# SeqWatch: Unsupervised Sequence-based Intrusion Detection System for Automotive Ethernet

**Maurício S. G. A. Leandro**[1,2], **Paulo Freitas de Araujo-Filho**[1], **Divanilson R. Campelo**[1], **Luigi F. Marques da Luz**[1,2]

[1]Centro de Informática – Universidade Federal de Pernambuco (CIn - UFPE)
Av. Jorn. Aníbal Fernandes – s/n – Recife – PE – Brazil

{msgal,pfreitas,dcampelo,lfml}@cin.ufpe.br

[2]Centro de Estudos e Sistemas Avançados do Recife (C.E.S.A.R.)
Rua Bione, 220 – 50030-390 – Recife – PE – Brazil

{msgal,lfml}@cesar.org.br

***Abstract.*** *Modern connected vehicles are increasing the demand for Ethernet in automotive networks due to its ability to provide high-bandwidth and flexible in-vehicle communication. However, Ethernet lacks built-in authentication and encryption, which has led to growing interest in Intrusion Detection Systems (IDS) as a defense mechanism to detect malicious activities when other security mechanisms are not present or fail. In this work, we present SeqWatch, an unsupervised IDS that uses a sequence-based deep learning model capable of capturing the temporal relationships in network traffic. SeqWatch can identify previously unseen (zero-day) attacks by training only on normal traffic data. Our experimental results show that SeqWatch outperforms other state-of-the-art unsupervised automotive IDSs, achieving higher detection rates in attacks from two publicly available datasets.*

## 1. Introduction

Connected vehicles are becoming a reality, providing significant benefits such as improved efficiency through optimized traffic flow and fuel consumption, improved comfort with personalized infotainment systems and seamless smartphone integration, and increased safety through features such as collision avoidance and emergency braking systems [Wu et al. 2020]. To achieve these advantages, connected vehicles depend on advanced electronic components and sophisticated sensors, enabling functionalities such as 360-degree surround-view parking assistance and real-time data exchange [Wu et al. 2020].

Modern sensors, such as cameras, the distribution of automotive functions across multiple Electronic Control Units (ECUs), and the shift to service-oriented architectures have revealed the limitations of Controller Area Network (CAN) in managing data-heavy applications due to its restricted bandwidth and scalability [da Luz et al. 2024]. The standardization of IEEE 100BASE-T1 has established Ethernet as a high-bandwidth, flexible In-Vehicle Network (IVN) solution, enabling advanced vehicle technologies. To support Quality of Service (QoS), standards from Audio Video Bridging (AVB) and Time Sensitive Networking (TSN) ensure time synchronization, low latency, and reliable communication in switched Ethernet networks [Matheus and Königseder 2021]. For example, IEEE 1722-2016 facilitates reliable time-sensitive traffic transmission via the Audio

Video Transport Protocol (AVTP), while the generalized Precision Time Protocol (gPTP) synchronizes nodes for stream alignment [da Luz et al. 2024]. Despite these advancements, Ethernet must coexist with CAN, which remains cost-effective and efficient for safety-critical applications.

Although increased car connectivity provides numerous benefits, it also expands the attack surface, making vehicles more vulnerable to cyberattacks [Dibaei et al. 2020]. Hackers have demonstrated that cars can be exploited through these vulnerabilities, raising serious security concerns about the safety of drivers and passengers [Checkoway et al. 2011]. Exploiting connectivity points such as the On-Board Diagnostic II (OBD2) port, Bluetooth, and car radio can result in network hijacking, enabling attackers to remotely control critical functions of a vehicle, such as turning off the engine, steering, and braking. As a result, defending againts security threats in connected vehicles is an urgent priority [da Luz et al. 2024].

Traditional network security mechanisms, such as encryption and authentication, are challenging to implement in resource-constrained environments such as IVNs, as they introduce computing and transmission overhead that may conflict with strict timing requirements [Jo and Choi 2022]. As an alternative, Intrusion Detection Systems (IDSs) provide a second line of defense by monitoring networks and devices to detect and report malicious activities without requiring modifications to existing nodes [Wu et al. 2020]. IDSs are classified as either signature-based, which rely on known attack patterns and need frequent updates, or anomaly-based, which detect deviations from normal behavior. The latter addresses the limitations of signature-based methods but often producing higher false positive rates [Nisioti et al. 2018]. While traditional statistical methods can be used in anomaly-based IDSs, machine learning (ML) approaches are increasingly favored for their superior detection capabilities and ability to identify more complex attacks [Freitas de Araujo-Filho et al. 2021].

Machine learning-based IDSs can be trained using supervised or unsupervised approaches. A supervised IDS, often referred to as an attack classifier, is limited to detecting attacks present in its training dataset [Alkhatib et al. 2021]. This poses a significant drawback in the field of cybersecurity, where new threats, such as zero-day attacks, continuously emerge [Nisioti et al. 2018]. In contrast, unsupervised IDSs are trained solely on benign data, identifying malicious activity by detecting deviations from the system's normal behavior. This approach enhances the IDS's ability to detect unknown attacks.

In addition, an offline IDS analyzes network traffic after collection, enabling deep inspection of patterns to detect both known and unknown threats, such as zero-day vulnerabilities, by examining historical data and subtle deviations from normal behavior. This approach offers advantages, including superior detection accuracy through dedicated analysis resources, cost-effectiveness due to reduced infrastructure demands, and value for post-incident forensics. Offline IDSs complement real-time systems to provide a second layer of security for a comprehensive network protection. They can generate detailed periodic reports with actionable insigths, adding useful value to the intrusion detection strategy.

