Received February 18, 2020, accepted March 17, 2020, date of publication March 23, 2020, date of current version April 6, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2982544

CANet: An Unsupervised Intrusion Detection System for High Dimensional CAN Bus Data

MARKUS HANSELMANN[1], THILO STRAUSS[1], KATHARINA DORMANN[2], AND HOLGER ULMER[1]
[1]Machine Learning Group at ETAS GmbH, Robert Bosch GmbH, 70469 Stuttgart, Germany
[2]Powertrain Solutions, Robert Bosch GmbH, 71638 Ludwigsburg, Germany

Corresponding author: Thilo Strauss (thilo.strauss@etas.com)

ABSTRACT We propose a neural network architecture for detecting intrusions on the controller area network (CAN). The latter is the standard communication method between the electronic control units (ECUs) of automobiles. However, CAN lacks security mechanisms and it has recently been shown that it can be attacked remotely. Hence, it is desirable to monitor CAN traffic to detect intrusions. In order to find both, known and unknown intrusion scenarios, we consider a novel unsupervised learning approach which we call CANet. To our knowledge, this is the first deep learning based intrusion detection system (IDS) that naturally handles the data structure of the high dimensional CAN bus, where different message types are sent at different times. Our method is evaluated on real and synthetic CAN data. A comparison with previous machine learning based methods shows that CANet outperforms them by a significant margin. For reproducibility of the method, our synthetic data is publicly available.

INDEX TERMS CAN bus, deep learning, intrusion detection.

I. INTRODUCTION

Automobiles are getting more and more connected by technologies such as Bluetooth, Wifi or smart phone plug-ins. While this simplifies the driver's life, it simultaneously opens new paths for potential remote attacks on the electronic control units (ECUs) of cars. Hijacking an ECU can allow attackers to place messages on the vehicle-internal communication network. Thereby the attacker could e.g. invoke sudden breaking or turning off the engine which can cause traffic accidents [1], [2]. Hence, detecting the attempt of attacks in car networks is in the interest of traffic safety.

In this paper, we focus on the controller area network (CAN) bus as it is the most common vehicle bus standard. Typically, CAN messages are used to transmit signals between ECUs. For example, an ECU can send the information about objects on the road so that the brake assist can react accordingly.

An extensive overview about previous work on CAN intrusion detection systems (IDS) can be found in [3]. Currently, a strong focus lies on rule based and statistical methods to detect known attack scenarios. While many types of intrusions can be detected efficiently by these approaches, the configuration of such an IDS is time-consuming, requires domain expertise, and it is unlikely that unknown attack scenarios can be detected. Moreover, it is challenging to generate rules that capture the complex behavior of multiple signals that are transmitted via the CAN bus.

With the advances in deep learning in the recent years [4]–[6] new tools are becoming available that have the potential of detecting unknown attacks. Prior work on intrusion detection with neural networks on single CAN signals can be found in [7]–[9]. However, to the best of our knowledge, there is no neural network architecture that can handle the CAN bus data structure in the multidimensional signal space. On the CAN bus, at every point in time at most one message is transmitted. As a result, CAN traffic consists of consecutive messages with different IDs. These messages contain different kinds of signals (see Table 1). The data structure makes it difficult to feed the data of the CAN bus directly into any kind of standard neural network.

The contributions of this manuscript are the following: We introduce CANet, a novel neural network architecture tailored to work on the signal space of CAN data. We show that it outperforms existing methods by a significant margin. We describe how this architecture can be trained in an unsupervised manner and evaluated in such a way that a variety of unknown attack types can be detected in the moment they

The associate editor coordinating the review of this manuscript and approving it for publication was Shancang Li.

TABLE 1. Schematic representation of CAN bus data after preprocessing its payload bytes to signals.

CAN Bus Data after Preprocessing													
Time Stamp	ID	Signals of A						Signals of B			Signals of C			Signals of D
1.045	B	-	-	-	-	-	-	54.71	0	7.24	-	-	-	-
3.102	D	-	-	-	-	-	-	-	-	-	-	-	-	31.47
4.978	A	12	44.15	38.02	2	0	1	-	-	-	-	-	-	-
7.014	C	-	-	-	-	-	-	-	-	-	17.79	7	2	-
8.993	B	-	-	-	-	-	-	55.02	1	7.21	-	-	-	-
9.750	A	13	44.01	39.67	1	0	2	-	-	-	-	-	-	-
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

occur while identifying normal data correctly. We show that our method is especially strong in finding certain manipulations of signals that are difficult to detect by classical approaches. Hereby, we exploit the fact that the network is able to learn the physical relationships between the signals under consideration.

We point out that CANet may have other applications beyond intrusion detection like anomaly detection in general, e.g. early detection of technical failures.

This document is organized in the following way: In Section II, the background of the CAN bus and the relevant literature on CAN IDS is briefly covered. In Section III, the proposed network architecture and its training process is described. In Section IV, the method is evaluated on real and synthetic CAN data and compared to related work. Finally, in Section V, we present our conclusions.

II. RELEVANT BACKGROUND

A. CONTROLLER AREA NETWORK

The following is a brief description of the IDS relevant characteristics of the Controller Area Network. For a more detailed presentation we refer to [10], [11].

The Controller Area Network is a vehicle bus standard [11] designed to allow automotive Electronic Control Units to communicate with each other. This broadcast communication is done via messages, where each message type has a unique identifier ID. The identifier can be used to derive which signals are encoded in the message. Different IDs can encode different numbers of signals. For example, messages with a certain ID might encode the vehicle speed whereas a message with a different ID might contain information like the engine temperature and the engine speed.

The ID has further implications on the CAN bus. Specifically, a low ID implies sending with priority over all messages with a higher ID. Messages are only sent when having priority on the CAN bus and otherwise are resent with delay. Many message types that transmit physical values like the vehicle speed are sent periodically with a specified cycle time. Due to their high priority, cyclic messages with a low ID typically show a lower variance in the observed cycle times than messages with a high ID.

From the various characteristics each message carries, we consider the time stamp, the ID and a typically 8-byte payload field as relevant for a CAN IDS. Here, the payload refers to the signal values encoded in the message. The encoding of a signal value ranges from single bits to several bytes of the payload. A so-called CAN matrix specifies which payload bits encode which signal for each ID. The CAN matrix is considered intellectual property by most car producers which makes it difficult for many research groups to convert the raw binary payloads into the corresponding signals. In this paper, we assume that the decoding of raw payload bits into signal values has already been performed.

As a result, we get a data structure as seen in Table 1, where at each point in time exactly one ID is observed together with its associated signals. Hence, the received signal numbers and types are varying over time. This makes is particularly challenging to use any kind of algorithm, like a neural network approach, that requires the same input parameter set at each point in time.

B. SECURITY RELEVANT ATTACKS

The objective of an IDS is to detect intrusions into the CAN bus communication. We assume that the attacker has already gained access to the CAN bus, e.g. by hijacking one of the connected ECUs. This follows the path of demonstrated security relevant attacks [2]. We further assume that an attacker tries to influence the vehicle behavior by manipulating messages.

Our goal is to detect such attacks by identifying signals deviating from their "normal" behavior or signals breaking out of physical relationships. These physical relationships are typically complex, potentially unknown and hard to derive by rule based intrusion detection systems for CAN. Note that CANet is not designed to identify which message is explicitly attacked, but rather to detect if there is any attacked message.

