IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 14 September 2023, accepted 2 October 2023, date of publication 11 October 2023, date of current version 18 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3323891

RESEARCH ARTICLE

A Deep Learning-Based IDS for Automotive Theft Detection for In-Vehicle CAN Bus

JUNAID AHMAD KHAN[1], DAE-WOON LIM[1], AND YOUNG-SIK KIM[2], (Member, IEEE)
[1]Department of Information and Communication Engineering, Dongguk University, Seoul 04620, South Korea
[2]Department of Electrical Engineering and Computer Science, Daegu Gyeongbuk Institute of Science and Technology (DGIST), Daegu 42988, South Korea

Corresponding author: Dae-Woon Lim (daewoonlim@gmail.com)

This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korean Government [Ministry of Science and ICT (MSIT)] under Grant NRF-2021R1A2C2011082.

ABSTRACT Driver behavior features extracted from the controller area network (CAN) have potential applications in improving vehicle safety. However, the development of a classifier-based intrusion detection system (IDS) for in-vehicle networks remains an open research problem. To address this challenge, we incorporate novel n-fold cross-validation windowing techniques on two publicly available driving behavior datasets. A driver classification-based IDS is proposed using the LSTM-FCN model that utilizes the strengths of both fully convolutional network (FCN) and long short-term memory (LSTM) networks. These modules allow the model to learn spatial and temporal features and utilize contextual information. In addition, we combine three squeeze and excite (SnE) layers following FCN layers to incorporate adjacent spatial locations and augment a scaled dot product attention mechanism into the LSTM to improve its feature selection and extraction capabilities. Our proposed IDS uses hacking and countermeasure research lab (HCRL) and test datasets, which achieve an improvement in accuracy of 4.18% and 13.99% respectively, from the baseline LSTM-FCN model. The experimental results of our method exhibited an overall accuracy of 99.36% and 96.36% for both datasets and outperformed various state-of-the-art methods.

INDEX TERMS Attention, anomaly detection, automotive IDS, controller area networks, driver classification, FCN, in-vehicle networks, LSTM, squeeze and excitation.

I. INTRODUCTION

The automotive industry has made technological advances that promise unprecedented levels of security, productivity, and ecological benefits. This shift toward the acceptance of novel technologies is accompanied by the rapid growth of sensor technologies in autonomous vehicles [1]. Original equipment manufacturers (OEMs) support these digital transformations in smart/autonomous vehicles through distributed processing techniques [2] such as zonal architectures and electronic control units (ECUs) [3], which communicate only through in-vehicle automotive networks supporting high bandwidth and low latency, such as a controller area network (CAN), a CAN with flexible data rate (CAN-FD), local interconnect network (LIN), and Flexray [4]. Contrary to this, the intelligent transportation system (ITS) incorporates vehicle-to-everything (V2X), which includes vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) technologies to broaden the communication spectrum [5]. Furthermore, the enhanced hardware complexity and advanced integrated functionality have introduced a plethora of security vulnerabilities and attack scenarios for the automotive sector [6], [7]. Among the current security concerns in the automotive industry, vehicle theft is a serious problem. Both hacking and smart vehicles have advanced over time. By exploiting vulnerabilities in a sophisticated vehicle system, car thieves have advanced from stealing a physical car key to gaining remote access [8]. In light of real-world vehicular security, the industry has begun to develop automotive security systems for futuristic vehicles as on-board computer hacking-related vehicle thefts are becoming more prevalent. In order to combat automotive theft and hacking, driver identification can be used to authenticate drivers based on their unique intrinsic characteristics [9]. Biometrics, among

The associate editor coordinating the review of this manuscript and approving it for publication was Vicente Alarcon-Aquino.

other methods, is no longer a futuristic technology for the automotive industry. Nonetheless, it has been used for driver identification systems in the past [10]. Currently, biometrics in industry are associated with several challenges. These challenges include the incorporation of biometric technology into mobile authentication procedures and its integration into large-scale programs, as well as the privacy of personal biometric data against attack vulnerabilities. In addition, the implementation of AI technology in biometrics presents a significant challenge, such as face recognition and feature extraction [11]. Biometric systems are based on human body characteristics such as voice, fingerprints, the face, or the iris that can be automatically analyzed. However, biometric systems can have a high false positive rate (FPR) and are frequently used in uncontrolled environments with sensor noise and other factors that can easily compromise the accuracy of the system [11]. Furthermore, if these factors are not adequately addressed, biometric systems can be costly to implement and maintain. Recently, driver profiling-based intrusion detection has been the center of attention in the research community. However, they have some limitations that push the requirement for novel IDS using driver behavior. Most methods [12], [13] use standard n-fold cross-validation techniques to split the dataset from the same driving trip. This leads to a high correlation if the driver continues to drive in the same pattern. Many previously published schemes [14], [15] confirm the reliability of their results when using a single dataset. In order to address these issues, we have used two publicly available driving datasets in the experiments for generality [12], [16]. The CAN network allows multiple connected ECUs to communicate multivariate data. In fact, this temporal data when extracted from ECUs show high correlation over time while accomplishing mutual operational tasks. For example, the ECUs connected to brake mechanism collectively generate spatiotemporal signals [17]. Therefore, to obtain high detection accuracy, the spatial dependency of multiple ECUs along the temporal dimension must be considered during analysis. Motivated by these reasons, this study seeks to enhance the classification performance of CAN network-based IDS by reconfiguring an existing network using the long short-term memory (LSTM) and fully convolutional network (FCN) [18]. The proposed model incorporates several modules, including queeze and excite (SnE) and scaled dot product attention modules. The FCN and LSTM modules extract spatial and temporal features, allowing the model to learn both types of features. In order to effectively utilize contextual information, SnE layers were added after each FCN layer to aggregate information from adjacent spatial locations. Additionally, a scaled-dot product attention mechanism was introduced into the LSTM model to effectively focus on important features, leading to improved feature selection and extraction capabilities. Overall, the integration of these modules enhances the model's performance by capturing both spatial and temporal features and leveraging contextual information. The contributions of this manuscript can be summed up as follows:

1) Previously published approaches used the standard cross-validation technique [12], [13], [19]. Although these methods attained high accuracy, the data generated by ECUs for different drivers remained highly interrelated i.e., when the need for acceleration arises during driving and the gas pedal is engaged, strongly correlated signals are generated in the in-vehicle network [20]. To address this issue, a novel data-splitting algorithm that does not use training and validation data from the same driving trip is used. In order to reduce correlations between attribute values, the first $n - 1$ trips covered by each driver are used for training, while the final trip is devoted to validation.
2) Existing methods [21], [22] have not evaluated the performance of their algorithms for various window sizes. However, the authors in [23] showed that using windowing techniques can improve performance. Based on this, we present a thorough performance analysis that considers various window sizes and shifts using a temporally-based contingent windowing algorithm to maintain the integrity of the data segments. The windowing algorithm generates overlapping windows of size W and enhances accuracy by splitting the data into smaller, more manageable sections.
3) Earlier methodologies may be limited in their ability to generalize their results because they use a single dataset [12], [13], [21]. However, this research adopted two datasets [12], [16] that were openly accessible to the public in order to generalize and validate the authenticity of the proposed IDS.
4) We conducted extensive experiments (i.e., accuracy and loss evaluation, confusion matrix, and classification report) on two publicly available datasets to evaluate the accuracy of the proposed IDS-based theft detection. The results show that our proposed IDS achieves an improvement in accuracy of 4.18% for the hacking and countermeasure research lab (HCRL) dataset and 13.99% for the test dataset from the baseline LSTM-FCN model [18], [24].

The rest of this paper is structured as follows: Section II presents the related work. Section III provides a dataset description and an introduction to the methodology. Section IV outlines and describes the performance evaluation of the proposed methodology based on experimental analysis as well as the results obtained. Section V compares the results with the competing methodologies. Finally, Section VI concludes our work and identifies gaps on which future researchers can focus.

II. RELATED WORK

Classification-based IDS utilizes data acquired from driving behavior and patterns. Recently, there has been a notable surge in the widespread adoption of artificial intelligence (AI) technologies, i.e., the utilization of machine learning and deep learning methodologies for driver identification and profiling. In the following section, the existing

AI-based driver classification and profiling techniques are discussed and broadly categorized into two groups: traditional machine learning-based approaches and deep learning-based approaches.

A. TRADITIONAL MACHINE LEARNING BASED TECHNIQUES

There are a plethora of studies that utilize several traditional machine-learning techniques to classify driver behavior [25], [26]. In a recent study, Misra et al. [27] analyzed the data extracted from bio-signals and vehicular sensors to classify drivers. The study applied multiple machine learning algorithms to investigate the driving behavior of 40 drivers in different scenarios. However, the study included only young drivers who drove for a short period of time for the dataset generation. Similarly, researchers in another study [28] used bio-signals to assess driver fatigue and explored various ensemble learning methods to evaluate performance. However, their study lacks validation across diverse datasets and hardware implementations, which could limit the generalizability and practical applicability of their findings. Deng and Söffker [29] developed an improved prediction model for driver behavior by combining a Hidden Markov Model (HMM) with fuzzy logic. The results were validated for various driving scenarios and with the participation of seven test drivers. However, further optimization of the objective function in their model could improve its performance. Furthermore, in [30] the study used wearable sensor to non-invasively detect distractions during driving. A progressive classifier was used to categorize driving gestures and generate Hidden Markov Model (HMM) databases to identify disruptive sequences. However, their approach requires data from a wider range of driving scenarios to be generalizable.

Researchers in [31] successfully addressed a classification problem pertaining to aggressive and normal driving styles. Their approach utilized a modified support vector machine (SVM) that mitigated the need for an extensive amount of labeled data during the training phase. Nevertheless, it is worth noting that further enhancements to the results can be achieved by considering SVM variants. In a different study [32] authors utilized an unsupervised Gaussian mixture model (GMM) to detect driver cognitive fatigue. The analysis is based on data extracted from simulated driving scenarios that exhibit similarities and correlations. However, the methodology employed in this study lacks practical utility, as it focuses solely on upper body posture patterns. Likewise, quantitative-only methods are good at measuring things that can be easily quantified, whereas cross-validation can face issues like overfitting and inter-data dependency. Moreover, predictive models are required to make a choice to reduce data to usable information, and predictive models present challenges in terms of interpretability [33]. Therefore, a choice must be made to reduce data to usable information, and novel approaches are required to overcome the limitations of existing methodologies. Driver behavior for intrusion detection utilizing ECU traffic can be considered a temporally-based multivariate driver classification task because data frames from the automotive network incorporate temporal characteristics sequentially. As a result, various machine learning-based models for multivariate classification have been investigated, as shown in Table 1, along with their respective limitations. These limitations highlight the need for a novel IDS to achieve improved classification results.

