Expert Systems With Applications 221 (2023) 119771

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Review

A survey of deep learning-based intrusion detection in automotive applications

Brooke Lampe [a,b], Weizhi Meng [a,*]

[a] Technical University of Denmark, Anker Engelunds Vej 101, Kongens Lyngby, 2800, Denmark
[b] Georgia Institute of Technology, 225 North Avenue NW, 30332, Atlanta, GA, USA

ARTICLE INFO

Keywords:
Deep learning
Automotive security
Internal vehicle network
Controller Area Network (CAN)
Automotive Ethernet
Intrusion detection system

ABSTRACT

Modern automobiles depend on internal vehicle networks (IVNs) to control systems from the anti-lock brakes to the transmission to the locks on the doors. Many IVNs, particularly the Controller Area Network (CAN) bus, were developed with little regard for security, since the IVNs of the past were isolated from the outside world. In the present day, the assumption of isolation no longer applies. Cellular service, Wi-Fi, and Bluetooth are just a few examples of the connectivity of contemporary automobiles. Researchers have explored a number of automotive security enhancements, but such enhancements are often roadblocked by implementation challenges, complexity, and expense. An intrusion detection system (IDS) is a promising automotive security enhancement that requires little, if any, adjustment to a vehicle's existing infrastructure. Deep learning techniques can augment the capability of automotive IDSs, improving detection accuracy and precision. This paper provides a comprehensive overview of deep learning-based IDSs in automotive networks. We assemble various deep learning schemes, categorize them according to their topologies and techniques, and highlight their distinct contributions. In addition, we analyze each scheme's evaluation in terms of datasets, attack types, and metrics. We summarize the results of the schemes and assess the advantages and disadvantages of different deep learning architectures.

1. Introduction

The modern-day automobile is an intricate cyber–physical machine. Embedded systems known as Electronic Control Units (ECUs) communicate information and instructions across the vehicle's internal network (IVN). The Controller Area Network (CAN) bus is the de facto standard for IVNs. At the time of its design in the 1980s, reliability and error-handling were prioritized, as security was considered a non-issue for a system insulated from the outside world. In terms of reliability and error-handling, the CAN bus performs exceedingly well. It is all but immune to electrical interference and capable of correcting errors (Bozdal et al., 2020).

Fig. 1 depicts the CAN bus as it is configured in automotive environments. The CAN protocol hinges on differential signaling; that is, the voltage difference between `CAN HIGH` and `CAN LOW` wires. If `CAN HIGH` > `CAN LOW`, then the dominant logic, "0", is transmitted. If `CAN HIGH` ≤ `CAN LOW`, then the recessive logic, "1", is transmitted. ECUs communicate by driving the CAN bus to the dominant state. In the absence of ECU input, terminating resistors return the CAN bus to the recessive state. Differential wiring insulates the CAN bus from electrical interference, since the dominant and recessive logic states are determined by the *difference* in voltage between `CAN HIGH` and `CAN LOW`, not the voltage measurements themselves (Bozdal et al., 2020; Narayanan et al., 2019).

CAN frames adhere to a particular structure, detailed in Fig. 2. The *start of frame (SOF)* announces the start of transmission. Next, the *arbitration identifier (ID)* specifies the priority of the frame and, generally, identifies the transmitting ECU. The *control* field contains several control bits. The *data length code (DLC)* specifies the size of the data field. The *data* field contains the actual data—the actual information—to be transmitted. The *cyclic redundancy checksum (CRC)* is a data integrity checksum. The *acknowledge (ACK)* allows a receiving node to acknowledge that the message was received successfully. Finally, the *end of frame (EOF)* identifies the end of the transmission.

1.1. Vulnerabilities of the CAN bus

However, its security shortcomings have come into the spotlight in recent years. All CAN traffic is broadcast to all nodes, and each node identifies and captures the messages it deems relevant. CAN traffic is not encrypted. The broadcast nature of the CAN bus means one

* Corresponding author.
E-mail addresses: blam@dtu.dk (B. Lampe), weme@dtu.dk (W. Meng).

https://doi.org/10.1016/j.eswa.2023.119771
Received 11 October 2022; Received in revised form 18 February 2023; Accepted 26 February 2023
Available online 28 February 2023

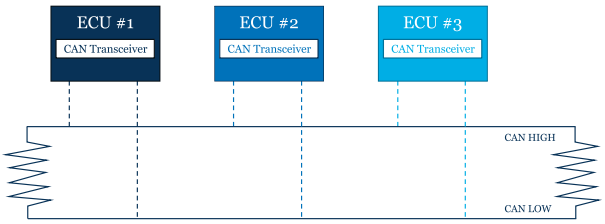

Fig. 1. Structure of the Controller Area Network (CAN) Bus.

compromised node would provide an adversary with access to all the information crossing the bus. Automotive manufacturers are using their vehicles to capture more and more information about the customers driving those vehicles, so the lack of encryption lends itself to potential privacy invasions.

The CAN protocol facilitates several error-checking mechanisms, including a checksum. The transmitting node calculates the cyclic redundancy checksum (CRC) and transmits it as part of the CAN frame. Upon receipt of the CAN frame, the receiving node also computes the CRC. If the receiving node's computed CRC does not match the CRC transmitted in the CAN frame, then an error flag will be sent. In this manner, the CAN protocol attempts to provide a guarantee of data integrity. Unfortunately, this integrity protection is only effective against incidental transmission errors. It is trivial for an adversary to craft a CAN frame with the correct CRC. Furthermore, because the CAN bus lacks authentication, an adversary can masquerade as any node. The receiving node cannot verify that the purported sender was the actual sender.

To preserve the availability of the network, the CAN bus provides a non-destructive arbitration scheme to resolve conflict when two nodes begin to transmit simultaneously. Each node in the network has a preset priority level, and when two nodes come into conflict, the highest priority node continues transmitting without disruption. In addition, the CAN protocol calls for an error confinement mechanism, which blocks faulty nodes from participating in—and disrupting—the bus traffic. Each of these protections can be thwarted by an adversary. To conduct a Denial of Service (DoS) attack, an adversary would send valid (error-free) messages with the highest priority arbitration ID. The adversary would send these messages quickly and continuously, and the malicious messages would supersede all genuine traffic (Bozdal et al., 2020; Koscher et al., 2010).

Palanca et al. crafted a selective DoS attack capable of evading detection at the frame level. This type of stealth attack avoids transmitting complete frames; as such, it is invisible to frame-level analysis. Because the attack leverages a vulnerability in the CAN protocol itself, it is effective against any implementation of said protocol. Further, the attack is relatively straightforward and inexpensive to conduct (Kim et al., 2021; Palanca et al., 2017).

Analyzing the CAN protocol in terms of the CIA triad—confidentiality, integrity, and availability (Friedman & Singer, 2014)—we can see that the CAN bus is vulnerable in all three arenas. The protocol's broadcast architecture and lack of encryption lead to confidentiality issues, while the existing integrity and availability protections prove insufficient to provide either integrity or availability in the event of an attack.

1.2. Attacks on the CAN bus

In addition to the Denial of Service attack described in the previous section, a number of attacks have proven effective against the CAN bus. Unsophisticated fuzzing attacks can disrupt the network, and fuzzing can also be used to map the CAN bus and determine which commands would prove useful in an attack. The adversary would construct random or semi-random CAN messages and monitor the resulting behavior of the system. The information obtained during the fuzzing process would then enable the adversary to craft suitable CAN messages for targeted attacks (Koscher et al., 2010).

In a replay attack, the adversary reads messages from the CAN bus and then injects (or "replays") the previously captured messages. Replay attacks can prove difficult to detect because the replayed messages are normal, expected messages. Using a collection of captured messages as a starting point, an adversary can also craft custom messages for injection. Generally, these messages consist of a valid arbitration ID—which belongs to the ECU that the adversary wishes to imitate—and a data field tailor-made to elicit a particular response from the receiving system (Abbott-McCune & Shay, 2016).

Diagnostic packets are a special class of packets intended to be used by mechanics to issue commands that would not be appropriate during normal operation of the vehicle. For example, `InputOutputControl` provides testing functionality. A mechanic can provide arbitrary inputs to an ECU and observe the resulting behavior. This functionality is intended to help detect and debug malfunctions. If an adversary were able to leverage this functionality on a vehicle operating at speed, then the safety of the occupants of the vehicle and others on the road would be in question. Fortunately, automotive manufacturers have devised authentication controls to protect against unauthorized diagnostic access. Unfortunately, Miller and Valasek discovered that these authentication controls leave much to be desired. Authentication controls are implemented per ECU. Some ECUs send the same seed, opening the door to replay attacks. Some ECUs expect an authentication response that is small enough to brute force. Better-protected ECUs would require the adversary to extract the key from the firmware, but even then, malicious diagnostic access is still feasible (Miller & Valasek, 2014, 2016b).

If an adversary injects spoofed CAN messages while the real ECU is still sending real messages, then the receiving system might fall back on "failsafe" values or refuse to function. To circumvent these confliction issues, an adversary would generally have two options: timing attacks and ECU flashing attacks. Some ECUs receive messages that contain a counter. The counter should be incremented with every message, so if the ECU receives two messages that share the same counter value, it will assume that the second message is a duplicate. The second message will be discarded. If the ECU receives too many duplicate messages in a given window, it may disable itself. The adversary can time his or her attack so that the injected messages arrive just before the real messages. The injected messages will be accepted, while the real messages are discarded (Miller & Valasek, 2016a).

Going a step further, the adversary can put the real ECU into Bootrom mode. In this mode, the old version of the firmware on the ECU is erased, and the updated version of the firmware is written back to the ECU. If an adversary were to start this process, erase the old firmware, and then stop, then the ECU would have no code and no ability to send CAN messages. The adversary would no longer have to contend with confliction. The process of updating the firmware on an ECU is called flashing an ECU. The adversary could start the flashing process and then stop once the conflicting ECU is disabled, but if the adversary can flash the ECU, then he or she can update it with malicious firmware. Malicious firmware would allow the adversary to control the ECU and use it to transmit malicious commands (Miller & Valasek, 2016a, 2016b).

1.3. Motivation for deep learning in automotive intrusion detection

The deep learning approach demonstrates a number of advantages over traditional machine learning IDSs as well as unsophisticated intrusion detection techniques. Five notable advantages are summarized in Fig. 3 and particularized below:

1. **Detect Novel Attacks** - Deep learning IDSs have proven especially capable of detecting novel attacks. Many of the deep learning paradigms discussed in our paper do not use labeled data to train; thus, they are not constrained to known attacks.

SOF	ID	CONTROL	DLC	DATA	CRC	ACK	EOF
Start of Frame	Arbitration Identifier	Control	Data Length Code	Data	Cyclic Redundancy Checksum	Acknowledge	End of Frame
1-bit	11-bit	3-bit	4-bit	0-8 bytes	16-bit	2-bit	7-bit

Fig. 2. Structure of the Controller Area Network (CAN) Frame.

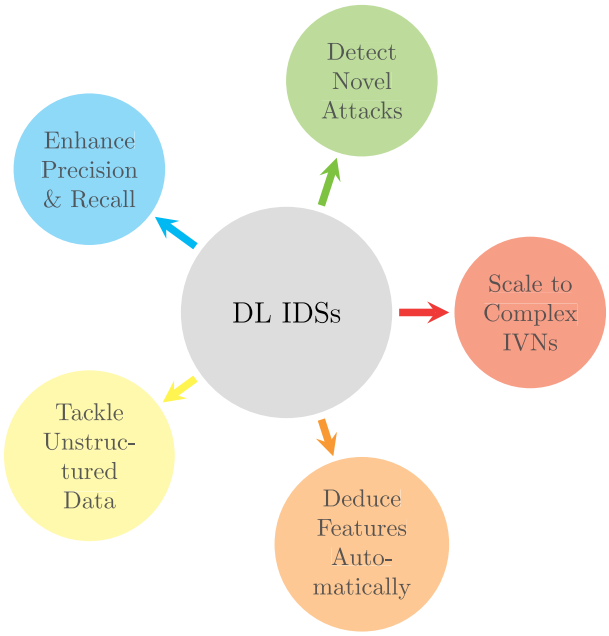

Fig. 3. Advantages of Deep Learning Intrusion Detection Systems in Automotive Applications.

2. **Enhance Precision & Recall** - The "deep" in deep learning refers to the number of layers in DL architectures. The layers facilitate fine-grained learning, which, in turn, leads to gains in precision, recall, and accuracy. Multiple authors have found that DL-based IDSs supersede traditional machine learning IDSs in automotive applications (Mehedi et al., 2021).
3. **Tackle Unstructured Data** - Deep learning IDSs are adept at generating valuable analyses from unstructured data. By using DL, we can save on the expensive human effort of reverse engineering CAN traffic. In many cases, DL does not require nearly as much pre-processing as traditional machine learning methods, either.
4. **Deduce Features Automatically** - A deep learning IDS can determine which features of the data are important and how they should be optimized. The need for manual feature extraction is reduced or even eliminated (Rashid, 2021).
5. **Scale to Complex IVNs** - The layers of a deep learning model allow it to continue learning where a traditional machine learning model would plateau. As such, a deep learning model can scale more effectively to highly complex IVNs while maintaining efficacy.

1.4. Related surveys in automotive intrusion detection

To our knowledge, our work constitutes the first survey of *deep learning* automotive intrusion detection systems. However, there are a number of relevant survey and review papers in the broader context of automotive intrusion detection. Much of the literature was published in 2019 and concentrated on intrusion detection systems for the automotive CAN bus.

Lokman, Othman, and Abu-Bakar provide valuable insights into the domain of automotive intrusion detection by deconstructing the core attributes of automotive IDSs, such as deployment strategies, detection techniques, attack types, and logistical challenges. They subdivided deployment strategies into (1) the CAN bus, (2) ECUs, and central gateways. An IDS deployed to the CAN bus itself or a central gateway would be analogous to a network-based IDS, while an IDS deployed to a particular ECU would be considered a host-based IDS. In terms of detection techniques, the authors categorized the literature into signature-based, specification-based, and anomaly-based IDSs. Signature- and specification-based IDSs depend on "knowns"—either known signatures or known behaviors translated into rules. Meanwhile, the class of anomaly-based IDSs aspires to detect unknown, novel attacks by establishing a baseline of normal behavior and flagging any deviation from that baseline as anomalous (Lokman et al., 2019).

Wu et al. explore attacks on in-vehicle networks (IVNs) at different layers of the protocol—physical, data link, application, etc. At the physical layer, various sensors, such as the Tire-pressure Monitoring System (TPMS) and perception systems (camera, LiDAR, etc.), avail themselves to spoofing attacks. At the data link layer, existing literature explores security threats in the realm of Denial of Service (DoS) and system failure. At the application layer, sniffing, replay, DoS, and spoofing attacks come into play. To counteract such threats, the authors introduce two types of automotive intrusion detection systems: (1) domain-aware and (2) context-aware. Domain-aware IDSs recognize that different ECUs—or clusters of ECUs—fulfill different functions and send different types of messages. Domain-aware IDSs leverage the dissimilarities in CAN bus communication between "domains" to detect anomalies. Context-aware IDSs are big picture—they exploit information from many sensors to determine the current condition of the vehicle. CAN bus data that is inconsistent with the true condition of the vehicle is flagged as anomalous (Wu et al., 2019).

Dupont et al. survey existing work by emphasizing three dimensions; namely, (1) Number of CAN frames, (2) Data leveraged, and (3) Model construction process. The three dimensions refer to the detection window (in number of CAN frames), the features analyzed during the detection procedure (arbitration ID, data length code, etc.), and the techniques used to generate a model of normal traffic (deviations from the model would then be recognized as abnormal traffic), respectively. They observe that some IDSs can detect an attack by analyzing a single message, while others review a fixed window of messages to identify anomalies. A third, middle option assesses pairs of consecutive messages (Dupont et al., 2019).

Young et al. review and compare IDSs by detection feature. They look at IDSs which...

1. Use signatures to detect known attacks
2. Target either message frequency or message interval
3. Utilize the concept of entropy
4. Examine the data fields of CAN frames
5. Augment the CAN bus with attack-detecting sensors
6. Leverage the cyber–physical nature of the CAN bus to detect attacks
7. Implement deep learning to identify attacks

The authors detail the advantages and disadvantages of each method, as well as the mode of evaluation and the types of attacks detected (Young et al., 2019).

While not specific to automotive intrusion detection, Lee et al. analyze and review deep learning IDSs. They provide a taxonomy of deep learning methodologies, as follows:

- Autoencoder (AE)
  - Stacked
  - Denoising
  - Non-symmetric
  - Sparse
  - Variational
  - Convolutional
- Restricted Boltzmann Machine (RBM)
- Deep Boltzmann Machine (DBM)
- Deep Neural Network (DNN)
- Convolutional Neural Network (CNN)
- Generative Adversarial Network (GAN)
- Long Short-term Memory (LSTM)
- Recurrent Neural Network (RNN)
- Hybrid

Lee et al. include a number of graphs to illuminate the trajectory of research on this topic. They highlight the high volume of deep learning schemes that center around autoencoders as well as the prevalence of hybrid approaches. They identified relatively few LSTM-based schemes, even when accounting for the hybrid-LSTM models (Lee et al., 2021). Interestingly, LSTM-based schemes constitute the majority of the automotive IDSs identified during our literature survey.

1.5. Deep learning

Deep learning is a specialization of machine learning that leverages artificial neural networks (ANNs). ANNs are built upon the concepts and processes observed in biological brains. Deep learning architectures can be distinguished from broader machine learning methods by the use of multiple layers in the network. It is the layers that give a deep learning architecture its "depth". Researchers have devised a number of approaches to deep learning, including deep neural networks (DNNs), convolutional neural networks (CNNs), long short-term memory (LSTM), generative pre-trained transformers (GPTs), and generative adversarial networks (GANs) (Arnold et al., 2011).