C. DEEP LEARNING

In this section we briefly describe the neural network building blocks used in the proposed CANet architecture.

1) LONG SHORT-TERM MEMORY (LSTM)

The LSTM [12] is a neural network architecture specifically designed to work on time series [13] or natural language processing problems [14]. LSTMs belong to the class of recurrent neural networks (RNNs). Unlike fully connected neural networks they have feedback connections. They were originally developed to handle the vanishing gradient problem of standard RNNs [12] and are widely used nowadays.

2) AUTOENCODER

The class of neural network structures that is known as autoencoders has the objective to learn the structure of a training data set, e.g. the normal behavior of a technical system, in an unsupervised manner [15]. This is, no labeled data is needed in the training process. A trained autoencoder can be used to identify data points that deviate from the normal behavior and therefore can function as anomaly detection system [16]. The basic idea behind autoencoders is that high dimensional data that originates from some underlying system often contains correlations. Therefore, a meaningful embedding into a lower dimensional subspace typically exists. An autoencoder consists of two neural network blocks, an encoder and a decoder. Both can be realized by a set of consecutive fully connected layers. The task of the encoder is to map the high dimensional input data into the lower dimensional embedding space, which is called autoencoder bottleneck. The task of the decoder is to reconstruct the input data from its representation in the embedding space. Whereas the deviation between the real input data and its reconstruction should be small on normal data, large deviations are expected for anomalous data that has not been seen by the system in the training process.

3) EXPONENTIAL LINEAR UNIT (ELU)

ELU is a nonlinear activation function used in deep neural networks. It is defined as

$$ELU(x) = \begin{cases} x, & \text{if } x > 0 \\ a(e^x - 1), & \text{otherwise,} \end{cases}$$

where $a \geq 0$ is a constant. It has been shown that that ELUs outperform rectified linear units (ReLUs) in many tasks [17].

D. RELATED RESEARCH

In this section, we discuss previous approaches for intrusion detection on CAN bus data. In [7], a LSTM network structure for predicting the next payload of a single ID is proposed. A similar strategy is implemented in [18] on a mobile edge. Other deep learning based results include [8], where an autoencoder like network architecture is used on a sliding window over the appearances of a specific signal. In [9], a lightweight on-line detector of anomalies (LODA) is proposed. While these methods show promising results, they all build a model for a single time series operating on the values from one signal or signals of one ID.

We extend this by enabling CANet to work with all signals of multiple CAN IDs simultaneously. This gives our model the advantage that it can detect intrusions by inferring physical dependencies that exist between signals on the CAN bus. For example, assume that the attacker gains control over the ECU that sends the vehicle speed into the CAN bus (e.g. every 10ms) and replays the vehicle speed signals from a different traffic situation. It might be hard to detect this intrusion when only given access to the messages containing the vehicle speed. However, if the network also has access to messages of a different ECU that contain some correlated signal, say the engine speed (appearing e.g. every 5 ms), it gives CANet the possibility to infer that the current messages on the CAN bus are anomalous. Previous methods are not able of this reasoning because they do not possess a network topology capable of handling different messages appearing at different times and with different frequencies on the CAN bus.

Non neural network based methods for anomaly detection on CAN bus payloads include signature based methods [19], finger printing [20], clustering methods [21], fuzzy logic [22], Hidden-Markov-Model based methods [23], entropy based methods [24], [25], analyzing time intervals [26], one class support vector machine based methods [27], and consistency and frequency analysis on the edge [28]. A comprehensive review of the strengths and weaknesses of these and other non-payload based methods can be found in [3], [29].

III. CANet

CANet is an unsupervised learning method. The basic idea is to handle the challenging structure of CAN data by introducing an independent LSTM input model for each ID that can capture the temporal dynamics of the corresponding signals. The output of all input models is aggregated and fed into a fully connected subnetwork with an autoencoder structure. This enables the network to take interdependencies of signals of all IDs into account. At each point in time, all potential input signals are reconstructed. The reconstruction error between the true signal values and their reconstruction can be used to calculate the anomaly score. The topology of CANet with the corresponding layer sizes and activation functions are summarized in Table 2. A visualization of the architecture can be found in Figure 1.

TABLE 2. CANet architecture for detecting intrusions on the CAN bus.

CANet Architecture	
Layer Type	Size
LSTM per ID	Number of Signals of ID $\times$ h_{scale}
Joint Latent Vector	$N \times h_{scale}$
Fully Connected	$N \times h_{scale}/2$
ELU Activation	-
Fully Connected	$N - 1$
ELU Activation	-
Fully Connected	N
ELU Activation	-

A. NETWORK ARCHITECTURE

In order to describe the network architecture efficiently, we first establish some notations. Let $\mathbf{A} = \{A_1, \ldots, A_K\}$ be the ordered set of all $K \in \mathbb{N}$ considered IDs. For each ID $A \in \mathbf{A}$ we take n_A corresponding signals into account that

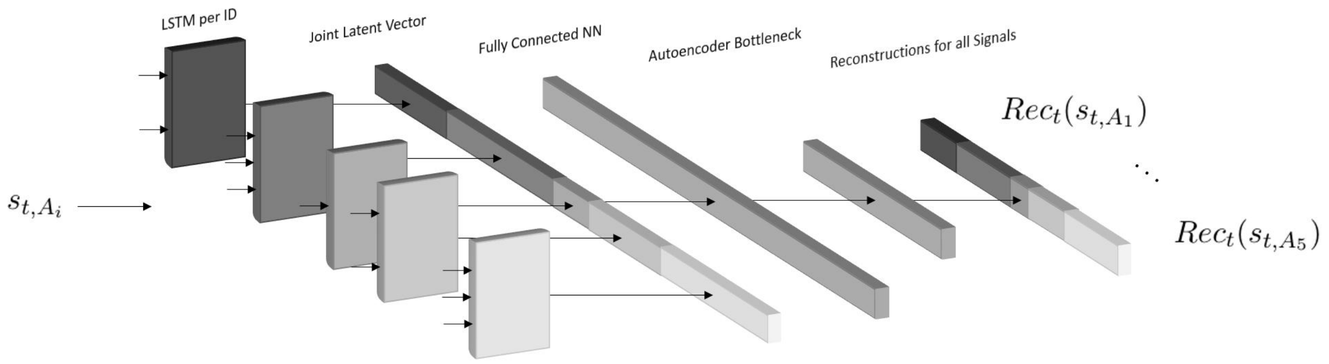

FIGURE 1. Schematic representation of the CANet architecture for intrusion detection on CAN bus data. Each ID has its own LSTM input model. When the payload s_{t,A_i} of an ID is fed into its input model, only the corresponding memory in the joint latent vector is updated. The entire latent vector is used to reconstruct the signals of all IDs. The deviation between the true input payload and the corresponding reconstruction is used for calculating the anomaly score.

are encoded in the payload. We denote the total number of signals by N. Whenever an ID $A \in \mathbf{A}$ is observed at time step t, we denote the vector containing the corresponding signal values by $s_{t,A} \in \mathbb{R}^{n_A}$. We hereby discretize the time.