B. DEEP LEARNING BASED TECHNIQUES

Several significant challenges in various domains, such as medicine [34], [35], agriculture [36], [37], and computer science [38], [39], have been successfully addressed using deep learning models for prediction and classification tasks [40]. Likewise, deep learning models have also shown promising potential in the realm of driver behavior classification. A deep neural network-based energy consumption driving model using stacked LSTM was proposed in [41] to mitigate air pollution, considering car dynamics and sensor data to make a personalized user predictor. However, the model can be further improved by considering the affect of abrupt events associated with driving behavior in traffic. In a later study, Alamri et al. [42] proposed a classification technique employing driving-related bio-signals based data for aggressive driving classification, implementing a deep convolutional neural network (DCNN) model and edge technology to reduce automobile crashes. Nevertheless, the utilization of different models and regularization techniques has the potential to enhance the performance of their monitoring system. Authors in [43] classified driving styles using CNN-LSTM. Unsupervised grouping and voting techniques were used to collect and label driving data from various simulator scenarios. However, network optimization can improve performance in different driving scenarios. In a later study, Wang and Wang-Hei Ho [44] utilized GPS sensor data to achieve driver characterization. They implemented statistical methods to develop a joint histogram map to capture driving behavior. However, the used dataset lacks real-world traffic conditions, and the performance can be further improved using data acquired from ECUs connected to in-vehicle networks.

Recurrent neural networks (RNNs) have found widespread application in a multitude of time series problems, including driver behavior classification and other similar domains [45], [46]. Notably, RNNs have faced the challenge of vanishing gradients when processing sequential data [47]. However, the introduction of LSTM networks has effectively addressed this concern. By leveraging memory cells capable of preserving information over extended periods, the LSTM architecture has successfully mitigated the vanishing gradient issue. Moreover, in contrast to conventional neural networks, LSTM models exhibit the ability to accommodate input sequences of varying lengths, effectively capture long-term dependencies, and discern the significance of irrelevant data, thereby enhancing their utility as classifiers [48].

Zhang et al. in [13] developed CNN and GRU/LSTM-based neural networks to recognize identical driving patterns in automotive network data using temporal dynamics. However, the model has not been validated on larger-scale, realistic driving studies. In reference [18], authors showed that LSTM can improve the performance of FCN models with a marginal increment in parameters, which is a significant advantage for temporal-based multivariate classification tasks. The authors in [24] used marginal data preprocessing and a deep-learning network with a fully convolutional block and an LSTM module for the driver profiling task, as proposed in [18]. Aggressive driving is the primary cause of traffic accidents, and its early detection is essential for safe driving. The authors in [49] investigated speed prediction and energy optimization using a hybrid prediction model. Based on their driving characteristics, they also classified drivers as timid, aggressive, or neutral. However, choosing different clustering techniques can further improve the classification performance. LSTM-FCN based models have shown enhanced performance; however, the authors have outlined in [18] that the performance of LSTM-FCN can be further improved through the application of fine-tuning and refinement techniques. The effective use of LSTM-FCN as a classifier for univariate time series has prompted its application in the analysis of multivariate time series [50], necessitating the requirement of a novel IDS. In Table 2, the contributions and shortfalls of deep learning based models for multivariate classification are discussed in detail.

III. METHODOLOGY

Modern vehicles are equipped with multiple ECUs, which facilitate the extraction of data from the in-vehicle network. This data can be obtained through two primary means: the on-board diagnostics (OBD) connector and CarbigsP, an OBD scanning tool. The purpose of this data extraction is to analyze the behavior of the driver. Using this temporal-based multivariate data, driver classification can be defined as an intrusion detection problem.

A. PROBLEM FORMULATION

A multivariate time-series-based IDS is proposed in this study using a modified LSTM-FCN model. A multiclass driver identification problem as an intrusion detection system is denoted as a total of K-class supervised classification problems. The input to the deep learning-based model is the measurements extracted from N number of ECUs representing unique driving behaviors performed by K drivers. We assume the training dataset as features/K-label pairs are extracted during driving as follows

$$\begin{aligned} D, Y = \{(D_i, y_i)\}_{i=1}^{K} &= (D_1, y_1), (D_2, y_2), \\ &\ldots, (D_i, y_i), \ldots, (D_K, y_K). \end{aligned} \tag{1}$$

D is the temporal based driving behaviors of the drivers and y_i represents the K-class label, where y_i: $y_i \in [1, 2, 3, \ldots, K]$. D_i will be a multivariate time series with N-dimensions, $D_i = [ECU_1, ECU_2, ECU_3, \cdots, ECU_N]$ representing driving behaviors recorded from N numbers of ECU. Given the input D_i, which is driving behaviors extracted from ECU_N, we have to find a function $\digamma$ that classifies driver behaviors. Let $\{f_1, f_2, \cdots, f_n, \cdots, f_N\}$ be the set of N features set obtained from D_i and for multivariate case, $n > 1$ and $1 \leq n \leq N$. The goal of this problem is to find a classifier function $\digamma : X \rightarrow Y$ as below, from sequence to the driver y_i with the behaviors,

$$y_i = \digamma(f_1, f_2, \cdots, f_n, \cdots, f_N). \tag{2}$$

B. DATA DESCRIPTION AND ACQUISITION

This research made use of two publicly available datasets in order to validate the credibility of our suggested IDS. The detailed features of the datasets [12], [16] used for evaluation are presented in Tables 3, 4 and 5. The HCRL driving dataset [12] was compiled using an OBD-II scanner from ten different drivers. All driver data is generated using one vehicle. The dataset consists of 23 hours of driving with 51 features extracted from various sensors linked to the automotive network. The generated data has a sample rate of one hertz (Hz) and was extracted using a single vehicle from KIA Motors to obtain a total of 94,401 data samples. Experimentation used four paths with radically different conditions. Likewise, the test driving dataset [16] contains 51 attributes extracted from four drivers with varying driving styles. The four drivers completed 30 trips in total. Every feature, as well as the trip, is recorded every 1 second (1 Hz), for a total of 58584 samples in the dataset using a single vehicle from Hyundai Motors.

The data obtained from the automotive network through the OBD connection is unsuitable for direct use [52]. The logged data is raw and needs essential preprocessing, i.e., removing any missing values, before being used for intrusion detection. Different features extracted from the CAN bus data may have different scales; therefore, data normalization is required to transform all features on the same scale to use as classifier input [48]. Dimension reduction is also a necessary part of data preprocessing [19] and is required to choose a subset of unique features that contribute more to enhancing the accuracy of the classifier used for intrusion detection. As a result, appropriate preprocessing steps are required, such as cleaning in-vehicle network data, normalizing data features, and identifying unique features to be used as input into the neural network. Because the data is generated by multiple ECUs, data normalization is required to standardize all features. Since each retrieved feature has a distinct range, the normalizing procedure transforms the data for use in IDS. All instances of each feature, i.e., $f_f = \{x_1^f x_2^f, \cdots, x_i^f, \cdots, x_I^f\}$ are transformed using the normalizing procedure. The i_{th} data point x_i^f of a feature f_f is transformed to the x_{norm}^f by using

$$x_{i_{(norm)}}^f = \frac{x_i^f - \mu(f_f)}{\sigma(f_f)}. \tag{3}$$

TABLE 1. Traditional machine learning based techniques.

Proposed by	Year	Contribution	Limitation
Misra *et al.* [27]	(2023)	The study analyzed various machine learning methodologies for classifying driver cognitive distraction using physiological and eye-tracking data, revealing that varying models and training scenarios can improve accuracy.	The generation of the dataset included the participation of a single age group, specifically young drivers, with each participant driving for a short amount of time. The quality of the dataset can be further improved by using a realistic driving simulator.
Subasi *et al.* [28]	(2022)	The study employed a machine learning approach that utilizes bio-signals for detecting driver fatigue and promptly notifying potential hazards associated with driving.	The evaluation of the model must involve multiple datasets, and subsequently, its deployment should be implemented in real-world scenarios.
Ansari *et al.* [32]	(2022)	A unsupervised GMM model was used to detect cognitive fatigue in drivers. The analysis is based on data from simulated driving scenarios that show similarities and correlations.	This study lacks efficient time-series feature modeling and statistical driver models to help drivers detect hazards
Sun *et al.* [30]	(2021)	To identify disruptive sequences, a progressive classifier was used to categorize driving gestures and generate Hidden Markov Model (HMM) databases.	For generalization, data from different driving scenarios is required.
Deng *et al.* [29]	(2018)	Enhance detection and accuracy while lowering false alarm rates for driving prediction	Traditional HMM algorithms do not take interaction-related aspects into account in their prediction methods
Wang *et al.* [31]	(2017)	This paper employs a supervised method to improve classifier performance for both labeled and unlabeled data sets. Furthermore, the amount of data labeling required is lessened	The experiments were performed on a simulator that lacked a functional vehicle

C. DATA SPLITTING AND WINDOWING TECHNIQUE

Cross-validation is the most commonly followed method in a large number of published articles on driver identification. In fact, their methodologies divide the information from every trip into training and validation datasets. Despite the fact that these study results were accurate, when the data was plotted, a strong correlation between the attributes was revealed. To address this issue, we created a data splitting algorithm that does not use training and validation data from the same driving trip. Let n be the total number of trips covered by each driver; out of these n trips, the first $n-1$ trips were used for training, and the last trip data was dedicated to the validation set. In Algorithm 1, the driving dataset is divided into parts by allocating portions to training and validation sets. Each class in the drivers dataset is copied to an empty list, and data is appended from each class to the list using for loop. Two counters, numberOfTraining and numberOfValidation are initiated. It then loops through each driver in a list of drivers and identifies the indices where the time is set to 1, marking the start of a new trip. For each driver, the loop then iterates through these trip start indices and splits the driver's data into trip segments, which are then appended to the two counters. The number of trips for training and validation sets is separated depending on their frequency. Every last trip is appended to numberOfValidation counter. This algorithm is useful for processing large datasets of driving data, allowing for easy separation of the data into training and validation sets. The available driving dataset is partitioned into numerous smaller data chunks using a temporally-based windowing method for driver identification. To maintain the integrity of the data chunk, temporal-based contingent attributes are retrieved. The windowing approach retrieves overlapping data chunks of size W. The driver dataset has been pre-processed into training and validation sets. To enhance the model accuracy, a windowing approach is used to split the data into smaller, more manageable sections. The window size W is set at 60 seconds for main experimentation but can be changed according to requirement. The window is shifted by same amount for each iteration to create overlapping segments. In Algorithm 2, a new empty list is created, which will be used to store each of the individual windows of the dataset. Within the loop, each driver dataset is split into a 60-second window and overlaps with the previous segment by a specified shift. The windowing process is an important step in driver identification, as it allows for more granular analysis of the data and can help to identify subtle differences in driving behavior that may not be apparent in larger segments of data for intrusion detection systems. Algorithm 2 defines the window size and shift values using two variables, windowSize and Shift, which can be altered as required. For each window added, the numberOfWindows variable is incremented. The total number of generated windows is shown in the final output.

