Received 9 September 2020; accepted 15 September 2020. Date of publication 18 September 2020;
date of current version 13 October 2020. The review of this article was arranged by Associate Editor M. Pan.

Digital Object Identifier 10.1109/OJVT.2020.3024755

Attention-Based Event Characterization for Scarce Vehicular Sensing Data

NIMA TAHERIFARD [1], MURAT SIMSEK [1] (Senior Member, IEEE), CHARLES LASCELLES [2], AND BURAK KANTARCI [1] (Senior Member, IEEE)

[1] School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, ON, Canada
[2] Product Development, Raven Connected, 441 MacLaren St, Ottawa, ON K2P 2H3, Canada

CORRESPONDING AUTHOR: BURAK KANTARCI (e-mail: burak.kantarci@uottawa.ca)

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) under Grant RGPIN/2017-04032 and Ontario Centres of Excellence (OCE) TalentEdge Internship Project 32815 in collaboration with Raven Connected.

ABSTRACT Characterizing risky driving behavior is crucial in a connected vehicle environment, particularly to improve driving experience through enhanced safety features. Artificial intelligence-backed solutions are vital components of the modern transportation. However, such systems require significant volume of driving event data for an acceptable level of performance. To address the issue, this study proposes a novel framework for precise risky driving behavior detection that takes advantage of an attention-based neural network model. The proposed framework aims to recognize five driving events including harsh brake, aggressive acceleration, harsh left turn and harsh right turn alongside the normal driving behavior. Through numerical results, it is shown that the proposed model outperforms the state-of-the-art solutions by reaching an average accuracy of 0.96 and F1-score of 0.92 for all classes of driving events. Thus, it reduces the false positive instances compared to the previous models. Furthermore, through extensive experiments, structural details of the attention-based neural network is investigated to provide the most viable configuration for the analysis of the vehicular sensory data.

INDEX TERMS Attention model, auto-encoder, connected vehicles, LSTM, machine learning, recurrent neural networks, vehicular sensing.

I. INTRODUCTION

With the advent of connected and autonomous driving paradigm, detection of risky driving behavior (e.g., harsh cornering, harsh braking, aggressive acceleration) has become an essential component in a connected vehicle setting [1]. There have been a variety of solutions presented to characterize such events, which have proven to be effective. Driving event characterization systems as the fundamental basis of accident prevention models coupled with low latency vehicular connectivity provided by 5G and Beyond [2] can be utilized in intelligent transportation systems (ITS) for centralized traffic control systems. Centralized control systems can be effective tools for road safety and congestion control [3].

A major building block of an intelligent transportation system is the Artificial Intelligence (AI) backbone to detect the behavior of the system entities [4]. Some event characterization approaches leverage visual data to detect or predict upcoming events. Using camera feed, existing systems can extract information such as the rate of change in velocity, direction, and the proximity to the surrounding objects to be used in statistical models. Statistical models are designed to output the occurrence probability of the road events from the visual context. It is worth to note that visual systems are dependant on the line-of-sight of the visual sensors which can be blocked at times [5], therefore for the sake of reliability, in-vehicle sensory data are needed for precise and reliable analysis of vehicular events.

Most commonly used event characterization methods are the threshold-based rules applied to Inertial Measurement Unit (IMU) sensor data. Such methods employ diverse signal processing and filtering techniques to acquire noiseless data that best reflect the driving event characteristics. The threshold-based models categorize the events upon the signals exceeding static or dynamic thresholds [6]. The static thresholds are usually acquired based on physical characteristics of the vehicles in experimental settings. However, dynamic threshold-based systems often utilize more complex mathematical calculations to obtain the desired thresholds at any time. The threshold-based methods rely heavily on the physical characteristics of an individual vehicle [7] for effectiveness; hence threshold-based systems cannot be easily adopted world-wide.

To build systems that can be widely used in an intelligent vehicular environment, AI-based models can bypass the aforementioned issues by learning the characteristics of the driving events. Recurrent neural networks [8] are a category of neural networks that are capable of learning the characteristics of high frequency signals. Recurrent networks do not build on the physical features of a vehicle; therefore allowing for the models to be globally used on different types of vehicles. On the other hand, neural networks are data hungry and are able to achieve higher levels of accuracy and precision under the availability of sufficient volume of data for training [9]. Although neural networks can overcome the shortcomings of the previous issues concerning accuracy and reliability in the detection of various event types from vehicular sensory data, the lack of anomalous driving patterns to properly train a neural network for the task remains a challenge.

In our earlier work, we proposed a long short-term memory (LSTM)-based auto-encoder network which resulted in 0.93 accuracy in classifying the risky driving behaviour under limited training data [10]. This article substantially differs from the previous work by proposing a novel solution for the first time to characterize risky driving behavior using limited vehicular sensor data while minimizing the false positives which are the primary factors for low reliability of a detection system. To this end, an attention-based auto-encoder network is proposed to reconstruct and precisely classify driving event data. Specifically, the attention-based neural network performs a self-supervised task to encode behavior characteristics of the input event data as fixed-size vectors. The encoded representations is then used by the decoder network to reconstruct the input signal. The decoder network is capable to output accurate synthetic signals and classify the signals through an attention operation which enables the network to gain importance information about the signals. Extensive experiments are carried out to study the effect of the network internal structure on the characterization task and to enable the maximum network performance potential. Experimental results show that the proposed attention-based neural network model can result in an average accuracy of 0.96 and an F-1 score of 0.92 for all classes of driving events.

The rest of the article is organized as follows: section II discusses the state-of-the-art driving event characterization methods as well as the latest advancements in neural networks. Section III investigates the proposed attention auto-encoder model in further details. The event characterization performance as well as internal structure investigation results are presented in Section IV. Finally, Section V concludes the article and discusses future research directions, challenges and open issues.

II. RELATED WORK

Recent research shows that vehicular event characterization can be achieved through a variety of approaches including threshold-based and intelligent models. Threshold-based methodologies generally fall into static and dynamic systems while intelligent systems can be categorized based on the system input detailed in this section. Intelligent systems perform the characterization task by using either visual driving context, inertial sensor event data, or hybrid in which one type of data is applied to augment the other input [11]. Moreover, there is a growing body of studies in the field of neural networks that makes time-series analysis via neural networks more viable than before [12].

A. THRESHOLD-BASED SYSTEMS

The systems in this category mainly depend on the rate of change in inertial or positional sensor data during a driving event. Noise reduction, force monitoring, and unsafe threshold calculation are the major components of the research related to threshold-based systems [13].

In [14], the authors propose a static threshold applied on a unique system that combines velocity and acceleration of the vehicles while considering the vehicle dynamics and the road conditions to classify driving behaviors. The experiments are conducted on pre-recorded minibus taxi trip data in order to characterize the driving behavior which is beneficial for fleet management systems. Driving speed and correlation between lateral acceleration and velocity are studied in [15] where the authors conclude that lateral acceleration and vehicle speed are inversely related to each other by analyzing trip data of vehicles on highways with different topographies and speed limits. The study shows that lateral acceleration decreases as the vehicle speed increases, therefore lateral threshold value should be dynamically changing with the speed of the vehicles. By utilizing positional satellite data, the study in [16] proposes an irregular driving identification system. The proposed model in the study aims to increase positional measurement accuracy to less than 50 centimeters and were able to accurately identify driving behavior applying static threshold to the calibrated GPS data. A driving identifier system is proposed in [17] so to recognize dangerous and non-dangerous modes. The proposed system is reported to be useful for insurance fee estimation by first categorizing the sensor data into four distinct events using their pre-defined set of rules, and then applying static thresholds in order to label each event as dangerous or non-dangerous.

Hu *et al.* [18] aim to dynamically identify the normal driving behavior and alert the abnormal events when the measurements exceed the normal region. Smartphone-based detection systems are regularly studied in the literature [19]–[21]. Such studies rely on the accessible on-board sensors smartphones provide, mainly accelerometer, gyroscope, and GPS [22],

[23]. Vavouranakis *et al.* [24] contribute to sensor accuracy improvements with sensor calibration and data pre-processing methods. Using the calibrated sensor data, researchers design a driving event recognition system that can characterize safe and unsafe events. Furthermore, several studies attempt to provide more sophisticated mathematical calculations [25] to compensate the inaccurate measurements of portable sensors [26], [27]. These systems aim to distinguish between device and vehicle movements in order to reduce false event characterizations caused by device movements.

B. INTELLIGENT EVENT RECOGNIZER SYSTEMS

Safety, efficiency, and driving comfort form the highest priority targets for the automotive industry that are aimed to be ensured through data-driven approaches [28]. Modern automotive industry relies on machine learning and neural networks for driver assistance and autonomous driving systems [29]–[31]. Machine learning algorithms are also proven to be effective in the connectivity infrastructures such as ITS [32], [33]. Computer vision methods have been in the center of focus in the literature to characterize driving events [34], [35]. The systems of this type mainly extract abnormal information from visual context taken from dashboard mounted camera sensors. Maaloul *et al.* The research in [36] adopts a statistical approach on optical flow of road videos to intelligently define driving accidents. A combined model of road condition data and driver's eye movement detection is proposed in [37] to predict driving behaviors ahead of time. The research in [38] presents the idea of driver's brain signal processing with machine learning-based clustering and classification algorithms to study the driving behavior.