The network architecture consists of an input LSTM subnetwork for every ID $A \in \mathbf{A}$. The i-th LSTM subnetwork is associated with ID A_i, has n_{A_i} inputs, and a hidden dimension of size $n_{A_i} \cdot h_{scale}$. Here, $h_{scale} \in \mathbb{N}$ represents the computational power of CANet. In the evaluation part, the performance with different h_{scale} values is compared (see Section IV-D). Whenever a new payload of an ID $A \in \mathbf{A}$ is fed through the corresponding LSTM subnetwork, its output of size $n_A \cdot h_{scale}$ is used to update the corresponding memory in the joint latent vector. This latent vector is realized as concatenation of the hidden states of all LSTM input models and therefore has length $N \cdot h_{scale}$. It represents and stores the current state of the CAN traffic. The joint latent vector is followed by a set of fully connected layers where the penultimate layer has strictly less neurons than the output layer, which has N neurons. The task of the output layer is to reconstruct all potential current input signals from all IDs.

During training, at each time step $t \in \mathbb{N}$ the payload of an ID, A_i say, is fed through its corresponding LSTM input model. It then is used to update the joint latent vector in order to reconstruct the payload at time step t for all $A \in \mathbf{A}$. Formally, the reconstruction R_t of the payload for the time step t is denoted by

$$R_t(s_{t,A_i}) = (Rec_t(s_{t,A_1}), \ldots, Rec_t(s_{t,A_K})),$$

where $Rec_t(s_{t,A_j})$ represents the reconstruction of the payload associated to ID A_j.

We then compare the true signal values from the payload of the current ID A_i with their reconstructions $Rec_t(s_{t,A_i})$. We use the quadratic error loss function, given by

$$loss(s_{t,A_i}) = ||Rec_t(s_{t,A_i}) - s_{t,A_i}||^2_{\ell_2}. \tag{1}$$

Since the temporal dependencies are stored for each ID separately in the corresponding LSTM subnetwork, the training process has the advantage that the model as a whole is not sensitive to the exact order of consecutive message IDs. In fact, in real CAN data, there is some variability in the order of IDs, even within a short time interval.

B. ANOMALY SCORE

The quadratic error between a signal and its reconstruction can be used to predict whether or not a signal at time step $t \in \mathbb{N}$ is anomalous. This prediction can be made by testing if the error is above a fixed threshold. Due to considering an unsupervised learning problem, the threshold must be chosen solely based on normal data. It is computed for each signal separately and is given by a percentile P of all corresponding quadratic errors on a small data set. During evaluation, an individual anomaly indicator is stored for every signal. Every time an ID is processed by the model, the anomaly indicator of the respective signals is updated and set to 1 if the reconstruction error exceeds the corresponding percentile P and set to 0 otherwise. The global anomaly score at a time step t is set to 1 if and only if at least one of the stored signal anomaly indicators is 1 and set to 0 otherwise.

IV. EXPERIMENTS

In this section, we evaluate our method on both, real and synthetic CAN data.

The real data was collected on a test vehicle. In our experiments, 13 IDs with a total number of 20 signals are taken into consideration. The signals are chosen in such a way that they contain physical values and that, for each signal, there is at least one other signal with a physical dependency to it. We divide about 13 hours of recorded data into 12.5 hours of training and 0.5 hours of test data. We only consider data representing the normal driving mode. Hence, we exclude e.g. starting and turning off the engine. All payloads are preprocessed into their signal value space (see Table 1).

In the case of the synthetic data, we consider a data set consisting of 10 different message IDs, each with different amounts of signals per ID and different noisy time frequencies. The total amount of signals is 20. The data is created in such a way that it is similar to real CAN traffic.

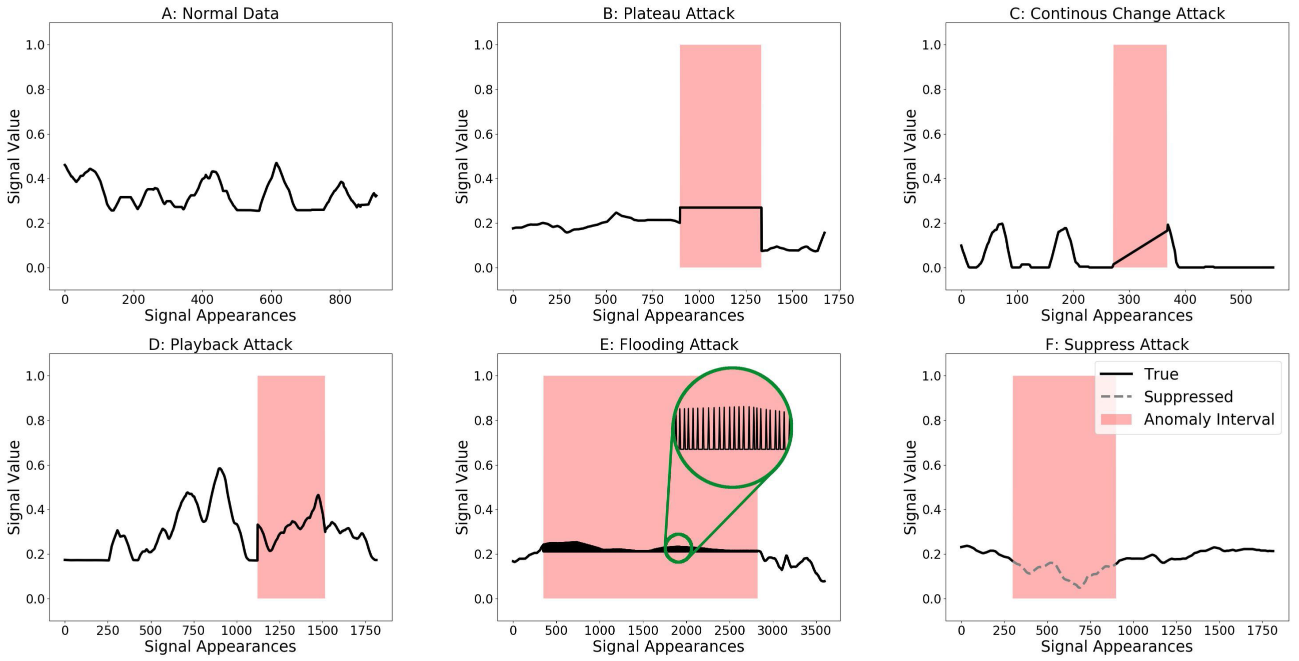

FIGURE 2. **Visualization of CAN data with different attack types on short time intervals. The flooding attack contains high frequency anomalies between its real signal values. In the suppress attack the dotted line represents real signal values that are not transmitted onto the CAN bus.**

The data contains physical values, counters and signals that are dependent on one or multiple other signals. We use a training data set of about 16.5 hours and a test data set of about 7.5 hours of CAN traffic. The data set is available at https://github.com/etas/SynCAN.

