Toward a real-time intrusion detection system for modern in-vehicle networks

Hellemans, W.; Le Jeune, L.; Rabbani, M.M.; Preneel, B.; Mentens, N.

Toward a Real-Time Intrusion Detection System for Modern In-Vehicle Networks

Wouter Hellemans, *Graduate Student Member, IEEE*, Laurens Le Jeune, Md Masoom Rabbani, Bart Preneel, and Nele Mentens, *Senior Member, IEEE*

***Abstract*— Over the past decade, it has been demonstrated that the In-Vehicle Network (IVN) of a modern Intelligent Transportation System (ITS) is vulnerable to several cyberattacks. Given the collaborative nature of these systems, detecting (remote) cyberattacks is of utmost importance in ensuring trusted interactions. One key technique that has been explored to detect adversarial presence in IVNs are Intrusion Detection Systems (IDSs). However, many existing solutions focus on legacy architectures or are not practically feasible due to their hardware requirements or inability to operate in real-time. To this end, we propose Modular Reduced Temporal Convolutional Network (MR-TCN), an efficient IDS architecture that can effectively be accelerated on hardware to enable real-time intrusion detection in low-cost embedded platforms. Additionally, we evaluate variants of MR-TCN on a Field-Programmable Gate Array (FPGA) platform across a diverse range of IVN traffic (i.e., CAN CC, CAN FD, and Automotive Ethernet), demonstrating its suitability in real-world applications.**

***Index Terms*— In-vehicle network security, intrusion detection, machine learning, FPGA, controller area network, automotive Ethernet.**

I. Introduction

In THE past decades, the transport sector's sustainability has come under severe pressure, underlined by escalating emissions and surges in traffic congestion. To address these issues, countries have committed to promoting Intelligent Transport Systems (ITSs). These systems collect and communicate data and information to improve the safety, efficiency and sustainability of transportation [1]. As part of this trend, modern Connected and Autonomous Vehicles (CAVs) are evolving into collaborative, software-driven, cyber-physical systems. Central to these systems are the Electronic Control Units (ECUs), that control the vehicle by integrating data from internal and external sources. While the advanced software functionalities in modern CAVs have the potential to improve safety, it has also been demonstrated that In-Vehicle Networks (IVNs) can be subject to various cyberattacks [2], [3], [4], [5].

Although several IVN protocols have been proposed for the exchange of sensor, control, and actuator information within vehicles, the Controller Area Network (CAN) [6] has been the predominant choice since its introduction in 1986. Nevertheless, to meet the evolving demands of modern CAVs and to cope with the increasing bandwidth required for collaborative applications, automotive Original Equipment Manufacturers (OEMs) are now exploring alternative protocols for high-speed data communication. These emerging protocols include new generations of the classical CAN CC (i.e., CAN FD [7] and CAN XL [8]), as well as Automotive Ethernet (AE). The term AE encompasses a range of new technologies, including a new physical layer and numerous higher-layer protocols. These innovations enable Ethernet communication to meet the stringent requirements of vehicles, such as time synchronization, multicasting, and message prioritization [9]. Correspondingly, to cope with the increasing number of ECUs (up to 150 [10]), modern CAVs present several architectural advancements. One such advancement is the zonal architecture, in which the ECUs are grouped in a cluster per physical zone in the vehicle. The ECUs within one zone are under the direct control of a zone controller, that can communicate with other zone controllers and with a computationally performant central system over a high-bandwidth AE interface [11].

Despite the recent advances in vehicular networking technology, modern IVNs often lack in security. Additionally, the recently introduced AE inherits the security issues of *traditional Ethernet* networks [12]. Due to the safety implications of cyberattacks on IVNs, both industry and academia have urged the development of sophisticated security solutions. Relevant industrial efforts include the UNECE WP.29 R155 norm [13] and ISO/SAE 21434 standard [14]. These initiatives require the implementation of a cybersecurity management system throughout the life cycle of road vehicles. A pivotal element within such a system is the cyberattack detection mechanism. In this context, the alliance known as AUTomotive Open System ARchitecture (AUTOSAR) provides a framework for automotive Intrusion Detection Systems (IDSs) [15]. In addition to the aforementioned industrial efforts, several studies have proposed security solutions for IVNs, focusing primarily on cryptographic approaches [16] and IDSs [17].

IDSs have been conceived as a promising security solution for IVNs based on their alignment with industrial requirements and their ability to be deployed as an add-on without

Received 20 December 2024; revised 24 May 2025; accepted 9 July 2025. Date of publication 28 July 2025; date of current version 3 November 2025. This work was supported in part by the Cybersecurity Initiative Flanders under Grant VR20192203. The work of Wouter Hellemans was supported by the Research Foundation Flanders (FWO) under Grant 1SH3824N. The Associate Editor for this article was X. Li. *(Corresponding author: Wouter Hellemans.)*

Wouter Hellemans, Laurens Le Jeune, and Bart Preneel are with COSIC, ESAT, KU Leuven, 3000 Leuven, Belgium (e-mail: wouter.hellemans@kuleuven.be; laurensle.jeune@kuleuven.be; bart.preneel@kuleuven.be).

Md Masoom Rabbani is with the Department of Computer Science and Engineering, Chalmers University of Technology, 412 96 Gothenburg, Sweden, and also with the University of Gothenburg, 405 30 Gothenburg, Sweden (e-mail: mdmasoom.rabbani@chalmers.se).

Nele Mentens is with COSIC, ESAT, KU Leuven, 3000 Leuven, Belgium, and also with LIACS, Leiden University, 2311 EZ Leiden, The Netherlands (e-mail: nele.mentens@kuleuven.be).

Digital Object Identifier 10.1109/TITS.2025.3590301

alterations to the network architecture. Notably, existing IDS solutions for IVNs have primarily focused on traditional CAN CC networks. In this domain, multiple detection paradigms have been proposed, such as rule-based, statistical-based, and Machine Learning (ML)-based approaches [18]. Among these solutions, the ML-based IDSs have garnered considerable attention owing to their high performance and ability to adapt to new attacks. While numerous studies have focused on detecting cyberattacks in CAN networks, there is a notable lack of security solutions for AE. This technology necessitates novel approaches, as it cannot directly rely on systems designed for *traditional Ethernet* networks due to the distinct nature of the network traffic. Therefore, to address these issues, we propose a lightweight IDS for CAN and AE that can be integrated within emerging hybrid FPGA-based ECUs (e.g., zone controllers). By accelerating our solution on a Field Programmable Gate Array (FPGA), our IDS is capable of lightweight, real-time intrusion detection in modern high-bandwidth automotive networks. Specifically, we summarize our contributions as follows:

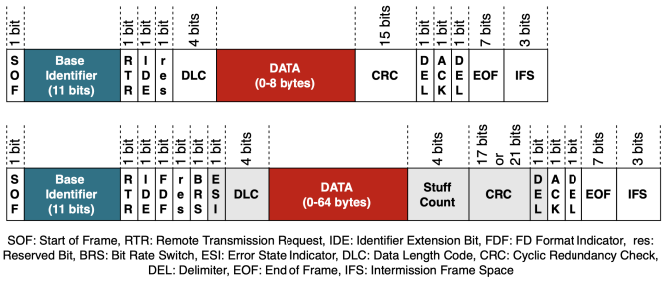
- We propose Modular Reduced Temporal Convolutional Network (MR-TCN), an efficient and flexible ML architecture that can be adapted and scaled for intrusion detection in both AE and CAN scenarios;
- We provide a comprehensive evaluation of MR-TCN on various datasets encompassing relevant automotive protocols, including CAN CC, CAN FD, and AE;
- To the best of our knowledge, we present the first hardware accelerator for real-time intrusion detection in modern AE and CAN FD networks;
- We demonstrate MR-TCN's real-world applicability through a proof-of-concept implementation using an automotive representative FPGA platform.

II. Background

In recent decades, various IVN communication protocols have been developed to facilitate data exchange among the ECUs in vehicles, including Local Interconnect Network (LIN), CAN, FlexRay, Media Oriented Systems Transport (MOST), and AE. Although they served an important role in reducing the wiring cost in vehicles, protocols such as CAN prove to be inadequate for next-generation automotive applications (e.g., autonomous driving). A promising solution to complement these protocols is AE. In this section, we provide the necessary background on CAN and AE. Additionally, we present details on the FINN framework [19], [20], which is used to generate the hardware accelerators for our MR-TCN model.

A. Controller Area Network

The CAN is a signal-oriented IVN communication protocol that features multi-manager, prioritized broadcast communication over a bus topology. Standardized in ISO 11898, CAN CC has a maximum transmission speed of 1 Mbps and allows for transmitting 4 different message types: *error*, *remote*, *overload* and *data*. CAN data frames are identified by an ID and exist in two forms: the base format (11-bit IDs) and the extended format (29-bit IDs). The ID value itself is used to resolve collisions during the arbitration and to indicate the message priority. To avoid collisions, every message ID should only be transmitted by one node, and an acceptance filter in the receiver's controller enables a node to listen to specific messages in a publish-subscribe manner. The payload of CAN CC frames consists of 0-64 data bits, containing the actual signals.[1] To cope with the increasing bandwidths of modern automotive networks, CAN FD enables a bitrate switch to support speeds of up to 8 Mbps during the *data phase*. Additionally, CAN FD increases the payload to a maximum of 64 bytes. Fig. 1 illustrates the structure of a typical CAN CC and CAN FD frame in the base format. It is important to note that the CAN protocol itself lacks in cryptographic security, as it does not offer any form of encryption or data-origin authentication. The CRC is present solely to enhance robustness by protecting against random modifications.



SOF: Start of Frame, RTR: Remote Transmission Request, IDE: Identifier Extension Bit, FDF: FD Format Indicator, res: Reserved Bit, BRS: Bit Rate Switch, ESI: Error State Indicator, DLC: Data Length Code, CRC: Cyclic Redundancy Check, DEL: Delimiter, EOF: End of Frame, IFS: Intermission Frame Space

Fig. 1. Structure of a CAN CC frame (top) and a CAN FD frame (bottom) in base format.

B. Automotive Ethernet

To cope with the increasingly stringent requirements of modern vehicular applications, there has been a growing demand to bring Ethernet to IVNs. The Ethernet technology, renowned for its high bandwidth, scalability and flexibility, offers a promising alternative to traditional control networks with the potential to unify in-vehicle communication. However, the best-effort service model of *traditional Ethernet* poses reliability issues which inhibit its direct application in vehicles. As such, AE aims to address these issues by extending on *traditional Ethernet* to guarantee a certain Quality of Service (QoS). One key advancement of AE lies in its physical layer standards. Notably, 100BASE-T1 [21], initially introduced as Open Alliance BroadR-Reach, enables full duplex communication over a single unshielded twisted pair at speeds of 100 Mbps.

Next to new physical layer standards, AE introduces several higher-layer protocols to support the inherent requirements of vehicles, including low-latency, time-sensitive and prioritized communication [22]. These protocols typically rely on Virtual Local Area Network (VLAN) technology to enable virtual segmentation of the vehicle in multiple domains. Additionally, VLAN supports the allocation of packet priorities, which can reduce the latency of critical messages. Prominent higher-layer

[1]How the signals should be interpreted is determined by the OEM and is often kept confidential in CAN Database (DBC) files.