D. NETWORK ARCHITECTURE

Various approaches have been proposed for multivariate time series classification to capture the interrelationship between different features. Zheng et al. [53] segmented multivariate time series into univariate components, allowing for individual feature learning and achieving promising results for classification tasks. Nevertheless, their approach failed to adequately capture the interrelationship between different univariate time series. Later, Zhao et al. in [54] addressed

TABLE 2. Deep learning based techniques.

Proposed by	Year	Contribution	Limitation
CAI *et al.* [43]	(2023)	The authors used CNN-LSTM to classify driving styles and collected driving data from various scenarios using the simulated environment.	Network optimization and integrating driving style information can enhance classification performance.
Li *et al.* [49]	(2023)	This study used a hybrid prediction model to examine speed prediction and energy optimization for electric automobile.	The use of various clustering techniques has the potential to improve classification performance.
Alamri *et al.* [42]	(2020)	A DCNN-based aggressive behavior detection system by utilizing cloud-based solutions	Regularization techniques can be used to improve performance.
El Mekki *et al.* [24]	(2019)	In this paper, the LSTM-FCN model is tested against increasing sensor data anomalies. The model was tested on multiple datasets before being implemented as a Linux-based real-time anti-theft system	Lack of a cloud-based learning system in which the classification model can be trained on new datasets to improve identification accuracy
Moukafih *et al.* [51]	(2019)	This paper proposes a LTSM-FCN based classifier. The validity is evaluated using the UAH dataset with various processing window sizes and the F-measure score	The model was validated on a single dataset and tends to lack generality. The model was trained and validated on the same subset, showing a strong correlation between input data
Jimenez *et al.* [41]	(2018)	A new energy consumption estimation model using stacked LSTM was proposed.	The model's performance exhibited a decline in the context of aggressive driving.
Wang *et al.* [44]	(2018)	To characterize driver behavior, extensive simulations were carried out on large-scale GPS data using DNN and LSTM.	The model's performance can be further improved using data acquired from sensors connected to the vehicle.
Karim *et al.* [18]	(2017)	Diverse variants of LSTM-FCN with compact model size are investigated for time series classification	More research is needed for applications in other real-time systems, as well as an understanding of why attention LSTM performs poorly compared to LSTM on some datasets

this limitation by using a modified approach that involved jointly training the multivariate time series for feature extraction, instead of learning individually. Subsequently, Karim et al. [18] proposed a deep learning model with a similar architecture to the one proposed in [54], aiming to tackle a univariate classification task that relies on temporal information. Afterwards, they further improved their model for multivariate classification tasks in [50] by integrating SnE block [55] to FCN architecture and demonstrated improved performance while requiring minimal preprocessing steps. Although the model presented in [18] exhibited promising results for temporal-based classification tasks, the authors acknowledged the need for refinements to achieve even better performance. In this study, we have built upon their work and placed specific emphasis on enhancing the performance through the incorporation of various refinements. These include the selection of different hyperparameters such as window size and shift, as well as the integration of additional modules. The authors in [18] and [50] also emphasized the importance of dimension shuffling in the input to the LSTM block as a means to improve performance. Building upon this insight, we have incorporated dimension shuffling as an input to the LSTM block in our proposed model. Consequently, the LSTM, augmented by the scale dot product attention module, receives input as a form of $f_n \times W$ due to dimension shuffle, where W represents the window size (timestep) and f_n signifies the number of features for $1 \leq n \leq N$. Our proposed model incorporates a FCN architecture consisting of three distinct blocks. Each block encompasses a 1-D Convolutional layer, a batch normalization layer, a Rectified Linear Unit (ReLU) layer. These components are stacked together within each block. In our model, the batch normalization layer is used to stabilize gradients to accelerate convergence by making gradients less sensitive to network weights [56]. The SnE [55] along with a scaled-dot product attention mechanism [57] has been incorporated into our model. The FCN architecture contains a set of convolutional layers, which include a combination of 128, 256, and 128 filters. These filters are equipped with 8, 5, and 3 kernels, respectively. The output of a preceding FCN block with 128 filters and 8 kernels is received by the next FCN block with 256 filters and 5 kernels in this stack and is fed into the last block with 128 filters and 3 kernels. The hyperparameters were fine-tuned using the categorical cross-entropy loss function and the Adam optimizer. During the training phase, a total of 50 epochs were selected for all experiments. A dropout layer with a rate of 0.8 was chosen, and the batch size was configured to be 128. The initialization of the convolution kernels is based on the work referenced in [58]. According to Fig. 1, the proposed method classifies drivers based on temporal data extracted from ECUs.

1) FCN WITH SNE COMPUTATIONAL MODULE

Each of the FCN modules includes a computational SnE module, which dynamically adjusts the input [55]. In our implementation, the dimension reduction ratio r is a hyperparameter. Its value is found to be empirically similar to that used in [55]. When the dimension reduction ratio r

Algorithm 1 Drivers Trips

```
Input  : driver dataset D,Y= $\{(D_i, y_i)\}_{i=1}^{K}$, $where\ D_i = [ECU_1, ECU_2, ECU_3, \cdots, ECU_N]$
Output: training and validation sets
$driversLabel = [y_i], \quad i = 1, 2, \cdots, K$
$driversData \leftarrow [\,]$
for $each\ y_i\ in\ driverLabel$ do
    $driversData$.append(D,Y[$driversLabel$] == $y_i$)
end for
$numberOfTotalTrips \leftarrow [\,]$
for $each\ y_i\ in\ driversData$ do
    $numberOfTrainingSet = 0$
    $numberOfValidationSet = 0$
    $index = [driversData[y_i][Time(s)] == 1]$
    for $each\ j\ in\ len(index)$ do
        if $i < len(index) - 1$ then
            $numberOfTrainingSet = numberOfTrainingSet + 1$
            $numberOfTotalTrips$.append($driversData[y_i][index[j] : index[j+1]]$)
        else if $j == len(index) - 1$ then
            $numberOfValidationSet = numberOfValidationSet + 1$
            $numberOfTotalTrips$.append($driversData[y_i][index[j] :]$)
    end for
end for
return $numberOfTotalTrips,\ numberOfTrainingSet,\ numberOfValidationSet$
```

TABLE 3. Features of HCRL dataset [12] and test dataset [16] used for evaluation.

Dataset features	HCRL dataset [12]	Test dataset [16]
Number of drivers	10	4
Trips for each driver	yes	yes
Number of features	51	51
Sampling frequency	1 sample/sec	1 sample/sec
Extraction port	OBD-II	OBD-II and CarbigsP
Total samples	94,401	58,584
Number of vehicles	Single (KIA Motors)	Single (Hyundai Motors)

Algorithm 2 Shift Parameter for Windowing

```
Input  : $windowSize, shift, driversData$
Output: $numberOfWindows$
$drivers \leftarrow [\,]$
$numberOfWindows = 0$
for $i\ in\ len(driversData)$ do
    $n = driversData/windowSize$
    $w = 0$
    for $j\ in\ n$ do
        $temp = driversData[i][w : w + windowSize]$
        $drivers.append(temp)$
        $numberOfWindows = numberOfWindows + 1$
        $w = w + shift$
    end for
end for
return $numberOfWindows$
```

is set to 8, the number of parameters required to learn these self-attention maps is reduced. The squeeze technique is used to extract global spatial information from each channel, resulting in one activation per channel. Global average pooling (GAP) [59] layers are used to minimize the spatial dimensions of the feature tensor input. The dimensions $[Batch \times h \times w \times C]$ are lowered to $[Batch \times 1 \times 1 \times C]$ by shrinking each feature map to a single vector of size n, where n represents the number of convolutional channels. Using GAP layers, the squeeze technique compresses each feature into a single vector. This compression reduces the overall parameters and thus the computational overhead [60]. A fully connected multi-layer perceptron (MLP) bottleneck structure is used to generate scaling weights with a single hidden layer. The hidden layer is used as a reduction block where the input space is transformed to a smaller dimension defined by the reduction factor r. The compressed space retains its original dimensionality, the same as the input, using weights that adaptively scale each channel of the feature map. Input is of shape $(1 \times 1 \times C)$. Thus, there are C neurons in the input layer. Hidden layer reduces this by a reduction factor r, thus leading to a total number of C/r neurons. Finally, the output is projected back to the same dimensional space as the input, returning to C neurons [61]. In total, you pass the $(1 \times 1 \times C)$ tensor as input and obtain a weighted tensor of the same shape $(1 \times 1 \times C)$ tensor from the excitation module, it is first passed through a sigmoid activation layer which

TABLE 4. Driver classification features used from HCRL dataset [12].

#	Features used	Features detail
1	Long_Term_Fuel_Trim_Bank1	Fuel trims are the long-term percentage changes in fuel consumption.
2	Torque_converter_speed	Torque converters are connectors which transmit rotational energy from prime movers.
3	Maximum_indicated_engine_torque	The upper limit of engine torque.
4	Transmission_oil_temperature	The temperature of the oil within the transmission.
5	Intake_air_pressure	This information is used for determining air density and the engine's air circulation rate.
6	Wheel_velocity_rear_left-hand	The rear left-hand tire's speed.
7	Calculated_LOAD_value	This metric denotes a percentage of the maximum attainable torque.
8	Wheel_velocity_front_right-hand	The front right-hand tire's speed.
9	Accelerator_Pedal_value	This sensor detects acceleration pedal movement.
10	Activation_of_Air_compressor	Working pressure of the air compressor.
11	Wheel_velocity_front_left-hand	The front left-hand tire's speed.
12	Fuel_consumption	The current value of fuel usage.
13	Engine_torque	The torque produced by the engine.
14	Engine_coolant_temperature	The temperature of the engine cooling fluid.
15	Torque_of_friction	The frictional force that develops from friction torque.

TABLE 5. Driver classification features used from test dataset [16].

