Time-Series Anomaly Detection in Automated Vehicles Using D-CNN-LSTM Autoencoder

Fatemeh Khanmohammadi and Reza Azmi

***Abstract*— It is reasonable to expect Connected and Automated Vehicles (CAVs) to revolutionize the intelligent world owing to the swapping of seamless and real-time data. Although CAVs provide many benefits to the environment and society, new challenges in term of security, privacy, and safety are emerged by anomalies, errors, cyber-security attacks, or malicious activities that led to accidents with fatal victims. This paper tackles the anomaly detection problem by introducing a novel framework in multi-sensor CAVs, which applies an efficient data preprocessing and deep learning method. In this paper, two preprocessing methods have been used and compared in the deep learning based models. The main contributions of this paper compared to previous works include two fundamental tasks. First, the quality of time series data is improved in the preprocessing phase by Differencing (DIFF) or Moving Standard Deviation (MSD). Second, the applied deep learning method is based on an autoencoder where a combination of Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) is utilized as the autoencoder architecture, which is referred to as D-CNN-LSTM Autoencoder in this paper. The D-CNN-LSTM Autoencoder method optimizes the anomaly detection rate for all of the anomalies, specifically in the case of low magnitude anomalies, enhancing F1-score up to 18.12% in single types of anomalies and 32.83% in mixed types of anomalies. The experimental results indicate the superiority of the proposed method for time series anomaly detection over the state-of-the-art and benchmark methods.**

***Index Terms*— Deep learning, CNN, LSTM, autoencoder, time-series anomaly detection, connected and automated vehicles (CAVs).**

I. Introduction

SINCE Connected and Automated Vehicles (CAVs) have revolutionized the industry of transportation and have brought excellent benefits in decreasing the likelihood of accidents, improving quality of life, and leveraging the efficiency of Intelligent Transportation Systems (ITS) [1]; most researchers have focused on this area. The technologies of connected vehicles and automated vehicles are integrated by CAVs that exploit the advantages of both, with a collective impact [2]. CAVs are envisaged to lead to sustainable environmental development and provide solutions to social and economic challenges [3]. CAVs are configured with sophisticated software and sensors. Computing and communication operations for In-Vehicle Network (IVN) data transfer and Vehicle-to-Everything (V2X) communication are performed through wireless technology. CAVs use both sensors and wireless technology to gain information. In order to reduce the limitations of sensor systems, communication systems such as Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I), and Vehicle-to- Cloud (V2C) communications [4], [5], [6] are used in vehicles [7]. These communications will provide environmental transportability, and security developments that current communication systems are not capable of providing. The technology will lead to a reduction in road accidents in CAVs, so that vehicle crashes are expected to be dropped by 80%, and will also decrease the traffic hours, in that Americans spend 6.9 billion hours annually in traffic [8].

Security attacks on CAVs are in two categories: 1- Internal attacks including attacks on IVN such as Controller Area Network (CAN) bus, OnBoard Diagnostics (OBD) system, GPS system and other sensors of CAVs. 2- External attacks such as data from Road Side Units (RSUs) and information from other vehicles. Despite the many advantages of using CAVs, there are some disadvantages. Increased connectivity and automation cause anomalous readings to affect more components [2] and improve the system's susceptibility to errors or attacks, which can disrupt critical actions such as acceleration, current speed, current position, braking, and so on. Not only malicious attacks but also other various reasons, such as numerous sorts of traffic, sensor age, low battery store, signal disconnection, weather conditions, and other unpredictable environmental disturbances may impact sensor behavior, and consequently leading to inaccurate data reporting. The correct functioning of CAVs relies on the accuracy of sensor data, as it can cause dangerous accidents if they are utilized without the anomaly detection of the observed sensor data [9]. It is significant to specify the faulty sensor in CAVs. Thus, CAVs demand suitable methods to detect anomalies.

Although plenty of anomaly detection methods have been proposed [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], only limited numbers of these works have concentrated on CAVs [9], [24], [25], [26], [27], [28], [29], [30]. Since vehicles will move towards CAVs in the near future, real-time anomaly detection is vital [1], [31]. Several studies apply traditional machine learning methods to detect anomalies in CAVs. The Kalman Filter (KF) has been also employed in CAVs for anomaly detection. Moreover, some recent studies have utilized deep learning such as Convolutional Neural Networks (CNNs) and Recurrent Neural

Manuscript received 3 October 2022; revised 3 March 2023, 7 November 2023, and 14 February 2024; accepted 24 February 2024. Date of publication 27 March 2024; date of current version 1 August 2024. The Associate Editor for this article was A. Hegyi. *(Corresponding author: Reza Azmi.)*

The authors are with the Department of Computer Engineering, Faculty of Engineering, Alzahra University, Tehran 1993893973, Iran (e-mail: azmi@alzahra.ac.ir).

Digital Object Identifier 10.1109/TITS.2024.3380263

1558-0016 © 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: National Chemical Laboratory. Downloaded on February 10,2026 at 13:40:57 UTC from IEEE Xplore. Restrictions apply.

TABLE I
KEY NOTATION DESCRIPTION

Notation	Description
CAV	Connected and Automated Vehicle
ITS	Intelligent Transportation Systems
IVN	In-Vehicle Network
V2X	Vehicle-to-Everything
V2V	Vehicle-to-Vehicle
V2I	Vehicle-to-Infrastructure
V2C	Vehicle-to- Cloud
KF	Kalman Filter
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
CAN	Controller Area Network
OBD	OnBoard Diagnostics
RSUs	Road Side Units
DIFF	Differencing
MSD	Moving standard deviation
AEKF	Adaptive Extended Kalman Filter
OCSVM	One Class Support Vector Machine
EKF	Extended Kalman Filter
IDS	Intrusion Detection System
HMM	Hidden Markov Model
DCNN	Deep Convolutional Neural Network
ADF	Augmented Dickey–Fuller
ReLU	Rectified Linear Unit
FC	Fully Connected
F1	F1-Score
d	Duration
P	Preprocessing
IP	Improvement percentage

Networks (RNNs). The hard temporal dependency has to be between time steps in time series data of the sensor. Recurrent Neural Networks (RNNs) can portray the time correlation in time series data. Several limitations are visible. 1) There is a shortage of using deep learning, especially RNNs, to detect CAVs anomalies, 2) There is a shortage of data preprocessing for converting original low-quality data to high-quality data, and 3) There is a shortage of performance detection for lower magnitudes of abnormal data. To cope with these challenges, our main contributions can be summarized as follows:

1) Our major contribution is introducing two new data preprocessing to optimize the performance of time series models. The transformation of data is based on the Differencing (DIFF) or Moving Standard Deviation (MSD) of time series data for fixing variance and mean, which converts low-quality data to high-quality data.
2) We aim to propose two deep learning methods exploiting CNN and LSTM layers to classify data as an anomaly or normal. These deep learning methods capture spatial and temporal patterns and long term dependencies from time series data. We design a autoencoder consisting of a multi-source sensor mechanism with CNN and LSTM, called as CNN-LSTM Autoencoder in this paper. CNN has been employed as an encoder position to learn spatial patterns, and LSTM has been extended into a decoder position to learn temporal patterns from the encoded spatial patterns. The CNN-LSTM Autoencoder method considers various segments of time series data to learn important features. The proposed method is evaluated using two different approaches, Multi-Channel CNN-LSTM Autoencoder and Independent CNN-LSTM Autoencoder. We also design a method, namely, CNN-BiLSTM, that consists of CNN and LSTM layers for detecting anomalies in CAVs.
3) Evaluating the capacity of the proposed method, the experiment results indicate that the D-CNN-LSTM Autoencoder method effectively improves the anomaly detection rate, not only in the smaller size of abnormal data samples but also in all cases. Comparing to the evaluation of single anomalies in the base paper [1], the Independent D-CNN-LSTM Autoencoder improves the F-score by 18.12%, 5.65%, 5.2% and 3.85% in terms of instant, constant, gradual drift, and bias anomalies. Comparing to the evaluation of mixed anomalies in the base paper [1], the Independent D-CNN-LSTM Autoencoder improves the F-score by 32.83%, 17.51%, 11.6% and 17.9% in terms of instant, constant, gradual drift, and bias anomalies. This robust detection process reduces road accidents caused by abnormal data. Table I shows the key notations in this paper.

II. RELATED WORK

Anomaly detection studies detect rare observations that deviate from other observations significantly. Although a considerable number of researches have been conducted in recent years on the streaming anomaly detection in various fields, such as industry [10], [11], [32], smart energy [16], [17], [18], [19], [33], smart city [21], [22], healthcare [13], [14], [15], intrusion detection [12], [23], only a limited number of works have concentrated on automated vehicles [1], [9], [24], [25], [26], [27].

Different types of methods have been introduced to detect anomalies. Examples of anomaly detection methods are Statistical-based [26], [32], Nearest Neighbor-based [13], Cluster-based [34], Ensemble-based [16], Tree-based [35], [36], Deep learning-based [9], [11], [18], [25]. In this paper, we study anomaly detection in CAVs using deep learning method. Deep learning methods are powerful subset of machine learning methods that utilize hidden layers for extracting hidden features in data, avoiding the step of extracting complex features compared to machine learning methods. Deep learning methods also in terms of their temporal dependencies are more useful than traditional learning methods.