Fig. 5 illustrates the deep learning process. Initially, training data is provided to the IDS's deep learning model. If needed, the training data may be standardized and sanitized by one or more pre-processing steps. The model is trained on the training data, and once the training is complete, the model is tested. If the model performs acceptably, then the IDS is ready and enters monitoring mode. If not, the IDS's deep learning model is put back into training (retraining). During monitoring mode, the IDS receives automotive network traffic. The IDS evaluates the traffic according to its anomaly detection scheme. If the IDS deems the traffic anomalous, then it will output an alert. If not, the IDS will continue monitoring.

1.5.1. Deep neural network (DNN)

Deep neural networks (DNNs) are a broad class of deep learning architectures. Constructs inspired by biological brains (neurons, synapses) as well as machine learning concepts (weights, biases, functions) are shared by all DNNs. Each DNN has an input layer, an output layer, and multiple layers in between that give it "depth". In the literature, the term "DNN" is generally associated with feed-forward networks. In a feed-forward neural network, data flows from the input layer to the output layer without backtracking or cycling (Arnold et al., 2011; Du et al., 2016). An example of a deep neural network is provided in Fig. 4. The basic architecture of the DNN example—an input layer, one or more hidden layers, and an output layer—can be generalized to all the neural networks described later in this section. However, this particular DNN exemplifies the feed-forward neural network architecture, as the data flows from the input nodes to the hidden nodes to the output nodes without producing a cycle.

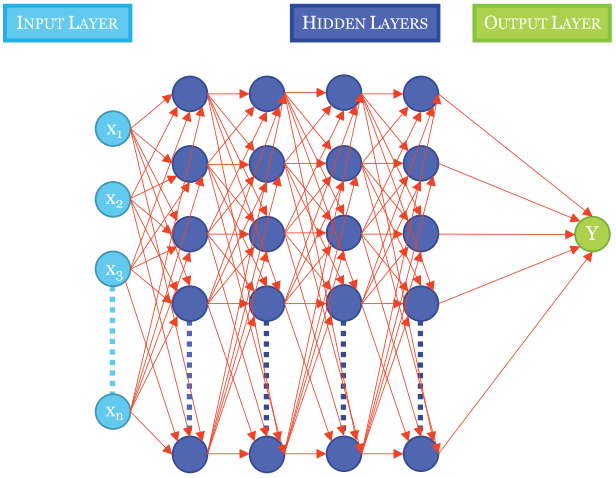

Fig. 4. Illustration of a Deep Neural Network Architecture.

1.5.2. Convolutional neural network (CNN)

Convolutional Neural Networks (CNNs) are popular architectures for problems such as image processing and computer vision (Du et al., 2016; Vargas et al., 2017). Typically, they consist of one or more convolution layers, pooling layers, and fully-connected layers. The convolution layers address feature extraction, while the pooling layers handle feature mapping. In image processing, the CNN becomes more complex with each layer. Earlier layers detect simple features, such as color, while a subsequent layer might recognize shapes and elements. Two-dimensional configurations of CNNs are more common, but one-dimensional CNNs have also proven useful (O'Shea & Nash, 2015).

1.5.3. Long short-term memory (LSTM)

Long Short-term Memory (LSTM) was conceived as an improvement upon recurrent neural networks (RNNs). RNNs are similar to the DNNs described earlier. The main distinction is the existence of directed cycles in RNNs. The output of a given node can cycle back to impact the input to that very node (DiPietro & Hager, 2020). A major limitation of traditional RNNs is susceptibility to the vanishing gradient problem, confining the use of RNNs to short-term memory activities. LSTM addresses the vanishing gradient problem by providing longer-lived short-term memory to preserve information across timesteps. Generally, an LSTM node contains a memory cell, an input gate, an output gate, and a forget gate. The memory cell serves as an information store, while the gates regulate the flow of that information (Du et al., 2016).

1.5.4. Deep transfer learning (DTL)

Deep neural networks are remarkably expensive to train—in terms of computing resources, training data, and time. The idea behind transfer learning is to apply a pre-trained DNN to a new problem. We find a DNN that was trained on a related problem, and we transfer the knowledge associated with that problem to our new problem—hence, "transfer" learning. Because the DNN is pre-initialized with knowledge from the related problem, the training for the new problem will be much less costly (Tan et al., 2018).

1.5.5. Attention & transformer

The concept of "attention" in neural networks was inspired by biological attention. Humans are confronted by overwhelming stimuli, but they focus on the elements that are important. Similarly, neural networks are expected to weigh certain elements of the input data more than others. Smaller elements that are important should be subject to greater attention than unimportant larger elements. The determination

of importance depends on the context of the problem and the training of the neural network (Niu et al., 2021).

Vaswani et al. developed a neural network paradigm dubbed "transformer" that implements the notion of attention. A self-attention scheme is used to calculate the significance of each aspect of the input data. As with all deep learning methodologies, it is composed of multiple layers. In this case, the layers are subdivided into a feed-forward network layer and a multi-head attention layer. The feed-forward network layer tracks the positions of the elements in the input sequence, while the multi-head attention layer stacks self-attention layers, empowering the model to focus on information from different representations of the input and output (Niu et al., 2021; Vaswani et al., 2017).

A generative pre-trained transformer (GPT) (Nam et al., 2021) is an autoregressive language model that leverages deep learning and attention to generate sentences. It was developed by OpenAI and builds upon the transformer model.

1.5.6. Generative adversarial network (GAN)

A generative adversarial network (GAN) is composed of two competing neural networks, a generator and a discriminator. The generator strives to "generate" realistic input for the discriminator, while the discriminate tries to "discriminate" between real input elements and generator-crafted input elements. The generator is trained to fool the discriminator, while the discriminator is trained to recognize the generator's work. The process is essentially a zero-sum game; one neural network's gain is the other's loss (Seo et al., 2018; Xie et al., 2021).

1.6. Key points

I The Controller Area Network (CAN) bus is extraordinarily vulnerable due to its lack of authentication, encryption, and integrity.
II Deep learning intrusion detection systems (IDSs) can combat the insecurity of the CAN bus by detecting attacks—even novel attacks—with high accuracy.
III Existing survey and review work has provided valuable insight into topics such as automotive IDSs and deep learning IDSs—but not the intersection of the two.

The remainder of the paper is organized as follows: Section 2 highlights related work in automotive security, Section 3 explores related work in automotive intrusion detection, Section 4 describes deep learning in automotive intrusion detection at length, Section 5 reviews open challenges on this topic, and Section 6 concludes our discussion.

2. Related work in automotive security

Considerable research has been devoted to addressing security concerns in automotive networks. A number of CAN bus security techniques are enumerated below:

1. Intrusion Detection Systems
2. Intrusion Prevention Systems
3. Authentication
4. Encryption
5. Honeypots
6. Firewalls & Filtering
7. Dynamic Arbitration Identifiers
8. Network Segmentation

One major issue with the CAN bus is lack of authentication and authorization. By the current design, every message goes to every ECU, and every message is assumed to be legitimate. At the time that the CAN bus was devised, automotive networks were closed systems and trust made sense. Now, researchers are developing authentication techniques. One challenge is that often, authentication requires major—and costly—changes in existing CAN nodes. More powerful hardware is needed, and there may be issues with latency. If keys are used, then protecting and updating the keys becomes a challenge. Even with authentication, there may be issues with replay: a signed message to

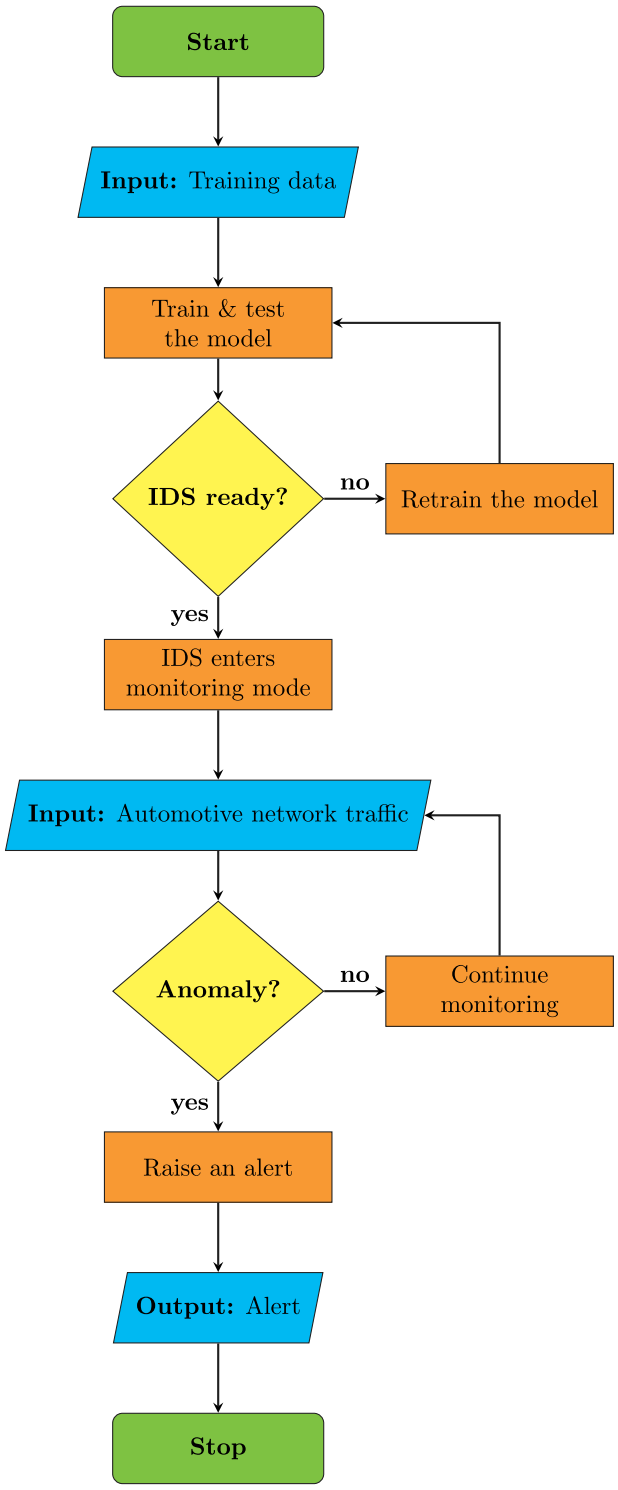

Fig. 5. Flowchart of the Deep Learning Process.

apply the brakes could be captured by an adversary and sent again, as it would still be signed.

Wang and Sawhney developed a practical, low-latency authentication technique dubbed "VeCure". It divides CAN nodes into trust groups, where each trust group shares a common symmetric key. This technique reduces the number of symmetric keys needed, but, by design, every message must be followed by an authentication message, which doubles the traffic volume. Additionally, if a node inside the trust group is compromised, then the whole trust group is compromised, since they share the symmetric key (Bozdal et al., 2020; Wang & Sawhney, 2014).

Groza et al. devised LiBrA-CAN, which uses the principles of key splitting and MAC mixing, such that authentication keys are divided between multiple groups of nodes. They also leverage systems of linear equations to construct message authentication codes (MACs) (Groza et al., 2012). The practical applicability of this method is constrained by its bandwidth demands and its incompatibility with conventional CAN (Bozdal et al., 2020).

Groza and Murvay constructed a broadcast authentication protocol that does not require pre-shared keys, either symmetric or asymmetric. Instead, the protocol leverages key chains and time synchronization to exploit symmetric primitives without pre-shared symmetric keys. The keys are updated each time a CAN message is broadcast. The price of authentication is a delay—which largely stems from the key disclosure delay—as well additional overhead on the bus (Groza & Murvay, 2013; Kim et al., 2021).

The lack of encryption in the CAN bus system allows attackers to read and analyze CAN messages and develop an understanding of the communication. Encryption would help protect automotive networks from this type of analysis. Doan and Ganesan crafted an FPGA chip that secures data transmitted on CAN FD systems. The chip uses AES-128, SHA-1, and HMAC to provide encryption, integrity, and authentication. The design has been modeled, verified, and fabricated in a simulation, and its performance is sufficient for ECUs. However, it is incompatible with classic CAN, and CAN FD is not yet popular in modern automobiles (Doan & Ganesan, 2017).

Siddiqui et al. proposed mutual authentication and encryption for the CAN bus using physical unclonable functions (PUFs). This technique builds the keys from physical features of the ECUs, and the keys are rebuilt each time the vehicle starts. The proposed technique eliminates the need to maintain and hide keys. However, it requires extortionate modifications to the ECUs (Bozdal et al., 2020; Siddiqui et al., 2017).

Verendel et al. advocated the use of honeypots to simulate in-vehicle networks and capture the behavior of an attacker. The honeypot would allow access via wireless interface, similar to the wireless interfaces of many modern vehicles. The honeypot should present as an authentic automotive network with functional ECUs and typical communications (Verendel et al., 2008). This technique faces a major hurdle in the form of realism; if the honeypot is not realistic, the attacker will not be fooled (Lokman et al., 2019). There is also a danger that, if the honeypot is not properly isolated, the attacker could escape the honeypot and attack real systems.

Lemke, Paar, and Wolf contemplated an automotive security technique based on the concept of firewalls. Safety-critical systems should be protected by firewalls that allow authorized messages and filter out unauthorized messages. Lemke, Paar, and Wolf pointed out that diagnostic interfaces usually enable access to all systems; as such, in order for the firewall to function effectively, diagnostic interfaces must be inaccessible during normal operations (Lemke et al., 2006; Lokman et al., 2019). Firewalls and filtering schemes require modifications to the prevailing implementation of the CAN system; indeed, firewalls and filtering schemes are themselves modifications to the network.

Many attacks on the CAN bus depend on traffic analysis and replay; that is, the attacker monitors the traffic, determines which arbitration ID sends a command to a given ECU, and then re-sends the desired command. Dynamic arbitration IDs help prevent these types of attacks by changing the arbitration IDs such that by the time the attacker sends a message, the arbitration ID is wrong. Islam and Refat devised a dynamic arbitration ID scheme and conducted an evaluation in a simulated environment. They were able to detect attacks in less than half a second (Islam & Refat, 2020). This work is promising, but it does assume that the attacker does not know about the dynamic arbitration IDs and does not attempt to counter the technique. This scheme would also place a significant re-engineering burden on automotive manufacturers.

Woo et al. developed a moving target defense (MTD) scheme for the CAN protocol. Similar to Islam and Refat (2020), the scheme shuffles arbitration IDs, hindering a would-be attacker's reconnaissance efforts and inhibiting simple replay attacks. What was once a static environment becomes a dynamic environment. Moreover, the scheme does not necessitate changes in the conventional CAN protocol, and the authors' performance evaluation indicated that the scheme will not increase the bus load (Kim et al., 2021; Woo et al., 2019).

Network segmentation is a basic tenant of security in the virtual world. If two networks are isolated, then the compromise of one does not automatically result in the compromise of the second. It is common practice in CAN systems to separate the safety-critical components from functionalities such as infotainment and climate control. One would think that network segmentation would be an obvious design choice. However, automotive manufacturers often wish to put data from the safety-critical components on the dashboard or infotainment screen. In many cases, to achieve this functionality, they connect the infotainment system, often the biggest attack surface in the vehicle, to the safety-critical systems. In fact, this particular design decision was exploited by Miller and Valasek in the 2015 Chrysler Jeep Hack (Miller & Valasek, 2015a, 2015b). Proper isolation of safety-critical components from systems that connect to the outside world is an important step in assuring the safety of the vehicles occupants. Even though segmentation does not prevent attack, it can turn a fatal attack into a nuisance attack. If the attacker gains access to the low-speed CAN system that is not safety-critical, he or she might turn up the radio to an obnoxious volume. But without access to the safety-critical CAN bus, the attacker cannot cut the brakes. That said, network segmentation, when implemented properly, increases costs and maintenance effort (Bozdal et al., 2020).

Argus (Shamah, 2015) offers security-enhanced ECUs and automotive security solutions that can be built into the vehicle at assembly time, but these offerings would be suitable for the manufacturer, not the consumer.

2.1. Key points

I. Considerable research work has been devoted to CAN bus security schemes such as authentication, encryption, honeypots, firewalls & filtering, dynamic arbitration identifiers, and network segmentation.
II. In general, authentication and encryption offer major security improvements—but at the cost of higher bus loads, communication delays, or even re-engineering of the CAN protocol.
III. In contrast, honeypots, firewalls & filtering, and dynamic arbitration identifiers are often more practical but contribute less to security.

3. Related work in automotive intrusion detection

This section describes a number of traditional automotive intrusion detection systems, which we have divided into the following categories:

- Interval
- ID Sequence
- Graph
- Specification
- Entropy

Table 1
Overview of traditional automotive intrusion detection systems.