**Contributions.** This work introduces SeqWatch, an offline unsupervised anomaly-based IDS designed to detect known and unknown (zero-day) attacks on au-

tomotive networks. SeqWatch leverages sequential machine learning models, specifically a sequence-to-sequence (Seq2seq) architecture based on Long Short-Term Memory (LSTM) networks [Sutskever et al. 2014], to capture the temporal relationships in network traffic, where each packet depends on its predecessors. Moreover, SeqWatch aims to classify benign and malicious packets in heterogeneous networks, analyzing traffic from AVTP, gPTP, and CAN protocols, representing video streaming, synchronization, and safety-critical applications. Its versatility is demonstrated using two datasets: one for Ethernet-based traffic and another for CAN-based traffic. In a nutshell, the main contributions of this paper are as follows:

- The proposal of SeqWatch, an unsupervised deep learning-based IDS capable of detecting previously unseen (zero-day) attacks.
- The experimental evaluation and comparison of SeqWatch with other state-of-the-art automotive IDSs, whose results highlight SeqWatch's effectiveness in the detection of different automotive cyber-attacks.

This paper is organized as follows. Section 2 reviews recent advances in automotive Ethernet IDSs. Section 3 describes the threat model considered in our work, including an analysis of attack complexity and temporal relationships. Section 4 presents our proposed architecture. Section 5 details the experimental setup and methodology used to evaluate the IDS. Section 6 presents the evaluation results and compares them with those of state-of-the-art automotive Ethernet IDSs from the literature. Finally, Section 7 concludes the paper and outlines the directions for future work.

## 2. Related Work

Although only a few researchers have proposed IDSs for detecting malicious activity in automotive Ethernet networks, most of these focus on supervised methods, with unsupervised approaches being even less explored. For instance, the authors in [Han et al. 2023] proposed an IDS based on Deep Convolutional Neural Network (DCNN) techniques to classify network traffic as benign or malicious. Their method uses a complex dataset from a heterogeneous automotive Ethernet network containing packets from protocols such as AVTP, CAN over User Datagram Protocol (UDP), and gPTP. Additionally, the authors introduced a novel feature extraction method using wavelet transforms. While the dataset includes several attack scenarios, the supervised approach limits the IDS to detecting only the attacks present in the dataset.

Similarly, [Jeong et al. 2021] proposed an IDS using a 2D-Convolutional Neural Network (2D-CNN) to detect injection attacks in AVTP packets. The authors also released a publicly available dataset of injection attacks. However, like the previous work, the IDS relies on supervised learning and supports only binary detection. Likewise, [da Luz et al. 2024] proposed a multi-stage deep learning IDS was proposed with two detection stages: an attack detector for fast detection and an attack classifier for higher accuracy. This system employed a Random Forest classifier in the detection stage and a Pruned CNN in the classification stage. Additionally, the authors proposed a new feature generator algorithm. However, both stages rely on supervised learning, and the classifier is unable to detect new attack scenarios.

Another IDS, presented in [Alkhatib et al. 2021], used sequence-based Recurrent Neural Networks (RNNs) to detect malicious activity. Unfortunately, it was trained using

a synthetic dataset based on supervised learning, making it less applicable to real-world scenarios. Additionally, this approach was designed specifically for scalable service-oriented middleware over IP (SOME/IP) networks and cannot be generalized to other protocols. Similarly, the work in [Alkhatib et al. 2023] proposed a sequence-based IDS employing a state-of-the-art self-attention mechanism to detect malicious activity on SOME/IP. The authors constructed and made available a synthetic dataset containing man-in-the-middle attacks. However, like other works, their IDS relies on supervised learning and is limited to the dataset used, making it challenging to compare with other methods.

In contrast, [Alkhatib et al. 2022] evaluated the detection time, model size, and detection metrics of two autoencoder-based models—a convolutional autoencoder (CAE) and a LSTM autoencoder (LSTMAE)—to develop an anomaly detector capable of detecting zero-day cyberattacks in AVTP packets. While this work took a step toward unsupervised learning, the experiments were conducted on a low-complexity dataset. Finally, [Jeong et al. 2024] proposed a real-time method called AERO, which incorporates a multi-modal feature extractor capable of capturing protocol changes, payloads, and packet timestamps. It also incorporates an unsupervised deep neural network architecture to detect cyberattacks. Although AERO can detect novel attacks, its high computational complexity requires advanced GPU devices to achieve real-time performance, which may limit its practicality in real-world deployments.

Our proposed method stands out by detecting cyberattacks in complex scenarios involving different protocols and attacks with subtle deviations from normal behavior, such as replay attacks. It uses unsupervised training in an offline setting—an underexplored but effective approach for identifying unknown threats without requiring extensive infrastructure. Table 1 summarizes the above-mentioned works, highlighting their methods, datasets, and key characteristics.