There have been several innovations in the field of signal processing using neural networks [39]. In particular, recurrent neural networks (RNNs) are specifically designed to store and learn features from time dependent and sequential data. The major issue of using such networks for high frequency signal processing is the phenomenon of vanishing gradient [40] that causes the recurrent networks to discard longer-term features. However, novel resolutions are proposed in [41], [42] which enables feasible longer-term data processing using RNNs. Moreover, auto-encoders (AEs) that are suitable tools for analysis where data is limited, are studied for anomaly detection [43], [44]. For instance, Malhotra *et al.* [45] attempt to perform 'Remaining Useful Life (RUL)' analysis of machines incorporating AE networks and anomaly detection techniques. Safe airplane navigation analysis is performed on rare data using such techniques in [46]. As stated by the researchers, AE networks have proven to be more reliable input representation schemes to systems.

In addition to these, attention-based neural network model [47] is proposed to mimic human brain behavior in a neural network. Attention is expanded to various use-cases including the AE networks [48], [49], and introduces the benefit of selectively focusing on sections of the input data that contain the desired events.

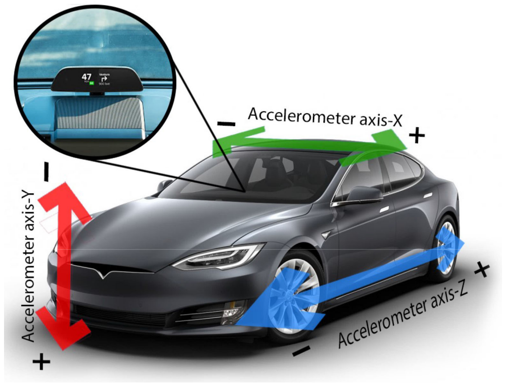


FIGURE 1. Raven sensor orientation. The android-based device is mounted on the dashboard.

Inspired by the recent research, our work utilizes attention-based encoder-decoder networks to advance driving event characterization systems and lower the false positives in the case of limited volume of training data. In order to achieve lower prediction delay, IMU data is utilized as the system input to gain direct measurements of forces applied to vehicles during the events [50]. Using the attention model, it is possible to strengthen the training of characterization network with precisely reconstructed data which leads to higher inference accuracy. Indeed, the impact of hyper-parameters and dimensions of the network require thorough analysis, which are also in the scope of this study as presented in the next sections.

III. METHODOLOGY

In this section, the process flow and the data gathering method are discussed and illustrated in Fig. 3.

At the beginning of the process, the sensing device, i.e. Raven, starts recording the accelerometer data along three directions. The signals are then sent to the pre-processing module to be distributed to the network. In the pre-processing module of the system, the event signals are sliced into the desired length signal windows. The windowed signals are duplicated and varied by random noise before being sent to the generator module for offline training process. Alternatively, real-time signals are sent to the characterization module, directly, for real-time inference process.

The second step is introduced to supplement the training data with more variety and quantity. In this module, the underlying behavioral features of the events are learned from the noisy variants of the input signal utilizing a denoising auto-encoder network [51]. The learned features are then passed through the decoder network to populate the training dataset with synthetically created signals. Further details of the module along with existing experimental proofs of the network's legitimacy is provided in the corresponding section.

Finally, the characterization network is trained with the more robust training dataset created by the generator module. The trained network is then used to infer the live-signals and

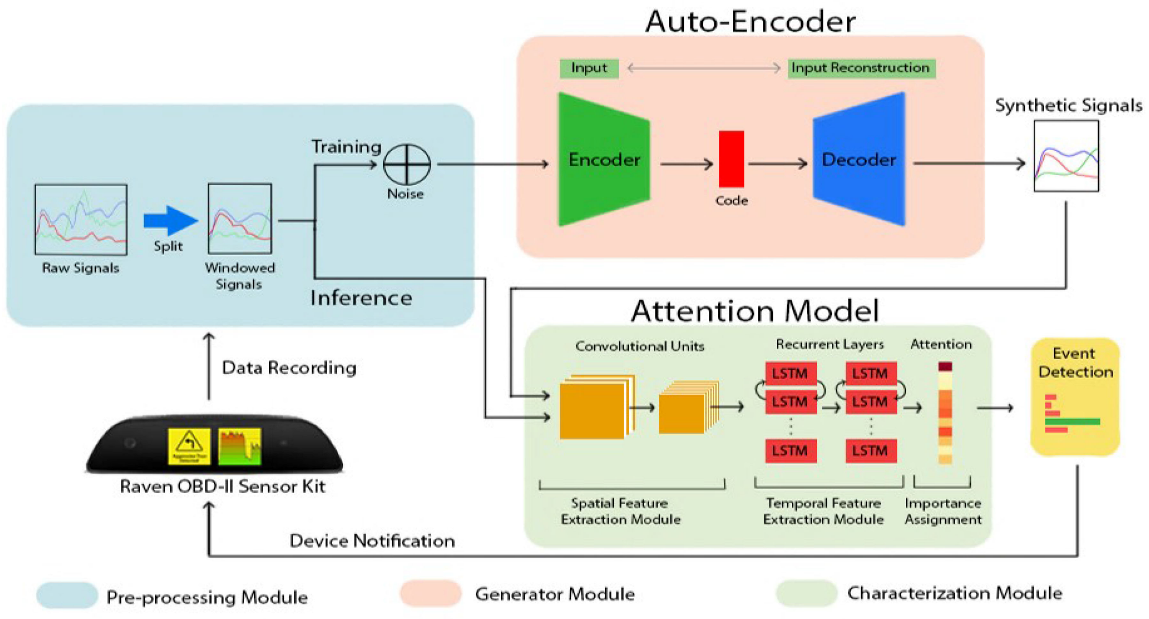


FIGURE 2. System pipeline overview. Signals are recorded by Raven sensors and sent to the generator module for training synthetic data reconstruction. Synthetic data is then used to train a self attention-based spatial-temporal encoding network for event classification. Generator module enhances the training data by denoising and augmenting the input data while the characterization module learns to put more emphasis on eventful sections of the signals.

detect risky driving events in real-time. This module consists of three individual networks to extract spatial, temporal, and attention encoding of the input signals, sequentially. Detected events form the output of this module which is sent to the device for safety notification and applications. The purpose of each module in the process flow along with their formulation are explained in detail in individual sections.

A. SYSTEM OVERVIEW

Our proposal for the driving event characterization builds on a three-stage model that employs recurrent and attention auto-encoder models to recognize various risky driving behavior in a signal. Fig. 2 illustrates an overview of the proposed process flow. Accelerometer signal along three orientation axes of the vehicles is captured and fed into the process flow as the input. In the pre-processing module, the data splits into n equally sized windows through a sliding window mechanism. Before training, the data windows are multiplied and fed into the denoising recurrent auto-encoder network with added random noise. The decoder network attempts to reconstruct the original signal, thus creating noise-less variations of the original input windows. Then the reconstructed training data are used to train an attention-based encoder network for event characterization. The characterization module uses $softmax$ output layer in order to classify the signals under multiple categories.

Inference can seamlessly be performed by passing the unseen signals to the characterization module. The more detailed explanation of each module can be seen in the following sections.

B. DRIVING EVENT DATASET GATHERING

The proposed system collects accelerometer sensor data as the input along x, y, and z axes of the vehicle over variable time spans. Raven OBD-II sensor kit [52] which is an android-based connected vehicle device is mounted on a car dashboard to record sensory data. Raven is equipped with an inertial measurement unit (IMU) sensor with the following orientation specifications: X-axis along the latitudinal axis of the vehicles, Y-axis along the altitude, and Z-axis along the longitudinal as shown in Fig. 1. The collected data is almost evenly distributed between five distinct driving behaviors, namely: Regular driving (RD), Harsh left lane change (LT), Harsh right lane change (RT), Harsh braking (HB), and aggressive acceleration (AA). It's worth noting that the data was recorded on a number of vehicles with different physical attributes and no filtering or signal processing methods were applied.

C. PRE-PROCESSING MODULE

Pre-processing module initiates the proposed pipeline. Multiple raw driving signals of various time length and three axes (i.e. X, Y, and Z) are fed into the module. Since the following modules are designed to take fixed-sized input, a sliding window mechanism splits the data into several event windows of size x. In order to keep the formulations consistent, a summary of the used parameters is presented in Table 1. Given a driving event signal of E with duration of t captured along axes $a = 3$ ($E_t^a \in \mathbb{R}^{t \times a}$), the windowing mechanism outputs several evenly split windowed signals $W_n \in \mathbb{R}^{a \times x}$, where W is the signal window, n is the number of created windows, a