Detection method	Name of IDS	Description
Interval	Song et al. (2016)	For each arbitration ID, the IDS captures the time interval between one message and the next. If the time interval is less than half of the established "normal," then an anomaly score is incremented. When the anomaly score reaches a preset threshold, an alert is raised.
Interval	Gmiden et al. (2016)	For each arbitration ID, the IDS captures the time interval between one message and the next. If the time interval is less than half of the established "normal," an alert is immediately raised.
Interval	OTIDS (Lee et al., 2017)	The IDS captures a window of messages from request to response. Given this window, the IDS determines the time interval and the offset—the number of intervening messages—between request and response. If the time interval and offset are irregular, then the traffic is anomalous.
Interval	Taylor et al. (2016)	Over a sliding window, the IDS tracks the average time between packets and the number of packets. If the values over a given window are inconsistent with historical values, then the traffic is anomalous.
ID sequence	Marchetti and Stabili (2017)	The IDS monitors transitions from one arbitration ID to the next. If the IDS sees a transition that is not present in the matrix of "normal" transitions, then the traffic is anomalous.
ID sequence	IDS for CAN (Lampe & Meng, 2022)	The IDS captures a sequence of arbitration IDs and checks each transition from one arbitration ID to the next. If the IDS sees a transition that is not present in the matrix of "normal" transitions, then the traffic is anomalous.
ID sequence	Dönmez (2021)	The IDS monitors k-length sequences of arbitration IDs. If the IDS sees a sequence that is not present in the queue of historical "normal" sequences, then the traffic is anomalous.
Graph	Islam et al. (2020)	The IDS captures a window of CAN messages and graphs them. The graph is compared to a baseline distribution, and if the distributions do not belong to the same population (according to the chi-squared test), then the traffic is anomalous.
Specification	Markovitz and Wool (2017)	The IDS follows a field classification procedure to develop specifications for CAN data fields. If the IDS identifies a CAN frame that violates the specifications, then the frame is an anomaly.
Specification	Larson et al. (2008)	Security specifications for the CAN protocol are developed a priori and provided to the IDS. If the IDS identifies a CAN frame that violates the specifications, then the frame is an anomaly.
Entropy	Wang et al. (2018)	The IDS calculates the entropy of arbitration IDs at the bit level to construct a template of "normal" entropy. If the bit-level fluctuation of the current CAN traffic exceeds the template's thresholds, then the traffic is anomalous.
Entropy	Wu et al. (2018)	The IDS calculates the entropy of arbitration IDs over a sliding window. If the entropy computation does not fall within the established "normal" range, then the traffic is anomalous.
Entropy	Müter and Asaj (2011)	The IDS tracks the conditional self-information and entropy of the CAN traffic at the bit level. If the conditional self-information or entropy falls outside the established "normal" range, then the traffic is anomalous.

An IDS is an *interval*-based IDS if it monitors either inter-message intervals or message frequencies in order to detect anomalies. An *ID sequence*-based IDS observes sequences of arbitration IDs and determines if the sequences are normal or anomalous. A *graph*-based IDS represents CAN traffic graphically and compares the current CAN traffic to an established baseline, whereas a *specification*-based IDS compares the current CAN traffic to some form of pre-defined specification. Lastly, we define an *entropy*-based IDS as an IDS which leverages the concept of entropy to detect anomalies.

Table 1 summarizes the automotive intrusion detection methodologies discussed in this section.

3.1. Interval-based intrusion detection

Researchers have developed a number of intrusion detection systems for the CAN bus relating to intervals. Some of these IDSs are described as "timing-based", while others are "interval-based" or "frequency-based". In all cases, the concept is the same—the IDS is dependent upon timing information, which it uses to determine if traffic is normal or abnormal. An IDS might monitor the time intervals between messages, or it might count the number of messages received during a set interval. The former would be called timing- or interval-based, while the latter is usually referred to as frequency-based (Lokman et al., 2019).

Song, Kim, and Kim developed a time interval-based intrusion detection system for the CAN bus. For each arbitration ID they observed, they analyzed the time interval between messages and established a baseline. In general, they found that a given arbitration ID would have a 0.1 time interval during normal traffic. While the CAN bus was under attack, the time intervals shortened to nearly 10% of the normal time interval. They noted that the proposed IDS could detect message injection attacks in milliseconds (Song et al., 2016). A similar scheme was also proposed by Gmiden et al. (2016).

IDSs that depend entirely on timing can often be defeated by sending attack packets at the expected fixed intervals. Furthermore, timing-based IDSs do not detect irregularities in the arbitration ID or the data field—merely the timing (Gmiden et al., 2016; Lee et al., 2017; Song et al., 2016; Taylor et al., 2016). Lee, Jeong, and Kim developed OTIDS—offset ratio and time interval-based intrusion detection system—which, as the name suggests, leverages the offset ratio and time interval between the request sent out from the bus and the response from the network (Lee et al., 2017). Unfortunately, this technique suffered from serious data loss due to the extensive hardware integration needed to implement the technique (Lokman et al., 2019).

Frequency-based IDSs are also dependent on accurate timing information, though perhaps to a lesser extent. Perhaps the technique used to capture CAN messages is inconsistent: messages accumulate in a queue and then are suddenly processed. In such cases, a frequency-based IDS would see relatively few messages, then a spike, then relatively few messages again, followed once more by a spike. Careful planning and implementation of the message capture system can prevent these issues, but it is certainly a limitation of this type of IDS. Taylor, Japkowicz, and Leblanc designed a frequency-based IDS that monitors multiple features of the CAN traffic during a specified interval. They selected a 1-s window to conduct an analysis of the current traffic, which was compared to historical analyses. In addition to the average time difference between packets during the given interval, they also tracked the number of packets, the average and variance of the Hamming distance between successive packet data fields, and several related metrics. Ultimately, they concluded that the Hamming distance of successive data packets could not reliably differentiate normal and abnormal traffic. The timing statistics were more promising, but the

authors noted that even these statistics could not detect anomalies at practical false positive rates (Taylor et al., 2016).

3.1.1. Analysis

Timing-related IDSs tend to be lightweight, and for periodic arbitration IDs, they are generally very reliable. However, some arbitration IDs are aperiodic. In many vehicles, one or more arbitration IDs handle cruise control functionalities. When cruise control is not engaged, these arbitration IDs appear infrequently, if at all. When cruise control is engaged, frequency jumps while time interval contracts. If the driver is incrementing or decrementing the cruise control setting to reach the desired speed, then the impact on frequency and time interval will be even more pronounced; many messages will be produced by the driver's actions, and once the desired speed is achieved, then frequency will drop while time interval expands. Such phenomena can yield false positives. Implementing thresholds in timing-related IDSs can reduce false positives but could potentially increase false negatives. Further, IDSs of this type can often be defeated by attack messages that are sent at the expected intervals, especially since the data field is never considered. Even if an attack does impact time interval or frequency, it might not cause enough fluctuation to trigger an alert.

3.2. Sequence-based intrusion detection

In sequence-based intrusion detection, the IDS analyzes the sequence of arbitration IDs crossing the CAN bus to detect abnormal traffic. In the training phase, the IDS establishes a baseline by capturing normal traffic and determining which arbitration IDs appear in succession. In the monitoring phase, the IDS reads sequences of identifiers off the CAN bus and confirms that they are known, normal patterns. If a pattern appears that is not found in the baseline, then it is flagged as anomalous.

Marchetti and Stabili constructed an ID sequence-based IDS that emphasizes transitions—that is, transitions from one arbitration ID to the next arbitration ID. A square matrix is declared; its rows and columns are the unique arbitration IDs observed in normal traffic. The matrix is initialized to `FALSE`. During the training phase, pairs of consecutive arbitration IDs are captured, and the matrix is updated such that the position corresponding to that pair of arbitration IDs is now `TRUE`. In the monitoring phase, the IDS again captures pairs of consecutive arbitration IDs. It checks that both arbitration IDs are found in the matrix. If not, the validation fails. Next, it checks that the pair of arbitration IDs represents a valid transition; that is, the value at the corresponding position in the matrix is `TRUE`. The memory requirement for this IDS is an $n * n$ square matrix, where n is the number of unique arbitration IDs, as well as an ID-position array, which correlates a position in the matrix to an arbitration ID. The ID-position array is of length n, and each element requires either 11- or 29-bit of memory, depending upon the size of the arbitration ID (Marchetti & Stabili, 2017). This type of IDS does not depend on timing information, but it can be hampered by packet loss. Packet loss in the training phase might add invalid transitions to the matrix, leading to false negatives. Packet loss in the monitoring phase might lead to false positives, as the packet that should exist between two consecutive packets has been dropped. If $A \rightarrow B$ and $B \rightarrow C$ are valid transitions but $A \rightarrow C$ is not a valid transition, then the loss of packet B will result in two anomalies. Lampe and Meng implemented this type of ID sequence-based IDS on an Android device (Lampe & Meng, 2022).

Dönmez developed an ID sequence-based IDS that builds upon the scheme described by Marchetti and Stabili. Dönmez reasoned that longer ID sequences would increase the sensitivity of the IDS and opted to use k-length sequences instead of pairs. Increasing k from two to three enhanced the sensitivity of the IDS but also marginally increased the false positive rate (Dönmez, 2021).

3.2.1. Analysis

ID sequence-based IDSs are not computationally expensive, and the memory requirements, even for vehicles with 100+ arbitration IDs, are quite reasonable. However, this technique depends upon predictable sequences generated by predictable driving behavior. If driving behavior during the monitoring phase deviates from driving behavior during the training phase, then false positives will be raised. For example, if cruise control was not used during training, then the arbitration IDs associated with that functionality will not be captured in the matrix—or queue—of "normal" sequences. The obvious mitigation is to activate the cruise control functionality during training, but this type of mitigation is infeasible in some cases. If the functionality missing from the matrix of "normal" sequences is associated with the vehicle's automated emergency features—emergency alert, emergency stop, etc.—then it may be difficult or dangerous to trigger. In any case, it would be difficult to capture every corner case during training. Moreover, as the name suggests, ID sequence-based IDSs consider a CAN frame's arbitration IDs but not its data field. This type of IDS would be unlikely to detect a compromised ECU.

3.3. Graph-based intrusion detection

Graph-based intrusion detection represents captured CAN messages as a graph or graphs. Graphs are used to record the normal behavior of the system, and in the monitoring phase, each newly-generated graph is compared to the baseline. If the graphs are sufficiently similar, then the new traffic is deemed normal. If the new graph deviates significantly, then it is deemed anomalous. Islam et al. conceived a graph-based IDS to improve upon ID sequence-based IDSs, which the authors found to be vulnerable to intelligent attacks. The graph-based IDS captures 200 raw CAN messages and graphs them. The authors refer to the 200 raw CAN messages as a single "window". This process is repeated until the IDS has sufficient data to represent the distribution of the windows, which the authors refer to as a "population window". This distribution is the baseline for the monitoring phase, in which newly graphed windows are compared to the base distribution using a base hypothesis and a chi-squared test. The authors found the base distribution to be normally distributed. DoS attacks resulted in a positively-skewed distribution, while fuzzing and spoofing attacks generated bimodal distributions. Even to human eyes, the difference between normal traffic and these types of attacks is readily apparent. However, replay attacks were shown to be normally distributed and indistinguishable from normal traffic. To improve prediction accuracy, the authors leverage the median test. Ultimately, the IDS was able to detect and classify DoS, fuzzing, spoofing, and replay attacks with a misclassification rate of 4.76% (Islam et al., 2020). The implementation described by Islam et al. shares a limitation with ID sequence-based IDSs: it does not consider the data field. If an adversary were to send spoofed CAN frames with suitable IDs and malicious data fields, this type of IDS would fail to detect them.

3.3.1. Analysis

Because graph-based intrusion detection leverages more data to assess the normality or abnormality of a window of CAN traffic, it is more resilient when confronted with fluctuations in attack-free traffic. As such, false positives are more readily managed and mitigated. The trade-off is detection latency: a pair of consecutive messages can be assessed much faster than a window of 200 messages. Further, this type of IDS is incapable of detecting replay attacks, as they mimic normal traffic when graphed. As with interval- and ID sequence-based IDSs, graph-based schemes do not consider the data field and cannot detect data tampering attacks.

3.4. Specification-based intrusion detection

A specification- or classification-based intrusion detection system derives semantic information from a CAN frame's data field and splits that data field accordingly. Once the IDS identifies and classifies portions of the data field, it builds models of these field types.

Markovitz and Wool developed a field classification-based IDS by analyzing CAN traces and developing a suitable field splitting and classification algorithm. Analysis allowed them to divide CAN data fields into five categories: Constant, Multi-Value, Counter, Sensor, and none of the above. Values that never deviated during the trace were deemed "Constant". "Multi-Value" fields exhibited only a few different values. "Counter" values appeared to be incrementing a counter, and "Sensor" values appeared to be reporting sensor data. Next, the field splitting and classification algorithm iterates over each arbitration ID and generates a field with a *type* and a set of *properties* associated with that *type*. The *properties* value of a Constant field is the observed constant value, while a Multi-Value field stores all values seen. For Counters and Sensors, the minimum and maximum values are preserved. Simulated CAN bus traffic yielded a median false positive rate of 1%, while real CAN bus traffic achieved a median false positive rate of 0% (Markovitz & Wool, 2017). The experimental results are promising, but the semantic IDS requires a substantial, complex training process, which involves the field detection, splitting, classification, and modeling described above. Moreover, this process would need to be repeated for each vehicle type.

Larson, Nilsson, and Jonsson describe a somewhat different specification scheme, in which specifications for the expected CAN message structure are determined a priori. Specifications can be made at the protocol level as well as for the data fields of specific arbitration IDs. If the value of data field of a given arbitration ID is expected to fall within a pre-defined range, then that range can be added as a specification—and enforced. The authors' experimental results showed that the specification-based IDS could detect most types of attacks, but the authors noted several limitations, such as the injection of valid values. Valid values would meet the specifications and would not be detected, even if they were used to conduct an attack (Larson et al., 2008).

3.4.1. Analysis

Specification-based IDSs prescribe the content and structure of the data field; as such, they can detect data tampering attacks that interval-, ID sequence-, and graph-based IDSs cannot. However, this type of IDS tends to emphasize the expected structure of *individual* CAN frames, not the CAN traffic as a whole. As such, a specification-based IDS would raise an alert if it saw an invalid value in the CAN data field, but it would not raise an alert if it saw a valid value that was inappropriate for the driving conditions. For example, if the specification stipulates that the arbitration ID associated with brake application is `0AA` and the valid values for the data field range from `0` (no brake pressure) to `100` (full brake pressure), then a message with arbitration ID `0AA` and value `200` would be flagged as anomalous. However, if the CAN bus is suddenly flooded with CAN frames containing ID `0AA` and value `0`, then the IDS would fail to detect an attack, even as the vehicle races toward a red light. Markovitz and Wool did not evaluate their proposed IDS in attack conditions, and Larson, Nilsson, and Jonsson acknowledged several detection-related limitations of their scheme.

3.5. Entropy-based intrusion detection

An entropy-based IDS detects anomalies in the *entropy* of traffic. Entropy can be envisaged as a measure of the randomness of data. In the context of automotive networks, the "data" is the traffic on the CAN bus. Some entropy-based IDSs focus exclusively on the arbitration IDs, while others factor in the data field as well. Entropy-based IDSs have been explored outside automotive networks, but they have not seen much success. The extreme variability in normal traffic on the internet and on computer networks will often yield fluctuating, even erratic, entropy values. By contrast, automotive networks are stable, predictable. The number of actors—that is, ECUs—is limited, as the number and format of valid messages. As such, an entropy-based IDS is much more viable on an in-vehicle network (Müter & Asaj, 2011; Wang et al., 2018).

Entropy can be computed at several data abstraction layers, such as the binary layer, the signal layer, and the protocol layer. The binary layer is the lowest level of abstraction; it is a binary stream. At the protocol layer, we have all the fields of a CAN frame. Many entropy-based IDSs focus on the ID field; others inspect both the ID field and the data field. Several information-theoretic measures can help capture the entropy of the CAN traffic. Conditional self-information measures how much information content has been transmitted with a message. Entropy is the expected value of conditional self-information, which measures the uncertainty (or randomness) of the traffic. Relative entropy measures the distance between two datasets in terms of entropy (Müter & Asaj, 2011).

Wang, Lu, and Qu propose an ID-based entropy IDS. With experimentation, they were able to determine that entropy distribution in normal driving conditions is sufficiently stable. If the drive turns cruise control on and off or adjusts the radio, the fluctuation in entropy will be noticeable but negligible. The entropy fluctuations that appear organically during normal driving operations can be distinguished from the fluctuations indicative of an attack—because an attack will generally have significantly greater impact on entropy. Attacks on the CAN bus typically require the adversary to send a high volume of malicious CAN packets in order to overwhelm the legitimate messages issued by real ECUs. As such, a threshold can be set to differentiate normal variation in entropy from an attack (Wang et al., 2018). However, an adversary who disabled an ECU and then masqueraded as that ECU would not need to send a high volume of malicious traffic and could subvert this type of entropy-based IDS.

Wu et al. developed a similar ID-based entropy IDS that incorporates a sliding window and the concept of simulated annealing. In the Wu et al. IDS, entropy is calculated for a pre-determined interval—either a specified number of messages or a specified period of time. This interval is the "sliding window". When one window ends, the next window begins. By analyzing the entropy calculations for each window, the IDS can determine if the CAN traffic is normal or abnormal. The sliding window in this entropy-based IDS serves to address entropy interference issues, such as entropy interference that stems from differences in the baud rate and aperiodic CAN messages. The sliding window can be optimized using a heuristic algorithm that builds upon the concept of simulated annealing (Wu et al., 2018). This IDS, as with many of the others described in this section, is limited by its neglect of the data field. If an adversary manipulates CAN data fields without significantly altering the pattern of arbitration IDs, then this type of IDS would fall short. Furthermore, Ohira et al. describe a specialized DoS attack, referred to as an "entropy-manipulated attack" which adjusts the entropy of arbitration IDs to circumvent a sliding window entropy-based IDS, such as the one proposed by Wu et al. To conduct such an attack, the adversary would select various high-priority arbitration IDs and flood the bus. Because the arbitration IDs are higher priority than normal IDs, they will inhibit legitimate communications, and because multiple arbitration IDs are used in the attack, the randomness—and entropy—of the traffic will be preserved (Ohira et al., 2020).

Müter and Asaj developed an entropy-based IDS that accounts for both the ID field and the data field when it computes an entropy value. In one scenario, the speed of vehicle fluctuates significantly during normal driving operations as the vehicle's operator accelerates rapidly, stops abruptly, and so on. As a result of the changing speed, the calculated entropy varies, but it does not exceed the threshold and produce a false positive. However, when the experimenters inject four spoofed speed values into the system, three of those four spoofed speed values produce spikes on a graph of entropy, indicating that they

could be detected. The value that was not detected differed by only one kilometer from the true speed. The authors noted that spurious data that is close enough to normal behavior cannot be detected using this approach. They also recognized that an entropy-based IDS would need to account for the vehicle status—accessory mode (vehicle on, engine off), idling (engine on, not moving), driving (engine on, vehicle in motion), etc. Many ECUs do not send messages while the vehicle is in accessory mode, and many others send the same message (such as "speed is zero") over and over again. The entropy would be markedly different and could lead to false positives if vehicle status were not considered (Müter & Asaj, 2011).