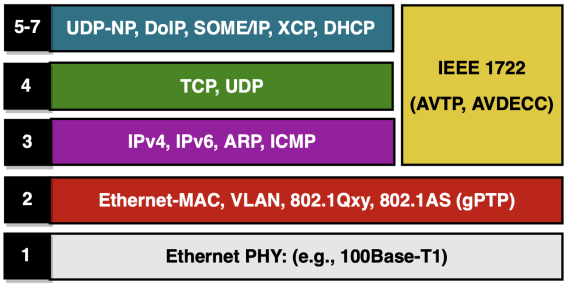


Fig. 2. Automotive Ethernet protocols.

protocols for AE include the Audio Video Bridging (AVB) standards for time-sensitive networking, which enable the reliable transmission of time-critical streaming data. These standards comprise several protocols including Stream Reservation Protocol (SRP), generalized Precision Time Protocol (gPTP), Forwarding and Queuing for Time-Sensistive Streams (FQTSS), and IEEE 1722 Audio Video Transport Protocol (AVTP) [22]. AVTP is a protocol that operates between an *AVB talker* and *AVB listener(s)* (with *AVB bridges* – AVB capable switches – in between) that allows for the transmission of time-sensitive data streams (e.g., video or audio), as well as control packets (e.g., tunnelled CAN frames). To guarantee the timely delivery of data within the IVN, the AVB standards specify the gPTP network-based time-synchronization protocol. gPTP is a specific PTP profile that performs synchronization of all endpoints in the AVB network [12].

In addition to the aforementioned AVB standards, there exist various AE protocols that run on top of Transmission Control Protocol (TCP) or User Datagram Protocol (UDP). One such protocol is Scalable service-Oriented MiddlewarE over IP (SOME/IP), which allows for service-oriented data transmission. This paradigm implies that data is only transmitted when it is needed by at least one receiver through a notification or upon request through a so-called Remote Procedure Call (RPC) [23]. Another relevant example is Diagnostiscs over IP (DoIP), which is an *extended transport protocol* that supports the transmission of several diagnostic services (e.g., Unified Diagnostics Services (UDS)). Furthermore, the flexible nature of Ethernet also allows for custom protocols such as tunnelling of CAN messages over UDP [12]. In Fig. 2, we provide an overview of relevant AE technologies.

C. Hardware Acceleration in FINN

Developed and maintained by the Integrated Communications and AI Lab of AMD Research & Advanced Development, FINN [19], [20] is an open-source experimental framework that explores the acceleration of Quantized Neural Networks (QNNs) on FPGA. Although the tool was originally developed for Binarized Neural Networks (BNNs), FINN currently also supports QNNs of arbitrary precision. The FINN flow[2] starts from a quantized model with few-bit weights and activations in QONNX [24] format. One framework that allows for the training and exporting of neural networks in QONNX format is Brevitas [25], a Python library for Quantization Aware Training (QAT) on top of PyTorch. The FINN tool ingests the QONNX graph and performs a series of transformations to convert the model to a representation that can be accelerated in hardware. First, the tool performs streamlining operations to eliminate non-synthesisable floating point operations. Second, FINN converts the graph into an abstract hardware representation whose nodes can later be mapped to specific hardware implementations. Third, the layers that can be accelerated in hardware are grouped into a dataflow partition and are specialised into synthesizable High-Level Synthesis (HLS) or Register-Transfer Level (RTL). Finally, FINN invokes the Vivado/Vitis toolchain to perform synthesis, implementation and bitstream generation to produce an FPGA accelerator.

In the FINN framework, neural network accelerators are implemented as heterogeneous streaming dataflow architectures with per-layer engines. Samples are streamed through this architecture in a pipelined fashion. To further enhance the throughput of the network, every layer can be parallelized (folded) by changing the number of Processing Elements (PEs) and Single Instruction, Multiple Data (SIMD) lanes. However, as the folding configuration has a profound impact on the resource utilization and strongly depends on the architecture of the model, the PE and SIMD values should be meticulously selected to optimize the trade-off between latency/throughput and resource utilization. As the generated accelerators are highly efficient and characterized by high throughput and low latency, FINN is well suited for high-speed applications such as real-time intrusion detection [26].

III. Related Work

ITSs rely on vast amounts of data to enable collective and cooperative decision-making. This surge in data presents challenges across several domains, including wireless transmission [27], real-time processing [28], and Vehicle-to-Everything (V2X) communication [29]. Moreover, while the increased digitization enhances vehicle safety and efficiency through advanced automation and connectivity, it also introduces potential vectors for cyberattacks [30]. Numerous studies have shown that the in-vehicle networks of modern vehicles are susceptible to cyberattacks [2], [3], [4], [5]. One key approach that has been introduced to detect these cyberattacks is network intrusion detection, which analyzes the network traffic to identify potentially malicious events. The concept of network intrusion detection is a well-studied field of research in traditional computer networks [31], [32], [33], [34]. However, it has been shown that vehicular traffic has distinct properties that necessitate custom solutions. In this section, we provide an overview of relevant ML-based IDSs for CAN and AE networks.

A. Controller Area Network

Several articles have explored the application of ML-based intrusion detection for automotive CAN CC networks. Related work can be broadly categorized into: *ID-based*, *payload-based*, and *frame-based* [35] solutions.

[2]Please note that the flow differs for older versions of the FINN (< v0.10.0).

- **ID-based** solutions monitor sequences of CAN IDs to detect attacks [36], [37], [38], [39], [40], [41]. These systems leverage the fact that many attacks on CAN networks involve message injection or dropping. As such, they disrupt the normal transmission cycles[3] of the messages and thereby alter the ID sequences. However, these approaches have difficulties with protecting against attacks that alter the payload of the messages (e.g., spoofing attacks).
- **Payload-based** solutions consider sequences of CAN payloads for the detection [42], [43], [44], [45], [46]. Those solutions exploit the fact that CAN typically follows a time-triggered communication model. As a result, many messages have payloads that show minimal variation between successive cycles of the same message. Moreover, it has been shown that deep learning models are able to interpret the raw bytes of the unencrypted CAN payloads. Since many of these solutions group the payloads per ID, they are often unable to capture dependencies between messages with distinct IDs.
- **Frame-based** solutions analyze both the IDs and payloads of the CAN messages [47], [48], enabling them to capture the spatial and temporal information embedded in the CAN data. Additionally, frame-based solutions may incorporate additional features such as the timestamp, DLC, or inter-message delay [49], [50], [51]. As a result, those solutions have been conceived a promising solution for detecting intrusions in CAN networks. Given their potential to detect a wide range of threats, MR-TCN adopts the Frame-based detection strategy.

Moreover, due to the high computational and memory demands of some IDSs, existing work typically suggested the deployment of specialized ECUs [52] or accelerators (e.g., GPUs) [36], [37] to speed up the task of intrusion detection. However, the adoption of such systems is challenged by high costs and energy consumption. More recently, mechanisms such as [53], [54], and [55] have considered the acceleration of IDSs on FPGA. Those solutions significantly increase the throughput of the IDS and enable real-time intrusion detection while offering reconfigurability after deployment.

B. Automotive Ethernet

In the context of *traditional Ethernet* networks, numerous IDS traffic datasets have been employed [56], with recent work predominantly utilizing the UNSW-NB15 [57] and CIC-IDS2017 [58] datasets. These datasets, and most other modern traffic datasets, use traffic flows to make up individual samples. A flow aggregates individual packets that belong together, typically through a quintuple (source address, destination address, source port, destination port, transport layer protocol) in Internet Procol v4 (IPv4)/TCP or IPv4/UDP packet series [56]. However, AE does not necessarily allow for a similar approach. AE features traffic that is not always based on IPv4 or TCP/UDP. For instance, AVTP packets have an Ethertype of $0\times22F0$ and do not involve IPv4 or TCP/UDP. As such, approaches that are common in traditional network intrusion detection might not be directly applicable for AE. Furthermore, the development and evaluation of IDSs for AE necessitates the introduction of datasets with AE-specific protocols.

The first dataset for intrusion detection in AE networks was introduced by Jeong et al. [22] and focuses on injection (replay) attacks in AVTP streams. In this work, the authors further present a 2D Convolutional Neural Network (CNN)-based IDS that operates on input samples consisting of the first 58 bytes of 44 consecutive messages. To improve the detection time, Carmo et al. [59] utilize the same feature construction method and propose a fast XGBoost-based IDS. Similarly, da Luz et al. introduce in [60] a pruned CNN model that leverages the LilNetX framework to simultaneously optimize the storage size, detection time and detection metrics during training. They further improve this result in [61] by integrating the pruned CNN in a two-stage IDS. The first stage consists of a Random Forest (RF) classifier acting as a fast attack detection model, while the second stage employs the pruned CNN to reduce false positives and serves as an attack identification mechanism. Apart from the previous supervised methods, Alkhatib et al. [62] explore unsupervised learning using different CNN- and Long Short-Term Memory (LSTM)-based autoencoder models.

Modern AE networks are heterogeneous networks operating across multiple protocols. Hence, datasets that are used for the development of automotive IDSs should reflect this heterogeneous nature. To overcome the single-protocol limitation of previous work, TOW-IDS [12] proposes a dataset that incorporates AVTP, gPTP, and tunnelled CAN traffic. Furthermore, the authors investigate 2D discrete wavelets as a data reduction technique and implement a ResNet-based model that operates on samples comprising the first 452 bytes of 452 consecutive network packets. To limit the computational demands, the authors in [63] have combined wavelets with a Swin Transformer architecture. Additionally, recent work has explored unsupervised learning based on 1D convolutional autoencoders [9] and an LSTM-based sequence-to-sequence model [64], as well as self-supervised learning with Generative Adversarial Network (GAN)-based data augmentation [65].

Our proposed technique differs from existing work by leveraging hardware acceleration to achieve real-time intrusion detection in emerging IVNs. Additionally, our solution incorporates a lightweight model which eliminates the need for feature preprocessing, making it suitable for deployment in automotive ECUs. Moreover, in contrast to most of the related works, MR-TCN takes into account the protocol heterogeneity of modern IVNs. As such, it does not only consider several AE technologies, but also provides a Frame-based solution for CAN CC and CAN FD. Finally, as ITSs require a scalable and flexible solution [66], MR-TCN is modular and can easily be adapted or scaled to support a certain bandwidth.

IV. Our Solution: MR-TCN

In this section, we present our solution, MR-TCN. Additionally, we outline the system and threat model and provide details on the datasets considered in this paper.

[3]Most messages in CAN networks are transmitted with a fixed period.

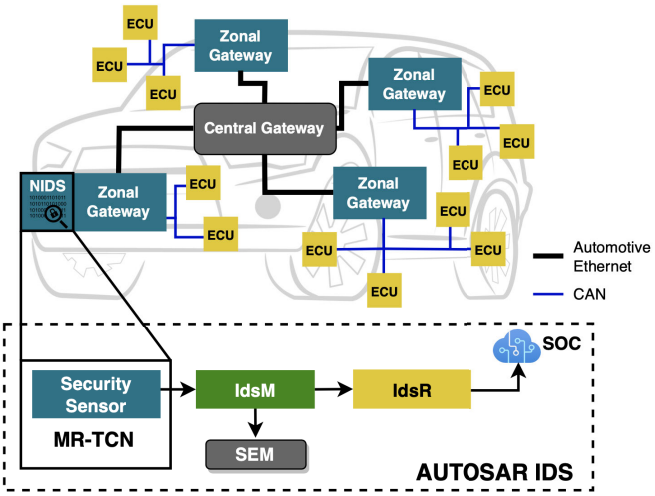


Fig. 3. System model and integration within the AUTOSAR IDS architecture.

