Titre: Title:	Open-Set Anomaly Detection using Transformers: A Case Study on Anomaly Detection in J1939 CAN Bus
Auteur: Author:	Ranim Rahali
Date:	2026
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Rahali, R. (2026). Open-Set Anomaly Detection using Transformers: A Case Study on Anomaly Detection in J1939 CAN Bus [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/76023/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/76023/
Directeurs de recherche: Advisors:	Omar Abdul Wahab
Programme: Program:	Génie informatique

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Open-Set Anomaly Detection using Transformers: A Case Study on Anomaly Detection in J1939 CAN Bus

RANIM RAHALI
Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie informatique

Janvier 2026

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Open-Set Anomaly Detection using Transformers: A Case Study on Anomaly Detection in J1939 CAN Bus

présenté par **Ranim RAHALI**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Martine BELLAÏCHE, présidente
Omar ABDUL WAHAB, membre et directeur de recherche
Alejandro QUINTERO, membre

DEDICATION

To *my beloved mother and dearest father, for their endless love, support, and empowerment.*
To *my siblings, whose presence and guidance have been my pillars of strength.*
To *my friends, for believing in me and bringing laughter along the way.*
Thank you all, it is time to make you proud...

ACKNOWLEDGEMENTS

I would like to express my warmest thanks and gratitude to all those who have made this Master's thesis possible.

First, I extend my deepest gratitude to Dr.Omar Abdul Wahab and Dr.Adel Abusitta, for selecting me to work on this project and for their guidance, enouragements, and invaluable advice throughout, which helped me overcome many challenges.

I am also sincerely grateful to my professional supervisor at Thales, Simon Bellemare, for his expertise and support during my internship. Thank you for your time, patience, and inwavering support. I further extend my sincere appreciation to Audrey Lacoursiere Lamonde from Thales team for her valuable assistance and steady collaboration throughout this work.

I would like to thank MITACS and Thales for their generous support, which greatly contributed to the completion of this thesis.

Finally, I warmly thank the members of the jury for honoring me by evaluating this work, all my teachers at Poly, and everyone who supported me until the very end.

RÉSUMÉ

À mesure que les cyberattaques continuent d'évoluer, la détection des nouvelles menaces devient de plus en plus difficile. Ces menaces inconnues apparaissent dans des conditions ouvertes, où de nouveaux comportements d'attaque peuvent surgir à tout moment et ne peuvent être anticipés à l'avance. En revanche, les approches traditionnelles à ensemble fermé partent du principe que tous les types d'attaques possibles sont connus à l'avance et classifient souvent de manière erronée les comportements inédits ou ont du mal à gérer l'interaction entre les modèles opérationnels et les seuils de détection, ce qui entraîne des performances instables et des taux de faux positifs élevés. Par conséquent, les menaces inconnues doivent être traitées comme des anomalies, car tout écart par rapport au comportement opérationnel établi peut indiquer une activité malveillante.

L'analyse actuel de la détection des anomalies révèle qu'aucune solution ne peut réellement anticiper un ensemble ouvert de comportements d'attaque futurs. Par conséquent, une détection efficace des anomalies doit non seulement se concentrer sur l'apprentissage des comportements normaux dans des conditions ouvertes, mais aussi préserver l'interprétabilité, afin de permettre aux opérateurs humains de comprendre et de confirmer les anomalies signalées.

Pour remédier à ces limites, cette thèse présente un cadre de fusion de décisions multimodèle pour les anomalies, dans lequel chaque modèle spécifique à une modalité capture une perspective comportementale différente et produit un indicateur d'anomalie interprétable. En fait, cette conception sépare explicitement les indices comportementaux en modalités distinctes, qu'elles soient temporelles, structurelles, sémantiques ou contextuelles, afin que chacune puisse être modélisée clairement plutôt que d'être cachée dans une architecture monolithique. Nos modèles permettent de comprendre de manière fiable le comportement normal du système sans aucune connaissance préalable des menaces spécifiques. Chaque modèle exploite le mécanisme d'attention des Transformers pour extraire des modèles comportementaux solides. Les scores d'anomalie sont calculés par rapport à une limite de décision SVDD (Support Vector Data Description) entraînée sur un comportement normal afin d'éviter la subjectivité des seuils statiques et d'aligner les décisions directement sur l'espace de représentation appris. La fusion des résultats entre les modalités renforce la robustesse face aux menaces invisibles et en constante évolution, tout en maintenant la transparence opérationnelle.

Le bus CAN (Controller Area Network), un système de communication largement utilisé dans les véhicules, a constitué un banc d'essai idéal pour notre approche. Sa combinaison

de complexité structurelle et de comportement dynamique crée un environnement difficile pour l'évaluation de notre cadre. Le trafic du bus CAN se décompose naturellement en deux modalités dominantes: le comportement temporel et l'évolution des données utiles. Un modèle apprend les régularités temporelles, tandis que l'autre capture la sémantique des données utiles et la dynamique des séquences de messages, reflétant ensemble l'enveloppe opérationnelle réelle des systèmes embarqués dans les véhicules. Entraîné sur un ensemble de données CAN Bus J1939 réel, le cadre détecte de manière fiable les anomalies invisibles avec un taux de faux positifs proche de zéro, atteignant ainsi un objectif essentiel dans la détection des anomalies automobiles.

En combinant la séparation des modalités, la notation SVDD alignée sur la représentation et la fusion des décisions interprétables, ce travail offre une stratégie pratique et fiable pour détecter les anomalies inconnues dans les réseaux automobiles modernes.

ABSTRACT

As cyber-attacks continue to evolve, detecting novel threats becomes increasingly challenging. These unknown threats arise under open-set conditions, where new attack behaviors can emerge at any time and cannot be anticipated in advance. In contrast, traditional closed-set approaches assume that all possible attack types are known beforehand and often misclassify unseen behavior or struggle to manage the interaction between operational patterns and detection thresholds, leading to unstable performance and elevated false-positive rates. Consequently, unknown threats must be treated as anomalies, since any deviation from established operational behavior may indicate malicious activity.

Analyzing the current anomaly-detection landscape reveals that no solution can realistically anticipate an open set of future attack behaviors. As a result, effective anomaly detection must not only focus on learning normal behavior under open-set conditions but also preserve interpretability, enabling human operators to trace and confirm flagged anomalies.

To address these limitations, this thesis introduces a multi-model decision fusion framework for anomaly detection, where each modality-specific model captures a different behavioral perspective and produces an interpretable anomaly indicator. In fact, this design explicitly separates behavioral cues into distinct modalities whether temporal, structural, semantic, or contextual, so that each can be modeled clearly rather than being hidden within a monolithic architecture. Our models build a robust understanding of normal system behavior without any prior knowledge of specific threats. Each model leverages the attention mechanism of Transformers to extract strong behavioral patterns. Anomaly scores are computed relative to a Support Vector Data Description (SVDD) decision boundary trained on normal behavior to avoid the subjectivity of static thresholds and align decisions directly with the learned representation space. Fusing the outputs across modalities enhances robustness against unseen and evolving threats while maintaining operational transparency.

The Controller Area Network (CAN) Bus, a communication system widely used in vehicles, served as an ideal testbed for our approach. Its combination of structural complexity and dynamic behavior creates a challenging environment for evaluating our framework. CAN Bus traffic naturally decomposes into two dominant modalities, i.e., timing behavior and payload evolution. One model learns temporal regularities, while the other captures payload semantics and message-sequence dynamics, together reflecting the true operational envelope of in-vehicle systems.

Trained on a real-world J1939 CAN Bus dataset, the framework reliably detects unseen anomalies with near-zero false positives, achieving a critical objective in automotive anomaly detection. By combining modality separation, representation-aligned SVDD scoring, and interpretable decision fusion, this work offers a practical, reliable strategy for detecting unknown anomalies in modern automotive networks.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

CAN	Controller Area Network
SVDD	Support Vector Data Description
CPS	Cyber Physical System
AI	Artificial Intelligence
ICS	Industrial Control System
ECU	Electronic Control Unit
IDS	Intrusion Detection System
ADS	Anomaly Detection System
MCU	Microcontroller Unit
SOF	Start of Frame
RTR	Remote Transmission Request
CRC	Cyclic Redundancy Check
ACK	Acknowledgment
EOF	End of Frame
EMI	Electromagnetic Interference
DOS	Denial of Service
CIA	Confidentiality, Integrity, Availability (security triad)
OBD	On-Board Diagnostics
DLC	Data Length Code
HCRL	Hacking and Countermeasure Research Lab
LSTM	Long Short-Term Memory
CNN	Convolutional Neural Network
CSV	Comma-Separated Values
GAN	Generative Adversarial Network
AID	Automotive Intrusion Detection
VBS-D	Voltage-Based Spoofing Detection
BCNN	Binary Convolutional Neural Network
ANN	Artificial Neural Network
CCID	Cross-chain-based Intrusion Detection
MSG	Message Sequence Graphs
VBIN	Rule-based valid Bit Index
HMM	Hidden Markov Model
AAE	Adversarial Autoencoder

ACDM	Anomaly Classification Detection Model
ADDM	Anomaly Data Detection Model
GNN	Graph Neural Network
GAT	Graph Attention Network
TCN	Temporal Convolutional Network
GNB	Gaussian Naive Bayes
OCSVM	One-Class Support Vector Machine
CWT	Continuous Wavelet Transform
SRCAE	Sparse Recurrent Convolutional Autoencoder
DNN	Deep Neural Network
SVM	Support Vector Machine
TMR	Triple Modular Redundancy
FPS	Frames Per Second
IOT	Internet of Things
SAE	Society of Automotive Engineers
HDBSCAN	Hierarchical Density-Based Spatial Clustering of Applications with Noise
PGN	Parameter Group Number (J1939/CAN protocol)
SPN	Suspect Parameter Number (J1939/CAN protocol)
OEM	Original Equipment Manufacturer
PDU	Protocol Data Unit
PCA	Principal Component Analysis
POT	Peaks Over Threshold
GPD	Generalized Pareto Distribution
LLM	Large Language Model
DBC	Database Container (CAN signal database file format)
LSB	Least Significant Bit
GPU	Graphics Processing Unit
BAM	Broadcast Announce Message (J1939/CAN protocol)
RTS	Request To Send (J1939/CAN protocol)
CTS	Clear To Send (J1939/CAN protocol)
NSERC	Natural Sciences and Engineering Research Council (Canada)

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

Chapter Overview. This chapter introduces the problem context and motivation for the research. It defines the main research objectives and contributions, and presents the structure of the remainder of the thesis.

1.1 Problem Context

Cyber-physical systems (CPS) are becoming more advanced than ever [1]. Modern systems now feature complex inter-dependencies, real-time feedback loops, cloud connectivity, and adaptive interfaces. As these systems evolve, the attack surface expands simultaneously [2]. Each new feature, from over-the-air firmware updates to sophisticated AI-enabled control logic, provides attackers with fresh vectors and rules to exploit.

Many traditional defense mechanisms still operate under a closed-set assumption — they are optimized to recognize known attack signatures and maintain internal efficiency, leaving them poorly equipped to identify covert, previously unseen behaviors [3].

The consequence is that advanced attackers can operate subtly using zero-day vulnerabilities or weakly anticipated behaviors. These stealthy techniques often go undetected precisely because the defensive architecture was never designed to look beyond a limited, static threat set. A striking example of this occurred in January 2024, when the FrostyGoop [4] malware disrupted the heating system in Lviv, Ukraine, leaving over 600 buildings without heat in freezing temperatures [5]. The attackers manipulated Modbus registers and ICS (Industrial Control Systems) controllers to create direct physical effects, demonstrating how even relatively small operational exploits in CPS can have immediate and tangible real-world impact.

In automotive systems, the stakes are similarly high, but the scale and immediacy of consequences can be even greater. Modern vehicles rely on in-vehicle networks such as the Controller Area Network (CAN bus) to coordinate critical functions, from engine management and braking to advanced driver assistance systems [6]. Each electronic control unit (ECU) communicates over this bus, forming a tightly coupled cyber-physical environment. A subtle attack on the CAN bus can directly compromise vehicle safety, allowing attackers to manipulate speed, steering, or braking [7]. Unlike industrial CPS, automotive systems demand near-instant detection and response [8], highlighting the necessity for anomaly detection mechanisms capable of identifying both known and unknown threats.

The observed behavior of attackers in CAN bus traffic and the inherent inability to pre-

dict unexpected anomalies have led current solutions to make architectural compromises, often tailored to specific tasks and constrained by client specifications. Like typical Intrusion Detection Systems (IDS), these solutions rely on predefined rules crafted by cybersecurity experts and informed by threat intelligence platforms [9]. In CAN bus networks, this approach proves insufficient, as an attacker with domain knowledge can manipulate these rules to bypass these static defenses. This is particularly concerning since most standards and protocol specifications are publicly available online [10].

Widely used commercial tools, such as **PlaxidityX CAN IDPS** [11] and **ESCRYPT CycurIDS-CAN** [12], implement rule- or specification-based monitoring, sometimes enhanced with statistical or heuristic methods. While these solutions provide real-world detection capabilities, their effectiveness remains constrained when confronting novel attacks, as their operation is bound by the scope defined in the official specifications [13].

To address this limitation, other solutions incorporate data-driven methods and machine learning to reduce dependence on static rules and improve adaptability. Yet, even these systems remain largely constrained, as they target specific behaviors within a closed set of rules. This leads either to the problem of missed emerging threats or to a flood of false positives [14].

Building on this concept, several commercial hybrid IDS frameworks have been introduced, such as **Securyzr IDS** [15], which combine rule-based detection with anomaly and machine learning components. While these frameworks broaden the scope of monitoring by integrating multiple approaches, they remain far from flawless. In practice, even hybrid IDS often generate a high rate of false positives, creating significant challenges for deployment in embedded automotive environments where resources and response times are tightly constrained [16].

Moreover, despite their use of anomaly detection and machine learning, these solutions still struggle to identify truly novel or highly sophisticated attacks that closely mimic legitimate CAN bus traffic [17]. Hence, current implementations represent an important step forward, but they do not yet provide comprehensive protection against the evolving threat landscape in vehicular networks.

Considering the persistent challenges, Automotive industry groups have been motivated to develop their own solutions [18], integrating their operational experience with the latest research findings.

Nevertheless, these proposed methods are far from perfect and often require substantial customization to function reliably in practice. In particular, despite claims in the literature that machine-learning-based IDS can detect unknown attacks, these models are typically

trained on previously observed attack patterns [19]. As a result, their detection capabilities are effectively limited to those attacks or closely related variants, expanding the coverage somewhat, but still fundamentally bounded by the training set.

Against this backdrop, the most recent proposed research have sought to bridge the traditional dichotomy between intrusion detection and anomaly detection, aiming to cover both known and unknown threats [20–22]. Many of these startegies treat deviations from normal behavior as anomalies while attempting to satisfy open-set conditions, enabling the detection of novel attacks. While these designs demonstrate strong performance, they often sacrifice usability and trust, particularly in the context of safety-critical systems like automotive CAN networks. Most leverage deep learning architectures, which are currently among the most powerful learning models [23]. However, their decision-making processes are frequently opaque, operating as black boxes [24]: even domain experts may struggle to interpret why a particular anomaly was flagged, and non-experts have virtually no means of tracing an alert back to its source.

Compounding this issue, the majority of these methods are not transparent about their decision boundaries. This implies that human judgment steers the process. Consequently, both trust and reproducibility are undermined, since the anomaly criteria are not derived solely from observed system behavior.

Another major challenge confronted by industry practitioners while implementing recent research anomaly detection approaches is the persistent high rate of false positives. Discussions with data scientists and automotive cybersecurity experts at Thales[1] revealed that several implementations of anomaly detection solutions repeatedly produced alerts that did not correspond to real threats. This issue is largely attributed to what they describe as "noise" inherent in time-series representations of CAN bus data [25]. Even sophisticated machine learning models, when applied to these sequential signals, tend to flag normal variations or transient fluctuations as anomalies [26]. Thus, despite extensive tuning and optimization, false positives remain prevalent, complicating deployment in resource-constrained safety-critical automotive environments and eroding operator trust in the system's reliability.

This context motivated our development of an anomaly detection framework that balances performance, interpretability, and reliability in complex CAN bus networks.

[1]This discussion was led by Simon Bellemare and Audrey Lacoursiere Lamonde from Thales Cortaix Labs

1.2 Research objectives

This thesis aims to address the persistent limitations of existing CAN bus intrusion and anomaly detection systems. Specifically, we address the challenges of restricted open-set coverage, opaque decision logic, high false-positive rates, and dependence on manually tuned thresholds. In response to industry demands and insights gathered from current solutions, this work establishes the following objectives:

- Systematically review the state of the art in CAN bus security, benchmark datasets, intrusion detection systems, anomaly detection methodologies, and Transformer-based approaches. The objective is to define the operational, methodological, and architectural gaps that current solutions fail to bridge.
- Develop an interpretable anomaly detection framework for CAN bus data that unifies temporal dynamics, CAN ID structure, payload semantics, and time-series behavior, while offering transparency to support expert analysis and decision-making.
- Leverage Transformer architectures to learn rich temporal-semantic representations, leveraging deep learning high performance.
- Replace human-driven decision boundaries with an adaptive detection mechanism based on learned normal-behavior patterns.
- Validate the framework on the J1939 protocol, a complex CAN variant, to demonstrate performance and generalizability across diverse vehicular environments.

1.3 Thesis contributions

This thesis makes the following key contributions toward enhancing the reliability, interpretability, and robustness of anomaly detection in CAN bus networks:

- **Modular Transformer-SVDD anomaly detection Framework:** An architecture designed to expand *open-set coverage* while sustaining operational discipline. Built strictly on normal-behavior learning without any prior knowledge of attack patterns. Its modular construction ensures maintainability, scalability, and transparency, enabling each component to be inspected, interpreted, and refined independently. Every module is intentionally engineered to surface meaningful intermediate insights that domain experts can assess and validate.

- **Dual-Stream Processing Pipeline:** Separate temporal and payload processing paths that isolate time-series dynamics from payload semantics and sharpens signal clarity. This contributes to reducing noise and cross-feature interference while clarifying sequence patterns for more reliable anomaly detection.
- **Structured Data Representation:** Structures data and generates interpretable maps for CAN IDs and signals. Designed primarily to expose and clarify deep learning black-box transformations. This representation enhances traceability and allows experts to validate and interpret the origins of detected anomalies, directly mitigating *opaque decision logic*.
- **Adaptive SVDD-Based Boundaries:** A data-driven thresholding component that minimizes *high false-positive rates* and eliminates dependency on *manually tuned thresholds*. The boundary adapts to operational distributions, stabilizing detection even under shifting traffic patterns.
- **Transparent Anomaly Scoring:** A quantitative scoring mechanism that links deviations to specific signal attributes, enabling operational triage, and severity assessment across the monitoring pipeline.

1.4 Thesis structure

The remainder of the thesis is organized into four chapters, structured to present the research progressively from context to results.

Following the introduction, the second chapter provides background concepts and a detailed literature review. It covers CAN bus fundamentals, benchmark datasets, and state-of-the-art intrusion and anomaly detection methods, including statistical, machine learning, deep learning, hybrid, and Transformer-based approaches. The chapter highlights methodological and operational limitations that justify the development of the proposed framework.

The third chapter presents the main research contribution as a scientific article titled “Beyond Black-Box Detection: Interpretable Transformer-SVDD Framework for Unseen Anomalies in Heavy Vehicle CAN Bus.” It describes the design, implementation, and evaluation of the proposed framework, detailing the methodology, experimental setup, and obtained results.

The fourth chapter concludes the thesis by summarizing the main contributions, discussing limitations, and proposing future research directions to strengthen practical deployment of the proposed anomaly detection solutions in real-world vehicular networks.

1.5 Chapter Summary

This chapter introduced the research context and motivation, highlighting the critical challenges in CAN bus security, including limited open-set coverage, opaque decision logic, high false-positive rates, and reliance on manually tuned thresholds. It defined the main research objectives and contributions, emphasizing the development of an interpretable Transformer-SVDD anomaly detection framework built on normal-behavior learning. It also outlines the thesis structure, providing a roadmap for the detailed presentation of the background, methodological development, experimental evaluation, and discussion of results in the subsequent chapters.

CHAPTER 2 BACKGROUND AND LITERATURE REVIEW

Chapter overview. This chapter consolidates the core foundations required to situate our work within the broader landscape of CAN bus security and anomaly detection. We begin by revisiting the fundamentals of the Controller Area Network, detailing its protocol structure and inherent vulnerabilities that shape modern attack surfaces *(Section 2.1)*. We then catalogue the benchmark datasets widely adopted for evaluating CAN intrusion and anomaly detection systems, outlining their composition, design intent, relevance to real-world automotive environments and highlighting their specific limitations*(Section 2.2)*. In*(Section 2.3)* we shift to a focused review and criticism of intrusion detection approaches applied to CAN traffic, including statistical, machine learning, deep learning, and hybrid IDS methods. Building upon that, we broaden the analysis to anomaly detection methods, emphasizing their capabilities in identifying unknown or evolving threats *(Section 2.4)*.

2.1 Controller Area Network

2.1.1 CAN Protocol

The CAN bus is a communication system that underpins modern vehicles and machinery. It allows ECUs (Electronic Control Units) to exchange data without the need for a central computer. Like a nervous system, it facilitates rapid and reliable interaction between critical vehicle components such as the engine, brakes, etc. ECUs, or CAN nodes, function like body parts and operate autonomously to send and receive information over a shared two-wire twisted pair, consisting of CAN high (color-coded green) and CAN low (color-coded yellow) wires as illustrated in Figure 2.1. An ECU has three key parts [27]:

1. **Microcontroller (MCU):** The brain that processes and sends CAN messages.
2. **CAN controller:** Ensures communication follows the CAN protocol (message encoding, error detection, arbitration, etc.)
3. **CAN transceiver:** Connects the controller to the CAN bus and converts data into signals, while providing electrical protection.

The communication on the CAN bus occurs through structured packets known as CAN frames. These frames contain fields for [6]:

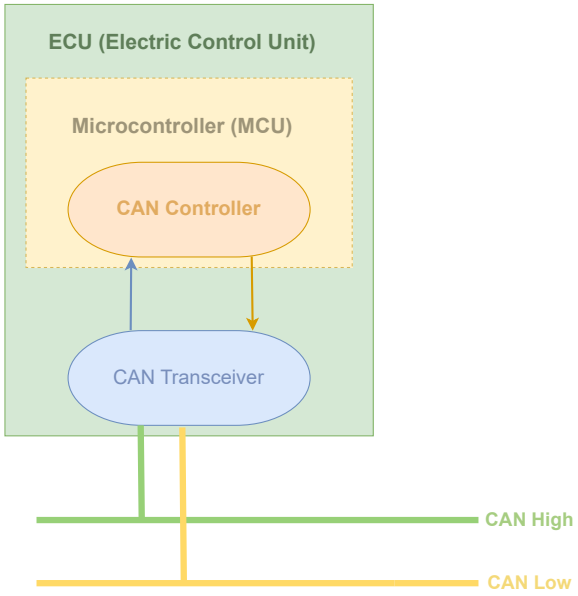


Figure 2.1 Electronic Control Unit architecture

1. **SOF:** The Start of Frame, which is usually a 'dominant 0' that notifies the other nodes if a CAN node intends to talk
2. **ID:** It's the frame identifier, lower values have higher priority.
3. **RTR:** The Remote Transmission Request indicates if a node sends data or requests dedicated data from another node.
4. **Control:** The Control contains the Identifier Extension Bit (IDE), a 'dominant 0' for 11-bit. In addition to the 4-bit Data Length Code (DLC), which specifies the length of the data bytes to be transmitted (0 to 8 bytes).
5. **Data:** The Data contains the data bytes (payload), including CAN signals that can be decoded for information.
6. **CRC:** The Cyclic Redundancy Check is used to verify data integrity.
7. **ACK:** The ACK slot indicates whether the node has acknowledged and received the data correctly.
8. **EOF:** The EOF indicates the end of the CAN frame

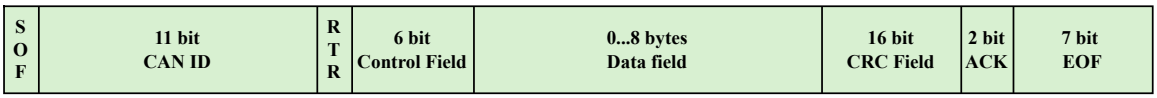


Figure 2.2 CAN frame

The CAN protocol ensures that high-priority messages take precedence, while error handling mechanisms manage transmission faults and prevent network congestion. Several other intrinsic characteristics of the CAN bus make it ideal for configuring and monitoring communication in complex systems.

2.1.2 Key characteristics of CAN bus

Locating abnormalities is made easier by the CAN bus's inherent robustness and fault tolerance. It uses differential signals [6] that broadcast messages to the entire network and are referred to as dominant and recessive. Whereas the dominant suggests that the differential voltage is typically higher than the minimal threshold, the recessive indicates that the differential voltage is typically lower. When a logic "0" is put onto the bus, a dominant condition is typically reached. Conversely, when a logic "1" is driven onto the bus, a recessive state is typically reached. This mechanism protects against electromagnetic interference (EMI), to send complementary signals (CAN high and CAN low) over the twisted pair. Even in difficult, noisy environments, the CAN bus guarantees dependable data transmission by evaluating the voltage differences between these signals. Unexpected departures from these typical patterns may serve as unmistakable warning signs of possible abnormalities [27].

Through the proper use of its bandwidth to manage message gaps, the CAN bus adequately regulates communication and aids in uncovering abnormal traffic patterns. It also prioritizes critical messages through an arbitration mechanism [6], minimizing collisions and ensuring timely delivery. Unexpected delays in high-priority messages or interruptions in the arbitration process can indicate possible problems, which makes this feature combination especially useful for anomaly identification.

The centralized architecture [27] of the CAN bus is an added asset since it allows for easy logging and diagnostics from a single location. This feature makes it easier to spot odd message patterns, delays, or traffic congestion by providing a comprehensive picture of the network's activity.

2.1.3 Core Weaknesses and Attacks on the CAN Bus

Although the CAN bus architecture was designed for industrial and automotive systems to be reliable, simple, and efficient, attackers are able to find creative ways to take advantage of its inherent rigidity.

One of the core vulnerabilities lies in the deterministic nature of the CAN protocol. While this predictability is critical for ensuring that safety-critical messages, like brake or engine control signals, are prioritized and delivered on time, it also creates an attack surface that is easy for malicious actors to study and exploit. Without encryption or authentication [28], the protocol assumes that any node transmitting data is trustworthy. This absence of verification has been exploited by unauthorized nodes to readily inject, alter, or intercept messages.

This is complemented even more by the flat topology of the CAN network. All nodes share the same communication medium, which means that once an attacker gains access, either through a physical connection or through an external access point such as a diagnostic port, telematics module, or compromised infotainment system, they can monitor and manipulate all network traffic [29]. Message spoofing and denial-of-service (DoS) attacks [30] are among many other attack techniques made possible by this shared communication channel. However, it should be noted that this issue was addressed by newer vehicular platforms, as some have mitigated these risks by segregating the CAN bus into smaller subsystems [31]. Network segmentation improves security by limiting access to the CAN bus, but does not fully protect against channel threats, and complicates maintenance while increasing costs [32].

The threats associated with the CAN bus increase dramatically when vehicles are connected to more external systems. Modern cars now include internet-connected telematics systems, smartphone integration, and even over-the-air software updates, creating additional attack vectors [33]. An attacker can compromise essential vehicle functions by taking advantage of the CAN bus's predictable and vulnerable structure after breaching any of these systems, which can present significant security threats.

What makes these attacks particularly concerning is their stealthy nature. The absence of integrated intrusion detection systems on the CAN bus makes it challenging to identify and react attacks instantly. Attackers might act covertly, frequently going undetected until the repercussions are evident.