There is a lack of anomaly detection methods in the ITS literature, despite the fatal consequences of failing sensor anomaly detection in CAVs. Limited researches have concentrated solely on cyber security in CAVs. Wyk et al. proposed an anomaly detection method in CAVs using a hybrid method of CNN - KF-χ2-detector [1]. Since KF-χ2-detector is incapable of detecting false data injection attacks statistically derived [37], the second phase of the model may not perform well when these attacks occur. Wang et al. proposed a method for detecting and recovering anomalies in CAVs using an Adaptive Extended Kalman Filter (AEKF) and a One Class Support Vector Machine (OCSVM) [2]. However, the performance of the Extended Kalman Filter (EKF) may be significantly reduced due to uncertainty in the process noise parameter and also the observation error [38]. Moreover, The KF is known to have a computational burden. Alotibi and Abdelhakim employed statistical learning and kinematic

Authorized licensed use limited to: National Chemical Laboratory. Downloaded on February 10,2026 at 13:40:57 UTC from IEEE Xplore. Restrictions apply.

model to detect anomalies in autonomous vehicles for cooperative adaptive cruise control [26]. Javed et al. presented an anomaly detection method in CAVs using a hybrid multi-stage attention method with CNN – LSTM (MSALSTM-CNN) and also a weight-adjusted fine-tuned ensemble is suggested. The results of the experiments show that MSALSTM-CNN gains better performance in detecting single and mixed anomalies [25]. However, MSALSTM-CNN also shows the poor performance in detecting small magnitudes of anomalous sensor values. Predictive votes above 50% are required for ensemble method, and it may not make stable predictions when all prediction votes are below 50%. Agrawal et al. presented anomaly detection models for CAN network traffic using different combinations of CNN and LSTM [40]. Liu and Park presented a framework called LIFE (LIDAR and Image data Fusion for detecting perception errors) using RNN and LSTM units and computer vision algorithms to detect perception errors in autonomous vehicles [31]. However, one of the limitations of LIFE is that the execution time is not optimized in real-time. B. Du et al. introduced a deep learning method for classifying urban traffic passengers using a convolutional residential network and LSTM units [41]. Ashraf et al. proposed a LSTM-based Autoencoder Intrusion Detection System (IDS) for handling time series data to find out anomalous events from ITS [39]. However, accurate detection of attack classes is its limitation, needing new adaptive functions for multi-class problems. Kim et al. presented a data augmentation method in lane detection using a Generative Adversarial Network [9]. Levi et al. proposed a hybrid approach to detect sophisticated and realistic anomalies in connected cars [29]. Normal behavior is learned by Hidden Markov Model (HMM) and then threshold of anomaly is marked by a regression model. However, there are limitations of memory and high computational resources. Song et al. proposed a Deep Convolutional Neural Network (DCNN) for protecting the CAN bus from attacks [30]. However, high false-negative and error rates are obtained. Christiansen et al. proposed a hybrid method of CNNs and anomaly detection to improve the homogenous specifications of scope for detecting anomalies/obstacles [42]. Ryan et al. presented a behavioral anomaly detection method to analyze the risk of end-to-end autonomous driving using CNNs and Gaussian Processes of machine learning [28].

The KF, deep learning, and other methods for detecting anomalies in CAVs have been explored in previous work. In this paper, for detecting abnormal sensor readings we propose a new deep learning method based on autoencoder that is efficient for detecting various types of anomalies especially subtle anomalies [43]. Our method, the D-CNN-LSTM Autoencoder, has not been explored for anomaly detection in CAVs. In previous studies, deep learning methods have been utilized to extract effective information from time-series data. In this paper, we introduce a new framework; the innovation of the proposed method is that it exploits the selected deep learning model and time-series data independently. Almost all previous works focus on more complex models and deep learning techniques to achieve better performance, and the use of input data preprocessing is usually ignored. Different from presented deep learning models for anomaly detection in CAVs, we propose a new model combining CNN, LSTM and autoencoder networks, which can catch more effectively spatial-temporal and periodical features from data stream sensors. Since autoencoder cosiders the correlation of data, CNN-LSTM Autoencoder is suitable for anomaly detection. Moreover, unlike previous works, in the proposed framework two data transformations are used to stabilize statistical properties of time series data for fitting the deep learning model. These data transformations significantly optimize the performance of the anomaly detection of time series. Table II shows a comparison of recent studies in detecting anomalies and attacks in CAVs. Since the anomalies and the dataset of [1] and [25] are similar to this paper, we have discussed these two in the results section.

III. Materials and Method

In this section, we explain the operation of the proposed framework to significantly improve the performance of a deep learning model. The general framework for detecting anomalies is shown in Figure 1.

A. Dataset

This paper utilizes the dataset used for anomaly detection in CAVs [1], [2], [25]. The dataset contains 29800 data rows with 3 attribute columns as follows:

1. **InVehicle_Longitudinal_Speed**: This attribute is vehicle speed (m/s) and is sampled from the CAN bus.
2. **GPS_Speed**: This attribute is the speed (m/s) according to GPS.
3. **InVehicle_Longitudinal_Accel**: This attribute is the vehicle acceleration (m/s) in the longitudinal direction.

Since the original dataset, Safety Pilot Model Deployment (SPMD), had no anomalies, Wyk et al [1] used artificial anomaly injection to dataset. The following four types of anomalies were added to the dataset:

- Instant: The instant anomaly type shows a rapid and unexplained change between two sequential CAV sensors.
- Constant: The constant anomaly type shows a constant behavior that is different from normal sensor readings for a limited period.
- Gradual drift: The gradual drift anomaly type shows a small and linear change during a period that adds to the base values of the sensors.
- Bias: The bias anomaly type shows a constant offset from the sensor's readings for a limited period.

The input data of sensor 1 has three columns as mentioned above, when the anomalies are injected, the data of all three columns are randomly changed and labels 0 and 1 are given based on the data of the first column (in-vehicle speed). The input of sensor 2 was generated in the same way, and if an anomaly has been added to the data of the second column (GPS speed), it gets a label of 1, and otherwise it gets a label of 0. Similarly, sensor 3 is labeled on its third column (in-vehicle acceleration).

Attribute domain values are Float and the data range is different depending on the type and size of the anomaly as well

Authorized licensed use limited to: National Chemical Laboratory. Downloaded on February 10,2026 at 13:40:57 UTC from IEEE Xplore. Restrictions apply.

TABLE II
COMPARISON OF RELATED WORKS

Ref	Year	Method	Type of Anomaly	Attacked system	P	Dataset
[29]	2018	HMM - regression model	Out-of-order, out-of-context, USB firmware update, communication with unknown vendor, OTA malicious updates and malicious application installation attacks	Network, CAN, OS	✓	Generating by simulator SUMO
[1]	2019	CNN - KF	False injection and sensor faults (instant, constant, gradual drift ,bias)	Sensors	×	SPMD
[30]	2020	DCNN	Message injection (DoS , fuzzy, gear spoofing, RPM spoofing)	CAN	×	Car-hacking
[2]	2021	AEKF- OCSVM	False injection and sensor faults (short, noise, gradual drift ,bias)	Sensors	×	SPMD
[25]	2021	- MSALSTM-CNN - Ensemble	False injection and sensor faults (instant, constant, gradual drift ,bias)	Sensors	✓	SPMD
[31]	2021	RNN-convolutional LSTM	Perception error attacks	Sensors	×	KITTI
[39]	2021	LSTM-based autoencoder	- Message injection (DoS ,fuzzy, gear spoofing, RPM spoofing) - Exploits, generic,DoS,fuzzer, and recon attacks	IVN, V2V , V2I	✓	- Car hacking - UNSWNB15
[40]	2022	CNN-LSTM	Message injection (DoS, fuzzy,gear spoofing, RPM spoofing)	CAN	✓	Car hacking

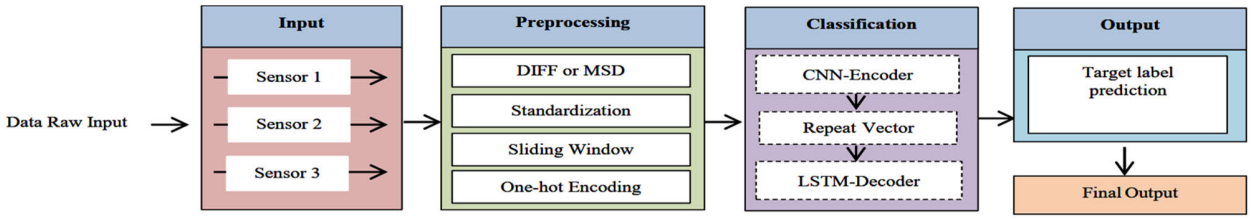


Fig. 1. Overview of the proposed framework.