A. System Model

In this work, we assume an IVN that utilizes the emerging zonal architecture depicted in Fig. 3. In such an architecture, the ECUs are clustered based on their physical location within the vehicle. Each group of ECUs within a specific zone of the vehicle is managed by a Zonal Gateway, that is connected to a Central Gateway. We consider the Zonal Gateways to be implemented as emerging hybrid FPGA-based ECUs, with our IDS realized in the reconfigurable fabric of the FPGA. These types of ECUs have shown great promise in recent years since they allow for the delegation of time- or security-critical tasks (e.g., ADAS) to dedicated hardware circuits.[4,5] Its strategic position in the network topology gives MR-TCN access to both the CAN and AE network branches, enabling real-time monitoring of all critical traffic in that particular zone. In correspondence with the heterogeneous nature of modern IVNs, we adapt and evaluate MR-TCN in both CAN and AE networks.

Moreover, we highlight that MR-TCN is compatible with existing industrial frameworks, such as AUTOSAR IDS [15], which provides a standardized framework for automotive IDSs. Within this framework, MR-TCN would function as a Security Sensor, detecting and reporting Security Events to an IDS Monitor (IdsM), as illustrated in Fig. 3. The IDS Monitor processes these events by buffering and filtering them into Qualified Security Events. Subsequently, it communicates the qualified events to an IDS Reporter (IdsR), and optionally stores them locally on the ECU in a Security Event Memory (SEM). Finally, the IDS Reporter communicates these qualified events to a Security Operations Center (SOC), where security experts can assess and respond to potential threats.

B. Threat Model

Over the last decade, it has been highlighted that modern IVNs can be subject to various attacks acting on a variety of protocols. In line with [9], [67], and [68], we consider (1) a physical adversary ($\mathcal{A}_{PHY}$) that is able to infiltrate the IVN through the On-Board Diagnostic (OBD)-II port, and (2) a (remote) software adversary ($\mathcal{A}_{SW}$) that is able to compromise one or multiple ECUs. By exploiting the compromised ECUs, the adversary can suspend the node, or inject arbitrary frames in the IVN. Consequently, the adversary can compromise or take control of critical vehicle functions. Note that adversaries with low-level bus access [69], [70] are considered out of scope. To protect against these low-level attacks, our proposed IDS can be complemented with existing technologies such as [71] and [72]. Based on this adversary model, in the following, we provide a high-level overview of relevant attacks on CAN and AE networks that are considered in this paper:

- **Media Access Control (MAC) Flooding.** In a MAC flooding attack, the adversary sends a large number of AE network packets with maliciously crafted source MAC addresses. These messages will eventually trigger overflow in the Content Addressable Memory (CAM) table of the switches, effectively turning them into hubs. As a result, all traffic is transmitted in broadcast, enabling the adversary to sniff the network communication.
- **PTP Sync.** A PTP sync attack disrupts the synchronization process between the clocks of networked devices. By flooding the network or manipulating the PTP sync messages, the adversary can cause a discrepancy in timing between time transmitter and time receiver, making synchronization impossible. A PTP sync attack can severely impact time-critical AVB communication, which relies on precise time synchronization among all communicating devices.
- **AVTP Frame Injection.** Frame injection attacks involve injecting malicious MPEG frames in an AVTP video stream. In an autonomous driving setting, these frames cause discrepancies in the recognition of surrounding objects, resulting in malfunctions of the vehicle.
- **CAN Denial of Service (DoS).** During a CAN DoS attack, the adversary floods the CAN bus with high-priority messages, causing critical messages to be dropped or delayed. The goal of this attack is typically to compromise the availability of the system.
- **CAN Replay.** CAN replay attacks involve retransmitting authentic CAN messages that were previously captured during a reconnaissance phase. As these messages are out-of-context, they can trigger malfunctions in the vehicle. Additionally, the captured sequences can be used as a basis to craft CAN spoofing or masquerading attacks.
- **CAN Fuzzing.** In a CAN fuzzing attack, the adversary injects messages with a randomized payload and/or ID. These attacks are often part of a reconnaissance phase where the adversary attempts to identify the bits responsible for critical functionalities.
- **CAN Spoofing.** In a CAN spoofing attack, the adversary injects messages with a targeted ID and a maliciously crafted payload on the CAN bus. The goal of this attack is typically to cause malfunctions or to take over critical functions of the vehicle (e.g., RPM or gear values). To enhance the stealthiness of CAN spoofing attacks, the adversary can leverage flam delivery [73], where

[4]https://www.intel.com/content/www/us/en/automotive/products/programmable/overview.html

[5]https://www.xilinx.com/publications/prod_mktg/Automotive_brochure.pdf

the malicious message is transmitted immediately after the benign message. This spoofing method causes an overflow in the RX buffers of the receiving controller, effectively overwriting the benign message.

- **CAN Masquerading.** A CAN masquerading attack is a combined attack to increase the stealthiness of attacks such as CAN spoofing. In a masquerading attack, the adversary turns a *weakly compromised* ECU in the bus-off state and continues its communication from a *strongly compromised* ECU under the control of the adversary. By doing so, the receiving node only receives the malicious messages, mitigating the message conflicts of traditional spoofing attacks.

It should be noted that modern vehicles often tunnel CAN messages over AE. Hence, the aforementioned CAN attacks can also manifest themselves in the form of tunneled AE attacks.

C. Datasets and Preprocessing Procedure

To train and evaluate MR-TCN across a diverse range of IVN traffic, we consider several IDS benchmark datasets targeting both CAN and AE. In the following we provide an overview of the datasets considered in this work, as well as a description of the preprocessing procedure.

For AE, we consider the TOW-IDS [12] dataset. This dataset comprises multiple AE protocols and presents five distinct attack profiles: MAC flooding, PTP sync, AVTP frame injection, CAN replay, and CAN DoS, with the last two being tunneled over UDP. The traffic data is provided as network captures in pcap format, accompanied by separate CSV files indicating the label for each packet.

Additionally, to evaluate MR-TCN on CAN CC traffic, we utilize the HCRL-CH [74] and ROAD [73] datasets. The HCRL-CH dataset has been most frequently used dataset for IVN intrusion detection in the literature [35], serving as a benchmark for comparing our solution to other work. This dataset contains Fuzzy (Fuzzing), DoS, and Spoofing (gear and RPM) attacks. However, recent work has highlighted shortcomings of this dataset, including the unstealthy nature of the attacks [75]. Therefore, to enhance the applicability of MR-TCN, we also evaluate our solution on the state-of-the-art ROAD dataset. The ROAD dataset encompasses more stealthy attacks, including Fuzzing, Fabrication (Spoofing), Masquerading, and Accelerator.[6] Lastly, we evaluate our work on the recent CAN FD [76] dataset containing Fuzzing, Malfunction (Spoofing), and Flooding (DoS) attacks. This enables the performance evaluation of MR-TCN on emerging CAN networks featuring higher bandwidths and larger payloads.

1) Automotive Ethernet: We construct samples by parsing the pcap files and modeling the raw network traffic as a multivariate time series. Specifically, we construct input sequences by arranging the first n bytes of l consecutive messages into a $n \times l$ matrix, where each value corresponds to one byte.

[6]Please note that the "Accelerator" attack in the ROAD dataset is not considered in this paper due to the lack of proper labels for this attack. The "Accelerator" attack places the vehicle in an invalid mode, and the authors label all samples in this mode as anomalous. Moreover, the traces only include traffic from the invalid state and do not include the attack itself.

TABLE I
TOW-IDS DATASET COMPOSITION AFTER PREPROCESSING WITH WINDOW SIZE 64

	Stepsize 1		Stepsize 64	
Class	Train	Test	Train	Test
Normal	550629	438331	8604	6847
MAC Flooding	104427	50765	1632	793
PTP Sync	193456	76274	3023	1192
AVTP Frame Injection	74784	36320	1169	569
CAN DoS (tunneled)	178729	86740	2792	1356
CAN Replay (tunneled)	101648	103117	1588	1611

As the TOW-IDS dataset contains packets ranging in size from 60 to 434 bytes, packets shorter than n bytes are zero-padded, whereas packets longer than n bytes are truncated. To balance model complexity and classification performance, we set both n and l to 64 guided by previous work [63] and preliminary experiments. Moreover, by selecting 64 bytes per network packet, we predominantly focus on header bytes while including only a limited number of payload bytes. We limit the number of payload bytes in our features to limit the dependency on potentially encrypted data, and additionally reduce the model complexity. We label a sequence as anomalous if it contains at least one attack packet.

The use of raw bytes eliminates the need for extensive feature preprocessing, which can inhibit real-time intrusion detection. To narrow the range of the variables during the software experiments, we employ min-max feature scaling. This aids in improving numerical stability and facilitates faster convergence. In the hardware experiments, we perform zero-centering of the input samples, mapping them from range $[0; 255]$ to range $[-128; 127]$. Please note that the 8-bit input quantization of the raw bytes significantly reduces the input size compared to floating point inputs without any loss of information. Additionally, the zero-centering can efficiently be achieved at line rate in hardware by toggling the most significant bit using an XOR operation.

Due to the limited number of resulting samples, we explore the use of sliding windows as dataset augmentation during training in line with [77]. In a sliding window, the window of l consecutive packets used to construct the input sample is advanced with step size $\delta \leq l$, giving the model multiple contexts to learn from and increasing the number of training samples. Since the authors of [12] provide independent curated training and test sets, we generate the windows on the respective set. We note that 30% of the training data is used as a validation set. Table I summarizes the attacks and number of samples per attack after our preprocessing procedure for windows of size 64 with step sizes 1 and 64.

2) Controller Area Network: Similar to the AE case, we construct input sequences based on the raw bytes of the network traffic. However, instead of selecting the n first bytes of every message, we combine the ID and DATA fields of l consecutive messages. The CAN messages are typically unencrypted, allowing MR-TCN to capture relevant information from the DATA field. Since the CAN ID can take 29 bits in the extended format, we represent the ID as 4 bytes with zero-padding. Please note that, although frames with an extended

TABLE II
COMPOSITION OF THE CAN DATASETS AFTER PREPROCESSING WITH WINDOW SIZE 64 AND STEP SIZE 64

HCRL-CH	Train	Valid	Test	**ROAD**	Train	Valid	Test	**CAN FD**	Train	Valid	Test
Benign	104982	34801	34868	Benign	242726	80902	80949	Benign	31731	10549	10704
Fuzzing (Fuzzy)	12203	4014	4100	Fuzzing	144	54	43	Fuzzing	11381	3829	3758
Spoofing (Gear & RPM)	37171	12587	12494	Spoofing (Fabrication)	7204	2397	2405	Spoofing (Malfunction)	7283	2429	2452
DoS	10251	3467	3410	Masquerading	6935	2317	2275	DoS (Flooding)	16358	5444	5339

arbitration identifier do not occur in the considered datasets, they are common in modern vehicles. To this end, MR-TCN supports both the base format and the extended format of CAN. Additionally, we represent the DATA as 8 bytes for the case of CAN CC and 64 bytes for CAN FD, with zero-padding as needed. Consequently, the input sequences have as shape 12×64 and 68×64 for CAN CC and CAN FD, respectively.

Since none of the considered CAN datasets provides curated independent training and test sets, we preprocess the samples per trace file separately. Subsequently, after the preprocessing, we concatenate all the samples per class (e.g., all traces containing masquerading attacks) and apply a 60%:20%:20% training:validation:test split per class. An overview of the dataset composition after preprocessing is provided in Table II.

D. MR-TCN Model

The Temporal Convolutional Network (TCN) [78] has been conceived as a promising architecture for the task of sequence modelling. In this context, variations of the architecture have demonstrated remarkable performance on tasks such as weather forecasting [79] and cardiac arrhythmia detection [80]. As vehicular network traffic inherently composes a time-series of network packets, TCNs have also found adoption in vehicular IDSs where they have outperformed other architectures [47], [81], particularly in reducing false positives [82]. Another property that makes TCNs well suited for real-time intrusion detection in IVNs is their high throughput compared to recurrent models such as Recurrent Neural Network (RNN), LSTM or Gated Recurrent Unit (GRU). This owes to the fact that the 1D convolutions can be parallelized, which also allows for efficient hardware acceleration.