**Table 1. Comparison of related work. AEID and TOW-IDS refer to datasets proposed in [Jeong et al. 2021] and [Han et al. 2023], respectively. The SOME/IP dataset was developed by the authors of the corresponding studies, while the SAD dataset was introduced in [Han et al. 2018].**

| Reference | Method | Dataset | Unsupervised |
|---|---|---|---|
| [Han et al. 2023] | DCNN | TOW-IDS | No |
| [Jeong et al. 2021] | 2D-CNN | AEID | No |
| [da Luz et al. 2024] | Multi-stage (RF and CNN) | TOW-IDS, AEID | No |
| [Alkhatib et al. 2021] | Sequential Model (RNN) | Synthetic SOME/IP | No |
| [Alkhatib et al. 2023] | SAID (Self-Attention) | Synthetic SOME/IP | No |
| [Alkhatib et al. 2022] | CAE and LSTM-AE Evaluation | AEID | Yes |
| [Jeong et al. 2024] | AERO (Multi-modal) | TOW-IDS | Yes |
| **Our Work** | **Sequential Model (SeqWatch)** | **TOW-IDS, SAD** | **Yes** |

## 3. Threat model

This section covers the six attack scenarios that we consider in our work and discusses their detectability, emphasizing the challenges involved. We used Wireshark [Combs and the Wireshark Contributors 2024] to examine how malicious packets differ from normal ones. The attack surface is defined by vulnerabilities in automotive Ethernet, particularly focusing on three key protocols: AVTP, gPTP, and CAN over UDP. It is

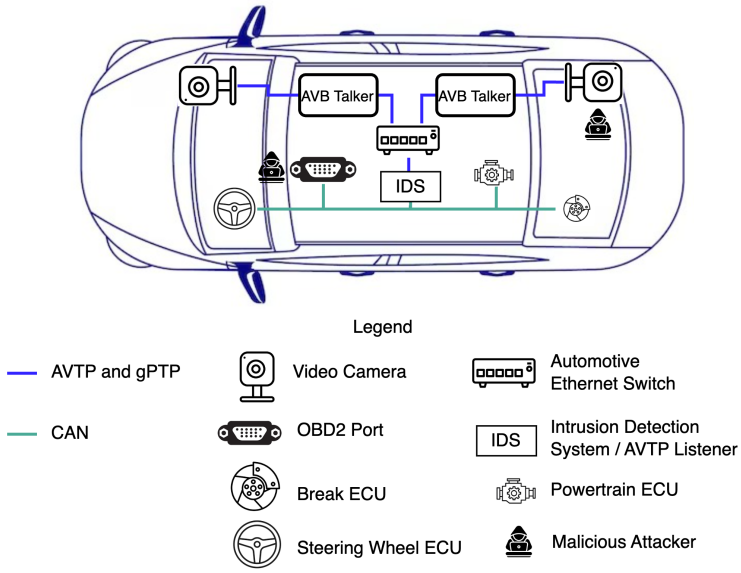



**Figure 1. In-vehicle network architecture and attack surface (Based on [da Luz et al. 2024]).**

assumed that attackers can gain access and inject malicious frames through compromised AVTP talkers and CAN nodes [Han et al. 2023].

We presented an IVN architecture that is illustrated in Figure 1. The system includes two AVTP talkers that send video streams from sensors using the AVTP protocol. These streams go through an automotive Ethernet switch to an AVTP listener. The listener converts the video streams into control signals that are sent over the CAN bus to control parts of the vehicle, like brakes, steering, and the powertrain [da Luz et al. 2024]. The synchronization between AVTP talkers and listeners is maintained via the gPTP protocol, ensuring timely transmission of data. The proposed IDS is deployed on the AVTP listener because it is capable of monitoring messages from the vehicle's in-vehicle networks, which include both automotive Ethernet and CAN. In the following, we describe each type of attack and how they can be detected, and Table 2 summarizes the analysis of these attacks, highlighting their detection challenges and characteristics.

- **Frame injection** attacks inject video frames into the IVN, so that the vehicle may misinterpret its surroundings [Jeong et al. 2021]. When a frame is reinjected, it disrupts the expected order of the AVTP sequence number [Matheus and Königseder 2021]. Since the sequence number follows a specific pattern to ensure data integrity, detecting such attack requires analyzing multiple frames, as a single frame may not reveal the anomaly.
- **PTP sync** attacks compromise the synchronization of clocks coordinated by the gPTP protocol across different nodes by transmitting incorrect or delayed synchronization messages. These malicious packets often differ noticeably from normal ones, making them potentially detectable through a single-packet analysis of PTP headers [Moussa et al. 2020].
- **Switch (MAC flooding)** attacks flood the switch's Media Access Control (MAC) table with packets that have random MAC addresses. When the MAC table is full, the switch sends packets to all devices on the network, causing congestion

[Han et al. 2023]. This attack can be easily detected as malicious packets have random and unusual MAC or IP addresses.

- **CAN replay** attacks reinject previously captured CAN messages, potentially causing ECUs to act out of sequence—posing serious safety risks, such as unintended braking [Jeong et al. 2021]. Since CAN IDs and headers look normal, identifying such an attack requires the detection of unusual data patterns or message sequences, which might be very challenging.
- **CAN Denial of Service (DoS)** attacks flood the network with high-priority messages, blocking legitimate traffic [Han et al. 2018]. It alters the CAN ID—mapped to a UDP destination port—as well as the data length and payload, which appear in the UDP data field. This well-known attack can be effectively detected by validating CAN IDs and filtering out messages with unknown IDs [da Luz et al. 2024].
- **CAN malfunction** attacks send random CAN messages with manipulated payloads and IDs, causing vehicles to behave unpredictably [Han et al. 2018]. Once random messages are injected, the attack may occasionally produce messages that resemble legitimate ones, which are harder to detect. However, the attack is more likely to generate messages that differ significantly from normal traffic, making detection easier and ultimately reducing the overall effectiveness of the attack.