3.5.1. Analysis

When an entropy-based IDS analyzes the arbitration IDs but not the data fields, then it cannot detect a data-tampering attack. Some of the schemes outlined above exhibit this weakness. However, M′uter and Asaj described an entropy-based approach that evaluates the data field and can detect this type of attack. That said, entropy-manipulated attacks could still prove problematic. Entropy-based IDSs evaluate the entropy of the CAN traffic, not the actual values. An adversary could generate attack traffic that matches the "normal" entropy profile for both the arbitration ID and the data field. If the IDS analyzes the entropy of the arbitration ID, then the attack described by Ohira et al. would evade detection (Ohira et al., 2020). If the IDS also analyzes the entropy of the data field, then the adversary would also need to customize the data field to fit the "normal" entropy profile. For a DoS attack, the content of the data field does not matter, so the adversary could simply pick values that yield a suitable entropy. If the vehicle speed is 20 miles per hour and the adversary wishes to spoof 50 miles per hour, then he or she could gradually increment spoofed speed until 50 is achieved.

3.6. Advantages & disadvantages

A significant advantage of specification-based IDSs and some entropy-based IDSs over interval-, ID sequence-, and graph-based IDSs involves the CAN frame's data field. Specification- and entropy-based IDSs can detect data-tampering attacks by accounting for the data field. Even if the attacker compromises an ECU and forces it to send (1) an expected arbitration ID (2) at the expected interval and (3) in the expected sequence, abnormalities in the data field could still be detected. However, specification-based IDSs only know if a given data field is appropriate for the given arbitration ID. They know each arbitration ID, each subdivision of its data field, and a set or range of valid values. Specification-based IDSs lack the "big picture" view that is essential to detecting replay attacks. If the attack message matches the specification, then the IDS cannot detect it. The inherent limitation of entropy-based IDSs is similar: if the entropy of the data field is appropriate, even if the actual value is not, then the IDS cannot detect it. Each type of IDS described in this section is vulnerable to some form of intelligent attack. For interval-based IDSs, an intelligent adversary would manipulate the timing of the attack in order to evade detection. For ID sequence-based IDSs, an intelligent adversary would ensure that an attack message accommodates the expected sequence. For the graph-based IDS, an intelligent adversary would need to manipulate the attack traffic such that it is normally distributed. Specification-based IDSs merely require that the attack traffic matches known, valid values. To fool an entropy-based IDS, an entropy-manipulated attack, as described by Ohira et al. would suffice (Ohira et al., 2020). Intelligent attacks are not far-fetched, either. CAN traffic is unencrypted—anyone who can access the bus can sniff the traffic. By observing normal traffic, the adversary can discern its intervals, its sequences, its entropy, etc. Armed with this information, the adversary conducts an intelligent attack.

3.7. Key points

I. A number of traditional intrusion detection systems—interval-, ID sequence-, graph-, specification-, and entropy-based—have been applied to automotive environments.
II. Traditional IDSs tend to be lightweight in terms of computation and memory; however, they also tend to have detection shortcomings.
III. For each type of traditional IDS described above, there exists an intelligent attack which would evade detection.

4. Deep learning in automotive intrusion detection

This section details a number of deep learning automotive intrusion detection systems, which we have organized into the following categories:

- Deep Neural Networks (DNNs)
- Convolutional Neural Networks (CNNs)
- Long Short-term Memory (LSTM)
- Deep Transfer Learning (DTL)
- Attention & Transformer
- Generative Adversarial Networks (GANs)

Section 1.5 describes the architecture and features of the deep learning schemes we will explore in this section.

Table 2 summarizes the automotive intrusion detection methodologies discussed in this section.

4.1. DNN-based intrusion detection

Sami et al. describe an intrusion detection system architected as a fully-connected feed-forward neural network (FFNN) with two hidden layers and one output layer. They optimize the FFNN-based IDS in terms of the number of neurons in the hidden layers, the batch size, and the activation functions. Note that feedback is not allowed in FFNN; a given layer can never have dependencies on itself. Sami et al. opt for supervised learning: two datasets are curated and pre-processed, and for each CAN frame in the two datasets, a label is applied to distinguish normal and attack frames. One of the datasets was obtained from the Hacking Countermeasures and Research Lab (HCRL), while the second dataset was collected from a Mercedes ML350 via its OBD-II port. The authors explored three types of attacks: DoS, fuzzing, and impersonation. The FFNN-based IDS was exposed to DoS and fuzzing attacks during training, then it was expected to detect the impersonation attack during the evaluation. The FFNN-based IDS demonstrated high accuracy when pitted against both the HCRL and ML350 datasets. Accuracy for HCRL was 0.9999, while accuracy for ML350 was 0.9860. Precision, recall, and F1-scores for both datasets were also very high; for all of these metrics, the value was 0.99+ for detection thresholds between 0.1 and 0.95 (Sami et al., 2020).

Zhang and Ma present a hybrid of a rule-based IDS and a deep learning IDS. They sought to exploit the advantages of each type of IDS—the rule-based IDS is more efficient, while the deep learning IDS is much more accurate but also much more computationally expensive. Zhang and Ma leverage feature engineering tactics to lower the computational cost of the deep learning model while simultaneously avoiding overfitting. They filter out low variance features, as such features will not be decisive, and they delete redundant features. Ultimately, the authors settled on the following five features: (1) Arbitration ID, (2) Hamming distance between the data fields of two consecutive messages, (3) Entropy of the data field, (4) Byte of highest importance, and (5) Byte of second highest importance. The selected features are manufacturer and model dependent, which the authors note as a limitation. Feature extraction and training are required for each new manufacturer and model of vehicle. Zhang and Ma leverage feature

Table 2
Overview of deep learning automotive intrusion detection systems.

Detection method	Name of IDS	Key features
DNN	NESLIDS (Sami et al., 2020)	Feed-forward neural network, supervised learning
DNN	Zhang and Ma (2022)	Hybrid rule- and DNN-based detection, feed-forward neural network
CNN	Hossain et al. (2020a)	1-dimensional CNN, binary and multiclass variants, supervised learning
CNN	Song et al. (2020)	2-dimensional CNN, supervised learning
CNN	Ahmed et al. (2021)	VGG-16 architecture, transfer learning
LSTM	Hossain et al. (2020b, 2020c)	One LSTM, binary and multiclass variants, supervised learning
LSTM	CANet (Hanselmann et al., 2020)	One LSTM per arbitration ID, unsupervised learning
LSTM	MLIDS (Desta et al., 2020b)	One LSTM per arbitration ID, stateful
LSTM	Desta et al. (2020a)	ID sequence-based detection
LSTM	Jedh et al. (2021)	Graph-based detection, hybrid RNN and LSTM architecture
LSTM	CANnolo (Longari et al., 2020)	Hybrid LSTM and autoencoder architecture, unsupervised learning
LSTM	NovelADS (Agrawal et al., 2022)	Hybrid LSTM and CNN architecture, unsupervised learning
LSTM	Zekry et al. (2021)	Hybrid LSTM and CNN architecture, supervised learning
DTL	Mehedi et al. (2021)	Four DTL architectures (FCN, IncepNet, ResNet, and LeNet), supervised learning
Attention & Transformer	NasrEldin et al. (2021)	Attention-based model, one single-head attention layer and one self-attention layer
Attention & Transformer	Nam et al. (2021)	Generative pre-trained transformer, one multi-head self-attention layer
GAN	GIDS (Seo et al., 2018)	Discriminator: 3-layer deep neural network; generator: 5-layer deconvolutional neural network
GAN	Yang et al. (2021)	Discriminator: 2 convolutional layers and 2 linear layers; generator: 2 convolutional layers and 1 linear layer
GAN	Xie et al. (2021)	OEM's CAN communication matrix is built into the discriminator

Table 3
Summary of DNN-based automotive intrusion detection systems.

Name of IDS	Accuracy	Precision	Recall	TPR	FPR	F1
NESLIDS (Sami et al., 2020)	0.9999[a]	0.99992	0.999977			
Zhang and Ma (2022)	0.9991[b]			0.999	0.00066	0.999948464

[a]HCRL dataset, threshold of 0.5.
[b]Strong adversary, dataset #1.

vectors to embed the five features into the deep learning IDS, which processes CAN traffic and monitors variations in the features to detect anomalies. Meanwhile, the rule-based element of the IDS was given two objectives: rapid detection and minimization of false positives. With the rule-based element, the authors check that the arbitration ID is valid (it is trivial to collect all valid arbitration IDs for a given vehicle). Invalid IDs indicate abnormal traffic. Similarly, the authors note that, per the CAN protocol specifications, a valid data length code (DLC) should be between zero and eight bytes, inclusive. Furthermore, messages with the same arbitration ID should share the same DLC. If the DLC for a given arbitration ID is inconsistent, then the inconsistency would be indicative of abnormal traffic. In general, CAN messages of a particular arbitration ID are sent periodically; therefore, time intervals can also be used as rules. By combining multiple techniques, such as a rule-based IDS (which encompasses specification- and interval-based IDSs) and a deep learning IDS, Zhang and Ma are able to overcome many of the limitations of the individual schemes. The authors evaluate the approach against six types of attacks: (1) Random ID injection attack, (2) Zero-ID injection attack, (3) Replay attack, (4) Spoofing attack, (5) Drop attack, and (6) Masquerade attack. The rule-based mechanism achieved a false positive rate of 0.00005, an average true positive rate of 0.8325, and an average accuracy of 0.8883. The authors noted that the somewhat low accuracy stemmed from the rule-based mechanism's inability to detect masquerade attacks. The DNN-based mechanism attained an average true positive rate of 0.8167 and an average accuracy of 0.8772; the DNN was hampered by its inability to detect drop attacks. When the rule-based and DNN-based mechanisms were combined, the IDS consistently achieved a true positive rate and accuracy above 0.99, even when the strength of the adversary varied from weak to medium to strong (Zhang & Ma, 2022).

Deep neural networks have been applied to automotive security strategies beyond intrusion detection—such as CAN bus authentication. Xiao et al. developed a deep reinforcement learning mechanism to extract authentication features and reduce latency in the authentication process. They engage two lightweight DNNs capable of analyzing and leveraging message features at the physical layer, such as messaging intervals and signal voltages (Xiao et al., 2021) (see Table 3).

4.1.1. Analysis

The two DNN-based IDSs discussed in this section ascribe to the feed-forward neural network architecture. As such, signals move in one direction—forward—and backtracking is strictly prohibited. The seeming simplicity of FFNNs is often cited as an advantage; however, they still have many parameters and, thus, require optimization in order to function most effectively. The main drawback of FFNNs is lack of memory—because they are acyclic, they cannot remember or learn from previous input and previous iterations of training. That said, both DNN-based IDSs performed very well on various metrics—accuracy, precision, recall, F1-score, etc. On paper, Zhang and Ma's hybrid IDS (Zhang & Ma, 2022) would have more detection capability overall: the rule-based component can detect attacks that pure DNN-based IDSs might miss. The complexity and sophistication of DNN—and deep learning in general—seems to lend itself to the detection of sophisticated attacks, while flagrant attacks (fuzzing, invalid arbitration IDs, etc.) can evade detection. The rule-based component of Zhang and Ma's hybrid IDS is a lightweight, inexpensive addition that precludes straightforward, overt attacks.

4.2. CNN-based intrusion detection

Hossain et al. detail two variations of a one-dimensional convolutional neural network-based intrusion detection system. One variation

Table 4
Summary of CNN-based automotive intrusion detection systems.

Name of IDS	Accuracy	Precision	Recall	TPR	FPR	F1
Hossain et al. (2020a)	0.999856[a]		0.9993		0.0001	0.9997
Song et al. (2020)		0.9999[b]	0.9994			0.9996
Ahmed et al. (2021)	0.95			0.96	0.006	

[a]Multiclass classifier, Toyota.
[b]RPM spoofing attack.

is a binary model with one convolutional layer; the second is a multiclass model with two convolutional layers. In the pre-processing phase, the authors convert a hexadecimal CAN traffic dataset to decimal. They concentrate on ten features: the arbitration ID, the data length code (DLC), and the eight bytes of the data field, each of which is treated as a distinct feature. Both the binary and multiclass paradigms monitor the CAN traffic and detect unusual fluctuations in the values of the features. They evaluated the two CNN-based IDSs on four attacks: (1) DoS, (2) fuzzing, (3) RPM spoofing, and (4) gear spoofing. The binary model and the multiclass model both excelled at DoS and spoofing attack detection; each model achieved 100% accuracy when pitted against a Toyota, a Subaru, and a Suzuki. For the fuzzing attack, however, the models achieved 99+%, but never 100%. While 99+% is exceptional, it is noticeably lower than the accuracy values achieved in all the other attack scenarios. Hossain et al. implemented the fuzzing attack by injecting random arbitration IDs between 0 × 000 and 0x7FF—with random lengths and random data field contents. An entropy-based IDS would detect the spike in randomness. An ID sequence-based IDS would detect the abnormal ID sequences. A specification-based IDS would recognize that the arbitrary data does not match the expected data for the given arbitration ID. All of these IDSs are less complex in terms of implementation and resources, yet they excel where the CNN-based IDS appears to fall short (Hossain et al., 2020a).

Song, Woo, and Kim developed a two-dimensional CNN-based intrusion detection system. Much of the literature surrounding CAN bus security addresses the 11-bit CAN protocol, which is standard for consumer automobiles (sedans, vans, SUVs, light trucks, etc.). However, Song, Woo, and Kim focus on 29-bit CAN (also called "extended CAN"), which is implemented in heavier vehicles as well as commercial and industrial machinery (buses, semi-trailer trucks, agricultural tractors, etc.). In the approach outlined by Song, Woo, and Kim, the deep learning component is supervised; the CNN uses labeled CAN traffic for the training data but unlabeled CAN traffic for the detection phase. The CAN frames are labeled one if they constitute an attack and zero if they are attack-free. During the training phase, arbitration IDs are extracted and assembled into CAN frames of 29 consecutive arbitration IDs. Next, a 29 * 29 matrix is constructed. Each full row of the matrix is a single arbitration ID represented in binary, such that the 29 bits of the 29-bit arbitration ID fill in all the columns. CNNs are typically used for two-dimensional image processing, so the authors formulated the 29 * 29 matrix as a sort of image. Instead of pixels, the rows and columns characterize arbitration IDs. Song, Woo, and Kim leverage a popular image classification solution, Inception-ResNet, which they have adapted to suit the automotive intrusion detection problem. Rather than categorizing an image into one of a thousand classes, the CNN-based IDS needs to classify a 29 * 29 matrix into either normal or abnormal traffic. The IDS was tested offline, though the authors noted that, given the ongoing improvements in computing power, the IDS could be capable of real-time detection in the future. During the evaluation, Song, Woo, and Kim observed the training loss of the proposed IDS on each type of attack. Training loss indicates how well the CNN model fits the training data and should converge toward zero. The DoS attack saw the earliest convergence of training loss, while the fuzzing attack saw the latest convergence. In the best case scenario, the false negative rate and the error rate were less than 0.1% for the DoS attack and the two spoofing attacks, but they were considerably higher for the fuzzing attack—0.24% for the false negative rate and 0.18% for the error rate. As with the CNN implementation described by Hossain et al. it would seem that fuzzing attacks are challenging to CNNs, even though they are one of the easier attacks for simple IDSs—entropy-, ID sequence-, and specification-based—to recognize (Song et al., 2020).

Ahmed, Ahmad, and Jeon leveraged a convolutional neural network, specifically VGG-16, to detect DoS and fuzzing attacks. "VGG" is an acronym for Visual Geometry Group, and the "16" refers to the number of convolutional layers—16 (Parashar, 2020). Detection accuracy for DoS and fuzzing attacks was 0.95. For the attack-free state, accuracy increased to 0.96 and the false positive rate was 0.006. The authors compared the VGG-16 CNN-based IDS to the following paradigms: K-Nearest Neighbor (KNN), Random Forest (RF), Gradient Boosting, AdaBoost, and Support Vector Machine (SVM). The detection accuracy of the alternative paradigms ranged from 0.93 to 0.95, while the false positive rates varied between 0.04 and 0.05, inclusive. While the accuracy values are comparable, VGG-16 demonstrates a significantly better false positive rate than the others (Ahmed et al., 2021) (see Table 4).

4.2.1. Analysis

Generally speaking, convolutional neural networks have been optimized for the image processing use case, and, typically, the input is expected to be two-dimensional. While CAN traffic can be vectorized and adapted into an image format, it is not natively pictorial. As such, many of the CNN-based IDSs described in this section impose an extra step—some type of transformation from traffic capture to images. Notably, Hossain et al. (2020a) opted for a one-dimensional CNN and, as such, did not need to convert the data to a two-dimensional format—though they did convert from hexadecimal to decimal. In terms of metrics, Hossain et al.'s one-dimensional CNN-based IDS performed exceedingly well, achieving 99.9+% accuracy on all attacks. Further, this IDS incorporates the arbitration ID, the data field, and the DLC into its analysis, so it is theoretically capable of detecting data tampering attacks. By contrast, the two-dimensional IDS described by Song et al. (2020) focuses purely on the arbitration IDs, so it would not detect data tampering attacks. However, converting the CAN traffic captures into two-dimensional images enables the authors to leverage CNNs that have been specially optimized for two-dimensional input and image processing tasks. Ahmed et al. (2021) selected the VGG-16 architecture and transformed the CAN traffic captures into byteplots to achieve the required dimensionality. While the IDS performed well, it has been noted that VGG can be very slow to train and is particularly vulnerable to the exploding gradients problem.