TCN networks rely on 1D dilated convolutions to create a receptive field that grows exponentially with the number of layers, as is demonstrated in Fig. 4. This implies that every output value is influenced by many input values, resulting in long memory retention, which in practice is longer than the –theoretically infinite– memory retention of recurrent architectures with the same capacity [78]. Additionally, through zero-padding, the TCN keeps the sequence length constant throughout the layers of the network. The TCN ensures with 1D convolutions that the weight filters are invariant over the time axis. This enables a more intuitive processing, w.r.t. other IDSs leveraging 2D CNNs with 2D input features [26]. Indeed, while neighbouring pixels in an image are related, this may not be the case for reshaped network packets (e.g., a pattern detected in the IP address significantly differs in meaning from the same pattern detected in the MAC address).

In many scenarios, such as prediction tasks, TCNs rely on causal convolutions as the future data is not available.

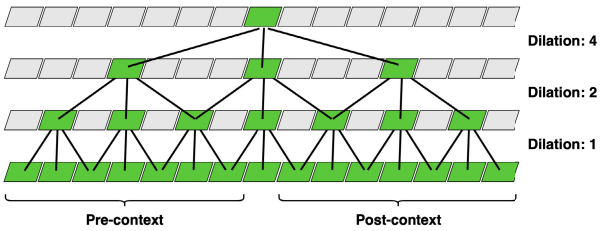


Fig. 4. Dilated non-causal convolutions.

TABLE III
ML MODEL USED IN THE EXPERIMENTS WITH n EQUAL TO 64, 12, AND 68 FOR AE, CAN, AND CAN FD, RESPECTIVELY

Layer	Input	Output	Kernel	Stride	Dilation	Padding
Conv1D	(1,n,64)	(1,64,64)	(3)	(1)	(1)	(1)
Dropout	(1,64,64)	(1,64,64)	-	-	-	-
Conv1D	(1,64,64)	(1,48,64)	(3)	(1)	(2)	(2)
Dropout	(1,48,64)	(1,48,64)	-	-	-	-
Conv1D	(1,48,64)	(1,32,64)	(3)	(1)	(4)	(4)
Dropout	(1,32,64)	(1,32,64)	-	-	-	-
MaxPool1D	(1,32,64)	(1,32,8)	(8)	(8)	-	-
Flatten	(1,32,8)	(1,256)	-	-	-	-
Dropout	(1,256)	(1,256)	-	-	-	-
Linear	(1,256)	(1,1)	-	-	-	-

However, in the context of vehicular data, notably CAN data, recent work has highlighted that providing the model with pre- and post-context can lead to better performance [83]. This can be explained by the periodic character of most CAN messages, while clock drifts in different ECUs can shift the patterns. As such, messages can be delayed, leading to a reorganization of the messages in a window. Using non-causal convolutions, it is possible to mitigate this issue by giving the model both pre- and post-context to learn from. Moreover, the non-causal convolutions allow for hardware acceleration in FINN.

To enable real-time hardware accelerated intrusion detection, we propose a lightweight, redesigned version of the original TCN model, which we call MR-TCN. Table III highlights the structure of our model. To extract high-level features, we leverage 3 temporal blocks consisting of a non-causal dilated 1D convolution followed by a dropout regularizer. This dropout helps to prevent overfitting by randomly deactivating neurons during the training process. Consequently, it helps the model to reduce reliance on any single feature and helps it to learn redundant representations aiding in the generalization. Moreover, to create a large receptive field, we exponentially scale the dilation values to 1, 2, and 4, respectively. Additionally, we reduce the temporal blocks by removing the residual connections and incorporating only a single convolution. Furthermore, we use batch normalization instead of weight normalization as it enables efficient acceleration in the FINN framework. Lastly, we utilize a MaxPool, Dropout, and Linear layer for the final classification.

E. Quantized MR-TCN

When accelerating on custom hardware, resources are typically scarce to limit the cost of the device. As a result, ML models should be made as small as possible before deployment, which can be expedited through quantization. The quantization process involves reducing the numeric representation of weights and activations from 32-bit Floating Point to integer arithmetic with only a few bits. The resulting models have lower computational complexity and require less storage. We opted for a 3-bit quantized model as recent work has demonstrated that this configuration only has a limited impact on the classification performance while keeping the resource consumption at a reasonable level for FPGA acceleration [84].

V. Evaluation

In this section, we evaluate MR-TCN across a diverse range of IVN traffic and analyze its feasibility in real-world scenarios. Specifically, we evaluate its classification performance in software on AE and CAN network traffic. Concretely, we consider the following metrics with TP, FP, TN, and FN denoting the true positives, false positives, true negatives, and false negatives, respectively:

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

$$F_1 = \frac{2TP}{2TP + FP + FN} \tag{2}$$

$$FPR = \frac{FP}{FP + TN} \tag{3}$$

$$FNR = \frac{FN}{FN + TP} \tag{4}$$

Additionally, we deploy MR-TCN on an automotive representative FPGA platform and assess its resource consumption, throughput, latency, and power consumption.

A. Software Evaluation

Before developing a custom hardware accelerator, we first design a Floating Point software variant of MR-TCN. This software variant allows for rapid design space exploration and enables us to assess its classification performance on AE and CAN traffic w.r.t. the state of the art. To train MR-TCN, we leverage the PyTorch framework with our datasets preprocessed as NumPy arrays. In all our experiments, we choose the Adam optimizer and the BCEWithLogits loss function. By doing so, the output activation function is integrated into the loss function, leading to better numerical stability in FINN, which will be used to generate the custom hardware accelerator.

1) Automotive Ethernet: For AE, we explore two variants of MR-TCN: (1) the regular variant that processes the time series as described in Sec. IV-C and (2) a variant that transposes the input. The transpose operation implies that convolving is done over the feature dimension rather than the time dimension, capturing the dependencies between the features at each time step. Additionally, we train and evaluate both a ResNet50 [85] and DWS [26] architecture for comparison. We adjust the input layers to process windows of 64 packets reshaped to $(1 \times 64 \times 64)$ and $(64 \times 8 \times 8)$ images for the DWS and ResNet50 models, respectively.

TABLE IV
Software Results of MR-TCN for AE

Model	Window	Step	Dropout	Acc	FPR	FNR	F1
MRTCN-AE	64	64	0.0	0.8837	0.0070	0.2518	0.8518
MRTCN-AE	64	64	0.55	0.8866	0.0042	0.2489	0.8553
MRTCN-AE^T	64	64	0.0	0.8858	0.0705	0.1684	0.8666
MRTCN-AE^T	64	64	0.55	0.8528	0.1501	0.1435	0.8386
MRTCN-AE	64	1	0.0	0.8890	0.0018	0.2465	0.8584
MRTCN-AE	64	1	0.55	0.8888	0.0010	0.2478	0.8580
MRTCN-AE^T	64	1	0.0	0.9792	0.0200	0.0217	0.9768
MRTCN-AE^T	**64**	**1**	**0.55**	**0.9926**	**0.0039**	**0.0118**	**0.9916**
DWS	64	1	0.4	0.9664	0.0260	0.0429	0.9622
ResNet50	64	1	0.0	0.9593	0.0346	0.0482	0.9543
TOW-IDS [12]	452			0.9965	-	-	0.9974
AERO [9][a]	2048			-	-	-	0.9862
Swin-T [63]	512			0.9982	0	0.0057	0.9974
LSGAN [65][b]	1			0.97	0.01	0.05	0.95
Multi-IDS [61]	45			0.9962	-	-	0.9960
SeqWatch [64][a]	128			0.9931	-	-	0.9889

[a] Unsupervised learning approach
[b] Semi-supervised learning approach

In all AE experiments, the sliding window step size is set to 1 or 64 for training and 64 for testing. The decision threshold is determined using the Youden index [86] and various dropout configurations are explored. For the MR-TCN and ResNet models, we use a learning rate of 10^{-5}, while the DWS model's learning rate is set to 10^{-6}. Moreover, we set the number of epochs to 700 for the MR-TCN models, 1000 for the DWS model, and 300 for the ResNet model with early stopping enabled. Furthermore, we compare our model to recent work. The results of our software evaluation are presented in Table IV for the various configurations.

Our analysis demonstrates that the transposed model with step size 1 and dropout 55% achieves an F1 score of 99.1639% and thereby significantly outperforms the other MR-TCN, DWS, and ResNet50 architectures. However, it is slightly outperformed by the more complex [12] and [63]. Compared to those solutions that leverage windows of size 452 and 512, our model keeps the window size to 64 to limit the model's complexity and buffering latency. Additionally, the sliding window technique during training indeed positively affects the classification performance. Lastly, we note that the dropout has a significant impact with the MR-TCN model requiring high dropout rates. We argue that the transpose operation likely introduces additional regularization, improving the performance of the model.

To investigate the sliding window dataset augmentation in more detail, we evaluate the behaviour of MR-TCN based on the number of attack packets contained in a window. In Fig. 5a and Fig. 5b, we plot the distribution of the attack count relative to the number of detected and missed samples for the MR-TCN model trained with dropout 55% and step sizes 64 and 1, respectively. This analysis highlights that the network with sliding window augmentation effectively detects windows containing $>\approx 10$ attack packets. For windows with fewer attacks (small-scale attacks), the detection rate decreases in line with [9]. However, without the sliding window augmentation, the network fails to adequately train. Even for the frames with many samples, it experiences difficulties in the detection.

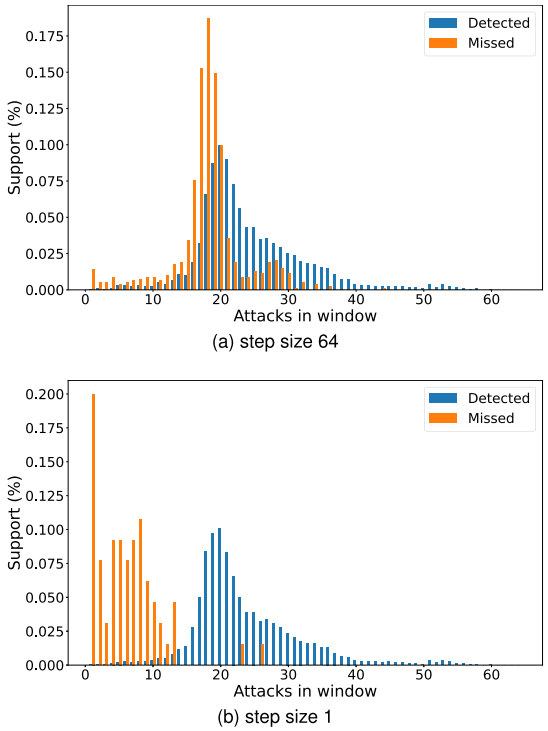


Fig. 5. Attack count distribution relative to the number of detected an missed samples.

2) Controller Area Network: We now assess the performance of MR-TCN on a second category of IVN traffic by adapting the model to process CAN messages. Specifically, we change the input layer of the model to take as input sequences of shape (12×64) and (68×64) for CAN CC and CAN FD, respectively. Additionally, similar to the AE experiments, we compare MR-TCN against the DWS architecture processing windows of 64 CAN messages as $(12 \times 8 \times 8)$ and $(68 \times 8 \times 8)$ images. Note that for our CAN evaluation, we do not apply the sliding window dataset augmentation as adequate data is available and our preliminary experiments already indicated excellent performance with a training step size of 64. Moreover, we set the number of epochs to 200 (with early stopping), the decision threshold to 0.5, and the learning rate to 0.001. Guided by the AE experiments, we select a dropout of 55% for MR-TCN and 40% for the DWS model. The results of our evaluation and comparison to related work are summarized in Table V. As some work only reports results for individual attack classes, we compute the macro average over these results to facilitate comparison.