A. SIMULATED ATTACKS

In both, the real and the synthetic data set, the test data is divided into six subsets of equal time length. We use one subset to evaluate our model on normal data. The other five test data sets are used to evaluate our model on the following attack types:

1) *Plateau attack*: A single signal is overwritten to a constant value over a period of time, i.e. a jump or freezing the signal.
2) *Continuous change attack*: A signal is overwritten so that it slowly drifts away from its true value. This assumes that the attacker wishes to set a signal to a concrete value while trying to fool the IDS with realistic small changes in the signal.
3) *Playback attack*: A signal value is overwritten over a period of time with a recorded time series of values of that signal. The attacker hopes to trick the IDS by sending completely real looking signal values of a different traffic situation.
4) *Flooding attack*: The attacker sends messages of a particular ID with high frequency to the CAN bus. This attack is easier to perform in practice than the aforementioned ones, since the attacker does not need to control an ECU. It only requires to send additional messages to the CAN bus in order to "overwrite" the real message values.
5) *Suppress attack*: The attacker prevents an ECU from sending messages, for example, by turning it off. This kind of attack implies that messages of some particular ID do not appear in the CAN traffic for some period of time.

The length of a typical attack interval in both, the real and the synthetic data set, is between 2-4 seconds. A visualization of each attack type can be found in Figure 2.

B. NETWORK TRAINING DETAILS

In this section, we present the training details of our method in order to make the results reproducible. All code for training and evaluation is written in Python 3.5.3 with pyTorch 0.3.0 [30]. All computations are performed on a 3.5 GHz system with 4 cores and 32 GB of installed physical memory (RAM). We use the network architecture described in Table 2 for different h_{scale} values (cf. Table 4). The optimizer of choice is Adam [31] with a initial learning rate of 0.01. The input signals of the network are rescaled with a signal-wise 0-1 normalization.

We train the network for 1000 iterations with batch size 25. Every element in a batch is a series of 5000 consecutive messages at random starting position in the training data. At the beginning of each iteration the hidden and cell state vector of all LSTM models are initialized with zero. During a single iteration, a back-propagation is performed every 250 time steps in order to update the network weights. For a more robust training, the loss function is multiplied by a fixed scalar for different IDs. That is, the scalar is linearly smaller the more frequent its corresponding ID appears in the training data set. This is to ensure that IDs that appear more

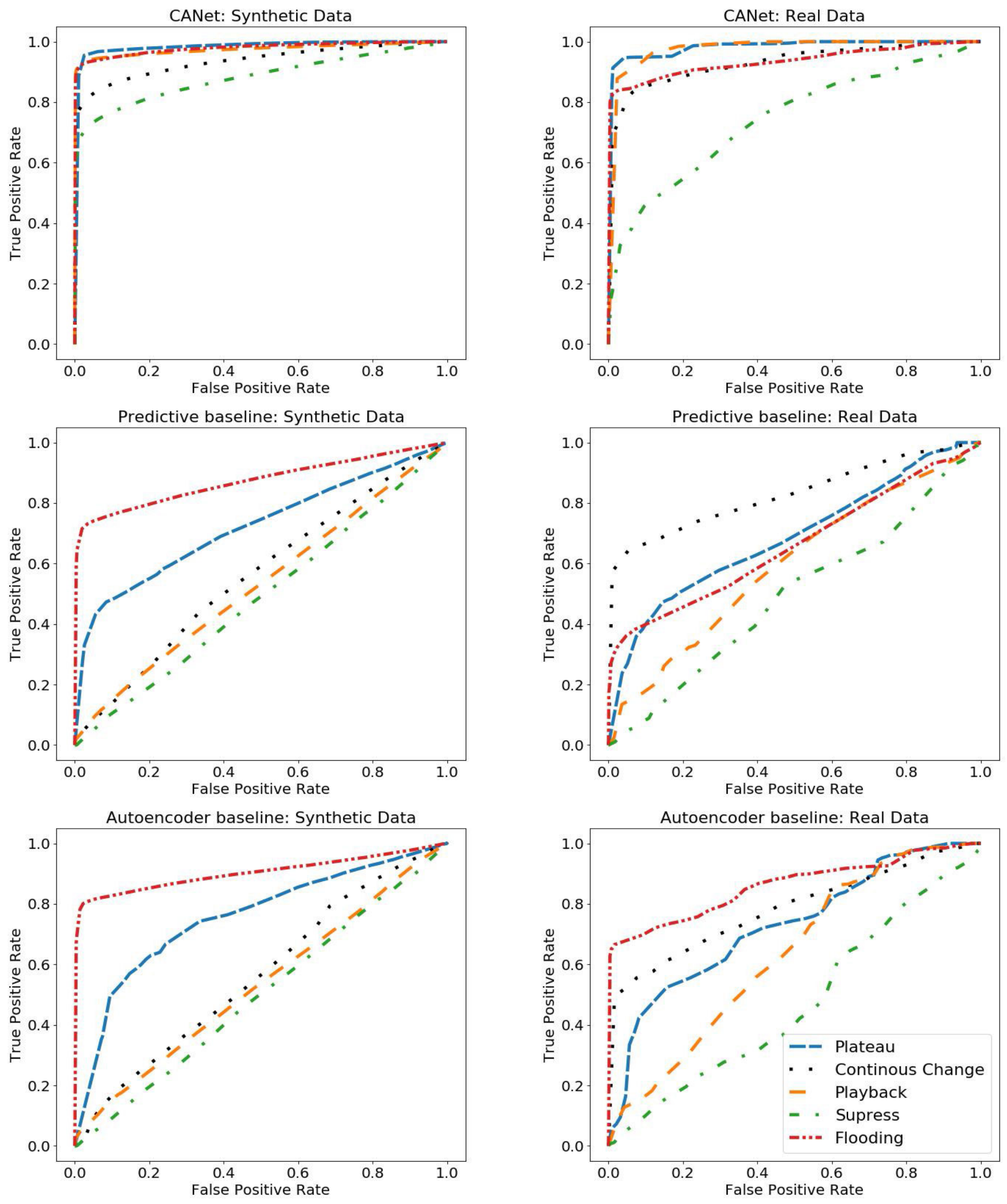

FIGURE 3. **ROC curves of CANet with $h_{scale} = 10$ and the baseline models on all attack types. The corresponding AUC values can be found in Table 3.**

often do not get more weight than less frequent IDs during training.

The training is performed on the training data set after removing a small subset to compute the thresholds for the anomaly score.

C. COMPARISON WITH RELATED RESEARCH

CANet is the first approach capable of handling the data structure of CAN bus data with multiple CAN IDs simultaneously within a single neural network model. Therefore, a one to one comparison with an existing method is not possible. In order to compare CANet with a baseline, we adapted the following methods:

1) *Predictive Baseline*: In [7], the basic idea is to learn a separate model for each ID. At each time step, the model predicts the payload of the next occurrence of its associated ID. The network directly processes the bit representation of the payload. As preprocessing step, the raw data is fed to a subnetwork of fully connected layers. The output is then processed by a combination of LSTM and fully connected layers that

TABLE 3. AUC of the ROC curves on real and synthetic CAN data.

Area Under the Curve (AUC)					
Synthetic Data					
Method	Plateau	Continuous	Playback	Flooding	Suppress
CANet	**0.983**	**0.936**	**0.974**	**0.979**	**0.882**
Predictive	0.722	0.561	0.530	0.874	0.489
Autoencoder	0.755	0.563	0.532	0.903	0.496
Real Data					
Method	Plateau	Continuous	Playback	Flooding	Suppress
CANet	**0.982**	**0.930**	**0.975**	**0.935**	**0.743**
Predictive	0.685	0.828	0.595	0.661	0.494
Autoencoder	0.727	0.782	0.633	0.860	0.470

perform the prediction. The difference between the true value and the prediction is used for the anomaly score. We adapt this method by training one predictive model per ID. Since we have access to the signal representation of the payload, we omit the preprocessing subnetwork and feed the signal values directly into the LSTM layers that are followed by a set of fully connected layers.