4.3. LSTM-based intrusion detection

Hossain et al. describe an automotive intrusion detection system built upon the long short-term memory (LSTM) deep learning architecture. Much of the approach overlaps with the authors' proposed CNN-based IDS (Hossain et al., 2020a), which was explored in the previous section; the main difference is the deep learning mechanism that underlies the intrusion detection system—LSTM vs. CNN. Hossain et al. investigate both binary and multiclass classification models for the LSTM. The authors designate 11 features to be fed into the two models: arbitration ID, DLC, data field (each of the eight bytes is regarded as

Table 5
Summary of LSTM-based automotive intrusion detection systems.

Name of IDS	Accuracy	Precision	Recall	TPR	FPR	F1
Hossain et al. (2020b, 2020c)	0.999949[a]	1.0000	0.9998	0.9998	0.00004	
CANet (Hanselmann et al., 2020)	0.996[b]			0.852		
MLIDS (Desta et al., 2020b)		1.0[c]	1.0			1.0
Desta et al. (2020a)		0.968[d]	0.896			0.93
Jedh et al. (2021)	0.98[e]					
CANnolo (Longari et al., 2020)		0.9017[f]	1.0000			0.9483
NovelADS (Agrawal et al., 2022)		0.9991[g]	0.9990			0.9991
Zekry et al. (2021)		0.98[h]	0.97			0.97

[a]Optimized LSTM.
[b]Real data, $H_{scale} = 5$.
[c]prv_data (test vehicle), insertion attack.
[d]Insertion attack.
[e]Pearson correlation.
[f]Real-world attacks.
[g]Model M2, RPM spoofing attack.
[h]Model II, without orientation sensors, RPM spoofing attack.

a separate feature), and a label (“benign”, “DoS”, “fuzzing”, or “spoofing”). The labels correspond to the five attack scenarios that the models will encounter during the evaluation: (1) attack-free, (2) DoS, (3) fuzzing, (4) handle angle spoofing, and (5) speed spoofing. The binary classifier achieved perfect 1.0000 accuracy for the DoS and spoofing attacks, and 0.9998 for the fuzzing attack (Hossain et al., 2020b). The result is reminiscent of the CNN-based IDS also proposed by Hossain et al. in which the IDS saw a decline in performance when faced with a fuzzing attack (Hossain et al., 2020a). In an effort to optimize the multiclass LSTM, the authors varied the number of layers in the LSTM (1, 2, 3, 4, and 5), the optimizer (Adam, Adagrad, Adadelta, Adamax and Nadam), the learning rate of the optimizer (0.0001, 0.001, 0.01, and 0.5), the activation function (sigmoid, relu, and tanh), and the loss function (categorical_crossentropy, MAE, MSE, and KL_divergence). Adam and Nadam effectuated the highest accuracy values for the binary and multiclass classification schemes, respectively (Hossain et al., 2020c). Ultimately, the authors architected a multiclass LSTM with sigmoid for the activation function, categorical_crossentropy for the loss function, one layer, and 0.0001 as the learning rate for the Nadam optimizer. The multiclass LSTM was able to achieve perfect detection accuracy for DoS and spoofing attacks—no false positives, no false negatives. The fuzzing attack proved comparably difficult, with 0.00002 for the false positive rate and 0.0007 for the false negative rate. Though the rates are small, they are still non-zero. In fact, we can see an interesting trend in deep learning automotive intrusion detection systems: fuzzing attacks are proving more of a challenge than DoS, RPM spoofing, gear spoofing, handle angle spoofing, and speed spoofing attacks. The authors note that the LSTM-based IDS is more efficient than some approaches because it does not need to decode raw CAN packets to recognize attacks. As such, the authors believe it will translate to any type of automobile (Hossain et al., 2020b, 2020c).

Hanselmann et al. fashioned a novel unsupervised LSTM-based IDS in which each arbitration ID is processed by an independent LSTM. In this manner, the temporal dependencies specific to each arbitration are explored and preserved by the processing LSTM. Ultimately, the outputs of the independent LSTMs coalesce into a fully connected network; as such, the interdependencies of all the arbitration IDs, as well as the identifiers’ respective values, are considered. The proposed approach centers around the concept of a “signal”. The authors characterize a signal as a single semantic value that is encoded in one or more bytes of the CAN data field. They analyze arbitration IDs to determine which signal or signals are encoded in the data field, such as acceleration, vehicle speed, or gear (park, drive, reverse, etc.). An anomaly score quantifies the normality—or abnormality—of the CAN traffic. To calculate the anomaly score, all potential input signals are reconstructed at every timestep. The difference—or error—between the reconstructed values and the true values determines the anomaly score. If the reconstruction error exceeds the threshold, then the anomaly score is 1; else, 0. The threshold and the anomaly score are per arbitration ID. If any arbitration ID has a non-zero anomaly score, then the CAN traffic is deemed anomalous. Because the temporal dependencies are tracked by independent LSTM models, the authors note that this scheme is not sensitive to the precise ordering of sequential arbitration IDs. They cite this fact as an advantage of the scheme, since there are aperiodic arbitration IDs that result in variations in the order of IDs. However, insensitivity to the ID sequence would suggest that anomalies that appear in ID sequences—but not in the data patterns of individual identifiers—would not be detected. Thus, there is a loss of detection capability with the loss of sensitivity to sequencing. Hanselmann et al. evaluated the proposed LSTM-based IDS against five types of attacks: (1) plateau, (2) continuous change, (3) playback, (4) flooding, and (5) suppress. In a *plateau attack,* exactly one signal is repeatedly overwritten to a constant value, such that the signal appears to be frozen at that value. In a *continuous change attack,* an adversary overwrites a selected signal with an incrementally different value, such that the signal gradually drifts from its true value. We would expect this type of attack from an adversary who wishes to detection: by limiting the adjustments to small, realistic increments, an adversary can push the signal toward a desired constant value. A *playback attack* is essentially a replay attack, in which the adversary overwrites a signal value by replaying previously captured values for that signal. The idea behind the attack is to fool the IDS with valid values that reflect very different operating conditions. A *flooding attack* is exemplified by the injection of a high volume of CAN traffic with the same arbitration ID. In a *suppress attack,* an adversary manages to disable a particular ECU, such that messages with that ECU’s arbitration ID disappear from the bus traffic. In much of the literature, the attacks fabricated to evaluate an automotive IDS are variations of the following: DoS, fuzzing, replay, spoofing, and masquerade. The attack compilation described by Hanselmann et al. contains attacks that are novel, subtle, and indicative of an adversary who seeks to evade detection. The authors reported that, generally, the accuracy of the LSTM-based IDS was 0.99 or higher for normal traffic, between 0.85 and 0.95, inclusive, for both plateau and playback attacks, and 0.70 or higher for the continuous change attack. They did not report the numbers for the flooding and suppress attacks, but they mentioned that flooding attacks were detected with a high true positive rate, while the LSTM-based IDS struggled with suppress attacks (Hanselmann et al., 2020).

Desta et al. developed an LSTM-based IDS that, similar to earlier work by Hanselmann et al. provides a separate LSTM for each arbitration ID. In the pre-processing phase, the arbitration ID is used to sort all the packets, and then the hexadecimal packets are converted to binary. The authors analyze packets by window; they select a one-second window for the proposed IDS, though they note that the time interval or “size” of the window is adjustable. The model is stateful;

as such, it is capable of generating predictions for any interval. For each arbitration ID, the LSTM-based IDS analyzes the packets received during the specified window and forecasts the packets that should be captured during the next window. As the next window occurs, the IDS compares the captured packets with the predicted packets to calculate the anomaly score for the given arbitration ID. The overall anomaly score is the weighted average of the anomaly scores computed for each arbitration ID. The proposed LSTM-based IDS was evaluated against an attack-free dataset, a DoS attack, a fuzzing attack, an RPM spoofing attack, and a gear spoofing attack. Given a detection window of one second, the model was able to achieve perfect 1.0 scores for both precision and recall for all attacks. The authors note that a one-second delay in detection could be catastrophic in a safety-critical system such as the automotive CAN bus, and they plan to reduce the detection window—without reducing accuracy—in future work (Desta et al., 2020b) (see Table 5).

4.3.1. ID sequence-based LSTM

Desta et al. describe an ID sequence-based IDS that leverages long short-term memory to look back at the previous 20 arbitration IDs in order to presage the next arbitration ID. The IDS emits a prediction every 20 arbitration IDs. The authors' subject vehicle had 42 unique arbitration IDs crisscrossing its CAN bus; however, many modern vehicles have significantly more ECUs and arbitration IDs, even double the 42 identified in the authors' subject vehicle. In addition, the number of ECUs and arbitration IDs in contemporary automobiles continues to rise. Prediction accuracy may well be impacted when the number of possible IDs for the prediction is higher. In any case, when the authors programmed the LSTM to simply predict the next arbitration ID, the accuracy value was a mere 0.6. As such, the authors opted to utilize log loss. If the log loss exceeded a pre-determined threshold, then the CAN traffic would be deemed anomalous. The log loss technique furnishes an anomaly assessment every 100 arbitration IDs—instead of every 20 arbitration IDs, as in the original approach—because the anomaly assessment is an average of the preceding five log loss computations (20 * 5 = 100). Desta et al. experiment with an insertion attack, a drop attack, and an illegal ID injection. The F1-scores were 0.913, 0.952, and 1.0, respectively (Desta et al., 2020a).

4.3.2. Graph-based LSTM

Jedh et al. propose a graph-based intrusion detection system that combines three detection techniques: (1) Thresholds, (2) Change Point Detection (CPD), and (3) Long Short-term Memory (LSTM). Sequences of CAN frames are captured in directed graphs, and the similarities of consecutive graphs are quantified using the concepts of cosine similarity and Pearson correlation. Cosine similarity is calculated as the cosine of the angle between two vectors. It is frequently employed to measure document similarity for the purposes of plagiarism detection. Meanwhile, Pearson correlation is a calculation of the linear correlation between two vectors. It is cited as the de facto standard when it comes to comparing two datasets. Jedh et al. also explored t-test and Levene's test, but they found them to be far less accurate. The similarity metrics are calculated for two consecutive "windows"—pre-defined intervals of successive CAN frames. The authors use a sliding window of 100 sequential CAN frames. The evaluations were conducted on attack-free, RPM spoofing, and speed spoofing datasets. Jedh et al. leverage three detection schemes of increasing complexity. Of least complexity are the thresholds. The authors selected 0.87 as a suitable threshold for both cosine similarity and Pearson correlation. The threshold scheme ran into issues with false positives: in the attack-free dataset, the false positive rate for cosine similarity was 0.106, while the false positive rate for Pearson correlation was 0.132. Across the attack-free, RPM spoofing, and speed spoofing datasets, accuracy values ranged from 0.868 to 0.973. The approach dependent on Pearson correlation exhibited a wider range in accuracy and false positive rate than its cosine similarity counterpart. At the next level of complexity, the change point detection method showed potential, but when using both cosine similarity and Pearson correlation, CPD wrongly detected a change point in the attack-free dataset. For the LSTM-based detection mechanism, cosine similarities and Pearson correlations were computed for each pair of consecutive time windows. Depending on window size, the accuracy of cosine similarity for the two spoofing attacks was between 0.7343 and 0.9845 for speed spoofing and between 0.3793 and 1.000 for RPM spoofing. When using Pearson correlation, the experimental evaluation demonstrated that accuracy values varied between 0.7343 and 1.000 for speed spoofing and between 0.3790 and 1.000 for RPM spoofing (Jedh et al., 2021).

4.3.3. LSTM + autoencoders

Longari et al. developed an unsupervised intrusion detection system utilizing LSTM autoencoders. An LSTM autoencoder is an autoencoder that implements LSTM as its learning layers. The proposed LSTM autoencoder-based IDS does not require a priori knowledge of CAN traffic semantics. Instead, the IDS anticipates future CAN traffic (reconstruction) and identifies anomalies by evaluating the difference (error) between the reconstruction and the actual CAN traffic. First, Longari et al. pre-process the raw CAN traffic: for each 64-bit CAN frame, they calculate the rate at which each bit changes value and use this information to divide the CAN frame into blocks that are considered individual pieces of data. Recall that autoencoders compress or "encode" data; then, they decode it once more. In the compression process, data that uninformative or redundant is discarded, while apposite data is preserved. As such, the LSTM autoencoder-based IDS determines which features of the CAN data are relevant to its predictions. The baseline for the intrusion detection system is the distribution of the reconstruction error in attack-free traffic. The reconstruction error is calculated as the difference between the reconstructed and actual CAN traffic. Then, the Mahalanobis distance between the current reconstruction error and the distribution of reconstruction error in attack-free traffic is computed. If the Mahalanobis distance exceeds the pre-determined threshold, then the traffic is deemed anomalous. Longari et al. evaluated the LSTM autoencoder-based IDS in four scenarios: (1) interleave attack, (2) discontinuity attack, (3) data field attack, and (4) attack-free traffic. In an *interleave attack,* the adversary transmits spoofed messages on the CAN bus without deactivating the real ECU; this type of attack results in confliction. In a *discontinuity attack,* the adversary disables an ECU; as such, the arbitration IDs associated with that ECU will disappear from the CAN traffic, resulting in discontinuities. The final attack type is a *data field attack,* which involves the manipulation of the data field alone. A value in the data field might be set to its minimum or maximum value, a constant, a random value, or a replayed value that was previously captured (Longari et al., 2020). Longari et al. use Area Under the Curve (AUC) as the evaluation metric for the LSTM autoencoder-based IDS. AUC plots the true positive rate (sensitivity) and the false positive rate (specificity). An AUC of 1.0 is perfect and an AUC of 0.0 is 100% wrong (Developers, 2022). The average AUC across arbitration IDs was 0.9986 for interleave attack, 0.9777 for discontinuity attack, and 0.9677 for data field attack (Longari et al., 2020). The authors mention that they did not consider attacks involving invalid CAN frames. Certain fields in a CAN frame cannot be altered without invalidating the packet. The bits of the cyclic redundancy checksum (CRC) field, for example, are fixed. If the CRC field is invalid, then the packet is invalid (Bozdal et al., 2020). Longari et al. assert that this type of attack is trivial and should be detected by a resource inexpensive rule-based IDS, not a deep learning IDS (Longari et al., 2020).

4.3.4. LSTM + CNNs

Agrawal et al. describe an intrusion detection system that combines convolutional neural networks and long short-term memory to extract spatio-temporal features and long-term dependencies from automotive CAN traffic. LSTM fulfills the reconstruction functionality, while CNN

is used to capture veiled spatial features. The attributes fed into the IDS are the timestamp, arbitration ID, data field, data length code (DLC), and a label. Attack data is labeled "T", while attack-free data is labeled "R". Agrawal et al. utilize sliding windows to compare sequences of CAN traffic. If the CAN traffic is anomalous, then we expect to see unusually high reconstruction error compared to the historical reconstruction error during attack-free traffic. The traffic is deemed abnormal if it exceeds the threshold. Rather than a preset threshold, the authors developed a novel thresholding scheme to calculate a suitable threshold for the reconstruction error between the predicted traffic and the actual traffic. The thresholding scheme is independent of the trained model and can be used for any anomaly detection procedure. The threshold for the reconstruction error is learned during the training phase, which also teaches the model how to most accurately predict or "reconstruct" future CAN traffic. Agrawal et al. evaluated the approach against four types of attacks: (1) DoS, (2) fuzzing, (3) RPM spoofing, and (4) gear spoofing.

In addition, the authors tested four distinct architectures:

1. Single-layer LSTM
2. CNN succeeded by single-layer LSTM
3. Two-layer stacked LSTM
4. CNN succeeded by two-layer stacked LSTM

In all four attack scenarios—DoS, fuzzing, RPM spoofing, gear spoofing—the architecture composed of CNN succeeded by single-layer LSTM outperformed all the others in terms of precision, recall, and F1-score. The precision, recall, and F1-scores for the "CNN succeeded by single-layer LSTM" architecture ranged from 0.9989 to 1.0, inclusive (Agrawal et al., 2022).

Zekry et al. fabricated a convolutional long short-term memory-based IDS specifically for the problem of automotive intrusion detection in connected autonomous vehicles (CAVs). CAVs are vehicles capable of detecting and communicating with nearby IoT devices of every class and function. Information gleaned from nearby devices empowers CAVs to make shrewd decisions and take appropriate actions. However, this interconnectivity opens up CAVs to all manner of attacks or even malfunctions. As such, it is prudent to develop detection capabilities—specifically, the capability to detect corrupt data in the vehicle navigation system and the powertrain (engine, transmission, and drivetrain). To capture data, Zekry et al. mounted a mobile phone on a vehicle and tapped into the same vehicle's OBD-II port. The phone collected data points such as latitude, longitude, speed, heading, etc., while the OBD-II port provided diagnostic information, including engine load, engine RPM, fuel flow rate, etc. The data rate for the mobile phone was 25 Hz, while the data rate for the OBD-II port was 1 Hz. As such, the two datasets had to be synchronized during post-processing. The data, when passed into the IDS, was multi-dimensional, forming rows, columns, and channels. The data for one single sensor at one single timestep was represented two-dimensionally. The data was stacked in a zigzag pattern in order to preserve continuity in the two-dimensional space. The third dimension represented all of the sensors. For the detection phase, the authors selected a sliding window of 16 timesteps. For the evaluation, Zekry et al. constructed anomalies that could occur during either attacks or malfunctions:

1. **Frozen-at-Upper-Limit:** An anomaly in which a value that becomes stuck when it reaches the upper limit of the acceptable range. When this range is exceeded, the upper limit is still reported.
2. **Frozen-at-Initial-Value:** An anomaly in which the initial value of the signal is reported continually, even as the true value is changing.
3. **Spike:** A sudden, extreme change in the value of the signal, which quickly returns to normal.
4. **Jump:** A sudden, extreme change in the value of the signal, which does not return to normal. The change becomes a fixed offset to the true value.