From our experiments, it can be observed that MR-TCN achieves state-of-the-art results across all CAN datasets with F1 scores of 99.8899%, 99.7455%, and 99.9350% for HCRL-CH, ROAD, and CAN FD, respectively. Hence, it can be concluded that, in addition to AE, MR-TCN effectively generalizes to both CAN CC and CAN FD traffic. This protocol flexibility not only reduces the design effort by re-using the same network architecture, but it also opens the path for cross-protocol IDSs capable of simultaneously analyzing both CAN and AE traffic. Indeed, while different in nature, CAN and AE packets often exhibit correlations, including tunneled messages. Although the DWS architecture also achieves similar results on the HCRL-CH and CAN FD datasets, it fails to generalize on the ROAD dataset. The ROAD dataset features sophisticated attacks, making detection more challenging. This is also reflected by the marginally lower F1 score of MR-TCN on the ROAD dataset. Additionally, MR-TCN, being a TCN-based architecture, performs remarkably well in terms of false positives. This observation was also highlighted in [82] and is particularly relevant for automotive IDSs. Indeed, frequent alerts will quickly be ignored by the driver of the vehicle.

TABLE V
SOFTWARE RESULTS OF MR-TCN FOR CAN

Model	Dataset	Acc	FPR	FNR	F1
MRTCN-CAN_CC	**HCRL-CH**	**0.9992**	**0.0000**	**0.0022**	**0.9989**
DWS	HCRL-CH	0.9993	0.0000	0.0019	0.9991
TCAN-IDS [47][a,b]	HCRL-CH	0.9994	-	0.0005	0.9996
DCNN [37][a,b]	HCRL-CH	0.9993	-	0.0015	0.9991
BiTCN [81]	HCRL-CH	1.0000	-	0.0000	1.0000
MTH-IDS [52]	HCRL-CH	1.0000	0.0000	-	1.0000
NovelADS [87][c]	HCRL-CH	-	-	0.0009	0.9993
ECACNN [41]	HCRL-CH	0.9998	-	0.0001	0.9999
MRTCN-CAN_CC	**ROAD**	**0.9997**	**0.0000**	**0.0042**	**0.9975**
DWS	ROAD	0.9428	0.0601	0.0083	0.6565
DESC-IDS [88][a,c]	ROAD	0.9041	-	0.0685	0.9024
GRU-Latent AE [83][a]	ROAD	1.00	0.000	0.000	1.00
CANLP [89]	ROAD	-	-	-	0.997
MRTCN-CAN_FD	**CAN FD**	**0.9993**	**0.0000**	**0.0013**	**0.9994**
DWS	CAN FD	0.9992	0.0000	0.0015	0.9993
EM [90][a]	CAN FD	0.9999	-	0.00	0.9999

[a] Macro-averaged results.
[b] Although the model is a binary classifier, a separate IDS is trained per attack type.
[c] Unsupervised learning approach.

B. Hardware Implementation

We now detail our custom hardware accelerator which can be deployed on emerging hybrid FPGA-based ECUs to enable real-time intrusion detection. As resources are typically scarce in embedded FPGA platforms, it is often necessary to reduce the storage size of the network before deploying it on hardware. Thus we first quantize MR-TCN and revisit its classification performance on AE and CAN network traffic.

To assess the effect of quantization and identify the optimal configuration, we conduct a sensitivity analysis on the model. Indeed, it is ultimately the performance in hardware that is relevant for the practical deployment of MR-TCN. Moreover, we repeat this analysis on the considered datasets to gain insight into how each dataset contributes to MR-TCN's performance. Specifically, we re-train our models with the Brevitas QAT framework, exploring a range of quantization configurations with few-bit weights, activations, and biases. In our experiments, we set the learning rate to 10^{-5} for AE and 0.001 for CAN, similar to the software experiments. Additionally, due to the regularizing effect of the quantization, we reduce the dropout of the CAN variant to 0%. For AE, we evaluate two variants of the (Q)MRTCN-AE^{T} model: (1) the variant as considered in the software experiments (Sec. V-A), and (2) a modified variant with 1024 neurons in the final linear layer. For the CAN datasets, we only consider the updated (Q)MRTCN-CAN variants with 1024 linear nodes. The results of our analysis are depicted in Fig. 6.

From the analysis, it can be concluded that the impact of the quantization significantly differs per dataset. Whereas for

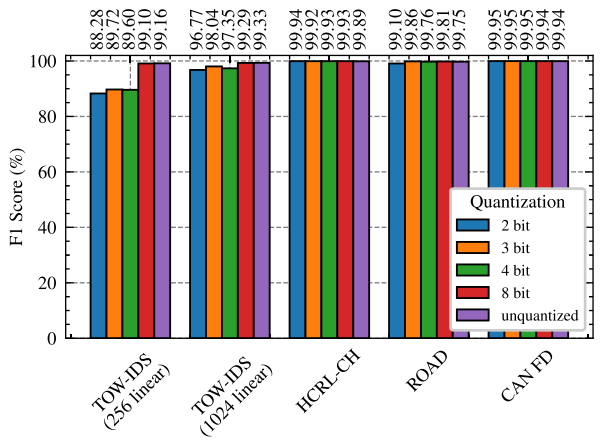


Fig. 6. Sensitivity of MR-TCN to quantization for AE and CAN datasets.

TABLE VI
HARDWARE RESULTS FOR AE USING THE 3-BIT QUANTIZED MODEL

Model	Linear neurons	Acc	FPR	FNR	F1
QMRTCN-AE^{T}	256	0.9076	0.0891	0.0965	0.8972
QMRTCN-AE^{T}	512	0.9794	0.0334	0.0092	0.9750
QMRTCN-AE^{T}	1024	0.9824	0.0232	0.0107	0.9804

TABLE VII
HARDWARE RESULTS FOR CAN USING THE 3-BIT QUANTIZED MODEL WITH 1024 LINEAR NEURONS

Model	Dataset	Acc	FPR	FNR	F1
QMRTCN-CAN_CC	HCRL-CH	0.9995	0.0000	0.0015	0.9993
QC2F-IDS [53][a]	HCRL-CH	0.9983	-	0.0047	0.9975
CQMLP-IDS [91][a,b]	HCRL-CH	-	-	0.0013	0.9990
QMRTCN-CAN_CC	ROAD	0.9998	0.0000	0.0025	0.9986
QMRTCN-CAN_FD	CAN FD	0.9995	0.0000	0.0010	0.9995

[a] Macro-averaged results.
[b] Does not consider gear-spoofing attack.

AE, there is a notable difference between the unquantized software implementation and the quantized hardware implementations, this difference is less pronounced for the CAN datasets. One intuition for the performance drop in AE lies in the fact that the dataset is more complex than the CAN datasets, reflected by the lower classification performance in the software experiments. The MR-TCN models are small models with limited capacity, which is further reduced by the quantization procedure. This is also reflected in the results for the (Q)MRTCN-AE^{T} model with 256 nodes in the last layer, with the F1 score dropping from 99.16 % for the unquantized model to 88.28 % for the 2-bit quantized model. However, in line with other work [84], 8-bit quantization does not have a significant impact on the performance but requires a significant amount of resources. Quantizing beyond 8 bits produces unstable results.

While the early convolutional layers of the network are important for effective high-dimensional feature extraction, the final linear layer serves an important role in the classification. As MR-TCN only has a single linear layer for classification, the quantization has a significant impact on the performance when a limited number of neurons are used. To address this issue, we also consider a configuration of MR-TCN with additional nodes (1024) in the last linear layer. The intuition is that this configuration compensates for the performance penalty caused by the quantization by introducing additional capacity. Please note that a more detailed overview with configurations of MR-TCN considering 256, 512, and 1024 nodes in the last linear layer is provided in Table VI. Based on the results of the (Q)MRTCN-AE^{T} model with 1024 linear nodes, we again observe that 8-bit quantization does not show a notable performance degradation. Further, in contrast to the 2-bit quantized model, both the 3- and 4-bit models exhibit limited performance penalties, making them suitable candidates for deployment on hardware. However, since both models exhibit similar performance, we opt for the 3-bit model as it significantly reduces the resource consumption.

As previously highlighted, the considered CAN datasets exhibit lower complexity compared to the AE dataset, demonstrated by the consistenly higher F1 scores. In particular, the HCRL-CH and CAN FD dataset contain unstealthy attacks and place minimal demands on the model capacity. Consequently, for these datasets, quantization has a negligible impact on performance, with the quantized hardware models yielding results comparable to those of the unquantized software models. In some cases, the quantized models exhibit marginally improved performance compared to the unquantized model, which can likely be attributed to the regularizing effect of the quantization. The ROAD dataset, by contrast, contains more advanced attacks, resulting in a performance degradation when quantizing down to 2 bits. Thus, for CAN, we also consider 3-bit quantization as an ideal candidate for hardware acceleration. Lastly, in Table VII, we provide a detailed overview of the 3-bit quantized CAN models, as well as a comparison w.r.t. state-of-the-art models on the HCRL-CH dataset.

With the quantized models trained, it is now possible to deploy MR-TCN in hardware. To generate the FPGA accelerators, we leverage the FINN framework. Specifically, we target a PYNQ-Z2 development board featuring an AMD/Xilinx Zynq XC7Z020 SoC. However, it should be noted that the proposed accelerators can also be deployed on other AMD/Xilinx FPGAs. In our experiments, we assume a fixed clock speed of 100 MHz. Moreover, the resource and power consumption are reported based on the out-of-context (OOC) implementation results obtained from Vivado 2022.2, while the latency and throughput are derived from the post-synthesis simulation waveforms. To allow for a fair comparison w.r.t. the state of the art, we report the throughput in packets per second (pps) rather than features per second. Please note that for pipelined (streaming dataflow) architectures, the throughput considers a full pipeline and is therefore solely defined by the slowest stage in the pipeline. As such, the throughput can be computed as $1/latency_{slowest\_stage}$. To obtain the throughput in pps, we multiply the obtained value with the window size. The latency, by contrast, considers the delay for a single sample from input to output of the entire empty pipeline.