When analyzed through the lens of the CIA Triad (Confidentiality, Integrity, and Availability), these attacks reveal their full operational footprint on system security. The following attacks were collected from prior research and systematically classified. For clarity and strategic coherence, we present them in increasing order of stealthiness, moving from overt,

disruptive behaviors to subtle, covert manipulations that challenge conventional detection pipelines:

Availability

Denial of Service (DoS) Attacks: By overwhelming or deactivating the network, Denial of Service (DoS) attacks pose a significant threat to the reliability of CAN bus communication and often result in critical system failures. The most impactful DoS variants include **Grounding attacks** [34], which halt all ECU transmissions by adding noise or creating improper ground connections, holding the bus at ground potential (dominant bit), and **Bus-Off exploits** [35], which force an ECU into an error state by repeatedly sending bits until it is effectively disabled. Likewise, **Priority exploitation** [35] of the bus arbitration mechanism by flooding the network with high-priority messages, preventing lower-priority traffic, while flooding Attacks overload the bus by transmitting specific message IDs at excessively high frequencies. **Basic DoS** [36] attacks disrupt communication on a broader scale by injecting large volumes of meaningless or false data, such as fabricated alerts. In contrast, more targeted variants like **Message Suspension attacks** [37] achieve localized disruption by stealthily blocking critical messages from reaching secondary ECUs, often through a compromised proxy. This variant of attacks can be extended into Suppression Attacks, where an attacker goes one step further by stopping an ECU from transmitting any messages. **Suppression attacks** [34] can exclude the ECU from the rest of the network, which may aggravate the effect of the suspension attack and potentially cause broader system failures.

Fuzzing Attacks: Fuzzing attacks use random message injection to investigate and take advantage of the CAN bus system's hidden or unexpected features. Some of these attacks pose serious safety dangers, and others vary in subtlety and intensity. For example in the context of automotive vehicles, the **Self-Destruct Fuzzing** [7] can start extreme behaviors like locking car doors or turning off the engine, putting passengers in danger and triggering system malfunctions. Similar to this, **Light-Out Fuzzing** [7] disables the vehicle's lights under specific conditions, such as high-speed travel, creating dangerous driving scenarios. Conversely, less covert but moderately severe methods include **Speedometer Fuzzing** [7], which tampers with speed data to deceive drivers or onboard systems, possibly interfering with vehicle functions. Beyond that, we mention **Basic fuzzing** [38] introducing arbitrary messages into the system to check for vulnerabilities, frequently revealing flaws that could be further exploited.

Integrity

Message Injection Attacks: Message injection attacks exploit the CAN bus's lack of authentication by introducing fake or misleading messages to disrupt operations or cause undesirable behavior. Among the most disruptive and noticeable attacks are **Massive Injection** [39], which floods the bus with pre-ordered or random CAN IDs to sever communication, and **Flam Injection** [40], where fake messages are injected immediately after legitimate ones to evade detection. **Malfunction Injection** [41] also has a high impact, since it alters or forges messages to trigger system failures or abnormal behavior in vehicle systems.

Spoofing and Impersonation Attacks: Those attacks exploit the CAN bus by mimicking legitimate nodes or ECUs to deliver malicious messages, often with a high degree of subtlety. Among the most severe and covert methods are **Fabrication Spoofing** [42], which manipulates sensor or actuator data by flooding the bus with conflicting or false messages at high frequencies, and **Masquerade Spoofing** [43], where a disabled ECU is replaced with a malicious node that seamlessly mimics normal communication. In a similar vein, **Proxy attacks** [38] use hacked ECUs to discreetly transmit malicious messages. A more moderate but equally covert method is the **Error Passive Spoofing Attack** [44], which forces a target ECU into a passive state and then overwrites specific frames with spoofed payloads, compromising the integrity of the system while remaining difficult to detect.

Expanding on these attacks, more subtle variants with moderate impact and high stealth include **Signal Plateau Manipulation** [34], where signal values are overwritten with constant data, and **Gradual Signal Drift** [34], in which values are gradually altered to create deviations from their true readings.

Protocol Exploitation Attacks: Protocol exploitation exploits use the CAN protocol's built-in design flaws to disturb communication or alter system behavior. Due to their subtle nature, these attacks can vary in severity and are frequently challenging to identify. One of the most impactful attacks is the **Janus Attack** [45], which crafts a single CAN frame containing two different payloads, exploiting the dual interpretation of the frame by receivers.

This may have unexpected repercussions, since the system may process falsified information without detecting the manipulation. Another high-impact attack is the **Double Receive Attack** [44], which exploits a bit error in the end-of-frame (EOF) field, to push receivers to process the frame twice, even though it has already been accepted. Although the likelihood of this failure is low, the vulnerability exists due to the protocol's design and can lead to unintended incidents.

On the more obscure end, **Timing Opaque (T.O.) Attacks** [40] are designed to operate without disrupting the normal timing or ID patterns of CAN messages, making them particularly difficult to detect using basic intrusion detection systems (IDS). To identify these attacks, more advanced methods such as side-channel analysis or deeper payload examination are required. On the opposite, **Timing Transparent (T.T.) Attacks** [40] introduce unusual timing or new IDs, making them transparent to the frequency-based IDS systems. Despite being simpler to spot than T.O. attacks, they may still have serious effects, especially if the introduced irregularities are not addressed immediately.

These strategies of protocol exploitation draw attention to the CAN protocol's flaws which often remain undetected until the system fails or exhibits unexpected behavior.

Replay attacks: Even though they are typically less disruptive than certain other attack types, their extreme concealment and capacity to evade detection make them a serious security issue. These **Replay attacks** [46] repurpose previously recorded legitimate messages to achieve malicious objectives. For instance, they can replay message sequences to unlock doors or start the engine, enabling attackers to mimic normal vehicle functions while remaining undetected.

Confidentiality

Sniffing and Eavesdropping Attacks: These attacks are passive threats that concentrate on spying and analyzing CAN bus traffic. They are considerately stealthy but usually have minimal impact. Passively intercepting traffic to obtain private information covertly is known as **Sniffing or Eavesdropping attacks** [47]. Subsequently, **Frame Injection, Sniffing, and Falsifying attack** [7] go a step further by actively intercepting, crafting, and injecting fake frames into the CAN bus. A malicious node can observe all the frames transmitted on the bus and design fake frames that carry misleading or false data. These fabricated frames can then be broadcasted across the network, where they are accessible to all nodes, thanks to the CAN bus's broadcast characteristic. By setting the correct frame ID, an attacker can deceive specific ECUs into accepting these malicious frames as legitimate, potentially leading to disruptions or manipulation of vehicle operations without detection.

Table 2.1 Categorization of CAN Bus Attacks by Category and stealth level

Category	**Attack**	**Variant**	**Description**	**Stealthiness**
Availability	DoS	Grounding [34]	Halt ECU traffic (bus dominant)	Low
	DoS	Bus-Off [35]	Force ECU shutdown via errors	Low–Med
	DoS	Priority Exploit [35]	Block lower-priority frames	Low
	DoS	Flooding	Overload bus traffic	Low
	DoS	Basic DoS [36]	Inject false/noisy data	Low
	DoS	Suspension [37]	Block critical ECU messages	Med
	DoS	Suppression [34]	Stop ECU transmissions	Med–High
	Fuzzing	Self-Destruct [7]	Unsafe behaviors (engine off, doors lock)	Low
	Fuzzing	Light-Out [7]	Disable vehicle lighting	Low
	Fuzzing	Speedometer [7]	Distort displayed speed	Low–Med
	Fuzzing	Basic Fuzzing [38]	Random injection to expose flaws	Low
Integrity	Injection	Massive [39]	Flood bus with forged IDs	Low
	Injection	Flam [40]	Inject forged frames after legit ones	Med
	Injection	Malfunction [41]	Alter/fabricate messages	Med
	Spoofing	Fabrication [42]	Overwrite sensor/actuator signals	Med–High
	Spoofing	Masquerade [43]	Replace ECU with malicious node	High
	Spoofing	Proxy [38]	Compromised ECU relays traffic	High
	Spoofing	Error Passive [44]	Force ECU passive, overwrite frames	High
	Signal Manip.	Plateau [34]	Replace dynamic signals with constants	Med
	Signal Manip.	Drift [34]	Gradually shift values	Very High
	Protocol Exploit	Janus [45]	Dual-interpretation frame	Med–High
	Protocol Exploit	Double Receive [44]	EOF error $\rightarrow$ double process	High
	Protocol Exploit	Timing Opaque [40]	Hidden timing/ID manipulation	High
	Protocol Exploit	Timing Transparent [40]	Visible timing/ID manipulation	Med
	Replay	Replay [46]	Reuse legit messages	Very High
Confidentiality	Sniffing	Sniffing [47]	Passive traffic interception	High
	Sniffing	Sniff+Inject+falsify [7]	Intercept/inject fake frames	High

2.2 Benchmark CAN Bus Datasets

The assessment of intrusion and anomaly detection approaches, in the context of in-vehicle communication networks, particularly CAN networks, has been supported recently by a variety of available datasets. Each dataset provides different scenarios, attack types, and feature sets that enable comprehensive testing of detection algorithms. In this section, we will discuss such datasets in more details and highlight their key specifications.

2.2.1 OTIDS Dataset (Offset Ratio and Time Interval based Intrusion Detection System):

This dataset was initially collected during the development of a novel Intrusion Detection System by Lee et al [38]. In their paper, they introduced OTIDS (Offset Ratio and Time Interval-based Intrusion Detection System) as a pioneering approach to securing in-vehicle networks. The dataset was created by logging CAN traffic from a Kia Soul via the OBD-II port under both normal driving conditions and simulated message injection attacks. Although the specific duration of the data collection was not disclosed, the experiments included three distinct types of attacks: DoS, Fuzzy, and Impersonation. In the DoS attack, messages with the CAN ID '0x000' were injected at short intervals, while the Fuzzy attack involved injecting spoofed random CAN IDs and data values. The Impersonation attack mimicked a legitimate node using CAN ID '0x164.' The dataset captured critical attributes such as timestamps, CAN IDs, Data Length Codes (DLC), and the actual data values transmitted in bytes.

Lee et al. utilized this dataset to showcase how offset ratios and time intervals can effectively identify intrusions, with key features like Instant Reply and Lost Reply detecting anomalous patterns with high accuracy [38].

Strengths and Weaknesses: The dataset presents a unique fuzzing attack that operates stealthily by spoofing only IDs that appear in normal traffic, making it the sole example of such an attack available in an open dataset. It is also the only open dataset that includes remote frames along with their corresponding responses, providing a distinctive feature for research purposes.

However, several notable drawbacks must be considered [40]. The injected messages are not labeled, and the documentation regarding injection intervals is unclear, with potential inaccuracies. Although the authors assert that Denial-of-Service (DoS) attacks occur throughout the capture and that fuzzing and impersonation attacks begin after approximately 250 seconds, it seems that all attacks span the entire capture, which contradicts the fuzzing attack injection at 0.1565 seconds mentioned in their paper. It is also unclear how to characterize

the dataset's "impersonation attack," which is referred to as a masquerade attack in the accompanying paper because it does not halt transmissions from the real node. Likewise, the inclusion of remote frame requests and responses introduces subtle timing variations that are not typically observed in ambient traffic, potentially causing delays or inconsistencies that deviate from typical traffic patterns, leading to false positives and bias.

2.2.2 The SynCAN dataset:

Hanselmann et al. gathered this data while building an unsupervised intrusion detection system [34]. It is the first dataset that targets signal-based models and consists of Synthetic signals (no "raw" binaries, translated time series) [40]. They developed CANet, an innovative system designed to detect anomalies in high-dimensional CAN bus data using unsupervised learning. The SynCAN dataset, central to their research, consists of both real and synthetic CAN traffic. It includes 12.5 hours of normal driving data from a test vehicle, complemented by 16.5 hours of synthetic traffic designed to simulate attack scenarios. Over 8 hours, various attacks were performed, with 0.5 hours targeting real vehicle data and 7.5 hours involving synthetic data. The attack scenarios included in the dataset comprise Plateau Attacks, Continuous Change Attacks (gradually altering signal values), Playback Attacks, Flooding Attacks (General DoS), and Suppress Attacks. Key attributes captured in the dataset were timestamps, CAN IDs, and decoded signal values from the payload bytes.

CANet utilized this dataset to compute reconstruction errors, presenting a crucial metric for spotting irregularity in CAN communication [34]. The experiment's results utilizing this data showed that variances in reconstruction errors successfully drew attention to anomalous patterns, confirming CANet's ability to identify intrusions without knowing the sorts of attacks beforehand.

Strengths and Weaknesses: This dataset is one of the few signal-based datasets available, except for ROAD [40], and represents a significant contribution, as the encoding of signals in most vehicles' CAN data remains largely unknown. It includes some of the most advanced masquerade attacks currently available, such as signal values gradually drifting from a real value to a target value or being replayed from normal traffic. Unlike other datasets, this one focuses on attacks targeting a single signal, rather than the entire 64-bit data field, making it particularly valuable for testing sophisticated, signal-based intrusion detection systems (IDS).

Nevertheless, synthetic is an imperfect representation of real-world data. Also, simulated attacks present challenges, as their impact on a vehicle cannot be verified. Lastly, the dataset does not include a raw untranslated version, limiting its usability for many CAN detectors

that rely on the standard format of time-stamped IDs with 64-bit data fields.

2.2.3 The Real ORNL Automotive Dynamometer dataset (ROAD):

This dataset was introduced by Verma et al. as the first dataset that combines real CAN traffic with physically verified advanced attacks to provide a comprehensive benchmark for automotive intrusion detection systems (IDS) [40]. Collected from a passenger vehicle using SocketCAN on a Linux system with a Kvaser Leaf Light V2 device connected to the OBD-II port, the ROAD dataset includes over 3.5 hours of real CAN data recorded during on-road driving and dynamometer tests. It features 33 labeled attack captures, encompassing both real and simulated attacks. These attacks are further categorized into Timing Transparent (T.T.) attacks, detectable by frequency-based IDS methods due to their disruption of message timing or ID patterns, and Timing Opaque (T.O.)attacks, which preserve normal timing and ID distributions to evade simpler detection methods [40]. Actual attacks such as fuzzing, falsification, and flam attacks are included in the collection. These involve discreetly replacing real messages with hostile ones. By precisely mimicking real traffic patterns, spoofing messages, and suppressing genuine ones at realistic intervals, simulated masquerade attacks make detection even more difficult. Important characteristics including timestamps, CAN IDs, and signal-translated time series are recorded, making the dataset suitable for both raw data-based and signal-based IDS evaluation.

According to Verma et al., all real attacks were physically verified, distinguishing ROAD as one of the highest-fidelity CAN IDS datasets available, and addressing critical gaps in the availability of realistic, high-quality attack data for research.

Strengths and Weaknesses: Among the ROAD dataset strengths, it provides a diverse set of real CAN data, encompassing ambient driving scenarios and a wide range of attacks that vary from transparent to highly subtle. The use of a dynamometer enables the safe execution of attacks during driving and includes diverse activities such as reversing, accelerating, and door operations. This variety makes the dataset particularly useful for training detectors, identifying non-hacking anomalies, and testing false positives. Moreover, unique captures, such as Accelerator Attacks, and detailed metadata on driving activities and attack effects further enhance its value.

Despite its strength, It is still limited to data from a single vehicle, which restricts its generalization across different models or configurations [40]. Additionally, dynamometer-collected data differs subtly from real road driving conditions, potentially impacting its realism. The dataset's 100μs timestamp resolution may be inadequate for certain time-based detectors, and the obfuscation of ID order can pose challenges for IDSs that rely on ID priority. Also,

it lacks physical-layer intrusion detection data and relies on limited simulation for masquerade attacks, underscoring a broader gap in publicly available datasets for such advanced scenarios.

2.2.4 The HCRL CAN-FD Intrusion Dataset:

The HCRL (Hacking and Countermeasures Research Lab) developed the CAN-FD Intrusion Dataset [48], which provides real-time CAN-FD traffic for intrusion detection system evaluation. Data was collected mostly in urban areas during a round-trip drive of one hour near Korea University. To ensure a variety of temporal characteristics, attack scenarios were presented over 500 seconds, with random intervals ranging from 3 to 5 seconds. The dataset includes three types of attacks: Flooding Attacks, where messages with CAN ID '0x000' were injected every 0.1 ms; Fuzzing Attacks, involving random Arbitration IDs, DLCs, and payloads injected every 0.2 ms; and Malfunction Attacks, with messages of CAN ID '0x125' injected every 1 ms. Key attributes captured include timestamps, Arbitration IDs, DLCs (ranging from 0 to 64 bytes), and data values, with labels distinguishing injected (T) from normal (R) messages. The dataset's combination of real CAN-FD traffic and varied attack scenarios supports the evaluation and development of intrusion detection methods for modern vehicular networks.

Gao et al. evaluated their proposed model using two main datasets, including the CAN-FD Intrusion Dataset [49]. It was essential in evaluating the model's capacity to identify and classify anomalies in the CAN-FD bus data. The authors used this dataset to show the effectiveness of their approach works in identifying multi-attack scenarios and obtaining higher intrusion detection accuracy than current LSTM-based and CNN-LSTM-based techniques.
Strengths and Weaknesses: The HCRL CAN-FD Intrusion Dataset makes an important contribution to CAN IDS research by offering real CAN data sourced from actual vehicles, which plays a crucial role in designing practical intrusion detection systems. It encompasses several types of attacks, including DoS and fuzzing, with labeled injected messages that facilitate effective training and validation of IDS models. The dataset's detailed timestamps enable the development of timing-based anomaly detection techniques.

However, it does have some limitations. The data comes from only one vehicle, which restricts its applicability to a broader range of models. Moreover, there are issues with labeling consistency, discrepancies in the documentation regarding injection intervals, and an overall lack of detailed documentation, which can pose challenges for reproducibility. Finally, the dataset does not cover more sophisticated attack scenarios and the existing attacks are relatively easy to detect using simple methods.

2.2.5 The Survival dataset:

The Survival Dataset was introduced in Han et al.'s timing-based CAN IDS paper [50], as part of their work on testing a CAN anomaly detection algorithm using a survival analysis model, aiming to detect malicious CAN messages and classify the normality or abnormality of vehicle networks without requiring semantic knowledge of CAN ID functions.

This dataset stands out as the only publicly available resource that contains real attack data from multiple vehicles, offering a valuable benchmark for testing IDS systems on a variety of vehicles It is composed of real CAN data collected from three vehicles: The Hyundai YF Sonata (2010), the KIA Soul (2015), and the Chevrolet Spark (2015). It was recorded using a Raspberry Pi3 with a PiCAN2 module, capturing normal vehicle operation during regular driving. Three different kinds of injection attacks were used in the attack scenarios to create vehicle malfunctions including DoS (Denial of Service), fuzzing, and targeted ID attacks. The DoS attack, referred too as "flooding," involved injecting messages with high frequency, while fuzzing involved injecting random CAN IDs and data. The targeted ID attack, which they referred to as a "malfunction" attack, manipulated specific CAN ID values, though the functionality of the targeted IDs was not disclosed in detail. In each attack capture, up to four five-second injection intervals were included, which had a duration of 25 to 100 seconds and each injected message was labeled for convenient identification.

In the paper by Han et al., the variability in the Survival Dataset was leveraged to demonstrate how different vehicles respond uniquely to real attacks [50], particularly fuzzing attacks, which caused significant differences in bus load and message timing across the tested vehicles. **Strengths and Weaknesses:** This dataset is unique in that it contains real attacks on multiple vehicles, with the same set of attacks repeated several times on each. Furthermore, the authors confirm that these attacks have tangible effects on the vehicle, making this dataset a good option for training and testing a vehicle-agnostic, simple timing-based detector.

A significant limitation of this dataset is that the attacks are overt and easily detectable, even by a basic timing-based detector [40]. The attack causes a dramatic disruption to the bus frequency due to the unusually high injection frequency, unlike the more subtle fuzzing attack used in the other datasets. The payloads injected are either random values (on the Soul vehicle) or a single random value (on the Spark and Sonata), which limits the diversity of the attack. Additionally, the ambient data provided (60-90 seconds per vehicle) is likely insufficient for robust training or testing false positive rates. This ambient data and attack data are stored in differently formatted CSVs, which complicates the analysis.

2.2.6 The HCRL Car Hacking Dataset:

The Car Hacking dataset, the latest release from HCRL, was originally collected to evaluate GIDS [51], a Generative Adversarial Network (GAN) developed to detect anomalies in the Automotive Intrusion Detection (AID) sequence. It was later utilized to evaluate other deep learning-based Intrusion Detection Systems. Data was logged through the vehicle's OBD-II port across various driving scenarios such as city driving, highway travel, parking, and idling. It was constructed using real CAN traffic data from a Hyundai YF Sonata, the dataset captures over 500 seconds of ambient driving data and includes four distinct attack types. Each instance includes 300 attack intervals, with each intrusion lasting 3 to 5 seconds, and up to 40 minutes of CAN traffic. Notably, the attack captures are large, with many examples per attack, which makes them more useful for research.

The dataset includes three types of flooding fabrication attacks: DoS attacks, where the CAN ID '0000' is injected every 0.3 milliseconds; Fuzzing attacks, which inject random CAN IDs and data every 0.5 milliseconds; and two targeted ID spoofing attacks that flood the network with false drive gear and RPM gauge data every millisecond. In contrast to other HCRL datasets, this one gives researchers important context by explicitly describing the functionality of the targeted IDs, like the drive gear and RPM gauge. Every message is meticulously labeled to differentiate between malicious, injected traffic (designated as "T") and legitimate traffic (designated as "R"). Thus, it records crucial characteristics such as timestamps, CAN IDs, Data Length Code (DLC), and data values (DATA[0 7]), providing necessary data to analyze traffic patterns.

In the CAN IDS literature, this dataset appears to be the most often used [51] one because of its comprehensive coverage of real and fabricated attack scenarios, combined with its large number of instances per attack.

Strengths and Weaknesses: The attack captures in this dataset are extensive, containing a large number of instances per attack, thereby offering a comprehensive dataset. In contrast to other HCRL datasets, it provides the function of the IDs in the targeted ID attacks (drive gear, RPM gauge...) which adds valuable contextual information. This dataset is frequently used in the CAN IDS literature due to its broad coverage of both real and fabricated attack scenarios, as well as the substantial number of instances per attack.

However, a significant drawback is that all attack captures exhibit a data collection artifact that may pose challenges for researchers, particularly since it is not noted in the documentation [40]. Specifically, after each attack (around the 300th injection), there is a noticeable gap in message transmission, during which no data is recorded. Berger et al. and Verma et al. both also identified these timestamp jumps, which can impact the training and testing

process. Moreover, similar to the HCRL survival dataset, the attacks are overt, causing significant disruption to the overall bus timing. Moreover, the ambient and attack data are stored in different formats (fixed-width and CSV), which complicates the analysis.

2.2.7 Summarized gaps:

While datasets such as OTIDS, SynCAN, ROAD, HCRL CAN-FD, Survival, and Car Hacking have significantly contributed to research on CAN bus intrusion detection, they exhibit common limitations that constrain their utility for real-world evaluation. Many rely on synthetic or simplified traffic (e.g., SynCAN) or are limited to single-vehicle captures (e.g., ROAD, HCRL CAN-FD), reducing their representativeness of diverse operational conditions. Attack scenarios are often overt, deterministic, or artificially injected, lacking the subtlety and variability seen in real vehicle networks. Several datasets suffer from inconsistencies in labeling, documentation, or data formatting (e.g., Survival, Car Hacking), complicating reproducibility and model validation. Furthermore, some datasets introduce artifacts such as timestamp jumps, repeated payload cycles, or non-standard remote frame behavior (e.g., OTIDS, Car Hacking), which can bias models and inflate detection metrics. Overall, these shortcomings limit the generalization and realism of models trained or evaluated on these resources.

2.3 Inrusion Detection focus

Due to the growing number of electronic control units (ECUs) and the interconnectedness of contemporary automobiles via Controller Area Network (CAN) bus systems, strong cybersecurity measures for vehicles are now required. Over the past three years, numerous intrusion detection systems (IDSs) have been proposed. With vulnerabilities in CAN networks enabling variety of sophisticated attacks, researchers have turned their focus to developing Intrusion Detection Systems (IDSs) that specifically detect different kind of attacks. This section reviews recent advancements in IDSs proposed over the last two years, categorizing them by the techniques employed and emphasizing their intrusion detection capabilities.

2.3.1 Statistical IDS

Junaid et al. introduced a statistical IDS that uses brute-force optimization over various window sizes to optimize detection thresholds [52]. By analyzing the statistical properties of attack patterns, the system was able to identify malicious behavior in the Car Hacking Dataset and the Survival Analysis Dataset. Junaid team assume that the method's simplicity

and targeted focus on detecting malicious patterns make it a lightweight yet effective solution.

2.3.2 Machine Learning and Deep Learning-Based IDS

Ming et al. has introduced an adaptive lightweight IDS designed for the resource-constrained environment of onboard ECUs [53]. The system leverages message cycle characteristics to monitor and detect malicious intrusions efficiently. Simulated data from the CANoe platform was utilized to validate the model's capability in identifying intrusion attempts while minimizing computational overhead. The emphasis on intrusion detection rather than generalized anomaly detection highlights the system's targeted approach to identifying unauthorized access.

Building on the need for adaptable security controls, Khatri et al. applied transfer learning to enhance the detection of known intrusion categories across new environments [54]. Their approach fine-tunes models on labeled intrusion datasets, enabling the extraction of high-fidelity features tied to established attack behaviors. This method, validated on real-world data, highlighted the importance of systems that can generalize across diverse scenarios.

Adopting a supervised tree-based strategy, Anjum et al. [55] have applied Extreme Gradient Boosting (XGBoost) to normalized CAN features, such as message identities and payloads. With minimal false alarm rates, the model was able to classify a variety of intrusion types, such as spoofing, fuzzy, and malfunction attacks from Car Hacking and Survival datasets.

An improved Generative Adversarial Network (GAN) model was used by Wang et al. to develop an IDS tailored for CAN-FD networks [56]. By converting ID segments into binary image encodings (ID images) and employing a dual discriminator architecture, the IDS achieved high precision in distinguishing between legitimate and attack messages. Wang team evaluated their solution on real-time data from Honda and KIA vehicles to confirm its suitability for intrusion detection in high-speed CAN-FD environments.

Levy et al. enhanced a Voltage-Based Spoofing Detection (VBS-D) mechanism with deep learning to detect spoofing attacks [57]. By leveraging physical side-channel data, such as voltage signals, the system identifies silent malicious devices on the CAN bus. The IDS's adaptability to environmental changes and its ability to detect intrusions immediately upon vehicle start were validated using data collected from a moving vehicle.

A Binarized Convolutional Neural Network (BCNN)-based IDS introduced by Zhang et al. [58] and designed to capture the temporal and spatial characteristics of CAN messages. By utilizing a novel input generator, the system enhances its learning process, ensuring high detection accuracy while significantly reducing memory usage and processing latency.

The BCNN identified malicious intrusions in datasets collected via the OBD-II interface, confirming its practical application.

Khishore et al. proposed an IDS based on Bidirectional Long Short-Term Memory (LSTM) networks [59]. By processing CAN bus message sequences in both forward and backward directions, this deep learning model captures contextual relationships, enabling the detection of complex intrusion patterns, including spoofing and DoS attacks. The system's performance, validated on the Car Hacking Dataset from the Attack and Defense Challenge-2020, exhibited high accuracy in differentiating between normal and attack messages.

A combination between Artificial Neural Networks (ANNs), Long Short-Term Memory (LSTM) and Federated Learning was proposed by Muzun et al. [60]. Muzun team aimed to eliminate the need for rule set management and database maintenance by considering the periodic nature of the CAN protocol. Additionally, Federated Learning was used to enable the IDS to adapt to new types of intrusions without requiring updates to the detection model. The approach was validated on SONATA, SOUL, and SPARK vehicle data, showcasing its capability to handle different intrusion scenarios.

2.3.3 Hybrid IDS

The Cross-chain-based Intrusion Detection Model (CCID-CAN) by Sun et al. [61] employs a rule-based valid bit index (VBIN) and a combination of Kalman filters with Naïve Bayes models to detect intrusions. The system incorporates cross-chain attack log exchanges among vehicles to improve detection coverage and reduce the likelihood of missed intrusions. Real vehicle data from XPeng G9 was used to evaluate the IDS's performance against intrusion attempts.

Along the same line, Lm et al. proposed an IDS combining machine learning models with rule-based filters [62]. This dual approach enabled the system to initially detect potential intrusions using ML techniques before verifying the outputs through cross-checking message IDs and payloads with predefined rules. The system's capacity to address attacks such as DoS, spoofing, and fuzzing was validated on the Car Hacking Dataset.