Zekry et al. experimented with two different configurations of the CNN-LSTM IDS. For both configurations, an RPM spoofing and a speed spoofing attack were conducted, as well as a combined attack (RPM and speed were spoofed). Both models performed better when faced with one attack than with two. The authors remarked that anomalies become more difficult to distinguish when they become more frequent. Of the two models and the three attack types, the lowest F1-score was 0.82, while the highest was 0.97 (Zekry et al., 2021).

4.3.5. Analysis

As the name would suggest, long short-term memory provides a longer lived "short-term memory" compared to traditional recurrent neural networks; the goal of this long "short-term memory" is to combat the vanishing gradient problem. In addition to this advantage, if an IDS supplies one LSTM model per arbitration ID, then it can better preserve the temporal dependencies specific to that arbitration ID. This is the main advantage of IDSs proposed by Hanselmann et al. (2020) and Desta et al. (2020b) when compared to Hossain et al.'s single LSTM scheme (Hossain et al., 2020b, 2020c). That said, the single LSTM architecture is lighter and less complex.

Desta et al. (2020a) described an ID sequence-based LSTM, which looks back at previous arbitration IDs in order to predict future arbitration IDs. The downfalls of this type of IDS are scalability—especially as the number of arbitration IDs in modern CAN traffic continues to increase—and vulnerability to data tampering attacks. An ID sequence-based IDS, even augmented with LSTM, is still an ID sequence-based IDS. It does not analyze the data field and cannot detect attacks in which the sequences match expectations. However, Jedh et al.'s graph-based intrusion detection system (Jedh et al., 2021) differs from its non-learning counterpart. While the graph-based IDS described in Section 3 focuses exclusively on the arbitration IDs (Islam et al., 2020), Jedh et al.'s graph-based LSTM graphs both a CAN packet's arbitration ID and data field. As such, the IDS would be capable of detecting data-tampering attacks. Unfortunately, the graph-based LSTM demonstrated a relatively high false positive rate, which calls its practicality into question.

The remaining LSTM-based IDSs covered in this section are actually hybrids; Longari et al. (2020) described a hybrid LSTM autoencoder-based IDS, while Agrawal et al. (2022) and Zekry et al. (2021) developed hybrid LSTM- and CNN-based IDSs. An LSTM supplemented with an autoencoder has the advantage of better data; the autoencoder can discard uninformative data while preserving useful data. This process reduces the amount of data that the LSTM must process, and the remaining data should be more informative. Meanwhile, the combination of LSTMs and CNNs can capture spatio-temporal features and long-term dependencies. Neither architecture could fully leverage these features alone. While a hybrid IDS can exploit the strengths of several techniques, it is also more complex. Each component has its own parameters, which must be properly tuned and optimized for the problem at hand. If the hybrid IDS does not fit the problem, then it is likely to bring more complications than benefits.

4.3.6. DTL-based intrusion detection

Deep learning IDSs, and even traditional machine learning techniques, necessitate vast quantities of training data; all too often, such training data is not available for proprietary IVNs. As such, Mehedi et al. sought to transfer knowledge from a suitable source domain to the problem of intrusion detection on the CAN bus—the target domain. The authors call attention to five features: the timestamp, arbitration ID, DLC, data field, and label. Some approaches treat each of the eight bytes of the data field as distinct features; Mehedi et al. regard the data field as one feature. The authors present a deep transfer

Table 6
Summary of DTL-based automotive intrusion detection systems.

Name of IDS	Accuracy	Precision	Recall	TPR	FPR	F1
Mehedi et al. (2021)	0.9810	0.9814	0.9804			0.9783

learning-based LeCun Network (LeNet) as the backbone of an intrusion detection system for the CAN bus (Mehedi et al., 2021). LeNet is a particular convolutional neural network architecture developed by Lecun et al. (1998). Mehedi et al. consider three types of attacks—flooding, fuzzing, and spoofing—which they combine into one dataset. As such, there is an attack-free dataset and a triple-attack dataset. Before evaluation, the data is extensively pre-processed in three phases: (1) data cleaning, (2) data integration, and (3) data transformation. In the data cleaning phase, noise and inconsistencies are expunged from the dataset. Rosner's test detects outliers which need to be removed, and predictive mean matching identifies missing values. Next, the authors convert qualitative values into quantitative values. Then, in the data integration phase, they removed data that is not informative—that is, redundant data. Lastly, the data transformation phase normalizes the data. Mehedi et al. evaluated four traditional machine learning architectures, four deep learning architectures, and four deep transfer learning architectures:

1. Traditional Machine Learning (TML)
    (a) Decision Tree (DT)
    (b) Random Forest (RF)
    (c) Support Vector Machine (SVM)
    (d) K-Nearest Neighbor (KNN)
2. Deep Learning (DL)
    (a) Neural Network (NN)
    (b) Recurrent Neural Network (RNN)
    (c) Convolutional Neural Network (CNN)
    (d) Long Short-term Memory (LSTM)
3. Deep Transfer Learning (DTL)
    (a) Fully Convolutional Networks (FCN)
    (b) Inception Network (IncepNet)
    (c) Residual Neural Network (ResNet)
    (d) LeCun Network (LeNet)

Among traditional machine learning schemes, the decision tree proved superior. Accuracy, precision, and recall scores were 0.9532, 0.9463, and 0.9558, respectively. The LSTM architecture outperformed all deep learning models in terms of accuracy, precision, and recall. The values were 0.9762 for accuracy, 0.9808 for precision, and 0.9392 for recall. The LSTM model supersedes the decision tree (and all the traditional machine learning approaches evaluated) as well. Upon evaluation of the deep transfer learning IDSs, LeNet excelled with an accuracy of 0.9810, a precision of 0.9814, and a 0.9804. LeNet had the highest accuracy and the highest recall of all deep transfer learning IDSs, though FCN had a higher precision at 0.9832. Overall, the deep transfer learning schemes eclipsed both the traditional machine learning and deep learning architectures evaluated (Mehedi et al., 2021) (see Table 6).

4.3.7. Analysis

The core advantage of deep transfer learning is well known: transferring knowledge from the target domain to a pre-trained deep neural network. Because the deep neural network is pre-trained, it requires less data from the target domain and also less training time. Furthermore, in terms of overall performance, Mehedi et al. demonstrated that deep transfer learning supersedes both traditional machine learning and deep learning for the architectures evaluated (Mehedi et al., 2021).

4.4. Attention & transformer-based intrusion detection

NasrEldin, Bahaa-Eldin, and Sobh developed an intrusion detection system that capitalizes on the concept of attention, which is used in Natural Language Processing (NLP) and is the building block of the "transformer" deep learning architecture. The proposed attention-based IDS is composed of five main layers: (1) an input embedding and positional encoding layer, (2) a feed-forward network, (3) a single-head attention layer, (4) a self-attention layer, and (5) a softmax output layer. Note that there are two distinct attention layers. Layer 1 enables the IDS to concentrate on important features and disregard unimportant features—this layer determines importance by computing the attention between the input features and the target. Layer 2 is a self-attention layer, which assesses the relationships between all of the features in the data. The inherent advantage of this approach over LSTM is that an attention-based IDS can process long sequences of data in parallel, whereas LSTM processes short sequences in fixed-size windows. The authors outline four attacks for the purposes of experimental evaluation: DoS, fuzzing, RPM spoofing, and gear spoofing. Precision, recall, and F1-scores for the two spoofing attacks were perfect 1.000s. For the DoS attack, precision was 0.999, recall was 1.000, and F1-score was 1.000. The fuzzing attack was marginally more difficult to detect than the others; precision, recall, and F1-scores were 0.993, 0.998, and 0.995, respectively (NasrEldin et al., 2021).

Nam, Park, and Kim construct a transformer-based IDS—specifically, a generative pre-trained transformer (GPT)—to detect attacks such as flooding, fuzzing, replay, and spoofing. Transformer architectures fit the pattern of encoder/decoder schemes facilitated by the attention technique, and they are regularly employed to generate sentences. The layers of a GPT network include (1) a word and positional embedding layer, (2) a masked multi-head self-attention layer, (3) a feed-forward layer, and (4) a softmax layer to estimate sentence probability. Nam, Park, and Kim interpret a sequence of arbitration IDs as a sentence and leverage two generative pre-trained transformers to learn the pattern of a "sentence" of normal CAN traffic. The arbitration IDs are the words which compose the sentence. With experimentation, the authors determined that a sequence length of 256 arbitration IDs was optimal for the GPT-based IDS. The two GPTs are linked in a bi-directional manner, enabling the IDS to assess both past and future arbitration IDs. When it comes to detecting attacks, the authors discovered that two bi-directional GPTs will appreciably outperform one uni-directional GPT, even when very few arbitration IDs are involved in the attack. The bi-directional architecture fortifies the prediction performance of the IDS for all arbitration IDs no matter where they are in the sequence. As such, in the proposed approach, one of the GPT networks is backward, one is forward, and the outputs are concatenated. The GPT-based IDS minimizes the negative log-likelihood (NLL) value of a normal arbitration ID sequence. When the NLL for the sequence exceeds a predetermined threshold, then it is classified as an intrusion. The F1-scores for the four attacks were as follows: 0.9897 for flooding, 0.9881 for fuzzing, 0.9860 for replay, and 0.9524 for spoofing (Nam et al., 2021) (see Table 7).

4.4.1. Analysis

Attention and transformer-based IDSs are somewhat similar to autoencoders in the sense that they are fed a high volume of input, and they have to concentrate on the part of the input that is important. Autoencoders discard uninformative input, while attention and transformer-based schemes generally "weigh" some pieces of the input more than others. As such, attention and transformer-based IDSs are better equipped to select the data that can best differentiate normal traffic and attack traffic. Compared to LSTM, such IDSs are better suited to processing long sequences of data, since LSTM is constrained to short sequences and fixed-size windows. NasrEldin, Bahaa-Eldin, and Sobh's IDS (NasrEldin et al., 2021) has one single-head attention layer and one self-attention layer, while Nam, Park, and Kim's IDS (Nam et al.,

Table 7
Summary of attention & transformer-based automotive intrusion detection systems.

Name of IDS	Accuracy	Precision	Recall	TPR	FPR	F1
NasrEldin et al. (2021)		1.00[a]	1.00			1.00
Nam et al. (2021)				0.9372[b]	0.005	0.9524

[a]RPM spoofing attack.
[b]Spoofing attack.

Table 8
Summary of GAN-based Automotive Intrusion Detection Systems.

Name of IDS	Accuracy	Precision	Recall	TPR	FPR	F1
GIDS (Seo et al., 2018)	0.980[a]	0.983	0.990			
Yang et al. (2021)	0.998[b]	0.997	0.999			0.998
Xie et al. (2021)		0.997[c]	0.999			0.998

[a]Second discriminator, RPM spoofing attack.
[b]Table 6, RPM spoofing attack.
[c]Injection attack.

2021) has one multi-head self-attention layer. Multi-head attention has the advantage of learning multiple "interpretations" of the data—each head can index the data differently. That said, single-head attention, when layered, can yield similar benefits (Lin et al., 2022; Niu et al., 2021). However, Nam, Park, and Kim's IDS is bi-directional; as such, it can learn from both past and future arbitration IDs. However, the IDS is also completely dependent upon the arbitration IDs and cannot detect data tampering attacks.

4.5. GAN-based intrusion detection

Seo, Song, and Kim describe a Generative Adversarial Network (GAN)-based IDS, composed of a discriminator and a generator. The generator is a three-layer deep neural network, while the discriminator is a five-layer deconvolutional neural network. In pre-processing stage, the authors leveraged one-hot vectors to convert sequences of arbitration IDs into images. From there, they evaluated two discriminators. Discriminator 1 was trained using known attack data, while discriminator 2 was trained using normal CAN traffic—in the form of images—and images generated by the generator. As such, discriminator 1 had no adversarial component; meanwhile, discriminator 2 competed with the generator. During the adversarial process, discriminator 2 sought to distinguish between normal CAN images and generator-crafted images, while the generator strove to craft images that would fool the discriminator. The authors conducted four attacks: DoS, fuzzing, RPM spoofing, and gear spoofing. Ultimately, discriminator 1's detection capability depended on its training dataset. If it was trained on the DoS dataset, then it could reliably detect DoS attacks, but not fuzzing or spoofing attacks. The same was true if it was trained on fuzzing attacks, RPM spoofing attacks, or gear spoofing attacks. In many cases, detection rate was zero percent for the attacks that were not in the training dataset. Discriminator 2, by contrast, achieved a detection rate of 99+% for DoS, fuzzing, and RPM spoofing attacks, and 96.5% for the gear spoofing attack. Looking at accuracy, DoS was 0.979, fuzzing was 0.980, RPM spoofing was 0.980, and gear spoofing was 0.962 (Seo et al., 2018).

To Yang et al. Seo, Song, and Kim's hybrid GAN was the state-of-the-art; as such, Yang et al. sought to build upon and upgrade it. The previous GAN model, an amalgamation of GAN and DNN, performed poorly in terms of detection time. Yang et al. reduced the complexity—and thus, the detection time—of the IDS by reducing the number of convolution and deconvolution layers. In Yang et al.'s single GAN model, the discriminator consists of two convolutional layers and two linear layers, while the generator consists of 2 convolutional layers and one linear layer. The generator's convolutional layers are transposed and function as deconvolutional layers. Ultimately, they were able to achieve an accuracy of 0.998 and a detection time of between 0.09 and 0.15 ms, inclusive (Yang et al., 2021).

Xie et al. discuss a novel GAN-based IDS that leverages the CAN communication matrix. The CAN communication matrix is specified by the Original Equipment Manufacturer (OEM) and encompasses information such as a CAN message's arbitration ID, classification, period, and so on. In much of the literature, windows of consecutive CAN messages—with varied arbitration IDs—are juxtaposed for the purposes of intrusion detection. Yang et al. selected a window size of 64; that is, 64 consecutive CAN messages and a mix of arbitration IDs. To improve precision, Xie et al. opted to sort CAN messages according to arbitration ID, such that each window contains messages with the same identifier. As with Yang et al. the authors consider the timestamp, arbitration ID, and DLC of the CAN frame, but the data field is discarded. The absence of the data field inhibits the ability of the IDS to detect data tampering attacks; that is, attacks in which the data field is manipulated, but the pattern of arbitration IDs remains normal. To detect data tampering attacks, Xie et al. include the OEM's CAN communication matrix in the discriminator. The authors conducted the following types of attacks to evaluate the GAN-based IDS with the CAN communication matrix: (1) DoS, (2) Injection, (3) Spoofing, and (4) Data tampering. Upon evaluation, precision varied from 0.997 to 0.998, while the recall and the F1-score were 0.999 and 0.998, respectively, across all attacks (Xie et al., 2021). There are substantial limitations to this approach. For one, the discriminator would need to be updated with a new CAN communication matrix for each vehicle type, which hampers the portability of this technique. Moreover, there is no guarantee that the CAN communication matrix will be consistent—or even available—across all OEMs (see Table 8).

4.5.1. Analysis

Yang et al. observed that Seo, Song, and Kim's hybrid GAN (Seo et al., 2018) depends heavily on DNN for its accuracy. In addition, Yang et al. noted that the hybrid scheme is slower and more complex, requiring more training time as well as more detection time (Yang et al., 2021). By contrast, the single GAN model is significantly less complicated, easier to tune, and more efficient in terms of training and detection. Moreover, because the model is more efficient, it is easier to adapt to the unique in-vehicle networks of different vehicle manufacturers and models. We have covered several GAN-based IDSs in this section, none of which acknowledge the CAN frame's data field. The first IDS considers only the arbitration ID; the remaining IDSs evaluate the timestamp and DLC in addition to the arbitration ID. As such, these GAN-based IDSs are vulnerable to data tampering attacks. That said, the generator-discriminator dynamic is well suited to training an IDS to detect unknown attacks. It is unrealistic to expect the designers of a given IDS to account for every possible attack or anomaly. In generative adversarial networks, the adversarial training data is offloaded from the researchers to the generator. A well-designed generator is more than capable of crafting examples that researchers would not think to craft, and a well-designed generator can train a discriminator to detect many types of unknown attacks.

4.6. Automotive Ethernet & deep learning

The Ethernet protocol is gaining traction in the automotive industry, as it facilitates higher data rates (Agarwal & Shilpa, 2021). The proliferation of interconnected ECUs, which engage in time-critical

transmissions, strains the limits of the de facto standard, the CAN bus. The Ethernet protocol, applied to automobiles, could prove indispensable to the future-proofing of in-vehicle networks. Furthermore, because the Ethernet protocol sees abundant use outside the automotive industry, Ethernet IVNs can benefit from pre-existing optimizations and advances (Alkhatib et al., 2021). Perhaps most importantly, automotive Ethernet offers security features that are unavailable in traditional Controller Area Networks.

Scalable service-Oriented Middle-warE over IP (SOME/IP) is an application layer protocol designed for the developing domain of automotive Ethernet. Unfortunately, SOME/IP specifies no security features—no authentication, no encryption. To address the security gap, Alkhatib, Ghauch, and Danger designed a sequential deep learning IDS architected as a recurrent neural network (RNN). The authors opted for a sequential scheme due to the fact that traffic in SOME/IP networks demonstrates a significant temporal correlation. Alkhatib, Ghauch, and Danger detailed four types of attacks in SOME/IP networks: (1) Request without Response, (2) Response without Request, (3) Error on Error, and (4) Error on Event.

1. **Request without Response:** We expect a request to be answered with either a response or an error. An unanswered request is indicative of a man-in-the-middle attack.
2. **Response without Request:** We expect exactly one response to a request. A response without a request, or multiple responses to a request suggests an injection attack.
3. **Error on Error:** Specification dictates that an error message should not be answered with an additional error message. As such, an error message answered with an error message is indicative of network intrusion.
4. **Error on Event:** Specification dictates that a notification should not be answered with an error message. Again, we would suspect a network intrusion.