TABLE III
DATA RANGE OF SENSOR

	Min	Max
In-vehicle Speed	-0.578392961	26.02790171
GPS Speed	-0.371344948	26.02050124
In-vehicle acceleration	-3.034234538	2.535557857

TABLE IV
RATIO ABNORMAL SAMPLES TO NORMAL SAMPLES

Type of Anomaly	Sensor	#abnormal samples/ #normal samples
Instant (base value + 25 * N(0,0.01)	1	538/29262
	2	493/29307
	3	492/29308
Constant (base value + U(0,1),d=10)	1	4897/24903
	2	4601/25199
	3	4583/25217
Gradual Drift (base value + linspace(0,2), d=10)	1	4897/24903
	2	4601/25199
	3	4583/25217
Bias (base value + U(0,1), d=10)	1	4897/24903
	2	4601/25199
	3	4583/25217

as the sensor. For example, the data range of instant anomaly in sensor1 with Magnitude (base value + 25∗ N(0,0.01)) is in the table III.

Consistent with the literature [1], [44], we focus on identifying and detecting abnormal behavior caused by faulty sensors or cyber attacks, occurring as instant, constant, gradual drift and bias anomalies. These types of anomalies have the highest threat and danger for CAVs [1], [45]. The number of abnormal samples to normal samples used in this paper is shown in Table IV; surprisingly there are few instant anomaly. The anomalies of constant, gradual drift and bias have almost similar proportions in the dataset and are relatively infrequent. Anomalies similar to these anomalies are observed in real life datasets, but are relatively infrequent. Nevertheless, significant fraction of samples can be damaged, if an anomaly is not detected immediately [44].

Speed and acceleration sensors considered in this study are vulnerable to cyber-attacks. An injection attack, through OBD system or CAN bus, an acoustic injection attack and spoofing/jamming attacks respectively to in-vehicle speed, in-vehicle acceleration sensors and GPS speed can lead to 4 types of anomalies. Rates of 1% or 5% anomalies are added to the dataset collected every 0.1 seconds with a journey length of 2980 seconds. Anomalies occur in randomly selected attack times and randomly selected sensors. Then these abnormalities are added to the normal value of the damaged sensor that is in the original SPMD dataset, which is called 'base value'. Simulated anomalies change in type, magnitude and duration. The types of anomalies, magnitude and duration have been carefully examined in the results section.

B. Preprocessing

Data preprocessing is an important step in mining and learning from data. Although deep learning methods are accurate for anomaly detection in time series, preprocessing and deep learning are combined to achieve more accurate results in shorter computational time. Since most models of deep learning and time series presume the characteristic of being stable [46], that is, variance, mean, and covariance do not modify over time, it is not the rule in most real datasets and is the exception in practice [47], data preprocessing

Authorized licensed use limited to: National Chemical Laboratory. Downloaded on February 10,2026 at 13:40:57 UTC from IEEE Xplore. Restrictions apply.

is important in predicting time series. Generally, statistical properties of time series are stabilized and seasonality and trends are detected by preprocessing methods [48]. In this paper, first, a data transformation, differencing or moving standard deviation, has been applied to abnormal data, and then the data has been scaled with Standardization and finally is given to the model in a sequence of fixed sliding windows.

1) Data Transformation:

In this paper, time series data are mapped to another representation by transformations. A new time series is produced by passing every of its observations to a mathematical action. The transformed time series that are considered as input usually show properties that are useful for prediction methods [47]. In this paper, two data transformations are used and compared in algorithms based on deep learning. These data transformations are based on moving standard deviation and differencing.

a) Moving standard deviation (MSD):

One of the most applied transformations is moving average smoother [47], [49]. It is useful in finding long-term trends and seasonality and detects changes in behavior of the time series by decreasing the effect of noise. In this paper, we utilized the standard deviation as a statistical function instead of using the mean function, because in our experiments standard deviation reported better performance than mean. Therefore, we have the moving standard deviation transformation. This data transformation has not been applied according to our knowledge in the literature. Moving standard deviation transforms the sequence S into $S^{(k)}$, where the standard deviation of k successive values of S is every value in $S^{(k)}$. The moving standard deviation is calculated for consecutive values (window size = k) as:

$$S^{(k)}[i] = \sqrt{\frac{\sum_{j=i}^{i+k-1} \left(X_j - \overline{X}\right)^2}{k-1}}, \quad 1 \le \mathrm{i} \le \mathrm{n} - \mathrm{k} + 1 \tag{1}$$

where X_j indicates the original time series values and $\overline{X}$ denotes the mean of all the values in the window of size k. In this paper, different window sizes are examined and the best window size used in our models is k=5.

b) Differencing (DIFF):

The use of DIFF preprocessing is common in domains of finance and economics [50] to solve the time series data stationarity problem. It is also used in industry [48], but it is used in the domain of CAVs for the first time in this paper. The stationarity of time series is guaranteed by Augmented Dickey–Fuller (ADF) test [51], which is a statistical significance test. The lack of stationary time series is the null hypothesis in ADF. If test statistic is less than the crucial values (%1, %5, %10) or significance level is below the p-value (0.05), the time series can be considered stationary and null hypothesis can be rejected. The ADF test guarantees the non-stationary feature in our time series data. For example, Table V shows the result of the test on instant anomaly (N(0,0.01)* 25 + base value). The value of the test statistic is less than the crucial values (%1, %5, %10) and the significance level is less than p-value = 0.05.

TABLE V
ADF TEST STATISTIC

Test statistic	-17.030067261781788
significance level	8.337580583419933e-30
1% level	-3.430569642782574
5% level	-2.8616370761459278
10% level	-2.566821670674757

A differencing transformation is a simple method that can omit the seasonal structure and trend from the time series through a few computation [48], which is done by just subtracting the previous value from each value. This is referred as first order differencing. This process can be done again or a number of times. In this paper, first order differencing is used with the deep learning algorithms. Data transformation is based on the first differences of the series:

$$\widehat{x_t} = \nabla x_t = x_t - x_{t-1} \tag{2}$$

Although differencing transformation seems to be very simple process, it can gain prediction results that not only are very accurate and promising but also do not disturb the properties of its time series [48].

2) Data Normalization:

Two common methods of data normalization are min-max normalization and standardization. Normalization is the process of scaling numerical features in a dataset to a consistent range, typically between 0 and 1. This is done by subtracting the minimum value of the feature from each data point and then dividing by the range (the difference between the maximum and minimum values). Mathematically, for a value f of a feature F is scaled to f' as following:

$$f' = \frac{f - min(F)}{max\,(F) - \min(F)} \tag{3}$$

Standardization, also known as z-score normalization, transforms numerical features to have a mean of 0 and a standard deviation of 1. It helps make the distribution of the data more standard by centering it around on zero and scaling it. Mathematically, the values of a feature F are scaled based on the mean (μ) and standard deviation (σ). A value f of feature F is scaled to f' as following:

$$f' = \frac{f - \mu(F)}{\sigma(F)} \tag{4}$$

In this paper, the performance of the proposed model was experimented with standardization and normalization. The result of experiments showed that standardization has a better performance, so it was considered in the data preprocessing. The standardization was applied to the data after the data transformation stage.

3) Sliding windows:

The vehicle sensor network consists of three sensors used to collect information. Each sensor provides readings

Authorized licensed use limited to: National Chemical Laboratory. Downloaded on February 10,2026 at 13:40:57 UTC from IEEE Xplore. Restrictions apply.

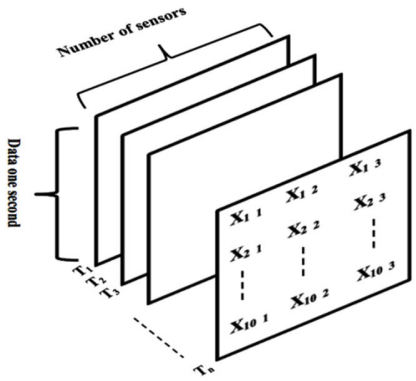


Fig. 2. The three-sensor data matrix structure in the proposed framework.

with 3 columns (in-vehicle speed, GPS speed and in-vehicle acceleration) that is finally given to the model in a sequence of fixed sliding windows with a size of $10 * 3$. Sliding windows are examined in different sizes and strides. The best performance were achieved with window size $= 10$ and stride $= 1$. Data of sensors can be shown in Figure 2.

The label of each window is OR operation on labels of window elements. Each data row in the sliding window has been labeled 0 or 1. If each data row in the window has an anomaly (label 1), the entire window has an anomaly. Therefore, we used OR operator for labeling the entire window since labels are binary (0, 1).

The target label is converted to label vectors using one hot transformer and then will be used for sequence classification by the deep learning methods.

C. Method

Early detection of sensor anomalies is essential. Two separate blocks, feature extraction and classification, are generally utilized in typical systems. These customized and hand-crafted features may be a sub-optimal selection and require a high computational cost for using in real-time applications. This paper uses fast and accurate anomaly detection methods to combine the stages of classification and feature extraction into a single learning body.

1) CNN-BiLSTM:

The CNN networks are a class of deep learning networks that have showed a promising performance in image analysis tasks and recently in vehicular network security applications [40], because of being able to capture local features better than other neural networks [52]. We experimented a variety of deep learning models to detect 4 different types of anomalies. The results showed that CNN layers have been beneficial for detection. Hochreiter et. al presented LSTMs for solving the long-term dependency problems. Since the popularity of using LSTMs is increasing in the temporal analysis [40], we use LSTM layer in our models. Although LSTM is capable of performing well on long sequences, we apply BiLSTM to efficiently use feed-forward and backward hidden states. Our proposed CNN-BiLSTM method consists of CNN and BiLSTM layers to do a thorough spatial and temporal analysis of time series data to detect anomalies in CAVs.