**Table 2. Summary of attack scenarios, including the targeted protocols, changes in packet structures, alignment with normal system behavior, and detectability from a single packet. Key indicators: Exists in Normal Messages (ENM) and Single Packet Detection (SPD).**

| Attack | Protocol | Packet Changes | ENM | SPD |
|---|---|---|---|---|
| Frame injection | AVTP | Sequence number and payload | Yes | No |
| PTP sync | gPTP | PTP headers | No | Yes |
| MAC flooding | AVTP | MAC and IP addresses | No | Yes |
| CAN replay | CAN over UDP | Payload | Yes | No |
| CAN DoS | CAN over UDP | CAN ID and payload | No | Yes |
| CAN malfunction | CAN | Payload | No | Yes |

## 4. Proposed architecture

This section explains our proposed IDS architecture, including its design principles and two main components: the feature extractor, which outputs a sequence of feature vectors, and the anomaly detector. As described in Section 3, the IDS is placed on the AVTP listener. This is because the AVTP listener can monitor messages from the IVN, which include automotive Ethernet and CAN. The IDS collects data from the AVTP listener, which undergoes a feature extraction process to create a sequence of vectors. These vectors are then sent to the Anomaly Detector, which classifies the sequence as malicious or normal, as shown in Figure 2.

To group AVTP packets into sequences, we use the Feature-based Sliding Window (FSW) method [Zhang et al. 2020], which segments the data into fixed-size windows ($w\_size$), a crucial approach as some attacks require considering the temporal relationships between packets. Each sequence is represented as $S = \{\mathbf{p}_0, \ldots, \mathbf{p}_t, \ldots, \mathbf{p}_T\}$, where $\mathbf{p}_t \in \mathcal{D}$ is a packet at time $t$, $\mathcal{D}$ indicates the original dataset, and $T = w\_size - 1$ defines the sequence length. Packet sizes vary by protocol, such as AVTP packets (434 bytes), gPTP packets (60–90 bytes), and CAN messages converted to UDP packets (60

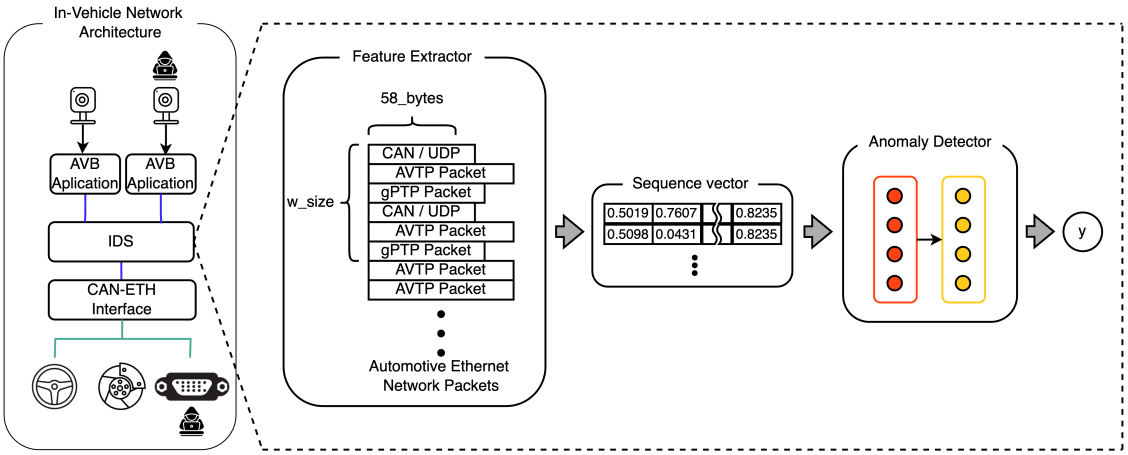



**Figure 2. Block diagram of our proposed IDS main components.**

bytes). To standardize this, we preprocess all packets by selecting the first 58 bytes, as evaluated in [Jeong et al. 2021], which demonstrated that these bytes contain sufficient information, including headers and the initial part of the payload, to detect attacks effectively across protocols. These bytes are represented as decimal values between 0 and 255. To ensure consistent feature scaling and improve model performance during training and inference, we normalize them to a 0–1 range by dividing by 255 [Han et al. 2023].

For the Anomaly Detector, we employ a deep learning algorithm widely recognized for identifying intrusions and anomalies in network traffic [Alkhatib et al. 2023], [Alkhatib et al. 2021]. Unlike typical models, our approach uses an unsupervised Seq2seq model, originally developed for tasks like text translation between languages [Sutskever et al. 2014], but adapted to reproduce input sequences as output. This adaptation enables the model to compress and reconstruct input data, learning the underlying normal patterns in the process. By training exclusively on normal sequences, the model becomes proficient at accurately reconstructing benign data. However, when malicious packets, which deviate from the learned patterns, are introduced, the model struggles to compress and reproduce them effectively. This reconstruction discrepancy serves as an effective mechanism for detecting attacks in the data.

Figure 3 and Table 3 outline the architecture of SeqWatch. The input to the Encoder has a shape of $(w\_size, 58)$, where 58 represents the number of bytes provided by the feature extractor. We use the hidden and cell states from the LSTM Encoder's output to generate the sequence. The sequence is generated recursively by calling the Decoder with the previous hidden state, cell state, and output. During the first call, the Decoder receives the hidden and cell states from the Encoder, along with the first packet in the sequence, $\mathbf{p}_0$. The output of the LSTM Decoder is then passed through a Dense Layer to produce the final output. The output shape of the Decoder is $(1, 58)$, where 1 indicates that the Decoder generates one packet at a time.