When evaluated, the sequential RNN-based IDS performed well, with a low F1-score of 0.79 and a high score of 0.99. The IDS achieved higher F1-scores for Error on Event and Missing Request anomalies than for Error on Error and Missing Response anomalies (Alkhatib et al., 2021).

Audio Video Transport Protocol (AVTP), like SOME/IP, is an application layer protocol for the automotive Ethernet. It specifies how audio, visual, and control data should be handled in a Time-Sensitive Networking (TSN) environment. Alkhatib et al. evaluated two deep learning IDSs, as well as several traditional machine learning implementations, in the context of automotive Ethernet. The deep learning architectures were convolutional neural network and long short-term memory models augmented with autoencoders. To group and compare CAN frames, they leveraged pre-determined sliding windows, experimenting with window sizes of 8, 16, 24, 32, and 40 CAN frames. In general, the deep learning IDSs outperformed their machine learning counterparts; both the CNN autoencoder and the LSTM autoencoder achieved superior recall and F1-scores no matter the window size. However, in terms of precision, the Local Outlier Factor (LOF) machine learning IDS did overtake the LSTM autoencoder at larger window sizes (Alkhatib et al., 2022).

4.6.1. Analysis

Ethernet is a well-established protocol; automotive Ethernet is not. Both SOME/IP and AVTP suffer security shortcomings, but a secure automotive Ethernet protocol is possible. Ethernet supports a number of security features, even if they have not been adopted by an automotive Ethernet protocol as of yet. In the meantime, deep learning IDSs demonstrate a lot of potential: both of the schemes described above have achieved excellent F1-scores. For the IDSs applied to the AVTP protocol, we can see that deep learning generally outperforms traditional machine learning. That said, automotive Ethernet is a relatively new research area, and the IDSs discussed in this section have been evaluated against a limited number of attacks. To assess the potential—and the limitations—of intrusion detection for the automotive Ethernet, further exploration and evaluation is needed.

4.7. Advantages & disadvantages

When we evaluate the advantages and disadvantages of particular architectures, we observe that lines are blurred by hybrid architectures and by IDSs which incorporate specific tactics (ID sequencing, graphing) into a deep learning scheme. A graph-based LSTM which analyzes both the arbitration ID and the data field differs greatly from an CNN which encodes arbitration IDs as two-dimensional images and excludes the data field entirely. However, we cannot generalize that the LSTM architecture is superior to the CNN architecture—or vice versa. Design decisions beyond the choice of deep learning architecture can have far greater impact than LSTM vs. CNN. An IDS trained by unsupervised learning might prove more capable of detecting novel attacks—because such an IDS is forced to focus on normal vs. abnormal, rather than the particular patterns of a particular subset of attacks. An IDS which analyzes the data field in addition to the arbitration ID will—theoretically—be able to detect data tampering attacks, whereas an IDS which drops the data field cannot detect that type of attack.

Nonetheless, a few trends emerge as we compare various deep learning architectures as applied to the problem of automotive intrusion detection. The deep neural networks described in this section were purely feed-forward, meaning that signals move in one direction—forward—and can never move backward or form cycles. As such, the DNN architecture can be simpler to implement successfully, but it cannot learn from prior input or prior iterations of training. Compared to fluctuation and variability of the internet, CAN traffic is highly predictable. Looking back can yield valuable information when evaluating future traffic. DNNs are unable to capitalize on that knowledge.

CNNs have been popularized by image processing applications, but CAN frames are not images. To leverage CNN's image processing optimizations, researchers have to convert CAN frames into images. This extra step constitutes extra overhead, and, depending on how this extra step is implemented, it can also result in loss of valuable information (e.g., frequency, sequencing).

LSTM provides a longer lived "short-term memory" that can capture temporal dependencies. Generally, it appears to be well suited to automotive intrusion detection. However, to capture the temporal dependencies of specific arbitration IDs, then each arbitration ID must be assigned its own LSTM. This multi-LSTM architecture captures much more information, but it is also much more complicated, and it is certainly not a lightweight scheme.

DTL lightens the training burden on an IDS—both in terms of the data requirements and the time requirements. The proprietary CAN bus does not lend itself to ample amounts of data; indeed, there are very few publicly available CAN bus datasets. The few available datasets are specific to particular manufacturers and even particular models, so DTL is an obvious choice to combat the shortage of training data. Moreover, deep learning models are notoriously resource-intensive to train. DTL significantly reduces this training burden by leveraging pre-trained models.

Attention- and transformer-based architectures automatically focus on informative data; however, the benefits are limited when we look at a CAN frame. Generally, researchers have selected the CAN frame's arbitration ID, data field (sometimes), and data length code (occasionally). There is limited data for an attention- or transformer-based scheme to evaluate as informative or uninformative—especially compared to deep learning problems which involve dozens or hundreds of features.

GAN does not depend on the researcher to construct realistic attacks. Instead, the generator is responsible for the adversarial input

supplied to the discriminator. Offloading responsibility for the adversarial component of the training data is perhaps the main advantage of GAN. Furthermore, the discriminator learns to "discriminate" between real traffic and generated traffic, which helps it prepare to detect many types of anomalies, even unknown attacks.

Hybrid architectures, which combine one or more of the above deep learning methodologies, have also shown promise. They combine the advantages of several architectures, but they are necessarily more complex. If implemented properly, they can be remarkably successful. If implemented improperly, they might just be harder to debug.

4.8. Key points

I. Deep neural networks (DNNs), convolutional neural networks (CNNs), long short-term memory (LSTM), deep transfer learning (DTL), attention and transformer, and generative adversarial networks (GANs) are all deep learning architectures that have been successfully applied to the problem of automotive intrusion detection.
II. Choice of architecture is important, but design decisions such as supervised or unsupervised learning, feature selection, number of layers, etc. can have greater impact on the outcome.
III. Hybrid architectures (e.g., LSTM autoencoders) combine the advantages of several schemes, but if each component is not properly configured and optimized, then they will be outperformed by simpler architectures.

5. Open challenges in deep learning automotive intrusion detection

Fig. 6 enumerates open challenges in deep learning as it pertains to automotive intrusion detection. Several of these challenges, such as *Mitigation* and *Intelligent Attacks,* have carried over from traditional intrusion detection systems.

5.1. Complexity & resources

Deep learning automotive intrusion detection strategies tend to be higher in complexity and resource consumption than traditional approaches, even traditional machine learning approaches. In-vehicle networks possess extremely limited computing and memory resources; as such, deployment of resource-intensive deep learning strategies is an open challenge (Hanselmann et al., 2020; Wu et al., 2019). Future work will have to answer the question: How can we deploy a deep learning model in an environments with constrained resources?

5.1.1. Solutions

Rather than deploy a deep learning IDS on one or more existing ECUs, perhaps a deep learning IDS could be appended as an additional node. Due to the broadcast nature of many IVNs, particularly the CAN bus, the IDS would be privy to all communication across the bus. The new node containing the IDS would feature all of the hardware and software needed to support deep learning. Thus, the resource burden would not fall on the IVN itself.

5.2. Datasets & training

Deep learning models demand ample quantities of training data. Much of the automotive realm is proprietary; as such, obtaining the necessary data can prove difficult. If the approach is supervised, then the training data must also be labeled, and it must cover as many situations as possible.

In general, a DL-based automotive intrusion detection system must be trained on the vehicle it will be monitoring. If a DL-based IDS is moved from one model of vehicle to a different model, then it will need to be trained on the new model. Even two vehicles produced by the same manufacturer will differ in terms of the data flowing across the CAN bus; thus, there is an expensive training (and retraining) requirement applied to most—if not all—DL-based approaches.

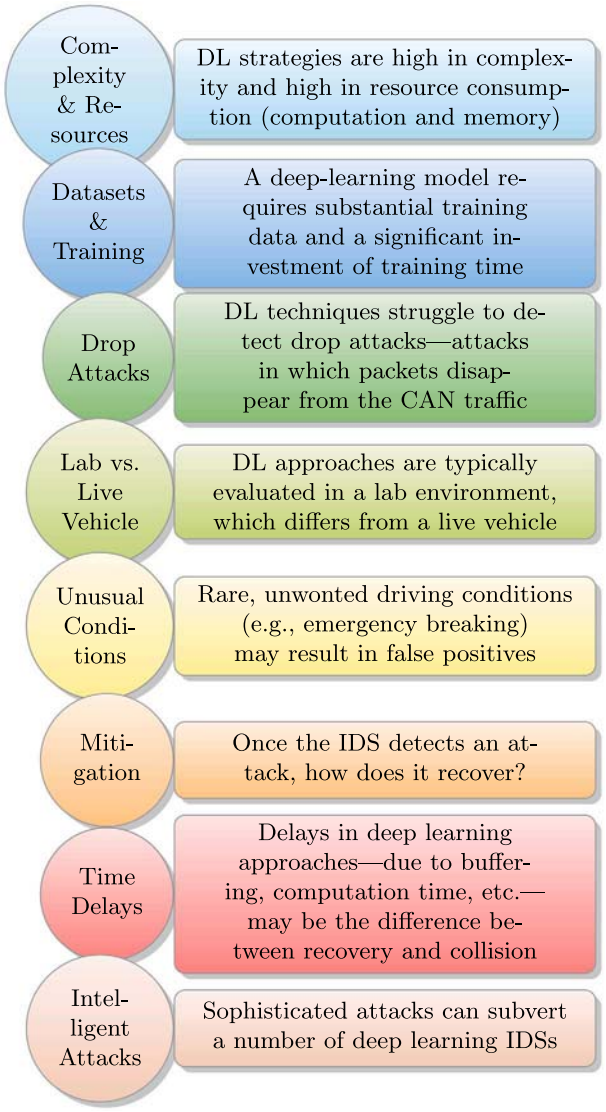

Fig. 6. Open Challenges in Deep Learning Intrusion Detection Systems in In-Vehicle Networks.

5.2.1. Solutions

If automotive IDSs were sold to consumers, then they could be designed to begin a "training" phase when initially connected to the vehicle, followed by a "monitoring" phase once training is complete. If the consumer can plug the IDS into the vehicle and forget about it, then it would not matter if the training phase took ten hours of driving to complete, since automotive cyberattacks are not yet commonplace. For researchers, though, especially researchers with limited access to an actual vehicle, more publicly-available IVN data is needed.

5.3. Drop attacks

Both deep learning and traditional IDSs struggle with anomalies referred to as "drop attacks". In this type of attack, an adversary disables an ECU—perhaps in order to eliminate confliction—and there is a corresponding gap in the CAN traffic. An IDS is expected to identify this "dropped" traffic and raise an alert, but in many cases, it is easier to recognize traffic that does not belong than traffic that is simply missing. The IDS cannot afford to wait for the adversary to masquerade as the disabled ECU. Detection might be easier with the adversary's spoofed messages, but it might also be too late. Miller and Valasek describe

an attack in which they disable the power steering control module, so that it cannot intervene when they use the parking assist module to turn the wheel at speed (Miller & Valasek, 2016a, 2016b). In such an attack, disabling the power steering module is a mandatory first step. Importantly, it is also an opportunity for an IDS to detect a drop attack. If an IDS recognizes that the power steering control module's CAN messages are absent from the bus traffic, then it can alert the driver or begin mitigation efforts. Once the parking assist module is instructed to turn the wheel sharply at speed, it is too late for warnings or countermeasures; the vehicle has veered off the road. As such, it is imperative to detect drop attacks as quickly as possible.

5.3.1. Solutions

A singular intrusion detection system capable of accurately and efficiently detecting every type of attack is perhaps unattainable. Instead, a hybrid IDS could combine the strengths of multiple approaches, improving detection capability while keeping resource consumption at a reasonable level. Such an IDS might be comprised of a deep learning component and one or more lightweight components. The deep learning component would focus on (1) intelligent attacks and (2) unknown attacks. Meanwhile, the lightweight components would handle conspicuous attacks: (1) drop, (2) fuzzing, (3) DoS attacks, etc. For example, a lightweight IDS could maintain a dictionary of every arbitration ID and its frequency. If a given arbitration ID should appear every 0.25 s, and 5.0 s pass with no communication, then it is clear that something is wrong. A lightweight IDS could monitor the bus and check for this condition. Similarly, if a fuzzing attack generates arbitration IDs that are not valid and have never appeared on the bus before, then a lightweight IDS can raise an alert.

5.4. Lab vs. live vehicle

Experimental evaluations have yielded promising precision, recall, and F1-scores for a number of DL-based IDSs. However, such experimental evaluations are conducted in lab settings, using previously-captured datasets that are replayed on a computer. Personal computers and automobiles are vastly different machines. The CAN bus has different hardware, different computing and memory resources, and a different operating environment. No lab environment can perfectly simulate real-time application in a moving vehicle. As such, we are left with the following question: How well does a deep learning approach translate from the lab to the open road?

5.4.1. Solutions

A hardware-based test harness, composed of actual ECUs connected by twisted-pair wiring, could more closely simulate a live vehicle. The hardware would be genuine automotive hardware with constrained computing and memory resources. However, to evaluate the capabilities of an intrusion detection system in a live vehicle, the best and most accurate option is to conduct a live vehicle test.

5.5. Unusual conditions

To reduce false positives, deep learning IDSs must be exposed to as many driving conditions as possible. Extensive data must be collected while parked, while idling, while driving, etc. Unfortunately, no matter how complete the dataset, new and unusual conditions will occur during driving. It is the responsibility of the intrusion detection system to differentiate two types of abnormal traffic, (1) adversarial and (2) uncommon. Emergency braking, genuine mechanical failure of one or more systems, evasive maneuvers to avoid a collision—all of these are rare but non-malicious events that can result in abnormal CAN traffic. A false positive in these situations would place further stress an already-stressed driver, especially if the intrusion detection system activates some form of mitigation mechanism.

5.5.1. Solutions

We can introduce thresholds to reduce false positives—though the trade-off is often false negatives. If the IDS is equipped with mitigation mechanisms, then said mechanisms can be tiered according to certainty. For example, at 50% certainty, the IDS might alert the driver with a chime, while at 95% certainty, the IDS might short-circuit the bus, forcing the vehicle into a failsafe state. Lastly, building context-awareness into the IDS could help it to understand the "big picture" and differentiate an emergency situation from an attack.

5.6. Mitigation

A sophisticated intrusion detection system is an important first step in securing automobiles, but an intrusion detection system, in and of itself, has limited efficacy when time is of the essence. An attack might happen in a matter of milliseconds, leaving the driver with no time to respond to an alert. As such, we see mitigation and recovery efforts as open challenges on this topic. An intrusion prevention system (IPS) would be a logical next step from an IDS, though it comes with its own challenges.

5.6.1. Solutions

The CAN bus provides a number of error-handling functionalities, which can block an ECU from participating in the bus traffic if it malfunctions. If an intrusion prevention system can leverage the bus's error-handling capabilities, then it can disable communication from a suspected attack node. A more extreme measure would be to short-circuit the CAN bus by connecting `CAN HIGH` to `CAN LOW`. If the two are connected, then the voltage will equalize, and without differential voltage, there is no differential signaling. The vehicle's ECUs will enter failsafe mode, and attack traffic will no longer be possible.

5.7. Time delays

Assuming that DL-based intrusion detection system recognizes an intrusion and activates a mitigation and recovery system, we still have the open issue of timing. If there is even a 1-s delay (Desta et al., 2020b), then the attack might be over and the vehicle might be in the ditch—or entangled in the vehicle with which it collided. A number of authors acknowledged a time delay in the detection capability of the proposed approach. For DL-based IDSs that leverage sliding windows, a shorter window decreases detection time but also reduces accuracy. Finding a balance between detection time and accuracy is an open issue in DL-based automotive intrusion detection.

5.7.1. Solutions

The concept of certainty, which we explored earlier in this section, could help address this issue. With a shorter detection window, certainty is reduced, so milder forms of mitigation would be appropriate. With longer detection windows, certainty can be much higher, and an intrusion prevention system could pursue more aggressive mitigation strategies. An IDS that monitors two or more detection windows simultaneously would be better equipped to select a suitable mitigation action. Moreover, some attacks are more blatant than others. If the bus is flooded with 100x more CAN frames than usual, all with the arbitration ID `000` (the highest-priority identifier) then the IDS should recognize the attack—with high certainty—in a short amount of time.

5.8. Intelligent attacks

One underexplored issue in automotive intrusion detection, both in deep learning and traditional paradigms, is that of intelligent attacks. At this time, the average vehicle does not have added security, so adversaries would not need to conduct sophisticated attacks. However, when security is added, adversaries will develop intelligent attacks to thwart security. In the realm of intrusion detection, an intelligent

attack would be one that exploits gaps in the detection mechanism. For example, a detection scheme that monitors the arbitration ID but not the data field would miss an attack that manipulates the data field alone. Frequency-, interval-, and timing-based deep learning methods cannot detect attacks in which the adversary calculates the timing of the attack such that it coincides with expected, normal intervals. Statistical IDSs can be deceived by traffic that is customized to match the statistical patterns of normal traffic. Adversaries capable of sniffing the normal CAN traffic should have no trouble computing the statistics of normal traffics and then matching them. And since CAN traffic is unencrypted, we would expect adversaries who have access to the network to be capable of sniffing.

5.8.1. Solutions

An intelligent adversary can probably craft an attack that evades timing-related detection mechanisms while maintaining its efficacy. An intelligent adversary might also prove capable of devising an attack that matches the entropy of normal traffic while still fulfilling its malicious objectives. However, an effective attack that evades both timing- and entropy-related detection mechanisms would be much more challenging. If we augment our hybrid timing- and entropy-based IDS with several more statistical detection mechanisms, then our intelligent adversary would have to tailor his or her attack to elude every timing-, entropy-, and statistic-based detection mechanism. With each additional detection mechanism, such a stealthy attack becomes less and less feasible. As such, a hybrid IDS would be better equipped to detect intelligent attacks than an IDS that implements just one technique.