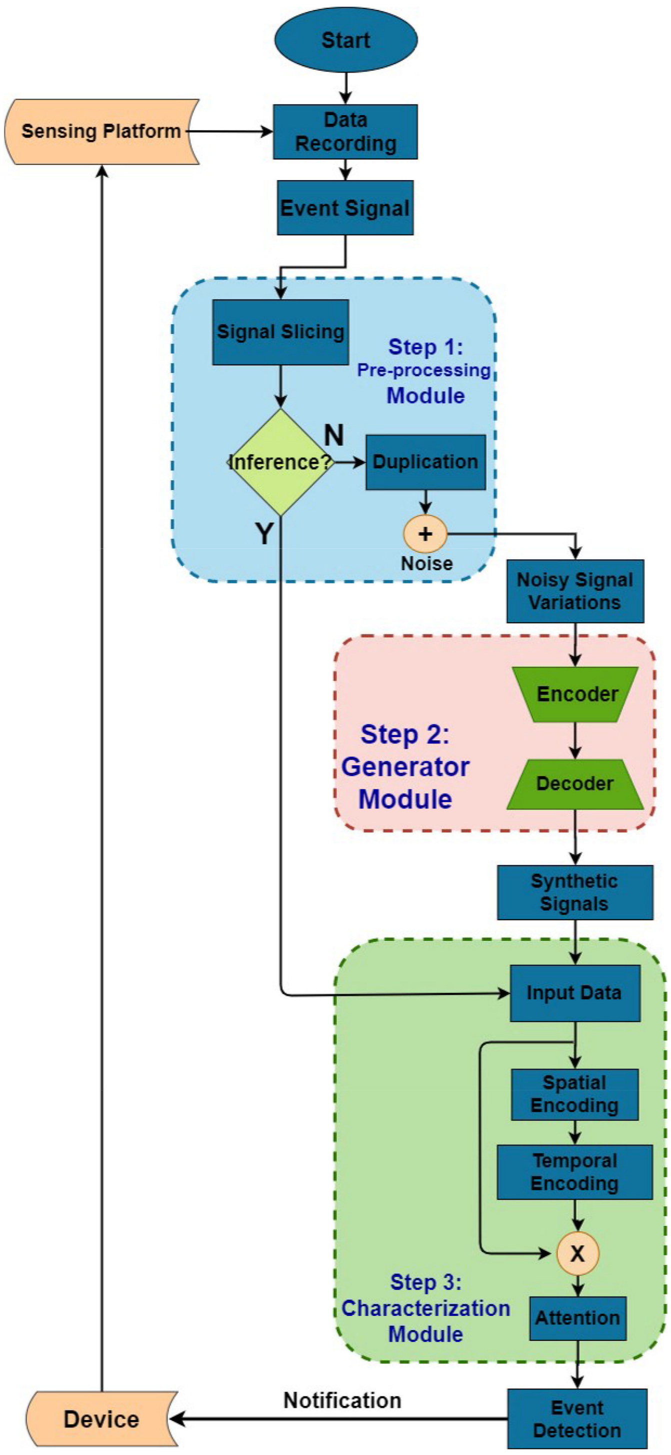


FIGURE 3. General flow diagram for training and inference process of the proposed solution through the three modules.

TABLE 1. Network Parameter Descriptions

Parameter	Description
x	Input window size
a	Input dimension
t	Raw signal duration
E	Event raw signal
W	Windowed Signals
n	Number of signal windows
d	Bottleneck dimension
h_e	Hidden states of the encoder
h_d	Hidden states of the decoder
S_n	Spatial encoded vector of input window n
r_n	Temporal encoded vector of input window n
v_i	Attention vector
y	Input labels
p	Predicted probability

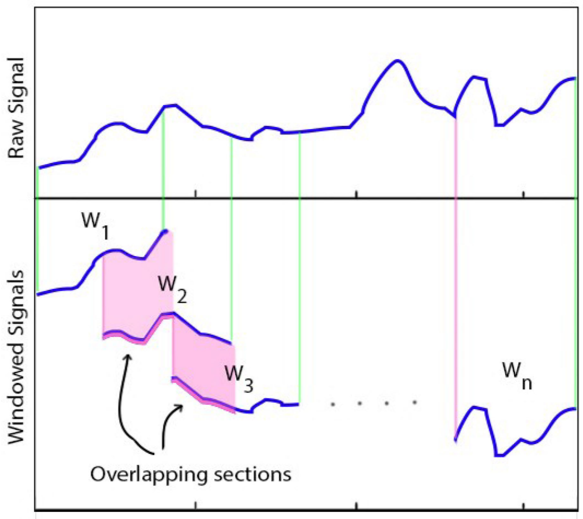


FIGURE 4. Single dimensional raw signal windowing with variable overlapping into *n* flexible sections. The signal is simplified for clearer depiction.

is the number of axes in the signal, and x denotes input size required by the auto-encoder network. To preserve signal correlations and provide the system with more signal variation, the sliding mechanism [53] splits the signal with overlap as demonstrated in Fig. 4. To perform experiments on the result of overlapping section of the data on the model performance, and due to the variable input data length, this module can split data into windows with different overlap values; thus, the overlap between the last two windows is flexible to avoid data loss. The aforementioned overlap value results are presented in the next section.

At the second stage, the pre-processing module identifies the module that the data is going to be streamed through. The training data is sent to the generator module while the inference signals are passed directly to the characterization module. Each training data window is duplicated into m varied training data windows with added random Gaussian noise. We then have $W_{nm} \in \mathbb{R}^{a \times x}$, obtained from each input signal window W_n. The data duplication is performed in order to overcome the issue of scarce risky driving data which often causes data imbalance for deep learning-based event characterization systems. Each copy of the signal is augmented with random noise to gain variations of an event signal. The noisy data windows are then fed into the generator module to initiate denoising before being classified in the characterization module. However, inference data skips this stage and is

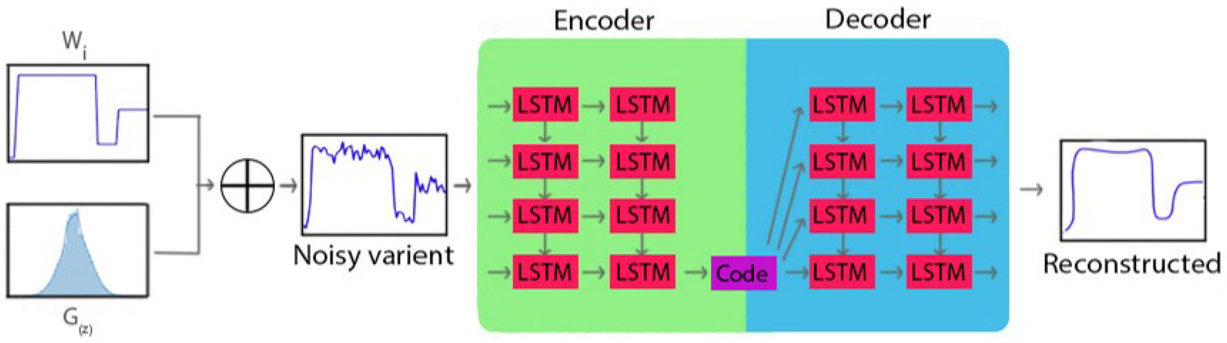


FIGURE 5. Generator module network architecture. To gain multiple variants of individual events, each event window is first fused with noise. The LSTM auto-encoder learns to reconstruct each noisy variant of the input signal. The network is also an effective noise filtering technique.

directly fed to the Characterization module to be identified to the corresponding categories.

D. GENERATOR MODULE

The generator module is responsible for enriching the training dataset. This subsection introduces the proposed deep neural network utilized in the generator module to not only learn the underlying features of the signals but to also denoise the noisy signals in a vehicular environment. The learned features of the signals by the auto-encoder network allows us to multi-fold the existing training dataset and introduce precise variations of event signals to the dataset. The gained improvements of the method is explained and illustrated in the following sections. The solution adopted here is a recurrent deep auto-encoder network which is suitable for extraction of event behavior in sequential data and is capable of reproducing feature-rich and noise-free data [54]. The objective of such network is to reconstruct synthetic data learned from the underlying features of input operating as a self-supervised process. The auto-encoder utilized in this module is a denoising auto-encoder which learns to derive the original input from the noisy input. Signals with added noise are fed as input to this type of auto-encoders and the network does not see the original signal. Therefore, the network is not able to predict the output without learning the under-lying features of the signals. Denoising auto-encoders take disrupted input and learn to identify its features. High dimensional data and massive size are often simpler to identify in lower dimensions. Therefore, the network maps the input onto a bottleneck of lower dimension which carries the input features.

The auto-encoder consists of three components: encoder, code, and decoder as illustrated in Fig. 5. The encoder maps the input onto a fixed-size context vector of lower dimensionality known as code or the bottleneck. The code is a reduced and compressed representation vector that contains the intrinsic characteristics of the input. The decoder network then pursues reconstruction of the original signal from the noisy signal only when the last node of the code is generated. This component works backwards and generates each data point in reverse order, i.e., from the last to the first [43]. Furthermore, auto-encoders can be implemented using variety of layer and neuron types. While convolutional kernels can be utilized to extract spatial features, recurrent layers that can be employed for temporal feature learning are constructed using feed-forward networks. Since accelerometer sensor data analysis depends on timely features, a recurrent auto-encoder network is chosen as a classifier in this study.

The encoder network of this module is designed with 2 recurrent hidden layers to capture the temporal characteristics of the signals. Though, the conventional recurrent networks lead to the vanishing gradient issue which causes the network to be unable to update its weights and biases properly during the back-propagation. To overcome the issue, this study implements the network with Long short-term memory (LSTM) [55] layers of decreasing sizes that map the input to the bottleneck of size $d = 100$ in the last hidden layer. The symmetrical recurrent decoder network uses the encoded bottleneck vector to reconstruct noise-less signals. Utilizing Adam optimization method [56], the network minimizes the mean squared error between each data-point of the hypothesis and the original signal at each batched train as shown in Eq. 1:

$$\sum_j \sum_i \left(x^{(j)} - y^{(j)}\right)^2 \qquad (1)$$

where $j \in \{1, 2, \ldots, t\}$ and $i \in \{1, 2, \ldots, a\}$ where a is the axis dimension of the input signal and t is the length of the input signal. This module utilizes the Mean Square Error (MSE) loss function as opposed to the cross-entropy loss since the span of sensor measurements exceed the range [0, 1]. Given a sequential input signal of single axis and variable length t: $X = \{x^1, x^2, \ldots, x^t\}$, the last encoder hidden layer h_{2e} learns of the intrinsic input features : $h_{2e} = \{h^1_{2e}, h^2_{2e}, \ldots, h^d_{2e}\}$ where d denotes bottleneck size which is set to 100 in our proposed model. The symmetrical decoding network starts to predict y^t upon generation of the last node in the bottleneck (i.e. h^d_{2e}). This process is performed by predicting from the last state towards the start (i.e. $\{y^t, y^{t-1}, \ldots, y^1\}$) considering the previous states in each prediction, i.e., $y^t = f(h^t_{2^{\sim}d}, h^{t-1}_{2^{\sim}d})$.