#	Features used	Features detail
1	Long_fuel_bank	Long term changes measured on engine's fuel while it is running.
2	Steering_wheel_angle	The steering wheel angle used to turn the vehicle tires
3	Mission_oil_temp	Indicator for transmission oil temperature
4	Friction_torque	The frictional force that develops from friction torque
5	Fuel_usage	The current value of fuel usage.
6	Engine_idle_slippage	Transmission idle slippage.
7	Engine_torque_min	The minimum torque that measures the rotational force of the engine.
8	Throttle_position_abs	A sensor used to track an engine's air inflow.
9	Engine_torque_revised	The rotational force supplied by the crankshaft
10	Engine_torque	The torque that measures the rotational force of the engine.
11	Cooling_temperature	The temperature of the engine cooling fluid
12	Inhale_pressure	The cylinder pressure of the engine
13	Road_slope	The amount of force acting on the vehicle
14	Short_fuel_bank	Short term changes measured on engine fuel while it is running.
15	Engine_torque_max	The maximum torque that measures the rotational force of the engine.

scales the values to a range of $[0 - 1]$ [62]. Subsequently the output is applied directly to the input by a simple broadcasted element-wise multiplication represented by $\otimes$ in Fig. 1, which scales each channel/feature map in the input tensor with its corresponding learned weight from the MLP in the Excitation module. The dimension reduction feature of SnE module plays a vital role in improving the accuracy of our proposed model. In the case of multivariate datasets, each feature map affects the following layers differently. However, the baseline LSTM-FCN model, as used in previous works [18], [24] lacks the ability to adaptively rescale feature maps, i.e., the learned self-attention mechanism applied to the output feature maps of preceding layers. At each time sample, the SnE module incorporates self-attention with the intercorrelation among the features of multivariate data [50]. This integration of SnE module significantly improves the accuracy of the proposed model over baseline LSTM-FCN [18], [24].

2) LSTM WITH SCALED DOT PRODUCT ATTENTION

The LSTM network is designed to address the issue of vanishing gradients in recurrent neural networks (RNN) [63]. However, LSTM only addresses unidirectional long-term dependencies. The most common forms of attention exercised are additive and scale dot product [57]. These

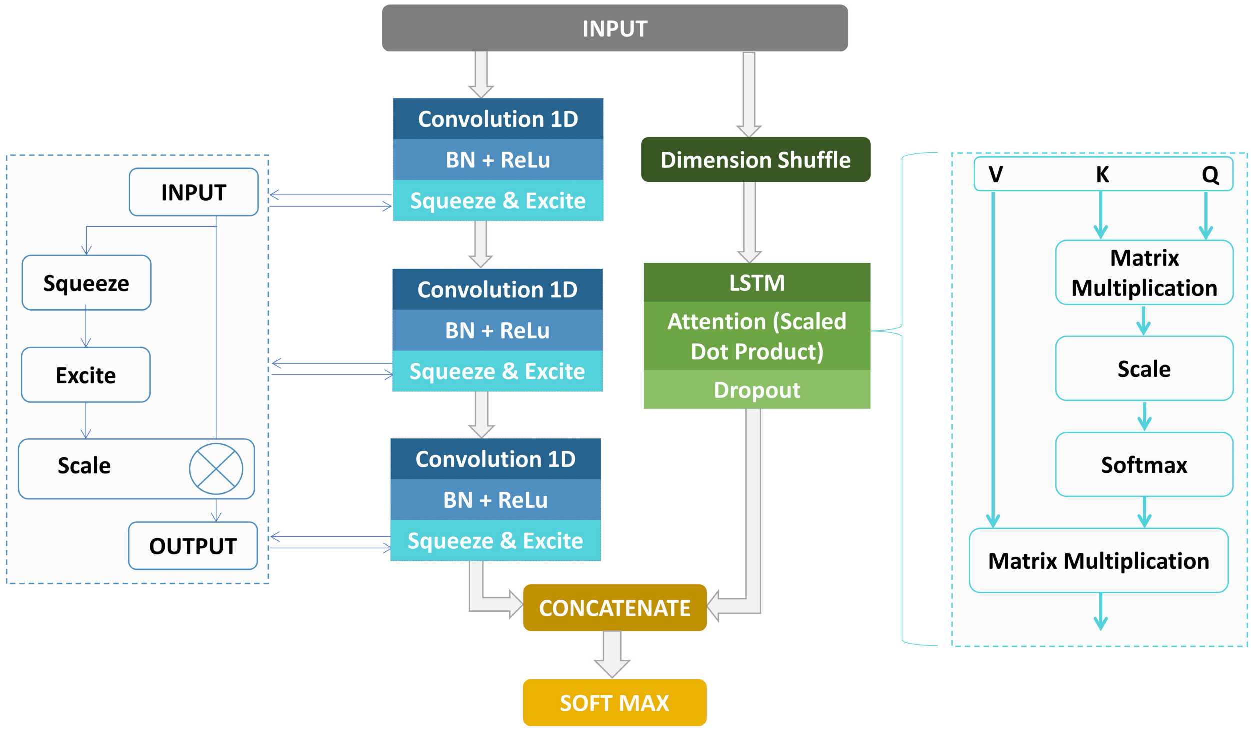


FIGURE 1. Proposed method for driver classification.

mechanisms produce nearly identical results, but in different ways. Multiplicative-based attention is more effectively optimized because it can be implemented using efficient code on high-speed hardware [64]. Our attention module is based on scaled dot product attention [57]. In scaled dot product attention, the similarity between the query vector and the key vector is computed by multiplying their dot products by the inverse square root of their dimension L. Here, L represents the dimension of the query vector and the key vector. The enhancement of attention captures the relationship between time-based sequential data and its long-term dependencies. A softmax function receives the output of the dot product. For large values of L, the resulting dot product will become increasingly large, and the output of the softmax function will be in a region with a small gradient. To mitigate the effects of the exploding effect, the output of the dot product is divided by a scaling factor L and is given by

$$Attention = softmax(\frac{QK^T}{\sqrt{L}})V, \tag{4}$$

where Q, K and V denote the query, key, and value vectors.

IV. PERFORMANCE EVALUATION

We have utilized two publicly available driving datasets for performance evaluation. The datasets were extracted using the CAN bus at a per-second sample rate. The first dataset [12] is denoted as the HCRL dataset, whereas the second dataset [16] is referred to as the test dataset and is used for classifier generalization. Each dataset incorporates 51 features. Tables 6 and 7, show the detailed distribution of driver classes for both datasets.

A. EVAUATION OF HCRL DRIVING DATASET

This subsection provides a comprehensive discussion of the results and evaluations obtained from the training and test sets of the processed HCRL driving dataset [12]. In total, 94,401 data samples were extracted for ten different drivers from the driving dataset. The samples are separated according to each driver, as indicated in Table 6. Each row in the dataset represents a sample received every second from N different ECUs. The data set is segmented into training, validation, and test segments. The training portion receives 66,081 samples (70%) of the data, while the remaining 28,320 samples (30%) are divided between validation and testing. Validation data is used to assess the performance of the model during training, while test data is used to evaluate unseen data. The validation and test sets are kept separate from the training process and are used to evaluate the final performance. Driver A has the fewest samples (7,240) in the entire dataset, while Driver D has the maximum number of samples (i.e., 13,244). Figs. 2 and 3 depict the accuracy and loss of the proposed model for the training and test sets over the course of the 50 epochs, respectively. The gap between the training and test graphs suggests that the model may have been overfitted to the training data. Overfitting occurs when a model learns the training data too well, and as a result leading to suboptimal performance when applied to

TABLE 6. Distribution of each driver class.

Drivers	Training samples	Test smaples	Entire samples
Driver A	5,068	2,172	7,240
Driver B	9,020	3,865	12,885
Driver C	5,250	2,250	7,500
Driver D	9,271	3,973	13,244
Driver E	5,905	2,531	8,436
Driver F	7,708	3,304	11,012
Driver G	5,244	2,248	7,492
Driver H	6,916	2,964	9,880
Driver I	5,466	2,342	7,808
Driver J	6,233	2,671	8,904
Total	**66,081**	**28,320**	**94,401**

unseen data [65]. The smaller the difference between the training and test accuracy, the less likely overfitting is to occur. As shown in Fig. 2, the gap between the training and test accuracy decreases as the number of epochs increases. This suggests that the model is not overfitting to the training data. By the end of the training, the gaps had closed, and the model was performing similarly on both the training and test data. In addition to the accuracy graph, the loss graph can also provide insights into the performance of the model. The loss graph shows the number of errors that the model makes on the training data. As the model learns, the loss should decrease. According to Fig. 3, the loss decreases steadily as the number of epochs increases. This indicates that the model is learning effectively. Fig. 4 depicts the 10 × 10 confusion matrix for a disjointed multiclass driver identification as an indicator of IDS performance. Each column of the matrix along the x-axis reflects the occurrences of a predicted label for Drivers A through J, whereas each row along the y-axis reflects the occurrences of a driver's actual label. An entry in the confusion matrix at a given row and column indicates the frequency of observations for both the predicted and actual labels. The classification report includes a per-class micro-average, weighted average, and accuracy score, in addition to recall, F-1, and precision for the driver identification problem. Fig. 5 depicts the summary of classification model performance, which measures the quality of a classifier predictions. The report indicates that the precision of all drivers falls within a good range. Almost all driver classes exhibited recall rates close to 100 %. The F1, along with precision and recall scores, demonstrate that the output of the classifier is not biased and does not exhibit overfitting or underfitting.

B. EVAUATION ON TEST DRIVING DATASET

For the driving dataset [16], a total of 58,584 data samples were extracted from 4 different drivers. Where each row in the dataset reflects a sample received from N separate ECUs every second. Training, validation, and test segments are part of the data set. 70% of the data is allocated to training, with the remaining 30% allocated to validation and

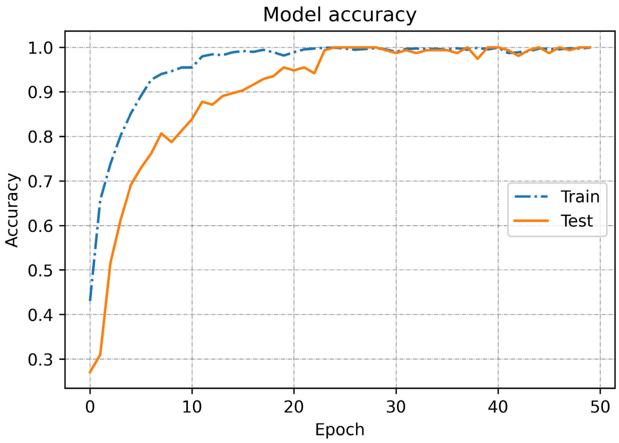


FIGURE 2. Accuracy vs. epoch for HCRL driving dataset [12].

