MTH-IDS: A Multitiered Hybrid Intrusion Detection System for Internet of Vehicles

Li Yang, *Member, IEEE*, Abdallah Moubayed, *Member, IEEE*, and Abdallah Shami, *Senior Member, IEEE*

***Abstract*—Modern vehicles, including connected vehicles and autonomous vehicles, nowadays involve many electronic control units connected through intravehicle networks (IVNs) to implement various functionalities and perform actions. Modern vehicles are also connected to external networks through vehicle-to-everything technologies, enabling their communications with other vehicles, infrastructures, and smart devices. However, the improving functionality and connectivity of modern vehicles also increase their vulnerabilities to cyber-attacks targeting both intravehicle and external networks due to the large attack surfaces. To secure vehicular networks, many researchers have focused on developing intrusion detection systems (IDSs) that capitalize on machine learning methods to detect malicious cyber-attacks. In this article, the vulnerabilities of intravehicle and external networks are discussed, and a multitiered hybrid IDS that incorporates a signature-based IDS and an anomaly-based IDS is proposed to detect both known and unknown attacks on vehicular networks. Experimental results illustrate that the proposed system can detect various types of known attacks with 99.99% accuracy on the CAN-intrusion-dataset representing the IVN data and 99.88% accuracy on the CICIDS2017 data set illustrating the external vehicular network data. For the zero-day attack detection, the proposed system achieves high F1-scores of 0.963 and 0.800 on the above two data sets, respectively. The average processing time of each data packet on a vehicle-level machine is less than 0.6 ms, which shows the feasibility of implementing the proposed system in real-time vehicle systems. This emphasizes the effectiveness and efficiency of the proposed IDS.**

***Index Terms*—Anomaly detection, Bayesian optimization (BO), controller area network (CAN) bus, Internet of Vehicles (IoV), intrusion detection system (IDS), zero-day attacks.**

I. Introduction

With the increasing research and rapid development of the Internet of Vehicles (IoV) technology, connected vehicles (CVs), and autonomous vehicles (AVs) are becoming increasingly popular in the modern world [1]. IoV serves as a primary vehicular communication framework that enables reliable communications between vehicles and other IoV entities, such as infrastructures, pedestrians, and smart devices [1].

IoV consists mainly of intravehicle networks (IVNs) and external vehicular networks [1]. IVNs involve an increasing number of electronic control units (ECUs) to adopt various functionalities [2]. All ECUs in a vehicle are connected by a controller area network (CAN) bus to transmit messages and perform actions [3]. On the other hand, external networks connect modern vehicles to the outer environment by vehicle-to-everything (V2X) technologies. V2X technology allows modern vehicles to communicate with other vehicles, roadside infrastructures, and road users [4], [5].

However, with the increasing level of connectivity and complexity of modern vehicles, their security risks have become a significant concern. Cyber threats may decrease the stability and robustness of IoV, as well as cause vehicle unavailability or traffic accidents. A real-life example can be found in [6]: two attackers compromised and fooled a jeep car into performing dangerous actions, including turning the steering wheel and activating the parking brake at highway speeds, causing severe accidents. In IVNs, CANs are mainly vulnerable to message injection attacks due to their broadcast communication strategy and the lack of authentication [4]. In external networks of IoV, vehicle systems are exposed to various common cyber-attacks, such as Denial-of-Service (DoS), sniffing, and global positioning system (GPS) spoofing attacks [7]. This is because, in large external vehicular networks comprising various types of networks and entities, every node is a potential entry point for cyber-attacks.

Many traditional security mechanisms, such as certain authentication and cryptographic techniques, are unsuitable for IVNs because they are not supported in CANs or may violate timing constraints of CAN communications [8]. Thus, intrusion detection systems (IDSs) have become an essential component in modern IoV to identify malicious threats on vehicular networks [9]. IDSs are often incorporated into external networks as an essential component of the defense system to identify malicious attacks that can breach firewalls and authentication mechanisms. Although many previous works have made some success developing IDSs, intrusion detection is still a challenging problem due to the high volume of network traffic data, numerous available network features, and various cyber-attack patterns [7].

Machine learning (ML) and data mining algorithms have been recognized as effective models to design IDSs [10]. In this article, a multitiered hybrid IDS (MTH-IDS) is proposed to efficiently identify known and zero-day cyber-attacks on both intravehicle and external networks using multiple ML algorithms. The proposed MTH-IDS framework consists of two traditional ML stages (data preprocessing and feature engineering) and four tiers of learning models.

Manuscript received November 12, 2020; revised March 4, 2021, April 5, 2021, and May 15, 2021; accepted May 24, 2021. Date of publication May 28, 2021; date of current version December 23, 2021. *(Corresponding author: Li Yang.)*

The authors are with the Department of Electrical and Computer Engineering, Western University, London, ON N6A 3K7, Canada (e-mail: lyang339@uwo.ca; amoubaye@uwo.ca; abdallah.shami@uwo.ca).

Digital Object Identifier 10.1109/JIOT.2021.3084796

1) Four tree-based supervised learners—decision tree (DT), random forest (RF), extra trees (ETs), and extreme gradient boosting (XGBoost)—used as multiclass classifiers for known attack detection.
2) A stacking ensemble model and a Bayesian optimization with tree Parzen estimator (BO-TPE) method for supervised learner optimization.
3) A cluster labeling (CL) *k*-means used as an unsupervised learner for zero-day attack detection.
4) Two biased classifiers and a Bayesian optimization with Gaussian process (BO-GP) method for unsupervised learner optimization.

Therefore, tiers 1 and 3 of the MTH-IDS are designed for basic known and unknown attack detection functionalities, respectively, while tiers 2 and 4 are designed to optimize the base learners in tiers 1 and 3 for model performance enhancement. A comprehensive and robust IDS with both known and unknown attack detection functionalities can be obtained after the model learning and optimization procedures. Additionally, the quality of the used data sets can be improved by data preprocessing and feature engineering procedures to achieve more accurate attack detection.

The performance of the proposed MTH-IDS is evaluated on two public network data sets, the CAN-intrusion-dataset [9] and the CICIDS2017 data set [11], representing the intravehicle and external network traffic data, respectively. The model's feasibility, effectiveness, and efficiency are evaluated using various metrics, including accuracy, detection rates (DRs), false alarm rates (FARs), F1-scores, and model execution time.

To the best of our knowledge, no previous work proposed such a hybrid IDS that optimizes learning models to accurately detect existing and zero-day attack patterns on both intravehicle and external vehicular networks.

The main contributions of this article are as follows.

1) It proposes a novel MTH-IDS that can accurately detect the various surveyed types of cyber-attacks launched on both intravehicle and external vehicular networks.
2) It proposes a novel feature engineering model based on information gain (IG), fast correlation-based filter (FCBF), and kernel principal component analysis (KPCA) algorithms.
3) It proposes a novel anomaly-based IDS based on CL-k-means and biased classifiers to detect zero-day attacks.
4) It discusses the use of Bayesian optimization (BO) techniques to automatically tune the parameters of each tier in the proposed IDS for model optimization.
5) It evaluates the performance and overall efficiency of the proposed model on two state-of-the-art data sets, CAN-intrusion-dataset and CICIDS2017, and discusses its feasibility in real-world IoV devices.

The remainder of this article is organized as follows. Section II discusses the related work. Section III presents the vulnerabilities of intravehicle and external vehicular networks, as well as the attack scenarios and IDS deployment. In Section IV, all the tiers and algorithms in the proposed MTH-IDS are discussed in detail. Section V presents and discusses the experimental results. Section VI concludes this article.

