 (2020).

[35] Teemo. 2023. *AutoBots-W1.* https://www.teemoauto.top/Skateboard-chassis-W1.html

[36] Arya Voronova. 2022. *Hacker Liberates Hyundai Head Unit, Writes Custom Apps.* https://hackaday.com/2022/07/18/hacker-liberates-hyundai-head-unit-writes-custom-apps/

[37] Qi Wang, Wajih Ul Hassan, Ding Li, Kangkook Jee, Xiao Yu, Kexuan Zou, Junghwan Rhee, Zhengzhang Chen, Wei Cheng, Carl A Gunter, et al. 2020. You Are What You Do: Hunting Stealthy Malware via Data Provenance Analysis.. In *Network and Distributed Systems Security*.

[38] Pengcheng Wei, Bo Wang, Xiaojun Dai, Li Li, and Fangcheng He. 2023. A novel intrusion detection model for the CAN bus packet of in-vehicle network based on attention mechanism and autoencoder. *Digital Communications and Networks* 9, 1 (2023), 14–21.

[39] Lei Xue, Yangyang Liu, Tianqi Li, Kaifa Zhao, Jianfeng Li, Le Yu, Xiapu Luo, Yajin Zhou, and Guofei Gu. 2022. SAID: State-aware defense against injection attacks on in-vehicle network. In *USENIX Security*. 1921–1938.

[40] Yijie Xun, Zhouyan Deng, Jiajia Liu, and Yilin Zhao. 2023. Side channel analysis: A novel intrusion detection system based on vehicle voltage signals. *IEEE Vehicular Technology* 72, 6 (2023), 7240–7250.

[41] Yijie Xun, Yilin Zhao, and Jiajia Liu. 2021. VehicleEIDS: A novel external intrusion detection system based on vehicle voltage signals. *IEEE Internet of Things Journal* 9, 3 (2021), 2124–2133.

[42] Li Yang, Abdallah Moubayed, and Abdallah Shami. 2021. MTH-IDS: A multitiered hybrid intrusion detection system for internet of vehicles. *IEEE Internet of Things Journal* 9, 1 (2021), 616–632.

[43] Qingling Zhao, Mingqiang Chen, Zonghua Gu, Siyu Luan, Haibo Zeng, and Samarjit Chakrabory. 2022. CAN bus intrusion detection based on auxiliary classifier GAN and out-of-distribution detection. *ACM Embedded Computing Systems* 21, 4 (2022), 1–30.

[44] Chang Zhu. 2026. *MINIDS: Multi-domain In-Vehicle Network Intrusion Detection Dataset for SCIMI.* doi:10.5281/zenodo.19429093