2) CNN-LSTM Autoencoder:

Autoencoders have different uses, the most common of which may be automatic feature extraction. Time series of data show various characteristics such as autocorrelation. In order to consider the data correlation, an autoencoder is suitable for detecting anomalies [53].

This section explains the concepts of autoencoder, CNN and LSTM, as the proposed methods consists of them. Autoencoders are used to learn self-supervised encoding of input data. Because autoencoders learn to encode input data in a compressed representation, the statistical correlation properties of the main distribution can be preserved from the most related inputs for the reconstruction. An autoencoder is a deep learning model that attempts to copy its input to its output.

The encoder learns how to interpret the input and compresses it to the internal representation defined by the bottleneck layer. The encoding layers take the input vector x and map it to the hidden representation $z = f_{\emptyset}(x)$, through the mapping:

$$z = f_{\emptyset}(x) = \sigma(Wx + b) \tag{5}$$

where the function $\emptyset$ indicates the encoding function and is represented by (W, b), W is the weight matrix and b is the bias vector. σ is a Rectified Linear Unit (ReLU) function that acts as a non-linear function on matrix elements. The decoder takes the output of the bottleneck layer and tries to reconstruct the input.

The mapped representation of z is decoded to have the reconstruction matrix $x' = g_{\psi}(z)$.

$$x' = g_{\psi}(z) = \sigma\left(W'z + b'\right) \tag{6}$$

where the function ψ indicates the decoding function and is represented by (W', b').

The parameters of the autoencoder W, W', b, and b' are improved by minimizing a loss function. We employed the binary cross-entropy loss function because it is usually known for binary classification.

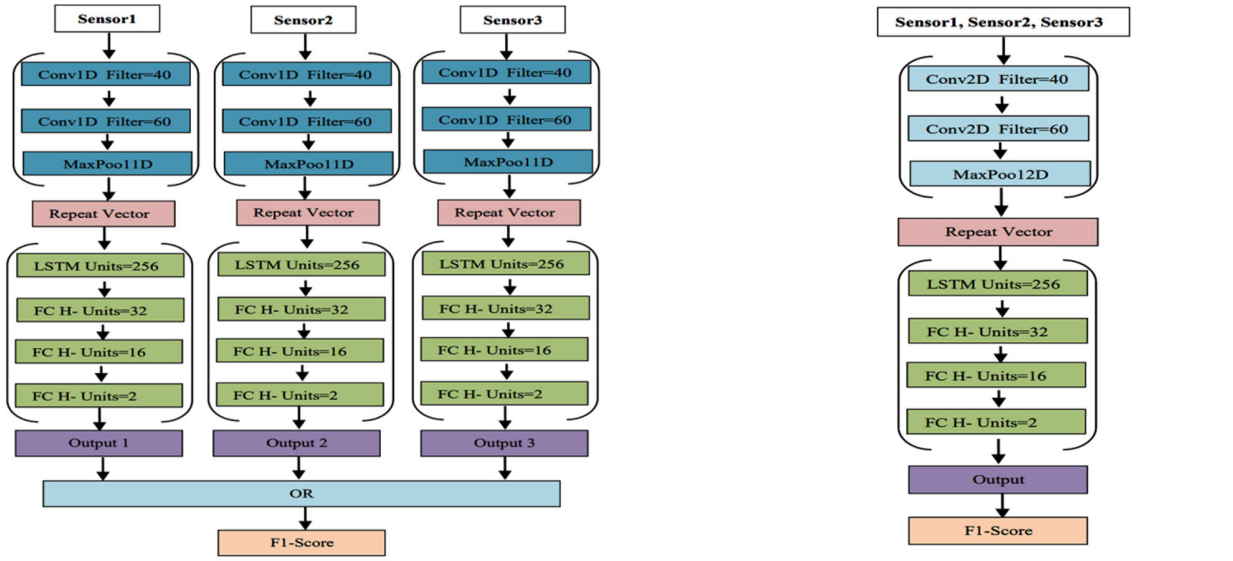
$$BCE = -\sum_{k=1}^{K}\left[x_k \log(x'_k) + (1 - x_k)\log\left(1 - x'_k\right)\right] \tag{7}$$

A CNN-based encoder and a LSTM-based decoder constitute two significant elements of the proposed method. In our proposed autoencoder model, CNN layers are used as encoders and LSTM layers are used as the decoder. CNN layers act as the encoder to extract spatial features. The spatial feature maps generated by CNN-Encoder are temporarily dependent on previous time steps. An LSTM-Decoder is designed to represent those temporal dependencies. It decodes the CNN-Encoder output and reconstructs the sequence data using hidden states (feature maps).

3) Modeling Approaches:

The diagram of the CNN-LSTM Autoencoder model is illustrated in Figure 3. We present two approaches

Authorized licensed use limited to: National Chemical Laboratory. Downloaded on February 10,2026 at 13:40:57 UTC from IEEE Xplore. Restrictions apply.



(a) Independent CNN-LSTM Autoencoder (b) Multi-Channel CNN-LSTM Autoencoder

Fig. 3. The autoencoder structure of the proposed method.

for anomaly detection. Figure 3.a shows that separate models are trained for each sensor with the same network structure. Three sensor signals are measured every 0.1 s to feed the system. The window length is considered 1s (10*3), as suggested by [1]. Figure 3.b shows that we also present a single model with data of three sensors called as Multi-Channel CNN-LSTM Autoencoder. We performed a concatenation between data of three sensors. The size of windows changed to 30 * 3. Moreover, the label of sensors that is finally given to Multi-Channel model is the result of applying OR operation on labels of three sensors. Therefore, three sensors are trained at once. We also used two-dimensional convolution layers(Conv2D) instead of one-dimensional(Conv1D).

The implementation code is available on the GitHub link.

IV. Results

The anomalies change in type, magnitude and duration. Duration (d) means time steps, each time step is measured with an interval of 0.1 seconds. We examine the performance of the proposed methods by considering the DIFF and MSD preprocessing and without them. Also we compare the results with the results of [1] and [25] methods. F1-score is the evaluation metric.

Python with Keras is applied as the programming language. Due to the limitation of funds, all experiments are performed in the Google Colab that enables researchers of deep learning to use a free Tesla T4 GPU.

We experimented a variety of deep learning models to detect 4 different types of anomalies. The CNN and the LSTM layers were selected by experiments with different numbers of layers. The CNN-Encoder consists of two convolutional layers and one max-pooling layer with a random dropout rate of 0.1 between layers to prevent overfitting. The two convolution layers have 40 and 60 filters in 3*3 sizes without striding and with the same padding. The non-linear activation function ReLU was applied in the convolutional layers. Feature maps produced by the CNN-Encoder temporally rely on previous time steps. These temporal dependencies are represented by the design of an LSTM decoder. The CNN-Encoder output is decoded and reconstructed data using feature maps. The LSTM decoder consists of one LSTM layer and three Fully Connected (FC) neural network layers. The one LSTM layer has 256 units and fully connected layers have 32, 16, and 2 hidden units. In the CNN-BiLSTM Model, the CNN and BiLSTM layers were also selected by experiments with different numbers of layers, filters and units. Two convolutional layers have 40 and 60 filters in 3*3 sizes. There is one max-pooling layer with a random dropout rate of 0.2 between layers to prevent overfitting. We use one LSTM layer that has 60 hidden units and two fully connected layers with 30 and 2 hidden units.

In both models, the final fully connected layer has 2 hidden units and a sigmoid activation function; if an anomaly is available, the response variable equals to 1. Otherwise, it equals to 0. The L2 regularization was used in the last dense layer by default value (0.01) to prevent overfitting. This paper used early stopping for checking the accuracy of the validation set. Moreover, we used the saving model, checkpoints, and CSV Logger to store all weights per epoch and select the best weight, and to store accuracy and loss in the validation set. Cross-validation can be applied in different techniques. Since the data order is important for time series data, time series cross-validation has been used in this paper. We have utilized 10-fold cross validation to evaluate the model's performance. In the experiments, we used batch size 128 and epoch 500.

A. Single Anomaly Types

The ability to detect anomalies with the proposed methods according to experimental models [1], [25] for anomalies instant, constant, gradual drift, and bias is checked.

1) Instant: The ability to detect anomalies by CNN-LSTM Autoencoder and CNN-BiLSTM are indicated in Table VI.

Authorized licensed use limited to: National Chemical Laboratory. Downloaded on February 10,2026 at 13:40:57 UTC from IEEE Xplore. Restrictions apply.