Marfo et al. employed a hybrid approach combining graph machine learning with time-series analysis [63]. They worked on improving detection of masquerade attacks by representing CAN bus frames as message sequence graphs (MSGs) enriched with statistical attributes. The ROAD dataset was used to validate the model's intrusion detection capabilities.

2.3.4 Identified Gaps

Despite the breadth of machine learning, deep learning, and hybrid IDS solutions proposed for CAN bus security and their demonstrated ability to detect a range of sophisticated, well-known attacks, several structural gaps remain unresolved across the current landscape.

The dominance of supervised learning has created a heavy reliance on large, labeled attack datasets. As direct consequence, the vast majority of IDS models operate under a *closed-set training paradigm*, optimizing exclusively for predefined attack categories rather than for deviations in intrinsic vehicle behavior. This drives models to memorize attack patterns instead of understanding system dynamics. As a result, they consistently fail when exposed to previously unseen manipulations, timing irregularities, or payload deviations that fall outside their labeled training regimes.

These limitations highlight the field's pivot toward *behavior-driven anomaly detection*, where the objective shifts from attack-pattern recognition to modeling normal timing, payload evolution, Signals dynamics, and inter–CAN-ID relationships. Such approaches aim to identify deviations rooted in system behavior rather than relying on fixed, predefined signatures.

2.4 Anomaly Detection focus

Most of recent efforts have been made to use intrusion detection systems to detect known types of attacks (closed-set). However, several researchers have recognized that this traditional approach may be insufficient for detecting new or evolving threats (open-set). During the same period mentioned in the previous section, a parallel research direction emerged that focused on identifying any deviations from established normal vehicle behavior, regardless of whether they corresponded to known attack patterns. This shift makes it clear that in a highly dynamic CAN bus environment, previously unknown attack vectors or faulty components can manifest themselves as subtle anomalies that conventional detection models do not detect.

Therefore, anomaly detection approaches (ADS) aim to model the normal operation of the CAN network and trigger alarms in the event of significant deviations, creating a broader and more adaptable defense mechanism against known and unknown threats.

This section provides an overview of current anomaly detection techniques specifically tailored to CAN bus systems and categorizes them based on the modeling techniques and detection strategies used to detect deviations from normal traffic behavior.

2.4.1 Statistical ADS

Li et al. presented an anomaly detection approach that aims to improve support vector data description (SVDD) by incorporating Markov- and Gaussian-based variants [64]. The method classifies incoming CAN messages, exploiting temporal characteristics to detect out-of-sequence attacks and kernel-based limits to capture data modifications. Evaluation on Luxgen U5 vehicle data with simulated attacks has confirmed its detection capability.

Following a similar approach, Dong et al. developed an ADS based on a multiple-observation hidden Markov model (HMM) that models both temporal and semantic patterns in CAN message identifiers and payloads [65]. By estimating the probability of each frame under multiple learned sequences, the system has identified various types of attacks in real time, including DDoS, fuzzy, replay, and masking scenarios.

Another lightweight ADS was introduced by Fang et al. using windowed Hamming distance to identify anomalies in CAN traffic [66]. By comparing sliding message windows to normal references and applying a statistical threshold, the system was able to detect injection intrusions such as DoS attacks, spoofing, and fuzzing, as validated by the Car-Hacking and other datasets.

2.4.2 Tradional Machine Learning-Based ADS

Machine learning-based approaches have gained traction for their ability to model behavioral patterns beyond static rule sets. While diverse in formulation, many recent systems share a focus on temporal modeling, supervised learning, or adaptability to open-world settings.

Maliha et al. presented a temporal anomaly detection framework that captures temporal dynamics in message intervals by constructing a latent state space using L1 quantile regression [67]. Anomalies were identified when stateful residuals —deviations from predicted latent behavior— exceed learned quantile thresholds. This design intended to make the system learn benign behavior thresholds without relying on message identifiers. The method was evaluated using the OTIDS dataset, demonstrating robust performance against DoS, fuzzy, and impersonation attacks under changing ON-OFF cycles.

Alternatively, Lalouani et al. proposed a multitask learning-based IDS that combines signal fingerprinting with contextual correlation of decoded messages [68]. By leveraging unique voltage signal fingerprints caused by semiconductor imperfections, the system accurately detects identity falsifications and packet manipulations. The method effectiveness was demonstrated using the Attack Free State dataset from OTIDS.

Du et al. have extended intrusion detection into the open-set recognition domain. They devel-

oped a clustering-based approach to improve class separation and lowers intra-class variation through metric learning. CLUSTER [69] was designed to be integrated into a vehicle-cloud framework leveraging its capacity for incremental learning and unknown class rejection. Its performance has been validated on the Car-Hacking dataset, where it outperformed reference methods such as OpenMax [70] and ACGAN-based [71] approaches.

2.4.3 Deep Learning-Based ADS

Building on the foundations laid by traditional machine learning techniques, recent research has increasingly turned to deep learning to overcome limitations in feature engineering and complexity. Deep learning approaches have emerged as powerful tools for anomaly detection, particularly due to their capacity to model nonlinear temporal dependencies and extract hierarchical latent representations. Among these, architectures leveraging recurrent networks, attention mechanisms, adversarial learning, and graph-based modeling have demonstrated noteworthy performance across various benchmarks and real-world datasets.

A prominent line of research emphasizes Generative Adversarial Networks (GANs), to improve model robustness and generalization. Hoang and Kim employed an adversarial autoencoder (AAE) that combines reconstruction learning with adversarial regularization using both labeled and unlabeled data [72]. The method constructs a latent representation of normal patterns while adversarial training enhances its capability to detect anomalies with minimal supervision. Evaluated on the Car-Hacking Dataset, the model showed strong detection capabilities even against unseen attacks with only 40% labeled data. DAdAE [73] builds on the foundational architecture of AAEs by introducing domain adversarial training to learn vehicle-invariant latent representations, enabling robust anomaly detection across different vehicle domains without retraining. In a complementary direction, Im et al. utilized a 1-D WaveGAN that transforms CAN payloads into audio-like waveforms [74]. Through GAN-based training on normal traffic from the Car-Hacking Dataset, the model learns to discriminate anomalous waveform patterns with high fidelity, achieving high detection rates across several attack types.

Another group of studies has focused on on integrating Long Short-Term Memory (LSTM) networks with attention mechanisms to enhance temporal sequence modeling. Zhao et al. proposed a model that couples LSTM with a self-attention module to analyze time-series physical semantics extracted from CAN-FD messages [75]. The LSTM component captures long-range dependencies, while attention enables the model to focus on salient input features. The system achieved an overall significant detection rates for DoS, fuzzing, and spoofing attacks from a real-world CAN-FD simulation dataset. Similarly, Gao et al. introduced a two-stage architecture combining LSTM layers with an attention mechanism

to form their Anomaly Classification Detection Model (ACDM) and Anomaly Data Detection Model (ADDM) [76]. Trained exclusively on normal data from the CAN-FD Intrusion Dataset collected by HCRL , the method demonstrated an improved classification accuracy over standard LSTM and CNN-LSTM baselines, showing resilience against a diverse set of attacks including scaling and ramp manipulations

Beyond sequential modeling and adversarial learning, some recent advances have applied graph-based deep learning, particularly Graph Neural Networks (GNNs), to exploit the structural and relational characteristics of CAN traffic. Xiao et al. formulated a graph attention network (GAT) framework in which each CAN message forms a node, and edges encode temporal relationships [77]. Evaluated using the HCRL dataset, the GAT assigns attention scores to critical connections, allowing the model to capture both spatial and temporal features critical for anomaly detection. The method indicated an acceptable detection metrics across all evaluated attacks, and by analyzing attention entropy, it offered an interpretable mechanism to identify influential message links.

In parallel, convolution-based strategies like the one proposed by Koltai et al. leverage temporal convolutional networks (TCNs) combined with correlation-based signal clustering to detect anomalies via time-series forecasting [78]. Signals are hierarchically grouped by Pearson correlation, and for each cluster, a TCN forecasts future values, flagging deviations as anomalies. This design was tested on an unlabeled real-world CAN trace dataset. It was able to mitigate the effects of isolated signal manipulation and demonstrates a high detection rate for modification attacks, outperforming the mentioned baseline approaches.

2.4.4 Hybrid ADS

Deep learning models, often accompanied by complementary machine learning techniques, have become a prevalent choice for in-vehicle anomaly detection due to their capacity to capture complex patterns and adapt to evolving threats.

One such approach is proposed by Mansourian et al. that blends ConvLSTM and LSTM architectures with a Gaussian Naïve Bayes (GNB) classifier [79]. The deep learning modules capture both spatial and temporal dependencies among ECU signals, while the GNB component statistically models the prediction errors to distinguish benign from malicious deviations. This combination proved highly effective on the Car Hacking Dataset, where the system outperformed traditional baselines like One-Class Support Vector Machine (OCSVM) and Isolation Forest.

Pushing the analysis into the time-frequency domain, Wang and Zhang integrated Continuous Wavelet Transform (CWT) with Convolutional Neural Networks (CNNs) to enhance anomaly

detection in automated vehicles [80]. CWT converts raw sensor data into two-dimensional scalograms, capturing both temporal and frequency features, which the CNN processes to identify abnormal patterns. Tested on open-source driving data mixed attack types, including hijacking and replay, the method achieved a reduction in false positivess.

Cheng et al. have also leveraged temporal features. presenting a deep stream clustering framework that combines a Sparse Regularized Convolutional Autoencoder (SRCAE) with a dynamic, unsupervised clustering algorithm [81]. While the SRCAE extracts compact spatial-temporal features, the evolving clustering mechanism continuously adapts to streaming CAN data to detect outliers in real time. The DESC-IDS framework was evaluated on the HCRL and ORNL datasets and achieved strong generalization to both known and unknown attacks.

Taking a more ensemble-based route, another contribution suggested by Maruf and Azim, leverages Naive Bayes, SVM, and DNN classifiers under a Triple Modular Redundancy (TMR) scheme to detect anomalies in CAN communication [82]. The method relies on model consensus to improve detection reliability and reduce the impact of individual classifier errors. Synthetic evaluations reported high classification accuracy and robustness.

2.4.5 Transformer-Based ADS

Expanding upon the strengths of deep learning architectures, transformer models have been applied to in-vehicle anomaly detection due to their ability to represent long-range dependencies and contextual relationships in sequential data. Originally introduced for natural language processing, transformers rely on self-attention mechanisms that assign dynamic weights to elements within a sequence, allowing the model to capture relevant temporal patterns without the limitations of recurrence or fixed receptive fields.

In the context of CAN bus communication, where message sequences often exhibit both local and global dependencies, this architecture offers a structured way to model normal traffic behavior across different message identifiers, payloads, and timing intervals. By learning the distribution of normal sequences in a self-supervised manner, transformers can be used to detect deviations associated with various types of anomalies.

Transformer-based anomaly detection systems for in-vehicle networks often rely on learning the sequence structure of CAN traffic from normal data and identifying deviations as anomalies. One such approach is CAN-Former [20] by Cobilean et al., which tokenizes each CAN message into one CAN ID and eight payload bytes. These tokens are passed through embedding layers and combined with positional encodings before being fed into a transformer decoder trained to predict the next token in the sequence. Anomalies are flagged when the

predicted token diverges from the actual one by more than a learned threshold. This method focuses on malfunction attacks, where adversaries reuse known IDs with altered payloads. Evaluations were conducted using the Survival Dataset, showing a decent detection rate across attacks such as malfunction injection, DoS, fuzzy, and replay.

Following a similar next-token prediction strategy, Nguyen et al. introduced the Transformer-Based Attention Network for In-Vehicle Intrusion Detection [83], which also encodes CAN IDs and payload bytes alongside positional information to capture temporal dependencies using a lightweight transformer decoder. This model compares predicted tokens to observed ones and flags discrepancies as anomalies. Like CAN-Former the method was tested on the Survival Dataset as well as the HCRL and Car Hacking datasets. It demonstrated solid performance across a variety of attack types.

While the previously discussed approaches rely on token-level prediction for anomaly detection, an alternative formulation is offered by the Fast-Gated Attention (FGA) transformer introduced by Yang et al. [25], which shares the focus on temporal modeling but shifts the objective toward sequence prediction. Rather than predicting the next token, FGA learns to forecast future multivariate signals derived from normalized payloads and sparsely encoded CAN IDs, using reconstruction error to identify anomalies. Despite differences in input encoding and detection logic, FGA aligns with CAN-Former and the Attention Network model in leveraging raw CAN ID and payload data for sequence learning. Emphasizing real-time feasibility, FGA incorporates cross-attention windows and linearized attention mechanisms to achieve inference speeds exceeding 3000 FPS. Its effectiveness was validated on the Car-Hacking and SynCAN datasets, demonstrating resilience to signal injection and fuzzing attacks.

In contrast, models like In-Vehicle Network Using Self-Supervised Learning (IVNSL) [21] by Cao et al. and CAN-BERT [22] by Alkhatib et al. operate on different principles. IVNSL extracts physical signals from payloads to reduce dimensionality and masks random signals within sequences, allowing a hierarchical transformer to reconstruct them based on context. This transformer architecture separates spatial and temporal dependencies and enables anomaly detection through deviations in reconstructed signals. IVNSL also supports vehicle-cloud collaboration to adapt to concept drift—when significant changes are detected in onboard signal distributions, the vehicle requests retraining from the cloud. Evaluated on the Car-Hacking Dataset, IVNSL was reported achieving perfect detection rates across DoS, spoofing, and fuzzing attacks. In parallel efforts, CAN-BERT leverages masked language modeling applied solely to CAN ID sequences and payload, omitting timing data. It trains a bidirectional BERT transformer on unlabeled data by masking random CAN IDs and pre-

dicting them from surrounding context, under the assumption that deviations from learned ID patterns signify anomalies. This lightweight and label-free approach showed high performance on both the Car-Hacking and Survival datasets and demonstrated strong real-time performance with inference times under 1 ms.

2.4.6 Identified Gap and Proposed Direction

As reviewed in the preceding section, recent anomaly detection systems (ADS) for CAN bus networks have made significant progress by satisfying open-set conditions, demonstrating the ability to detect deviations from normal vehicle behavior without relying on prior knowledge of attack types.

Despite these advancements, these ADS remain heavily dependent on benchmark datasets that have been criticized earlier in this chapter for their imperfections (section 2.2). Datasets such as OTIDS, Car-Hacking, Survival, and HCRL often fail to capture the full complexity of CAN bus signal generation and authentic noise distributions. Consequently, models tuned on these datasets demonstrate inflated accuracy but struggle in production-grade environments, where system states drift continuously.

Moreover, most of these approaches rarely model the intrinsic data-generation process of CAN signals. Even advanced architectures such as LSTMs, CNNs, GANs, BCNNs, and hybrid statistical ML solutions, operate primarily as opaque classifiers or anomaly recognizers. Their internal representations are largely inaccessible, limiting their applicability for root-cause analysis or safety-critical decision-making. The lack of interpretable intermediate states is especially problematic in automotive cybersecurity, where standards increasingly demand explainable and auditable behavior-monitoring mechanisms.

Another critical limitation lies in the obscurity of how many ADS handle CAN arbitration and detection thresholds. These aspects are often left unspecified, implicitly relying on hand-tuned parameters. This opacity undermines trust in the system and reduces confidence in its predictions for safety-critical environments.

Finally, complex CAN bus variants, such as the J1939 protocol, remain largely underexplored in anomaly detection research. Most existing models target passenger vehicles or CAN 2.0 networks, limiting applicability to heavy-duty fleets with richer PGN structures, multi-signal payloads, and higher operational diversity.

Taken together, while anomaly detection methods have advanced open-set detection and reduced dependence on explicit attack knowledge, they remain constrained by dataset limitations, lack of interpretability, underexplored complex protocols, and hand-tuned assump-

tions.

In this thesis, we develop a comprehensive framework that leverages the strengths of existing approaches while addressing these gaps by explicitly modeling the intrinsic CAN signal process, incorporating interpretable intermediate states, and carefully accounting for arbitration and thresholding to improve trust and deployability in production-grade vehicular networks.

2.5 Chapter Summary.

This chapter outlined the foundations of anomaly detection in the CAN bus, beginning with the protocol's architecture and inherent weaknesses. Its deterministic design, lack of authentication, and shared medium expose structural vulnerabilities that modern attacks readily exploit. When analyzed through the lens of the CIA Triad, these attack vectors demonstrate their broad impact on system security, affecting data trust, message integrity, and real-time system availability.

We then examined the primary benchmark datasets used in CAN security research, emphasizing their limitations in realism, attack diversity, temporal coverage, and generalizability, factors that constrain fair evaluation of IDS and ADS models.

Our review of intrusion detection approaches highlighted persistent gaps: reliance on closed-set assumptions, sensitivity to distribution shifts, and limited ability to generalize to unseen attacks. The survey of anomaly detection techniques further revealed challenges in modeling payload–timing interactions, maintaining clear traceability of anomalous behaviors and supporting open-set operational conditions.

Overall, this chapter clarified the technological landscape while exposing the shortcomings of existing datasets and detection pipelines. These gaps motivate the need for a more adaptive, sequence and content-aware, anomaly detection framework developed in the following chapters.

CHAPTER 3 ARTICLE 1: BEYOND BLACK-BOX DETECTION: INTERPRETABLE TRANSFORMER-SVDD FRAMEWORK FOR UNSEEN ANOMALIES IN HEAVY VEHICLE CAN BUS

Authors Ranim Rahali[1], Simon Bellemare[2], Omar Abdel Wahab[1], Audrey Lacoursiere Lamonde[2], Adel Abusitta[1]

Institutions [1] Polytechnique Montréal, Canada; [2] CortAIx Labs, Thales Canada, Québec, Canada.

This article was submitted to the Journal IEEE Transactions on Dependable and Secure Computing on 2 December 2025.

Abstract

Modern vehicle networks exhibit high complexity and substantial noise, making anomaly and attack detection challenging and prone to false positives. Among these networks, the J1939 protocol, commonly deployed in heavy-duty and commercial vehicles, manages critical in-vehicle communications across multiple electronic control units (ECUs). Accurately detecting subtle deviations in J1939 is crucial, as even minor deviations in normal behavior can create safety risks and costly downtime, and therefore must be captured with explainable insights. Traditional rule-based systems struggle to detect subtle and unknown attacks, while AI-based solutions often operate as black boxes, limiting traceability of alerts. To address this, we propose a Transformer-Powered Interpretable Decision Fusion Framework (TransVDD) designed to build a robust understanding of normal system behavior without relying on predefined attack signatures, labeled anomalies, or prior knowledge of threats. Each model leverages the attention mechanism of Transformers to capture distinct behavioral patterns, while anomaly scores are computed relative to a Support Vector Data Description (SVDD) trained on normal data, thereby avoiding the subjectivity of static thresholds. Fusing these scores enables robust detection of unseen and evolving anomalies. Applied to real-world J1939 data, our approach achieves near-zero false positives while detecting unseen anomalies with a true positive rate up to 0.889, approximately 8.7% higher than baseline models, demonstrating robust alignment with real-world vehicular behavior.

Keywords J1939 protocol, anomaly detection, Novel attacks, Transformers, SVDD, interpretable AI.

3.1 Introduction

Driven by the rapid evolution of the Internet of Things (IoT), today's infrastructure operates as a unified, data-centric ecosystem. Vehicles now function as intelligent, networked systems in which Electronic Control Units (ECUs) communicate and collaborate to maintain control, enhance safety and reduce delays and inefficiencies.

By leveraging this interconnected framework, vehicles have evolved from isolated mechanical units into integrated electronic systems capable of more efficient internal coordination and smart control. Real-time communication among ECUs not only optimizes individual performance but also enables secure traffic management and resilient operation across broader transportation networks. This evolution has dramatically increased both the number of ECUs and the complexity of in-vehicle communication networks. To accommodate growing data exchange requirements, modern vehicles now integrate multiple communication standards, such as the classical Controller Area Network (CAN) [27], [6] and the more recent Automotive Ethernet (100Base-T1) [84], which offers higher data throughput. Originally developed by Bosch in 1986 [85], the CAN bus was designed to provide reliable and efficient communication among ECUs without the need for a central controller. Its robust and real-time characteristics made it the backbone of in-vehicle communication, enabling dependable coordination across vehicle subsystems. Despite the emergence of newer technologies, CAN remains dominant due to its cost efficiency, reliability, and maturity.

However, the evolution of vehicular networks has introduced significant security challenges. Modern vehicles are increasingly complex, with numerous ECUs communicating over dense networks, generating high traffic and noise that complicate anomaly and attack detection. The classical CAN bus, despite its durability and performance advantages, was not built with intrinsic security mechanisms such as authentication or encryption [28]. As a result, it is vulnerable to spoofing, replay, and injection attacks, many of which can be executed through direct physical access, for example via the On-Board Diagnostic (OBD) port, or even remotely via compromised telematics or infotainment interfaces [29]. Numerous exploit scenarios were detailed in [34] - [47], including masquerade and denial-of-service (DoS) attacks [30], demonstrating that the original safety claims of CAN do not guarantee immunity against malicious activity.

As connectivity expands, the attack surface broadens [33], particularly in heavy-duty and industrial vehicles where the SAE J1939 protocol is dominant. Unlike isolated consumer-vehicle, these vehicles underpin supply chains and fleet operations where attacks can cascade across fleets and industrial systems, disrupt freight and critical services, and create high-value

economic and operational leverage for extortion or sabotage. J1939 extends the CAN bus to handle higher-level messaging and standardize data exchange across complex subsystems. It provides a common set of standards and specifications, enabling reliable and interoperable communication between ECUs from different. Its openness and critical role in system safety and performance make it a high-value target for attackers [86]. To identify such vulnerabilities, many Intrusion Detection System (IDS) solutions, mostly relying on predefined rules and signatures, have been proposed.

These methods can identify known attacks, but often struggle with new, subtle, or evolving anomalies, requiring extensive manual tuning for each network configuration [87]. More recently, AI-based data-driven solutions have been proposed to automatically learn behavioral patterns from network traffic, offering greater adaptability and detection coverage. However, these AI approaches frequently function as black boxes, limiting interpretability and operational confidence which is a crucial factor in safety-critical domains [88].

Despite notable progress in CAN bus security research, existing intrusion detection solutions remain constrained by fundamental limitations: rule-based rigidity, poor sensitivity to subtle or evolving attacks, and the opaque "black-box" nature of many deep learning models that hinders operational confidence. Moreover, the vast majority of these approaches overlook the SAE J1939 protocol the complexities. Its layered message hierarchies, strict temporal dependencies, and structured payloads create dynamics that standard CAN traffic does not exhibit, leaving heavy vehicles more exposed and underprotected against attacks. This risk is amplified by the protocol's deployment in mission-critical domains such as commercial transportation, construction machinery, agricultural equipment, public transit, and defense logistics. In these environments, any disruption or manipulation of in-vehicle communication can rapidly cascade into financial, safety, and infrastructure-level consequences.

To address these limitations, we propose a Transformer-Powered Interpretable Anomaly Detection Framework for the J1939 CAN bus (TransVDD). Rather than focusing solely on known attack patterns, our approach targets anomalies in a broader sense, aiming to detect both evident and subtle deviations from normal behavior that may signify emerging or previously unseen attacks. The framework builds a comprehensive understanding of normal system dynamics without relying on predefined attack signatures, labeled anomalies, or prior threat knowledge.

The first step of our solution consists in analyzing data to extract meaningful timing and payload features, which are then used to construct a structured representation that preserves contextual relationships and traceability. This representation facilitates expert interpretation of model outputs and enables the derivation of thresholds based on genuine behavioral

patterns rather than arbitrary parameters. Transformer-based models are employed for their demonstrated ability to capture sequential dependencies and complex behavioral dynamics [89].

Their attention mechanisms allow the network to identify subtle contextual cues and long-range correlations across time and payload sequences, effectively modeling normal operational behavior. Anomaly scores are then computed relative to a Support Vector Data Description (SVDD) trained exclusively on normal latent representation, ensuring thresholds reflect true behavioral distributions. Applied to real-world J1939 datasets, the proposed approach reliably identifies deviations in timing and payload patterns while providing interpretable, expert-traceable insights aligned with the operational and safety requirements of heavy-duty vehicle networks.

The main contributions of this paper are summarized as follows:

- An **Insight-Oriented Multi-modal Transformer-SVDD** architecture that operates under real open-set detection conditions, uncovering unknown and subtle anomalies in heavy-vehicle CAN traffic.
- A **Dual-path Design** of our detection solution that separates temporal evolution from payload semantics. This enables a structured decision-fusion mechanism that mitigates feature interference and strengthens detection fidelity.
- An **Interpretable Representation Layer** that structures the timing and payload embeddings to reveal the model's internal reasoning. This helps shift a traditionally opaque sequence model into an insight engine, which is crucial in security analysis.
- A **Data-Driven Anomaly Boundaries** component that generates adaptive detection thresholds, which reduces the reliance on brittle heuristics that are commonly used in many existing detection approaches.
- An **Anomaly Scoring Mechanism** that provides a transparent and quantitative measure of deviation. This enables experts to trace the predictions to specific signal features and assess the underlying severity.

The remaining sections of this paper are as follows: In Section 3.2, we provide an overview of related works and key background information. Section 3.3 presents the threat model. Section 3.4 describes in details the architecture of our Anomaly detection framework. In Section 3.5, we present the performance evaluation of our solution. Section 3.6 delivers a comprehensive discussion. Finally, Section 3.7 concludes this article and includes future work recommendations.

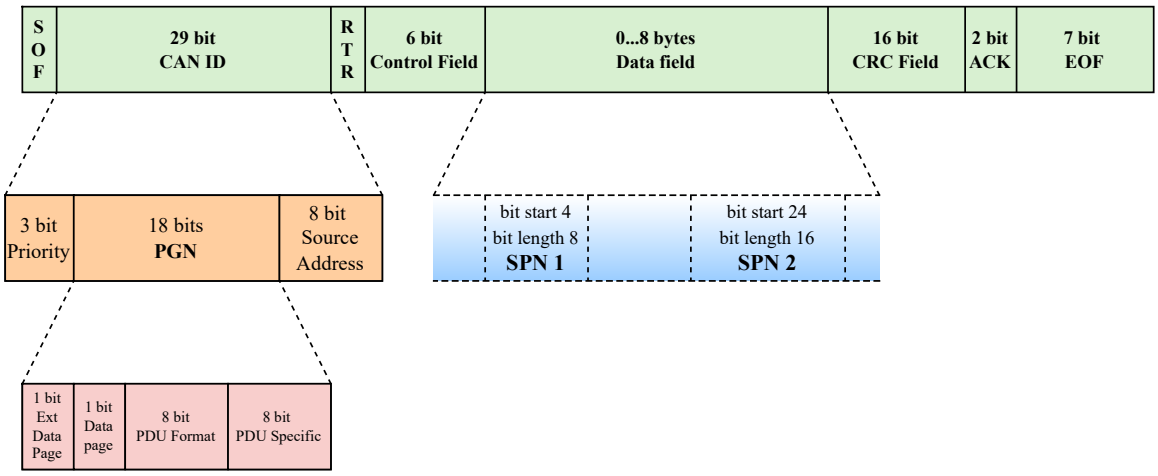


Figure 3.1 J1939 CAN frame structure. Acronyms: SOF = Start of Frame, PGN = Parameter Group Number, PDU = Protocol Data Unit, CRC = Cyclic Redundancy Check, ACK = Acknowledgment, EOF = End of Frame

3.2 Related work

The intrinsic vulnerabilities of the CAN bus render the network highly susceptible to both stealthy and disruptive attacks. Consequently, developing robust CAN bus intrusion detection systems (IDS) or anomaly detection systems (ADS) presents a significant challenge. This has driven intrusion detection research for CAN bus systems to evolve beyond the traditional reliance on rule-based and signature-based methods. These approaches, while foundational, now serve primarily as preliminary layers within more sophisticated, multi-stage detection pipelines.

We surveyed recent literature with a specific focus on intrusion detection approaches and identified several notable methodologies. Most of designs increasingly incorporate statistical modeling [52], machine learning [53], [54], [68], [56], and deep learning [57], [58], [59], [60] to address the complexity and dynamism of vehicular networks. Although these IDS approaches demonstrate adaptability across diverse scenarios [36], [7], [38], [41], [43], and resource-efficient designs, many rely on labeled or simulated datasets that may not capture real-world variability, making them less effective against novel or evolving threats that fall outside their training scope.