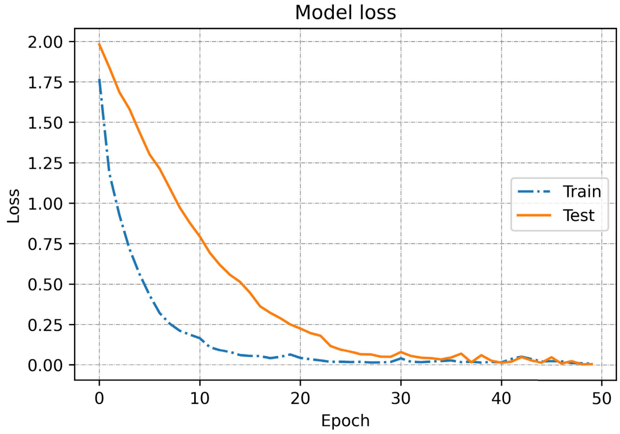
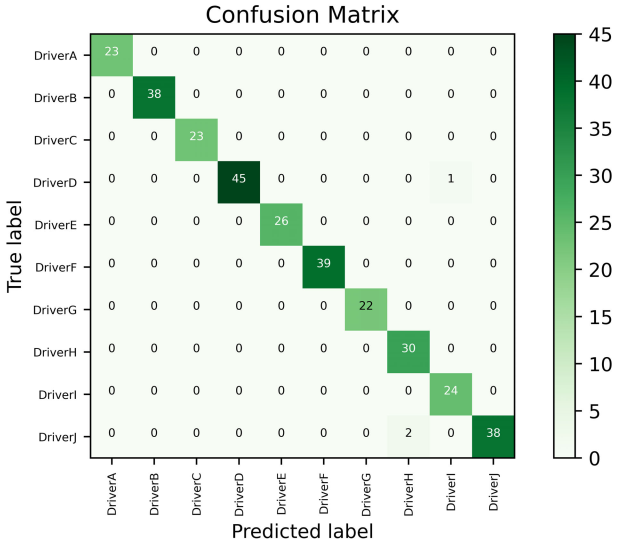


FIGURE 3. Loss vs. epoch for HCRL driving dataset [12].



	DriverA	DriverB	DriverC	DriverD	DriverE	DriverF	DriverG	DriverH	DriverI	DriverJ
DriverA	23	0	0	0	0	0	0	0	0	0
DriverB	0	38	0	0	0	0	0	0	0	0
DriverC	0	0	23	0	0	0	0	0	0	0
DriverD	0	0	0	45	0	0	0	0	1	0
DriverE	0	0	0	0	26	0	0	0	0	0
DriverF	0	0	0	0	0	39	0	0	0	0
DriverG	0	0	0	0	0	0	22	0	0	0
DriverH	0	0	0	0	0	0	0	30	0	0
DriverI	0	0	0	0	0	0	0	0	24	0
DriverJ	0	0	0	0	0	0	0	2	0	38

FIGURE 4. Confusion matrix for HCRL driving dataset [12].

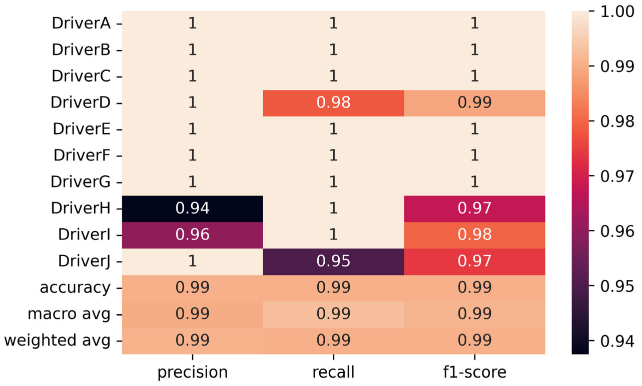


FIGURE 5. Classification report for HCRL driving dataset [12].

TABLE 7. Distribution of each driver class for test driving dataset [16].

Drivers	Training samples	Test smaples	Entire samples
Driver A	10,937	4,687	15,624
Driver B	11,427	4,897	16,324
Driver C	6,812	2,920	9,732
Driver D	11,833	5,071	16,904
Total	**41,009**	**17,575**	**58,584**

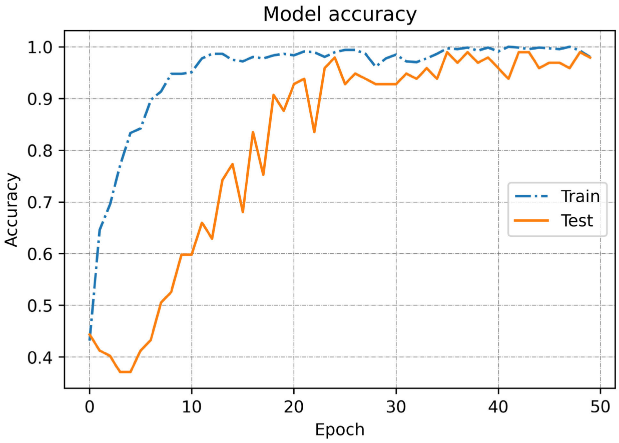


FIGURE 6. Accuracy vs. epoch for test driving dataset [16].

testing. As shown in Table 7, the data samples are segregated according to each of the drivers. Driver C has the fewest samples in the overall data set (9,732), whereas Driver D has the most (16,904). Figs. 6 and 7 depict the accuracy and loss of the proposed model for the training and test sets during the duration of the 50 epochs, respectively. Fig. 8 displays the 4 × 4 confusion matrix for multiclass driver identification as a performance indicator for IDS. Each x-axis column of the matrix represents the instances of a predicted label for Drivers A through D, whereas each y-axis row represents the instances of a driver's actual label. A given row and column entry in the confusion matrix represents the frequency of observations for both the predicted and actual labels.

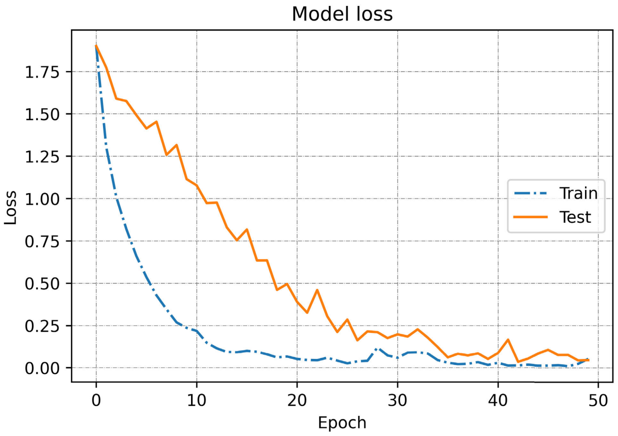


FIGURE 7. Accuracy vs. epoch for test driving dataset [16].

A confusion matrix presented in Fig. 8 is used to evaluate the performance of our proposed driver identification model against a test dataset [16]. The rows of the table correspond to the actual classes, and the columns correspond to the predicted classes. The diagonal elements represent the number of correct predictions made for each class, and the off-diagonal elements represent the number of incorrect predictions. Driver A has the highest true positive value, and Driver C has the lowest true positive. On the other hand, Driver A has three false negatives and three false positives. The classification report for the driving dataset [12] demonstrates lower values for the per-class, micro-average, and weighted average scores for F-1, as well as lower recall and precision for the driver identification problem as compared to the HCRL dataset. Fig. 9 depicts the summary of classification model performance, which measures the accuracy of a classifier's predictions on a test dataset. The report indicates that all drivers' precision falls within the acceptable range, but driver C achieves the best results. Nearly all driver classes exhibit decreased recall values. The F1 score indicates that the output of the classifier is also within an acceptable range, that the dataset is not biased, and that there are no indications of overfitting or underfitting.

V. PERFORMANCE SUMMARY AND COMPARISON

The window size and window shift are important hyper-parameters that directly impact the accuracy of the model [23], [66]. In order to examine the effect of these hyperparameters on accuracy, we conducted a series of experiments on both datasets. The performance comparison graphs in Figures 10 to 15 illustrate the relationship between the number of epochs, ranging from 0 to 50, and the classification accuracy, as depicted on the x-axis and y-axis, respectively.

A. PERFORMANCE COMPARISON USING DIFFERNET WINDOW SIZES AND SHIFT

The datasets are divided into training and test sets using the data splitting method outlined in Algorithm 1. 15 features were selected from a total of 51 to generate performance

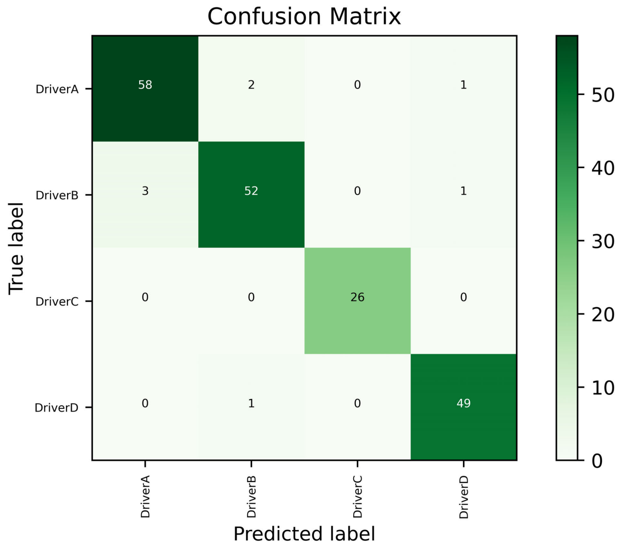


FIGURE 8. Confusion matrix fot test driving dataset [16].

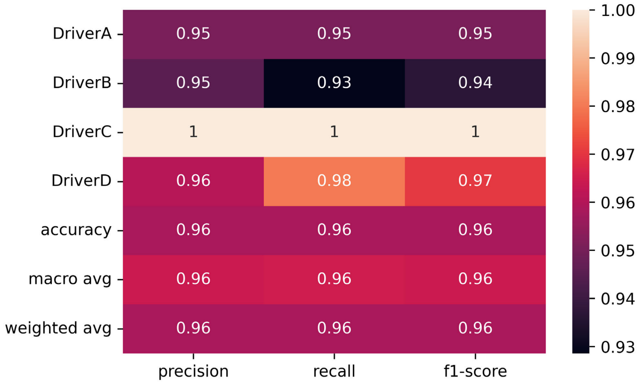


FIGURE 9. The classification report for test driving dataset [16].