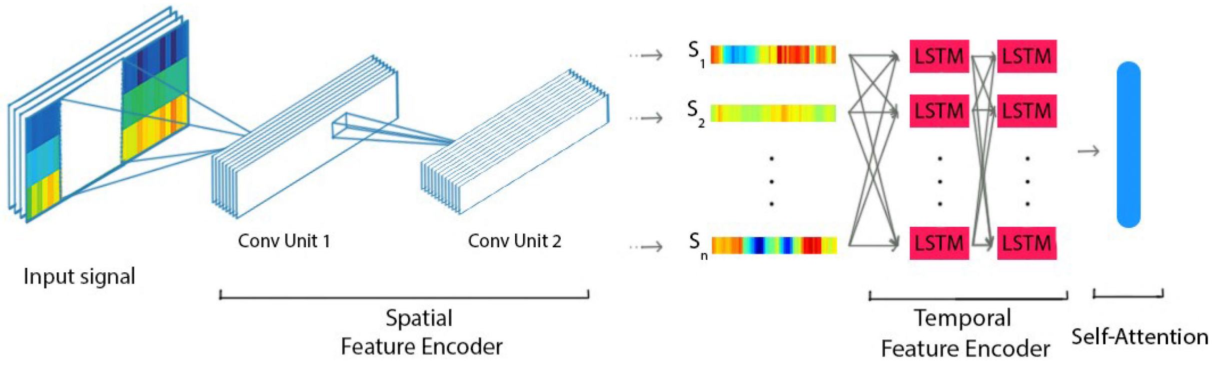


FIGURE 6. Network architecture for the characterization module.

The Generator module runs 10 times for each signal input. Each time with a noisy variation of the input to generate 10 synthetic signals that have the same average statistics with the input but with different values. The synthetic signal generation which share the same behavior features as the raw input signals grants the possibility to expose the characterization network to higher quantities of training data as well as training data with slight variation. This exposure results in a more robust training process which leads to the higher performance outcome.

It is worth to note that extensive experiments on several recurrent cell types, the number of the hidden layers, and various hyper-parameters of the network are carried out in order to choose the network details. The impact of all the details on the network performance are presented in the next section.

E. CHARACTERIZATION MODULE

The characterization module is used to categorize the unseen signals under the five aforementioned driving events and gets activated to train on the generator module outputs or directly operates to characterize the inference signals as shown in Fig. 6.

In the training process, the reconstructed data from the generator module is fed into the characterization module as the input. The module encodes the spatial features of the signals utilizing the custom-sized convolutional filters. Conventional convolutional layers are designed as square-shaped filters to output features along both directions in a 2D input which make them effective tools on images with observable objects. However, accelerometer signal $E_t^a \in \mathbb{R}^{t\times a}$ is unique two-dimensional data in a sense that the data only has immediate correlation and feature along the axis of time. The other axis consists of signals independent of each other with longer-term relations. Therefore, we implemented a specific convolutional kernel in order to capture short-term timely features as well as long-term relations between signals of different axes. The convolutional kernel is implemented with the height equal to number of signal axes a and with stretched kernel width. Doing so lets the convolutional kernel observe a wider body of the signals and therefore have longer-term information. Moreover, the operational nodes in the convolutional network are utilizing Leaky ReLU [57] activation functions which passes a small positive gradient when the unit has a value of zero as Eq 2:

$$f(x = wx + b) = \begin{cases} x & x > 0 \\ 0.01x & \text{otherwise} \end{cases} \tag{2}$$

where x is a function of weights and biases of convolutional layer. Using Leaky ReLU allows us to avoid the problem of "dead ReLU" which restricts the network learning from nodes with a derivative of zero. In order to downsize the extracted features by the convolutional network and store more prominent characteristics, an average pooling layer is applied to the convolutional network output as illustrated in Fig. 9.

The second unit of the networks is an attention-based recurrent network to encode temporal dynamics of the signals. The downsized features $S_n = AvgPool(f(x))$ are fed to the Long Short-Term Memory layers. There are two LSTM layers with cell size of equal to the number of windowed input of the network. At any moment, the recurrent cells utilize the memory of the previous stages, therefore preserving the temporal history of the signals.

The utilized driving event dataset consists of periods of regular driving along with sudden risky event signals. In order to lower the false negative predictions, emphasizing the distinct periods that contain the events is necessary. The recurrent network output $r_n = LSTM(S_n)$ is fed into the attention layer for importance assignment to the each input data section. The hidden state of the second LSTM layer feeds the input to the attention layer as the last step before classification. The main purpose of this layer is to weigh out the false detections caused by the dynamics of certain sections in a signal where there is no significant movement information.

As illustrated in Fig. 7, each encoded signal section is mapped to a latent space utilizing the following function in Eq. 3 where $W_i \in \mathbb{R}^{l\times h_a}$ and $b_i \in \mathbb{R}^{h_a}$ are weight and bias matrix of the direct input to the hidden layer of size h_a.

$$H_i = tanh(W_i h_i' + b_i) \tag{3}$$

To learn the importance of each section of the signal, the nonlinear representation H_i is fed to a softmax activation

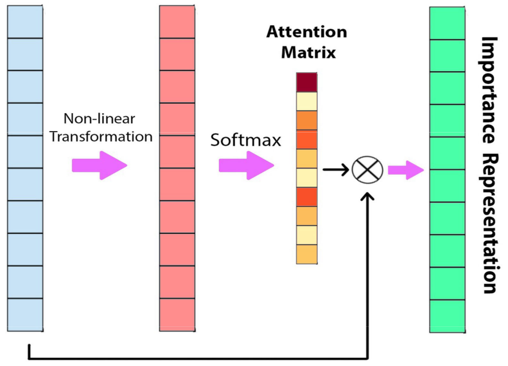


FIGURE 7. Self-attention layer. The encoded inputs are transformed using a non-linear function and then normalized into attention weight matrix of each signal section. Encoded inputs are then multiplied with the attention weights in order to output weighted input representation.

function which is formulated in Eq. 4 where v_i denotes to attention vector of each training iteration. The attention vector is learned during the training, and is then connected to a softmax layer for the purpose of final signal classification.

$$V_i = \frac{\exp(H_i^T v_i)}{\sum_i exp(H_i^T v_i)} \tag{4}$$

The classification layer aims to minimize the cross-entropy error of all labeled sections of the signals, calculated through Eq. 5 where p is the network predictions which is a function of network input and y denotes to the target label of the signals.

$$CE = -\sum_c y \, \log(p_{(x_n)}) \tag{5}$$

The characterization module also utilizes the Adam optimizer function. Adam optimizer is a decaying momentum optimizing method modeled on physical friction [56]. It updates parameters through Eq. 6, and stores exponentially decaying average of past squared gradients ($v_t = \beta_2 v_{t-1} + (1-\beta_2)g_t^2$) and an exponentially decaying average of past gradients ($m_t = \beta_1 m_{t-1} + (1-\beta_1)g_t$).

$$\Theta_{t+1} = \Theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\hat{m}_t \tag{6}$$

Adam optimizer has a tendency of 0 bias during the initial steps. To fix the bias issue in the gradients, corrected estimates of initial moments are calculated as shown in Equations 7 and 8.

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{7}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{8}$$

IV. PERFORMANCE EVALUATION

In this section, we describe the setting values used in the data gathering as well as the training process of the proposed model in Sections A and B. The detailed experiments regarding the internal structure of the system is provided in Section C. Last but not least, a comparison of the final result of the pipeline in comparison with state-of-the-art models is presented in Section D.

TABLE 2. Dataset Details

Event Type	Event Count	Window Count
Harsh brake	13	104
Aggressive acceleration	12	108
Harsh left turn	16	126
Harsh right turn	15	121
Regular driving	14	113

A. DATASET AND COLLECTION SETTINGS

To collect proper sensor data for the study, accelerometer sensor data along three axes of vehicle direction is chosen to record four risky driving events, namely aggressive acceleration, harsh braking, harsh left and right lane change as well as various non-risky driving sessions. Using the Raven OBD-II sensor kit, a total of 70 sessions were recorded from several vehicles. The sessions are roughly distributed among all five categories evenly and are of various durations from 2 to 3.6 seconds. The IMU sensor sampling frequency is set to 25 Hz. In total, 16 harsh left turn, 15 harsh right turn, 13 harsh braking, 12 aggressive acceleration, and 14 event-less sessions are captured and sliced into 572 windows of 600 milliseconds using 50% overlap setting between two sequential windows. A more detailed distribution of the dataset can be found in Table 2.

Last but not least, randomly selected 70% of the data sessions were split as training and the rest as testing set. To keep the comparisons fair, the same train and test set were utilized in all our experiments.

B. EXPERIMENTAL SETTINGS

Initially, ten random Gaussian noise of mean zero and standard deviation of 0.1 by probability distribution are added to each window of data to obtain noisy variations of the input as shown in Eq. 9

$$p_G(W_i) = \frac{1}{0.1\sqrt{2\Pi}} e^{-\frac{w_i^2}{2(0.1)^2}} \tag{9}$$

The noisy data is then fed into the generator module which is a three-layer stacked denoising auto-encoder network with layer sizes of 600, 300, 100, respectively. The last encoder layer is used as the vector bottleneck layer. Google's Tensorflow framework was employed to deploy the auto-encoder networks. The decoding network is symmetrical to the encoder i.e. the decoder is implemented with layers of 100, 300, and 600 nodes, respectively. The layers consist of LSTM cells in order to preserve longer-term memory of the data.

The generator network optimizes its error, calculated by Mean Squared Error (MSE), utilizing Adam optimizer with decay values of $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$ on each iteration. A learning rate of 0.03 is chosen for network parameter optimization on each training epoch.