2) *Autoencoder Baseline*: In [8], an autoencoder model for a single signal is used. That is, at time step t the network has the task to reconstruct the input vector that consists of the signal values on a sliding window at the time steps $(t-7, \ldots, t)$. We use their network architecture to obtain one model for each signal.

Following our approach in Section III-B, we use a percentile P of the quadratic errors on a small data set (signal wise) to combine it to a final anomaly score.

D. EVALUATION

As a first evaluation step, the receiver operating characteristic (ROC) curve of CANet and the baseline methods is computed. For a clear presentation, we only consider $h_{scale} = 10$ for the CANet model. The ROC curves are computed by calculating the true and false positive rates with respect to the anomaly score and a suitable set of percentiles P (cf. Section III-B). The ROC curves of all models are presented in Figure 3.

The performance of a binary classifier can be measured by the area under the curve (AUC) value of its ROC curve. AUC values close to 1.0 indicate a good overall performance. The AUC values of all methods are summarized in Table 3. It can be seen that CANet outperforms both baseline approaches by a significant margin on every attack type on both, real and synthetic data.

The ROC curves and the AUCs confirm a good overall performance of CANet. However, for the in vehicle use case of a CAN IDS it is required to have a true negative rate close to 1.0. Hence, only the parts of the ROC curves with very small false positive rates are of interest for application. From now on, we follow this requirement by evaluating the anomaly score at the $P = 99.99\%$ percentiles. We evaluate the performance of CANet with $h_{scale} \in \{5, 10, 20\}$. A summary of the numerical results can be found in Table 4. The accuracies, true positive rates (i.e. rate of attacks that where successfully detected) and true negative rates (i.e. rate of normal data that was found to be normal) are presented in the table.

We find that in the real as well as in the synthetic data setting the CANet models identify normal data correctly in a solid way, with an accuracy typically larger than 0.99. The attack types plateau, continuous change and playback are detected reliably. The plateau and the playback attack show particular high detection rates, typically in the range of [0.85, 0.95]. The continuous change attack has a detection rate normally larger than 0.70.

TABLE 4. Summary of experimental results on real and synthetic CAN data.

Evaluation Table							
Synthetic Data							
Model Specification		Accuracy	True Positive Rate / True Negative Rate				
Method	h_{scale}	No Attack	Plateau	Continuous	Playback	Flooding	Suppress
CANet	5	0.991	0.896 / 0.980	0.740 / 0.994	0.896 / **0.997**	**0.900** / **0.997**	0.496 / **0.996**
CANet	10	0.990	**0.955** / 0.975	0.765 / 0.994	**0.905** / **0.996**	**0.901** / **0.996**	**0.613** / **0.996**
CANet	20	0.992	0.885 / **0.993**	**0.771** / **0.996**	**0.906** / **0.997**	0.884 / **0.997**	0.581 / **0.995**
Predictive	-	**0.996**	0.330 / 0.974	0.015 / 0.994	0.020 / 0.996	0.644 / 0.994	0.003 / 0.993
Autoencoder	-	0.983	0.361 / 0.926	0.016 / 0.975	0.029 / 0.995	0.688 / 0.995	0.001 / 0.993
Real Data							
Model Specification		Accuracy	True Positive Rate / True Negative Rate				
Method	h_{scale}	No Attack	Plateau	Continuous	Playback	Flooding	Suppress
CANet	5	0.996	**0.937** / 0.963	**0.792** / 0.975	0.852 / 0.911	**0.808** / 0.943	0.082 / **0.997**
CANet	10	0.994	0.913 / 0.988	0.701 / 0.985	**0.878** / **0.977**	0.802 / 0.996	0.176 / 0.989
CANet	20	0.995	**0.936** / 0.968	0.724 / **0.988**	0.862 / 0.954	0.761 / 0.992	**0.254** / 0.991
Predictive	-	0.995	0.269 / 0.949	0.577 / **0.987**	0.134 / 0.964	0.182 / **0.998**	0.001 / **0.998**
Autoencoder	-	**0.999**	0.055 / **0.992**	0.491 / 0.983	0.079 / **0.977**	0.627 / 0.996	0.007 / 0.995