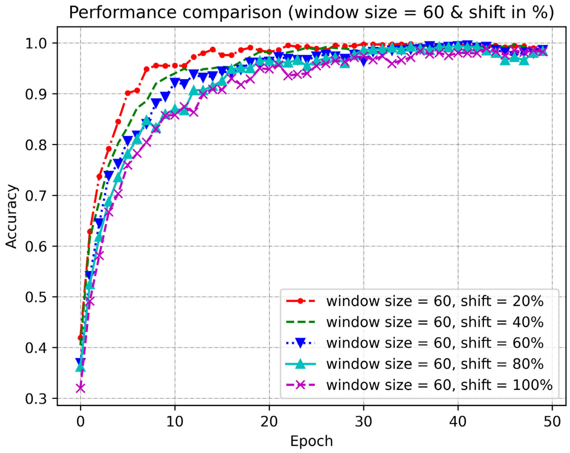


FIGURE 10. Performance comparison (window size = 60 & shift in %) for HCRL driving dataset [12].

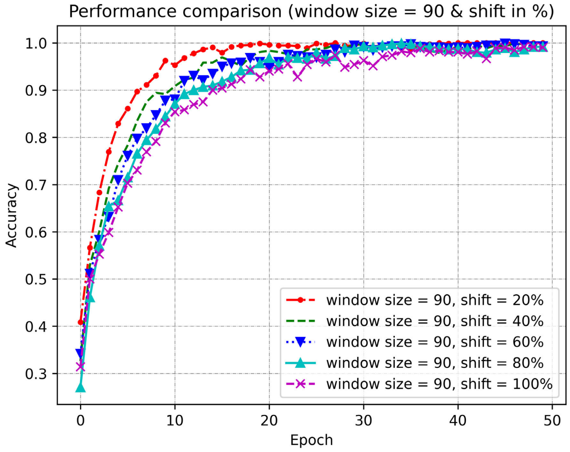


FIGURE 11. Performance comparison (window size = 90 & shift in %) for HCRL driving dataset [12].

comparison graphs using feature selection techniques in the Weka software [67]. Fig.10, Fig.11 and Fig.12 illustrate the classification accuracy of the HRCL driving dataset. Additionally, Fig.13, Fig.14, and Fig.15 present the corresponding classification accuracy for the test dataset. The previous subsection III-D provided a description of hyperparameters, which include the number of layers and kernel dimensions within each layer. In this section, our attention is directed towards additional hyperparameters, specifically the window size and shift. For each window size (60, 90, and 120), different experiments are conducted with varying amounts of shift, and the corresponding accuracy is calculated. A larger window size can improve accuracy because it allows the classification model to capture more long-term temporal dependencies in the data. However, larger window sizes have the inherent disadvantage of slow processing as the model adapts to changes in the data more slowly [68]. Moreover, all of the figures demonstrate that the window shift affects accuracy because it defines the amount of shift between consecutive windows. For a smaller shift with a higher percentage of overlap in the window, the model will only receive a small amount of new data each time the window is shifted, increasing the likelihood that the model will learn the specifics of the training data and be unable to generalize to new data [69]. Alternatively, when employing a 100% shift between windows, the model may not be able to capture essential details of the dataset. This limitation becomes evident through the analysis of Figures 10, 11 and 12, which demonstrate that the accuracy of the HCRL dataset graph is significantly higher with a 20% shift compared to a 100% shift across various window sizes. The observed higher accuracy with a 20% shift suggests that a smaller window shift allows the model to capture temporal patterns within the dataset. A smaller window shift provides a larger amount of overlapping data during each epoch, enabling the model to learn spatial and temporal features and utilize contextual information. On the other hand, Fig.13, Fig.14, and Fig.15 indicate that a 60% window shift yields higher accuracy for the test dataset. This finding suggests that a larger shift is more effective in capturing the relevant information for the test dataset and helps the model focus on a broader context and capture long-term dependencies.

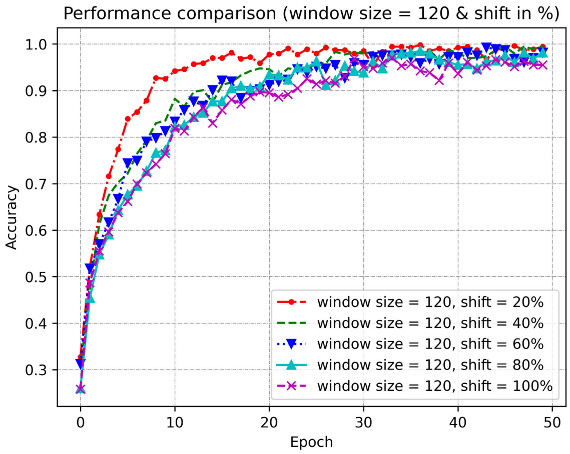


FIGURE 12. Performance comparison (window size = 120 & shift in %) for HCRL driving dataset [12].

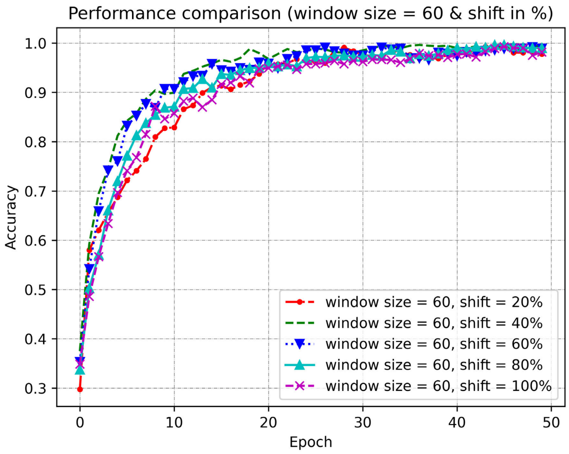


FIGURE 13. Performance comparison (window size = 60 & shift in %) for test driving dataset [16].

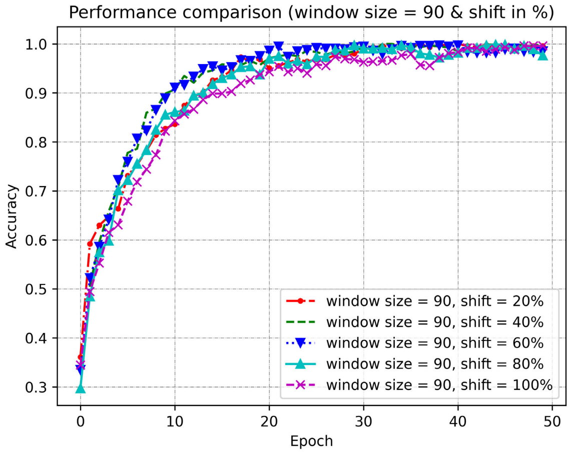


FIGURE 14. Performance comparison (window size = 90 & shift in %) for test driving dataset [16].

B. COMPARISON OF ACCURACY RESULTS WITH OTHER METHODS

In Figs. 16 and 17, we have compared the classification results of the proposed method with those of other deep learning methodologies for HRCL and test datasets. All deep learning models are trained using the Adam optimizer with a sparse categorical cross-entropy function. Throughout the experiment, consistent values for all hyperparameters,

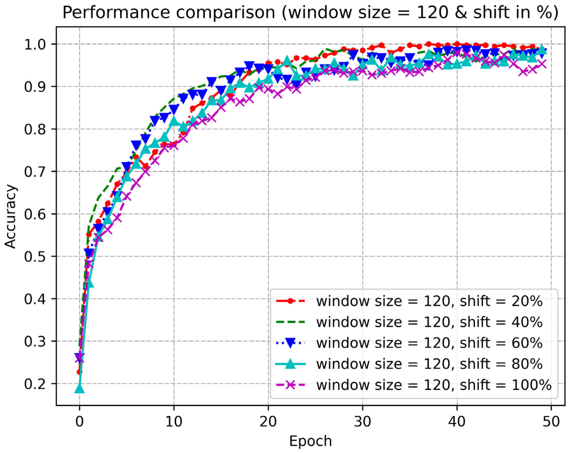


FIGURE 15. Performance comparison (window size = 120 & shift in %) for test driving dataset [16].

TABLE 8. Performace comparison with other models for the HCRL dataset.

Publication	Year	Method	Accuracy
Jiménez et al [41]	2018	Stacked-LSTM	70.74
Mekki et al [24]	2019	LSTM-FCN	95.18
Wang et al [44]	2018	DNN	72.67
Dong et al [70]	2016	CNN	61.74
Alamri et al [42]	2020	DCNN	93.25
Proposed scheme	**2023**	**Attention+LSTM-FCN+SnE**	**99.36**

TABLE 9. Performace comparison with other models for the test dataset.

Publication	Year	Method	Accuracy
Jiménez et al [41]	2018	Stacked-LSTM	60.62
Mekki et al [24]	2019	LSTM-FCN	82.38
Wang et al [44]	2018	DNN	61.14
Dong et al [70]	2016	CNN	51.30
Alamri et al [42]	2020	DCNN	90.16
Proposed scheme	**2023**	**Attention+LSTM-FCN+SnE**	**96.37**

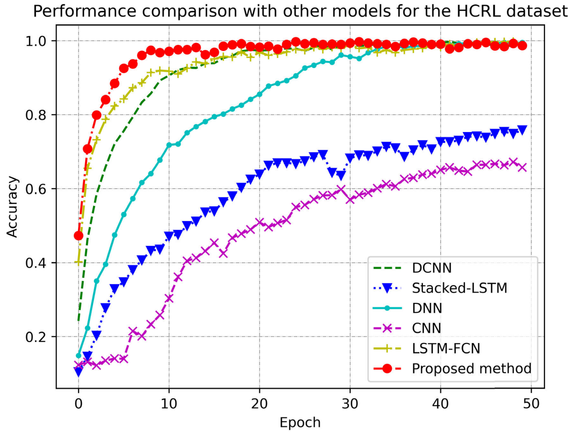


FIGURE 16. Performance comparison with other models for the HRCL dataset [12].