II. Related Work

A. CAN Intrusion Detection

The research on IDS development for IoV and CVs has been considered critical in recent years. Many research works have a focus on detecting attacks on CAN-based IVNs. Alshammari *et al.* [12] proposed an intrusion classification model to identify CAN intrusions on in-vehicle networks utilizing support vector machine (SVM) and *k*-nearest neighbors (KNNs) algorithms. Barletta *et al.* [13] proposed a distance-based IDS for CAN intrusion detection using an X–Y fused Kohonen network with the *k*-means algorithm (XYF-K). The proposed method shows high accuracy on the CAN-intrusion-dataset, but its main limitation is the high computational complexity. Olufowobi *et al.* [14] proposed a specification-based real-time IDS named SAIDuCANT to detect the in-vehicle network attacks. The effectiveness of SAIDuCANT is evaluated on a synthetic data set and the CAN-intrusion-dataset. Olufowobi *et al.* [15] proposed an anomaly-based IDS for CAN attack detection using the adaptive cumulative sum (CUSUM) algorithm. This technique can effectively detect intrusions with low delay based on statistical changes. Lee *et al.* [16] proposed an offset ratio and time interval-based IDS (OTIDS) to detect CAN attacks in in-vehicle networks. They also created a CAN data set by simulating DoS, fuzzy, and impersonation attacks for IDS evaluation.

Deep learning (DL) methods are also widely used for IVN IDS development. Lokman *et al.* [17] proposed an unsupervised DL-based anomaly detection model named stacked sparse autoencoders (SSAEs) to discover anomalies in CAN-bus data for IVN security enhancement. Song *et al.* [18] proposed a deep convolutional neural network (DCNN) method named Reduced Inception-ResNet to detect intravehicle attacks and achieve high detection performance on the CAN-intrusion-dataset. Ashraf *et al.* [19] proposed a DL-based IDS for IoV using a long-short term memory (LSTM) autoencoder algorithm. The effectiveness of the proposed model is evaluated on the CAN-intrusion-dataset and UNSW-NB15 data set, representing in-vehicle network and external network data sets, respectively. DL methods can often achieve high accuracy, but they are computationally expensive due to high model complexity.

B. External Network Intrusion Detection

Intrusion detection in IoV or external vehicular networks has also attracted significant attention. Alheeti and Mc Donald-Maier [20] proposed an intelligent IDS using backpropagation neural networks to detect DoS attacks in external vehicular networks using the Kyoto 2006+ data set, but did not consider other attack types. Rosay *et al.* [21] proposed a multilayer perceptron (MLP)-based network IDS for cyber-attack detection in IoT and CVs. The proposed model has been implemented on an automotive microprocessor, and its performance is evaluated on the two variants of the CICIDS2017 data set. Aswal *et al.* [22] analyzed the applicability of six classical ML algorithms for Bot attack detection on IoV. They used the Bot attack files in the CICIDS2017 data set to represent the Botnets in vehicular networks, but they did not consider other attacks.

Aloqaily *et al.* [23] proposed a network IDS for IoV and CVs using deep belief network (DBN) and DT algorithms. This method shows high accuracy on the NSL-KDD data set. Gao *et al.* [24] proposed a distributed network IDS for distributed DoS (DDoS) attack detection in vehicular networks and V2X systems. Two general network benchmark data sets, the NSL-KDD and UNSW-NB15 data sets, are used to present the vehicular network data sets and evaluate the IDS. Schmidt *et al.* [25] proposed a spline-based IDS for vehicular networks using the knot flow classification (KFC) method and used the NSL-KDD data set to represent vehicle networks for model evaluation.

Several other research works also pay attention to the IDS development of general networks and use benchmark data sets for method evaluation. Min *et al.* [26] proposed a semisupervised learning model, named SU-IDS, by combining the autoencoder algorithm with *k*-means to detect cyber-attacks using the NSL-KDD and CICIDS2017 data sets. Yao *et al.* [27] proposed a DL model named spatial–temporal DL on communication graphs (STDeepGraph) by combining the convolutional neural network (CNN) and long short-term memory (LSTM) methods. The performance of STDeepGraph is evaluated on the UNSW-NB15 and CICIDS2017 data sets. Injadat *et al.* [28] proposed a novel multistage optimized ML-based IDS for network attack detection and evaluated the model's performance on the CICIDS2017 and UNSW-NB15 data sets. The ML models used in this article are optimized by hyperparameter optimization (HPO) methods.

C. Literature Comparison

Although various studies about vehicular network IDS development have been published, most of them are only designed for known attack detection on either intravehicle [12]–[18] or external networks [20]–[26]. Additionally, several papers [20], [22], [24] only consider a specific type of attack, such as Botnets or DoS attacks. However, in real-world applications, both intravehicle and external networks are vulnerable to various types of attacks with both existing and new patterns. The IDS proposed in [19] is the only research that considers both CAN bus and external networks, and the IDS proposed in [20] is the only technique that can detect both known and unknown attacks. Thus, there still should an IDS designed for the detection of both known and zero-day attacks on both intravehicle and external vehicular networks. Our proposed IDS aims to achieve this.

On the other hand, for the deployment of IDSs in real-world vehicle systems, vehicle-level model testing and real-time analysis should be performed to validate the feasibility of the IDSs. However, only five papers [14], [16], [18], [21], and [23] did vehicle-level testing or real-time analysis, so the feasibility of other techniques in real-world IoV is not proven. Therefore, our proposed IDS has been evaluated in a vehicle-level machine to verify whether it meets the real-time requirements of vehicular networks.

An effective IDS should achieve a high DR and a low FAR. Moreover, to meet the real-time requirements of IoV, an IDS should have low computational complexity and high efficiency.

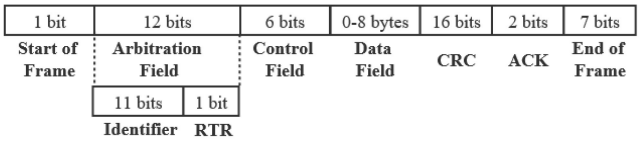

Fig. 1. CAN packet structure.

Thus, three important procedures, including data sampling, feature engineering, and model optimization, are implemented in our proposed MTH-IDS to improve the efficiency and accuracy of IoV attack detection. The details of how these three procedures can improve the model's performance are provided in Sections IV-B–IV-D. However, only six existing techniques [12], [19]–[22], [28] performed feature engineering and only one research work [28] implemented model optimization. The use of efficiency and accuracy enhancement techniques enables our proposed MTH-IDS to outperform most existing research works in terms of DR, FAR, and execution speed.

To summarize, the main functionalities and components of the proposed MTH-IDS are listed in Table I. Table I also clearly compares 17 existing literature with the proposed MTH-IDS based on the important functionalities that an effective and efficient vehicular network IDS should have.

III. Vehicular Networks, Vulnerabilities, and IDS Deployment

A. Vulnerabilities of Intravehicle Networks

Modern vehicles often contain 70–100 ECUs that are in-vehicle components used to enable various functionalities [2]. CAN [9] is a bus communication protocol that defines an international standard for efficient and reliable intravehicle communications among ECUs. A CAN-bus is built based on differential signaling and comprises a pair of channels, CAN-High and CAN-Low, representing the two signals, 1 and 0, respectively, [29]. CAN is the most common type of IVN due to its low cost and complexity, high reliability, noise-resistance, and fault-tolerance properties [7], [9]. However, CAN is vulnerable to various cyber threats due to its broadcast transmission strategy, lack of authentication and encryption, and unsecured priority scheme [3].

CAN messages, or packets, are transmitted via CAN-bus. The data frame is the most important type of CAN packet used to transmit user data [30]. Fig. 1 shows the structure of a CAN packet, which consists of seven fields [4]: 1) start of frame; 2) arbitration field; 3) control field; 4) data field; 5) cyclic redundancy code (CRC) field; 6) acknowledge (ACK) field; and 7) end of frame. Among all fields, the data field with the size of 0–8 bytes is the most important and vulnerable one, since it contains the actual transmitted data that determines the node actions [9]. An attacker can intrude or take control of a vehicle by injecting malicious messages into the data field of CAN packets, resulting in compromised nodes or vehicles; so-called message injection attacks.