Building on these limitations, recent research has increasingly turned toward anomaly detection as a complementary or alternative strategy. Approaches such as those by Li et al. [64], Dong et al. [65], and Fang et al. [66] demonstrate the strength of statistical modeling in cap-

turing temporal irregularities, semantic deviations, and injection-based anomalies in CAN traffic. Their lightweight design and reliance on probabilistic or distance-based metrics make them particularly suitable for deployment in resource-constrained vehicular environments. Nonetheless, these approaches also share several inherent limitations. Their performance often depends on stable statistical baselines, which can degrade under dynamic driving conditions or evolving network configurations. Notably, they generally lack the ability to model deep contextual dependencies across message sequences, constraining their effectiveness in identifying subtle, coordinated, or multi-stage intrusion patterns.

Machine learning and deep learning–based anomaly detection systems (ADSs) share a powerful commonality. They excel at modeling complex behavioral patterns in CAN traffic beyond static rule sets. Whether through supervised learning, temporal modeling, or unsupervised clustering, these approaches are designed to learn nuanced representations of normal vehicle behavior and flag deviations with high precision. Techniques such as Maliha et al.'s [67]latent state modeling, Anjum et al.'s [55] XGBoost classifier, and Du et al.'s [69] open-set recognition framework highlight the strength of machine learning in adapting to dynamic environments and rejecting unknown intrusions. Deep learning methods further enhance this capability by capturing nonlinear dependencies, extracting hierarchical features, and leveraging architectures like LSTM [75], GANs [72], and GNNs [77] to detect evolving attack patterns. What distinguishes these systems is their ability to generalize across domains and operate with minimal supervision. For example, Hoang and Kim's [72] adversarial autoencoder and DAdAE [73] demonstrates robust detection even with limited labeled data, while Im et al.'s [74] WaveGAN and Gao et al.'s [76] attention-enhanced LSTM models show resilience against diverse attack types. Xiao et al.'s [77] graph attention network introduces semantic visibility by assigning attention scores to message relationships, and Koltai et al.'s [78] TCN-based forecasting offers a scalable solution for real-time anomaly detection in unlabeled environments.

Transformer-based anomaly detection systems have recently emerged and been adopted by several researchers. They showed ability to learn the structure of normal CAN traffic in a self-supervised manner and detect anomalies as deviations from expected sequences. Models like CAN-Former [20] and Nguyen et al.'s [83] Attention Network tokenize CAN IDs and payloads to predict next-message tokens, while Yang et al.'s [25] Fast-Gated Attention (FGA) focuses on forecasting multivariate signals using reconstruction error. Others, like IVNSL [21] and CAN-BERT [22], apply masked modeling to reconstruct missing signals or predict masked CAN IDs, capturing both spatial and temporal dependencies without depending on labeled data. Even though it was not validated on CAN Bus data, the model TranAD [90] leverages an adversarial-training regime on its Transformer architecture and validates on multiple

time-series datasets using adaptive thresholding, illustrating complementary strategies that could be adapted to vehicular anomaly detection.

What sets these approaches apart is their flexibility in data representation by combining tailored input encodings with attention mechanisms. However, these systems exhibit certain limitations. Simplified tokenization has obscured nuanced payload semantics, particularly when timing or signal-level information is excluded. Deep learning architectures also face limited interpretability, which complicates explainability and validation in fault-intolerant automotive contexts. Exceptionally, Transformers provide greater transparency via attention maps however attention alone is insufficient, as it only highlights model focus areas and requires additional context.

Moreover, most existing models rely on hardcoded detection thresholds, which restrict adaptability across varying operating conditions and vehicle networks. Ultimately, their performance often depends on clean and well-structured benchmark datasets, with limited evaluation on complex and safety-critical protocols such as SAE J1939 [1], leaving robustness under realistic in-vehicle conditions largely unverified.

Despite its critical role in heavy-duty vehicle communication, the J1939 protocol has largely been left behind in the evolution of anomaly detection research.

The security weaknesses inherent in SAE J1939-specific features were first analyzed, leading to the proposal of an authentication protocol in [91]. Building on this understanding of vulnerabilities, Burakova et al. [92] demonstrated that injection and replay attacks on J1939 CAN buses within commercial vehicles can result in severe consequences. Further highlighting the protocol's susceptibility, Mukherjee et al. [93] investigated a Denial-of-Service (DoS) attack targeting the J1939 transport protocols responsible for multi-packet transmissions.

To mitigate these vulnerabilities, several defense strategies have been proposed. Among them, the use of encrypted J1939 communication that was evaluated in [94], but only in the context of diagnostic messages. Later, a specification-based approach proposed by Camil et al. [95] leverages J1939 message semantics and procedures to detect and neutralize malicious frames in real time. Complementing these efforts, another approach has integrated timing and data analysis to identify spoofing and masquerade attacks in both J1939 and NMEA2000 networks [96]. From a different architectural standpoint, TruckSentry [97] introduces a next-generation, stateful firewall designed specifically for SAE J1939 networks in medium- and heavy-duty vehicles. It adopts a rule-based mechanism that evaluates transmission intervals and contextual parameters in real time, allowing the system to classify and block malicious

[1]SAE specification documents for J1939 are available at: https://www.sae.org/standards/j1939_202306-serial-control-communications-heavy-duty-vehicle-network-top-level-document

messages during transmission.

To the best of current knowledge, Hossein et al. [98] propose the only modular framework that profiles PGNs and extracts live CAN features to perform real-time classification using dedicated machine learning models. This highlights that, despite advances in machine learning, studies targeting J1939 remain scarce, leaving the protocol comparatively underexplored for anomaly detection and the identification of previously unseen cyberattacks.

Addressing these limitations, our approach TranSVDD leverages Transformer-based sequence modeling with SPN global vectors and HDBSCAN clustering to capture both temporal and payload-level anomalies. Adaptive SVDD thresholding enables detection of previously unseen deviations while maintaining interpretability, allowing precise identification of abnormal messages and signals in real-world J1939 CAN traffic. Further details are provided in the following section.

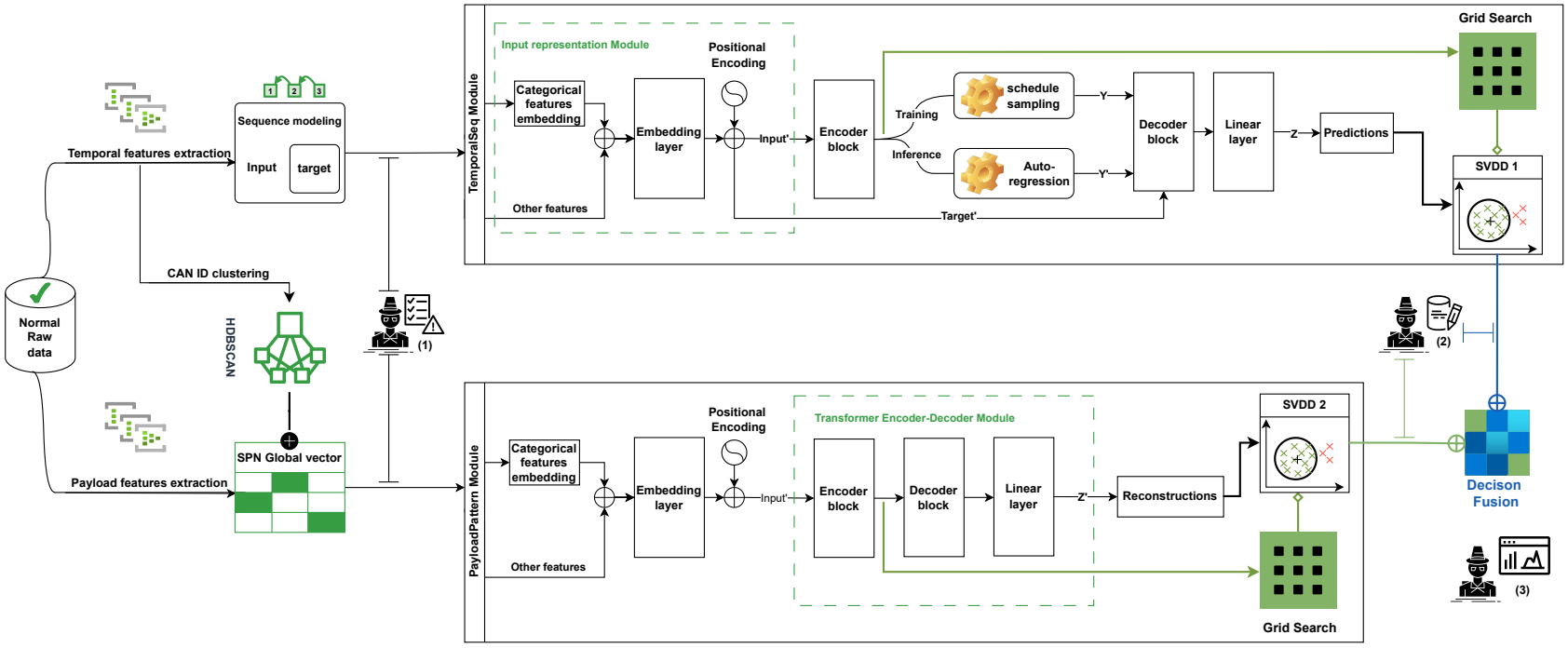


Figure 3.2 Architecture of the proposed TransVDD framework. Raw CAN data are processed into temporal and payload feature modules, with CAN IDs clustered via HDBSCAN to generate a global SPN vector. Both modules use transformer-based encoder-decoder structures—temporal features for sequence prediction, payload features for reconstruction. SVDD models evaluate outputs in parallel, and final anomaly decisions are obtained through fusion of both SVDD scores

3.3 Threat Model

Our framework targets active adversaries capable of injecting, replaying, or modifying J1939 traffic on the vehicle's high-speed CAN bus, as the focus is on detecting behavioral deviations that manifest through transmitted frames. We assume the adversary has access to the network through a standard CAN controller via a compromised telematics unit, aftermarket device, or misconfigured diagnostic interface. Consistent with modern heavy-duty vehicle designs, we do not consider attackers who can bypass the controller and manipulate the bus at the electrical or bit-timing level. All injected traffic must therefore conform to standard CAN arbitration, frame formatting, and error-handling rules. These constraints ensure the adversary must manifest detectable timing or payload deviations in the observable message stream, which is precisely the domain TransVDD monitors, including previously unseen attacks with only subtle departures from nominal behavior.

Within this constraint, the adversary is assumed to possess typical platform-agnostic knowledge of J1939 networking, .i.e., message periodicity, arbitration priorities, and the ability to record and replay valid traffic. They are not assumed to have proprietary Interface Control Documents (ICDs) or Original Equipment Manufacturer (OEM)-specific signal definitions. This capability level aligns with realistic remote-access compromises and widely available automotive security tools.

Under these assumptions, the system is exposed to three operational threat categories:

- Availability attacks, where the adversary degrades network performance by flooding high-priority or legitimate CAN IDs.
- Integrity attacks, where authentic-looking traffic is replayed, spoofed, or temporally misaligned to disrupt subsystem coordination.
- Confidentiality attacks, including low-rate payload modification designed to evade naïve detectors.

The attack scenarios defined in our evaluation (Section 3.5.1) instantiate these capabilities directly. Each stresses a distinct axis of the J1939 threat landscape while remaining fully consistent with CAN-controller-constrained adversaries.

3.4 Proposed Transformer-SVDD Detection Framework: TransVDD

In the following section, we provide a detailed description of the implemented anomaly detection framework. This section outlines the core architecture and highlights the essential

components of the anomaly detection pipeline for J1939 CAN bus data. Figure 3.2 presents the overall framework, where the pipeline is structured into the following main stages:

- Data Preprocessing and Feature Extraction: Raw J1939 CAN data transformation into a structured format, including timestamp and CAN ID representation, and SPN-based feature computation.
- Input Representation Module: Encode categorical features (CAN IDs and clusters IDs) and numerical features into embeddings, and add positional encodings to capture sequential information.
- Transformer Encoder–Decoder Module: Process sequences of embedded messages to capture temporal and inter-signal dependencies, producing latent representations for anomaly detection.
- Support Vector Data Description (SVDD) Module: Compute distances in the latent space to identify deviations from normal behavior, providing a score for anomaly detection.
- Temporal Anomaly Detection module: Capture temporal dependencies across sequences of messages to identify anomalies based on unusual timing patterns and correlations between signals.
- Payload Anomaly Detection Module: Learn the normal patterns of payload values and detect deviations indicating anomalies in message content.
- Decision Fusion Module: Combine outputs from temporal and payload-based models to produce a final anomaly decision per message, improving overall detection accuracy.
- Interpretability & Expert Insight Module: Translate model outputs, embeddings, and anomaly scores into human-readable explanations, enabling domain experts to understand, validate, and act on detected anomalies.

3.4.1 Data Preprocessing and Feature Extraction

Raw CAN/J1939 data description

The raw J1939/CAN data detailed in figure 3.3 is recorded as timestamped messages. Each entry consists of a timestamp, a CAN identifier, and an 8-byte payload. The timestamp captures precise message timing, which is critical for analyzing inter-arrival times and delays.

The identifier, as shown in figure 3.1, encodes priority, the Parameter Group Number (PGN), and the source address, thereby indicating the message type and origin ECU [6]. The payload contains raw bytes that map to Suspect Parameter Numbers (SPNs), representing physical signals such as engine speed, torque, or temperatures. Starting from such raw data is essential in anomaly detection, since irregularities can occur in timing, identifiers, or payload content. However, this representation is complex and low-level, with meaningful semantics hidden in protocol definitions [99]. Therefore, feature extraction is required to transform the raw stream into more compact and semantically rich representations that better capture system behavior and provide a stronger basis for detecting anomalies.

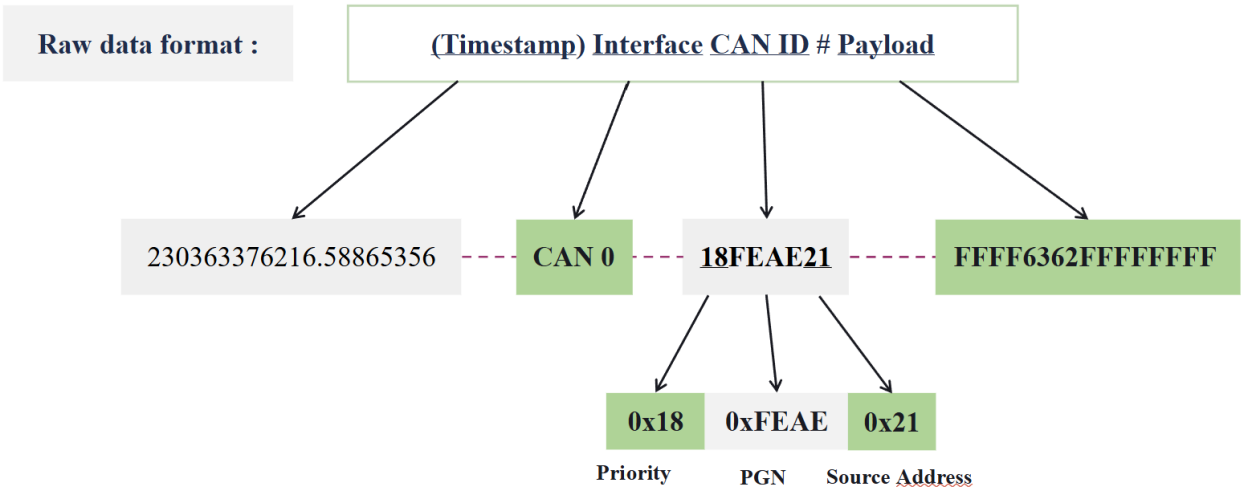


Figure 3.3 Raw CAN message format (J1939, PDU2) with Timestamp, Interface, 29-bit Identifier, and Payload

Temporal features extraction

In CAN Bus, temporal patterns are critical, as anomalies often manifest as irregular message timing. After evaluating several strategies, we focus on **time deltas augmented with dynamically updated statistics and decay factors** as core temporal features. This choice is guided by two key questions:

1. How can we detect irregularities in timing without being overly sensitive to absolute timestamps?
2. How can we ensure adaptability to new data while minimizing false positives?

We distinguish two types of time deltas for each message:

1. **Global time delta**: the time difference between a newly received message and the immediately preceding message, irrespective of its CAN ID:

$$d_{\text{delta},i} = t_i - t_{i-1} \tag{3.1}$$

2. **CAN ID-Specific delta**: the time difference between the current message and the previous message with the *same CAN ID*:

$$\Delta t_{\text{CAN},i} = t_i^{\text{CAN}} - t_{i-1}^{\text{CAN}} \tag{3.2}$$

Using the CAN ID-specific delta, we compute a **Decayed mean** that adapts to changes in message patterns over time. Let μ_{i-1}^{CAN} denote the previous decayed mean for that CAN ID. Upon receiving a new delta $\Delta t_{\text{CAN},i}$, we update it using a decay factor α :

$$\mu_i^{\text{CAN}} = \alpha\, \mu_{i-1}^{\text{CAN}} + (1-\alpha)\, \Delta t_{\text{CAN},i} \tag{3.3}$$

This ensures older intervals gradually lose influence, allowing the model to adapt to *new timing patterns* without being biased by outdated statistics.

The temporal feature vector for each message is then:

$$\mathbf{x}_i^{\text{temporal}} = [\tilde{d}_{\text{delta},i}, \tilde{\mu}_i^{\text{CAN}}, \tilde{\Delta} t_{\text{CAN},i}] \tag{3.4}$$

where $\tilde{\cdot}$ indicates normalization.

This temporal feature representation allows the model to dynamically adapt to variations in message timing by combining multiple measures of temporal behavior. The **global time delta** captures system-wide irregularities in message arrival, reflecting short-term fluctuations across the entire bus. The **CAN ID-specific delta** together with the **Decayed mean** captures the expected timing behavior for each CAN ID, allowing the detection of localized anomalies.

The decay factor must be carefully tuned to balance responsiveness and stability. A small decay factor emphasizes recent observations, allowing rapid detection of sudden anomalies but potentially overreacting to normal short-term fluctuations. Conversely, a large decay factor emphasizes historical data, providing stability and reducing false positives, yet it may

delay detection of genuine anomalies. Domain experts or system designers empirically tune it based on expected CAN traffic variability, operational context, anomaly sensitivity, and the need to ignore pauses in data collection (as encountered in our experimental setup) , ensuring the model reacts to true behavioral changes while remaining robust to evolving message intervals.

Payload features extraction

The payload encodes the actual vehicle signals. Anomalies often manifest as irregular signal values or unusual payload structures. Capturing abnormal behavior at this level requires presenting this information in a form the model can interpret, so it can differentiate between normal variations and genuine anomalies. To achieve this, we combine signal extraction, clustering of CAN IDs, and a global vector representation.

Signal extraction On this step, we primarily use the Database Container (DBC), the standard CAN database format that defines how raw message payload bytes map to physical signals, including scaling, offsets, units, and signal lengths.

For CAN IDs with available DBC decoding, we extract signals corresponding to each message and store them in a consistent manner, ensuring proper alignment across messages. This allows the model to interpret meaningful signal values, such as engine speed, fuel rate, or brake pressure, without relying on raw byte statistics. For signals that are missing or undefined in the available DBC version, reconstruction can be performed using methods such as CAN-D (CAN Decoder) [100]. CAN-D is a modular pipeline that determines each signal's start bit, length, endianness, and signedness, and uses diagnostic standards along with signal boundary classifiers to infer previously undefined signals. This process enables recovery of missing signal information, increasing the set of CAN IDs that can be analyzed for anomaly detection.

CAN ID clustering To monitor and analyze CAN bus messages, it is helpful to group CAN IDs into clusters. The assignment of clusters will reduce the complexity of modeling by grouping messages with similar patterns. This will provide more explainable results, making it easier to trace abnormal behaviors or potential attacks to specific message types or homogeneous subsystems. In order to cluster CAN IDs meaningfully, we considered criteria that capture both the identity and behavior of messages. These criteria fall into three categories:

- **Categorical criteria:** Reflect protocol-level metadata, representing the functional role of a message, the source address identifying the sending ECU, the CAN ID struc-

ture (broadcast versus peer-to-peer), and message frequency class (high- versus low-frequency messages).

- **Behavioral criteria:** Capture temporal and statistical patterns, such as payload byte stability, bit or byte entropy, correlation over time, time between messages, and burstiness.

- **Technical criteria** Describe signal and byte-level characteristics, including payload length consistency, byte-level roles, signal scaling and range, payload transition rates, and structural similarities like recurring checksums or counters.

These criteria are translated into statistical features to serve as input for clustering. For example, payload byte stability is mapped to metrics such as **rate of change** and standard deviation, as discussed in the context of CAN bus data analysis. Correlation over time can be represented by **variance**, autocorrelation, or skewness, which are standard measures in time series analysis [101]. Bit or byte entropy is quantified using **entropy** or kurtosis, as explored in studies on signal processing and data analysis [102] . Timing information, including the time between messages, is captured through **average intervals** and standard deviation, methods commonly used in network traffic analysis [103]. Categorical features like PGN are treated as knowledge-based labels, providing functional context for the clustering process. Grouping messages with similar functional and behavioral characteristics increases the interpretability of detected anomalies, as deviations can be traced back to specific clusters and their underlying features, enhancing the explainability and traceability of the anomaly detection process.

Given the heterogeneous nature of the resulting feature space, which includes non-spherical clusters, varying densities, and potential outliers, traditional clustering methods such as K-Means or K-Medoids are unsuitable. These methods assume spherical, equally sized clusters and are sensitive to noise. Instead, we use Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) [104], a density-based clustering algorithm that automatically determines the number of clusters, handles varying densities, and identifies noise points. HDBSCAN is particularly well-suited for CAN bus data, where message patterns vary widely and some messages may not clearly belong to any cluster. Hierarchical clustering is usually employed when it is valuable to visually analyze the relationships and structure of clusters using dendrograms (Figure 3.6).

To handle varying densities on our feature vectors (Average interval, variance, entropy, and rate of change), HDBSCAN [105] defines the distance between two points p and q as:

$$d_{\text{mreach}}(p, q) = \max\left(\text{core}_k(p),\ \text{core}_k(q),\ d(p, q)\right) \tag{3.5}$$

where:

- $d(p, q)$ is the original distance between p and q (e.g., Euclidean),
- $\text{core}_k(p)$ is the distance from p to its k-th nearest neighbor (core distance).

This ensures that sparse regions (low density) do not distort clustering.

Using the mutual reachability distance, HDBSCAN builds a Minimum Spanning Tree (MST) that represents the hierarchical relationship between points and their densities.

The hierarchy is pruned by selecting clusters with the highest *stability*, defined as:

$$\text{Stability}(C) = \sum_{p \in C} \left(\lambda_{p,\text{death}} - \lambda_{p,\text{birth}}\right), \quad \lambda = \frac{1}{\text{distance}} \tag{3.6}$$

is a density measure. Clusters with higher stability are more "persistent" and considered meaningful, while low-stability ones are discarded as noise.

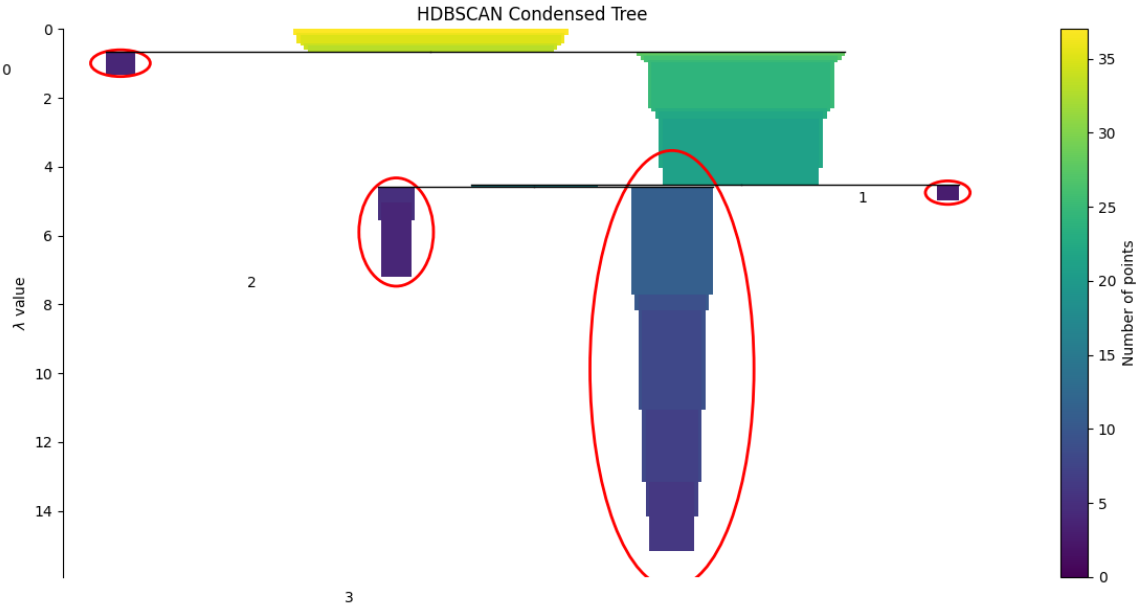


Figure 3.4 HDBSCAN condensed tree generated from 35 CAN identifiers. The vertical axis ((λ) value) indicates cluster stability, while the color gradient encodes cluster size (0–35 points). Red ellipses highlight clusters of interest across varying stability levels, illustrating the trade-off between compact, high-density clusters and broader, more stable groupings

Signal global vectors The challenge in representing CAN payloads for anomaly detection is to maintain a flexible and interpretable design for the input format. Drawing from conclusions and observations of prior approaches the representation should:

- Preserve individual signal identity to detect deviations at a fine-grained level.
- Record interactions between signals originating from different CANID messages.
- Remain compact even as the number of signals grows.

Different encoding strategies were considered, including per-message modeling [20], sparse matrix encodings [25], temporal interpolation [106], and raw binary representations [34]. While per-message approaches are effective for local anomaly detection, they cannot capture dependencies across different CAN IDs. Temporal interpolation risks smoothing out sharp anomalies, and raw binary encodings, while universal, remain difficult to interpret. Although sparse triplets are known to scale well, our experiments (Figure3.5) showed that their latent embeddings tend to collapse, reducing discriminative power.

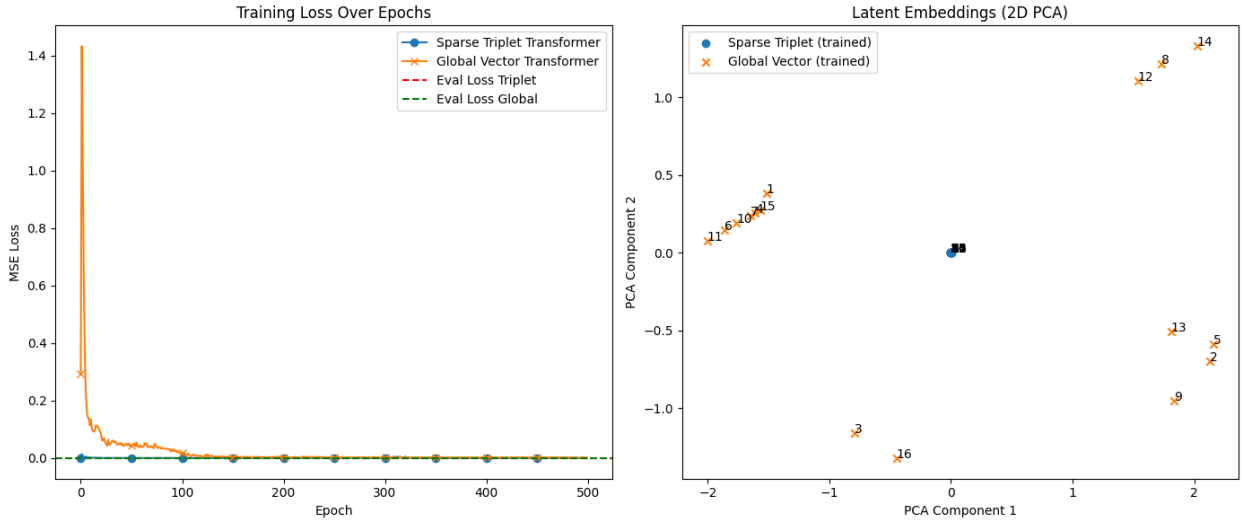


Figure 3.5 Comparative performance of Sparse Triplet Transformer and Global Vector Transformer models. (Left) Training and evaluation loss over 500 epochs, showing rapid convergence within the first 50 epochs and stabilization near zero. (Right) 2D PCA projection of latent embeddings, where Sparse Triplet embeddings cluster tightly near the origin, while Global Vector embeddings are more dispersed across 35 CAN identifiers, illustrating differences in representation compactness and distribution.