TABLE 3. Signal Overlap Test Results

Overlap Value	Train Accuracy	Test Accuracy	Precision	Recall	F1-Score
no-overlap	0.9114 ±0.0108	0.8953 ±0.0127	0.7368 ±0.0131	0.7778 ±0.0129	0.7568 ±0.0134
20%	0.9347 ±0.0140	0.9244 ±0.0153	0.8286 ±0.0138	0.8056 ±0.0136	0.8169 ±0.0137
30%	0.9444 ±0.0128	0.9419 ±0.0131	0.8824 ±0.0129	0.8333 ±0.0126	0.8571 ±0.0125
40%	0.9562 ±0.0117	0.9521 ±0.0106	0.8889 ±0.0108	0.8889 ±0.0131	0.8889 ±0.0136
50%	**0.9704 ±0.0073**	**0.9651 ±0.0089**	**0.9429 ±0.0085**	**0.8919 ±0.0098**	**0.9167 ±0.0096**
60%	0.9859 ±0.0063	0.9593 ±0.0085	0.9167 ±0.0065	0.8919 ±0.0071	0.9041 ±0.0070
70%	0.9837 ±0.0077	0.9588 ±0.0108	0.8824 ±0.0109	0.9091 ±0.0103	0.9055 ±0.0101

The auto-encoder network is trained for 1000 iterations before stopping and the output signals are kept as training dataset to the characterization encoding network.

The characterization module trains on the reconstructed dataset for 1000 iterations. It optimizes the cross-entropy error using Adam optimizer with decay values of $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\epsilon = 10^{-8}$ and the learning rate of 0.03 over the iterations. Dropout layers of probability 0.6 are added to avoid over-training the network. The spatial feature extraction units are operating with custom-sized convolution kernels of size 3×15 and filter depth of 64. The output of the spatial feature extraction unit is a linear code of size $1 \times (n - 14) \times 64$, where n is time length of each signal window. We select no-padding in the convolutional steps. The pooling layers are averaging 30 nodes i.e. filter size of 1×30 on each step with stride value of 10. Temporal feature extraction network is made of LSTM layers of 128 nodes, feeding the encoded information into a self-attention module of size 256. Lastly, a softmax layer is set as the output layer to classify the features into five event classes.

C. INTERNAL STRUCTURE INVESTIGATION

To gain knowledge of how the structure and hyper-parameters of the networks affect the performance of the pipeline, extensive experiments ware conducted. The experiments are done in a controlled manner to isolate the effect of the subject parameter from other parameters. Moreover, all the experiments are performed 10 times and the average results are presented with 95% confidence levels. In this section, first the effects of the pre-processing and generator networks are presented and then, we explore the internal structure effects of the characterization network.

As the first experiment, the effect of the overlap percentage in the signal windowing mechanism is investigated. Introducing data overlap significantly improves the overall network performance. The performance boost grows as the overlapping section expands, though it reaches a point of diminishing return after 50% and values of over that number do not show performance benefits to the process flow (Table 3).

To study the generator module in detail, the number of symmetrical and asymmetrical [58] LSTM auto-encoders are deployed and tested on the dataset (Table 4). We conclude that symmetrical designs are more consistent than and outperform the auto-encoders of asymmetrical shape. Moreover, while the model performance improves with the number of stacked hidden layers, networks with more than three stacked layers do not show any signs of improvement. As a result of this experiment, a symmetrical network of 3 encoder and 3 decoder layers is chosen for further network investigations.

To test out the effectiveness of several recurrent and non-recurrent neurons, a diverse range of state-of-the-art neurons are selected and the final classification results of the process flow are recorded as presented in Table 5. The most superior performance is obtained under the LSTM and Bi-LSTM nodes. LSTMs are chosen since they reduce computation complexity of the network compared with Bi-LSTMs, thus reducing the training time.

Lastly, dimensionality of the signals are reduced to 2D using Principal Component Analysis (PCA) method. PCA has shown to perform poorly when the input data is noisy. To study the usefulness of the generator module in denoising of signals, the reconstructed signals exhibit more separation of values, compared with original signals, in two dimensions which can be described as better understanding of the underlying features in the signals by the network. The 2D representation of original and reconstructed signals are shown in Fig. 8.

The first experiment regarding to the characterization module layer size is presented in Table 6. The characterization encoding network is implemented with one to three spatial feature extraction units as demonstrated in Fig. 9 (i.e. a combination of convolutional and pooling layers as a unit). Stacking two spatial feature extraction units is shown to be the most effective. With two convolutional units as default, the number of needed recurrent layers is studied. The experimental results show that stacking two recurrent layers slightly enhances the classification outcome whereas three layers have virtually no positive effect on the performance. Moreover, for practicality of the attention mechanism in the tests, the module is first trained with no attention layer, and then with an attention layer of different sizes. As reported in the table, the introduction of the attention mechanism has the most significant impact on the performance, though high layer size is shown to lead to the diminishing return of the performance.

As mentioned, a custom-sized convolutional filter is employed that is suitable for vehicular sensor data in the characterization module. The experiments in Table 7 are performed on the filter size in order to find the best fit kernel for the vehicular sensory signals. Additionally, the impact of the convolutional padding and stride on the system performance are investigated. Numerical results concerning the impact of the

TABLE 4. Generator Network Depth Test Results

Shape	Layers	Train Accuracy	Test Accuracy	Precision	Recall	F1-Score
Symmetrical	2-2	0.9005 ±0.0196	0.8659 ±0.0217	0.7347 ±0.0164	0.7660 ±0.0172	0.7500 ±0.0170
	3-3	**0.9765 ±0.0073**	**0.9607 ±0.0087**	**0.9403 ±0.0089**	**0.8892 ±0.0106**	**0.9113 ±0.0101**
	4-4	0.9776 ±0.0091	0.9592 ±0.0103	0.9384 ±0.0109	0.8921 ±0.0118	0.9147 ±0.0116
	5-5	0.9604 ±0.0111	0.9603 ±0.0086	0.9224 ±0.0091	0.9008 ±0.0088	0.9114 ±0.0089
Asymmetrical	3-2	0.8654 ±0.0181	0.7733 ±0.0170	0.4884 ±0.0196	0.5526 ±0.0169	0.5185 ±0.0173
	3-4	0.9371 ±0.0139	0.8895 ±0.0140	0.7500 ±0.0138	0.7692 ±0.0148	0.7595 ±0.0141
	3-5	0.9449 ±0.0093	0.9075 ±0.0117	0.8049 ±0.0142	0.8049 ±0.0123	0.8049 ±0.0136

TABLE 5. Generator Network Node Test Results

Node Type	Train Accuracy	Test Accuracy	Precision	Recall	F1-Score
ReLU	0.7617 ±0.0164	0.7368 ±0.0183	0.4808 ±0.0149	0.5208 ±0.0181	0.5000 ±0.0173
LSTM	**0.9711 ±0.0087**	**0.9611 ±0.0107**	**0.9415 ±0.0088**	**0.8924 ±0.0106**	**0.9162 ±0.0095**
BiLSTM	0.9703 ±0.0081	0.9592 ±0.0102	0.8734 ±0.0116	0.8828 ±0.0109	0.8780 ±0.0112
GRU	0.9667 ±0.0099	0.9509 ±0.0103	0.8804 ±0.0117	0.8734 ±0.0121	0.8768 ±0.0120

TABLE 6. Characterization Network Module Test Results

Tested	Dimension	Train Accuracy	Test Accuracy	Precision	Recall	F1-Score
Conv	1	0.9549 ±0.0086	0.9477 ±0.0119	0.8824 ±0.0106	0.8571 ±0.0119	0.8696 ±0.0112
	2	**0.9741 ±0.0065**	**0.9593 ±0.0086**	**0.9394 ±0.0068**	**0.8611 ±0.0071**	**0.8986 ±0.0069**
	3	0.9703 ±0.0075	0.9535 ±0.0083	0.9257 ±0.0084	0.8857 ±0.0077	0.9052 ±0.0081
LSTM	1	0.9471 ±0.0087	0.9128 ±0.0109	0.8000 ±0.0126	0.7778 ±0.0128	0.7887 ±0.0127
	2	**0.9760±0.0078**	**0.9708 ±0.0084**	**0.9394 ±0.0093**	**0.9118 ±0.0091**	**0.9254 ±0.0090**
	3	0.9732 ±0.0081	0.9649 ±0.0092	0.9116 ±0.0101	0.9121 ±0.0085	0.9118 ±0.0093
Attention	no-attention	0.9214 ±0.0116	0.8837 ±0.0151	0.7273 ±0.0117	0.6857 ±0.0138	0.7059 ±0.0136
	128	0.9624 ±0.0081	0.9591 ±0.0084	0.9091 ±0.0067	0.8824 ±0.0078	0.8955 ±0.0073
	256	**0.9706 ±0.0066**	**0.9649 ±0.0084**	**0.9091 ±0.0074**	**0.9091 ±0.0073**	**0.9091 ±0.0074**
	512	0.9697 ±0.0061	0.9617 ±0.0067	0.9017 ±0.0065	0.8913 ±0.0061	0.8964 ±0.0063