Message injection attacks are the primary type of intravehicle attack and can be further classified as DoS attacks, fuzzy attacks, and spoofing attacks by their objectives [9]. In DoS

TABLE I
COMPARISON OF RECENT INTRUSION DETECTION TECHNIQUES FOR IOV

Paper	In-vehicle Network Attack Detection	External Network Attack Detection	Multiple Types of Known Attack Detection	Zero-Day Attack Detection	Vehicle-Level Model Testing or Real-time Analysis	Data Pre-Processing and Sampling	Feature Engineering	Model Optimization
Alshammari *et al.* [12]	✓		✓				✓	
Barletta *et al.* [13]	✓		✓					
Olufowobi *et al.* [14]	✓		✓		✓			
Olufowobi *et al.*[15]	✓		✓					
Lee *et al.* [16]	✓		✓		✓			
Lokman *et al.* [17]	✓		✓					
Song *et al.* [18]	✓		✓		✓			
Ashraf *et al.* [19]	✓	✓	✓				✓	
Alheeti *et al.* [20]		✓		✓			✓	
Rosay *et al.* [21]		✓	✓		✓		✓	
Aswal *et al.* [22]		✓					✓	
Aloqaily *et al.* [23]		✓	✓		✓			
Gao *et al.* [24]		✓						
Schmidt *et al.* [25]		✓	✓					
Min *et al.* [26]		✓	✓					
Yao *et al.* [27]		✓	✓					
Injadat *et al.* [28]		✓	✓				✓	✓
Proposed MTH-IDS	✓	✓	✓	✓	✓	✓	✓	✓

attacks, a CAN is flooded with massive high-priority messages to cause latencies or unavailability of other legitimate messages. Similarly, fuzzy attacks can be launched by injecting arbitrary messages with randomly spoofed identifiers (IDs) or packets, causing compromised vehicles to exhibit unintended behaviors, such as sudden braking or gear shift changes. Spoofing or impersonation attacks, such as gear spoofing and revolutions per minute (RPM) spoofing attacks, are launched by injecting messages with certain CAN IDs to masquerade as legitimate users and take control of the vehicles.

B. Vulnerabilities of External Vehicular Networks

In a similar fashion, V2X technology enables interactions and communications between vehicles and other IoV entities, including pedestrians, infrastructures, smart devices, and network systems [4], [31]. With the increasing connectivity of modern IoV, external vehicular networks are becoming large networks that involve various other networks and devices. Thus, external vehicular networks are vulnerable to various general cyber threats because each vehicle or device is a potential entry point for intrusions. Typical attacks in IoV include DoS, GPS spoofing, jamming, sniffing, brute-force, Botnets, infiltration, and Web attacks [11], [32]. The description and IoV scenarios of these common external vehicular network attacks are summarized in Table II.

C. Attack Scenarios and IDS Deployment

The attack scenarios and general architecture of a vehicle protected by IDS are shown in Fig. 2. As the physical interface for ECU communications, the onboard diagnostics II (OBD-II) interface is often exploited by internal attacks to inject malicious CAN messages into the vehicle systems, therefore, taking control of the CAN nodes to perform malicious actions, like sudden braking. On the other hand, the external attackers can launch the cyber-attacks listed in Table II through various wireless interfaces, including WiFi, cellular network, and Bluetooth [19].

TABLE II
COMMON ATTACK TYPES ON EXTERNAL VEHICULAR NETWORKS

Attack Type	Description and IoV Scenarios
DoS [32]	Send a large number of requests to exhaust the compromised nodes' resources, causing vehicle unavailability or accidents.
GPS Spoofing [32]	Masquerade as authorized IoV users to provide a node with false information, like false geographic information, therefore causing fake evidence, event delay, or property losses.
Jamming [32]	Jam signals to prevent legitimate IoV devices from communicating with connected vehicles.
Sniffing [32]	Capture vehicular network packets to steal confidential or sensitive information of vehicles, users, or enterprises.
Brute-force [11]	Crack passwords in vehicle systems to take control of vehicles or machines and perform malicious actions.
Botnets [11]	Infect multiple connected vehicles and IoV devices with Bot viruses to breach them and launch other attacks.
Infiltration [11]	Traverse the compromised vehicle systems and create a backdoor for future attacks.
Web Attack [11]	Hack IoV servers or web interfaces of connected vehicles to gain confidential information or perform malicious actions.

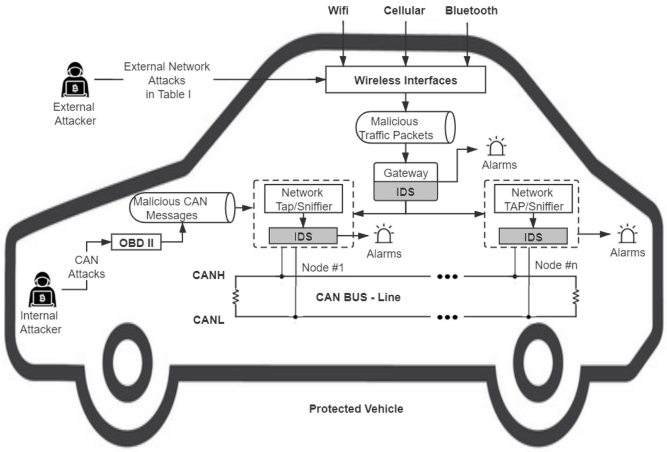

Fig. 2. Proposed IDS-protected vehicle architecture.

To secure IoV, the proposed IDS can be placed in different locations in vehicular networks. In IVNs, the IDS can be deployed on top of the CAN-bus to detect malicious messages [33]. Since every message is broadcasted to all nodes, it will also be transmitted through the IDS when its signal

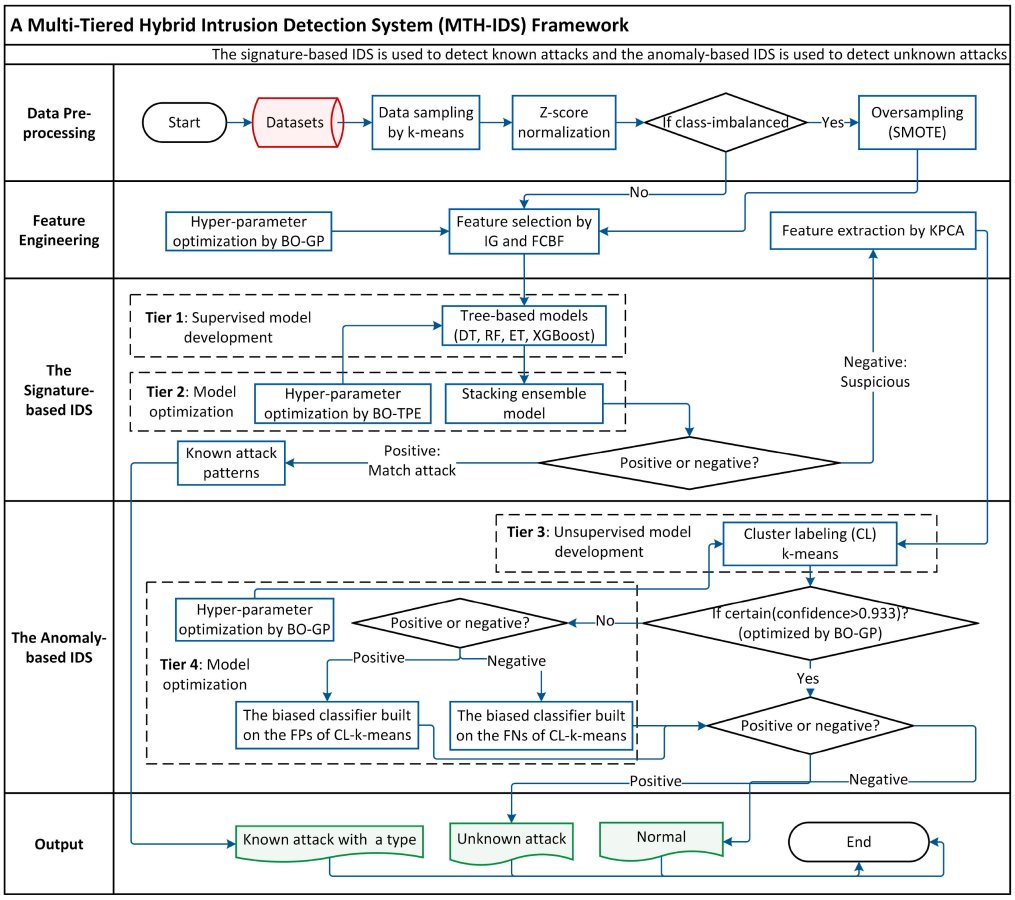

Fig. 3. Framework of the proposed MTH-IDS.

changes from CAN-High to CAN-Low. The IDS will monitor the CAN packets and identify potential intrusions. If an attack is detected, alarms will be triggered on every node.