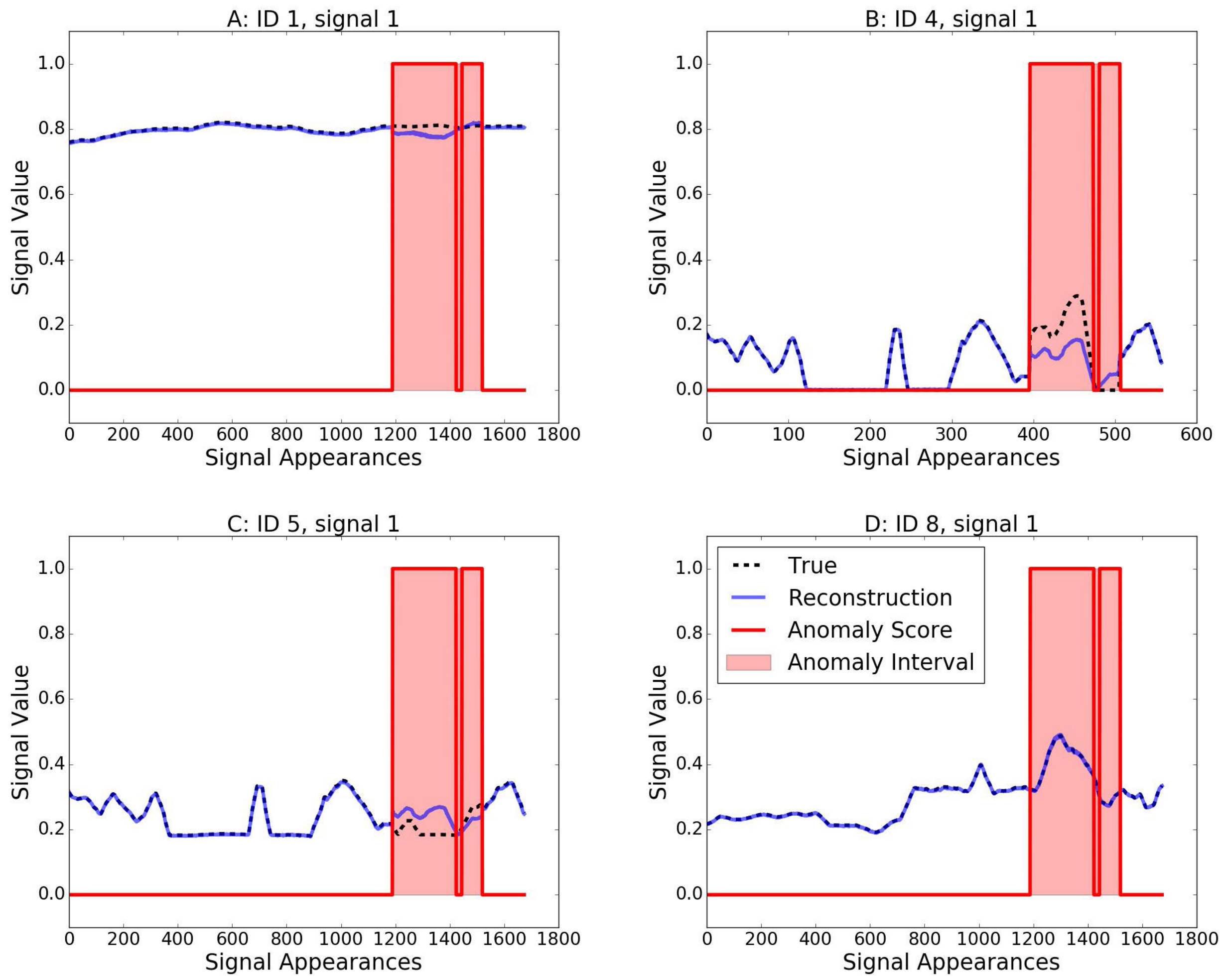

FIGURE 4. **The plots show four different signals of the synthetic CAN data set. They are extracted on the same time interval along with their corresponding reconstruction of CANet. The different number of signal appearances in *B* compared with *A*, *C* and *D* are a result of different message frequencies. A playback attack on *B* is performed. The corresponding attack interval is shaded in all plots. An intrusion is detected by CANet if the anomaly score (red line) equals one. It can be seen that on non-attacked data, the reconstruction (straight blue line) of the original signal (dashed black line) is almost exact. In the attack interval, deviations between the true signal and its reconstruction cannot only be observed in *B* but also in *A* and *C*. This is due to functional dependencies between these three signals. In contrast, *D* has no functional dependencies with *A*, *B* and *C*. Therefore, it remains unaffected by the attack.**

The evaluation is performed point wise, i.e. per time step. Hence, the method detects at average the vast majority of points in each attack interval correctly. A visualization of this finding can be found in Figure 4, where it can be seen that the method detects most of each attack interval and normal data correctly. Unsurprisingly, like in the AUC comparison, CANet outperforms the predictive and the autoencoder baseline by a significant margin. Moreover, the baseline approaches only detect the first few attacked messages of the attack interval but identify the rest of the interval as normal (see Figure 5). The baseline approaches perform particularly poorly on the playback attack. That is, a playback attack and the true signal are mostly indistinguishability if only one signal is taken into account (see Figure 2-D). Only at the beginning and the end of a playback attack interval there might be an unexpected jump. On the other hand, CANet has access to all signals. Hence, it can exploit physical dependencies between signals to find such attacks. This is visualized in Figure 4, where a short synthetic data interval is shown. It contains the corresponding plots of four different synchronized signals. The signal of the upper right plot *B* contains a playback attack. We observe that for all signals the reconstruction on normal data is usually really accurate. During the attack interval deviations between the true data and its reconstruction can be found. Note that this deviation also appears in signals that are not explicitly attacked but only correlated in some way with the attacked signal, whereas signals without any correlations stay unaffected. Furthermore, over an attack interval typically not the entire attack is detected as such. This is expected, because an attacked signal and its original counterpart may have similar values in some parts of the attack window. For example, in case of a continuous change attack (see Figure 2-C) the modified signal values lie in a realistic range at the beginning of

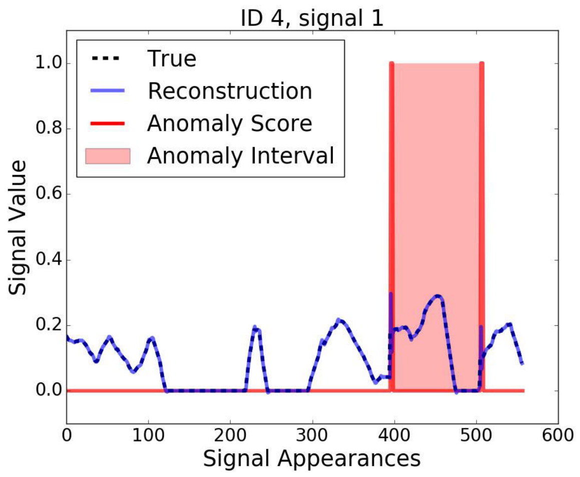

FIGURE 5. Performance of the predictive baseline on the same time interval as in Figure 4. This method only detects the first few elements of the attacked interval correctly. The second peak in the anomaly score is technically a false classification at the jump from the anomalous signal back to normal data.

the attack. As a consequence, the model detects an attack only after the deviation between the original and the modified signal exceeds a certain threshold.

Just for comparison, we also evaluate our models on two other common attack types: suppress and flooding. These attacks can be detected by a rule based approach in a straight forward way, e.g. by analyzing the frequencies of each ID [26]. Our model does not have access to the time stamp but only to the order in which IDs are recorded and is therefore not specifically designed to find such attacks. However, it still detects flooding attacks with a high true positive rate, whereas it struggles to detect suppress attacks. This is expected since the values that are added with a high frequency into the CAN bus during a flooding attack are much easier to be found than the gradual change of the network state that is the consequence of not sending a certain ID at all. When comparing with the baselines, we find that CANet is superior in all aspects. Nevertheless, both, the predictive and the autoencoder baseline, show relatively good results on the flooding attack. This is because in between the messages from the flooding attack the normal data points are still taken into account. Hence, during a single attack interval many anomalous large jumps in signal values might be found (see Figure 2-E).

We believe that for in vehicle usage it is reasonable to pair CANet with the strengths of some rule based IDS. For example, monitoring frequencies in order to detect e.g. flooding or suppress attacks can be efficiently covered by a set of simple rules [26].

We find that the different choices for the parameter h_{scale} have a relatively low effect on the performance of the models. Even small models with $h_{scale} = 5$ perform reasonably well. This is especially interesting for a potential use of such models on an embedded device where memory and computational power are limited.

When comparing the models on the real and synthetic data, we find that the performance is in a similar range in most cases. We believe that the synthetic data set is a good benchmark to test CAN IDS models, even if the data is somewhat "cleaner" than in the real case.

Since we believe that in real application, finding a large number of attack intervals is more important than a high overall point wise accuracy on attacks (i.e. true positives), we investigate this by redefining what it means that an attack is found (see Figure 6). That is, we define an attack as the entire period of time during which the attack is performed, i.e. an attack interval. We compute the percentage of attack intervals that are detected. Here, the criterion for identifying an interval as anomalous is that at least $Q\%$ of that interval is detected point wise as anomaly. We can see in Figure 6 that based on this definition CANet finds most anomaly intervals if $Q\%$ does not get too large. This is true for both, the real and the synthetic data case. However, both baseline methods have very low detection rates of anomaly intervals even for small $Q\%$ (see Figure 6).

Summing up the results, we find that CANet is capable of reliably detecting attacks on CAN bus data, while performing solidly on normal data. Our main findings are:

1) The presented architecture is the first method that is capable of handling the difficult data structure from signals of multiple CAN IDs in a single model.
2) CANet outperforms the baseline CAN IDS methods by a significant margin on all selected evaluation criteria.
3) Our model is trained in an unsupervised manner, i.e. it has never seen attacks during training. Hence, it is expected that the model finds further unknown attack scenarios beyond the ones presented in this paper.