The *multi-ID global vector with masking* [107] offers a balance between interpretability and compatibility with Transformer architectures. This context-aware vector allows the model

to learn activation pattern of signals from multiple messages at once. This representation allows detection of system-wide inconsistencies and minor cross-signal deviations.

We define a fixed-size vector where each position corresponds to a unique signal observed during training. At each time step, the vector is filled with available signal values, while missing signals are masked. Consider the set $\mathcal{S} = \{s_1, s_2, \ldots, s_N\}$, which denotes the set of all signals. The global signal vector at time t is defined as:

$$\mathbf{x}_t = \Big[v_t(s_1) \cdot m_t(s_1),\ v_t(s_2) \cdot m_t(s_2),\ \ldots,\ v_t(s_N) \cdot m_t(s_N) \Big],$$

where $v_t(s_i)$ is the value of signal s_i at time t, and $m_t(s_i) \in \{0, 1\}$ is a mask indicating whether s_i is present.

Although global vectors are high-dimensional and sparse, masking and the ability of Transformers to model structured dependencies mitigate these issues. Transformer embeddings built from global vectors exhibit diverse, well-separated latent structures as confirmed by PCA in figure 3.5 . This separation enhances interpretability of results, enabling traceability of individual signals and providing an improved visualization through time series plots or heatmaps.

3.4.2 Input Representation Module

This module prepares the Transformer encoder-decoder input by constructing a unified, position-aware representation for each message sequence. The first layer is an embedding layer, implemented as a lookup table that maps categorical features such as CAN IDs and SPNs into dense vectors of a fixed embedding size (a hyperparameter of the architecture). This transformation prevents the model from misinterpreting raw categorical identifiers as ordinal or continuous values, and at the same time it enables learning meaningful relationships and similarities among different signals within the embedding space.

These embeddings are complemented with numerical and engineered features, including timestamp deltas and payload statistics, providing a richer contextual information.
Let $\mathbf{H}_{\text{cat}} \in \mathbb{R}^{B \times N \times d_{\text{cat}}}$ be the concatenation of categorical embeddings , and
$\mathbf{H}_{\text{cont}} \in \mathbb{R}^{B \times N \times d_{\text{cont}}}$ be the concatenation of all continuous features. Then the combined feature representation is:

$$\mathbf{X}_{\text{cat}} = [\mathbf{H}_{\text{cat}} \,||\, \mathbf{H}_{\text{cont}}] \in \mathbb{R}^{B \times N \times (d_{\text{cat}} + d_{\text{cont}})}$$

This vector is projected into the model's hidden dimension and augmented with positional encodings to capture temporal dependencies. We employ a *sine-cosine positional encoding* similar to Vaswani et al. [89]. For a sequence of length n and embedding dimension d_{model}, the positional encoding vector for position i is defined as:

$$\mathbf{PE}_{i,j} = \begin{cases} \sin\left(\frac{i}{10000^{\frac{2j}{d_{\text{model}}}}}\right), & \text{if } j \text{ is even} \\ \cos\left(\frac{i}{10000^{\frac{2j}{d_{\text{model}}}}}\right), & \text{if } j \text{ is odd} \end{cases}$$
$$i = 1, \ldots, n, \quad j = 0, \ldots, d_{\text{model}} - 1$$

In our implementation, this positional encoding is precomputed as a matrix $\mathbf{PE} \in \mathbb{R}^{1 \times \text{max\_len} \times d_{\text{model}}}$ and added to the input embeddings:

$$\mathbf{X}_{\text{final}} = \mathbf{X} + \mathbf{PE}_{:,1:n,:}$$

3.4.3 Transformer Encoder-Decoder Module

The Transformer encoder–decoder module [89] is the core sequence modeling component of the framework, designed to predict and reconstruct sequences of J1939 CAN messages. It operates on the unified position-aware representations produced by the Input Representation Module, which enable capturing both temporal dependencies and inter-signal relationships in sequences of J1939 CAN messages.

The **Transformer Encoder** consists of a stack of $N = 6$ identical layers. Each layer contains two primary sub-layers: a *multi-head self-attention* mechanism and a *position-wise feed-forward network*. The self-attention sub-layer captures dependencies across the entire input sequence by projecting input vectors into *queries* (Q), *keys* (K), and *values* (V), and computing a scaled dot-product:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \tag{3.7}$$

Multi-head attention allows the model to capture multiple types of dependencies simultaneously. Each head computes its own attention, and the outputs are concatenated and linearly projected:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O, \tag{3.8}$$

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

where each head has its own projection matrices W_i^Q, W_i^K, and W_i^V.

The feed-forward network is applied independently to each position:

$$f(x) = \text{ReLU}(Wx + b) \tag{3.9}$$

Residual connections and layer normalization are applied around each sub-layer to avoid degradation and stabilize training:

$$\text{LayerNorm}(x + \text{Sublayer}(x)) \tag{3.10}$$

All sub-layers, including embedding layers, produce outputs of dimension d_{model}, ensuring consistent representations throughout the encoder stack.

The **Transformer Decoder** mirrors the encoder with $N = 6$ layers, but introduces a third sub-layer in each layer: a *multi-head attention* mechanism over the encoder outputs. This *cross-attention* allows the decoder to condition its predictions on the full context of the input sequence, integrating information learned by the encoder while generating the output sequence. In addition, the decoder employs *masked self-attention*, which prevents access to future positions in the sequence and preserves causality. As in the encoder, residual connections and layer normalization are applied around all sub-layers.

The decoder outputs are projected through a **Linear layer** that maps the hidden representations back into the SPN feature space. This layer acts as the bridge between abstract Transformer embeddings and concrete payload values to be reconstructed.

In our framework, this general encoder–decoder architecture provides a flexible backbone that can later be specialized into temporal- or payload-focused models.

3.4.4 Support Vector Data Description Module

A central challenge in anomaly detection lies in defining a threshold that separates normal from anomalous behavior. In practice, many approaches rely on expert knowledge or subjective rules to set such thresholds, which introduces bias and limits generalizability. Our objective, therefore, was to identify a method that reduces expert intervention and subjectivity while still adapting to variations in the data distribution. Specifically, we asked whether thresholds should be fixed heuristically, estimated statistically, or learned directly from the

data in a self-adaptive manner.

Several approaches offer different solutions. Traditional works often rely on *heuristic multi-thresholds*, where upper and lower bounds are set based on statistical rules such as mean and standard deviation [108]. While simple, such thresholds remain static and may fail to adapt when the data distribution shifts.

A more advanced alternative is the *Peak Over Threshold (POT)* method, which models extreme values using a Generalized Pareto Distribution (GPD) and sets a dynamic cutoff [90]. POT improves flexibility compared to fixed rules, but it still requires a predefined initial threshold to determine which values are considered extreme, introducing dependency on prior assumptions.

In contrast, *Support Vector Data Description (SVDD)* [109] defines the threshold implicitly by learning a minimal hypersphere that encloses the normal data in a latent space. The decision boundary, given by the radius of the hypersphere, is derived directly from the training distribution, without requiring manual thresholding or prior tuning of extremes. This property makes SVDD inherently adaptive to varying payload dynamics. Prior work in industrial anomaly detection has successfully applied SVDD to Transformer embeddings, highlighting its effectiveness for capturing compact data boundaries in high-dimensional spaces [110].

For these reasons, we adopt SVDD as our thresholding strategy. Unlike heuristic or POT-based methods, SVDD provides a self-adjusting and data-driven decision boundary. This ensures that anomalies are detected relative to the true distribution of normal data, reducing reliance on human interventions and improving robustness to distributional shifts in J1939 CAN traffic.

3.4.5 Temporal Anomaly Detection Module: *TemporalSeq*

Leveraging the Transformer is central to our architecture, as its self-attention mechanism provides capabilities that are crucial for modeling J1939 CAN traffic. Unlike recurrent models [76–82], the self-attention mechanism can capture short-term fluctuations (such as irregular timing intervals) as well as long-term dependencies (such as periodicities or rare patterns in message sequences). This dual capability aligns perfectly with the nature of J1939 traffic, where normal behavior combines predictable regularity with bursty, irregular events.

In our Transformer-based sequence-to-sequence architecture, each input sequence is constructed from raw messages and transformed into structured representations that include CAN identifiers, and message-derived temporal features (e.g., time deltas and derived delta variations). These sequences are split into a source (historical context) and a target (pre-

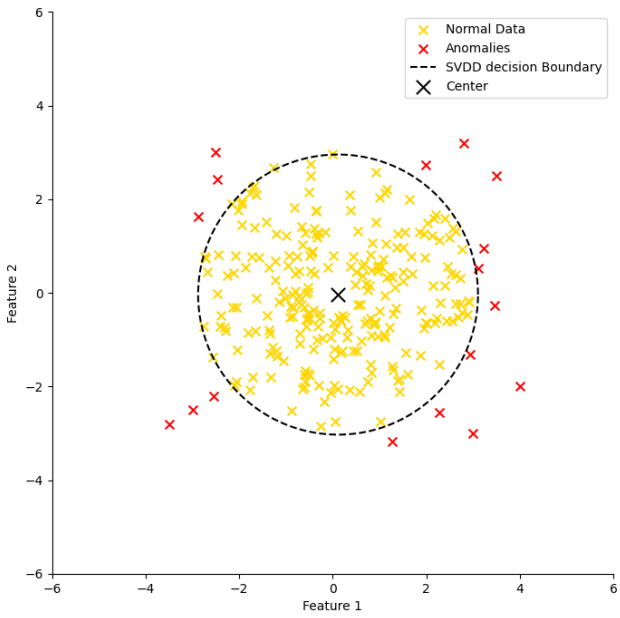


Figure 3.6 SVDD separates normal from anomalous points via a learned hypersphere in the model's latent space, illustrating data-driven thresholding

diction window), allowing the model to learn dependencies between past and future message dynamics.

The transformer encoder maps input message sequences into contextualized latent embeddings using stacked self-attention and feed-forward layers, thereby encoding the temporal evolution of messages conditioned on CAN identifiers. The decoder then autoregressively reconstructs future message features by attending to both its own history and the encoder outputs, enabling predictive modeling of temporal patterns.

During training, we gradually expose the decoder to its own predictions instead of only ground-truth inputs, thereby reducing error accumulation at inference. During validation, the average reconstruction error over predicted sequences is computed to assess the training peformance. At inference, anomaly detection is further reinforced through Support Vector Data Description (SVDD) applied to encoder embeddings of normal data, allowing deviations in the latent temporal representation space to be detected and complementing reconstruction-based evaluation.

Specifically, each predicted sequence is encoded and its embeddings are flattened and scaled using a pre-trained SVDD scaler. The SVDD model then computes distances of these embeddings to the learned hypersphere boundary, producing a per-step anomaly score:
Let the predicted sequence be $\hat{X} \in \mathbb{R}^{B \times T \times F}$, where B is the batch size, T is the sequence

length, and F is the feature dimension. Let the encoder embeddings of the predicted sequence be:

$$Z = \text{Encoder}(\hat{X}) \in \mathbb{R}^{B \times T \times D}.$$

Flatten the embeddings per sequence step:

$$\tilde{Z} = \text{reshape}(Z, B \cdot T, D),$$

and scale with the pre-trained SVDD scaler:

$$\hat{Z} = \text{Scaler}(\tilde{Z}).$$

The SVDD model computes distances of each embedding to the hypersphere boundary:

$$\text{SVDD\_score}_i = f_{\text{SVDD}}(\hat{Z}_i), \quad i = 1, \ldots, B \cdot T,$$

where $f_{\text{SVDD}}(\cdot)$ is the decision function of the trained SVDD. Reshape back to sequence form:

$$\text{SVDD\_score} \in \mathbb{R}^{B \times T}.$$

The predicted sequence is flagged as anomalous if any step is inside the hypersphere:

$$\text{Anomaly}(\hat{X}) = \begin{cases} 1 & \text{if } \min_{i=1,\ldots,T} \text{SVDD\_score}_{b,i} \leq 0, \\ 0 & \text{otherwise,} \end{cases} \quad \forall b = 1, \ldots, B.$$

This formulation ensures that even subtle deviations in the latent temporal embeddings trigger anomaly detection.

A central advantage of this architecture lies in its ability to detect abnormal CAN ID behavior from temporal characteristics, without introducing noise from the diversity of payload content. While the sequence as a whole may be flagged as anomalous, the model retains the ability to identify exactly which messages contribute to the anomaly. This is facilitated by SVDD scoring at the embedding level.

As a result, experts can pinpoint the precise CAN IDs exhibiting abnormal behavior, providing actionable insights in an interpretable and human-readable manner.

3.4.6 Payload Anomaly Detection Module: *PayloadPattern*

Complementing the temporal modeling capabilities, we propose a Transformer-based autoencoder designed for reconstructing J1939 CAN payloads and detecting anomalies at a fine-grained, signal-specific level. Each input sequence is composed of SPN values, categorical cluster identifiers, and binary masks indicating which signals are active.

The encoder captures interdependencies among signals through stacked self-attention and feed-forward layers, generating contextualized representations of the input sequence. The transformer decoder additionally incorporates cross-attention to the encoder outputs, allowing its reconstructions to be conditioned on the global context. The decoder hidden states are then projected back into the SPN feature space via a linear layer, bridging the abstract embedding space with concrete payload values.

Training is performed using a masked loss function, applied selectively to active SPNs, ensuring inactive signals do not influence optimization while providing robustness to outliers and irregular payload dynamics. During validation, reconstruction errors are aggregated per message and compared to a dynamic threshold to detect anomalies. At inference, Support Vector Data Description (SVDD) is applied to encoder representations. To detect subtle attacks, we consider the minimum SVDD score raised by any signal in a message:

$$\text{Score}_{\text{msg}} = \min_{i \in \{1,\ldots,N_{\text{SPN}}\}} \text{Score}_i,$$

where Score_i denotes the distance of the i-th SPN embedding to the learned hypersphere boundary. A message is flagged as anomalous if $\text{Score}_{\text{msg}}$ falls below zero. This approach ensures that even minor deviations at the signal level trigger anomaly detection, improving sensitivity to fine-grained payload anomalies.

A key feature of this design is its focus on detecting unusual behavior directly at the payload level. Instead of flagging only entire messages or sequences as anomalous, the model reconstructs each SPN individually, allowing it to pinpoint exactly which signals behave abnormally. Along with the feedback from the temporal model, this actionable information helps security experts understand the root cause of an alert, making it easier to investigate, assess risks, and respond to potential threats in J1939 CAN traffic.

3.4.7 Decision Fusion Layer

The temporal and payload modules individually capture distinct dimensions of J1939 CAN traffic. Each detector has strengths but also blind spots: temporal modeling excels at capturing abnormal timing or sequence-level irregularities but deliberately ignores payload diversity, while payload reconstruction pinpoints abnormal signal behaviors but is less sensitive to structural or timing-related deviations. Given these complementary strengths and weaknesses, their outputs must ultimately be reconciled into a single, operationally relevant anomaly decision. This design choice is grounded in the principle that **multi-view fusion increases robustness, reduces false positives, and enhances operational interpretability**.

Inputs

The layer receives as input the anomaly indicators produced by the two modules. Depending on the configuration, these may be raw scores (e.g., reconstruction errors or SVDD distances) or binary flags (normal/abnormal). Importantly, the layer is agnostic to the form of the inputs, as its role is not to generate new features but to arbitrate among the decisions already produced.

Arbitration Mechanism

At the heart of the module lies an arbitration process that governs how the final decision is reached. The rationale is that anomaly detection in safety-critical domains such as J1939 cannot be treated as a "one-size-fits-all" problem. Different operational contexts (e.g., diagnostics, security monitoring, or forensic analysis) demand different trade-offs between sensitivity and specificity. For this reason, the fusion layer is deliberately **configurable by domain experts** rather than locked into a rigid rule.

The framework supports multiple arbitration strategies, each suitable for some particular anomaly detection scenarios:

- **Strict Arbitration (OR-rule).** A message is flagged as anomalous if *any* module raises an alert. This maximizes sensitivity, ensuring that subtle attacks caught by only one detector are not missed.

$$D_{\text{OR}} = d_t \vee d_p \tag{3.11}$$

- **Weighted Fusion** Experts may assign weights to the detectors, producing a weighted vote or score aggregation. This allows the system to privilege the detector empirically shown to be more reliable for a given deployment.

Let $s_t, s_p \in [0,1]$ denote normalized anomaly scores from each module and $w_t, w_p \geq 0$ their respective weights. The fused score is:

$$S_{\text{fused}} = \frac{w_t s_t + w_p s_p}{w_t + w_p} \tag{3.12}$$

A message is flagged as anomalous if $S_{\text{fused}} \geq \tau$, where τ is a user-defined threshold.

- **Temporal Alert Correlation** The module can also enforce escalation when anomalies occur in succession across detectors. If one model flags a message and the other flags the next, the second alert is promoted to abnormal. The rationale is that temporally correlated alerts across distinct views of the traffic are unlikely to be spurious. This captures anomalies that manifest as distributed patterns across different dimensions of traffic, making this a lightweight yet effective fusion mechanism.

  Let $d_t^{(i)}$ and $d_p^{(i)}$ denote the current message, and $d_t^{(i-1)}, d_p^{(i-1)}$ the previous message's decisions. The sequentially consistent decision is:

$$D_{\text{seq}}^{(i)} = \begin{cases} 1, & \text{if } (d_t^{(i)} \wedge d_p^{(i-1)}) \vee (d_p^{(i)} \wedge d_t^{(i-1)}) \\ d_t^{(i)} \vee d_p^{(i)}, & \text{otherwise} \end{cases} \tag{3.13}$$

- **Graduated Decisions.** The layer can output a third state (*yellow flag*) when only one module indicates an anomaly. This intermediate state supports practical operations, where not all alerts need immediate escalation but should still be monitored.

  Let $D_{\text{grad}} \in \{0, 0.5, 1\}$ be a ternary decision where 0 indicates normal, 0.5 indicates a yellow flag, and 1 indicates anomalous:

$$D_{\text{grad}} = \begin{cases} 1 & \text{if } d_t = 1 \text{ and } d_p = 1 \\ 0.5 & \text{if } d_t \neq d_p \\ 0 & \text{if } d_t = 0 \text{ and } d_p = 0 \end{cases} \tag{3.14}$$

From a methodological standpoint, the Decision Fusion Layer embodies *ensemble-based anomaly detection*. Rather than assuming a single model can capture the full complexity of J1939 traffic, it acknowledges that heterogeneity in detectors is an asset. By providing configurable arbitration strategies, it enables practitioners to tune the balance between *false positive control* and *attack sensitivity* according to the deployment scenario. The outcome is a unified

anomaly flag that is not only technically rigorous but also adaptable to real-world deployment requirements and transformed into a trustworthy decision that can be adapted to diverse use cases.

3.4.8 Interpretability and Expert Feedback

To mitigate the black-box nature of deep learning–based anomaly detectors, the proposed framework integrates a multi-level interpretability mechanism that functions as a diagnostic agent, enabling experts to probe and understand model behavior across progressively abstract representations (as illustrated in Fig.3.2). Each observation point, depicted by the expert icons, marks a specific level of interpretive access to the system's internal reasoning (feature clusters, message-level reconstructions, or contextual anomaly scores). At every stage, the framework selectively logs the Transformer's outputs, and these logs are then fed into an operational LLM [111] that converts the model's internal diagnostics into clear, human-readable explanations (Representative log excerpts and example outputs are provided in Appendix B). This ensures that system insights remain transparent and actionable, with the LLM providing model-agnostic, role-aware explanations tailored to both AI and non-AI experts (Figure 3.7).

At the feature level (1), the diagnostic agent engages with temporal and payload-based descriptors prior to training. Through HDBSCAN clustering, the framework groups CAN signals by statistical similarity, uncovering unregistered or irregular SPNs, unexpected payload correlations, latent dependencies among PGNs, and previously unseen CAN IDs (see Appendix C for illustrative examples). This stage produces a navigable "map" of the input space that allows the expert to validate data integrity and refine preprocessing assumptions. For instance, the output of the operational LLM component at this level would be:

"Cluster 7 groups SPN_190 (engine speed) and SPN_183 (fuel rate) based on synchronized variance/entropy, and includes signals transmitted via a previously unseen CAN ID 0x18FEEE00, indicating both normal and emerging system behavior"

For the automotive experts the LLM would elaborate:
"The signal corresponds to a newly activated engine ECU module that should be verified for correct operation"

At the message level (2), interpretability is achieved through SVDD distances and reconstruction losses derived from Transformer encoder–decoder representations. Here, the diagnostic agent observes how each message is reconstructed, identifying which SPNs contribute most

to deviations. A message whose predicted payload deviates strongly in torque or temperature components, combined with a large SVDD distance, signals a localized anomaly within the latent embedding space. In fact, the Transformer behaves like an expert auditor, decoding its own latent space to expose how normality is organized, where anomalies separate, and how far each message lies from the learned manifold of expected behavior. The operational LLM output would look like:

"SPN_190 (engine speed) is driving the anomaly. Its reconstruction error is 3.4× above baseline, and the SVDD distance is outside the normal envelope. This indicates a concrete deviation in system behavior, not a noise artifact"

For cybersecurity experts leveraging available threat intelligence, the insight would be:

"Irregular engine speed readings from the Engine ECU (0x00) suggest a possible speed-spoofing or DoS attack that could cause false dashboard measurements and should be immediately investigated"

At the contextual level (3), information from clustering, reconstruction, and SVDD evaluation is consolidated into a decision-fusion layer. This layer aggregates anomaly indicators into interpretable dashboards, where experts can visualize operational context, trace error origins, and iteratively adjust decision thresholds. For example, when the agent detects recurrent deviations in a specific cluster, it flags that operational mode for further inspection, linking quantitative scores to semantic diagnostic reasoning. The LLM will report the following information:

"Cluster 7 shows recurring anomaly spikes tied to consistent timing–payload mismatches. This pattern signals a repeatable operational issue and warrants direct investigation of the subsystem mapped to this cluster"

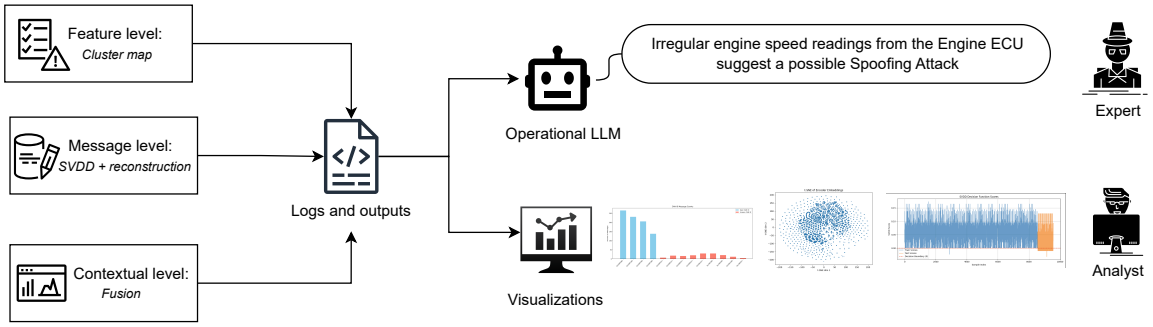


Figure 3.7 Multi-level interpretability pipeline. Outputs include LLM-generated textual explanations and visual analytics tailored for expert and analyst review

By coupling algorithmic reasoning with human interpretive feedback, the framework trans-

forms deep anomaly detection into a cooperative diagnostic process, one where the Transformer not only detects irregularities but also explains the rationale behind each decision. This synergy bridges the gap between machine inference and expert domain knowledge, effectively demystifying the model's internal logic without sacrificing predictive power.

3.5 Performance Evaluation

In this section, we present a comprehensive evaluation of the proposed anomaly detection framework. This section highlights the experimental setup, performance metrics, and key observations compared to other approaches, emphasizing how each model and their complementary fusion contribute to detecting anomalies in J1939 CAN bus traffic.

3.5.1 Dataset

The benign data was collected from an International ProStar civilian truck presented in Figure 3.8 during standard operation . The vehicle was not equipped with a trailer during testing to simplify data collection and analysis. Data was captured through the vehicle's J1939 connector located under the steering wheel, allowing direct access to the CAN bus. The dataset spans 5 hours of normal operation, including idle periods, engine start-up, and shutdown sequences. To ensure robust evaluation, only a subset corresponding to 1.5 hours of active vehicle operation was used, excluding low-occurrence transitionary states (e.g., power-on/off), which were found to negatively impact model testing.

Figure 3.8 International Prostar

The dataset contains timestamped J1939 messages, including CAN identifiers and 8-byte

payloads, representing a broad spectrum of operational states, more details were provided in section 3.4.1. While vehicle-specific ICDs were not available for this platform, the dataset still provides sufficient variability across different operating conditions for evaluating temporal- and payload-based anomaly detection. The logs were carefully curated to ensure coverage of diverse vehicle behaviors, allowing the models to learn both typical message patterns and rare variations that are critical for effective anomaly detection in safety-critical vehicular systems.

The normal dataset contains approximately 114 distinct CAN IDs and 695 signals, decoded using an adjusted DBC file to align signal boundaries and scaling, covering roughly 20% of all signals defined in the vehicle's DBC file while including critical functions such as engine, braking, cruise control, and transmission systems.

Attacks were produced using a modular set of scripts that automates anomalous log generation. Depending on the attack type, logs were either recorded on-vehicle or synthesized by modifying baseline captures. As vehicle access was limited, we relied primarily on controlled replay for repeatable, instrumented testing. Each attack variant and its associated metadata were recorded for every experiment. The selected scenarios span orthogonal adversary objectives, including arbitration exploitation, congestion, temporal evasion, payload authenticity, and identity impersonation enabling us to characterize detector behavior across distinct threat axes.

Guided by the adversary capabilities outlined in Section 3.3, we instantiated a representative set of attack scenarios to operationalize these threat categories under controlled conditions. The key variants and their purpose in our evaluation are summarized as follows:

- **Max-priority DoS**: Sustained floods were injected at the highest-priority ID to to demonstrate worst-case arbitration exploitation.
- **Random / Burst DoS**: Stochastic, bursty traffic was injected to emulate realistic congestion and false-positive behavior.
- **Known CAN-ID DoS**: Different legitimate ECUs identifier were flooded at abnormal rates to test whether the detector can identify anomalous throughput from an valid source.
- **Covert-channel (LSB payload)**: A hidden bitstream was embedded into the least-significant bits of selected non-critical signals (fixed offset, fixed chunk width) across successive frames to evaluate detection of low-rate, payload-level exfiltration while preserving pay-

load plausibility [2].

- **Replay attack**: Recorded valid frames were re-injected out-of-context to test whether the detector flags authentic-looking payloads used in improper temporal contexts.
- **CAN-ID spoofing**: Messages with legitimate CAN IDs were crafted to impersonate ECUs and assess the detector's ability to identify origin inconsistencies.
- **Periodic Spoofing**: Timed injections aligned with legitimate periodic traffic were used to evaluate temporal evasion and false-negative risk.

These scenarios comprehensively cover the foundational security pillars, aligning with confidentiality-, integrity-, and availability-driven threat categories: covert-channel manipulation exercises confidentiality risks, replay and spoofing families probe integrity violations, and the various DoS profiles stress availability degradation.

3.5.2 Experiment setup

Model training and evaluation were executed on a GPU-accelerated PyTorch stack with experiment logging via TensorBoard. The compute node used an NVIDIA Titan GPU with CUDA 12.8 (driver/runtime visible via nvidia-smi 570.86.15). Training, validation, and inference pipelines ran with PyTorch's native GPU APIs to leverage device memory and parallelism; checkpoints and model artifacts were persisted to disk and TensorBoard was used to record losses, metrics, learning-rate schedules, and embedding visualizations for offline analysis. GPU utilization and thermal/power state were monitored with nvidia-smi during long runs to ensure stable and repeatable experiments. This configuration provided a production-grade, reproducible environment for benchmarking model performance and conducting the controlled replay experiments described above.