On the other hand, the central gateways are common network devices to incorporate IDSs, so the proposed IDS can be placed inside the gateway to monitor the external network traffic [34]. Thus, abnormal network traffic can be detected when it is passed through the gateways in external vehicular networks. The potential deployment of the proposed IDS is also shown in Fig. 2.

To protect the vehicle from being breached, all packets or messages transmitted in the protected vehicular network are captured by packet taps or sniffers (e.g., NetFlow), and then analyzed by the proposed IDS before being forwarded to the protected vehicle [35]. For example, if an attacker is launching a DoS attack by sending a large volume of malicious traffic to a vehicle system, the malicious traffic will be detected by the proposed IDS by processing the traffic data captured by the sniffer; alarms will then be generated, and the attacker's access will be denied, therefore, protecting the vehicle from being breached [36]–[38].

D. Real-Time Requirements of Vehicle Systems

The standardized vehicular communications specify the performance requirements of vehicle safety services based on two primary metrics: 1) packet delivery ratio (PDR) and 2) latency [39]. For IDS development, the latency metric should be considered to meet the real-time requirements. Latency indicates the time needed to transmit a packet from its source to its destination. According to the U.S. Department of Transportation, the highest priority vehicle safety services, such as collision and attack warnings, should have a latency of 10 to 100 ms at the utmost [39]. On the other hand, for autonomous or cooperative driving, the V2X traffic safety applications require a stringent latency requirement of 10–20 ms [38], [40]. Thus, for a vehicle-level IDS, the time needed to process each network packet is required to be less than 10 ms to meet the real-time or latency requirements.

IV. Proposed MTH-IDS Framework

A. System Architecture

The purpose of this work is to develop an IDS that can protect both intravehicle and external networks from being breached by the various common attacks presented in Sections III-A and III-B. In this article, a novel MTH-IDS is proposed to detect both known and unknown cyber-attacks on vehicular networks with optimal performance. Fig. 3 demonstrates the architecture of the proposed system, comprising four main stages: 1) data preprocessing; 2) feature engineering; 3) a signature-based IDS; and 4) an anomaly-based IDS.

First, intravehicle and external network traffic data sets are collected for the purpose of system performance evaluation on both types of vehicular networks. Data preprocessing consists

TABLE III
RATIONALE AND PERFORMANCE IMPACT OF EACH COMPONENT OF THE MTH-IDS

Stage	Algorithm	Rationale and Description	Performance Impact
Data pre-processing	K-means cluster sampling	Network traffic data is often large, while IoV devices often have limited computational power and resources. The k-means sampling method can generate highly representative subsets for more efficient training because the removed data is mostly redundant data.	Improve model training efficiency.
	SMOTE	Network traffic data is often imbalanced data because most data samples are collected under normal conditions in real-world vehicle systems. SMOTE can create high-quality samples for minority classes to avoid class-imbalance and ineffective classifiers.	Improve detection rate.
	Z-score	Different features often have different ranges, which can bias the model training. The Z-score method can normalize features to a similar scale and handle outliers.	Improve model accuracy and training efficiency.
Feature engineering	IG	For certain tasks like intrusion detection, many collected features can be irrelevant, causing additional training time. The IG method can remove those unimportant features.	Improve model training efficiency.
	FCBF	Certain features are redundant because they contain very similar information. FCBF can remove redundant features by calculating the correlation between each pair of features.	Improve model accuracy and training efficiency.
	KPCA	The anomaly-based IDSs are sensitive to the quality of features. KPCA can further extract the most relevant features to reduce dimensionality and noisy information.	Improve model accuracy and training efficiency.
The signature-based IDS	DT, RF, ET, and XGBoost	Tree-based ML algorithms often perform better than other ML algorithms on complex tabular data to which IoV data belong. Four tree-based supervised algorithms are used to train base classifiers for known intrusion detection.	Detect various types of known attacks.
	BO-TPE	The default hyper-parameters of ML algorithms often cannot return the best model. BO-TPE can optimize the models' hyper-parameters to obtain the optimized base classifiers.	Improve accuracy of known attack detection.
	Stacking	Ensemble models can often achieve higher accuracy than any single model. Stacking ensemble can combine the base classifiers to obtain a meta-learner with better performance.	Improve accuracy of known attack detection.
The anomaly-based IDS	CL-k-means	For unknown attack detection, CL-k-means can generate a sufficient number of normal and attack clusters to identify zero-day attacks from the newly arriving data.	Detect unknown attacks.
	BO-GP	CL-k-means has an important hyper-parameter, the number of clusters, k. BO-GP is an effective HPO method to optimize k and obtain the optimized CL-k-means model.	Improve accuracy of unknown attack detection.
	Two biased classifiers	CL-k-means may return many errors when detecting complex unknown attacks. Two biased classifiers are trained on the FPs and FNs of CL-k-means to reduce the errors.	Improve accuracy of unknown attack detection.

of a k-means-based cluster sampling method used to generate a highly representative subset, and a SMOTE method used to avoid class imbalance. In the feature engineering process, the data sets are processed by information-gain-based and correlation-based feature selection (FS) methods to remove irrelevant and redundant features, and then passed to the KPCA model to further reduce dimensionality and noisy features. The proposed data preprocessing and feature engineering procedures can greatly improve the quality of the network data for more accurate model learning. The signature-based IDS is then developed to detect known attacks by training four tree-based machine learners as the first tier of the proposed MTH-IDS: DT, RF, ET, and XGBoost. In the second tier, a stacking ensemble model and the BO-TPE method are used to further improve the intrusion detection accuracy by combining the output of the four base learners from the first tier and optimizing the learners. In the next stage, an anomaly-based IDS is constructed to detect unknown attacks. In the anomaly-based IDS, the suspicious instances are passed to a CL k-means model as the third tier to effectively separate attack samples from normal samples. The fourth tier of the MTH-IDS comprises the BO-GP method and two biased classifiers used to optimize the model and reduce the classification errors of the CL-k-means. Ultimately, the detection result of each test sample is returned, which could be a known attack with its type, an unknown attack, or a normal packet. To summarize the rationale behind the algorithms used in the proposed IDS, the brief description and performance impact of each algorithm are presented in Table III. A detailed description is provided in Sections IV-B–IV-D.

B. Data Preprocessing

1) Data Sampling by K-Means Clustering: In real life, training ML models on massive amounts of network traffic data is unrealistic and may cost a massive amount of time, especially in the hyperparameter tuning process that needs to train an ML model multiple times. For model training efficiency improvement purposes, data sampling is a common technique that can generate a subset of the original data to reduce the training complexity of a model [41].

In the proposed system, to obtain a highly representative subset, a k-means-based cluster sampling method is utilized. Cluster sampling is a common data sampling method by which the original data points are grouped into multiple clusters; then, a proportion of data is sampled from each cluster to form a representative subset [41]. Unlike random sampling, which randomly selects every data sample with an equal probability, cluster sampling can generate a highly representative subset because the discarded data points are mostly redundant data.