TABLE 7. Convolution Module Setting Test Results

Test	Setting	Train Accuracy	Test Accuracy	Precision	Recall	F1-Score
Conv Kernel Size	3x5	0.9592 ±0.0093	0.9379 ±0.0091	0.8732 ±0.0093	0.8591 ±0.0085	0.8660 ±0.0086
	3x10	0.9647 ±0.0086	0.9593 ±0.0084	0.9028 ±0.0072	0.8920 ±0.0064	0.8973 ±0.0066
	3x15	**0.9715 ±0.0102**	**0.9610 ±0.0081**	**0.9048 ±0.0080**	**0.8706 ±0.0068**	**0.8873 ±0.0072**
	3x20	0.9706 ±0.0098	0.9593 ±0.0085	0.9118 ±0.0081	0.8857 ±0.0079	0.8986 ±0.0080
	3x25	0.9712 ±0.0088	0.9603 ±0.0084	0.9103 ±0.0079	0.8920 ±0.0075	0.9010 ±0.0077
Pooling Kernel Size	10	0.9730 ±0.0081	0.9610 ±0.0089	0.9115 ±0.0078	0.8903 ±0.0066	0.9007 ±0.0071
	20	0.9711 ±0.0086	0.9609 ±0.0093	0.9089 ±0.0084	0.8932 ±0.0086	0.9009 ±0.0085
	30	**0.9735 ±0.0078**	**0.9611 ±0.0103**	**0.9048 ±0.0075**	**0.8706 ±0.0071**	**0.8873 ±0.0072**
	40	0.9553 ±0.0102	0.9419 ±0.0094	0.8788 ±0.0105	0.8286 ±0.0113	0.8529 ±0.0109
	50	0.9244 ±0.0124	0.9070 ±0.0133	0.7941 ±0.0119	0.7500 ±0.0141	0.7714 ±0.0138
Conv Padding	**No-Pad**	**0.9711 ±0.0092**	**0.9564 ±0.0103**	**0.8926 ±0.0085**	**0.8742 ±0.0092**	**0.8833 ±0.0088**
	0-Pad	0.9703 ±0.0083	0.9468 ±0.0108	0.8874 ±0.0085	0.8617 ±0.0081	0.8743 ±0.0083
	Same-Pad	0.9694 ±0.0085	0.9573 ±0.0093	0.8894 ±0.0081	0.8723 ±0.0083	0.8807 ±0.0082
Stride	5	0.9701 ±0.0079	0.9588 ±0.0083	0.9010 ±0.0071	0.8988 ±0.0068	0.8998 ±0.0070
	10	**0.9721 ±0.0076**	**0.9593 ±0.0087**	**0.9384 ±0.0091**	**0.8921 ±0.0102**	**0.9147 ±0.0098**
	15	0.9719 ±0.0088	0.9542 ±0.0101	0.9273 ±0.0083	0.8968 ±0.0097	0.9117 ±0.0090
	20	0.9637 ±0.0083	0.9477 ±0.0081	0.8857 ±0.0075	0.8611 ±0.0085	0.8732 ±0.0081

activation functions in the convolution operations are reported in Table 8. The same experiments are repeated on the recurrent neurons in the generator module. Neurons are swapped in the temporal feature extraction unit of the characterization module in order to record the performance of the system with each neuron type. The top performing settings are shown in bold in the table.

D. NUMERICAL RESULTS

Ablation studies are carried out to examine the individual component's effectiveness on the models performance. Furthermore, several state-of-the-art as well as baseline models are tested and compared to the proposed model so to demonstrate the superiority of the model in terms of classification accuracy and lower false positive performance.

The effect of each module in the pipeline is studied with leave-one-out approach. First, the generator module is disabled and the characterization module is trained and tested on the original dataset to explore the effect the reconstructed signals have on the system performance. Then, spatial feature extraction, temporal feature extraction, and attention modules of the characterization network are abandoned in individual

TABLE 8. Module Activation Function Test Results

Module	Nodes	Train Accuracy	Test Accuracy	Precision	Recall	F1-Score
Convolutional	ReLU	0.9688 ±0.0093	0.9535 ±0.0105	0.8824 ±0.0086	0.8824 ±0.0077	0.8824 ±0.0081
	tanh	0.9664 ±0.0103	0.9477 ±0.0091	0.8857 ±0.0099	0.8611 ±0.0081	0.8732 ±0.0090
	LReLU	**0.9721 ±0.0096**	**0.9590 ±0.0084**	**0.9091 ±0.0091**	**0.8824 ±0.0090**	**0.8955 ±0.0091**
Recurrent	**LSTM**	**0.9703 ±0.0085**	**0.9593 ±0.0098**	**0.9118 ±0.0093**	**0.8857 ±0.0079**	**0.8986 ±0.0085**
	BiLSTM	0.9682 ±0.0087	0.9532 ±0.0095	0.9090 ±0.0079	0.8841 ±0.0086	0.8963 ±0.0082
	GRU	0.9702 ±0.0094	0.9438 ±0.0103	0.8824 ±0.0098	0.8333 ±0.0097	0.8571 ±0.0093

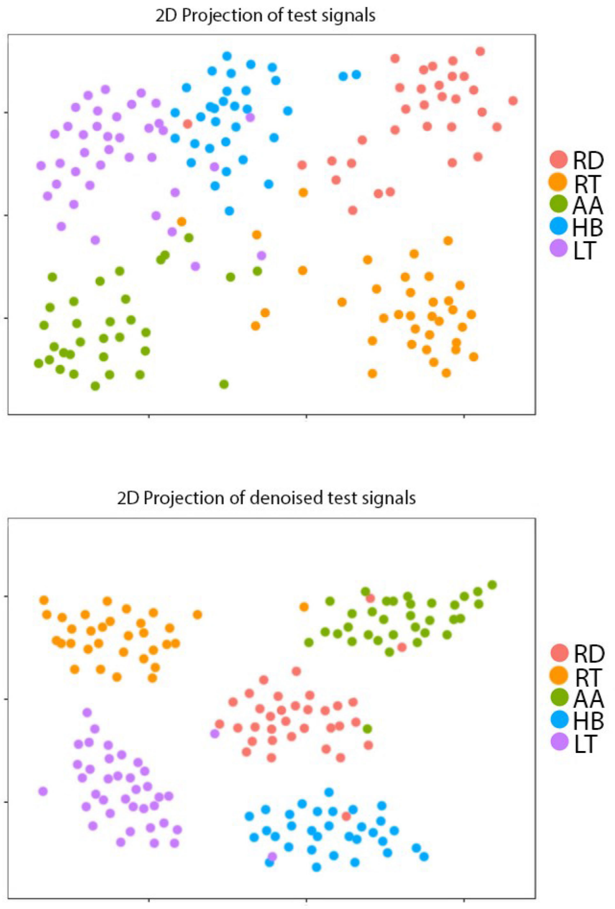


FIGURE 8. Signal 2D projections. The utilization of the generator module acts as a noise filter which causes more distinction between the encoded signals. Data-points are magnified for visualization purposes.

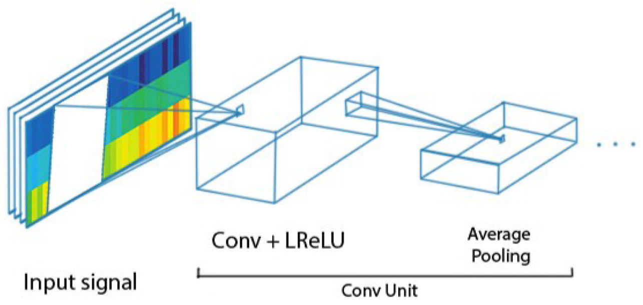
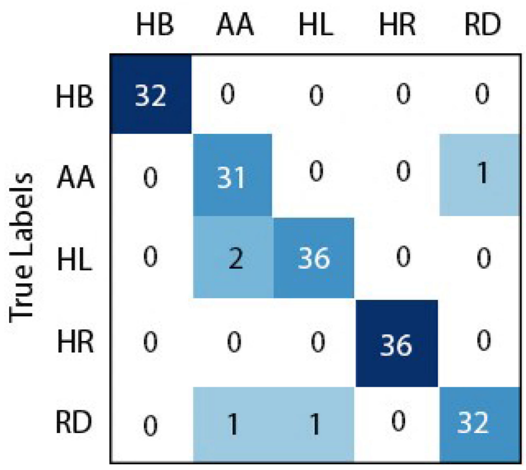


FIGURE 9. Demonstration of convolutional units. Each unit consists of custom size kernels operating with Leaky ReLU functions and an average pooling layer.



	HB	AA	HL	HR	RD
HB	32	0	0	0	0
AA	0	31	0	0	1
HL	0	2	36	0	0
HR	0	0	0	36	0
RD	0	1	1	0	32

FIGURE 10. Breakdown of the inference of the test-set using the proposed pipeline. The main diagonal axis represents the correctly characterized events.

experiments. It is worth noting that the structure and settings of all the components are chosen from the internal structure investigation reported in the previous section. The ablation study results are stated in Table 9, in which the generator module is referred to as “G,” the spatial and the temporal feature extraction modules of the characterization module as “S” and “T,” respectively. Last but not least, the attention module is referred to as “A.”

Moreover, state-of-the-art models are trained and tested on our dataset in order to illustrate a fair comparison of the models. As the baseline to our research, a modified recurrent LSTM driving event detection model [59] is implemented and optimized in our previous work [10]. The baseline model employs convolutional neural networks as well as recurrent layers as feature extraction methods and trains a fully connected neural network. The classification network of the baseline model attempts to output the probability distribution of the event categories using a softmax layer as the output. This model achieves the test accuracy of above 77% with an F1-score of 0.53 on the dataset. The “Regular Driving” event category of signals are the events that the baseline model detects best with 83% success rate while facing the worst detection rate of with 74% in “Harsh Left Turn” category. However, the success rate of the proposed model is provided in Fig. 10. The system is able to classify the “Harsh Left Turn” signals with only 2 miss-classified signals.