We rely on the reconstruction loss approach, widely used in autoencoder-based anomaly detection [Alkhatib et al. 2022], by calculating the Mean Squared Error (MSE) between the input sequence $S$ and the generated sequence $\hat{S}$. If the MSE is higher than a certain threshold $t$, the sequence $S$ is marked as malicious because it deviates from

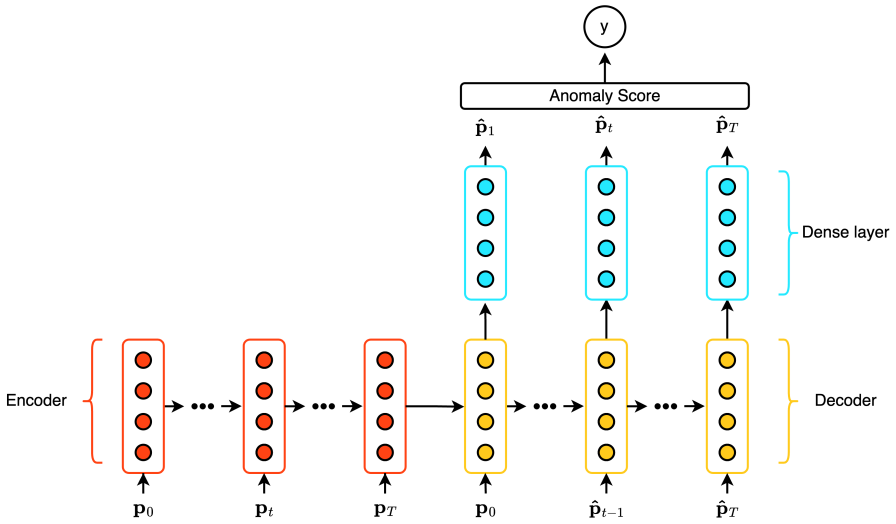



**Figure 3. Anomaly Detector Architecture based on Seq2seq model**

the normal pattern. We are using Youden's index technique to define the threshold $t$ to minimize the True Positive Rate (TPR) and minimize the False Positive Rate (FPR).

Our approach improves upon the limitations of the LSTMAE by focusing on the detailed relationships between consecutive packets. Unlike LSTMAE, which compresses an entire sequence into a single representation for reconstruction, potentially losing critical temporal dependencies, our method generates each packet in a sequence using the previously generated packet as input. This packet-by-packet generation captures the conditional probability of each packet given the prior ones, allowing the model to learn detailed patterns in packet sequences. Additionally, by training exclusively on normal data, the model becomes highly specialized in recognizing typical behavior, making even small deviations in sequence patterns more apparent. This step-by-step approach enhances sensitivity to irregularities and provides a more robust mechanism for detecting anomalies compared to the broader reconstruction focus of LSTMAE. As a result, our method is particularly effective in automotive network traffic, where temporal relation and content patterns are critical, and minor deviations can indicate malicious activity.

**Table 3. SeqWatch Architecture (hidden_size=hs)**

| Block | Layer | Dimensions | Num_layers | Dropout |
|---|---|---|---|---|
| Input | - | (w_size, 58) | - | - |
| Encoder | LSTM | (1, hs), (1, hs) | 2 | 0.5 |
| Decoder | LSTM | (1, hs) | 2 | 0.5 |
| | Linear | (1, 58) | - | - |
| Output | - | (w_size, 58) | - | - |

## 5. Method and Experimental Evaluation

In this section, we describe the datasets used for training and evaluation. We utilized the framework provided in [da Luz et al. 2024] and enhanced it by incorporating our model, baseline models, and a new dataset. These improvements make the framework more versatile and accessible, enabling researchers to easily reproduce results and experiment

with different models. The enhanced framework is available at Automotive IDS Evaluation Framework[1]. We used Python and PyTorch for development because they are widely used, well-documented, and have strong community support for machine learning and deep learning projects. All experiments, including training, validation and testing were performed on an Intel(R) Core(TM) i9-13900K CPU @ 5.80 GHz and an NVIDIA GeForce RTX 4090 GPU.

### 5.1. Dataset presentation

The datasets used to evaluate our proposed IDS are the TOW-IDS dataset [Han et al. 2023] and the SAD [Han et al. 2018]. The TOW-IDS dataset was chosen to demonstrate the IDS's generalization capability across various domains and to facilitate comparisons with existing works on automotive Ethernet intrusion detection, such as [Alkhatib et al. 2022]. The SAD dataset, on the other hand, was selected because it was captured in a real vehicle environment, providing authentic automotive data. Additionally, it contains the types of attack discussed in Section 3, such as CAN replay and malfunction attacks, which are particularly challenging to detect. This makes the SAD valuable for validating our approach against complex real-world attack scenarios.

The TOW-IDS dataset contains automotive Ethernet traffic captured in .pcap files, featuring five attack types targeting three protocols: AVTP, CAN over UDP, and gPTP. The attacks included in the dataset are CAN DoS, CAN replay, frame injection, MAC flooding, and PTP sync attacks. Each packet is labeled with its corresponding attack type, if applicable. In addition, the SAD dataset contains CAN protocol data collected from genuine traffic across three car models: Hyundai Sonata, Kia Soul, and Chevrolet Spark. The preliminary release includes data on fuzzing and malfunction attacks for all three models, while the second release focuses on replay and malfunction attacks for the Hyundai Sonata and Kia Soul. Since our study targets replay attack detection, we utilized the second release and concentrated on the Hyundai Sonata for evaluation. Narrowing the scope to a single car model streamlined the analysis process and allowed us to validate the IDS's performance on a representative dataset, leaving the evaluation of additional car models for future work.