Among all clustering algorithms, k-means is the most common one for data sampling due to its simple implementation and low computational complexity [42]. K-means clustering algorithms are used to divide the data points into k clusters based on their Euclidean, Manhattan, or Mahalanobis distances [43], [44]. The data samples in the same group can be considered similar samples, so sampling from each group can greatly reduce the size of data without losing important information. K-means aims to minimize the sum of squares of distances between all the data points and the corresponding centroid of the cluster, denoted by [42]

$$\sum_{i=0}^{n_k} \min_{u_j \in C_k} \left(\mathbf{x}_i - u_j\right)^2 \tag{1}$$

where $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is the data matrix; u_j, also called the centroid of a cluster C_k, is the mean of all the samples in C_k; and n_k is the total number of sample points in the cluster C_k. K-means has a linear time complexity of $O(nkt)$, where n is the data size, k is the number of clusters, and t is the number of iterations [42].

After implementing the k-means to cluster the original data samples into k clusters, random sampling is then applied to each cluster to select 10% of the data as the sampled subset. The percentage of data sampling can vary depending on the data scale and resource limitations. Additionally, the main hyperparameter of k-means, the number of clusters [45], k, is tuned by BO to improve the quality of the subset.

HPO is the process of building an optimized ML model for a specific problem or data set using an optimization algorithm [46]. BO algorithms are an efficient group of HPO algorithms that determine the next hyperparameter value based on the previous evaluation results [47]. In BO, a surrogate model is used to fit all the currently tested data points into the objective function; an acquisition function is then used to locate the next point. Two common surrogate models for BO are the Gaussian process (GP) and the tree Parzen Estimator (TPE) [45].

In GP surrogate models, the predictions follow a Gaussian distribution [45]:

$$p(y|x, D) = N\left(y|\hat{\mu}, \hat{\sigma}^2\right) \tag{2}$$

where D is the hyperparameter configuration space, and $y = f(x)$ is the value of the objective function for each hyperparameter configuration with its mean as μ and covariance as σ. The BO method with GP (BO-GP) exhibits great performance on optimizing a small number of continuous or discrete hyperparameters due to its fast convergence speed, but is inefficient for conditional hyperparameters since it treats each hyperparameter independently [45]. BO-GP has a time complexity of $O(n^3)$ and space complexity of $O(n^2)$ [45].

Since k-means methods generally only have one discrete hyperparameter, k, that needs to be tuned, BO-GP serves as an effective HPO method for k-means because of its fast convergence speed. The silhouette coefficient, a common distance-based metric that can effectively evaluate clustering performance, is chosen to be the metric of the k-means model and used as the objective function of BO-GP. It measures how similar a data point is to other data points within the same cluster and how different the data point is from the data points in other clusters [43].

2) Reduce Class Imbalance by Oversampling: Class-imbalance issues often occur in network traffic data, since the percentage of normal samples is often much larger than the percentage of attack samples in real-world network data, resulting in biased models and low DR [48].

Class-imbalance problems are mainly solved by resampling methods, including random sampling and synthetic minority oversampling techniques (SMOTE), which can create new instances for the minority classes to balance the data set [48]. Unlike random sampling, which simply replicates the instances and may cause overfitting, SMOTE [49] can synthesize high-quality instances based on the concept of KNN; thus, SMOTE is chosen in the proposed IDS to solve class imbalance. For each instance X in the minority class, assuming X_i is a sample randomly selected from the k-nearest neighbors of X, a new synthetic instance X_n can be denoted by [28]

$$X_n = X + \text{rand}(0, 1) * (X_i - X), \ \ i = 1, 2, \ldots, k \tag{3}$$

where rand(0, 1) represents a random number in the range of (0, 1). Thus, SMOTE is utilized in the proposed IDS to create high-quality instances for minority classes.

3) Data Normalization: After implementing the k-means and SMOTE methods to obtain a representative and balanced data set, several additional data preprocessing steps are completed for the next steps. First, the network traffic data sets are encoded with a label encoder used to transform categorical features into numerical features to support the inputs of ML algorithms, because many ML algorithms cannot support string features directly. After that, the network data sets are normalized by the Z-score algorithm since the collected features in network traffic data often have largely different ranges, and ML models often perform better on normalized data sets [20]. An unnormalized data set with largely different feature scales may result in a biased ML model that only lays emphasis on large-scale features. Through the Z-score method, the features can be normalized to have a mean of 0 and a standard deviation of 1. By implementing the Z-score method, each normalized feature value x_n is denoted by

$$x_n = \frac{x - \mu}{\sigma} \tag{4}$$

where x is the original feature value, and μ and σ are the mean and standard deviation of the feature values, respectively.

C. Feature Engineering

A high-quality and highly representative data set can be generated after data preprocessing. On the other hand, obtaining an optimal feature list by appropriate feature engineering can also improve the quality of data sets for more accurate and efficient model learning. A comprehensive feature engineering method that consists of IG, FCBF, and KPCA is implemented before ML model training to remove irrelevant, redundant, and noisy features while retaining the important features [50].

1) Feature Selection by Information Gain: As a common FS method, the IG method is used to select important features. IG, the amount of information gained or the changes in entropy, can be used to measure how much information a feature can bring to the targeted variable [51]. IG is chosen in the proposed system since it can obtain an importance score for each feature at a fast speed due to its low computational complexity of $O(n)$ [51]. The importance score of each feature enables us to select the most relevant features for the task. Assuming T is the target variable, for each feature denoted by a random variable X, the IG value of using the feature X is denoted by [51]

$$\text{IG}(T|X) = H(T) - H(T|X) \tag{5}$$

where $H(T)$ is the entropy of the target variable T, and $H(T|X)$ is the conditional entropy of T over X.

Therefore, the feature importance of a feature X, or the correlation between X and the target T, can be represented by the IG value of T over X, $\text{IG}(T|X)$. A feature X is considered more important to the target T than another feature Y if $\text{IG}(T|X) > \text{IG}(T|Y)$ [51].

To implement the IG-based FS method, the importance of each feature is calculated based on (5) and normalized to have

a sum of 1.0, denoting the relative importance. The features are then ranked by their importance and are selected from top to bottom until the total importance of selected features reaches the correlation threshold α. The remaining features, with the total feature importance less than $1-\alpha$, are discarded. To obtain an appropriate correlation threshold, α is optimized by BO-GP that uses the validation accuracy as the objective function for HPO. After this process, the irrelevant features are eliminated, and a reduced number of informative features with high importance are obtained for the next step.

2) Fast Correlation-Based Filter: Although the IG-based FS method eliminates the unimportant features to reduce time complexity, many redundant features still exist. Feature redundancy may increase time and space complexity, and degrade model performance by increasing the probability of being misled by noisy data, as well as increasing the risk of overfitting [50]. Thus, removing redundant features by calculating the correlations of input features is beneficial for model performance and efficiency.

Among the correlation-based FS algorithms, the FCBF [52] algorithm is selected since it has shown great performance on high-dimensional data sets by effectively removing redundant features while retaining informative features, and has a low time complexity of $O(n\log n)$ [53]. In FCBF, the symmetrical uncertainty (SU) is calculated to measure the correlations between features by normalizing the IG values [52]

$$\mathrm{SU}(X, Y) = 2\left[\frac{\mathrm{IG}(X|Y)}{H(X) + H(Y)}\right]. \tag{6}$$

$\mathrm{SU}(X, Y)$ is in the range $[0, 1]$ with the value 1 indicating a perfect correlation between the two features X and Y, while the value 0 indicates the two features are fully independent.

The FCBF method searches the features in the feature space based on their SU values until the entire feature space has been explored. The highly correlated features are regarded as redundant features, and only one of them will be retained.

In the proposed FS approach, the SU value of each pair of features is calculated as their correlations. The correlation threshold α is also optimized by BO-GP. When the correlation value between two features is larger than α, the one with higher feature importance is retained while the other is discarded. The correlation calculation and feature deletion procedures are repeated until each pair of features in the feature list are not highly correlated ($\mathrm{SU} <= \alpha$). The FS model that combines the IG method and the FCBF algorithm is named IG-FCBF.