Additionally, the adopted spatio-temporal classification method [60] that utilizes standard VGG-16 CNN architecture

TABLE 9. Module Importance Test Results

Network	Train Accuracy	Test Accuracy	Precision	Recall	F1-Score
Spatial	0.7849 ±0.0144	0.71820 ±0.0182	0.5000 ±0.0140	0.5405 ±0.0136	0.5195 ±0.0138
Temporal	0.7965 ±0.0178	0.7019 ±0.0156	0.5610 ±0.0167	0.5750 ±0.0170	0.5679 ±0.0169
ST	0.8372 ±0.0164	0.7857 ±0.0132	0.5946 ±0.0158	0.6286 ±0.0155	0.6111 ±0.0157
STA	0.9241 ±0.0124	0.9070 ±0.0130	0.7714 ±0.0119	0.7714 ±0.0109	0.7714 ±0.0114
GST	0.8814 ±0.0116	0.8663 ±0.0133	0.6857 ±0.0117	0.6667 ±0.0146	0.6761 ±0.0131
baseline	0.8517 ±0.0115	0.7753 ±0.0135	0.5476 ±0.0151	0.5227 ±0.0143	0.5349 ±0.0147
sparse AE	0.9509 ±0.0101	0.9296 ±0.0117	0.8491 ±0.0088	0.8824 ±0.0096	0.8654 ±0.0092
spatio-temporal	0.8953 ±0.0128	0.8142 ±0.0139	0.6341 ±0.0150	0.6047 ±0.0145	0.6190 ±0.0147
GSTA	**0.9774 ±0.0073**	**0.9608 ±0.0082**	**0.9275 ±0.0079**	**0.9071 ±0.0093**	**0.9171 ±0.0085**

for spatial feature extraction is able to reach 81% in classification accuracy and F1-score of 0.62. This model employs the pooling techniques to gain temporal information across space and time from the data and is end-to-end classification trainable using a classification loss. Furthermore, the adopted sparse auto-encoder classifier network [61] classifies the signals at 93% accuracy and results in 0.86 F1-score. Last but not least, our proposed system reaches the accuracy of 96% and attains 0.91 F1-score improving 15% upon the baseline.

Lastly, since lowering the false positive predictions is a major focus of ours, a prediction distribution of the test-set, classified by our system is specified in Fig. 10 to provide more insight into the efficiency of the model.

V. CONCLUSION

This article targets the rarely studied problem of characterizing the risky driving events in a connected vehicle setting. To this end, the article has proposed a novel neural network based process flow to not only advance the prediction scores of the existing models, but to also address the data scarcity problem in event characterization. The proposed model splits the driving event signals, learns the underlying behavioral features, and de-noises them for the training process. In training, the model extracts spatio-temporal features of the reconstructed signals, leverages the attention model-based neural network, and achieves more than 96% test accuracy which roughly translates to 15% improvement over the baseline. This article has also presented detailed experiments with regards to the underlying parameters of the networks and their effect on the task performance.

Extensions of this study and ongoing work include investigation of the possibility of augmenting the proposed model with knowledge-based approaches so to further improve the prediction performance of the proposed process flow in event characterization from vehicular sensor data.

REFERENCES

[1] M. Shams and V. Rahimi-Movaghar, “Risky driving behaviors in Tehran, Iran,” *Traffic Injury Prevention*, vol. 10, no. 1, pp. 91–94, 2009.

[2] C. Campolo, A. Molinaro, A. Iera, and F. Menichella, “5G network slicing for Vehicle-to-Everything services,” *IEEE Wireless Commun.*, vol. 24, no. 6, pp. 38–45, Dec. 2017.

[3] N. Taherkhani and S. Pierre, “Centralized and localized data congestion control strategy for vehicular ad hoc networks using a machine learning clustering algorithm,” *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 11, pp. 3275–3285, Nov. 2016.

[4] S. P. Biswas, P. Roy, N. Patra, A. Mukherjee, and N. Dey, “Intelligent traffic monitoring system,” in *Proc. 2nd Int. Conf. Comput. Commun. Technol.*, 2016, pp. 535–545.

[5] J.-G. Kim, J.-H. Yoo, and J.-C. Koo, “Road and lane detection using stereo camera,” in *Proc. IEEE Int. Conf. Big Data Smart Comput.*, 2018, pp. 649–652.

[6] S. Arumugam and R. Bhargavi, “A survey on driving behavior analysis in usage based insurance using big data,” *J. Big Data*, vol. 6, no. 1, pp. 86–106, 2019.

[7] T. Phiboonbanakit, V.-N. Huynh, T. Horanont, and T. Supnithi, “Detecting abnormal behavior in the transportation planning using long short term memories and a contextualized dynamic threshold,” in *Proc. Adjunct. ACM Int. Joint Conf. Pervasive Ubiquitous Comput. ACM Int. Symp. Wearable Comput.*, 2019, pp. 996–1007.

[8] L. R. Medsker and L. Jain, “Recurrent neural networks,” *Des. Appl.*, vol. 5, pp. 64–67, 2001.

[9] J. G. A. Barbedo, “Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification,” *Comput. Electron. Agriculture*, vol. 153, pp. 46–53, 2018.

[10] N. Taherifard, M. Simsek, C. Lascalles, and B. Kantarci, “Machine learning-driven event characterization under scarce vehicular sensing data,” in *Proc. IEEE Int. Workshop Comput. Aided Model. Des. Commun. Links Netw.*, 2020, pp. 1–6.

[11] H.-B. Kang, “Various approaches for driver and driving behavior monitoring: A review,” in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, 2013, pp. 616–623.

[12] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, “Deep learning for time series classification: A review,” *Data Mining Knowl. Discovery*, vol. 33, no. 4, pp. 917–963, 2019.

[13] T. Pholprasit, W. Choochaiwattana, and C. Saiprasert, “A comparison of driving behaviour prediction algorithm using multi-sensory data on a smartphone,” in *Proc. IEEE/ACIS 16th Int. Conf. Softw. Eng., Artif. Intell., Netw. Parallel/Distributed Comput.*, 2015, pp. 1–6.

[14] A. S. Zeeman and M. J. Booysen, “Combining speed and acceleration to detect reckless driving in the informal public transport industry,” in *Proc. Int. IEEE Conf. Intel. Transp. Syst.*, 2013, pp. 756–761.

[15] L. Eboli, G. Mazzulla, and G. Pungillo, “Combining speed and acceleration to define car users safe or unsafe driving behaviour,” *Transp. Res. Part C: Emerg. Technol.*, vol. 68, pp. 113–125, 2016.

[16] R. Sun, W. Y. Ochieng, and S. Feng, “An integrated solution for lane level irregular driving detection on highways,” *Transp. Res. Part C: Emerg. Technol.*, vol. 56, pp. 61–79, 2015.

[17] I. Han and K. Yang, “Characteristic analysis for cognition of dangerous driving using automobile black boxes,” *Int. J. Automot. Technol.*, vol. 10, no. 5, pp. 597–605, 2009.

[18] J. Hu, L. Xu, X. He, and W. Meng, “Abnormal driving detection based on normalized driving behavior,” *IEEE Trans. Veh. Technol.*, vol. 66, no. 8, pp. 6645–6652, Aug. 2017.

[19] R. Chhabra, S. Verma, and C. R. Krishna, “Detecting aggressive driving behavior using mobile smartphone,” in *Proc. Int. Conf. Commun., Comput. Netw.*, 2019, pp. 513–521.

[20] C. Ma, X. Dai, J. Zhu, N. Liu, H. Sun, and M. Liu, “Drivingsense: Dangerous driving behavior identification based on smartphone autocalibration,” *Mobile Inform. Syst.*, vol. 2017, pp. 64–67, 2017.

[21] Y. Hu, M. Lu, and X. Lu, “Driving behaviour recognition from still images by using multi-stream fusion CNN,” *Mach. Vis. Appl.*, vol. 30, no. 5, pp. 851–865, 2019.

[22] C. Kaptan, B. Kantarci, T. Soyata, and A. Boukerche, "Emulating smart city sensors using soft sensing and machine intelligence: A case study in public transportation," in *Proc. IEEE Int. Conf. Commun.*, 2018, pp. 1–7.

[23] C. Kaptan, B. Kantarci, and A. Boukerche, "Reliability-driven vehicular crowd-sensing: A case study for localization in public transportation," in *Proc. IEEE Global Commun. Conf.*, 2018, pp. 1–7.

[24] P. Vavouranakis, S. Panagiotakis, G. Mastorakis, C. X. Mavromoustakis, and J. M. Batalla, "Recognizing driving behaviour using smartphones," in *Proc. Beyond Internet Things*, 2017, pp. 269–299.

[25] S. M. Mousavi, "Identifying high crash risk roadways through jerk-cluster analysis," Ph.D. dissertation, Louisiana State Univ., Baton Rouge, LA, USA, 2015.

[26] H. M. Ali and Z. S. Alwan, *Car Accident Detection Notification System Using Smartphone*. Saarbrucken, Germany: LAP LAMBERT Academic, 2017.

[27] J. White, C. Thompson, H. Turner, B. Dougherty, and D. C. Schmidt, "Wreckwatch: Automatic traffic accident detection and notification with smartphones," *Mobile Netw. Appl.*, vol. 16/3, pp. 285–303, 2011.

[28] M. Soyturk, K. Muhammad, M. Avcil, B. Kantarci, and J. Matthews, "Chapter 8 - from vehicular networks to vehicular clouds in smart cities," in *Proc. Smart Cities Homes*, M. S. Obaidat and P. Nicopolitidis, Eds. Boston, MA, USA: Morgan Kaufmann, 2016, pp. 149–171.

[29] S. Ramos, S. Gehrig, P. Pinggera, U. Franke, and C. Rother, "Detecting unexpected obstacles for self-driving cars: Fusing deep learning and geometric modeling," in *Proc. IEEE Intell. Veh. Symp.*, 2017, pp. 1025–1032.