TABLE VI
DETECTION PERFORMANCE OF INSTANT ANOMALY

	Method	Independent CNN-BiLSTM						Independent CNN-LSTM Autoencoder					
	Preprocessing	-		DIFF		MSD		-		DIFF		MSD	
Row	Anomaly Magnitude	F1	Time	F1	Time	F1	Time	F1	Time	F1	Time	F1	Time
1	base value + 25* N(0,0.01)	56.73	1.757	83.59	1.811	73.33	1.945	**68.03**	1.573	**85.52**	1.619	**79.98**	1.57
2	base value + 100* N(0,0.01)	84.39	2.024	93.71	2.008	89.71	1.965	**92.82**	1.707	**94.65**	1.458	**92.95**	1.553
3	base value + 500* N(0,0.01)	92.05	1.904	98.50	1.689	97.40	1.956	**97.69**	1.614	**98.90**	1.774	**98.34**	1.76
4	base value + 1000* N(0,0.01)	93.47	1.806	**99.17**	1.891	97.90	1.775	**97.09**	1.655	99.06	1.734	**98.61**	1.571
5	base value + 10000* N(0,0.01)	98.49	1.93	99.57	1.684	**99.64**	2.001	**99.29**	1.644	**99.96**	1.616	99.61	1.735

TABLE VII
DETECTION PERFORMANCE OF CONSTANT ANOMALY

	Method		Independent CNN-BiLSTM						Independent CNN-LSTM Autoencoder					
	Preprocessing		-		DIFF		MSD		-		DIFF		MSD	
Row	Anomaly Magnitude	d	F1	Time	F1	Time	F1	Time	F1	Time	F1	Time	F1	Time
1	base value + U(0,5)	3	94.45	1.912	99.02	1.861	97.54	1.873	**96.01**	1.504	**99.05**	1.633	**97.99**	1.552
2	base value + U(0,5)	5	94.57	1.889	99.18	1.917	98.63	1.872	**96.84**	1.507	**99.52**	1.593	**98.89**	1.559
3	base value + U(0,5)	10	95.08	1.827	99.59	1.758	**99.17**	1.892	**97.34**	1.735	**99.59**	1.639	99.10	1.703
4	base value + U(0,3)	10	92.68	1.801	99.17	1.787	98.65	1.948	**95.38**	1.757	**99.29**	1.546	**99.02**	1.738
5	base value + U(0,1)	10	81.12	1.835	98.29	1.958	97.08	1.807	**90.01**	1.619	**98.35**	1.56	**97.37**	1.676

TABLE VIII
DETECTION PERFORMANCE OF GRADUAL DRIFT ANOMALY

	Method		Independent CNN-BiLSTM						Independent CNN-LSTM Autoencoder					
	Preprocessing		-		DIFF		MSD		-		DIFF		MSD	
Row	Anomaly Magnitude	d	F1	Time	F1	Time	F1	Time	F1	Time	F1	Time	F1	Time
1	base value + linspace(0,4)	10	92.85	1.911	99.77	1.822	98.94	1.823	**97.06**	1.646	**99.91**	1.64	**99.40**	1.796
2	base value + linspace(0,4)	20	95.82	1.842	99.96	1.974	**99.96**	1.756	**97.98**	1.541	**99.96**	1.609	**100**	1.632
3	base value + linspace(0,2)	10	86.29	1.802	99.08	1.875	97.28	1.91	**93.02**	1.533	**99.40**	1.553	**98.01**	1.652
4	base value + linspace(0,2)	20	90.56	1.806	**99.89**	1.806	98.95	1.791	**97.08**	1.638	**99.89**	1.636	**99.36**	1.52

TABLE IX
DETECTION PERFORMANCE OF BIAS ANOMALY

	Method		Independent CNN-BiLSTM						Independent CNN-LSTM Autoencoder					
	Preprocessing		-		DIFF		MSD		-		DIFF		MSD	
Row	Anomaly Magnitude	d	F1	Time	F1	Time	F1	Time	F1	Time	F1	Time	F1	Time
1	base value + U(0,5)	3	93.93	1.759	97.87	1.859	96.29	1.941	**96.12**	1.705	**98.05**	1.603	**97.68**	1.727
2	base value + U(0,5)	5	94.38	1.946	98.29	1.817	96.59	2.018	**96.58**	1.649	**98.33**	1.731	**97.75**	1.753
3	base value + U(0,5)	10	94.88	1.923	98.64	1.728	97.31	1.774	**96.61**	1.62	**98.75**	1.613	**97.81**	1.673
4	base value + U(0,3)	10	91.12	1.864	96.01	1.759	94.73	1.743	**94.42**	1.547	**96.52**	1.791	**96.07**	1.673
5	base value + U(0,1)	10	76.71	1.944	91.41	1.763	86.80	1.824	**84.70**	1.598	**92.08**	1.69	**89.67**	1.559

The anomaly detection performance increases with increasing anomaly magnitude. When the magnitude of the anomaly is the lowest "base value + 25 * N (0, 0.01)" in the first row of the table, the models performance is poor, whereas the models show much better performance by applying the DIFF and MSD process. The deep learning models perform poorly without considering the data preprocessing. As a result, the performance of the models increases significantly by applying the time series data transformations. As the magnitude of the anomaly becomes larger, rows 3 to 5 of the table, the model almost performs well with DIFF and MSD. The lowest performance of the F1-score of 56.73% is observed in the first row of the table for CNN-BiLSTM model, by applying the DIFF and MSD process on the series, the performance of the F1-score increases to 83.59%. The highest value of the F1-score of 99.96% is in the last row of the table when the size of the anomaly has the highest value and we applied the DIFF process for CNN-LSTM Autoencoder model. The using of DIFF or MSD process to stabilize statistical properties of time series has been effective in improving the performance of anomaly detection. The results show that in general, the CNN-LSTM Autoencoder model acts almost better than CNN-BiLSTM and two preprocessing methods, especially DIFF that has a significant effect on the performance of the two models.

2) Constant: The results of constant anomaly detection based on the models are shown in Table VII. When the magnitude of the anomaly is taken from the same random variable, rows 1 to 3 of the table, the model's performance increases in the duration of the anomaly. The model's performance increases at a fixed duration (i.e. d=10) by increasing the magnitude of the anomaly in rows 3 to 5. In general, the models have a very high performance in detecting constant anomalies. As can be seen in the results, the performance of the models with the DIFF or MSD process is high. The F1-score difference maximum is 8.89% for without considering preprocessing models when the anomaly magnitude is as lowest as base value + U (0, 1) with d=10.

3) Gradual Drift: The results of gradual drift anomaly detection by the models are shown in Table VIII. The proposed methods have a reasonable performance in detecting gradual drift anomalies. The minimum F1-score in the third row is 86.29% for CNN-BiLSTM with the anomaly magnitude of the base value + linear space (0, 2), d = 10, which increases to F1-score of 99.08% and 97.28% considering the DIFF and MSD process respectively. The results show that CNN-LSTM

Authorized licensed use limited to: National Chemical Laboratory. Downloaded on February 10,2026 at 13:40:57 UTC from IEEE Xplore. Restrictions apply.

Autoencoder model acts slightly better than CNN-BiLSTM with DIFF and MSD preprocessing.

4) Bias: The results of bias anomaly detection based on the models are presented in Table IX. As can be seen in this table, similar to constant anomalies in rows 1 to 3, the performance of the models increase with increasing duration of the anomaly, providing that the magnitude of the anomaly is taken from the same random variable. In rows 3 to 5 of the table, it can be seen that the performance of models increases in a fixed duration (i.e. d=10) with increasing magnitude of the anomaly. The maximum F1-score of 98.75% is seen in the third row with anomaly magnitude of the base value + U (0, 5), d = 10. The minimum F1-score is 76.71% for CNN-BiLSTM model in the fifth row with the anomaly magnitude of the base value + U(0, 1), d = 10, which increases to F1-score of 91.41% and 86.80% considering the DIFF and MSD process.

The results of tables VI to IX show that the both data preprocessing methods increase the performance of the two models, however DIFF preprocessing has a better performance than MSD. DIFF is effective in removing trends and seasonality from time series data. This makes the data more stationary, which is a fundamental assumption for many time series analysis and anomaly detection methods. Removing these patterns can simplify the data and make it easier to detect anomalies that are not related to trends or seasonality. As highlighted in Tables VI to IX, CNN-LSTM Autoencoder, D-CNN-LSTM Autoencoder and MSD-CNN-LSTM Autoencoder models increase the F1-Score compared to CNN-BiLSTM, D-CNN-BiLSTM and MSD-CNN-BiLSTM respectively. Although CNN-LSTM Autoencoder without preprocessing and with MSD preprocessing performs significantly better than the CNN-BiLSTM, with applying DIFF there is a slight difference between the two models. The reason that CNN-LSTM Autoencoder performs well is that using a combination of CNN, LSTM and Autoencoder methods provides more opportunities to discover the best and most complex features.

In tables VI to IX, time columns present the training time in seconds. Models are optimized for using real time anomaly detection. In our experiments, the parameter values and number of layers for these models are selected considering minimum training time in all types of anomalies. In addition, the parallel training of sensors is effective in reducing the training time. Since the 3 data sensors are trained independently, we considered the training time as the maximum time among the 3 sensors. We also calculated the prediction time as well as the training time for Models. The prediction time is much less compared to the time interval.

In Table X, the results of anomalies detection by Multi-Channel approach are indicated. Comparing the results of the Independent D-CNN-LSTM Autoencoder method in Tables VI to IX with the results of Multi-Channel D-CNN-LSTM Autoencoder in Table X, shows that the Multi-Channel D-CNN-LSTM Autoencoder improves the F-score by 5.39% in terms of instant anomaly with combining three data sensors. Independent D-CNN-LSTM Autoencoder method acts better by 1.46%, 1.54% and 1.48% in detecting constant, gradual drift and bias anomalies respectively. Using Multi-Channel D-CNN-LSTM Autoencoder, the average training time increases by 21.98%, 24.36%, 35.62% and 19.12% in detecting instant, constant, gradual drift and bias anomalies respectively. Although the Multi-Channel approach has a high performance in this case study, it may not be scalable by increasing the number of sensors or the feature dimension.