E. RISKS AND BENEFITS OF NEURAL NETWORK BASED IDS MODELS

The biggest advantage of using machine learning based approaches, such as the one presented in this manuscript, is that they are potentially capable of detecting unknown intrusions. That is, they are successful in a task in which most other methods fail. Classically, for each possible attack scenario, a defense mechanism must be chosen. However, this process is highly time consuming and requires a significant amount of CAN bus domain expert knowledge for a successful detection. Here, neural networks significantly reduce both, the development time and the required CAN domain knowledge.

On the other hand, the output of machine learning based methods can be complicated to analyze which makes it difficult to execute an automatic response once an intrusion is detected. Furthermore, neural networks require a large amount of training data and are typically more computational and memory consuming than many other approaches. A visualization of the approximate memory requirements of CANet can be found in Figure 7, where for simplification each ID consists of exactly one signal.

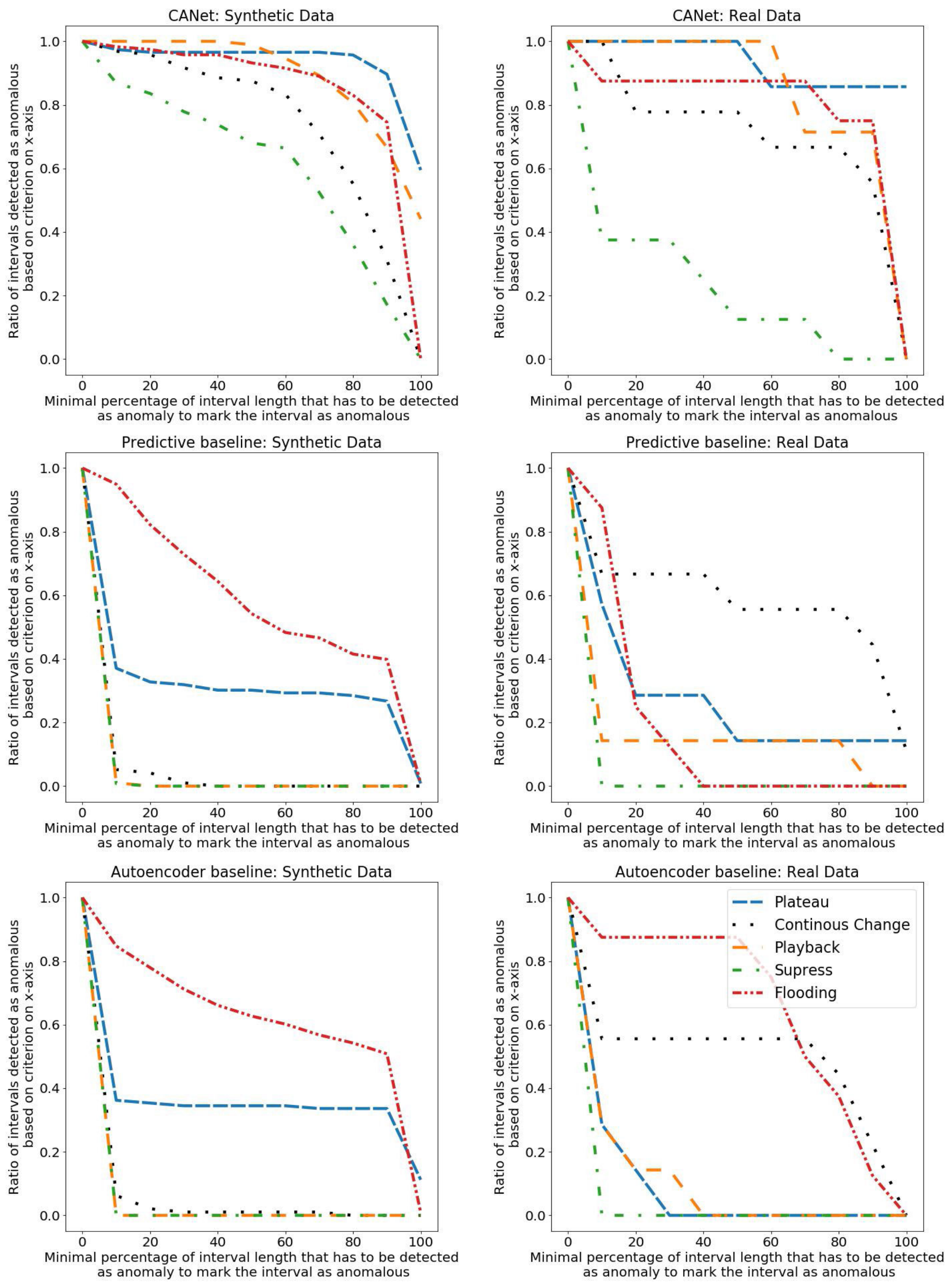

FIGURE 6. The plot represents the ratio of detected attacks for each attack scenario. Here an attack is considered as the interval in which the attack is performed. The criterion for an attack interval to be detected is: At least Q% of the interval have been point wise detected as attack (x-axis). The CANet model was used with $h_{scale} = 10$. By comparing the plots, it can be seen that CANet outperforms the other two approaches by a large margin.

F. REPRODUCIBILITY

Most CAN bus based intrusion detection methods are tested on real data. However, publishing real CAN traffic and the corresponding CAN matrix is usually not possible, since it is considered intellectual property by most car producers. Hence, to the best of our knowledge, there is no standard

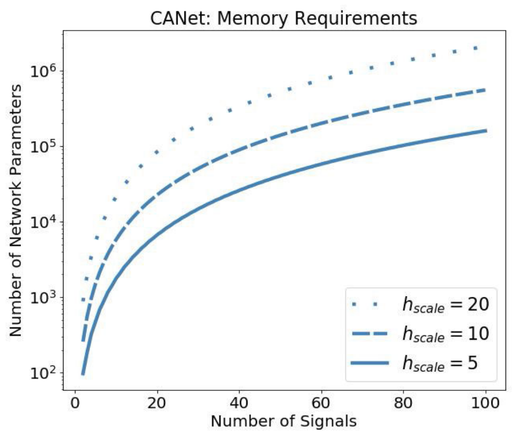

FIGURE 7. Visualization of the approximate number of network parameters in dependence of h_{scale} and the number of signals under consideration.

data set for comparing methods. We try to close this gap by evaluating our model on both real and synthetic data and we make the synthetic data publicly available.[1] We hope that this simplifies the work of researchers to compare their work with a baseline.

V. CONCLUSION

Cars are getting more and more connected. This opens ways for attacking the CAN bus of automobiles remotely. Since attacks can have a major impact on traffic safety, it is desirable that such attacks are detected in a robust manner.

We present CANet, a novel neural network architecture that is trained in a unsupervised manner to detect intrusions and anomalies on the CAN bus. It is the first model in the literature capable of working on messages with different IDs simultaneously. CANet models have a high true negative rate, typically over 0.99, which is necessary for real world applications. Additionally, along with the high true negative rate we are able to detect a large amount of the unknown attacks, both on real and synthetic data correctly.

ACKNOWLEDGMENT

(Markus Hanselmann and Thilo Strauss contributed equally to this work.) The authors would like to thank J. Gramm, M. Oechsle, and the ETAS Machine Learning Group for the useful discussions.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, views or statements, either expressed or implied, of the affiliated organizations of the authors.

REFERENCES

[1] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, "Comprehensive experimental analyses of automotive attack surfaces," in *Proc. USENIX Secur. Symp.* San Francisco, CA, USA, 2011, pp. 447–462.