[30] L. Li, K. Ota, and M. Dong, "Humanlike driving: Empirical decision-making system for autonomous vehicles," *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 6814–6823, Aug. 2018.

[31] T. Osman, S. S. Psyche, J. S. Ferdous, and H. U. Zaman, "Intelligent traffic management system for cross section of roads using computer vision," in *Proc. IEEE 7th Annu. Comput. Commun. Workshop Conf.*, 2017, pp. 1–7.

[32] A. Pavlo *et al.*, "Self-driving database management systems," in *Proc. CIDR 2017, Conf. Innov. Data Syst. Res.*, vol. 11, 2017, pp. 1–13.

[33] F. Tang, Y. Kawamoto, N. Kato, and J. Liu, "Future intelligent and secure vehicular network toward 6G: Machine-learning approaches," *Proc. IEEE*, vol. 108, no. 2, 2019, pp. 292–307.

[34] J. Wei, J. He, Y. Zhou, K. Chen, Z. Tang, and Z. Xiong, "Enhanced object detection with deep convolutional neural networks for advanced driving assistance," *IEEE Trans. Intell. Transp. Sys.*, vol. 21, no. 4, pp. 1572–1583, 2019.

[35] N. Taherifard, M. Simsek, and B. Kantarci, "Bridging connected vehicles with artificial intelligence for smart first responder services," in *Proc. IEEE Global Conf. Signal Inform. Process.*, 2019, pp. 1–5.

[36] B. Maaloul, A. Taleb-Ahmed, S. Niar, N. Harb, and C. Valderrama, "Adaptive video-based algorithm for accident detection on highways," in *Proc. IEEE Intl. Symp. Ind. Embedded Sys.*, 2017, pp. 1–6.

[37] A. Jain, H. S. Koppula, B. Raghavan, S. Soh, and A. Saxena, "Car that knows before you do: Anticipating maneuvers via learning temporal driving models," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 3182–3190.

[38] L. Yang, R. Ma, H. M. Zhang, W. Guan, and S. Jiang, "Driving behavior recognition using EEG data from a simulated car-following experiment," *Accident Anal. Prevention*, vol. 116, pp. 30–40, 2018.

[39] A. Maas, Q. V. Le, T. M. O'neil, O. Vinyals, P. Nguyen, and A. Y. Ng, "Recurrent neural networks for noise reduction in robust ASR," in *Proc. 13th Annu. Conf. Int. Speech Commun. Assoc.*, 2012.

[40] H. Salehinejad, S. Sankar, J. Barfett, E. Colak, and S. Valaee, "Recent advances in recurrent neural networks," 2018, *arXiv:1801.01078*.

[41] P. Voigtlaender, P. Doetsch, and H. Ney, "Handwriting recognition with large multidimensional long short-term memory recurrent neural networks," in *Proc. 15th Int. Conf. Frontiers Handwriting Recognit.*, 2016, pp. 228–233.

[42] L. Jing, C. Gulcehre, J. Peurifoy, Y. Shen, M. Tegmark, M. Soljacic, and Y. Bengio, "Gated orthogonal recurrent units: On learning to forget," *Neural Comput.*, vol. 31, no. 4, pp. 765–783, 2019.

[43] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. M. Shroff, "LSTM-based encoder-decoder for multi-sensor anomaly detection," 2016, *ArXiv, vol. abs/1607.00148*.

[44] A. Sood, M. Simsek, Y. Zhang, and B. Kantarci, "Deep learning-based detection of fake task injection in mobile crowdsensing," in *Proc. IEEE Global Conf. Signal Inform. Process.*, 2019, pp. 1–5.

[45] P. Malhotra *et al.*, "Multi-sensor prognostics using an unsupervised health index based on LSTM encoder-decoder," in *Proc. 1st SIGKDD Workshop Mach. Learn. Prognostics Health Manag.*, 2016, *arXiv:1608.06154*.

[46] E. Habler and A. Shabtai, "Using LSTM encoder-decoder algorithm for detecting anomalous ADS-B messages," *Comput. Secur.*, vol. 78, pp. 155–173, 2018.

[47] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[48] F. Huang, X. Zhang, C. Li, Z. Li, Y. He, and Z. Zhao, "Multimodal network embedding via attention based multi-view variational autoencoder," in *Proc. ACM Int. Conf. Multimedia Retrieval*, 2018, pp. 108–116.

[49] T. Bui *et al.*, "Tamper-proofing video with hierarchical attention autoencoder hashing on blockchain," *IEEE Trans. Multimedia*, 2020, vol. 22, no. 11, pp. 2858–2872, Nov. 2020.

[50] A. Patarot, M. Boukallel, and S. Lamy-Perbal, "A case study on sensors and techniques for pedestrian inertial navigation," in *IEEE Proc. Int. Symp. Inertial Sensors Syst.* 2014, pp. 1–4.

[51] X. Feng, Y. Zhang, and J. Glass, "Speech feature denoising and dereverberation via deep autoencoders for noisy reverberant speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2014, pp. 1759–1763.

[52] R. C. Inc. "Raven obd-ii sensor kit" 2020. [Online]. Available: https://ravenconnected.com/

[53] M. N. Azadani and A. Boukerche, "Performance evaluation of driving behavior identification models through CAN-bus data," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2020, pp. 1–6.

[54] C. Lu, Z.-Y. Wang, W.-L. Qin, and J. Ma, "Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification," *Signal Process.*, vol. 130, pp. 377–388, 2017.

[55] F. Karim, S. Majumdar, H. Darabi, and S. Chen, "LSTM fully convolutional networks for time series classification," *IEEE Access*, vol. 6, pp. 1662–1669, 2017.

[56] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Representations (ICLR 2015)*, San Diego, CA, USA, May 7–9, 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[57] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," 2015, *ArXiv, vol. 1505.00853*.

[58] A. Majumdar and A. Tripathi, "Asymmetric stacked autoencoder," in *Proc. Int. Joint Conf. Neural Netw.*, 2017, pp. 911–918.

[59] K. Saleh, M. Hossny, and S. Nahavandi, "Driving behavior classification based on sensor data fusion using LSTM recurrent neural networks," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst.*, 2017, pp. 1–6.

[60] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell, "Actionvlad: Learning spatio-temporal aggregation for action classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 971–980.

[61] W. Sun, S. Shao, R. Zhao, R. Yan, X. Zhang, and X. Chen, "A sparse auto-encoder-based deep neural network approach for induction motor faults classification," *Measurement*, vol. 89, pp. 171–178, 2016.

NIMA TAHERIFARD received the B.Sc. degree in electrical engineering. Since January 2019, he has been working toward the master's degree in electrical and computer engineering with the University of Ottawa, Ottawa, ON, Canada. He is a member of the NEXTCON Research Lab. as a Graduate Researcher. He joined the lab in June 2019. He has been an intern with Raven, Inc., as a part of the Ontario Centers of Excellence Talent Edge Internship Project titled "Vehicular Sensor Fusion for Detection and Characterization of Significant Events." His main areas of research interests are connected and autonomous vehicles, machine learning, deep learning, computer vision, and data analytics.

MURAT SIMSEK (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees from the Department of Electronics and Communication Engineering, Istanbul Technical University (ITU), Istanbul, Turkey, in 2003 and 2012, respectively. He is currently a member of the NEXTCON Research Lab. as a Senior Research Associate with the School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, ON, Canada. He was a Researcher and a Lecturer with the Department of Astronautical Engineering, ITU, from 2012 to 2019. From 1999 to 2012, he was a Research Assistant with the Department of Electronics and Communication Engineering, ITU. During his Ph.D. study, he studied as a Visiting Scholar with Prof. Q. J. Zhang with the Department of Electronics Engineering, Carleton University, Ottawa, Canada, between August 2009 and April 2010. His research interests include surrogate based modeling and optimization, space mapping based modeling and optimization, artificial intelligence, machine learning, artificial neural networks, and knowledge based modeling.

CHARLES LASCELLES received the B.Sc. degree (top of his class) in computer engineering from the University of Ottawa, Ottawa, ON, Canada, in 2013. He is a Firmware Engineer with Raven Connected. He has more than seven years of design and development experience in embedded Linux and Android full stack development, from boot loader to kernel drivers, hardware-abstraction layers, and user-space applications. As a core member of Raven's development team, he contributed to Raven's commercial success as an LTE-connected vehicle monitoring and security platform. When he is not developing you can find him tinkering on various electronic projects and computers, learning about upcoming technologies, playing music, and simply enjoying life.

BURAK KANTARCI (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in computer engineering from Istanbul Technical University, Istanbul, Turkey, in 2005 and 2009, respectively. During his Ph.D. study, he studied as a Visiting Scholar with the University of Ottawa, where he completed the major content of his thesis. He is currently an Associate Professor with the School of Electrical Engineering and Computer Science, and the Founding Director of the Next Generation Communications and Computing Networks (NEXTCON) Research Lab, University of Ottawa. From 2014 to 2016, he was an Assistant Professor with the ECE Department, Clarkson University, where he currently holds a courtesy appointment. He has coauthored more than 200 papers in established journals and conferences, and contributed to 13 book chapters. He is the Co-Editor of two books in the fields of cloud communications and connected health. He has served as the Technical Program Co-Chair of more than 15 international conferences/symposia/workshops. He serves on the editorial board for the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, IEEE NETWORKING LETTERS, IEEE INTERNET OF THINGS JOURNAL, IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING, *Sensors Journal*, and Elsevier *Vehicular Communications*. He is currently serving as the Chair of the IEEE ComSoc Communication Systems Integration and Modeling Technical Committee. He is a Distinguished Speaker of the ACM.