5.9. Key points

I There are a number of open issues in automotive intrusion detection, many of which relate to the detection capability of the IDS—drop attacks, intelligent attacks, lab vs. live vehicle, etc.

II In many cases, a hybrid IDS is proposed as a solution: hybrid IDSs combine the strengths of multiple approaches to overcome the weaknesses of individual techniques.

III "Certainty" and "thresholds" are two concepts that could guide an intrusion detection system to an appropriate mitigation technique.

6. Conclusion

We have learned about the controller area network (CAN) bus, including the CAN bus's vulnerabilities and how to exploit them. We have explored the intersection of automotive intrusion detection and deep learning to fill in the gaps in CAN bus security. In particular, we studied the following deep learning architectures:

- Deep Neural Networks (DNNs)
- Convolutional Neural Networks (CNNs)
- Long Short-term Memory (LSTM)
- Deep Transfer Learning (DTL)
- Attention & Transformer
- Generative Adversarial Networks (GANs)

We saw that feed-forward *DNNs* are a comparatively simple option, but that they lack the ability to learn from prior input and prior iterations of training. We found *CNNs* to be highly specialized for image processing applications. We can leverage highly specialized CNNs for non-image CAN frames—but we have to convert the CAN frames to images. *LSTM* gave us temporal dependencies, but if we wanted the temporal dependencies of specific arbitration IDs, then we needed to assign a separate LSTM to each arbitration ID. *DTL* promises us reduced training time and reduced data requirements if we start from a pre-trained model. *Attention and transformer* architectures help us emphasize informative data and de-emphasize uninformative data. *GANs* allow us to offload responsibility for the adversarial component of the training data. If we supply normal data, then the generator will train the discriminator to recognize abnormal data. Lastly, we evaluated *hybrid* architectures. We concluded that hybrid schemes can combine the advantages of multiple schemes—but only if properly tuned and optimized.

With each passing year, in-vehicle networks become more and more complex. This trend is expected to continue: more automotive systems will be converted from mechanical to electric, more features will be developed, and more ECUs will be added to support such modifications. The connectivity of the modern automobile also increases year by year. With this heightened connectivity comes a larger attack surface. As the opportunities to attack automotive networks burgeon, we anticipate an uptick in adversarial interest. When the severity of IVN attacks shifts from manipulation of the infotainment system—which is merely a nuisance—to spinning the steering wheel 180 degrees at speed (Miller & Valasek, 2016b).

If and when attacks on automotive networks proliferate, we would expect to see concurrent advances in the sophistication of the attacks. As in internet and computer networks, we can predict an arms race between IVN defenders and IVN attackers. As such, the more simplistic IDSs—IDSs which inspect only timing information or analyze only the ID fields of CAN frames—will not scale to newer and more intelligent attacks. Deep learning intrusion detection systems are capable of an amazing depth and breadth of analysis, and they can learn and develop alongside novel attacks. For the purposes of future proofing, deep learning is a promising direction for the automotive intrusion detection system.

CRediT authorship contribution statement

Brooke Lampe: Conceptualization, Methodology, Investigation, Writing – original draft. **Weizhi Meng:** Writing – review & editing, Resources, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

Abbott-McCune, S., & Shay, L. A. (2016). Intrusion prevention system of automotive network CAN bus. In *2016 IEEE international carnahan conference on security technology*.

Agarwal, Y., & Shilpa, D. R. (2021). Automotive ethernet physical optimization and IEEE 1588 implementation. In V. Nath, & J. K. Mandal (Eds.), *Proceeding of fifth international conference on microelectronics, computing and communication systems. Vol. 748* (pp. 489–500). Springer Singapore.

Agrawal, K., Alladi, T., Agrawal, A., Chamola, V., & Benslimane, A. (2022). NovelADS: A novel anomaly detection system for intra-vehicular networks. *IEEE Transactions on Intelligent Transportation Systems*, 1–11.

Ahmed, I., Ahmad, A., & Jeon, G. (2021). Deep learning-based intrusion detection system for internet of vehicles. *IEEE Consumer Electronics Magazine*.

Alkhatib, N., Ghauch, H., & Danger, J.-L. (2021). SOME/IP intrusion detection using deep learning-based sequential models in automotive ethernet networks. In *2021 IEEE 12th annual information technology, electronics and mobile communication conference*.

Alkhatib, N., Mushtaq, M., Ghauch, H., & Danger, J.-L. (2022). Unsupervised network intrusion detection system for AVTP in automotive ethernet networks. In *2022 IEEE intelligent vehicles symposium*.

Arnold, L., Rebecchi, S., Chevallier, S., & Paugam-Moisy, H. (2011). An introduction to deep learning. In *European symposium on artificial neural networks*.

Bozdal, M., Samie, M., Aslam, S., & Jennions, I. (2020). Evaluation of CAN bus security challenges. *Sensors*, *20*.

Desta, A. K., Ohira, S., Arai, I., & Fujikawa, K. (2020a). ID sequence analysis for intrusion detection in the CAN bus using long short term memory networks. In *2020 IEEE international conference on pervasive computing and communications workshops* (pp. 1–6).

Desta, A. K., Ohira, S., Arai, I., & Fujikawa, K. (2020b). MLIDS: Handling raw high-dimensional CAN bus data using long short-term memory networks for intrusion detection in in-vehicle networks. In *2020 30th International telecommunication networks and applications conference*.

Developers, G. (2022). Classification: ROC curve and AUC. Google Developers.

DiPietro, R., & Hager, G. D. (2020). Chapter 21 - deep learning: RNNs and LSTM. In S. K. Zhou, D. Rueckert, & G. Fichtinger (Eds.), *The Elsevier and MICCAI society book series*, *Handbook of medical image computing and computer assisted intervention* (pp. 503–519). Academic Press.

Doan, T. P., & Ganesan, S. (2017). CAN crypto FPGA chip to secure data transmitted through CAN FD bus using AES-128 and SHA-1 algorithms with A symmetric key. In *SAE international*.

Dönmez, T. C. M. (2021). Anomaly detection in vehicular CAN bus using message identifier sequences. *IEEE Access*, *9*, 136243–136252.

Du, X., Cai, Y., Wang, S., & Zhang, L. (2016). Overview of deep learning. In *2016 31st Youth academic annual conference of chinese association of automation*.

Dupont, G., den Hartog, J., Etalle, S., & Lekidis, A. (2019). A survey of network intrusion detection systems for controller area network. In *2019 IEEE international conference on vehicular electronics and safety*.

Friedman, A., & Singer, P. W. (2014). What do we mean by security anyway? In *Brookings*.

Gmiden, M., Gmiden, M. H., & Trabelsi, H. (2016). An intrusion detection method for securing in-vehicle CAN bus. In *2016 17th International conference on sciences and techniques of automatic control and computer engineering* (pp. 176–180).

Groza, B., & Murvay, S. (2013). Efficient protocols for secure broadcast in controller area networks. *IEEE Transactions on Industrial Informatics*, *9*, 2034–2042.

Groza, B., Murvay, S., van Herrewege, A., & Verbauwhede, I. (2012). LiBrA-CAN: A lightweight broadcast authentication protocol for controller area networks. In *International conference on cryptology and network security*.

Hanselmann, M., Strauss, T., Dormann, K., & Ulmer, H. (2020). CANet: An unsupervised intrusion detection system for high dimensional CAN bus data. *IEEE Access*, *8*, 58194–58205.

Hossain, M. D., Inoue, H., Ochiai, H., Fall, D., & Kadobayashi, Y. (2020a). An effective in-vehicle CAN bus intrusion detection system using CNN deep learning approach. In *GLOBECOM 2020 - 2020 IEEE global communications conference* (pp. 1–6).

Hossain, M. D., Inoue, H., Ochiai, H., Fall, D., & Kadobayashi, Y. (2020b). Long short-term memory-based intrusion detection system for in-vehicle controller area network bus. In *2020 IEEE 44th annual computers, software, and applications conference*.

Hossain, M. D., Inoue, H., Ochiai, H., Fall, D., & Kadobayashi, Y. (2020c). LSTM-based intrusion detection system for in-vehicle can bus communications. *IEEE Access*, *8*, 185489–185502.

Islam, R., & Refat, R. U. D. (2020). Improving CAN bus security by assigning dynamic arbitration IDs. *Journal of Transportation Security*, *13*, 19–31.

Islam, R., Refat, R. U. D., Yerram, S. M., & Malik, H. (2020). Graph-based intrusion detection system for controller area networks. *IEEE Transactions on Intelligent Transportation Systems*, 1–10.

Jedh, M., Othmane, L. B., Ahmed, N., & Bhargava, B. (2021). Detection of message injection attacks onto the CAN bus using similarities of successive messages-sequence graphs. *IEEE Transactions on Information Forensics and Security*, *16*, 4133–4146.

Kim, K., Kim, J. S., Jeong, S., Park, J.-H., & Kim, H. K. (2021). Cybersecurity for autonomous vehicles: Review of attacks and defense. *Computers & Security*, *103*, Article 102150.

Koscher, K., Czeskis, A., Roesner, F., Shwetak Patel, T. K., Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., & Savage, S. (2010). Experimental security analysis of a modern automobile. In *IEEE symposium on security and privacy*.

Lampe, B., & Meng, W. (2022). IDS for CAN: A practical intrusion detection system for CAN bus security. In *2022 IEEE global communications conference*.

Larson, U. E., Nilsson, D. K., & Jonsson, E. (2008). An approach to specification-based attack detection for in-vehicle networks. In *2008 IEEE intelligent vehicles symposium* (pp. 220–225).

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*, 2278–2324.

Lee, H., Jeong, S. H., & Kim, H. K. (2017). OTIDS: A novel intrusion detection system for in-vehicle network by using remote frame. In *2017 15th annual conference on privacy, security and trust* (pp. 57–5709).

Lee, S.-W., sidqi, H. M., Mohammadi, M., Rashidi, S., Rahmani, A. M., Masdari, M., & Hosseinzadeh, M. (2021). Towards secure intrusion detection systems using deep learning techniques: Comprehensive analysis and review. *Journal of Network and Computer Applications*, *187*, Article 103111.

Lemke, K., Paar, C., & Wolf, M. (2006). *Embedded security in cars*. Springer-Verlag.

Lin, T., Wang, Y., Liu, X., & Qiu, X. (2022). A survey of transformers. *AI Open*, *3*, 111–132.

Lokman, S.-F., Othman, A. T., & Abu-Bakar, M.-H. (2019). Intrusion detection system for automotive Controller Area Network (CAN) bus system: a review. *EURASIP Journal on Wireless Communications and Networking*, *2019*.

Longari, S., Valcarcel, D. H. N., Zago, M., Carminati, M., & Zanero, S. (2020). CANnolo: An anomaly detection system based on LSTM autoencoders for controller area network. *IEEE Transactions on Network and Service Management*, *18*, 1913–1924.

Marchetti, M., & Stabili, D. (2017). Anomaly detection of CAN bus messages through analysis of ID sequences. In *2017 IEEE intelligent vehicles symposium* (pp. 1577–1583).

Markovitz, M., & Wool, A. (2017). Field classification, modeling and anomaly detection in unknown CAN bus networks. *Vehicular Communications*, *9*, 43–52.

Mehedi, S. T., Anwar, A., Rahman, Z., & Ahmed, K. (2021). Deep transfer learning based intrusion detection system for electric vehicular networks. *Sensors (Basel)*.

Miller, C., & Valasek, C. (2014). Adventures in automotive networks and control units. IOActive.

Miller, C., & Valasek, C. (2015a). Remote exploitation of an unaltered passenger vehicle. IOActive.

Miller, C., & Valasek, C. (2015b). Remote exploitation of an unaltered passenger vehicle. BlackHat.

Miller, C., & Valasek, C. (2016a). Advanced CAN injection techniques for vehicle networks. BlackHat.

Miller, C., & Valasek, C. (2016b). CAN message injection. Illmatics.

Müter, M., & Asaj, N. (2011). Entropy-based anomaly detection for in-vehicle networks. In *2011 IEEE intelligent vehicles symposium* (pp. 1110–1115).

Nam, M., Park, S., & Kim, D. S. (2021). Intrusion detection method using bi-directional GPT for in-vehicle controller area networks. *IEEE Access*, *9*, 124931–124944.

Narayanan, S. N., Khanna, K., Panigrahi, B. K., & Joshi, A. (2019). Chapter 11 - security in smart cyber-physical systems: A case study on smart grids and smart cars. In D. B. Rawat, & K. Z. Ghafoor (Eds.), *Smart cities cybersecurity and privacy* (pp. 147–163). Elsevier.

NasrEldin, A., Bahaa-Eldin, A. M., & Sobh, M. A. (2021). In-vehicle intrusion detection based on deep learning attention technique. In *2021 16th International conference on computer engineering and systems*.

Niu, Z., Zhong, G., & Yu, H. (2021). A review on the attention mechanism of deep learning. *Neurocomputing*, *452*, 48–62.

Ohira, S., Desta, A. K., Arai, I., Inoue, H., & Fujikawa, K. (2020). Normal and malicious sliding windows similarity analysis method for fast and accurate IDS against DoS attacks on in-vehicle networks. *IEEE Access*, *8*, 42422–42435.

O'Shea, K. T., & Nash, R. (2015). An introduction to convolutional neural networks. ArXiv E-Prints.

Palanca, A., Evenchick, E., Maggi, F., & Zanero, S. (2017). A stealth, selective, link-layer denial-of-service attack against automotive networks. In *Detection of intrusions and malware, and vulnerability assessment* (pp. 185–206). Springer International Publishing.

Parashar, A. (2020). Vgg 16 architecture, implementation and practical use. Medium.

Rashid, Z. (2021). Deep learning approaches for intrusion detection. *Asian Journal of Research in Computer Science*.

Sami, M., Ibarra, M., Esparza, A. C., Al-Jufout, S., Aliasgari, M., & Mozumdar, M. (2020). Rapid, multi-vehicle and feed-forward neural network based intrusion detection system for controller area network bus. In *2020 IEEE green energy and smart systems conference*.

Seo, E., Song, H. M., & Kim, H. K. (2018). GIDS: GAN based intrusion detection system for in-vehicle network. In *2018 16th Annual conference on privacy, security and trust*.

Shamah, D. (2015). Israeli start-up leads fight against remote car-hacking. The Times of Israel.

Siddiqui, A. S., Gui, Y., Plusquellic, J., & Saqib, F. (2017). Secure communication over CANBus. In *2017 IEEE 60th international midwest symposium on circuits and systems*.

Song, H. M., Kim, H. R., & Kim, H. K. (2016). Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network. In *2016 International conference on information networking* (pp. 63–68).

Song, H. M., Woo, J., & Kim, H. K. (2020). In-vehicle network intrusion detection using deep convolutional neural network. *Vehicular Communications*, *21*.

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., & Liu, C. (2018). A survey on deep transfer learning. In *International conference on artificial neural networks* (pp. 270–279).

Taylor, A., Japkowicz, N., & Leblanc, S. (2016). Frequency-based anomaly detection for the automotive CAN bus. In *2015 World congress on industrial control systems security* (pp. 45–49).

Vargas, R., Mosavi, A., & Ruiz, R. (2017). Deep learning: A review. In *Advances in intelligent systems and computing*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Łukasz Kaiser, & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems. Vol. 30*.

Verendel, V., Nilsson, D. K., Larson, U. E., & Jonsson, E. (2008). An approach to using honeypots in in-vehicle networks. In *2008 IEEE 68th vehicular technology conference*.

Wang, Q., Lu, Z., & Qu, G. (2018). An entropy analysis based intrusion detection system for controller area network in vehicles. In *2018 31st IEEE international system-on-chip conference* (pp. 90–95).

Wang, Q., & Sawhney, S. (2014). VeCure: A practical security framework to protect the CAN bus of vehicles. In *2014 International conference on the internet of things*.

Woo, S., Moon, D., Youn, T.-Y., Lee, Y., & Kim, Y. (2019). CAN ID shuffling technique (CIST): Moving target defense strategy for protecting in-vehicle CAN. *IEEE Access*, *7*, 15521–15536.

Wu, W., Huang, Y., Kurachi, R., Zeng, G., Xie, G., Li, R., & Li, K. (2018). Sliding window optimized information entropy analysis method for intrusion detection on in-vehicle networks. *IEEE Access*, *6*, 45233–45245.

Wu, W., Li, R., Xie, G., An, J., Bai, Y., Zhou, J., & Li, K. (2019). A survey of intrusion detection for in-vehicle networks. *IEEE Transactions on Intelligent Transportation Systems*, *21*, 919–933.

Xiao, L., Lu, X., Xu, T., Zhuang, W., & Dai, H. (2021). Reinforcement learning-based physical-layer authentication for controller area networks. *IEEE Transactions on Information Forensics and Security*, *16*, 2535–2547.

Xie, G., Yang, L. T., Yang, Y., Luo, H., Li, R., & Alazab, M. (2021). Threat analysis for automotive CAN networks: A GAN model-based intrusion detection technique. *IEEE Transactions on Intelligent Transportation Systems*, *22*, 4467–4477.

Yang, Y., Xie, G., Wang, J., Zhou, J., Xia, Z., & Li, R. (2021). Intrusion detection for in-vehicle network by using single GAN in connected vehicles. *Journal of Circuits, Systems, and Computers*, *30*.

Young, C., Zambreno, J., Olufowobi, H., & Bloom, G. (2019). Survey of Automotive Controller Area network intrusion detection systems. *IEEE Design & Test*.

Zekry, A., Sayed, A., Moussa, M., & Elhabiby, M. (2021). Anomaly detection using IoT sensor-assisted ConvLSTM models for connected vehicles. In *2021 IEEE 93rd vehicular technology conference*.

Zhang, L., & Ma, D. (2022). A hybrid approach toward efficient and accurate intrusion detection for in-vehicle networks. *IEEE Access*, *10*, 10852–10866.