To validate and test our proposed IDS, we required labeled datasets for the feature extraction process. For the TOW-IDS dataset, we adopted the labeling criteria presented in [da Luz et al. 2024], where a sequence of packets is labeled as malicious if it contains at least one malicious packet. The sequence label corresponds to the most frequent attack type within it, given the presence of multiple attack types in a single sequence. For the SAD dataset, each file contains data from a single attack type. Therefore, sequences were labeled with the attack type if at least one malicious packet was present within the group. The distribution of both datasets is summarized in Tables 4 and 5.

### 5.2. Experimental setup

The experimental evaluation of our IDS consists of two phases: training-validation and testing. During the training-validation phase, the training dataset was split into a training set and a validation-test set. The training subset contained only benign sequences, as our

[1] https://github.com/MauricioSight/automotive-ids-evaluation-framework

**Table 4. Class distribution of sequences after labeling in the TOW-IDS dataset.**

| Class | Train | Validation | Test |
|---|---|---|---|
| Normal | 257,732 (100.00%) | 257,733 (42.83%) | 421,237 (53.22%) |
| PTP sync | 0 (0.00%) | 96,858 (16.09%) | 76,404 (9.65%) |
| Frame injection | 0 (0.00%) | 54,682 (9.09%) | 53,025 (6.70%) |
| MAC flooding | 0 (0.00%) | 52,279 (8.69%) | 50,830 (6.42%) |
| CAN replay | 0 (0.00%) | 50,856 (8.45%) | 103,182 (13.04%) |
| CAN DoS | 0 (0.00%) | 89,397 (14.85%) | 86,805 (10.97%) |

**Table 5. Class distribution of sequences after labeling for the SAD dataset.**

| Class | Train | Validation | Test |
|---|---|---|---|
| Normal | 182,277 (100.00%) | 182,277 (38.36%) | 219,855 (39.63%) |
| Replay | 0 (0.00%) | 170,703 (35.93%) | 171,410 (30.90%) |
| Malfunction | 0 (0.00%) | 122,158 (25.71%) | 163,519 (29.47%) |

anomaly detector uses unsupervised learning. Early stopping with a patience of 5 was applied. The training set was further divided into training-train and training-validation subsets. The training-train subset was used for model training, while the training-validation subset monitored training progress. If no improvement was observed for five consecutive epochs, training was stopped. Finally, the model was evaluated on the validation-test set, which included malicious sequences.

We calculated the anomaly score as the mean squared error between the input sequence $S$ and the generated sequence $\hat{S}$. Using these scores, we computed the Receiver Operating Characteristic Area Under Curve (ROC AUC) and determined the threshold $t$ using Youden's index technique, which optimizes the trade-off between TPR and FPR as described in Algorithm 1.

During the validation-test phase, we fine-tuned key hyperparameters—$hidden\_size$, $learning\_rate$, $batch\_size$, and $w\_size$—to optimize model performance based on validation-test results. This hyperparameter tuning was conducted using the TOW-IDS dataset, where a window size ($w\_size$) of 128 yielded the best results. Additionally, we found that setting $hidden\_size = 16$, $learning\_rate = 0.001$, and $batch\_size = 32$ provided optimal training conditions for our anomaly detection model. In the subsequent testing phase, we evaluated the model on the test set using the threshold $t$ defined during training-validation, simulating how the IDS would perform on unseen data in a real-world scenario.

**Algorithm 1** Calculate Youden Index and Optimal Threshold

**Require:** $y_{true}$: Ground truth labels, $y_{pred}$: Predicted anomaly scores
**Ensure:** $t$: Threshold maximizing TPR and minimizing FPR
1: Compute FPR, TPR, and thresholds using ROC curve:
$fpr, tpr, thresholds \leftarrow \text{roc\_curve}(y_{true}, y_{pred})$
2: Get Youden's index:
$youden\_index \leftarrow \arg\max(tpr - fpr)$
3: Get the optimal threshold:
$t \leftarrow thresholds[youden\_index]$
4: **return** $t$

## 6. Results and Discussion

In our experiments, we evaluated the detection rate and model performance of our proposed IDS across the five attacks considered in our threat model. We compared our solution's results to those of two other state-of-the-art automotive Ethernet unsupervised IDSs [Alkhatib et al. 2022].