To optimize the generated accelerator, FINN offers several options for fine-tuning the hardware. As a first step, a preferred implementation style can be specified per layer of the model. We choose the recently introduced RTL style rather than the

TABLE VIII
HARDWARE RESULTS FOR THE BEST 3-BIT QUANTIZD MODEL CONSIDERING 100BASE-T1, CAN CC, AND CAN FD

Model	Platform	Params ($\times 10^3$)	LUT	LUTRAM	FF	BRAM	DSP	Throughput (pps)	Latency (µs)	Power (W)
QMRTCN-AE$^T_{eff}$	FPGA	26.656	15152	703	14128	17	0	1313628	54.385	0.497
QMRTCN-AE$^T_{worst}$	FPGA	26.656	5938	577	7019	10	0	260331	279.025	0.228
QMRTCN-AE$^T_{avg}$	FPGA	26.656	4595	506	5575	12	0	86796	830.115	0.193
TOW-IDS [12]	GPU	978.284	-	-	-	-	-	15483	29191.65	-
AERO [9]	GPU	289	-	-	-	-	-	9638[a]	-	-
Swin-T [63]	GPU	-	-	-	-	-	-	26718	19163	-
Multi-IDS [61]	GPU	-	-	-	-	-	-	53956	834	-
QMRTCN-CAN_CC	FPGA	17.443	4062	191	3269	8.5	0	21700	3247.155	0.170
QC2F-IDS [53]	FPGA	-	33224	-	54175	138	0	115830/54744[b]	259/548[b]	2.29[c]
CQMLP-IDS [91]	FPGA	-	3999	-	4524	4	0	9090	110	2.15[c]
ECACNN [41]	GPU/CPU	156.034	-	-	-	-	-	106666/23272	1200/5500	-
QMRTCN-CAN_FD	FPGA	28.195	4274	518	4953	16	0	30635	2297.475	0.186

[a] Actual throughput depends on the chosen model configuration.
[b] Coarse-Only/Coarse-Fine.
[c] Power reported for the entire platform.

conventional HLS implementation, as it is more efficient for few-bit quantized models [92]. Additionally, to increase the throughput of the model, the FINN framework allows for the specification of a folding configuration. However, when selecting the optimal folding configuration, it is important to consider the application requirements. Indeed, unfolding of the model beyond the application's requirements introduces additional resources and power consumption. We adjust the folding configuration of MR-TCN so that it satisfies the real-time requirement at 100% bus load without excessive resource utilization. Based on the IVN architecture, we consider several folding scenarios for AE and CAN:

- **Automotive Ethernet.** For our AE accelerator, we assume folding configurations that guarantee real-time operation under average packet lengths[7] (150 bytes/packet), and worst-case packet lengths[8] (64 bytes/packet). As such, the required minimum throughputs[9] for the cases of 100BASE-T1 can be found as 83334 pps and 195313 pps, respectively. Additionally, to assess the scalability of MR-TCN in emerging IVN protocols (e.g., 1000BASE-T1), we provide a resource-efficient[10] folding configuration with high computational density. As the throughput of the streaming-dataflow accelerator is only limited by the slowest layer, the highest computational density can be achieved with a balanced pipeline.
- **Controller Area Network.** In the design of our CAN accelerators, we exclusively focus on configurations that guarantee real-time operation under worst-case packet lengths. For CAN CC, this implies packets of 47 bits (11-bit IDs, payloads of 0 bytes, and no bit stuffing) transmitted at 1 Mbps. However, for CAN FD, the worst-case packet length including the 6 fixed stuff bits in the *stuff count* and *CRC* fields is 62 bits. Of these, 29 bits are transmitted at the arbitration rate (1 Mbps), while the remaining 33 bits are transmitted at the data rate (8 Mbps). As such, the minimum required throughput can be found as 21 276 pps and 30 189 pps for CAN CC and CAN FD, respectively.

In Table VIII, we summarize the model size, resource consumption, latency, throughput, and power consumption of our accelerators and provide a comparison w.r.t. related IDSs. In addition, we provide an overview of our folding configurations in Appendix. Our evaluation highlights that MR-TCN is capable of low-power, lightweight, real-time intrusion detection in 100BASE-T1, CAN CC, and CAN FD networks. Particularly, for the AE case, our proposed accelerator requires $\approx 10-36\times$ less parameters than [9] and [12]. Moreover, the average and worst-case configurations achieve throughputs of 86 796 pps and 260 331 pps, while keeping the utilization of the most-utilized resource type (LUT) under 8.64% and 11.16% of the target platform, respectively. Additionally, we note that the resource-efficient configuration can effectively scale to 1000BASE-T1 as it achieves throughputs of 1.57 Gbps, assuming average-case packet lengths and maximum bus utilization. However, in a real-world setting the achieved throughput will typically be higher as, in practice, the bus load is $<$100% and many packets are longer than 150 bytes (e.g., in multimedia streams). In addition, the throughputs can further be increased by targeting a higher clock speed. For the CAN CC case, the proposed accelerator already achieved the real-time requirement at near-maximum folding, while achieving similar resource consumption to [91]. Lastly, we highlight that MR-TCN effectively scales to CAN FD settings, indicating its applicability in emerging CAN networks.

VI. DISCUSSION

In this section, we discuss the key properties of MR-TCN, focusing on the classification performance, model size, speed, and power consumption. We also make the case for on-device Edge ML on the Zonal Gateway and address the limitations of the proposed accelerators.

[7] We determine the average packet length based on the TOW-IDS dataset.
[8] We note that 64 bytes is the minimum length for Ethernet packets.
[9] The minimum throughputs only consider the link-layer overhead. In practice, packets on the physical layer will have an additional overhead of 8 bytes for the preamble and 12 bytes for the inter-frame gap. These overheads result in a decreased minimum throughput and allow for more relaxed folding configurations requiring fewer resources.
[10] Folding configurations with a higher computational density can be obtained. However, these folding configurations require resources beyond the capabilities of the target platform.

A. Classification Performance

MR-TCN demonstrates classification performance comparable to more complex state-of-the-art models. Notably, the TCN-based architecture excels in maintaining a low false positive rate. While most related work tends to overlook the importance of the false positive rate, it remains one of the most significant challenges for IDSs in IVNs [17]. One technique that can be used to further improve the detection performance of MR-TCN is to run inference on every incoming packet (i.e., to apply a step-size of 1 to the test set). However, this technique would significantly increase the required throughput and, hence, the required resources of the model. Our software experiments highlight that when leveraging the per-packet inference for AE, all attacks were successfully detected in at least one window.

B. Model Size

One key challenge in the automotive industry is the need for cost-effective solutions. However, we find that many of the existing IDSs for IVN propose overly complex models. As highlighted in Sec. V-B, MR-TCN proposes a lightweight architecture which reduces the number of parameters compared to existing work. For instance, the MR-TCN architecture requires 26 656, 17 443, and 28 195 parameters for AE, CAN CC, and CAN FD respectively. Considering the 3-bit quantization, these sizes correspond to 9.76 kB, 6.39 kB and, 10.33 kB, respectively. Therefore, MR-TCN is suited for deployment on automotive-grade platforms with limited computing power and resource availability.

C. Speed

MR-TCN is capable of real-time intrusion detection in modern automotive networks and can effectively be scaled to emerging high-bandwidth technologies such as multi-gigabit AE. While some existing work has already considered real-time detection capabilities for the current 100BASE-T1 [9], [22], [60], [61], it typically fails to meet the requirements for future 1000Base-T1 networks. Additionally, most related work assumes a detection time of 1000 μs per network packet as the threshold for real-time operation [22], [60], [61]. However, this threshold is solely based on observational data obtained from a physical testbed in [22]. Furthermore, the aforesaid assumption is in contradiction with [9], where a threshold of 3500 network packets per second is applied based on real IVN data. Given the safety-critical nature, automotive IDSs should be able to account for the worst-case scenario in terms of throughput. In addition, even with near-maximum folding, our proposed CAN CC and CAN FD accelerators achieve real-time detection under worst-case throughputs.

D. Power Consumption

We highlight that the overhead in power consumption introduced by MR-TCN is comparable to cryptographic approaches and negligible w.r.t. the overall power consumption of a modern Smart Electric Vehicle (SEV). Consider the TA100 automotive Hardware Security Module (HSM), which can be used for authentication of CAN messages [93]. Under typical conditions, this chip has a power consumption of 68-138 mW per ECU, which is of the same order of magnitude as MR-TCN (170-228 mW). Assuming a typical modern zonal architecture with 100 ECUs distributed evenly over four zones, this translates in a cumulative power consumption of 1.69-3.44 W per zone. Nevertheless, it is important to emphasize that cryptographic solutions remain essential, and that MR-TCN is designed to complement -rather than replace- these security solutions. Further, it is assumed that an SEV on average consumes 191 Wh/km [94]. Considering a cruising speed of 70 km/h, we find an average power consumption of 13.4 kW. Contrasting this to the worst-case power consumption introduced by MR-TCN, the overhead can be considered negligible.

E. Edge ML

To enable advanced functionalities, modern IVNs are challenged by increasing volumes of data. To process the increasing amount of data, several vehicular IDSs have explored techniques such as cloud offloading. However, given the real-time nature of IVN traffic, processing data in the cloud becomes challenging as it impairs the real-time capabilities of the IDS, requires high bandwidths and consumes substantial power. Consequently, there is growing interest in Edge intelligence, accelerating the IDS on the ECU. In the context of zonal architectures, the zonal gateway has access to all the network interfaces in its zone, making it an ideal candidate for deploying MR-TCN.

F. Limitations

The deployment of IDSs in IVNs has been challenged by false alarms. Although MR-TCN effectively minimizes the false positive rate, false alarms do, in fact, still persist. One key technique to mitigate this issue is to explore active response measures/warnings that operate on thresholds. Such threat analysis systems closely align with the framework proposed by AUTOSAR IDS, where an IDS monitor is responsible for the filtering and aggregation of security events to compose qualified security events [15]. Moreover, by combining MR-TCN with deterministic schemes such as Remote Attestation, the false positives can effectively be mitigated.

Another constraint challenging the real-world deployment of MR-TCN (and by extension any ML-based vehicular IDS), is that the classification performance heavily depends on the availability and quality of data. In practice, high-quality labelled data is typically scarce. Furthermore, as the optimal choice of the hyperparameters and the exact model configuration depend on the complexity of the dataset, a comprehensive analysis on a wide range of datasets should be conducted.

VII. Conclusion and Future Work

In this work, we propose MR-TCN, a lightweight ML architecture for intrusion detection in modern IVNs. We develop and evaluate variants of the architecture processing CAN CC, CAN FD, and AE network traffic. Through extensive evaluation on relevant datasets, we demonstrate that MR-TCN can

TABLE IX
FOLDING CONFIGURATIONS

Layer		FMPad0	ConvInpGen0	MVAU0	FMPad1	ConvInpGen1	MVAU1	FMPad2	ConvInpGen2	MVAU2	MaxPool0	MVAU3
Implementation Style		RTL	RTL	HLS	RTL	RTL	HLS	RTL	RTL	HLS	HLS	HLS
QMRTCN-AE$^{T}_{eff}$	PE/SIMD	-/1	-/4	8/24	-/1	-/4	8/16	-/1	-/2	4/16	1/-	1/1
QMRTCN-AE$^{T}_{worst}$	PE/SIMD	-/1	-/1	2/16	-/1	-/1	1/24	-/1	-/1	1/12	1/-	1/1
QMRTCN-AE$^{T}_{avg}$	PE/SIMD	-/1	-/1	1/12	-/1	-/1	1/8	-/1	-/1	1/4	1/-	1/1
QMRTCN-CAN_CC	PE/SIMD	-/1	-/1	1/1	-/1	-/1	1/2	-/1	-/1	1/1	1/-	1/1
QMRTCN-CAN_FD	PE/SIMD	-/1	-/1	1/4	-/1	-/1	1/3	-/1	-/1	1/2	1/-	1/1

achieve state-of-the-art performance in software while significantly reducing the model size. Additionally, for each of the variants, we develop a custom hardware accelerator which can realistically be deployed on emerging FPGA-based ECUs. The hardware accelerators are characterized by reduced resource and power consumption, and can operate in real-time under worst-case scenarios. Finally, we show that the accelerators can effectively scale to support emerging high-bandwidth protocols, demonstrating the real-world applicability of MR-TCN.

As a future work, we will evaluate MR-TCN on the recently introduced CAN XL protocol (for which no datasets are available yet). The fact that our CAN experiments with minimal resources have already achieved real-time intrusion detection implies that the model can easily be scaled to the recent CAN XL. Moreover, we will explore unsupervised and semi-supervised approaches that can operate with minimal amounts of labelled data. Lastly, we will explore transfer learning and evaluate how well MR-TCN can adapt to different vehicles and attack types.