3) Kernel Principal Component Analysis: Although utilizing IG-FCBF can return a better feature set than only using IG, FCBF has a major limitation that it only calculates the correlation between pairs of features, but does not consider correlations among three or more different features, resulting in undiscovered noisy features [53]. On the other hand, the unsupervised learning models in the anomaly-based IDS are more sensitive to appropriate features than supervised learning models since they rely on the changes of feature values instead of the ground-truth labels to process data [43], [50]. Hence, KPCA is utilized after implementing the IG-FCBF method for the anomaly-based IDS.

PCA [54] is a feature extraction algorithm that uses orthogonal transformations to transform a set of correlated features onto a smaller subset of uncorrelated features, named principal components. KPCA is an improved version of PCA that uses the kernel trick to learn a nonlinear function or decision boundary to reduce the dimensionality of nonlinear data [55]. KPCA is selected due to its adaptability to nonlinear data, as well as its capacity to reduce computational complexity, the risk of overfitting, and distracting noise [50].

Additionally, the two essential hyperparameters in KPCA, the number of extracted features and the kernel type, are optimized by the BO-GP method using validation accuracy as the objective function to improve the model performance. It is efficient for BO-GP to optimize these discrete and categorical hyperparameters. KPCA is used with IG-FCBF to construct the IG-FCBF-KPCA method to obtain an optimal data set with extracted features as the input of the anomaly-based IDS.

D. Proposed Hybrid IDS

IDSs are mainly classified as signature-based IDSs and anomaly-based IDSs. Signature-based IDSs are designed to detect the known attack patterns by training supervised ML models on labeled data sets. However, they often lack the capacity to detect new attack patterns that are not previously stored in the databases [56]. On the other hand, anomaly-based IDSs can distinguish unknown attack data from normal data by unsupervised learning algorithms based on the assumption that new attack data are more statistically similar to the known attack data than normal data, but they often return many false alarms [57]. Thus, a hybrid IDS that consists of a signature-based IDS and an anomaly-based IDS is proposed in this article to effectively detect both known and zero-day attacks.

1) Signature-Based IDS: After the data preprocessing and feature engineering procedures, the obtained labeled data sets are trained by an ensemble learning model to develop a signature-based IDS. In the proposed signature-based IDS, four tree-based ML algorithms—DT, RF, ETs, and XGBoost—are selected as the base learners.

DT [58] is a common ML algorithm that uses a tree structure to fit data and make predictions. DT algorithms have multiple hyperparameters that require tuning, including the tree depth, minimum sample split, minimum sample leaf, maximum sample nodes, minimum weight fraction leaf, etc. [45]. RF [59] is an ensemble learning model that uses the majority voting rule to combine multiple DT classifiers, while ET [60] combines a collection of randomized DTs built on different subsets of a data set. XGBoost [61] is a gradient-boosted DT (GBDT)-based algorithm designed for speed and performance improvement. For RF, ET, and XGBoost, the hyperparameters of DT are also important hyperparameters for them because they are all constructed by integrating multiple DTs. Additionally, they have an essential hyperparameter that needs to be tuned, being the number of base DTs to be built for each model, "n_estimators," which has a direct impact on model performance. XGBoost has another hyperparameter, the learning rate, which determines the convergence speed [45].

Assuming the number of instances is n, the number of features is f, and the number of DTs in ensemble models is t, the time complexity of DT, RF, ET, and XGBoost is $O(n^2 f)$, $O(n^2\sqrt{f}t)$, $O(nft)$, and $O(nft)$, respectively [7].

The algorithm choosing reasons are as follows [7], [62].

1) RF, ET, and XGBoost are all ensemble models that combine multiple DTs and can effectively work on nonlinear and complex data to which network traffic data belongs; hence, they often perform better than other ML algorithms, such as naïve Bayes (NB) and KNN, which often do not perform well on complex data sets.
2) They enable parallel execution, which significantly reduces model training time and improves efficiency.
3) They calculate feature importance during the model training process, which is beneficial for feature engineering procedures.
4) The tree-based algorithms have randomness in their construction process, which enables us to build a robust ensemble model that has better generalizability than using other ML algorithms.

After obtaining the four tree-based ML models, they are combined using stacking, an ensemble learning method, to improve model performance because the generalizability of a combination of multiple base learners is usually better than that of a single model [63]. Stacking is a standard ensemble learning technique that uses the output labels estimated by four base learners (DT, RF, ET, and XGBoost) as the input features to train a strong metalearner that makes the final prediction [63]. Using stacking can learn the information from all four base learners to reduce the errors of single learners and obtain a more reliable and robust metaclassifier. In the proposed system, the best-performing one among the four base models is chosen as the algorithm to build the metalearner because it is most likely to achieve the best performance.

The important hyperparameters of the four tree-based ML algorithms are optimized by an HPO method, BO-TPE. BO-TPE creates two density functions, $l(x)$ and $g(x)$, to act as the generative models for variables. With a prespecified threshold y^* to separate the relatively good and poor results, the objective function of TPE is modeled by the Parzen windows [45]

$$p(x|y, D) = \begin{cases} l(x), & \text{if } y < y^* \\ g(x), & \text{if } y > y^* \end{cases} \tag{7}$$

where $l(x)$ and $g(x)$ indicate the probability of detecting the next hyperparameter value in the well-performing regions and in the poor-performing regions, respectively. BO-TPE detects the optimal hyperparameter values by maximizing the ratio $l(x)/g(x)$. The Parzen estimators are organized in a tree structure, so the specified conditional dependencies of hyperparameters can be retained. Additionally, BO-TPE can optimize all types of hyperparameters effectively [45]. Therefore, BO-TPE is used to optimize the hyperparameters of the tree-based ML models that have many hyperparameters.

2) Anomaly-Based IDS: The proposed signature-based IDS can detect multiple types of known attacks effectively. However, attackers can still carry out zero-day attacks that are not included in the known attack patterns and can be misclassified as normal states. Therefore, the instances labeled "normal" by the signature-based IDS will be considered suspicious instances because some of them can be unknown attack samples. A novel anomaly-based IDS architecture is then developed to identify zero-day attacks by processing the suspicious instances.

After feature engineering, the optimized data set obtained from the output of the IG-FCBF-KPCA method is used to train the anomaly-based IDS. The first tier of the proposed anomaly-based IDS comprises the CL k-means developed by improving the k-means model introduced in Section IV-B1. Since there are millions of instances collected under many different situations in network traffic data sets, a sufficient number of clusters should be used to distinguish between normal and attack data. The main procedures of the proposed CL-k-means method are as follows.

1) Split the data set into a sufficient number of clusters using k-means.
2) Label each cluster by the majority label of data samples. In each cluster, the class label to which most of the instances belong, "normal" or "attack," is assigned to this cluster.
3) Label each sample in the test set as normal or attack based on the label of the cluster that this instance is classified into by k-means.
4) For each test sample i that is classified into a cluster, calculate the percentage of majority class samples in this cluster as the confidence or clustering probability, p_i.
5) Optimize the number of clusters (k) and distance metric as the major hyperparameters of k-means, by the BO-GP algorithm to obtain the optimal CL-k-means model. The validation accuracy on the test set is used as the objective function for BO-GP.

K-means is selected to distinguish between attack and normal data mainly due to the real-time requirements of vehicle-level systems. K-means is computationally faster than most other clustering algorithms because it has a linear time complexity of $O(nkt)$, where n is the data size, k is the number of clusters, and t is the number of iterations [42]. The model training time is further reduced by using mini-batch k-means, which uses randomly sampled subsets as mini-batches in each training iteration [64]. Additionally, k-means guarantees convergence and easily adapts to new samples.

To increase the DR and reduce the FAR of the CL-k-means method, the second tier of the proposed anomaly-based IDS uses two biased classifiers to reduce the false negatives (FNs) and false positives (FPs), respectively. Biased classifiers are developed by the following procedures.

1) Collect the FNs and FPs obtained from the training set using the proposed CL-k-means method.
2) Select the best-performing singular supervised learning model in the signature-based IDS (e.g., RF) as the algorithm to construct biased classifiers.
3) Train the first biased classifier B_1 on all the FNs along with the same amount of randomly sampled normal data to build a model that aims to reduce FNs.