Table XI presents the results of comparing the performance of the proposed models in detecting single anomalies with the previous approaches, i.e. CNN-KF [1] and MSALSTM-CNN [25]. Since DIFF preprocessing has a significant impact on the performance of the models, The D-CNN-LSTM Autoencoder enhances the overall performance in terms of F1-score. And D-CNN-BiLSTM enhances the overall performance in terms of F1-Score except in some cases with a slight difference. The MSD-CNN-LSTM Autoencoder and MSD-CNN-BiLSTM enhance the overall performance in terms of F1-score in many cases. Comparison of the proposed methods specially CNN-LSTM Autoencoder with the mentioned methods shows a better performance since we took preprocessing and the advantages of autoencoder and LSTM to extract useful knowledge from autocorrelation and long memory time series. Modeling time series may not be sufficient because the residuals of these models may show autocorrelation, which sets the statistical significance structure. For this reason, in addition to the selected deep learning model, the reliability of the deep learning model in anomaly detection has been increased by considering the characteristics of time series. Low quality data has been transformed to higher quality data by data transformations, and it is suitable for fitting deep learning model. DIFF preprocessing significantly highlights abrupt changes since the results show that DIFF has the most impact on the detection of instant anomaly and the least impact on the detection of bias anomaly. Comparing the results of Table XI shows that the Independent D-CNN-LSTM Autoencoder with DIFF data transformation optimizes the F1-Score metric up to 18.12%, 5.65%, 5.2% and 3.85% respectively for instant, constant, gradual drift and bias anomalies compared to CNN-KF model proposed by the base paper [1]. The Multi-Channel D-CNN-LSTM Autoencoder with DIFF data transformation optimizes the F1-Score metric up to 23.51%, 4.19%, 3.8% and 2.37% respectively for instant, constant, gradual drift and bias anomalies compared to CNN-KF model proposed by the base paper [1]. The results show that DIFF preprocessing significantly improves detecting instant anomaly type with abrupt changes. However, DIFF can be effective for addressing constant, gradual Drift and bias anomalies even though it's more commonly associated with capturing abrupt changes.

B. Mixed Anomaly Types

Table XII presents the results of comparing the performance of CNN- LSTM Autoencoder in detecting mixed anomalies with the previous methods, KF-CNN [1] and MSALSTM-CNN [25]. The proposed models are tested using four types of anomalies instant, constant, gradual drift, and bias. Models are trained and are tested in the presence of all 4 anomaly kinds. Models show the highest performance in the type of instant anomaly. Unlike the previous models, the proposed

Authorized licensed use limited to: National Chemical Laboratory. Downloaded on February 10,2026 at 13:40:57 UTC from IEEE Xplore. Restrictions apply.

TABLE X

DETECTION PERFORMANCE OF ANOMALIES BY USING MULTI-CHANNEL APPROACH

Method			Multi-Channel CNN-LSTM Autoencoder					
Preprocessing			-		DIFF		MSD	
Anomaly Type	Anomaly Magnitude	d	F1	Time	F1	Time	F1	Time
Instant	base value + 25* N(0,0.01)	-	67.35	1.99	**90.91**	1.781	87.60	1.857
	base value + 100* N(0,0.01)	-	95.28	1.804	**98.02**	1.892	97.38	2.034
	base value + 500* N(0,0.01)	-	98.85	2.066	**99.72**	1.866	99.41	1.893
	base value + 1000* N(0,0.01)	-	99.76	1.928	**99.96**	1.913	99.80	1.883
	base value + 10000* N(0,0.01)	-	100	1.824	**100**	1.848	99.96	1.942
Constant	base value + U(0,5)	3	95.95	1.836	**98.60**	1.894	97.96	1.882
	base value + U(0,5)	5	96.54	1.804	**98.33**	1.78	98.22	2.115
	base value + U(0,5)	10	96.57	1.799	**99.10**	1.776	98.17	1.907
	base value + U(0,3)	10	93.94	2.077	**98.28**	1.958	98.13	1.807
	base value + U(0,1)	10	84.95	1.921	96.89	1.781	**97.04**	1.982
Gradual Drift	base value + linspace(0,4)	10	95.24	1.945	**99.86**	1.836	98.99	2.051
	base value + linspace(0,4)	20	97.87	1.88	99.96	1.919	**100**	1.875
	base value + linspace(0,2)	10	90.65	1.766	**97.86**	1.838	96.16	1.867
	base value + linspace(0,2)	20	96.25	1.832	**99.70**	2.27	99.02	1.903
Bias	base value + U(0,5)	3	93.24	1.822	**96.57**	1.849	96.95	1.851
	base value + U(0,5)	5	95.28	1.983	**97.02**	1.821	96.99	1.883
	base value + U(0,5)	10	95.57	1.859	**98.71**	2.122	97.46	1.854
	base value + U(0,3)	10	91.81	1.898	**95.90**	1.782	93.22	1.901
	base value + U(0,1)	10	83.22	1.812	**91.92**	1.81	87.85	2.039

TABLE XI

COMPARISON WITH BASELINE APPROACHES FOR SINGLE ANOMALY KIND

			[1]	[25]	Independent CNN-BiLSTM		Independent CNN-LSTM Autoencoder		Multi-Channel CNN-LSTM Autoencoder	
					DIFF	MSD	DIFF	MSD	DIFF	MSD
Anomaly Type	Anomaly Magnitude	d	F1	F1	F1	F1	F1	F1	F1	F1
Instant	base value + 25* N(0,0.01)	-	67.4	70.18	**83.59**	**73.33**	**85.52**	**79.98**	**90.91**	**87.60**
	base value + 100* N(0,0.01)	-	91.7	93.80	93.71	89.71	**94.65**	92.95	**98.02**	**97.38**
	base value + 500* N(0,0.01)	-	97.8	**99.02**	98.50	97.40	98.90	98.34	**99.72**	**99.41**
	base value + 1000* N(0,0.01)	-	98.4	98.68	**99.17**	97.90	**99.06**	98.61	**99.96**	**99.80**
	base value + 10000* N(0,0.01)	-	99.5	99.34	**99.57**	**99.64**	**99.96**	**99.61**	**100**	**99.96**
Constant	base value + U(0,5)	3	94.5	94.65	**99.02**	**97.54**	**99.05**	**97.99**	**98.60**	**97.96**
	base value + U(0,5)	5	95.2	95.51	**99.18**	**98.63**	**99.52**	**98.89**	**98.33**	**98.22**
	base value + U(0,5)	10	97.0	97.41	**99.59**	**99.17**	**99.59**	**99.10**	**99.10**	**98.17**
	base value + U(0,3)	10	96.2	97.15	**99.17**	**98.65**	**99.29**	**99.02**	**98.28**	**98.13**
	base value + U(0,1)	10	92.7	94.55	**98.29**	**97.08**	**98.35**	**97.37**	**96.89**	**97.04**
Gradual Drift	base value + linspace(0,4)	10	96.1	97.46	**99.77**	**98.94**	**99.91**	**99.40**	**99.86**	**98.99**
	base value + linspace(0,4)	20	97.4	97.62	**99.96**	**99.96**	**99.96**	**100**	**99.96**	**100**
	base value + linspace(0,2)	10	94.2	95.58	**99.08**	**97.28**	**99.40**	**98.01**	**97.86**	**96.16**
	base value + linspace(0,2)	20	95.9	96.03	**99.89**	**98.95**	**99.89**	**99.36**	**99.70**	**99.02**
Bias	base value + U(0,5)	3	94.2	94.78	**97.87**	**96.29**	**98.05**	**97.68**	**96.57**	**96.95**
	base value + U(0,5)	5	94.9	95.62	**98.29**	**96.59**	**98.33**	**97.75**	**97.02**	**96.99**
	base value + U(0,5)	10	96.7	97.37	**98.64**	97.31	**98.75**	**97.81**	**98.71**	**97.46**
	base value + U(0,3)	10	95.5	**96.10**	96.01	94.73	**96.52**	96.07	**95.90**	93.22
	base value + U(0,1)	10	90.0	90.57	**91.41**	86.80	**92.08**	89.67	**91.92**	87.85

TABLE XII

COMPARISON OF WITH BASELINE APPROACHES FOR MIXED ANOMALY KINDS

		[1]	[25]	Independent CNN-LSTM Autoencoder	
				DIFF	MSD
Anomaly type	sensor	F1	F1	F1	F1
Instant, 1000 * N(0,0.01)	1	77.9	78.11	**99.85**	**99.85**
	2	72.8	73.38	**99.22**	**99.07**
	3	61.2	64.44	**96.70**	**97.27**
Constant, U(0,5), d = 20	1	90.1	90.43	**98.48**	**98.44**
	2	80.7	81.10	**98.15**	**98.61**
	3	76.0	77.79	**89.71**	**93.98**
GD, linespace(0,4),d=20	1	83.3	84.30	82.82	**92.02**
	2	80.2	81.35	**92.95**	**90.71**
	3	74.7	76.33	**85.48**	**85.07**
Bias, U(0,5), d=10	1	89.3	90.45	**98.50**	**98.16**
	2	82.1	81.43	**98.00**	**99.33**
	3	75.1	76.17	**91.31**	73.54

model has a high performance in detecting the instant anomaly, even though the instant anomaly is different from the other three of anomalies in terms of magnitude and duration. For the kind of instant anomaly, models indicate the highest F1-score of 99.85% for sensor 1. For the kind of constant anomaly, the MSD-CNN-LSTM Autoencoder indicates the highest F1-score of 98.61% with a slight difference from D-CNN-LSTM Autoencoder for sensor 2. For the kind of gradual drift anomaly, the D-CNN-LSTM Autoencoder indicates the highest F1-score of 92.95% for sensor 2. For the kind of bias anomaly, the MSD-CNN-LSTM Autoencoder indicates the highest F1-score of 99.33% for sensor 2. In all kinds of anomalies, the models mostly show performance metric for sensor 3 worse than the two other speed sensors, mostly due to the big change in sequential acceleration measurements of sensor 3. However, the values of metric indicate successful performance.