3.5.3 Evaluation Metrics

To evaluate the performance of our approach, we rely on standard classification metrics suited for imbalanced datasets: **Precision**, **Recall**, **F1-Score**, and **False Negative Rate (FNR)**.

Because anomalies are rare, the **F1-score** serves as the main indicator of model effectiveness. It reflects the balance between **Precision** (accuracy of detected anomalies) and **Recall**

[2]Signal selection, endianness and DBC encoding were respected and bit-rate was kept low to preserve plausibility and avoid disrupting normal ECU functions.

(coverage of actual anomalies). A higher F1-score, closer to 1.0, indicates stronger detection capability with fewer false alarms and missed events.

$$Precision = \frac{TP}{TP + FP} \tag{3.15}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.16}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{3.17}$$

The **False Negative Rate (FNR)** is also critical, as undetected anomalies can directly compromise safety and reliability:

$$FNR = \frac{FN}{TP + FN} \tag{3.18}$$

To assess threshold sensitivity, using the False Positive Rate (FPR), defined as:

$$FPR = \frac{FP}{FP + TN} \tag{3.19}$$

During tests on datasets containing only normal traffic, **Accuracy** is used instead of the F1-score, as no anomalies are present:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.20}$$

Where **TP** = True Positive, **FP** = False Positive, **TN** = True Negative, and **FN** = False Negative.

All metrics are computed at the message level, rather than at the broader attack level. Aggregating performance over continuous windows spanning an attack could inflate metrics, as detecting that some anomaly occurred is easier than pinpointing the exact messages involved. Evaluating at the message level provides a stricter, more realistic measure of detection accuracy and robustness. Moreover, this approach enhances traceability, allowing anomalous messages to be directly traced to specific attack components, which facilitates root-cause analysis and supports fine-grained evaluation of the model's response to diverse threat patterns.

3.5.4 Training setup

To ensure the framework closely mimics real-world functionality and remains reproducible in an industrial environment, the training procedure was carefully designed. Initially, temporal pauses in data collection were addressed by separating idle or faulty periods from active sequences, preventing misleading patterns in the temporal model. Subsequently, cross-validation was employed to select a model that generalizes effectively across diverse operating conditions. As illustrated in Figure 3.2, scheduled sampling was applied in the decoder so the model could learn from its own predictions while still leveraging ground truth, aligning training with the autoregressive decoding used during testing to replicate realistic message flows. As a result, the Smooth L1 loss function served both as the training objective and as a stopping criterion, ensuring robust convergence and providing a fallback if SVDD thresholding fails. Ultimately, grid search was used to optimize hyperparameters, identifying the best combination for this setup and perfecting the training procedure.

3.5.5 Evaluation Results

To ensure our models accurately learn normal behavior and to assess the robustness of our proposed anomaly detection framework in identifying unseen but normal operating conditions, we evaluated the framework models on two datasets. The first is our own dataset described in section 3.5.1, collected from an International ProStar. The second is a recently published dataset collected from a Renault Euro VI heavy-duty truck (Renault T520 6×2) [112]. This dataset comprises over 180 hours of CAN bus traffic, recorded from multiple driving sessions with different drivers and under diverse traffic conditions (urban and rural). All messages conform to the SAE J1939 standard and were stored as CSV files containing timestamps, CAN identifiers, data length, and payloads, each holding 50,000 frames (≈1 minute of recording).

Performance was compared across three configurations, each representing one of the proposed modules: TemporalSeq, PayloadPattern and Fusion. All models correctly identified the unseen normal data from the Renault dataset, demonstrating strong generalization across vehicle types and driving conditions. As shown in Figure 3.9, the TemporalSeq model exhibited a slightly higher false positive rate (FPR = 0.0132) compared to PayloadPattern (FPR = 0.001). This increase is attributed to session structures inherent to the Renault dataset, in contrast to the ProStar civilian truck dataset, where temporal context and latency distributions were explicitly modeled, resulting in a near-zero FPR. The Fusion model also shows a minor FPR increase on the Renault dataset, indicating residual sensitivity to timing variations. However, its overall accuracy (0.9998) remains effectively perfect.

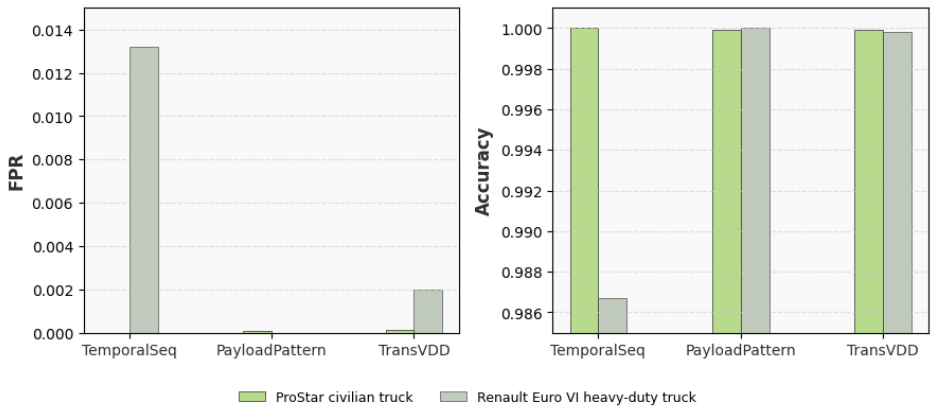
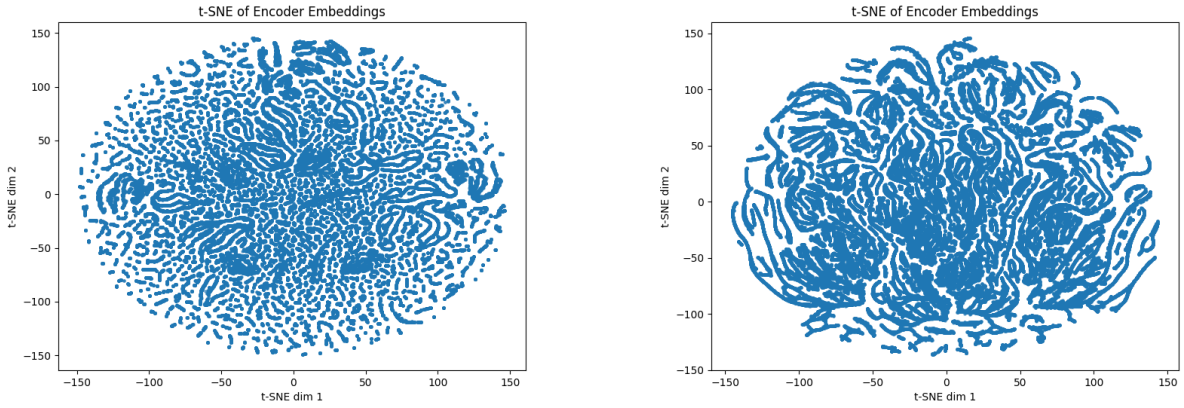


Figure 3.9 Cross-dataset evaluation of detection models on ProStar civilian and Renault Euro VI heavy-duty trucks.

To further support these findings, Figure 3.10a presents a t-SNE projection of encoder embeddings from the Renault dataset. The clustering pattern is visually similar to that observed in the ProStar dataset (Figure 3.10b), with dense, swirling structures that reflect consistent latent organization of normal CAN traffic. This similarity reinforces the model's ability to generalize across vehicle types and driving conditions, and supports the architecture's goal of learning a compact manifold of nominal behavior.



(a) Encoder embeddings from ProStar dataset (b) Encoder embeddings from Renault dataset

Figure 3.10 Comparison of t-SNE encoder embeddings across ProStar and Renault datasets.

Prior literature such as such as [20], [21], [22] typically reports results on canonical benchmark datasets with fixed post-processing (often hand-tuned static thresholds), which reduces comparability when datasets or preprocessing differ (Table 3.1). Because the original baseline

code and raw traces were unavailable, direct reproduction of these prior methods was not possible.

Approach	**Threshold**
CAN-BERT [22]	Not mentioned
IVN [25]	Static range (0.3–0.6)
IVNSL [21]	ε-based static cutoff
CAN-former [20]	Static scalar (3)
TranAD [90]	Dynamic POT + root cause
TransVDD	**Dynamic (SVDD + loss) + root cause**

Table 3.1 Comparison of thresholding strategies across representative CAN bus anomaly detection methods

To establish fair and interpretable baselines, we implemented reproducible, minimal proxy baselines on our Prostar dataset to ensure an apples-to-apples evaluation.

Specifically, we reimplemented (1) an IsolationForest (IForest) [113] as a widely adopted unsupervised statistical baseline , (2) an LSTM autoencoder [75] (LSTM-AE) as a standard sequence deep learning baseline, (3) an Transformer autoencoder (Transformer-AE) and (4) a TranAD-style transformer that follows the TranAD [90] design principles. We trained and evaluated all methods on the same raw J1939 windows, using aligned preprocessing, attack injections, and evaluation metrics.

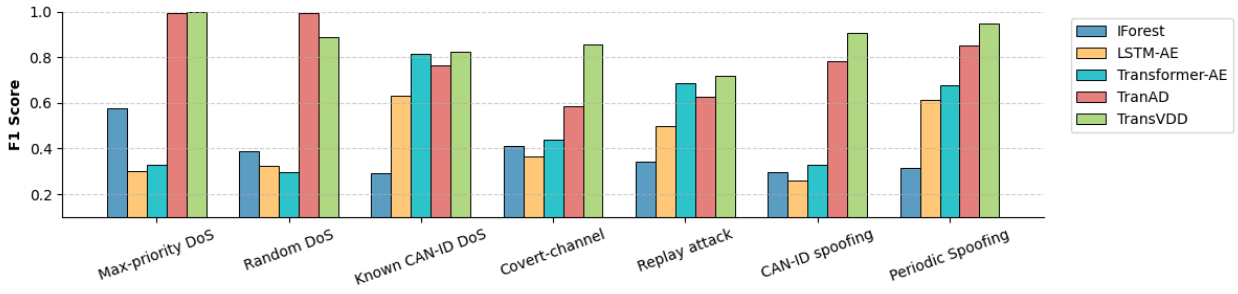


Figure 3.11 F1 Score comparison of anomaly detection models across seven CAN bus attack types. TransVDD achieves the highest F1 score in most attacks

TransVDD achieves superior overall detection performance compared with all evaluated baselines. Figure 3.11 presents the quantitative F1-score results, while Table 3.2 provides a detailed comparison across the four key metrics: TNR, TPR, FNR, and FPR. Across all examined CAN-Bus attack scenarios, the model demonstrates highly stable behavior, consistently attaining elevated TNR and TPR values while keeping FPR and FNR at minimal levels.

Table 3.2 Detection Performance per Attack Type (TNR / TPR / FNR / FPR) on the ProStar dataset

Model	Covert channel				Replay attack				CANID Spoofing				Periodic spoofing			
	TNR	TPR	FNR	FPR	TNR	TPR	FNR	FPR	TNR	TPR	FNR	FPR	TNR	TPR	FNR	FPR
Iforest	0.7179	0.3312	0.6687	0.2820	0.6940	0.2922	0.7078	0.3059	0.7092	0.3280	0.6719	0.2907	0.6968	0.2940	0.7059	0.3031
LSTM-AE	0.7753	0.2738	0.7261	0.2246	0.7831	0.4253	0.5746	0.2168	0.7602	0.2561	0.7438	0.2397	0.7768	0.6311	0.3688	0.2231
Transformer-AE	0.7490	0.3521	0.6478	0.2509	0.6265	0.7846	0.2153	0.3734	0.7288	0.3572	0.6427	0.2711	0.6584	0.8514	0.1485	0.3415
TranAD	1.0000	0.4123	0.5877	0.0000	0.9995	0.4584	0.5416	0.0005	1.0000	0.6403	0.3597	0.0000	0.9951	0.8180	0.1820	0.0049
Our Model	1.0000	0.7250	0.2749	0.0000	1.0000	0.6650	0.3350	0.0000	1.0000	0.8330	0.1670	0.0000	1.0000	0.8890	0.1110	0.0000

Model	Max-priority DoS				Random DoS				Known CAN-ID DoS			
	TNR	TPR	FNR	FPR	TNR	TPR	FNR	FPR	TNR	TPR	FNR	FPR
IForest	0.9644	0.4196	0.5803	0.0355	0.6739	0.2852	0.7147	0.3260	0.7848	0.1914	0.8085	0.2151
LSTM-AE	0.7564	0.2015	0.7984	0.2435	0.7483	0.2228	0.7771	0.2516	0.9028	0.4842	0.5157	0.0971
Transformer-AE	0.7667	0.2230	0.7769	0.2332	0.6452	0.2084	0.7915	0.3547	0.8736	0.7373	0.2626	0.1263
TranAD	0.9963	1.0000	0.0000	0.0037	0.9986	1.0000	0.0000	0.0014	0.9889	0.7611	0.2389	0.0111
Our Model	1.0000	1.0000	0.0000	0.0000	0.9000	0.9995	0.0005	0.0900	0.8630	1.0000	0.0000	0.1370

When exposed to the most subtle attack patterns (Covert channel, Replay attack, CANID Spoofing, Periodic spoofing), our framework consistently outperformed the competing models, delivering strong F1 scores in the 0.8560–0.9460 range and a perfect TNR. This validates that the architecture avoids overfitting to benign traffic despite being trained exclusively on normal sequences.

Among the baselines, the competitor approaches were the Transformer-AE and TranAD. Both lean heavily on reconstruction fidelity, which gives them some traction in identifying payload irregularities, a capability that our system also incorporates. However, their temporal dynamics are largely suppressed because the payload dominates the representation. In contrast, our design maintains a balanced treatment of temporal evolution and payload structure, ensuring neither signal overwhelms the other.

This imbalance in the baselines (including (1) and (2)) produces a noisy separation between timing and content, limiting the anomaly visibility and constraining their detection capability. As a result, all baselines continue to miss anomalies at rates between 0.1820 and 0.7438. Our model also encounters misses, but it secures the strongest TPR overall, ranging from 0.6650 to 0.8890. Performance is expected to further improve with a more flexible decision function.

Our model succeeds in detecting DoS attacks across varying levels of subtlety, achieving significantly high detection performance with recall (TPR) ranging from 0.9995 to 1.0 and F1-scores between 0.82 and 1.0. Competing baselines, including Iforest and LSTM-AE, fall substantially behind, missing up to 75% of DoS frames and reaching 0.7147–0.8000 FNR. These models fail even on the Max-Priority DoS scenario, whereas our approach detects it with perfect accuracy and zero false alarms. This behavior stems from the extreme impact of

Max-Priority DoS on arbitration, message frequency, and temporal structure. These deviations lie far outside the learned manifold, and was easily detected by our dynamic threshold.

The only competitive baseline in this category is TranAD, which achieves a TPR in the range of 0.7677–1.0, but exhibits a higher FNR (up to 0.2389) on Known CAN-ID injection. In this scenario, TransVDD also reports a higher FPR due to the attack's subtlety and the tight SVDD boundaries surrounding the latent space. These strict hypersphere constraints cause certain known CAN-ID sequences, whose payload content appears normal (passed the PayloadPattern model), to be flagged anomalous due to timing perturbations introduced by the attack (flagged by the TemporelSeq model). Even with these slight increases in FPR, our model still achieves the highest F1-score (0.8220) on Known-ID DoS, outperforming all baselines, which remain substantially lower.

A similar effect appears in Random DoS, where randomly injected CAN-IDs occasionally intersect with legitimate periodic patterns. Because our architecture jointly penalizes payload and timing deviations, the model raises alerts whenever the injected bursts distort temporal continuity.

This moderate FPR during DoS scenarios acts as an early warning signal, reflecting the network's gradual deviation from nominal timing and payload distributions as the attack begins to saturate the bus. This proactive sensitivity is desirable in high-frequency automotive networks, where early-stage detection significantly reduces operational risk.

3.6 Discussion and limitations

The observed performance of our framework highlights several key insights about anomaly detection in J1939 CAN bus systems. Its unique architecture, which treats temporal sequences and payload content with equal emphasis, is critical for robust detection. By separating these aspects, the Transformer can exploit its sequential modeling power without being dominated by payload noise, resulting in consistently stable detection patterns. This balance explains why subtle attacks, often missed by baselines, are captured effectively: temporal deviations that are imperceptible in isolation become detectable when considered alongside payload patterns. Notably, the reported metrics provide a realistic assessment of detection performance, thanks to message-level granularity and non-aggregated evaluation. This fine-grained analysis ensures that even minor anomalies are accurately measured, without averaging effects inflating results. The comparative struggle of Isolation Forest, LSTM-AE, and Transformer-AE with max-priority DoS attacks underscores a common limitation, they either ignore temporal structure or overemphasize reconstruction of payloads, flattening crit-

ical timing information. Our integration of SVDD, clustering and global vector addresses this by creating a compact latent representation that differentiates both extreme anomalies and emerging or previously unseen CAN IDs. This approach not only improves TPR and TNR but also provides interpretability, revealing which signals or clusters drive anomaly detection. However, performance is not yet perfect: the strict SVDD decision function in figure 3.12, trained solely on normal behavior and designed to prevent any human adjustment, leaves some normal points outside the learned envelope, introducing slight noise in anomaly scoring. This conservative thresholding preserves reliability but may lead to occasional false negatives for sparse or borderline anomalies. Moreover, due to computational limits, the SVDD module was trained on a reduced subset of normal traffic, restricting its exposure to the full variability of benign patterns. While this setup maintains robustness, expanding the normal training pool would likely tighten the SVDD boundary and further reduce scattered false positives.

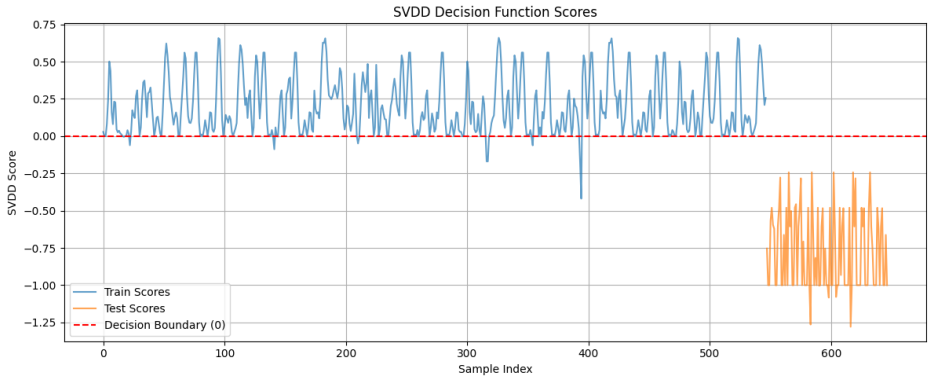


Figure 3.12 SVDD scores for training and testing samples. The decision boundary, trained on restricted normal data, separates most anomalies but excludes some benign points, reflecting cautious thresholding

Beyond detection metrics, the framework distinguishes itself through its strong focus on interpretability and expert insight, as exemplified in Figure 3.13. Its multi-level interpretability enables engineers to identify not only which messages are anomalous but also why these anomalies occur, enhancing real-world diagnostics and system monitoring. In contrast, most existing approaches prioritize raw performance and fail to reveal the reasoning behind their predictions, leaving occasional failures unexplained. TransVDD explicitly bridges this gap.

Given the architectural complexity of J1939 compared to simpler CAN bus protocols, the model's strong performance indicates it can be readily generalized and adapted across a wide range of CAN bus systems.

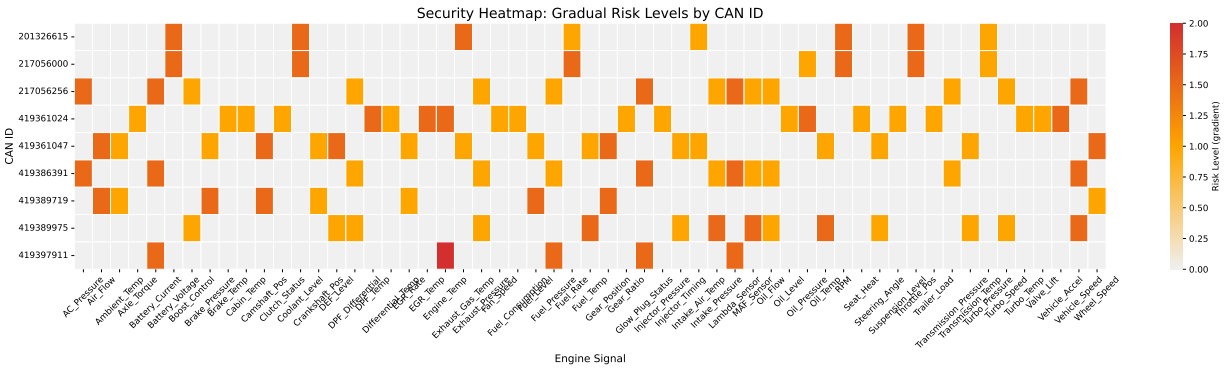


Figure 3.13 Layered risk interpretation from TransVDD, showing the relative diagnostic relevance of engine signals across different CAN IDs. Darker colors indicate higher risk contribution for a given signal-CAN ID pair

3.7 Conclusion and Future work

Detecting anomalies in complex in-vehicle networks remains a critical challenge, particularly for protocols like SAE J1939, where subtle deviations can compromise safety and operational reliability. This work introduces a Transformer-Powered, Interpretable Anomaly Detection Framework that learns normal J1939 behavior end-to-end and flags deviations without predefined attack signatures or labeled anomalies. The architecture establishes a clear division of labor: the Transformer captures temporal dynamics and structured payload evolution, while SVDD enforces a principled, boundary-driven notion of normality for classification. Despite widespread claims of "explainability," most solutions stop at surface-level visualizations. TransVDD goes further by operationalizing explanations by structuring timing and payload features, exposing latent-space distances, and layering expert-focused reasoning to turn opaque anomaly signals into clear, auditable, and actionable insights. This hybrid architecture delivers strong performance where traditional models either overfit or fail to generalize, improving true-positive rates with near-zero false alarms on real-world datasets.

The next phase will stress-test TransVDD under alternative arbitration mechanisms beyond the current "OR-rule" setup and scale SVDD training with broader normal-operation distributions to strengthen boundary reliability. We will evaluate the reassembly of multi-packet J1939 messages (BAM and RTS/CTS) to assess stability with longer payloads and extend testing across different vehicle types and CAN bus protocols to prove generalizability. In parallel, We aim to showcase the LLM component in greater detail, highlighting its ability to leverage Transformer diagnostics and custom logs and evaluating it under real industry conditions to demonstrate its readiness for integration into production-grade vehicle cybersecurity

workflows.

Acknowledgment

The authors gratefully acknowledge the support and collaboration of CortAIx Labs, Mitacs, and NSERC, whose contributions were essential to launching and sustaining this research.

CHAPTER 4 CONCLUSION

Chapter overview. In this thesis, we addressed the fundamental challenge of detecting open-set intrusions, i.e., unknown and evolving behaviors that fall outside predefined attack classes. This is done by developing an interpretable, multi-modal detection framework. While the methodology is domain-agnostic, it was extensively validated on real-world automotive data, demonstrating its effectiveness within the CAN Bus environment. As a conclusion to this dissertation, we summarize the three principal contributions of this work and outline the future research directions that naturally follow

4.1 Summary of Work

This thesis investigated the challenge of detecting open-set intrusions, i.e., threats that cannot be enumerated, labeled, or anticipated in advance, by moving beyond signature-driven and closed-world detection strategies. The goal was to develop an anomaly-detection framework that learns normal behavior with high fidelity, operates reliably under open-set conditions, and remains sufficiently interpretable to keep human expertise meaningfully involved in validating anomalous events.

Although the framework is broadly applicable, its development and validation were grounded in the automotive CAN Bus. This environment presents a uniquely demanding environment for anomaly detection, marked by heterogeneous message types, bursty and irregular timing patterns, and the inherent impossibility of forecasting the full space of potential future attacks. Such characteristics make it an ideal real-world setting for evaluating open-set learning principles under strict operational constraints.

At the outset, a comprehensive examination of CAN bus intrusion detection solutions was conducted, spanning statistical, rule-based, machine learning, and deep-learning proposals across the literature. This review also included a detailed analysis of benchmarking datasets and the attacks commonly used to evaluate these solutions, revealing notable flaws and gaps in coverage. By comparing these documented attacks to the current detection capabilities, it became evident that the CAN bus remains exposed to a broader threat landscape than what existing methods can reliably detect.

Overall, this assessment highlighted a persistent limitation in existing approaches. While some methods are fundamentally closed-set, others are opaque, with decisions that are not readily interpretable and boundaries that are manually set by human experts. These short-

comings underscore the need for a more principled solution, motivating the development of a modality-aware, representation-aligned open-set detection framework capable of generalizing to unseen behaviors and future attack variants.

The research progressed through several tightly connected stages. It began with a detailed characterization of real-world CAN Bus behavior, revealing that the most informative cues for anomaly detection arise from two distinct modalities: temporal regularities observable in inter-arrival patterns, and payload dynamics that reflect the semantic evolution of message content. This natural decomposition formed the conceptual foundation for a multimodal modeling strategy.

Drawing on these insights, we designed a pair of specialized Transformer-based models, with each module specialized for a distinct modality. The temporal log module processes time deltas and normalized CAN identifiers using a dedicated input representation pipeline that includes categorical feature embeddings and positional encodings tailored to asynchronous vehicular traffic. This input is passed through an encoder-decoder stack, where schedule sampling stabilizes training and auto-regression governs inference. In parallel, the payload pattern module operates on signal-conditioned payload vectors, which are first grouped by CAN ID clusters to reduce modeling complexity and enhance interpretability. These clusters are derived using HDBSCAN, incorporating categorical, behavioral, and technical criteria such as message frequency, payload entropy, and byte-level roles. By clustering messages with similar functional and statistical characteristics, the model can trace anomalies to specific subsystems or message types, improving explainability. The clustered payloads are then processed along with the spn global vector through a Transformer encoder-decoder pipeline for reconstruction-based anomaly detection.

To eliminate subjective or hand-tuned thresholds, the next stage integrated Support Vector Data Description (SVDD) into the inference pipeline. Encoder representations extracted exclusively from normal traffic were used to fit an SVDD hypersphere boundaries aligned with the learned embedding space. During inference, anomaly scores were computed as distances to this hypersphere and then combined with prediction-error metrics, creating a decision mechanism resilient under open-set conditions.

Because different intrusions distort different behavioral cues, the outputs of the modality-specific models were then fused into a unified anomaly-detection decision. This fusion mechanism reduced the false positives often produced by isolated fluctuations in a single modality.

The full framework was validated on a large, real-world j1939 CAN Bus dataset. Training was performed exclusively on normal traffic, while evaluation was conducted under open-set conditions involving unseen anomalies. Experiments assessed deviations in both timing

and payload behavior, measured performance across different scenarios, and compared the results with established baselines. Metrics were calculated at a fine-grained, per-message level, allowing each anomaly to be traced back to the specific message and highlighting which signal contributed to the detected anomaly. The system achieved near-zero false positives while reaching a true-positive rate of up to 0.889, satisfying key operational requirements for in-vehicle intrusion detection systems.

In its final form, this thesis contributes a multimodal Transformer-based architecture that models temporal and semantic behavior separately for enhanced clarity and robustness, an SVDD-driven scoring mechanism that replaces subjective arbitrations with principled, representation-aligned decision boundaries, and a fused anomaly-detection framework validated on real automotive data that delivers reliable, interpretable, and scalable open-set intrusion detection.