4) Train the second biased classifier B_2 on all the FPs along with the same amount of randomly sampled attack data to build a model that aims to reduce FPs.

After implementing the proposed CL-k-means model, each data sample whose clustering probability (p_i) is less than a threshold p_i^*, is regarded as an uncertain instance. The threshold p_i^* is a continuous variable and has been optimized to be 0.933 by BO-GP that uses the validation accuracy as the objective function. After obtaining the two trained classifiers, the uncertain instances will be passed to B_1 (if labeled normal by the CL-k-means) or B_2 (if a labeled attack by the CL-k-means) to obtain its final classification result.

The proposed anomaly-based IDS is constructed under the assumption that new attack patterns are unknown and future incoming data samples are unlabeled; hence, only the FNs and FPs obtained during the training phase are used to build biased classifiers. This enables the proposed IDS to detect new attack patterns without additional procedures that are difficult to perform, such as constant data labeling and model updates.

Compared to other unsupervised anomaly detection algorithms, such as isolation forest (iForest) and one-class SVM (OC-SVM) [65], the proposed CL-k-means method with biased classifiers has the following advantages to achieving high accuracy and efficiency.

1) The proposed CL-k-means model has the capacity of using a sufficient number of clusters to model the data samples with various attack and normal patterns. This makes the proposed method have better generalizability and data pattern modeling capability than other outlier detection methods, such as iForest and OC-SVM, which are mostly binary models.
2) The number of clusters in CL-k-means k is automatically optimized by BO-GP. This enables the proposed model to automatically fit different data sets and tasks according to the complexity of data patterns.
3) The main difficulty of unknown attack detection is that unsupervised learning models often return more misclassified samples than supervised learning models. Thus, using the biased classifiers can effectively reduce the FPs and FNs because they can learn the patterns of misclassified data samples that are difficult to be identified by the CL-k-means.
4) The use of the cluster probability p_i can greatly improve the model efficiency. This is because the new samples that are very similar to existing attack or normal patterns (with high confidence) can be labeled directly, and only the uncertain samples (with relatively low probability) are passed to the biased classifiers for further identification.
5) The use of mini-batch k-means can significantly reduce the execution time of the MTH-IDS to meet the real-time requirements of IoV.

However, certain attack patterns are very similar to normal patterns, which makes it difficult to distinguish them. Additionally, the samples collected under certain legitimate network events, like crowd events, may still be misclassified as attack samples because the new patterns they have are largely different from existing normal patterns. Additionally, although k-means is more suitable for IoV IDS development than other unsupervised learning algorithms due to its low complexity, the data distribution modeling limitation of k-means is another potential issue to be better addressed [42]. If time and budget permits, k-means can be replaced by other clustering algorithms with the same CL technique based on the specific data shape and distribution to further improve the system performance. Online learning techniques that can keep updating learning models based on the new attack patterns may also improve the generalizability and accuracy of the IDS, which will be our future work.

E. Runtime Complexity

The training process of the proposed MTH-IDS can be done at a server machine with high computational speed, while the testing process should be implemented in vehicle systems. Developing models with low runtime complexity enables the proposed IDS to reduce the latency of vehicle systems and meet real-time requirements. In the implementations, each test sample will be passed through the stacking model constructed with four tree-based algorithms, the CL-k-means method, and one of the biased classifiers. Since the runtime complexity of DT is $O(df)$ and the runtime complexity of RF, ET, and XGBoost is $O(dft)$, where d is the maximum depth of the trees, f is the number of features, and t is the number of trees, the maximum runtime complexity of the signature-based IDS is only $O(dft)$ [66]. For the proposed CL-k-means method in the anomaly-based IDS, its runtime complexity is $O(fk)$, where k is the number of clusters [67]. The biased classifier in the anomaly-based IDS is also the best-performing tree-based algorithm, so its maximum runtime complexity is also $O(dft)$.

Therefore, the maximum overall runtime complexity of the proposed IDS is at a low level, only $O(2dft + fk)$, since the values of d, f, t, and k are a few dozens at most. The model test time will be calculated in the experiments to evaluate the feasibility of the proposed IDS in vehicle systems.

F. Validation Metrics

To evaluate the generalizability of the proposed framework and avoid overfitting, both cross-validation and hold-out methods are used in the known attack detection experiments. Specifically, the train–test–validation split and model evaluation procedures for each data set are as follows.

1) Use 70%–30% train–test split to generate a training set with 70% of data samples and a test set with 30% of data samples. The test set will remain untouched before the final hold-out validation.
2) Implement tenfold cross-validation on the training set to evaluate the proposed model on different regions of the data set. In each iteration/fold of the tenfold cross-validation, 90% of the original training set is used for model training, and 10% of the original training set is used as the validation set for model testing.
3) Test the trained model obtained from step 2 on the untouched test set to evaluate the model performance on a new data set.

70%–30% train–test split and tenfold cross-validation were chosen because they are standard and sufficient numbers to construct powerful validation methods to avoid overfitting and concept drift issues [68]. If the proposed method can accurately detect intrusions in both cross-validation and hold-out validation, there are no underfitting or overfitting issues [69].

On the other hand, hold-out validation is used to evaluate the proposed system for unknown attack detection. For each attack type as a zero-day attack, a validation set consisting of all the instances of this attack type and the same amount of randomly sampled normal data is generated; all the other samples are used as the training set. Therefore, the validation results can be used to evaluate whether the proposed system has the capacity to detect the patterns of each type of unknown attack. This validation process is based on the assumption that new attack data is more statistically similar to certain other known attack data than normal data [57].

Several metrics, including accuracy (Acc), DR, FAR, and F1-score, are used to comprehensively evaluate the performance of the proposed IDS [28]. By calculating the true positives (TPs), true negatives (TNs), FPs, and FNs of the proposed model, the used metrics are calculated by the following equations [28]:

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{8}$$

$$\text{DR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{9}$$

$$\text{FAR} = \frac{\text{FP}}{\text{TN} + \text{FP}} \tag{10}$$

$$\text{F1} = \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}}. \tag{11}$$

Model execution time, calculated by the average of the model training and validation time in the tenfold cross-validation or hold-out validation, is used to evaluate model efficiency. An efficient IDS should be able to achieve a high F1-score and low execution time simultaneously.

V. Performance Evaluation

A. Experimental Setup

To develop the proposed MTH-IDS, the feature engineering and ML algorithms were implemented using the Pandas [70], Scikit-learn [71], and Xgboost [72] libraries in Python, while the HPO methods were implemented by extending the Skopt [73] and Hyperopt [74] libraries.[1] The experiments were carried out on a Dell Precision 3630 Tower machine with an i7-8700 central processing unit (CPU) (6-Core, 3.20 GHz) and 16 Gigabytes (GB) of memory, and a Raspberry Pi 3 machine with a BCM2837B0 64-bit CPU and 1 GB of memory, representing a server machine for model training and a vehicle-level machine for model testing, respectively.

The experiments are divided into three parts, one for the known intrusion detection by evaluating the signature-based IDS component on the labeled data sets, one for unknown intrusion detection by evaluating the anomaly-based IDS component on the unlabeled data sets, and one for the entire model evaluation by analyzing the CPU resource usage on a vehicle-level machine.

[1]The code for the major modules is available at: https://github.com/Western-OC2-Lab/Intrusion-Detection-System-Using-Machine-Learning.