V. CONCLUSION AND FUTURE WORK

This paper suggested a new framework based on the preprocessing process and autoencoder architecture to reinforce deep learning time series models for detecting a variety of anomalies in connected and automated vehicles. The proposed framework concentrated on transforming low-quality data into high-quality data that is suitable for training deep learning time series models. We utilized DIFF and MSD for the transformation to stabilize statistical properties of time

Authorized licensed use limited to: National Chemical Laboratory. Downloaded on February 10,2026 at 13:40:57 UTC from IEEE Xplore. Restrictions apply.

series. We designed a method with a CNN-based encoder and LSTM-based decoder, namely, D-CNN-LSTM Autoencoder. Another method is also designed, called, CNN-BiLSTM and is analyzed to show firstly the effectiveness of preprocessing, which is our major contribution, and secondly the CNN-LSTM Autoencoder method. Experiments demonstrate that the proposed framework improves the detection rate of anomalies successfully. Thus, it can be used as a reference to enhance the performance of anomaly detection in automated vehicles. It is worth mentioning that the proposed framework not only in the field of connected and automated vehicles but also in different scientific fields can be easily applied to detect anomalies in time series data. We intend to develop the present paper in multiple dimensions in the future. The first goal is to examine the performance of models for other time series datasets with different anomalies. The second goal is to research on more effective data imputation methods to develop anomaly detection in connected and automated vehicles.

REFERENCES

[1] F. van Wyk, Y. Wang, A. Khojandi, and N. Masoud, "Real-time sensor anomaly detection and identification in automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 1264–1276, Mar. 2020, doi: 10.1109/TITS.2019.2906038.

[2] Y. Wang, N. Masoud, and A. Khojandi, "Real-time sensor anomaly detection and recovery in connected automated vehicle sensors," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 3, pp. 1411–1421, Mar. 2021, doi: 10.1109/TITS.2020.2970295.

[3] G. Meyer and S. Deix, "Research and innovation for automated driving in Germany and Europe," in *Road Vehicle Automat.* Cham, Switzerland: Springer, 2014, pp. 71–81, doi: 10.1007/978-3-319-05990-7_7.

[4] Y. Wang, N. Masoud, and A. Khojandi, "Anomaly detection in connected and automated vehicles using an augmented state formulation," in *Proc. Forum Integr. Sustain. Transp. Syst.*, Nov. 2020, pp. 156–161, doi: 10.1109/FISTS46898.2020.9264885.

[5] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, "Connected vehicles: Solutions and challenges," *IEEE Internet Things J.*, vol. 1, no. 4, pp. 289–299, Aug. 2014, doi: 10.1109/JIOT.2014.2327587.

[6] L. Hobert, A. Festag, I. Llatser, L. Altomare, F. Visintainer, and A. Kovacs, "Enhancements of V2X communication in support of cooperative autonomous driving," *IEEE Commun. Mag.*, vol. 53, no. 12, pp. 64–70, Dec. 2015, doi: 10.1109/MCOM.2015.7355568.

[7] K. Dar, M. Bakhouya, J. Gaber, M. Wack, and P. Lorenz, "Wireless communication technologies for ITS applications topics in automotive networking," *IEEE Commun. Mag.*, vol. 48, no. 5, pp. 156–162, May 2010, doi: 10.1109/MCOM.2010.5458377.

[8] *Connected Vehicles and Cybersecurity*. Accessed: Nov. 12, 2021. [Online]. Available: http://www.its.dot.gov

[9] H. Kim, J. Park, K. Min, and K. Huh, "Anomaly monitoring framework in lane detection with a generative adversarial network," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 3, pp. 1603–1615, Mar. 2021, doi: 10.1109/TITS.2020.2973398.

[10] J. Liu, J. Gu, H. Li, and K. H. Carlson, "Machine learning and transport simulations for groundwater anomaly detection," *J. Comput. Appl. Math.*, vol. 380, Dec. 2020, Art. no. 112982, doi: 10.1016/j.cam.2020.112982.

[11] M. Kim, E. Ou, P.-L. Loh, T. Allen, R. Agasie, and K. Liu, "RNN-based online anomaly detection in nuclear reactors for highly imbalanced datasets with uncertainty," *Nucl. Eng. Des.*, vol. 364, Aug. 2020, Art. no. 110699, doi: 10.1016/j.nucengdes.2020.110699.

[12] S. Müller, J. Lancrenon, C. Harpes, Y. Le Traon, S. Gombault, and J.-M. Bonnin, "A training-resistant anomaly detection system," *Comput. Secur.*, vol. 76, pp. 1–11, Jul. 2018, doi: 10.1016/j.cose.2018.02.015.

[13] L. Meneghetti, M. Terzi, S. Del Favero, G. A. Susto, and C. Cobelli, "Data-driven anomaly recognition for unsupervised model-free fault detection in artificial pancreas," *IEEE Trans. Control Syst. Technol.*, vol. 28, no. 1, pp. 33–47, Jan. 2020, doi: 10.1109/TCST.2018.2885963.

[14] D. Zambon, C. Alippi, and L. Livi, "Concept drift and anomaly detection in graph streams," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5592–5605, Nov. 2018, doi: 10.1109/TNNLS.2018.2804443.

[15] F. C. Bauer, D. R. Muir, and G. Indiveri, "Real-time ultra-low power ECG anomaly detection using an event-driven neuromorphic processor," *IEEE Trans. Biomed. Circuits Syst.*, vol. 13, no. 6, pp. 1575–1582, Dec. 2019, doi: 10.1109/TBCAS.2019.2953001.

[16] M. Zhou, Y. Wang, A. K. Srivastava, Y. Wu, and P. Banerjee, "Ensemble-based algorithm for synchrophasor data anomaly detection," *IEEE Trans. Smart Grid*, vol. 10, no. 3, pp. 2979–2988, May 2019, doi: 10.1109/TSG.2018.2816027.

[17] U. Adhikari, T. H. Morris, and S. Pan, "Applying Hoeffding adaptive trees for real-time cyber-power event and intrusion classification," *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 4049–4060, Sep. 2018, doi: 10.1109/TSG.2017.2647778.

[18] S. Basumallik, R. Ma, and S. Eftekharnejad, "Packet-data anomaly detection in PMU-based state estimator using convolutional neural network," *Int. J. Electr. Power Energy Syst.*, vol. 107, pp. 690–702, May 2019, doi: 10.1016/j.ijepes.2018.11.013.

[19] Z. Zhong, A. Y. Sun, Q. Yang, and Q. Ouyang, "A deep learning approach to anomaly detection in geological carbon sequestration sites using pressure measurements," *J. Hydrol.*, vol. 573, pp. 885–894, Jun. 2019, doi: 10.1016/j.jhydrol.2019.04.015.

[20] A. Y. Sun, Z. Zhong, H. Jeong, and Q. Yang, "Building complex event processing capability for intelligent environmental monitoring," *Environ. Model. Softw.*, vol. 116, pp. 1–6, Jun. 2019, doi: 10.1016/j.envsoft.2019.02.015.

[21] R. Nawaratne, D. Alahakoon, D. De Silva, and X. Yu, "Spatiotemporal anomaly detection using deep learning for real-time video surveillance," *IEEE Trans. Ind. Informat.*, vol. 16, no. 1, pp. 393–402, Jan. 2020, doi: 10.1109/TII.2019.2938527.

[22] N. Cao, C. Lin, Q. Zhu, Y. Lin, X. Teng, and X. Wen, "Voila: Visual anomaly detection and monitoring with streaming spatiotemporal data," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 1, pp. 23–33, Jan. 2018, doi: 10.1109/TVCG.2017.2744419.

[23] D. Sovilj, P. Budnarain, S. Sanner, G. Salmon, and M. Rao, "A comparative evaluation of unsupervised deep architectures for intrusion detection in sequential data streams," *Expert Syst. Appl.*, vol. 159, Nov. 2020, Art. no. 113577, doi: 10.1016/j.eswa.2020.113577.

[24] Q. Chen, R. Luley, Q. Wu, M. Bishop, R. W. Linderman, and Q. Qiu, "AnRAD: A neuromorphic anomaly detection framework for massive concurrent data streams," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1622–1636, May 2018, doi: 10.1109/TNNLS.2017.2676110.