4.2 Limitations

While the proposed approach demonstrated efficiency in automotive CAN Bus scenarios, its domain-agnostic capabilities remain unproven, and broader applicability still needs to be validated, particularly given the limited resources employing similar approaches. A further consideration is that this version of the framework is tailored to specific industry requirements: the clustering criteria and arbitration mechanism are designed to reflect expert necessities and must be adjusted for each system based on traffic behavior and operational specifications. Moreover, due to computational limits in our experimental setup, the SVDD module was trained on a reduced subset of normal encoder output embeddings, limiting its exposure to the full variability of benign patterns. Expanding the normal training pool would likely improve performance. Even so, the conservative thresholding may need some additional tolerance to include normal points that remain outside the boundary. This adjustment helps detect borderline anomalies and reduces the likelihood of false negatives. Finally, although metrics are calculated at a fine-grained, per-message level, this relies on the availability of accurate signal mappings (SPNs). Incomplete or incorrect DBCs could reduce interpretability.

4.3 Future Research

As an immediate next step, we will address all the limitations discussed above. This begins with expanding SVDD training, refining thresholding for borderline anomalies, and testing the model under different arbitration mechanisms to evaluate their effects.

We also aim to make the interpretable module's logs and diagnostics more structured and concise, ensuring effective integration with LLMs and other threat intelligence operations. Our approach of bridging the gap between experts and non-experts through interpretability has already found adoption across industry. Notably, it has been considered as part of a large security project led by Thales. To support this process, we are extending the framework to operate seamlessly across the full J1939 protocol stack. We have already addressed the Data Link Layer (CAN Bus – ISO 11898)[1], the Network Layer (J1939 message structure), and the Application Layer (PGNs, SPNs, signals). The remaining step is full support for the Transport Layer, which manages multi-packet messages such as BAM and RTS/CTS transfers. We have already begun developing a multi-message reassembly module capable of handling Transport Protocol frames — TP.CM (Connection Management) and TP.DT (Data Transfer) — as detailed in the Appendix A. This module will serve as an additional interpretable component to reduce false positives through a second verification stage based on the reassembled messages. This will also provide the potential to support longer packet lengths.

Following this development, we will focus on generalizing the framework for broader use in other time-series systems, demonstrating its applicability beyond the automotive domain.

Ultimately, human expertise remains indispensable in validating anomalies. Our goal is to make human intervention as effective and meaningful as possible, ensuring that the system amplifies expert capabilities rather than replacing them.

Humans stay in the loop, we just make the loop smarter.

[1]ISO 11898 CAN bus standard, available at: `https://www.iso.org/standard/86384.html`

REFERENCES

[1] S. Kumar and B. Bhowmik, "Emergence, evolution, and applications of cyber-physical systems in smart society," in *Proceedings of the 2024 Fourth International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*. Bhilai, India: IEEE, 2024.

[2] W. Duo, M. Zhou, and A. Abusorrah, "A survey of cyber attacks on cyber physical systems: Recent advances and future directions," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 5, pp. 784–800, 2022.

[3] Y. A. Farrukh and I. Khan, "Detecting unknown attacks in iot environments: An open set classifier for enhanced network intrusion detection," in *Proceedings of the IEEE Military Communications Conference (MILCOM)*. IEEE, 2023, pp. 1035–1040.

[4] SANS Institute. (2024) What's the scoop on frostygoop? the latest ics malware and ics controls considerations. Accessed: Dec. 7, 2025. [Online]. Available: https://www.sans.org/blog/whats-the-scoop-on-frostygoop-the-latest-ics-malware-and-ics-controls-considerations

[5] D. Antoniuk. (2024) Frostygoop malware left 600 ukrainian households without heat this winter. Accessed: Dec. 7, 2025. [Online]. Available: https://therecord.media/frostygoop-malware-ukraine-heat

[6] S. Corrigan, "Introduction to the controller area network (can)," Texas Instruments, Application Report SLOA101B, 2002, revised May 2016. [Online]. Available: https://www.ti.com/lit/an/sloa101b/sloa101b.pdf?ts=1765374411738

[7] K. Koscher *et al.*, "Experimental security analysis of a modern automobile," in *2010 IEEE Symposium on Security and Privacy*, May 2010, pp. 447–462.

[8] W. Haddock. (2024, Mar.) Can bus vulnerabilities raise threats for cars and tanks. Accessed: Dec. 7, 2025. [Online]. Available: https://www.sealingtech.com/2024/03/05/can-bus-vulnerabilities-raise-threats-for-cars-and-tanks/

[9] S. Sharmin *et al.*, "Benchmarking frameworks and comparative studies of controller area network (can) intrusion detection systems: A review," *Journal of Cybersecurity*, vol. 32, no. 5, 2023.

[10] *Automotive Standards and Recommended Practices*, SAE International, 2024, standards such as SAE J1939, J1979, and related CAN specifications are publicly obtainable through SAE International. Available at: https://www.sae.org.

[11] PlaxidityX, "Can protection – intrusion detection and prevention system for can bus networks," https://plaxidityx.com/products/can-protection/, accessed: Dec. 8, 2025.

[12] E. GmbH, "Escrypt cycurids: Ids for can & ethernet networks," https://www.etas.com/ww/en/products-services/cybersecurity-products/escrypt-cycurids/, accessed: Dec. 8, 2025.

[13] Elektrobit. (2025) Plaxidityx can idps – cybersecurity for can bus networks. Accessed: Dec. 8, 2025. [Online]. Available: https://www.elektrobit.com/products/ecu/eb-tresos/can-idps/

[14] PlaxidityX, "Ids false positives: How alert noise drains oem cybersecurity budgets," https://plaxidityx.com/blog/blog-post/ids-false-positives-how-alert-noise-drains-oem-cybersecurity-budgets, Nov. 2023, accessed: Dec. 8, 2025.

[15] Secure-IC. (2024) How ai strengthens automotive security? the power of securyzr™ ids. Accessed: Dec. 8, 2025. [Online]. Available: https://www.secure-ic.com/blog/ai/how-ai-strengthens-automotive-security-the-power-of-securyzr-ids/

[16] GuardKnox, "Automotive cyber security essentials and ids/ips technology," https://blog.guardknox.com/automotive-cyber-security-essentials-ips-ids-technology, Feb. 2018, accessed: Dec. 8, 2025.

[17] D. Brumley, "Why i'm not sold on machine learning in autonomous security," https://www.csoonline.com/article/567689/why-im-not-sold-on-machine-learning-in-autonomous-security.html, Aug. 2019, accessed: Dec. 8, 2025.

[18] Thales Group, "Automotive cybersecurity solutions," https://www.thalesgroup.com/en/solutions-catalogue/enterprise/automotive/automotive-cybersecurity, 2025, accessed: Dec. 8, 2025.

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

[20] V. Cobilean *et al.*, "Anomaly detection for in-vehicle communication using transformers," in *IECON 2023- 49th Annual Conference of the IEEE Industrial Electronics Society*, Oct. 2023, pp. 1–6.

[21] J. Cao *et al.*, "Anomaly detection for in-vehicle network using self-supervised learning with vehicle-cloud collaboration update," *IEEE Trans. Intell. Transport. Syst.*, vol. 25, no. 7, pp. 7454–7466, 2024.

[22] N. Alkhatib *et al.*, "Can-bert do it? controller area network intrusion detection system based on bert language model," in *2022 IEEE/ACS 19th International Conference on Computer Systems and Applications (AICCSA)*, 2022, pp. 1–8.

[23] R. Rai *et al.*, "Securing the can bus using deep learning for intrusion detection in vehicles," *Scientific Reports*, vol. 15, p. 13820, 2025.

[24] V. Z. Mohale and I. C. Obagbuwa, "Evaluating machine learning-based intrusion detection systems with explainable ai: enhancing transparency and interpretability," *Frontiers in Computer Science*, vol. 7, 2025.

[25] D. Yang *et al.*, "A multivariate time series prediction method for automotive controller area network bus data," *Electronics*, vol. 13, no. 14, p. 2707, Jul. 2024.

[26] J. Backhus *et al.*, "Time series anomaly detection using signal processing and deep learning," *Applied Sciences*, vol. 15, no. 11, p. 6254, 2025.

[27] CSS Electronics, "Can bus explained - a simple intro," https://www.csselectronics.com/pages/can-bus-simple-intro-tutorial, [Online; accessed 2-December-2024].

[28] A. Alfardus and D. B. Rawat, "Evaluation of can bus security vulnerabilities and potential solutions," in *2023 Sixth International Conference of Women in Data Science at Prince Sultan University (WiDS PSU)*. Riyadh, Saudi Arabia: IEEE, 2023, pp. 90–97.

[29] A. D. F. Tron *et al.*, "Canflict: Exploiting peripheral conflicts for data-link layer attacks on automotive networks," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. Los Angeles, CA, USA: ACM, Nov 2022, pp. 711–723.

[30] W. Si, D. Starobinski, and M. Laifenfeld, "Protocol-compliant dos attacks on can: Demonstration and mitigation," in *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*. Montreal, QC, Canada: IEEE, 2016, pp. 1–7.

[31] M. Bozdal *et al.*, "Evaluation of can bus security challenges," *Sensors*, vol. 20, no. 8, p. 2364, Apr 2020.

[32] P. Hank *et al.*, "Automotive ethernet: In-vehicle networking and smart mobility," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. Grenoble, France: IEEE, 2013, pp. 1735–1739.

[33] E. Aliwa *et al.*, "Cyberattacks and countermeasures for in-vehicle networks," *ACM Comput. Surv.*, vol. 54, no. 1, pp. 1–37, Jan 2022.

[34] M. Hanselmann *et al.*, "Canet: An unsupervised intrusion detection system for high dimensional can bus data," *IEEE Access*, vol. 8, pp. 58 194–58 205, 2020.

[35] Z. El-Rewini *et al.*, "Cybersecurity challenges in vehicular communications," *Vehicular Communications*, vol. 23, p. 100214, 2020.

[36] S. Sharmin and H. Mansor, "Intrusion detection on the in-vehicle network using machine learning," in *2021 3rd International Cyber Resilience Conference (CRC)*, Langkawi Island, Malaysia, Jan. 2021, pp. 1–6.

[37] M. Rogers and K. Rasmussen, "Silently disabling ecus and enabling blind attacks on the can bus," arXiv, 2022.

[38] H. Lee, S. H. Jeong, and H. K. Kim, "Otids: A novel intrusion detection system for in-vehicle network by using remote frame," in *2017 15th Annual Conference on Privacy, Security and Trust (PST)*, Calgary, AB, Canada, 2017, pp. 57–5709.

[39] H. M. Song, H. R. Kim, and H. K. Kim, "Intrusion detection system based on the analysis of time intervals of can messages for in-vehicle network," in *2016 International Conference on Information Networking (ICOIN)*, 2016, pp. 63–68.

[40] M. E. Verma *et al.*, "A comprehensive guide to can ids data and introduction of the road dataset," *PLoS ONE*, vol. 19, no. 1, p. e0296879, Jan. 2024.

[41] M. L. Han, B. I. Kwak, and H. K. Kim, "Anomaly intrusion detection method for vehicular networks based on survival analysis," *Vehicular Communications*, vol. 14, pp. 52–63, 2018.

[42] P. Cheng, M. Han, and G. Liu, "Desc-ids: Towards an efficient real-time automotive intrusion detection system based on deep evolving stream clustering," *Future Generation Computer Systems*, vol. 140, pp. 266–281, 2023.

[43] M. R. Ansari *et al.*, "A low-cost masquerade and replay attack detection method for can in automobiles," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2017, pp. 1–4.

[44] C. Miller and C. Valasek, "Can security: Testing and securing vehicle bus communication," Canis Automotive Labs, White Paper, Feb. 2020, [Online]. Available: https://canislabs.com/downloads/2020-02-14-White-Paper-CAN-Security.pdf.

[45] K. Tindell, "The janus attack," Canis Automotive Labs, White Paper, Dec. 2021, [Online]. Available: https://www.can-cia.org/fileadmin/cia/documents/publications/cnlm/december_2021/21-4_p10_the_janus_attack_ken_tindell_canis_automotive_labs.pdf.

[46] A. Boudguiga *et al.*, "A simple intrusion detection method for controller area network," in *2016 IEEE International Conference on Communications (ICC)*, 2016, pp. 1–7.

[47] J. Liu *et al.*, "In-vehicle network attacks and countermeasures: Challenges and future directions," *IEEE Network*, vol. 31, no. 5, pp. 50–58, 2017.

[48] OCSLab, "Can-fd intrusion dataset," OCSLab, 2021, [Online]. Available: https://ocslab.hksecurity.net/Datasets/can-fd-intrusion-dataset. [Accessed: Dec. 20, 2024].

[49] F. Gao *et al.*, "Multi-attack intrusion detection for in-vehicle can-fd messages," *Sensors*, vol. 24, no. 11, p. 3461, May 2024.

[50] M. L. Han, B. I. Kwak, and H. K. Kim, "Anomaly intrusion detection method for vehicular networks based on survival analysis," *Vehicular Communications*, vol. 14, pp. 52–63, 2018.

[51] 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)*. Belfast, UK: IEEE, 2018, pp. 1–6.

[52] J. Khan, D.-W. Lim, and Y.-S. Kim, "Intrusion detection system can-bus in-vehicle networks based on the statistical characteristics of attacks," *Sensors*, vol. 23, no. 7, p. 3554, Mar. 2023.

[53] X. Ming, Z. Wang, and B. Xu, "Lightweight intrusion detection method of vehicle can bus under computational resource constraints," *SCPE*, vol. 24, no. 4, pp. 743–754, Nov. 2023.

[54] N. Khatri, S. Lee, and S. Y. Nam, "Transfer learning-based intrusion detection system for a controller area network," *IEEE Access*, vol. 11, pp. 120 963–120 982, 2023.

[55] A. Anjum *et al.*, "In-vehicle network anomaly detection using extreme gradient boosting machine," in *2022 11th Mediterranean Conference on Embedded Computing (MECO)*. Budva, Montenegro: IEEE, Jun. 2022, pp. 1–6.

[56] X. Wang *et al.*, "Intrusion detection system for in-vehicle can-fd bus id based on gan model," *IEEE Access*, vol. 12, pp. 82 402–82 412, 2024.

[57] E. Levy *et al.*, "Can-loc: Spoofing detection and physical intrusion localization on an in-vehicle can bus based on deep features of voltage signals," *IEEE Trans. Inform. Forensic Secur.*, vol. 18, pp. 4800–4814, 2023.

[58] L. Zhang, X. Yan, and D. Ma, "Efficient and effective in-vehicle intrusion detection system using binarized convolutional neural network," in *IEEE INFOCOM 2024 - IEEE Conference on Computer Communications*. Vancouver, BC, Canada: IEEE, May 2024, pp. 2299–2307.

[59] C. R. Kishore *et al.*, "Intelligent intrusion detection framework for anomaly-based can bus network using bidirectional long short-term memory," *J. Inst. Eng. India Ser. B*, vol. 105, no. 3, pp. 541–564, Jun. 2024.

[60] M. Althunayyan, A. Javed, and O. Rana, "A robust multi-stage intrusion detection system for in-vehicle network security using hierarchical federated learning," *Vehicular Communications*, vol. X, p. Art. no. 100837, 2024.

[61] H. Sun and et al., "Ccid-can: Cross-chain intrusion detection on can bus for autonomous vehicles," *IEEE Internet Things J.*, vol. 11, no. 15, pp. 26 146–26 159, Aug. 2024.

[62] H. Im, D. Lee, and S. Lee, "A novel architecture for an intrusion detection system utilizing cross-check filters for in-vehicle networks," *Sensors*, vol. 24, no. 9, p. 2807, Apr. 2024.

[63] W. Marfo *et al.*, "Detecting masquerade attacks in controller area networks using graph machine learning," *arXiv*, 2024.

[64] X. Li and et al., "Can bus messages abnormal detection using improved svdd in internet of vehicles," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3359–3371, Mar. 2022.

[65] C. Dong, H. Wu, and Q. Li, "Multiple observation hmm-based can bus intrusion detection system for in-vehicle network," *IEEE Access*, vol. 11, pp. 35 639–35 648, 2023.

[66] S. Fang *et al.*, "Windowed hamming distance-based intrusion detection for the can bus," *Applied Sciences*, vol. 14, no. 7, p. 2805, Mar. 2024.

[67] M. Maliha and S. Bhattacharjee, "A unified time series analytics based intrusion detection framework for can bus attacks," in *Proceedings of the Fourteenth ACM Conference on Data and Application Security and Privacy*. Porto, Portugal: ACM, Jun. 2024, pp. 19–30.

[68] W. Lalouani *et al.*, "A robust physical layer ids for intra-vehicle communication security," in *2024 International Conference on Smart Applications, Communications and Networking (SmartNets)*. Harrisonburg, VA, USA: IEEE, May 2024, pp. 1–6.

[69] L. Du *et al.*, "Open world intrusion detection: An open set recognition method for can bus in intelligent connected vehicles," *IEEE Network*, vol. 38, no. 3, pp. 76–82, 2024.

[70] Q. Zhao and et al., "Can bus intrusion detection based on auxiliary classifier gan and out-of-distribution detection," *ACM Trans. Embedded Comput. Syst.*, vol. 21, no. 4, pp. 1–30, Jul. 2022.

[71] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2642–2651.

[72] 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.

[73] H. S. Mavikumbure *et al.*, "Dadae: Domain adversarial autoencoder based in-vehicle can anomaly detection," in *IECON 2023 - 49th Annual Conference of the IEEE Industrial Electronics Society*. Singapore, Singapore: IEEE, Oct. 2023, pp. 1–7.

[74] H. Im, D. Kim, and S. Lee, "Listening to can: Anomaly detection to enhance in-vehicle network security," in *2023 IEEE 13th International Conference on Consumer Electronics - Berlin (ICCE-Berlin)*. Berlin, Germany: IEEE, Sep. 2023, pp. 172–175.

[75] R. Zhao *et al.*, "Application-layer anomaly detection leveraging time-series physical semantics in can-fd vehicle networks," *Electronics*, vol. 13, no. 2, p. 377, Jan. 2024.

[76] F. Gao *et al.*, "Multi-attack intrusion detection for in-vehicle can-fd messages," *Sensors*, vol. 24, no. 11, p. 3461, May 2024.

[77] J. Xiao *et al.*, "Robust anomaly-based intrusion detection system for in-vehicle network by graph neural network framework," *Applied Intelligence*, vol. 53, no. 3, pp. 3183–3206, 2023.

[78] B. Koltai, A. Gazdag, and G. Ács, "Improving can anomaly detection with correlation-based signal clustering," *Infocommunications Journal*, vol. 15, no. 4, pp. 17–25, 2023.

[79] P. Mansourian *et al.*, "Deep learning-based anomaly detection for connected autonomous vehicles using spatiotemporal information," *IEEE Trans. Intell. Transport. Syst.*, vol. 24, no. 12, pp. 16 006–16 017, 2023.

[80] L. Wang and X. Zhang, "Anomaly detection for automated vehicles integrating continuous wavelet transform and convolutional neural network," *Applied Sciences*, vol. 13, no. 9, p. 5525, Apr. 2023.

[81] P. Cheng, M. Han, and G. Liu, "Desc-ids: Towards an efficient real-time automotive intrusion detection system based on deep evolving stream clustering," *Future Generation Computer Systems*, vol. 140, pp. 266–281, 2023.

[82] M. A. Maruf and A. Azim, "Anomaly detection and functional testing for automotive can communication," in *2024 IEEE International Systems Conference (SysCon)*, Apr. 2024, pp. 1–8.

[83] T. P. Nguyen, H. Nam, and D. Kim, "Transformer-based attention network for in-vehicle intrusion detection," *IEEE Access*, vol. 11, pp. 55 389–55 403, 2023.

[84] Texas Instruments, "100base-t1 ethernet: The evolution of automotive networking," https://www.ti.com/lit/wp/szzy009/szzy009.pdf, 2018, [Online; accessed 15-October-2025].

[85] C. Szydlowski, "Can specification 2.0: Protocol and implementations," SAE International, Technical Paper 921603, 1992.

[86] P.-S. Murvay and B. Groza, "Security shortcomings and countermeasures for the sae j1939 commercial vehicle bus protocol," *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 4325–4339, May 2018.

[87] C. Young *et al.*, "Survey of automotive controller area network intrusion-detection systems," *IEEE Design & Test*, vol. 36, no. 6, pp. 48–56, Nov./Dec. 2019.

[88] V. Hassija *et al.*, "Interpreting black-box models: A review on explainable artificial intelligence," *Cognitive Computation*, vol. 16, pp. 45–74, 2024.

[89] A. Vaswani *et al.*, "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 6000–6010.

[90] S. Tuli, G. Casale, and N. R. Jennings, "Tranad: deep transformer networks for anomaly detection in multivariate time series data," *Proc. VLDB Endow.*, vol. 15, no. 6, pp. 1201–1214, Feb. 2022.

[91] P.-S. Murvay and B. Groza, "Security shortcomings and countermeasures for the sae j1939 commercial vehicle bus protocol," *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 4325–4339, May 2018.

[92] S. Mukherjee *et al.*, "Practical dos attacks on embedded networks in commercial vehicles," in *Proc. 12th Int. Conf. Inf. Syst. Secur.*, 2016, pp. 23–42.

[93] Y. Burakova *et al.*, "Truckhacking: An experimental analysis of the sae j1939 standard," in *Proc. 10th USENIX Workshop Offensive Technol.*, 2016, pp. 211–220.

[94] M. Zachos, "Securing j1939 communications using strong encryption with fips 140-2," SAE Tech. Paper, Tech. Rep. 2017-01-0020, Mar. 2017.

[95] C. Jichici *et al.*, "Effective intrusion detection and prevention for the commercial vehicle sae j1939 can bus," *IEEE Trans. Intell. Transport. Syst.*, vol. 23, no. 10, pp. 17 425–17 439, Oct. 2022.

[96] M. Rogers *et al.*, "Detecting can attacks on j1939 and nmea 2000 networks," *IEEE Trans. Dependable Secure Comput.*, vol. 20, no. 3, pp. 2406–2420, May/Jun. 2023.

[97] S. Mukherjee, R. Chatterjee, and J. Daily, "Trucksentry: Context aware intrusion detection and prevention system for j1939 networks," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 6, pp. 294–309, 2025.

[98] H. Shirazi, I. Ray, and C. Anderson, "Using machine learning to detect anomalies in embedded networks in heavy vehicles," in *Foundations and Practice of Security*, ser. Lecture Notes in Computer Science. Springer, 2020, vol. 12146, pp. 39–55.

[99] Özdemir *et al.*, "A survey of anomaly detection in in-vehicle networks," arXiv, 2024.

[100] M. E. Verma *et al.*, "Can-d: A modular four-step pipeline for comprehensively decoding controller area network data," *IEEE Trans. Veh. Technol.*, vol. 70, no. 10, pp. 9685–9700, Oct. 2021.

[101] S. Wan, X. Zhang, and L. Dou, "Shannon entropy of binary wavelet packet subbands and its application in bearing fault extraction," *Entropy*, vol. 20, no. 260, 2018.

[102] Penn State Eberly College of Science, "10.2 - autocorrelation and time series methods," STAT 462: Applied Regression Analysis. [Online]. Available: https://online.stat.psu.edu/stat462/node/188/, n.d.

[103] F. Pollicino, D. Stabili, and M. Marchetti, "Performance comparison of timing-based anomaly detectors for controller area network: A reproducible study," *ACM Trans. Cyber-Phys. Syst.*, vol. 8, no. 2, pp. Art. 15, 24 pages, Apr. 2024.

[104] A. A. Wani, "Comprehensive analysis of clustering algorithms: exploring limitations and innovative solutions," *PeerJ Comput. Sci.*, vol. 10, p. e2286, Aug. 2024.

[105] R. J. G. B. Campello *et al.*, "Hierarchical density estimates for data clustering, visualization, and outlier detection," *ACM Trans. Knowl. Discov. Data*, vol. 10, no. 1, pp. Art. 5, 51 pages, Jul. 2015.

[106] Y. Huang *et al.*, "Unsupervised interpolation recovery method for spectrum anomaly detection and localization," *Space: Science & Technology*, 2023.

[107] D. Hallac *et al.*, "Drive2vec: Multiscale state-space embedding of vehicular sensor data," arXiv preprint, 2018, [Online]. Available: https://arxiv.org/abs/1806.04795.

[108] H. Li *et al.*, "Anomaly detection of aviation data bus based on sae and imd," *Computers & Security*, vol. 137, p. 103619, 2024.

[109] R. Jiang, Z. Yang, and J. Zhao, "A complete deep support vector data description for one class learning," *IEEE Access*, vol. 11, pp. 117 494–117 507, 2023.

[110] M. Orabi *et al.*, "Anomaly detection in smart manufacturing: An adaptive adversarial transformer-based model," *Journal of Manufacturing Systems*, vol. 77, pp. 591–611, Dec. 2024.

[111] R. Xu and K. Ding, "Large language models for anomaly and out-of-distribution detection: A survey," arXiv preprint, 2024, [Online]. Available: https://arxiv.org/abs/2409.06765.

[112] University of Turku, "Can bus dataset collected from a heavy-duty truck," University of Turku, Version 1, 2021, [Online]. Available: https://doi.org/10.23729/3160254e-85e9-4268-a636-5b3e54091706.

[113] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 1, pp. 1–39, 2012.

[114] R. Chatterjee, S. Mukherjee, and J. Daily, "Exploiting transport protocol vulnerabilities in sae j1939 networks," 2023, available at: https://www.ndss-symposium.org/wp-content/uploads/2023/02/vehiclesec2023-23053-paper.pdf.

APPENDIX A MULTI-PACKET REASSEMBLY MODULE

A.1 Multi-message Transmission

In **J1939**, messages can contain more than 8 bytes of data using a multi-message transmission process known as the **Transport Protocol (TP)**.

Here's how it works as explained in Chatterjee et al. [114] paper:

- When a message exceeds 8 bytes, it must be split across multiple frames.
- The **Transport Protocol (TP)** is used to manage the data transfer. There are two types of TP:
  - **Destination-Specific Transfer (Peer to peer)**: A connection is established between two devices. The sender initiates a transfer by sending a **Request to Send (RTS)**, and the receiver replies with a **Clear to Send (CTS)** message. The sender then sends **Data Transfer (DT)** messages in sequence, each carrying 7 bytes of data.
  - **Broadcast Transfer**: The sender broadcasts a **Broadcast Announcement Message (BAM)** to all devices, followed by the DT messages. No acknowledgment is required, making it simpler but less reliable than destination-specific transfer.
- For both transfer types, the data is transmitted in multiple packets, and the receiver reassembles the message using sequence numbers included in each DT message.
- In destination-specific transfers, flow control is enforced by the CTS, which specifies how many packets the sender can transmit before needing further acknowledgment. This ensures that the receiver can process the data at a manageable pace.

Those messages are harder to analyze as the PGN related to the transmission is not part of the CAN ID of any of the transmitted frames, but is rather located in the payload of the first frame.

A.1.1 CM Packet Format (PGN: 0xEC00)

The CM packet specifies the initiation and control of multi-packet messages. Fields are:

Field	Size	Description
Control Byte	1 byte	Specifies the message type: RTS (0x10), CTS (0x11), End of Message Acknowledge (0x13), or BAM (0x20).
Total Message Size	2 bytes	Total number of bytes in the multi-message (Intel byte order).
Total Packet Count	1 byte	Total number of packets required to send the message.
Reserved/Max Packets	1 byte	Unused for BAM (0xFF). For RTS, indicates the max packets in a single CTS request.
PGN of Multi-Message	3 bytes	The PGN of the actual message being transmitted (Intel byte order).

Table A.1 CM packet fields for J1939 multi-packet messages

A.1.2 DT Packet Format (PGN: 0xEB00)

The DT packet carries the segmented payload data. Fields are:

Table A.2 DT packet fields for J1939 multi-packet messages

Field	Size	Description
Sequence Number	1 byte	Sequence number of the current packet (1–255).
Payload	7 bytes	Data bytes for this packet, including the sequence number.

The DT message uses the first byte as a sequence counter (1 to 255), while the remaining 7 bytes contain the segmented data payload. As a result, a single J1939 TP sequence can carry up to $7 \times 255 = 1785$ data bytes.

A.1.3 RTS/CTS (Peer-to-Peer)

1. **RTS Frame:**
   - The sender sends a CM packet with Control Byte `0x10` (RTS).
   - Includes the total message size, total packet count, and PGN of the intended message.
2. **CTS Frame:**
   - The receiver sends a CM packet with Control Byte `0x11` (CTS).

- Specifies how many packets it can process and the starting sequence number.