APPENDIX

In Table IX we summarize the folding configurations of the different MR-TCN models. Additionally, for each layer, we indicate the implementation style.

REFERENCES

[1] E. Commission. (2024). *Intelligent Transport Systems*. Accessed: Jul. 9, 2024. [Online]. Available: https://transport.ec.europa.eu/transport-themes/intelligent-transport-systems_en

[2] K. Koscher et al., "Experimental security analysis of a modern automobile," in *Proc. IEEE Symp. Secur. Privacy*, May 2010, pp. 447–462.

[3] S. Nie, L. Liu, and Y. Du, "Free-fall: Hacking Tesla from wireless to CAN bus," in *Proc. Black Hat USA*, Jul. 2017, pp. 1–16.

[4] K. S. Lab. (2018). *Experimental Security Assessment of BMW Cars: A Summary Report*. Accessed: Jan. 18, 2024. [Online]. Available: https://keenlab.tencent.com/en/whitepapers/Experimental_Security_Assessment_of_BMW_Cars_by_KeenLab.pdf

[5] S. Curry. (2023). *Web Hackers Vs. The Auto Industry: Critical Vulnerabilities in Ferrari, BMW, Rolls Royce, Porsche, and More*. Accessed: Jan. 18, 2024. [Online]. Available: https://samcurry.net/web-hackers-vs-the-auto-industry/

[6] *CAN Specification*, Robert Bosch GmbH, Stuttgart, Germany, 1991.

[7] *CAN With Flexible Data-Rate Specification*, Robert Bosch GmbH, Stuttgartth, Germany, 2012.

[8] *Controller Area Network Extra Long (CAN XL)*, CiA, Nuremberg, Germany, 2022.

[9] S. Jeong, H. K. Kim, M. L. Han, and B. I. Kwak, "AERO: Automotive Ethernet real-time observer for anomaly detection in in-vehicle networks," *IEEE Trans. Ind. Informat.*, vol. 20, no. 3, pp. 4651–4662, Mar. 2024.

[10] UNECE. (2020). *UN Regulations on Cybersecurity and Software Updates to Pave the Way for Mass Roll Out of Connected Vehicles*. Accessed: Jun. 9, 2024. [Online]. Available: https://unece.org/sustainable-development/press/un-regulations-cybersecurity-and-software-updates-pave-way-mass-roll

[11] D. Pannell et al. (2019). *Use Cases-IEEE P802. 1DG V0. 4*. Accessed: Jun. 9, 2025. [Online]. Available: https://www.ieee802.org/1/files/public/docs2019/dg-pannell-automotive-use-cases-0919-v04.pdf

[12] M. L. Han, B. I. Kwak, and H. K. Kim, "TOW-IDS: Intrusion detection system based on three overlapped wavelets for automotive Ethernet," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 411–422, 2023.

[13] *UN Regulation No. 155—Cyber Security and Cyber Security Management System*, UNECE, Geneva, Switzerland, 2021.

[14] *Road Vehicles—Cybersecurity Engineering*, ISO, Geneva, Switzerland, Mar. 2021.

[15] AUTOSAR. (2020). *Specification of Intrusion Detection System Protocol*. Accessed: Jun. 9, 2024. [Online]. Available: https://www.autosar.org/fileadmin/standards/R20-11/FO/AUTOSAR_PRS_IntrusionDetectionSystem.pdf

[16] A. Lotto, F. Marchiori, A. Brighente, and M. Conti, "A survey and comparative analysis of security properties of CAN authentication protocols," 2024, *arXiv:2401.10736*.

[17] B. Lampe and W. Meng, "Intrusion detection in the automotive domain: A comprehensive review," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 4, pp. 2356–2426, 4th Quart., 2023.

[18] G. Loukas, E. Karapistoli, E. Panaousis, P. Sarigiannidis, A. Bezemskij, and T. Vuong, "A taxonomy and survey of cyber-physical intrusion detection approaches for vehicles," *Ad Hoc Netw.*, vol. 84, pp. 124–147, Mar. 2019.

[19] M. Blott et al., "FINN- R: An end-to-end deep-learning framework for fast exploration of quantized neural networks," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 11, no. 3, pp. 1–23, Sep. 2018.

[20] Y. Umuroglu et al., "FINN: A framework for fast, scalable binarized neural network inference," in *Proc. ACM/SIGDA Int. Symp. Field-Program. Gate Arrays*, Feb. 2017, pp. 65–74.

[21] *IEEE Standard for Ethernet Amendment 4: Physical Layer Specifications and Management Parameters for 1 Gb/s Operation Over a Single Twisted-Pair Copper Cable*, IEEE Standard Std 802.3bp-2016, 2016.

[22] S. Jeong, B. Jeon, B. Chung, and H. K. Kim, "Convolutional neural network-based intrusion detection system for AVTP streams in automotive Ethernet-based networks," *Veh. Commun.*, vol. 29, Jun. 2021, Art. no. 100338.

[23] Q. Liu, X. Li, K. Sun, Y. Li, and Y. Liu, "SISSA: Real-time monitoring of hardware functional safety and cybersecurity with in-vehicle SOME/IP Ethernet traffic," *IEEE Internet Things J.*, vol. 11, no. 16, pp. 27322–27339, Aug. 2024.

[24] A. Pappalardo et al., "QONNX: Representing arbitrary-precision quantized neural networks," 2022, *arXiv:2206.07527*.

[25] G. Franco, A. Pappalardo, and N. J. Fraser, "Xilinx/brevitas," Zenodo, 2025, doi: 10.5281/zenodo.3333552.

[26] L. L. Jeune, T. Goedemé, and N. Mentens, "Feature dimensionality in CNN acceleration for high-throughput network intrusion detection," in *Proc. 32nd Int. Conf. Field-Program. Log. Appl. (FPL)*, Aug. 2022, pp. 366–374.

[27] H. Wang, P. Xiao, and X. Li, "Channel parameter estimation of mmWave MIMO system in urban traffic scene: A training channel-based method," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 1, pp. 754–762, Jan. 2022.

[28] K. Wang, J. Guo, K. Chen, and J. Lu, "An in-depth examination of SLAM methods: Challenges, advancements, and applications in complex scenes for autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 26, no. 7, pp. 11066–11087, Jul. 2025.

[29] T. Yoshizawa et al., "A survey of security and privacy issues in V2X communication systems," *ACM Comput. Surv.*, vol. 55, no. 9, pp. 1–36, Sep. 2023.

[30] P. Jing et al., "Revisiting automotive attack surfaces: A practitioners' perspective," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2024, pp. 2348–2365.

[31] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1153–1176, 2nd Quart., 2016.

[32] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, "A detailed investigation and analysis of using machine learning techniques for intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 686–728, 1st Quart., 2019.

[33] D. Chou and M. Jiang, "A survey on data-driven network intrusion detection," *ACM Comput. Surv.*, vol. 54, no. 9, pp. 1–36, Dec. 2022.

[34] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 1, p. 4150, Jan. 2021.

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

[36] E. Seo, H. M. Song, and H. K. Kim, "GIDS: GAN based intrusion detection system for in-vehicle network," in *Proc. 16th Annu. Conf. Privacy, Secur. Trust (PST)*, Aug. 2018, pp. 1–6.

[37] H. M. Song, J. Woo, and H. K. Kim, "In-vehicle network intrusion detection using deep convolutional neural network," *Veh. Commun.*, vol. 21, Jan. 2020, Art. no. 100198.

[38] T. P. Nguyen, H. Nam, and D. Kim, "Transformer-based attention network for in-vehicle intrusion detection," *IEEE Access*, vol. 11, pp. 55389–55403, 2023.

[39] M. Nam, S. Park, and D. S. Kim, "Intrusion detection method using bi-directional GPT for in-vehicle controller area networks," *IEEE Access*, vol. 9, pp. 124931–124944, 2021.

[40] J. Song, G. Qin, Y. Liang, J. Yan, and M. Sun, "SIDiLDNG: A similarity-based intrusion detection system using improved Levenshtein distance and N-gram for CAN," *Comput. Secur.*, vol. 142, Jul. 2024, Art. no. 103847.

[41] Z. Xia, L. Huang, J. Tan, Y. Yu, W. Hao, and K. Long, "A lightweight intrusion detection system for connected autonomous vehicles based on ECANet and image encoding," *J. Inf. Secur. Appl.*, vol. 92, Jul. 2025, Art. no. 104082.

[42] A. Taylor, S. Leblanc, and N. Japkowicz, "Anomaly detection in automobile control network data with long short-term memory networks," in *Proc. IEEE Int. Conf. Data Sci. Adv. Anal. (DSAA)*, Oct. 2016, pp. 130–139.

[43] H. Sun, M. Chen, J. Weng, Z. Liu, and G. Geng, "Anomaly detection for in-vehicle network using CNN-LSTM with attention mechanism," *IEEE Trans. Veh. Technol.*, vol. 70, no. 10, pp. 10880–10893, Oct. 2021.

[44] P. Wei, B. Wang, X. Dai, L. Li, and F. He, "A novel intrusion detection model for the CAN bus packet of in-vehicle network based on attention mechanism and autoencoder," *Digit. Commun. Netw.*, vol. 9, no. 1, pp. 14–21, Feb. 2023.

[45] P. Freitas De Araujo-Filho, A. J. Pinheiro, G. Kaddoum, D. R. Campelo, and F. L. Soares, "An efficient intrusion prevention system for CAN: Hindering cyber-attacks with a low-cost platform," *IEEE Access*, vol. 9, pp. 166855–166869, 2021.

[46] K. Stein, A. Mahyari, and E. El-Sheikh, "Vehicle controller area network inspection using recurrent neural networks," in *Proc. Int. Conf. Adv. Comput. Res.*, Jan. 2023, pp. 494–499.

[47] P. Cheng, K. Xu, S. Li, and M. Han, "TCAN-IDS: Intrusion detection system for Internet of Vehicle using temporal convolutional attention network," *Symmetry*, vol. 14, no. 2, p. 310, Feb. 2022.

[48] A. K. Desta, S. Ohira, I. Arai, and K. Fujikawa, "MLIDS: Handling raw high-dimensional CAN bus data using long short-term memory networks for intrusion detection in in-vehicle networks," in *Proc. 30th Int. Telecommun. Netw. Appl. Conf. (ITNAC)*, 2020, pp. 1–7.

[49] A. R. Javed, S. U. Rehman, M. U. Khan, M. Alazab, and G. T. Reddy, "CANintelliIDS: Detecting in-vehicle intrusion attacks on a controller area network using CNN and attention-based GRU," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 1456–1466, Apr. 2021.

[50] S. Tariq, S. Lee, and S. S. Woo, "CANTransfer: Transfer learning based intrusion detection on a controller area network using convolutional LSTM network," in *Proc. 35th Annu. ACM Symp. Appl. Comput.*, Mar. 2020, pp. 1048–1055.

[51] W. Lo, H. Alqahtani, K. Thakur, A. Almadhor, S. Chander, and G. Kumar, "A hybrid deep learning based intrusion detection system using spatial–temporal representation of in-vehicle network traffic," *Veh. Commun.*, vol. 35, Jun. 2022, Art. no. 100471.

[52] L. Yang, A. Moubayed, and A. Shami, "MTH-IDS: A multitiered hybrid intrusion detection system for Internet of Vehicles," *IEEE Internet Things J.*, vol. 9, no. 1, pp. 616–632, Jan. 2022.