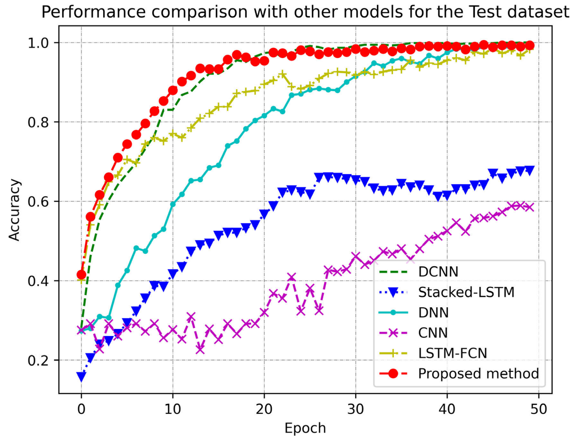


FIGURE 17. Performance comparison with other models for the test dataset [16].

such as window size, window shift, and *r*, were selected to ensure a fair and meaningful comparison with other models. Specifically, the DCNN method [42] employs 1D convolutional layers for the purpose of aggressive behavior detection. Meanwhile, the stacked LSTM approach [41] leverages convolutional LSTM to construct a predictive model aimed at mitigating air pollution. For driver maneuver classification, the DNN model [44] is employed, while the CNN framework [70] is applied to address the driver classification challenge. Lastly, the LSTM-FCN hybrid model [24] integrates both LSTM and FCN components to capture temporal dependencies within the data. Tables 8 and 9 provide accuracy comparison of the proposed model and other deep learning models for both HCRL and test datasets. The results indicate that the proposed model exhibits superior performance when compared to other existing driver classification models. Specifically, achieving the highest accuracy rates of 99.36% for the HCRL dataset and 96.37% for the test dataset. In the case of the HCRL dataset, the LSTM-FCN model demonstrated notable performance with an accuracy rate of 95.18%. Additionally, the DCNN-based driver classification model exhibited a competitive accuracy of 93.25%. Moreover, the stacked-LSTM, DNN, and CNN models achieved respective accuracies of 70.74%, 72.67 and 61.74%. In contrast, when considering the test dataset, the DCNN model emerged as the top performer with an accuracy rate of 90.16%, while the LSTM-FCN model achieved an accuracy rate of 83.38%. The DNN, stacked LSTM, and CNN models obtained accuracies of 61.14 60.62%, and 51.30%, respectively. It can be easily observed that the proposed model showed a significant performance improvement over other models, namely CNN, stacked-LSTM, and DNN. This is due to the fact that our proposed model consists of three FCN blocks extended by SnE blocks, along with an LSTM augmented by a scale dot product attention module and a dropout layer. These modules facilitate the acquisition of spatial and temporal features. The SnE layers, which come after the FCN layers, incorporate neighboring spatial locations, while the augmented scaled dot product attention mechanism contributes to enhancing feature selection and extraction capabilities. However, the other models are unable to extract the temporal and spatial dependencies of the input dataset.

VI. CONCLUSION

This research focuses on developing driver classification based IDS for in-vehicle networks in an effort to improve vehicle safety. The proposed model incorporates the benefits of both FCN and LSTM networks to capture the temporal dynamics of driver behavior using a novel cross-validation technique that yields reliable results to evaluate its performance on two publicly available driving datasets. Necessary preprocessing is performed on the input of the model, i.e., cleaning, normalizing data, and distinguishing unique features. For the proposed IDS, the LSTM-FCN based architecture is enhanced by adding SnE and scaled dot product attention modules. 10 drivers from the HCRL dataset and 4 drivers from the test dataset were classified using the proposed method. The experimental results show that our architecture performs better than other published models, accomplishing improved results with an accuracy of 99.36% on the HCRL dataset and 96.37% on the Test dataset. Furthermore, we achieved precision of 99.09%, recall of 99.04%, and an F1 score of 99.04% for HRCL (10 drivers), and precision of 95.85%, recall of 95.85%, and an F1 score of 95.85% for the test dataset (4 drivers). In conclusion, our obtained results indicate that we can attain better performance for the IDS proposed for vehicle security by augmenting behavioral characteristics extracted from drivers. This could lead to a more secure vehicle security system. In the future, we plan to incorporate FCN with other modules to further enhance the performance of the driver classification system. Furthermore, we intend to fine-tune hyperparameters to accurately capture the underlying patterns and dynamics of different driving scenarios.

REFERENCES

[1] J. Guerrero-Ibáñez, S. Zeadally, and J. Contreras-Castillo, "Sensor technologies for intelligent transportation systems," *Sensors*, vol. 18, no. 4, p. 1212, Apr. 2018.

[2] K. Jo, J. Kim, D. Kim, C. Jang, and M. Sunwoo, "Development of autonomous car—Part I: Distributed system architecture and development process," *IEEE Trans. Ind. Electron.*, vol. 61, no. 12, pp. 7131–7140, Dec. 2014.

[3] N. S. Tany, S. Suresh, D. N. Sinha, C. Shinde, C. Stolojescu-Crisan, and R. Khondoker, "Cybersecurity comparison of brain-based automotive electrical and electronic architectures," *Information*, vol. 13, no. 11, p. 518, Oct. 2022.

[4] Z. Khan, M. Chowdhury, M. Islam, C.-Y. Huang, and M. Rahman, "Long short-term memory neural network-based attack detection model for in-vehicle network security," *IEEE Sensors Lett.*, vol. 4, no. 6, pp. 1–4, Jun. 2020.

[5] V. Maglogiannis, D. Naudts, S. Hadiwardoyo, D. van den Akker, J. Marquez-Barja, and I. Moerman, "Experimental V2X evaluation for C-V2X and ITS-G5 technologies in a real-life highway environment," *IEEE Trans. Netw. Service Manage.*, vol. 19, no. 2, pp. 1521–1538, Jun. 2022.

[6] Y. Wang, B. Yu, H. Yu, L. Xiao, H. Ji, and Y. Zhao, "Automotive cybersecurity vulnerability assessment using the common vulnerability scoring system and Bayesian network model," *IEEE Syst. J.*, vol. 17, no. 2, pp. 2880–2891, Jun. 2022.

[7] J. Khan, D.-W. Lim, and Y.-S. Kim, “Intrusion detection system CAN-bus in-vehicle networks based on the statistical characteristics of attacks,” *Sensors*, vol. 23, no. 7, p. 3554, Mar. 2023.

[8] T. Saha, N. Aaraj, N. Ajjarapu, and N. K. Jha, “SHARKS: Smart hacking approaches for risk scanning in Internet-of-Things and cyber-physical systems based on machine learning,” *IEEE Trans. Emerg. Topics Comput.*, vol. 10, no. 2, pp. 870–885, Apr. 2022.

[9] K. M. Ali Alheeti, R. Al-Zaidi, J. Woods, and K. McDonald-Maier, “An intrusion detection scheme for driverless vehicles based gyroscope sensor profiling,” in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Jan. 2017, pp. 448–449.

[10] A. Jain, L. Hong, and S. Pankanti, “Biometric identification,” *Commun. ACM*, vol. 43, no. 2, pp. 90–98, 2000.

[11] A. Lumini and L. Nanni, “Overview of the combination of biometric matchers,” *Inf. Fusion*, vol. 33, pp. 71–85, Jan. 2017.

[12] B. I. Kwak, J. Woo, and H. K. Kim, “Know your master: Driver profiling-based anti-theft method,” in *Proc. 14th Annu. Conf. Privacy, Secur. Trust (PST)*, Dec. 2016, pp. 211–218.

[13] J. Zhang, Z. Wu, F. Li, C. Xie, T. Ren, J. Chen, and L. Liu, “A deep learning framework for driving behavior identification on in-vehicle CAN-BUS sensor data,” *Sensors*, vol. 19, no. 6, p. 1356, Mar. 2019.

[14] H. Abu-gellban, L. Nguyen, M. Moghadasi, Z. Pan, and F. Jin, “LiveDI: An anti-theft model based on driving behavior,” in *Proc. ACM Workshop Inf. Hiding Multimedia Secur.*, Jun. 2020, pp. 67–72.

[15] P.-Y. Tseng, P.-C. Lin, and E. Kristianto, “Vehicle theft detection by generative adversarial networks on driving behavior,” *Eng. Appl. Artif. Intell.*, vol. 117, Jan. 2023, Art. no. 105571.

[16] K. H. Park and H. K. Kim, “This car is mine! Automobile theft countermeasure leveraging driver identification with generative adversarial networks,” 2019, *arXiv:1911.09870*.

[17] P. Mansourian, N. Zhang, A. Jaekel, and M. Kneppers, “Deep learning-based anomaly detection for connected autonomous vehicles using spatiotemporal information,” *IEEE Trans. Intell. Transp. Syst.*, early access, Jun. 27, 2023, doi: 10.1109/TITS.2023.3286611.

[18] F. Karim, S. Majumdar, H. Darabi, and S. Chen, “LSTM fully convolutional networks for time series classification,” *IEEE Access*, vol. 6, pp. 1662–1669, 2018.

[19] S. Ullah and D.-H. Kim, “Lightweight driver behavior identification model with sparse learning on in-vehicle CAN-BUS sensor data,” *Sensors*, vol. 20, no. 18, p. 5030, Sep. 2020.

[20] M. Marchetti and D. Stabili, “READ: Reverse engineering of automotive data frames,” *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 4, pp. 1083–1097, Apr. 2019.

[21] F. Martinelli, F. Mercaldo, A. Orlando, V. Nardone, A. Santone, and A. K. Sangaiah, “Human behavior characterization for driving style recognition in vehicle system,” *Comput. Electr. Eng.*, vol. 83, May 2020, Art. no. 102504.

[22] J. Xu, S. Pan, P. Z. H. Sun, S. Hyeong Park, and K. Guo, “Human-factors-in-driving-loop: Driver identification and verification via a deep learning approach using psychological behavioral data,” *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 3, pp. 3383–3394, Mar. 2023.

[23] M. N. Rastgoo, “Driver stress level detection based on multimodal measurements,” Ph.D. thesis, School Elect. Eng. Comput. Sci., Queensland Univ. Technol., Brisbane, OLD, Australia, 2019. [Online]. Available: https://eprints.qut.edu.au/134144/

[24] A. E. Mekki, A. Bouhoute, and I. Berrada, “Improving driver identification for the next-generation of in-vehicle software systems,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7406–7415, Aug. 2019.

[25] A. D. McDonald, T. K. Ferris, and T. A. Wiener, “Classification of driver distraction: A comprehensive analysis of feature generation, machine learning, and input measures,” *Hum. Factors, J. Hum. Factors Ergonom. Soc.*, vol. 62, no. 6, pp. 1019–1035, Sep. 2020.

[26] P. Ping, W. Qin, Y. Xu, C. Miyajima, and K. Takeda, “Impact of driver behavior on fuel consumption: Classification, evaluation and prediction using machine learning,” *IEEE Access*, vol. 7, pp. 78515–78532, 2019.

[27] A. Misra, S. Samuel, S. Cao, and K. Shariatmadari, “Detection of driver cognitive distraction using machine learning methods,” *IEEE Access*, vol. 11, pp. 18000–18012, 2023.

[28] A. Subasi, A. Saikia, K. Bagedo, A. Singh, and A. Hazarika, “EEG-based driver fatigue detection using FAWT and multiboosting approaches,” *IEEE Trans. Ind. Informat.*, vol. 18, no. 10, pp. 6602–6609, Oct. 2022.

[29] Q. Deng and D. Söffker, “Improved driving behaviors prediction based on fuzzy logic-hidden Markov model (FL-HMM),” in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 2003–2008.