[25] A. R. Javed, M. Usman, S. U. Rehman, M. U. Khan, and M. S. Haghighi, "Anomaly detection in automated vehicles using multistage attention-based convolutional neural network," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4291–4300, Jul. 2021, doi: 10.1109/TITS.2020.3025875.

[26] F. Alotibi and M. Abdelhakim, "Anomaly detection for cooperative adaptive cruise control in autonomous vehicles using statistical learning and kinematic model," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 6, pp. 3468–3478, Jun. 2021, doi: 10.1109/TITS.2020.2983392.

[27] S. Khan, C. F. Liew, T. Yairi, and R. McWilliam, "Unsupervised anomaly detection in unmanned aerial vehicles," *Appl. Soft Comput.*, vol. 83, Oct. 2019, Art. no. 105650, doi: 10.1016/j.asoc.2019.105650.

[28] C. Ryan, F. Murphy, and M. Mullins, "End-to-end autonomous driving risk analysis: A behavioural anomaly detection approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 3, pp. 1650–1662, Mar. 2021, doi: 10.1109/TITS.2020.2975043.

[29] M. Levi, Y. Allouche, and A. Kontorovich, "Advanced analytics for connected car cybersecurity," in *Proc. IEEE 87th Veh. Technol. Conf.*, Jun. 2018, pp. 1–7, doi: 10.1109/VTCSpring.2018.8417690.

[30] H. M. Song, J. Woo, and H. K. Kim, "In-vehicle network intrusion detection using deep convolutional neural network," *Veh. Commun.*, vol. 21, Jan. 2020, Art. no. 100198, doi: 10.1016/j.vehcom.2019.100198.

[31] J. Liu and J.-M. Park, "'Seeing is not always Believing': Detecting perception error attacks against autonomous vehicles," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 5, pp. 2209–2223, Sep. 2021, doi: 10.1109/TDSC.2021.3078111.

[32] P. Duda, L. Rutkowski, M. Jaworski, and D. Rutkowska, "On the Parzen kernel-based probability density function learning procedures over time-varying streaming data with applications to pattern classification," *IEEE Trans. Cybern.*, vol. 50, no. 4, pp. 1683–1696, Apr. 2020, doi: 10.1109/TCYB.2018.2877611.

[33] J. De La Ree, V. Centeno, J. S. Thorp, and A. G. Phadke, "Synchronized phasor measurement applications in power systems," *IEEE Trans. Smart Grid*, vol. 1, no. 1, pp. 20–27, Jun. 2010, doi: 10.1109/TSG.2010.2044815.

Authorized licensed use limited to: National Chemical Laboratory. Downloaded on February 10,2026 at 13:40:57 UTC from IEEE Xplore. Restrictions apply.

[34] C. G. Bezerra, B. S. J. Costa, L. A. Guedes, and P. P. Angelov, "An evolving approach to data streams clustering based on typicality and eccentricity data analytics," *Inf. Sci.*, vol. 518, pp. 13–28, May 2020, doi: 10.1016/j.ins.2019.12.022.

[35] K. Gokcesu, M. M. Neyshabouri, H. Gokcesu, and S. S. Kozat, "Sequential outlier detection based on incremental decision trees," *IEEE Trans. Signal Process.*, vol. 67, no. 4, pp. 993–1005, Feb. 2019, doi: 10.1109/TSP.2018.2887406.

[36] F. M. J. Willems, "The context-tree weighting method: Extensions," *IEEE Trans. Inf. Theory*, vol. 44, no. 2, pp. 792–798, Mar. 1998, doi: 10.1109/18.661523.

[37] K. Manandhar, X. Cao, F. Hu, and Y. Liu, "Detection of faults and attacks including false data injection attack in smart grid using Kalman filter," *IEEE Trans. Control Netw. Syst.*, vol. 1, no. 4, pp. 370–379, Dec. 2014, doi: 10.1109/TCNS.2014.2357531.

[38] K. Salahshoor, M. Mosallaei, and M. Bayat, "Centralized and decentralized process and sensor fault monitoring using data fusion based on adaptive extended Kalman filter algorithm," *Measurement*, vol. 41, no. 10, pp. 1059–1076, Dec. 2008, doi: 10.1016/j.measurement.2008.02.009.

[39] J. Ashraf, A. D. Bakhshi, N. Moustafa, H. Khurshid, A. Javed, and A. Beheshti, "Novel deep learning-enabled LSTM autoencoder architecture for discovering anomalous events from intelligent transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4507–4518, Jul. 2021, doi: 10.1109/TITS.2020.3017882.

[40] K. Agrawal, T. Alladi, A. Agrawal, V. Chamola, and A. Benslimane, "NovelADS: A novel anomaly detection system for intra-vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 11, pp. 22596–22606, Nov. 2022, doi: 10.1109/TITS.2022.3146024.

[41] B. Du et al., "Deep irregular convolutional residual LSTM for urban traffic passenger flows prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 972–985, Mar. 2020, doi: 10.1109/TITS.2019.2900481.

[42] P. Christiansen, L. N. Nielsen, K. A. Steen, R. N. Jørgensen, and H. Karstoft, "DeepAnomaly: Combining background subtraction and deep learning for detecting obstacles and anomalies in an agricultural field," *Sensors*, vol. 16, no. 11, pp. 1904–1924, 2016, doi: 10.3390/s16111904.

[43] Y. Karadayı, M. N. Aydin, and A. S. Öğrenci, "A hybrid deep learning framework for unsupervised anomaly detection in multivariate spatio-temporal data," *Appl. Sci.*, vol. 10, no. 15, p. 5191, Jul. 2020, doi: 10.3390/app10155191.

[44] A. B. Sharma, L. Golubchik, and R. Govindan, "Sensor faults: Detection methods and prevalence in real-world datasets," *ACM Trans. Sensor Netw.*, vol. 6, no. 3, pp. 1–39, Jun. 2010, doi: 10.1145/1754414.1754419.

[45] J. Petit and S. E. Shladover, "Potential cyberattacks on automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 546–556, Apr. 2015, doi: 10.1109/TITS.2014.2342271.

[46] G. Marcus, "Deep learning: A critical appraisal," 2018, *arXiv:1801.00631*.

[47] R. Salles, K. Belloze, F. Porto, P. H. Gonzalez, and E. Ogasawara, "Nonstationary time series transformation methods: An experimental review," *Knowl.-Based Syst.*, vol. 164, pp. 274–291, Jan. 2019, doi: 10.1016/j.knosys.2018.10.041.

[48] N. Bokde, A. Feijóo, and K. Kulat, "Analysis of differencing and decomposition preprocessing methods for wind speed prediction," *Appl. Soft Comput.*, vol. 71, pp. 926–938, Oct. 2018, doi: 10.1016/j.asoc.2018.07.041.

[49] E. Ogasawara, L. C. Martinez, D. de Oliveira, G. Zimbrão, G. L. Pappa, and M. Mattoso, "Adaptive normalization: A novel data normalization approach for non-stationary time series," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2010, pp. 1–8, doi: 10.1109/IJCNN.2010.5596746.

[50] I. E. Livieris, S. Stavroyiannis, L. Iliadis, and P. Pintelas, "Smoothing and stationarity enforcement framework for deep learning time-series forecasting," *Neural Comput. Appl.*, vol. 33, no. 20, pp. 14021–14035, Oct. 2021, doi: 10.1007/s00521-021-06043-1.

[51] L. Seymour, P. J. Brockwell, and R. A. Davis, *Introduction to Time Series and Forecasting*, vol. 92. New York, NY, USA: Springer, 1997.

[52] R. Ke, W. Li, Z. Cui, and Y. Wang, "Two-stream multi-channel convolutional neural network for multi-lane traffic speed prediction considering traffic volume impact," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2674, no. 4, pp. 459–470, Apr. 2020, doi: 10.1177/0361198120911052.

[53] S. Kim, W. Jo, and T. Shon, "APAD: Autoencoder-based payload anomaly detection for industrial IoE," *Appl. Soft Comput.*, vol. 88, Mar. 2020, Art. no. 106017, doi: 10.1016/j.asoc.2019.106017.

Fatemeh Khanmohammadi received the B.Sc. degree in computer engineering from the University of Qom, Iran, and the master's degree in computer engineering from Alzahra University, Iran, in October 2021. Her current research interests include, but are not limited to, cybersecurity analysis, artificial intelligence, the Internet of Things (IoT), and machine learning.

Reza Azmi received the B.Sc. degree in electrical engineering from the University of Amirkabir, Iran, and the M.Sc. and Ph.D. degrees in electrical engineering from the University of Tarbiat Modares, Iran. He is currently an Associate Professor with the Department of Computer Engineering, University of Alzahra, Iran. He is the Founder of Tooba Tech Company and Co-Founder of ToobaMode a Fashion and AI-based search engine. He is also the Founder of many specialized laboratories such as Alzahra University's Digital and Intelligent Transformation Center and Deep Image Processing Laboratory.

Authorized licensed use limited to: National Chemical Laboratory. Downloaded on February 10,2026 at 13:40:57 UTC from IEEE Xplore. Restrictions apply.