[1]The data is publicly available at https://github.com/etas/SynCAN.

[2] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," *Black Hat USA*, vol. 2015, p. 91, Aug. 2015.
[3] A. Tomlinson, J. Bryans, and S. A. Shaikh, "Towards viable intrusion detection methods for the automotive controller area network," in *Proc. 2nd ACM Comput. Sci. Cars Symp.*, 2018, pp. 1–9.
[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[5] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.
[6] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.
[7] A. Taylor, S. Leblanc, and N. Japkowicz, "Anomaly detection in automobile control network data with long short-term memory networks," in *Proc. IEEE Int. Conf. Data Sci. Adv. Analytics (DSAA)*, Oct. 2016, pp. 130–139.
[8] M. Weber, G. Wolf, E. Sax, and B. Zimmer, "Online detection of anomalies in vehicle signals using replicator neural networks," in *Proc. 6th Escar USA*, Ypsilanti, MI, USA, Jun. 2018, pp. 1–14.
[9] M. Weber, S. Klug, E. Sax, and B. Zimmer, "Embedded hybrid anomaly detection for automotive can communication," in *Proc. 9th Eur. Congr. Embedded Real Time Softw. Syst. (ERTS)*, 2018, pp. 1–11.
[10] M. Bozdal, M. Samie, and I. Jennions, "A survey on CAN bus protocol: Attacks, challenges, and potential solutions," in *Proc. Int. Conf. Comput., Electron. Commun. Eng. (iCCECE)*, Aug. 2018, pp. 201–205.
[11] *Road Vehicles—Controller Area Network (CAN)—Part 2: High-Speed Medium Access Unit*, Standard ISO 11898-2, International Organization for Standardization (ISO), 2016.
[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[13] F. A. Gers, D. Eck, and J. Schmidhuber, "Applying LSTM to time series predictable through time-window approaches," in *Neural Nets WIRN Vietri-01*. London, U.K.: Springer, 2002, pp. 193–200.
[14] A. Graves, N. Jaitly, and A.-R. Mohamed, "Hybrid speech recognition with deep bidirectional LSTM," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, Dec. 2013, pp. 273–278.
[15] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," in *Proc. ICML Workshop Unsupervised Transf. Learn.*, Jun. 2012, pp. 37–49.
[16] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*. New York, NY, USA: ACM, 2017, pp. 665–674.
[17] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–14.
[18] K. Zhu, Z. Chen, Y. Peng, and L. Zhang, "Mobile edge assisted literal multi-dimensional anomaly detection of in-vehicle network using LSTM," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 4275–4284, May 2019.
[19] I. Studnia, E. Alata, V. Nicomette, M. Kaâniche, and Y. Laarouchi, "A language-based intrusion detection approach for automotive embedded networks," in *Proc. 21st IEEE Pacific Rim Int. Symp. Dependable Comput. (PRDC)*, 2014, pp. 1–11.
[20] K.-T. Cho and K. G. Shin, "Fingerprinting electronic control units for vehicle intrusion detection," in *Proc. USENIX Secur. Symp.*, 2016, pp. 911–927.
[21] A. Tomlinson, J. Bryans, and S. A. Shaikh, "Using a one-class compound classifier to detect in-vehicle network attacks," in *Proc. Genetic Evol. Comput. Conf. Companion (GECCO)*. New York, NY, USA: ACM, 2018, pp. 1926–1929.
[22] F. Martinelli, F. Mercaldo, V. Nardone, and A. Santone, "Car hacking identification through fuzzy logic algorithms," in *Proc. IEEE Int. Conf. Fuzzy Syst. (FUZZ-IEEE)*, Jul. 2017, pp. 1–7.
[23] S. N. Narayanan, S. Mittal, and A. Joshi, "OBD_SecureAlert: An anomaly detection system for vehicles," in *Proc. IEEE Int. Conf. Smart Comput. (SMARTCOMP)*, May 2016, pp. 1–6.
[24] M. Muter and N. Asaj, "Entropy-based anomaly detection for in-vehicle networks," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2011, pp. 1110–1115.
[25] M. Marchetti, D. Stabili, A. Guido, and M. Colajanni, "Evaluation of anomaly detection for in-vehicle networks through information-theoretic algorithms," in *Proc. IEEE 2nd Int. Forum Res. Technol. Soc. Ind. Leveraging Better Tomorrow (RTSI)*, Sep. 2016, pp. 1–6.

[26] H. M. Song, H. R. Kim, and H. K. Kim, "Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Jan. 2016, pp. 63–68.
[27] O. Avatefipour, A. S. Al-Sumaiti, A. M. El-Sherbeeny, E. M. Awwad, M. A. Elmeligy, M. A. Mohamed, and H. Malik, "An intelligent secured framework for cyberattack detection in electric Vehicles' CAN bus using machine learning," *IEEE Access*, vol. 7, pp. 127580–127592, 2019.
[28] F. Guo, Z. Wang, S. Du, H. Li, H. Zhu, Q. Pei, Z. Cao, and J. Zhao, "Detecting vehicle anomaly in the edge via sensor consistency and frequency characteristic," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 5618–5628, Jun. 2019.
[29] O. Y. Al-Jarrah, C. Maple, M. Dianati, D. Oxtoby, and A. Mouzakitis, "Intrusion detection systems for intra-vehicle networks: A review," *IEEE Access*, vol. 7, pp. 21266–21289, 2019.
[30] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *Proc. NIPS Workshop Autodiff*, 2017, pp. 1–4.
[31] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2015, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980

MARKUS HANSELMANN received the Diploma degree in mathematical sciences from the University of Stuttgart, in 2008, and the Ph.D. degree in mathematical sciences from the Ludwig Maximilian University of Munich, in 2012. He did his research work at the University of Stuttgart and the University of Oxford. He worked for almost two years with the Allianz Group before joining ETAS GmbH, in 2014, where he has served in the Product Development Team of ETAS ASCMO. He switched to the Machine Learning Group, ETAS, in 2017. His research interests include machine learning, security, and analytic number theory.

THILO STRAUSS received the M.Sc. and Ph.D. degrees in mathematical sciences from Clemson University, in 2011 and 2015, respectively. He has served as a Senior Postdoctoral Research Fellow with the University of Washington, Seattle, from 2015 to 2017, and joined the Machine Leaning Group, ETAS GmbH, in 2017. His research interests include machine learning, security, Bayesian statistics, and computer vision.

KATHARINA DORMANN received the master's degree in computer science from the Karlsruhe Institute of Technology, in 2017. She joined Powertrain Solutions at Robert Bosch GmbH, in 2018. Her research interest is in machine learning and security.

HOLGER ULMER received the Diploma degree in physics from the University of Konstanz, in 1997, and the Ph.D. degree in computer science with focus on artificial intelligence from the University of Tübingen, in 2004. He worked several years as a Software Developer with Daimler AG. In 2004, he started as a Research Engineer and the Project Manager with Robert Bosch GmbH. From 2009 to 2017, he was the Project Manager of ETAS ASCMO, a machine learning application for model-based ECU-calibration, especially for combustion engine ECUs. Since 2017, he has been serving as the Chief Expert of Machine Learning at ETAS GmbH. His research is on novel applications of machine learning for the automotive industry.