[30] W. Sun, Y. Si, M. Guo, and S. Li, “Driver distraction recognition using wearable IMU sensor data,” *Sustainability*, vol. 13, no. 3, p. 1342, Jan. 2021.

[31] W. Wang, J. Xi, A. Chong, and L. Li, “Driving style classification using a semisupervised support vector machine,” *IEEE Trans. Hum.-Mach. Syst.*, vol. 47, no. 5, pp. 650–660, Oct. 2017.

[32] S. Ansari, H. Du, F. Naghdy, and D. Stirling, “Automatic driver cognitive fatigue detection based on upper body posture variations,” *Expert Syst. Appl.*, vol. 203, Oct. 2022, Art. no. 117568.

[33] M. M. Malik, “A hierarchy of limitations in machine learning,” 2020, *arXiv:2002.05193*.

[34] X. Ji, Y. Li, and P. Wen, “3DSleepNet: A multi-channel bio-signal based sleep stages classification method using deep learning,” *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 31, pp. 3513–3523, 2023.

[35] M. S. Hossain and M. Shorfuzzaman, “Noninvasive COVID-19 screening using deep-learning-based multilevel fusion model with an attention mechanism,” *IEEE Open J. Instrum. Meas.*, vol. 2, pp. 1–12, 2023.

[36] N. Kaler, V. Bhatia, and A. K. Mishra, “Deep learning-based robust analysis of laser bio-speckle data for detection of fungal-infected soybean seeds,” *IEEE Access*, vol. 11, pp. 89331–89348, 2023.

[37] A. Alharbi, M. U. G. Khan, and B. Tayyaba, “Wheat disease classification using continual learning,” *IEEE Access*, vol. 11, pp. 90016–90026, 2023.

[38] M. Hassaballah and A. I. Awad, *Deep Learning in Computer Vision: Principles and Applications*. Boca Raton, FL, USA: CRC Press, 2020.

[39] H. B. Mahajan, N. Uke, P. Pise, M. Shahade, V. G. Dixit, S. Bhavsar, and S. D. Deshpande, “Automatic robot manoeuvres detection using computer vision and deep learning techniques: A perspective of Internet of Robotics Things (IoRT),” *Multimedia Tools Appl.*, vol. 82, no. 15, pp. 23251–23276, Jun. 2023.

[40] S. Y. Siddiqui, A. Haider, T. M. Ghazal, M. A. Khan, I. Naseer, S. Abbas, M. Rahman, J. A. Khan, M. Ahmad, M. K. Hasan, A. Mohammed. A, and K. Ateeq, “IoMT cloud-based intelligent prediction of breast cancer stages empowered with deep learning,” *IEEE Access*, vol. 9, pp. 146478–146491, 2021.

[41] D. Jiménez, S. Hernández, J. Fraile-Ardanuy, J. Serrano, R. Fernández, and F. Álvarez, “Modelling the effect of driving events on electrical vehicle energy consumption using inertial sensors in smartphones,” *Energies*, vol. 11, no. 2, p. 412, Feb. 2018.

[42] A. Alamri, A. Gumaei, M. Al-Rakhami, M. M. Hassan, M. Alhussein, and G. Fortino, “An effective bio-signal-based driver behavior monitoring system using a generalized deep learning approach,” *IEEE Access*, vol. 8, pp. 135037–135049, 2020.

[43] Y. Cai, R. Zhao, H. Wang, L. Chen, Y. Lian, and Y. Zhong, “CNN-LSTM driving style classification model based on driver operation time series data,” *IEEE Access*, vol. 11, pp. 16203–16212, 2023.

[44] Y. Wang and I. W. Ho, “Joint deep neural network modelling and statistical analysis on characterizing driving behaviors,” in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 1–6.

[45] W. Yu, I. Y. Kim, and C. Mechefske, “Analysis of different RNN autoencoder variants for time series classification and machine prognostics,” *Mech. Syst. Signal Process.*, vol. 149, Feb. 2021, Art. no. 107322.

[46] D. Alvarez-Coello, B. Klotz, D. Wilms, S. Fejji, J. M. Gómez, and R. Troncy, “Modeling dangerous driving events based on in-vehicle data using random forest and recurrent neural network,” in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 165–170.

[47] E. Carvalho, B. V. Ferreira, J. Ferreira, C. de Souza, H. V. Carvalho, Y. Suhara, A. S. Pentland, and G. Pessin, “Exploiting the use of recurrent neural networks for driver behavior profiling,” in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 3016–3021.

[48] C. Ravi, A. Tigga, G. T. Reddy, S. Hakak, and M. Alazab, “Driver identification using optimized deep learning model in smart transportation,” *ACM Trans. Internet Technol.*, vol. 22, no. 4, pp. 1–17, Nov. 2022.

[49] Q. Li, R. Cheng, and H. Ge, “Short-term vehicle speed prediction based on BiLSTM-GRU model considering driver heterogeneity,” *Phys. A, Stat. Mech. Appl.*, vol. 610, Jan. 2023, Art. no. 128410.

[50] F. Karim, S. Majumdar, H. Darabi, and S. Harford, “Multivariate LSTM-FCNs for time series classification,” *Neural Netw.*, vol. 116, pp. 237–245, Aug. 2019.

[51] Y. Moukafih, H. Hafidi, and M. Ghogho, "Aggressive driving detection using deep learning-based time series classification," in *Proc. IEEE Int. Symp. Innov. Intell. Syst. Appl. (INISTA)*, Jul. 2019, pp. 1–5.

[52] Y. I. Khan, *Automotive Cyber Security Challenges: A Beginner's Guide*. U.K.: Amazon Kindle, 2020. [Online]. Available: https://www.researchgate.net/publication/339484340_Automotive_Cyber_Security_Challenges_A_Beginner's_Guide

[53] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Time series classification using multi-channels deep convolutional neural networks," in *Proc. Int. Conf. Web-Age Inf. Manage.* Cham, Switzerland: Springer, 2014, pp. 298–310.

[54] B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu, "Convolutional neural networks for time series classification," *J. Syst. Eng. Electron.*, vol. 28, no. 1, pp. 162–169, Feb. 2017.

[55] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

[56] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[57] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 5998–6008.

[58] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1026–1034.

[59] M. Lin, Q. Chen, and S. Yan, "Network in network," 2013, *arXiv:1312.4400*.

[60] X. Zhang, Z. Xiao, R. Higashita, Y. Hu, W. Chen, J. Yuan, and J. Liu, "Adaptive feature squeeze network for nuclear cataract classification in AS-OCT image," *J. Biomed. Informat.*, vol. 128, Apr. 2022, Art. no. 104037.

[61] A.-M. Rickmann, A. Guha Roy, I. Sarasua, and C. Wachinger, "Recalibrating 3D ConvNets with project & excite," *IEEE Trans. Med. Imag.*, vol. 39, no. 7, pp. 2461–2471, Jul. 2020.

[62] N. Vosco, A. Shenkler, and M. Grobman, "Tiled squeeze-and-excite: Channel attention with local spatial context," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops (ICCVW)*, Oct. 2021, pp. 345–357.

[63] P. Le and W. Zuidema, "Quantifying the vanishing gradient and long distance dependency problem in recursive neural networks and recursive LSTMs," 2016, *arXiv:1603.00423*.

[64] B. Bilonoh and S. Mashtalir, "Parallel multi-head dot product attention for video summarization," in *Proc. IEEE 3rd Int. Conf. Data Stream Mining Process. (DSMP)*, Aug. 2020, pp. 158–162.

[65] X. Ying, "An overview of overfitting and its solutions," *J. Phys., Conf. Ser.*, vol. 1168, Feb. 2019, Art. no. 022022.

[66] A. Dehghani, O. Sarbishei, T. Glatard, and E. Shihab, "A quantitative comparison of overlapping and non-overlapping sliding windows for human activity recognition using inertial sensors," *Sensors*, vol. 19, no. 22, p. 5026, Nov. 2019.

[67] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," *ACM SIGKDD Explor. Newslett.*, vol. 11, no. 1, pp. 10–18, 2009, Nov. 2009.

[68] Z. Wang and A. M. Fey, "Deep learning with convolutional neural network for objective skill evaluation in robot-assisted surgery," *Int. J. Comput. Assist. Radiol. Surg.*, vol. 13, no. 12, pp. 1959–1970, Dec. 2018.

[69] M. Jaén-Vargas, K. M. R. Leiva, F. Fernandes, S. B. Gonçalves, M. T. Silva, D. S. Lopes, and J. J. S. Olmedo, "Effects of sliding window variation in the performance of acceleration-based human activity recognition using deep learning models," *PeerJ Comput. Sci.*, vol. 8, p. e1052, Aug. 2022.

[70] W. Dong, J. Li, R. Yao, C. Li, T. Yuan, and L. Wang, "Characterizing driving styles with deep learning," 2016, *arXiv:1607.03611*.

JUNAID AHMAD KHAN is currently pursuing the Ph.D. degree with Dongguk University, South Korea. His research interests include in-vehicle network security, code-based post-quantum cryptography, digital image processing, applications of machine learning, and deep learning.

DAE-WOON LIM was born in Greenwich, U.K. He received the B.S. and M.S. degrees from the Department of Electrical Engineering, KAIST, Daejeon, South Korea, in 1994 and 1997, respectively, and the Ph.D. degree in electrical engineering and computer science from Seoul National University, in 2006. From 1997 to 2002, he was with LG Industrial Systems as a Senior Research Engineer. He is currently a Professor with the Department of Information and Communication Engineering, Dongguk University, Seoul, South Korea. His research interests include signal processing, wireless communications, cryptography, and security.

YOUNG-SIK KIM (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electrical engineering and computer science from Seoul National University, in 2001, 2003, and 2007, respectively. He joined the Semiconductor Division, Samsung Electronics. He was the Team Leader in research and development of security hardware IPs for various embedded systems, including modular exponentiation hardware accelerator for RSA and elliptic curve cryptography in smart card and mobile application processors, until August 2010. He was a Professor with Chosun University, Gwangju, Republic of Korea, from September 2010 to August 2023. He is currently a Professor with the Department of Electrical Engineering and Computer Science, Daegu Gyeongbuk Institute of Science & Technology (DGIST), Daegu, Republic of Korea. He is also a Submitter for two candidate algorithms (McNie and pqsigRM) in the first round for the NIST Post Quantum Cryptography Standardization. His research interests include privacy and applied cryptography, such as post-quantum cryptography (PQC), fully homomorphic encryption (FHE), privacy-preserving machine learning (PPML), privacy-enhancing technologies (PET), and vehicular security. He is selected as one of 2025's 100 Best Technology Leaders (for Crypto-Systems) by the National Academy of Engineering of Korea.