3. **DT Frames:**
   - The sender transmits the data in DT packets. Each DT frame includes a sequence number and a 7-byte segment of data.
4. **EoMA Frame:**
   - Once all packets are sent, the receiver sends a CM packet with Control Byte `0x13` (EoMA), confirming successful receipt.

A.1.4 BAM (Broadcast) Transfer

1. BAM Frame:
   - The sender initiates a CM packet with Control Byte `0x20` (BAM).
   - Specifies the total message size, total packet count, and PGN of the broadcast message.
2. DT Frames:
   - The sender broadcasts the data in DT packets, using sequence numbers to organize the data. Receivers independently reassemble the data.

A.2 Steps to Trace and Reassemble Multi-Message Data

Step 1: Identify the multi-packet messages

- Look for a CM packet with:
  - PGN: `0xEC00` (CAN ID `0x18ECFFXX`).
  - Control Byte: RTS (`0x10`), CTS (`0x11`), End of Message Acknowledge (`0x13`), or BAM (`0x20`).
  - This indicates the start of a new broadcast sequence.
- Extract the following fields:
  - **Total Message Size**: Specifies the total payload size.
  - **Total Packet Count**: Indicates the number of DT packets expected.

  - **PGN of Multi-Message**: Extracted from the last 3 bytes of the payload.

Step 2: Convert PGN to 29-Bit Identifier

- Use the PGN of the multi-message (e.g., `0xFEE3`) to calculate the corresponding 29-bit CAN ID:
  - `29-bit CAN ID = Priority (3 bits) + PGN (18 bits) + Source Address (8 bits)`

Step 3: Collect and Validate DT Frames

- Look for DT packets (PGN `0xEB00`) with sequence numbers from 1 to the total packet count.
- Extract the 7-byte payload of each DT packet (excluding the sequence number).

Step 4: Construct the Reassembled Data

- Concatenate the 7-byte payloads in order of their sequence numbers.
- Ensure the total number of packets matches the expected count from the CM frame.
- Validate that the total data length matches the size specified in the CM frame.

Broadcast Message Reassembly Example

```
{ 0x18ECFF01, 0xEC00, [0x20, 0x2A, 0x00, 0x06, 0xFF, 0xE3, 0xFE, 0x00] } # BAM Frame
{ 0x18EBFF01, 0xEB00, [0x01, 0x41, 0x42, 0x43, 0x44, 0x45, 0x46, 0x47] } # DT Frame1
{ 0x18EBFF01, 0xEB00, [0x02, 0x48, 0x49, 0x4A, 0x4B, 0x4C, 0x4D, 0x4E] } # DT Frame2
{ 0x18EBFF01, 0xEB00, [0x03, 0x4F, 0x50, 0x51, 0x52, 0x53, 0x54, 0x55] } # DT Frame3
{ 0x18EBFF01, 0xEB00, [0x04, 0x56, 0x57, 0x58, 0x59, 0x5A, 0x5B, 0x5C] } # DT Frame4
{ 0x18EBFF01, 0xEB00, [0x05, 0x5D, 0x5E, 0x5F, 0x60, 0x61, 0x62, 0x63] } # DT Frame5
{ 0x18EBFF01, 0xEB00, [0x06, 0x64, 0x65, 0x66, 0x67, 0x68, 0x69, 0x6A] } # DT Frame6
```

1. **Identify the BAM Frame:**
   - CAN ID: `0x18ECFF01`, PGN: `0xEC00`.
   - Control Byte: `0x20` (BAM).
   - Total Message Size: `0x002A` (42 bytes).

- Total Packets: `0x06`.
- PGN of Multi-Message: `0xFEE3` (Engine Configuration 1).

2. **Convert PGN to 29-Bit CAN ID:**
   - PGN `0xFEE3` is converted to CAN ID `0x18FEE301`.
3. **Collect and Order DT Frames:**
   - DT Frame (Seq 1): `[0x41, 0x42, 0x43, 0x44, 0x45, 0x46, 0x47]`
   - DT Frame (Seq 2): `[0x48, 0x49, 0x4A, 0x4B, 0x4C, 0x4D, 0x4E]`
   - DT Frame (Seq 3): `[0x4F, 0x50, 0x51, 0x52, 0x53, 0x54, 0x55]`
   - DT Frame (Seq 4): `[0x56, 0x57, 0x58, 0x59, 0x5A, 0x5B, 0x5C]`
   - DT Frame (Seq 5): `[0x5D, 0x5E, 0x5F, 0x60, 0x61, 0x62, 0x63]`
   - DT Frame (Seq 6): `[0x64, 0x65, 0x66, 0x67, 0x68, 0x69, 0x6A]`
4. **Reassemble the Payload:**
   - Concatenate payloads from all DT frames:
     `{0x18FEE301, 0xFEE3, [0x41, 0x42, 0x43, 0x44, 0x45, 0x46, 0x47, 0x48, 0x49, 0x4A, 0x4B, 0x4C, 0x4D, 0x4E, 0x4F, 0x50, 0x51, 0x52, 0x53, 0x54, 0x55, 0x56, 0x57, 0x58, 0x59, 0x5A, 0x5B, 0x5C, 0x5D, 0x5E, 0x5F, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x67, 0x68, 0x69, 0x6A]}`
5. **Validate Completion:**
   - Verify the total number of packets (6) matches the expected count (6).
   - Check if the data length (42 bytes) matches the BAM frame size (42 bytes).

Peer-to-Peer Message Reassemble Example

```
RTS Frame: Request to send (transmitter initiates)
{ 0x18ECFF01, 0xEC00, [0x10, 0x2A, 0x00, 0x06, 0xFF, 0xE3, 0xFE, 0x00] }

CTS Frame: Receiver grants permission to send 3 packets
{ 0x18EC0101, 0xEC00, [0x11, 0x03, 0x01, 0x06, 0xFF, 0xE3, 0xFE, 0x00] }

DT Frames: Transmitter sends the first 3 packets
```

```
{ 0x18EBFF01, 0xEB00, [0x01, 0x41, 0x42, 0x43, 0x44, 0x45, 0x46, 0x47] }
{ 0x18EBFF01, 0xEB00, [0x02, 0x48, 0x49, 0x4A, 0x4B, 0x4C, 0x4D, 0x4E] }
{ 0x18EBFF01, 0xEB00, [0x03, 0x4F, 0x50, 0x51, 0x52, 0x53, 0x54, 0x55] }

CTS Frame: Receiver grants permission for the next 3 packets
{ 0x18EC0101, 0xEC00, [0x11, 0x03, 0x04, 0x06, 0xFF, 0xE3, 0xFE, 0x00] }

DT Frames: Transmitter sends the remaining 3 packets
{ 0x18EBFF01, 0xEB00, [0x04, 0x56, 0x57, 0x58, 0x59, 0x5A, 0x5B, 0x5C] }
{ 0x18EBFF01, 0xEB00, [0x05, 0x5D, 0x5E, 0x5F, 0x60, 0x61, 0x62, 0x63] }
{ 0x18EBFF01, 0xEB00, [0x06, 0x64, 0x65, 0x66, 0x67, 0x68, 0x69, 0x6A] }

EoMA Frame: Receiver acknowledges successful completion
{ 0x18ECFF01, 0xEC00, [0x13, 0x27, 0x00, 0x06, 0xFF, 0xE3, 0xFE, 0x00] }
```

1. **RTS Frame:**
   - Control Byte: `0x10` (RTS)
   - The transmitter requests to send a message with:
     - Total Size: 42 bytes (`0x2A`)
     - Total Packets: 6 packets (`0x06`)
     - PGN of Message: `0xFEE3`
2. **Convert PGN to 29-Bit CAN ID:**
   - PGN `0xFEE3` is converted to CAN ID `0x18FEE301`
3. **First CTS Frame:**
   - Control Byte: `0x11` (CTS)
   - The receiver grants permission to send 3 packets, starting with sequence 1
4. **First Set of DT Frames:**
   - DT Frame Sequence Numbers: `0x01`, `0x02`, `0x03`
   - Data extracted from payloads:
     - Seq 1: `[0x41, 0x42, 0x43, 0x44, 0x45, 0x46, 0x47]`
     - Seq 2: `[0x48, 0x49, 0x4A, 0x4B, 0x4C, 0x4D, 0x4E]`

  - Seq 3: `[0x4F, 0x50, 0x51, 0x52, 0x53, 0x54, 0x55]`

5. **Second CTS Frame:**
   - Control Byte: `0x11` (CTS)
   - The receiver grants permission for the next 3 packets, starting with sequence 4
6. **Second Set of DT Frames:**
   - DT Frame Sequence Numbers: `0x04`, `0x05`, `0x06`
   - Data extracted from payloads:
     - Seq 4: `[0x56, 0x57, 0x58, 0x59, 0x5A, 0x5B, 0x5C]`
     - Seq 5: `[0x5D, 0x5E, 0x5F, 0x60, 0x61, 0x62, 0x63]`
     - Seq 6: `[0x64, 0x65, 0x66, 0x67, 0x68, 0x69, 0x6A]`
7. **EoMA Frame:**
   - Control Byte: `0x13` (End of Message Acknowledge)
   - The receiver acknowledges successful reception of the entire message
8. **Reassemble the Payload:**
   - Concatenate payloads from all DT frames:
     `{0x18FEE301, 0xFEE3, [0x41, 0x42, 0x43, 0x44, 0x45, 0x46, 0x47, 0x48, 0x49, 0x4A, 0x4B, 0x4C, 0x4D, 0x4E, 0x4F, 0x50, 0x51, 0x52, 0x53, 0x54, 0x55, 0x56, 0x57, 0x58, 0x59, 0x5A, 0x5B, 0x5C, 0x5D, 0x5E, 0x5F, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x67, 0x68, 0x69, 0x6A]}`

Control Byte	**Message Size (2,3)**	**Total Packets (4)**	**Max. Packets (5)**	**PGN (6–8)**
16	Number of bytes	Total number of packets	Max packets in response to CTS (0xFF = No limit)	PGN of the multi-packet message (6 = LSB, 8 = MSB)

Table A.3 TP.CM_RTS (Connection Mode Request to Send)

Control Byte	**Total Packets (2)**	**Next Packet (3)**	**Reserved (4,5)**	**PGN (6–8)**
17	Should not exceed RTS byte 5	Next packet number	Should be filled with 0xFF	PGN of the multi-packet message (6 = LSB, 8 = MSB)

Table A.4 TP.CM_CTS (Connection Mode Clear to Send)

Control Byte	**Abort Reason (2)**	**Reserved (3–5)**	**PGN (6–8)**
255	See description	Should be filled with 0xFF	PGN of the multi-packet message (6 = LSB, 8 = MSB)

Table A.5 TP.Conn_Abort (Connection Abort)

Control Byte	**Message Size (2,3)**	**Total Packages (4)**	**Reserved (5)**	**PGN (6–8)**
19	Number of bytes	Total number of packages	Should be filled with 0xFF	PGN of the multi-packet message (6 = LSB, 8 = MSB)

Table A.6 TP.CM_EndOfMsgACK (End of Message Acknowledgment)

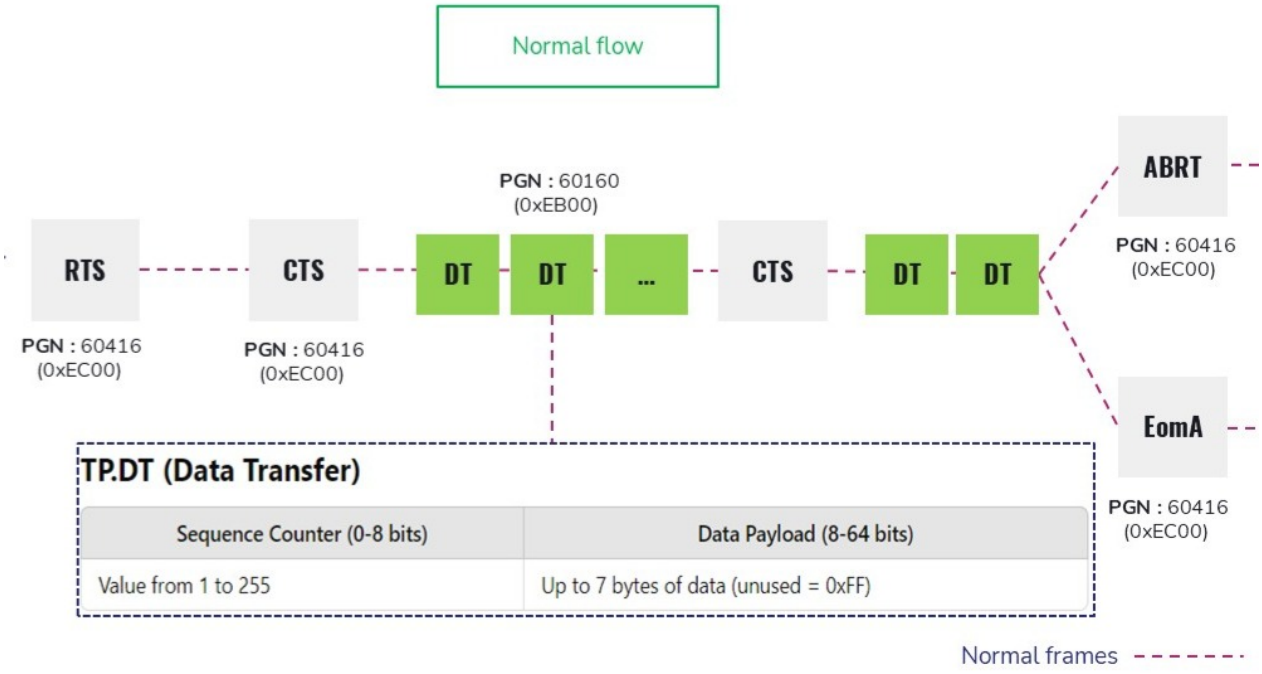


Figure A.1 Transport Protocol – Destination-Specific Transfer (Peer to peer) - Normal flow

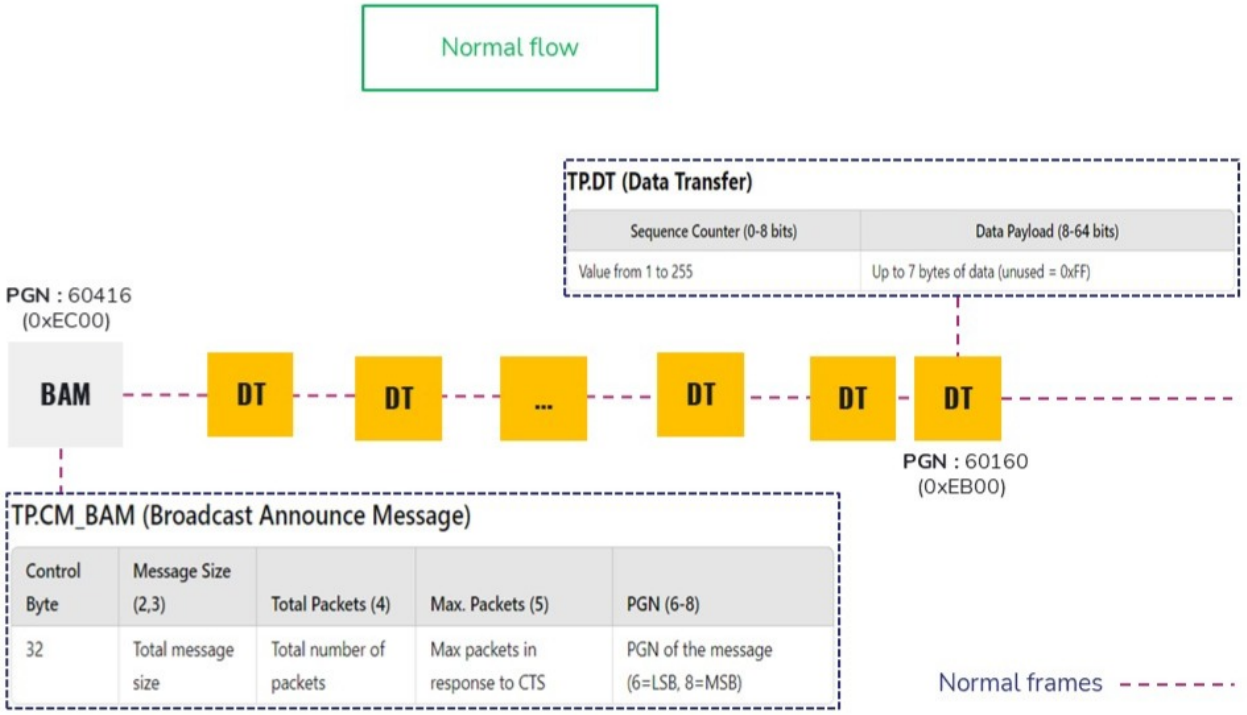


Figure A.2 Transport Protocol - Broadcast Transfer - Normal flow

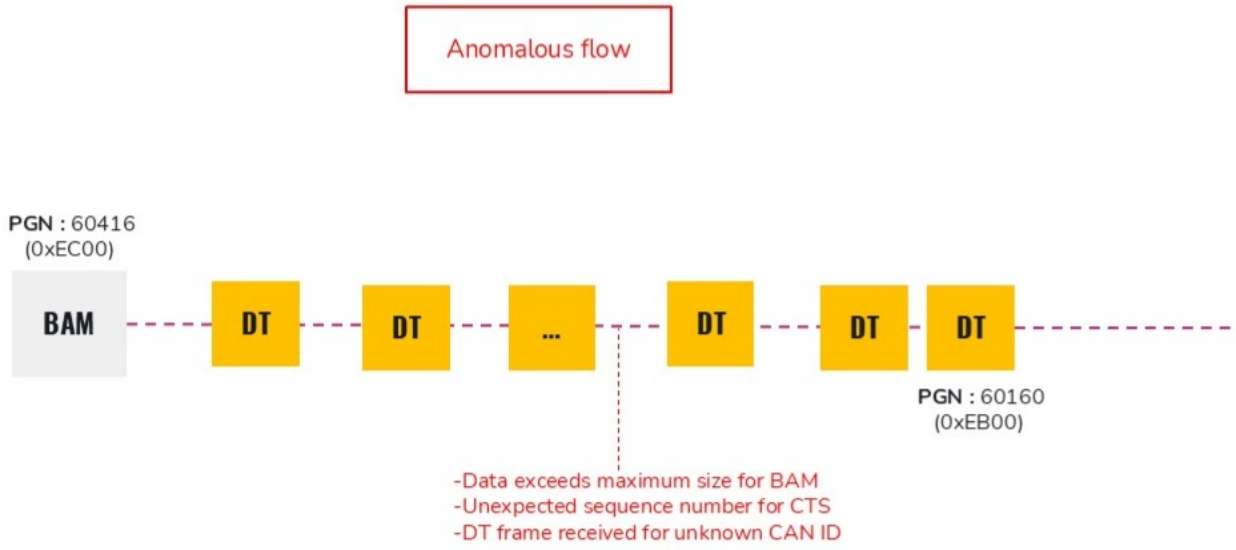


Figure A.3 Transport Protocol – Broadcast Transfer - Anomalous flow

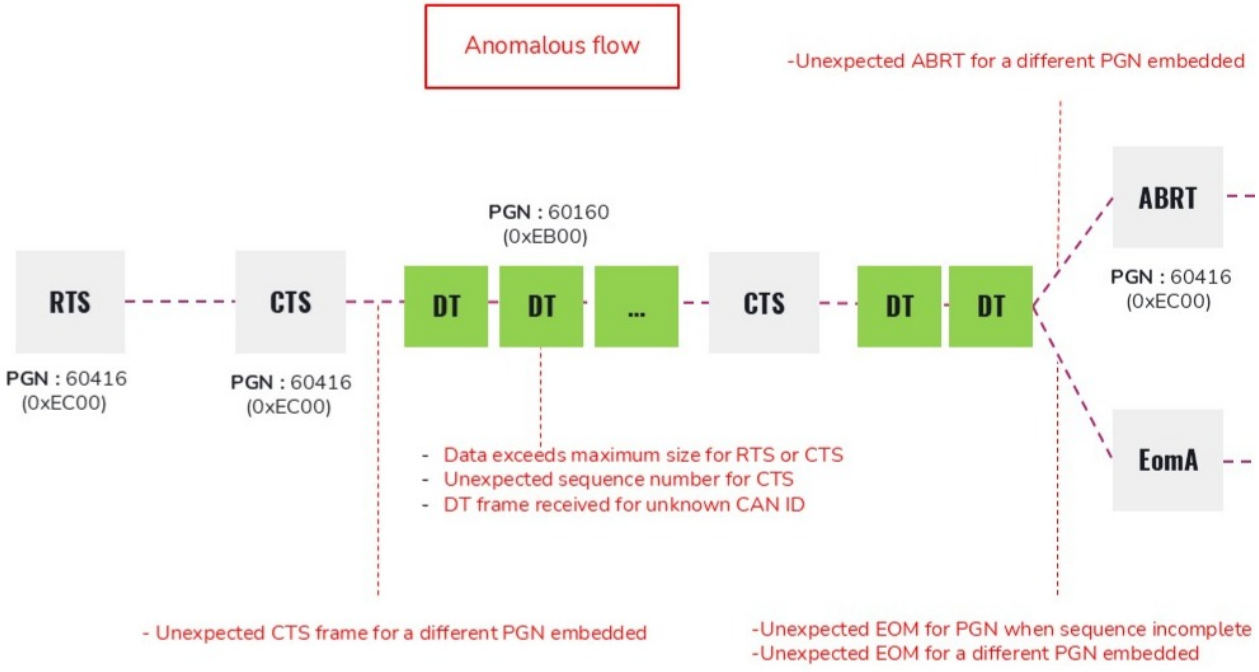


Figure A.4 Transport Protocol – Destination-Specific Transfer (Peer to peer) - Anomalous flow

APPENDIX B ILLUSTRATIVE EXAMPLES OF DIAGNOSTIC OUTPUTS

```
svdd_scores tensor([ 0.0000,  9.0884,  0.0000,  0.0000, -0.9070,  0.0000,  0.0000,  0.0000,
         0.0000,  9.0011,  9.1541,  0.0000,  5.4316,  0.0000,  0.0000,  0.0000,
         4.8624,  0.0000, -0.4578,  0.0000, 10.8588,  8.7041,  0.0000,  0.5598,
        11.7721,  0.0000, -0.4466,  0.0000,  0.5073,  1.2992,  0.0000,  5.4316],
       device='cuda:0')
losses tensor([0.0000, 0.0016, 0.0000, 0.0000, 0.0319, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0018, 0.0093, 0.0000, 0.0146, 0.0000, 0.0000, 0.0000, 0.0149, 0.0000,
        0.0545, 0.0000, 0.0120, 0.0013, 0.0000, 0.0141, 0.0111, 0.0000, 0.0552,
        0.0000, 0.0022, 0.0161, 0.0000, 0.0146], device='cuda:0')
[DEBUG] Messages with NO active SPNs: [15200, 15202, 15203, 15205, 15206, 15207, 15208, 15211, 15213, 15214, 15215, 15217, 15219, 15222, 15225, 15227]
[Traceback]Message 15200 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15202 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15203 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15204 anomalous → suspicious SPNs: [110, 109, 114, 113, 108], Potential outlier SPNs [108, 109, 110, 111, 112, 113, 114, 115]
[Traceback]Message 15205 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15206 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15207 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15208 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15211 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15213 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15214 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15215 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15217 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15218 anomalous → suspicious SPNs: [46, 39, 45, 32, 42], Potential outlier SPNs [24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36]
[Traceback]Message 15219 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15222 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15225 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15226 anomalous → suspicious SPNs: [46, 39, 40, 45, 41], Potential outlier SPNs [24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36]
[Traceback]Message 15227 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
[Traceback]Message 15230 anomalous → suspicious SPNs: [3, 4, 1, 0, 2], Potential outlier SPNs []
True Positives: 17
True Negatives: 12
False Positives: 3
False Negatives: 0
```

Figure B.1 Example of a captured inference-window diagnostic output, including reconstruction dimensions, encoder representations, SVDD scores, per-sample losses, SPN-level anomaly tracing, and final classification metrics

```
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0x4a0738f
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0x1928c17f
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0x5dc5f47
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0xe0b1fff
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0xab7b02f
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0x1535367f
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0x3d87e37
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0x2f60b4b
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0xf19d86f
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0x121efe1f
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0xa54856f
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0x23c9387
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0x196de0bf
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0x1bdb7a3f
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0x14bf7c7f
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][WARNING] - [test] Discarded unseen CAN ID: 0x7d71b7f
[2025-11-01 15:17:16,136][anomaly_detection_algorithm.seq_to_seq_models.dataset][INFO] - [test] canid_num=114, unknown_CANID_filtered_Messages=7218
[2025-11-01 15:17:16,137][anomaly_detection_algorithm.seq_to_seq_models.dataset][INFO] - [VAL] Unknown CAN IDs removed: 0
[2025-11-01 15:17:16,137][anomaly_detection_algorithm.seq_to_seq_models.dataset][INFO] - [TEST] Unknown CAN IDs removed: 7218
```

Figure B.2 Example of a diagnostic log showing how unseen CAN IDs are filtered during inference, including per-identifier warnings and final counts of removed frames

APPENDIX C ILLUSTRATIVE EXAMPLES OF FEATURE-LEVEL INTERPRETABILITY

CAN ID	PGN Name	Clustering Exclusion Reason
0x0	Unknown PGN	PGN is 0 (reserved/invalid), lacks usable content
0x18ea008b	Unknown PGN	PGN is known (59904) but reused generically (Request PGN), carries various message types
0x18eaff0b	Unknown PGN	Same PGN (59904) as above, generically used across multiple functions
0x18ec008b	Unknown PGN	PGN 60416 (Transport Protocol - DT), protocol layer, not a signal source
0x18fecb00	Unknown PGN	PGN 65227 lacks definition in the DBC, ambiguous content
0x18ff2e21	Unknown PGN	PGN 65326 undefined, highly irregular interval and entropy values
0x1ceb8b00	Unknown PGN	PGN 60299 undefined, possibly proprietary or encrypted content
0x1cec8b00	Unknown PGN	PGN 60555 undefined, lacks signal-level meaning
0xc000003	Unknown PGN	PGN is 0, possibly a malformed or reserved message
0xc000f03	Unknown PGN	PGN 15 is very low and typically reserved, uninformative for clustering

Figure C.1 Representative CAN IDs excluded from clustering. These examples were extracted from the analysis of clustering outputs and illustrate why certain CAN IDs were not grouped: they use undefined or reserved PGNs, involve protocol management messages, exhibit abnormal statistical characteristics

CAN ID	PGN Name	PGN (Hex)	Outlier Score	Clustering Exclusion Reason
0x18f0090b	Vehicle Dynamic Stability Control 2	0xf009	0.18554872699846825	Extremely low average interval; indicates bursty or high-frequency bursts
0x18fc2a00	Operator Inducement Information	0xfc2a	0.15631814131210445	Zero entropy and variance; likely static message or dead signal
0x18fcdc00	Cruise Control / Vehicle Speed 3	0xfcdc	0.1138092267918269	Very low entropy and variance; lacks dynamic behavior
0x18fce700	Engine Configuration 3	0xfce7	0.04928769878469285	High interval variance; unstable message timing
0x18fd7b00	Aftertreatment 1 Service 1	0xfd7b	0.2528271590628825	High mean rate of change; abrupt signal changes
0x18fea400	Engine Temperature 2	0xfea4	0.24464695319914395	Very high rate of change with near-zero variance
0x18fec100	High Resolution Vehicle Distance	0xfec1	0.2528271590628825	Rapid rate of change; no variance; suspiciously flat yet dynamic
0x18fec117	High Resolution Vehicle Distance	0xfec1	0.2515026522387016	Same as above; likely duplicate signal or redundant data channel
0x18fedd00	Turbocharger	0xfedd	0.05670954036024645	Low variance, borderline inactive
0x18fee500	Engine Hours, Revolutions	0xfee5	0.3297416144094198	High coefficient of variation; unstable frequency

Figure C.2 These examples were extracted from the analysis of clustering outputs and illustrate why certain CAN IDs were not grouped: they exhibit abnormal statistical characteristics (e.g., extreme variance, entropy, or rate of change), display nearly constant or static behavior, show unstable timing patterns such as bursty intervals or inconsistent frequency, or represent redundant or duplicate channels that reduce semantic distinctiveness.