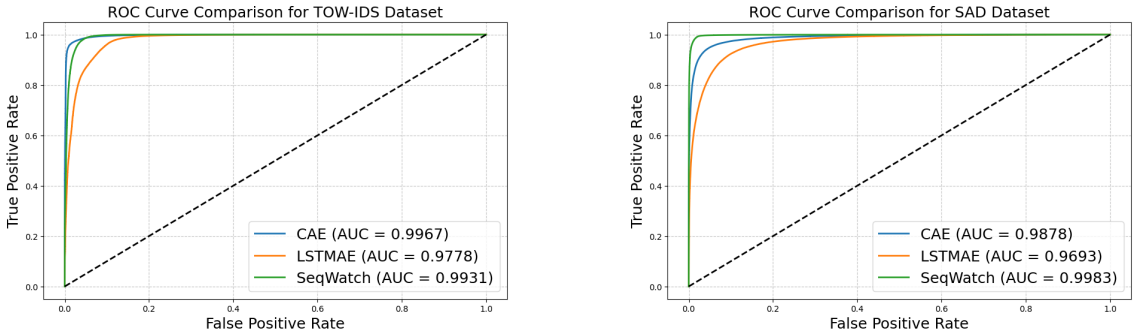
When examining the overall metrics for the TOW-IDS dataset in Table 6, our proposed IDS achieved the highest F1-Score (0.99147), demonstrating balanced detection performance. Although the CAE model achieved the highest AUC-ROC, it has a lower precision compared to SeqWatch, meaning it is less effective in correctly identifying normal sequences. Nevertheless, SeqWatch, with more balanced precision and recall metrics, achieved a higher F1-Score, making it more effective in real-world scenarios where minimizing false positives is crucial. The ROC curves of the IDS models are shown in Figures 4(a) and 4(b). In the SAD dataset (Table 7), our IDS outperformed the other models across all metrics. This superior performance can be attributed to the strong temporal relationships inherent in the attacks presented in the dataset, such as CAN replay and malfunction attacks 3. These types of attacks exhibit patterns that are effectively captured by SeqWatch, leveraging its strength in modeling temporal dependencies and sequential data.

**Table 6. TOW-IDS dataset detection metrics comparison**

| Model | AUCROC | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|---|
| CAE | **0.99666** | 0.96862 | 0.96648 | **0.99941** | 0.98267 |
| LSTMAE | 0.97784 | 0.95907 | 0.98682 | 0.96694 | 0.97678 |
| SeqWatch | 0.99310 | **0.98484** | **0.99318** | 0.98977 | **0.99147** |

**Table 7. SAD dataset detection metrics comparison**

| Model | AUCROC | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|---|
| CAE | 0.98775 | 0.95013 | 0.96164 | 0.95552 | 0.95857 |
| LSTMAE | 0.96932 | 0.91497 | 0.93088 | 0.92806 | 0.92947 |
| SeqWatch | **0.99832** | **0.98651** | **0.98314** | **0.99471** | **0.98889** |




(a) TOW-IDS dataset ROC Curve comparison.

(b) SAD dataset ROC Curve comparison.

**Figure 4. Comparison of ROC Curves for TOW-IDS and SAD datasets.**

Tables 8 and 9 present the detection rates for the overall normal and attack detection rates as well as for each type of attack across the considered datasets. Notably,

both the CAE and LSTMAE models exhibited significant difficulty in detecting the CAN replay attack in the TOW-IDS dataset, achieving detection rates of 0.82313 and 0.89946, respectively. As described in Section 3, this attack is among the most challenging to detect. Our proposed IDS, however, demonstrated consistent performance across all attacks in the dataset. Another point of concern is the CAE model's low detection rate for normal packets (0.71862), which indicates a high false-positive rate. In a real-world scenario, this would generate an overwhelming number of alerts, making the model impractical for deployment. On the SAD dataset, the CAE model maintained strong performance across all attacks. This is likely because the dataset consists only of CAN packets, making attacks more distinguishable compared to the automotive Ethernet dataset, which includes both CAN over UDP and AVTP packets. On the other hand, the LSTMAE model had a lower detection rate for normal packets (0.89519), also indicating a high false-positive rate, which poses similar challenges.

**Table 8. Test set detection rate for the TOW-IDS dataset.**

| **Model** | Normal | Attack | Frame injection | PTP sync | CAN replay | MAC flooding | CAN DoS |
|---|---|---|---|---|---|---|---|
| CAE | 0.71862 | 0.96954 | 0.95171 | 0.85035 | 0.82313 | 0.82254 | 0.87144 |
| LSTMAE | 0.89519 | 0.91078 | 0.97647 | 0.94426 | 0.89946 | 0.93385 | 0.87065 |
| SeqWatch | **0.94486** | **0.98616** | **0.97945** | **0.97068** | **0.96091** | **0.96523** | **0.97020** |

**Table 9. Test set detection rate for the SAD dataset.**

| **Model** | Normal | Attack | Replay | Malfunction |
|---|---|---|---|---|
| CAE | 0.94193 | 0.95552 | 0.94509 | 0.95058 |
| LSTMAE | 0.89502 | 0.92806 | 0.92448 | 0.89382 |
| SeqWatch | **0.97401** | **0.99471** | **0.98207** | **0.98387** |

The Youden index is used to define the threshold $t$, optimizing the balance between detection performance and false positives. This method maximizes the TPR while minimizing the FPR, improving overall classification. Although known attacks were used to set the threshold on the validation set, the system's potential to detect unknown attacks lies within its design and training strategy: any significant deviation from normal behavior—i.e., an anomaly score above $t$—is flagged as malicious. Since collecting large amounts of labeled attack data is challenging, requiring extensive log analysis and annotation, we train the model solely on normal data, with attack data used only during validation to assess performance.

## 7. Conclusions and Future Work

In this work, we proposed SeqWatch an unsupervised sequence-based IDS for automotive Ethernet, designed to detect malicious activity in heterogeneous environments under complex attack scenarios. Our approach leverages a Seq2seq-based model, which can analyze packet sequences, learn the system's normal patterns, and be trained using unsupervised learning. This capability enables the detection of zero-day attacks.

We evaluated SeqWatch using two publicly available datasets: the TOW-IDS dataset for automotive Ethernet and the SAD dataset for CAN networks, comparing its performance to state-of-the-art IDS approaches. SeqWatch outperformed the CAE and LSTMAE, achieving the highest F1-Scores of 0.99147 on the TOW-IDS dataset and

0.98889 on the SAD dataset. Notably, SeqWatch also demonstrated superior detection of CAN replay attacks, achieving a detection rate of 0.98207. This is significant as replay attacks, characterized by subtle differences from normal packets, are notoriously challenging for conventional IDSs to detect, with competing systems struggling in this area.