[53] A. Rangsikunpum, S. Amiri, and L. Ost, "An FPGA-based intrusion detection system using binarised neural network for CAN bus systems," in *Proc. IEEE Int. Conf. Ind. Technol. (ICIT)*, Jun. 2024, pp. 1–6.

[54] L. Zhang, X. Yan, and D. Ma, "A binarized neural network approach to accelerate in-vehicle network intrusion detection," *IEEE Access*, vol. 10, pp. 123505–123520, 2022.

[55] S. Khandelwal and S. Shreejith, "A lightweight FPGA-based IDS-ECU architecture for automotive CAN," in *Proc. Int. Conf. Field-Program. Technol. (ICFPT)*, Dec. 2022, pp. 1–9.

[56] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," *Comput. Secur.*, vol. 86, pp. 147–167, Jun. 2019.

[57] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2015, pp. 1–6.

[58] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy (ICISSP)*, 2018, pp. 108–116.

[59] P. R. Carmo, P. F. de Araujo-Filho, D. R. Campelo, E. Freitas, A. T. de Oliveira Filho, and D. F. Sadok, "Machine learning-based intrusion detection system for automotive Ethernet: Detecting cyber-attacks with a low-cost platform," *Anais do XL Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, vol. 2022, pp. 196–209, May 2022.

[60] L. F. M. da Luz, P. F. de Araujo-Filho, and D. R. Campelo, "Multi-criteria optimized deep learning-based intrusion detection system for detecting cyberattacks in automotive Ethernet networks," in *Proc. Anais do XLI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, Jun. 2023, pp. 197–210.

[61] L. F. M. de Luz, P. F. de Araujo-Filho, and D. R. Campelo, "Multi-stage deep learning-based intrusion detection system for automotive Ethernet networks," *Ad Hoc Netw.*, vol. 162, May 2024, Art. no. 103548.

[62] N. Alkhatib, M. Mushtaq, H. Ghauch, and J.-L. Danger, "Unsupervised network intrusion detection system for AVTP in automotive Ethernet networks," in *Proc. IEEE Intell. Veh. Symp. (IV)*, May 2022, pp. 1731–1738.

[63] S. Wang, H. Zhou, H. Zhao, Y. Wang, A. Cheng, and J. Wu, "A zero false positive rate of IDS based on Swin transformer for hybrid automotive in-vehicle networks," *Electronics*, vol. 13, no. 7, p. 1317, Mar. 2024.

[64] M. S. Leandro, P. F. de Araujo-Filho, D. R. Campelo, and L. F. M. da Luz, "SeqWatch: Unsupervised sequence-based intrusion detection system for automotive Ethernet," in *Proc. Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*, Sep. 2025, pp. 378–391.

[65] K. H. Shibly, M. D. Hossain, H. Inoue, Y. Taenaka, and Y. Kadobayashi, "A feature-aware semi-supervised learning approach for automotive Ethernet," in *Proc. IEEE Int. Conf. Cyber Secur. Resilience (CSR)*, Jul. 2023, pp. 426–431.

[66] Y. Liu et al., "Vehicular intrusion detection system for controller area network: A comprehensive survey and evaluation," *IEEE Trans. Intell. Transp. Syst.*, vol. 26, no. 7, pp. 10979–11009, Jul. 2025.

[67] W. Hellemans, M. M. Rabbani, B. Preneel, and N. Mentens, "Yes we can! Towards bringing security to legacy-restricted controller area networks. A review," in *Proc. 20th ACM Int. Conf. Comput. Frontiers*, May 2023, pp. 352–357.

[68] H. J. Jo and W. Choi, "A survey of attacks on controller area networks and corresponding countermeasures," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6123–6141, Jul. 2022.

[69] S. Kulandaivel, S. Jain, J. Guajardo, and V. Sekar, "CANNON: Reliable and stealthy remote shutdown attacks via unaltered automotive microcontrollers," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2021, pp. 195–210.

[70] A. de Faveri Tron, S. Longari, M. Carminati, M. Polino, and S. Zanero, "CANflict: Exploiting peripheral conflicts for data-link layer attacks on automotive networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Los Angeles, CA, USA, Nov. 2022, pp. 711–723.

[71] K.-T. Cho and K. G. Shin, "Fingerprinting electronic control units for vehicle intrusion detection," in *Proc. 25th USENIX Secur. Symp. (USENIX Secur.)*, 2016, pp. 911–927.

[72] K.-T. Cho and K. G. Shin, "Viden: Attacker identification on in-vehicle networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2017, pp. 1109–1123.
[73] M. E. Verma et al., "A comprehensive guide to CAN IDS data & introduction of the ROAD dataset," 2020, *arXiv:2012.14600*.
[74] H. K. Kim. (2018). *Car-Hacking Dataset*. Accessed: Aug. 19, 2024. [Online]. Available: https://ocslab.hksecurity.net/Datasets/car-hacking-dataset
[75] S. Rajapaksha, H. Kalutarage, G. Madzudzo, A. Petrovski, and M. O. Al-Kadri, "CAN-MIRGU: A comprehensive CAN bus attack dataset from moving vehicles for intrusion detection system evaluation," in *Proc. Symp. Vehicle Secur. Privacy*, May 2024, p. 11.
[76] H. K. Kim. (2021). *CAN-FD Intrusion Dataset*. Accessed: Aug. 19, 2024. [Online]. Available: https://ocslab.hksecurity.net/Datasets/can-fd-intrusion-dataset
[77] N. Alkhatib, M. Mushtaq, H. Ghauch, and J.-L. Danger, "Here comes SAID: A SOME/IP attention-based mechanism for intrusion detection," in *Proc. 14th Int. Conf. Ubiquitous Future Netw. (ICUFN)*, Jul. 2023, pp. 462–467.
[78] S. Bai, J. Zico Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, *arXiv:1803.01271*.
[79] P. Hewage et al., "Temporal convolutional neural (TCN) network for an effective weather forecasting using time-series data from the local weather station," *Soft Comput.*, vol. 24, no. 21, pp. 16453–16482, Nov. 2020.
[80] M. Thill, W. Konen, H. Wang, and T. Bäck, "Temporal convolutional autoencoder for unsupervised anomaly detection in time series," *Appl. Soft Comput.*, vol. 112, Nov. 2021, Art. no. 107751.
[81] Y. Mei, W. Han, and K. Lin, "Intrusion detection for intelligent connected vehicles based on bidirectional temporal convolutional network," *IEEE Netw.*, vol. 38, no. 6, pp. 113–119, Nov. 2024.
[82] I. Chiscop, A. Gazdag, J. Bosman, and G. Biczók, "Detecting message modification attacks on the CAN bus with temporal convolutional networks," 2021, *arXiv:2106.08692*.
[83] S. Rajapaksha, H. Kalutarage, M. O. Al-Kadri, A. Petrovski, and G. Madzudzo, "Beyond vanilla: Improved autoencoder-based ensemble in-vehicle intrusion detection system," *J. Inf. Secur. Appl.*, vol. 77, Sep. 2023, Art. no. 103570.
[84] L. Le Jeune, "Machine learning for network intrusion detection on FPGA," Ph.D. thesis, Dept. ESAT, KU Leuven, Leuven, Belgium, 2023.
[85] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[86] W. J. Youden, "Index for rating diagnostic tests," *Cancer*, vol. 3, no. 1, pp. 32–35, Jan. 1950.
[87] K. Agrawal, T. Alladi, A. Agrawal, V. Chamola, and A. Benslimane, "NovelADS: A novel anomaly detection system for intra-vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 11, pp. 22596–22606, Nov. 2022.
[88] P. Cheng, M. Han, and G. Liu, "Des.-IDS: Towards an efficient real-time automotive intrusion detection system based on deep evolving stream clustering," *Future Gener. Comput. Syst.*, vol. 140, pp. 266–281, Mar. 2023.
[89] K. Balasubramanian et al., "CANLP: NLP-based intrusion detection system for CAN," in *Proc. 39th ACM/SIGAPP Symp. Appl. Comput.*, Apr. 2024, pp. 212–214.
[90] M. S. Sreelekshmi and S. Aji, "Enhancing CAN-FD security with an ensemble learning approach for intrusion detection," in *Proc. 4th Int. Conf. Intell. Technol. (CONIT)*, Jun. 2024, pp. 1–6.
[91] S. Khandelwal and S. Shreejith, "Exploring highly quantised neural networks for intrusion detection in automotive CAN," in *Proc. 33rd Int. Conf. Field-Program. Log. Appl. (FPL)*, Sep. 2023, pp. 235–241.
[92] S. Asad Alam, D. Gregg, G. Gambardella, T. Preusser, and M. Blott, "On the RTL implementation of FINN matrix vector compute unit," 2022, *arXiv:2201.11409*.
[93] Microchip. (2025). *TA100 CryptoAutomotiveT Security IC*. Accessed: May 20, 2025. [Online]. Available: https://www.microchip.com/en-us/product/TA100
[94] Fully Charged. *Energy Consumption of Full Electric Vehicles*. Accessed: Aug. 30, 2024. [Online]. Available: https://ev-database.org/cheatsheet/energy-consumption-electric-car

Wouter Hellemans (Graduate Student Member, IEEE) received the joint B.Sc. and M.Sc. degrees in electronics and ICT engineering technology from KU Leuven and Hasselt University in 2021 and 2022, respectively. He is currently pursuing the Ph.D. degree in engineering technology with ES&S, COSIC Research Group, KU Leuven. He is an SB Ph.D. Fellow at the Research Foundation Flanders (FWO). His research interests predominantly include deep learning, in-vehicle network security, remote attestation, intrusion detection, and FPGAs.

Laurens Le Jeune received the joint M.Sc. degree in electronics and ICT engineering technology from KU Leuven and Hasselt University, Diepenbeek, Belgium, in 2019, and the Ph.D. degree from KU Leuven in 2023, focusing on how machine learning can be applied for network intrusion detection, and how it can be accelerated for high-throughput implementations on FPGA. His research interests include machine learning and deep learning, FPGAs, network security, and intrusion detection.

Md Masoom Rabbani received the Ph.D. degree from the University of Padova in 2020. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Chalmers University of Technology and the University of Gothenburg. After completing his Ph.D., he joined ES&S, COSIC Research Group, KU Leuven, as a Post-Doctoral Researcher. His research interests include attestation, blockchain applications, and embedded systems security.

Bart Preneel is currently a Full Professor with KU Leuven, Leuven, Belgium, where he heads the COSIC Research Group. He has authored more than 450 scientific publications and is the inventor of five patents. His main research interests include cryptography, information security, and privacy. He is a member of the Advisory Group of ENISA and of the Academia Europaea. He is a member of the Knowledge Center of the Belgian Data Protection Authority. In 2015, he was elected as an IACR Fellow. He received the RSA Award for Excellence in the Field of Mathematics in 2014, the IFIP TC11 Kristian Beckman Award in 2015, and the ESORICS Outstanding Research Award in 2017. He was the program chair of more than 20 international conferences. He has been an invited speaker at more than 120 conferences in 50 countries. He was the President of IACR and is the Co-Founder and the Chairperson of the Board of LSEC.

Nele Mentens (Senior Member, IEEE) received the master's and Ph.D. degrees from KU Leuven in 2003 and 2007, respectively. She was a Visiting Researcher with Ruhr University Bochum in 2013 and EPFL in 2017. She is currently a Professor with Leiden University and KU Leuven. She was/is the PI in around 25 finished and ongoing research projects with national and international funding. She is the (co-)author in more than 150 international publications. She served/serves as a program committee member for renowned international conferences on hardware design and security. She was the program chair of several conferences. She received numerous best paper awards and nominations. She gave keynote talks at several conferences.