Future work will focus on transitioning SeqWatch to a real-time framework by integrating faster and more efficient machine learning algorithms, such as Temporal Convolutional Networks (TCNs), which are well-suited for sequence analysis. These enhancements will enable SeqWatch to achieve real-time or near-real-time detection, increasing its applicability and effectiveness. Additionally, the evaluation will be expanded to include multiple car models in the SAD dataset, providing a more comprehensive assessment of SeqWatch's performance across diverse automotive environments. This will address the current limitation of focusing on a single car model and further validate its robustness against various attack scenarios. Finally, we plan to exclude some attacks from the validation set to better evaluate the model's ability to generalize to fully unseen threats.

## Acknowledgments

This paper was partly supported by CNPq (Grant 312368/2021-6) and is part of the INCT of Intelligent Communications Networks and the Internet of Things (ICoNIoT), supported by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES), Finance Code 88887.954253/2024-00.

## References

Alkhatib, N., Ghauch, H., and Danger, J.-L. (2021). SOME/IP intrusion detection using deep learning-based sequential models in automotive ethernet networks. In *2021 IEEE 12th Annu. Inf. Technol., Electron. Mobile Commun. Conf. (IEMCON)*, pages 0954–0962.

Alkhatib, N., Mushtaq, M., Ghauch, H., and Danger, J.-L. (2022). Unsupervised network intrusion detection system for AVTP in automotive ethernet networks.

Alkhatib, N., Mushtaq, M., Ghauch, H., and Danger, J.-L. (2023). Here comes SAID: A SOME/IP attention-based mechanism for intrusion detection. In *2023 14th Int. Conf. Ubiquitous Future Netw. (ICUFN)*, pages 462–467.

Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., Savage, S., Koscher, K., Czeskis, A., Roesner, F., and Kohno, T. (2011). Comprehensive experimental analyses of automotive attack surfaces. In *20th USENIX Security Symp. (USENIX Security 11)*, San Francisco, CA. USENIX Assoc.

Combs, G. and the Wireshark Contributors (2024). *Wireshark: The World's Foremost Network Protocol Analyzer*. Wireshark Foundation. Version 4.2.0.

da Luz, L. F. M., Freitas de Araujo-Filho, P., and Campelo, D. R. (2024). Multi-stage deep learning-based intrusion detection system for automotive ethernet networks. *Ad Hoc Netw.*, 162:103548.

Dibaei, M., Zheng, X., Jiang, K., Abbas, R., Liu, S., Zhang, Y., Xiang, Y., and Yu, S. (2020). Attacks and defences on intelligent connected vehicles: a survey. *Digit. Commun. Netw.*, 6(4):399–421.

Freitas de Araujo-Filho, P., Kaddoum, G., Campelo, D. R., Gondim Santos, A., Macêdo, D., and Zanchettin, C. (2021). Intrusion detection for cyber–physical systems using generative adversarial networks in fog environment. *IEEE Internet Things J.*, 8(8):6247–6256.

Han, M. L., Kwak, B. I., and Kim, H. K. (2018). Anomaly intrusion detection method for vehicular networks based on survival analysis. *Veh. Commun.*, 14:52–63.

Han, M. L., Kwak, B. I., and Kim, H. K. (2023). TOW-IDS: Intrusion Detection System Based on Three Overlapped Wavelets for Automotive Ethernet. *IEEE Trans. Inf. Forensics Secur.*, 18:411–422.

Jeong, S., Jeon, B., Chung, B., and Kim, H. K. (2021). Convolutional neural network-based intrusion detection system for AVTP streams in automotive ethernet-based networks. *Veh. Commun.*, 29:100338.

Jeong, S., Kim, H. K., Han, M. L., and Kwak, B. I. (2024). AERO: Automotive ethernet real-time observer for anomaly detection in in-vehicle networks. *IEEE Trans. Ind. Informat.*, 20(3):4651–4662.

Jo, H. J. and Choi, W. (2022). A survey of attacks on controller area networks and corresponding countermeasures. *IEEE Trans. Intell. Transp. Syst.*, 23(7):6123–6141.

Matheus, K. and Königseder, T. (2021). *Automotive Ethernet*. Cambridge Univ. Press, [S.l.].

Moussa, B., Kassouf, M., Hadjidj, R., Debbabi, M., and Assi, C. (2020). An extension to the precision time protocol (PTP) to enable the detection of cyber attacks. *IEEE Trans. on Ind. Inform.*, 16(1):18–27.

Nisioti, A., Mylonas, A., Yoo, P., and Katos, V. (2018). From intrusion detection to attacker attribution: A comprehensive survey of unsupervised methods. *IEEE Commun. Surv. Tutor.*, PP:1–1.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks.

Wu, W., Li, R., Xie, G., An, J., Bai, Y., Zhou, J., and Li, K. (2020). A survey of intrusion detection for in-vehicle networks. *IEEE Trans. Intell. Transp. Syst.*, 21(3):919–933.

Zhang, X., Cui, X., Cheng, K., and Zhang, L. (2020). A convolutional encoder network for intrusion detection in controller area networks. In *2020 16th Int. Conf. Comput. Intell. Secur. (CIS)*, pages 366–369.