B. Data Description

For the purpose of IVN IDS development, the first used data set is the CAN-intrusion-dataset proposed in 2018 [9]. The data set is generated by logging CAN traffic via the OBD-II port of a vehicle when CAN attacks are launched. The features of this data set include timestamp, CAN ID, data length code (DLC), and the 8-bit data field of CAN packets (DATA[0]–DATA[7]). Since the feature "timestamp" has a strong correlation with cyber-attack simulation periods and can lead to biased models and results, this feature was removed from the feature space. As Seo *et al.* [9] created the CAN-intrusion-dataset by injecting attacks into random CAN IDs, this CAN ID feature is retained. Based on the CAN packet structure shown in Fig. 1, ten features extracted from the ID field and data field of CAN packets, including CAN ID, DLC, and DATA[0]–DATA[7], were preliminarily selected for IDS development. The attack types of the CAN-intrusion-dataset are shown in Table IV. Moreover, the label distributions of the original data set, the training set, and the test set are shown in Table IV. As the minority classes have large numbers of samples (at least 491 847 samples), SMOTE is not required for balancing the CAN-intrusion-dataset.

For external network IDS development, there is a shortage of public IoV benchmark data sets due to popularization, privacy, and commercialization issues [21], [75], [76]. On the other hand, wireless local area networks (WLANs) and cellular networks are common communication strategies for IoV and CVs, so the attacks launched on conventional computer networks can be considered similar to the intrusions carried out on external vehicular networks [21], [77]. Therefore, many research projects and papers [19]–[25] develop external vehicular network IDSs using general network security data sets, including KDD-99, NSL-KDD, Kyoto 2006+, UNSW-NB15, and CICIDS2017 [77]. Among these cyber-security data sets, CICIDS2017 [11] is the most representative data set of current external networks because it is the most state-of-the-art data set and contains more features, instances, and cyber-attack types than other data sets [21]. Thus, the network traffic flow data in the CICIDS2017 data set is chosen in the proposed MTH-IDS to represent the complex external vehicular network data. Moreover, to better relate the CICIDS2017 data set to IoV applications, we have associated each type of attack in the CICIDS2017 data set with the external vehicular network threats described in Table II based on the detailed analysis of the CICIDS2017 data set in [78]. The specifics of the CICIDS2017 data set and the corresponding external vehicular attack types are shown in Table V. Since the Bot, brute-force, infiltration, and Web attack classes are minority classes with small numbers of samples (from 36 to 13 835), the SMOTE method described in Section IV-B2 was implemented to synthesize more samples to enable the minority classes to have at

TABLE IV
CLASS LABEL AND SIZE OF THE CAN-INTRUSION-DATASET

Class Label	Original Number of Samples	Number of Training Set Samples	Number of Test Set Samples
Normal	14,037,293	9,826,105	4,211,188
DoS	587,521	411,265	176,256
Fuzzy	491,847	344,293	147,554
RPM Spoofing	654,897	458,428	196,469
Gear Spoofing	597,252	418,076	179,176

TABLE V
CLASS LABEL, ATTACK TYPE, AND SIZE OF THE CICIDS2017 DATA SET

Class Label	Corresponding Attack Type in Table II [78]	Original Number of Samples	Number of Training Set Samples After Balancing	Number of Test Set Samples
BENIGN	-	2,273,097	1,591,168	681,929
Bot	Botnets	1,966	100,000	590
DDoS	DoS	380,699	266,489	114,210
DoS GoldenEye				
DoS Hulk				
DoS Slow-httptest				
DoS Slowloris				
Heartbleed				
Port-Scan	Sniffing	158,930	111,251	47,679
SSH-Patator	Brute-Force	13,835	100,000	4,150
FTP-Patator				
Infiltration	Infiltration	36	100,000	11
Web Attack – Brute Force	Web Attack	2,180	100,000	654
Web Attack – Sql Injection				
Web Attack – XSS				

TABLE VI
PERFORMANCE EVALUATION OF CLASSIFIERS ON THE CAN-INTRUSION-DATASET

Method	Acc (%)	DR (%)	FAR (%)	F1	Execution Time (S)
KNN [12]	97.4	96.3	5.3	0.934	911.6
SVM [12]	96.5	95.7	4.8	0.933	13765.6
XYF-K [13]	99.1	98.39	0.0	0.9879	-
SAIDuCANT [14]	87.21	86.66	1.76	0.92	-
SSAE [17]	-	98.5	2.0	0.98	-
DCNN [18]	99.93	99.84	0.16	0.9991	-
LSTM-Autoencoder [19]	99.0	99.0	0.0	0.99	-
MTH-IDS	**99.999**	**99.999**	**0.0006**	**0.99999**	**365.3**

least 100 000 samples. Addressing class imbalance can avoid obtaining biased models with low attack DRs.

C. Performance Analysis of Known Intrusion Detection

To evaluate the proposed IDS for known intrusion detection, the ML models in the signature-based IDS are trained and tested on the two labeled data sets that represent intravehicle and external vehicular network traffic data. The results of tenfold cross-validation of the proposed models on the CAN-intrusion-dataset [9] and the CICIDS2017 data set [11] are shown in Tables VI and VII, respectively.

As shown in Table VI, after utilizing the proposed IG-FCBF FS method with the optimized correlation threshold ($\alpha = 0.9$) to select the top four significant features ("CAN ID," "DATA[5]," "DATA[3]," and "DATA[1]"), the proposed signature-based IDS can reach high accuracy of 99.999% on the CAN-intrusion-dataset. The proposed method is compared with recent promising approaches proposed in [12]–[14] and [17]–[19], as shown in Table VI. As the attack and normal patterns in this data set can be obviously distinguished, most of the compared methods also achieve high accuracy. Song *et al.* [18] proposed a DCNN method that can achieve a high average F1-score of 0.9991, but it requires the model training machines to have graphics processing units (GPUs), making it difficult to be used in vehicle-level systems due to budget constraints. For six other compared approaches proposed in the recent literature, our proposed system achieves at least 0.89% accuracy and F1-score improvement.

TABLE VII
PERFORMANCE EVALUATION OF CLASSIFIERS ON THE CICIDS2017 DATA SET

Method	Acc (%)	DR (%)	FAR (%)	F1	Execution Time (S)
KNN [11]	96.3	96.2	6.3	0.963	15243.6
RF [11]	98.82	98.8	0.145	0.988	1848.3
SU-IDS [26]	99.13	99.65	1.4	-	-
STDeepGraph [27]	99.4	98.6	1.3	-	-
DBN [79]	98.95	95.82	4.19	0.9581	-
GAN-RF [80]	99.83	98.68	7.24	0.9504	-
DeepCoin [81]	99.811	94.10	0.986	-	-
Multi-SVM [82]	98.55	98.22	0.41	0.983	34896.5
PCA-RF [83]	99.6	99.6	1.0	0.996	-
MTH-IDS (Without FS & HPO)	99.861	99.753	0.110	0.99860	5238.4
MTH-IDS (Multi-Class Model)	**99.879**	**99.818**	**0.101**	**0.99879**	**1563.4**
MTH-IDS (Binary Model)	**99.895**	**99.806**	**0.084**	**0.99895**	**478.2**

The experimental results on the CICIDS2017 data set are shown in Table VII. By implementing the proposed IG-FCBF method with the optimized correlation threshold ($\alpha = 0.9$) to select 20 features from 80 original features and using HPO methods to obtain optimized ML models, the F1-score of the proposed IDS has improved from 99.861% to 99.879%, and the execution time has decreased by 70.2%, as shown in Table VII. This justifies the proposed FS method and the BO-TPE method can greatly improve the system efficiency and slightly improve the model accuracy. In addition to the multiclassification results used to evaluate the IDS's capacity to detect various types of attacks, the IDS is also implemented to train a binary classification model that can distinguish between normal and abnormal network traffic data and return one of these two labels. As shown in Table VII, the proposed IDS reaches 99.895% accuracy and saves 69.4% of the execution time by training the binary classification model. Binary classifiers and multiclassifiers can be chosen according to the specific needs of users.

Although there are many existing works that evaluate their models on the CICIDS2017 data set, most of them have different or inexplicit validation configurations. Nevertheless, the proposed model is quantitatively compared with recent promising methods that achieve good performance on the CICIDS2017 data set and have a similar validation configuration, proposed in [11], [26], [27], and [79]–[83]. The approaches proposed in [79]–[81] have achieved high accuracy

TABLE VIII
PERFORMANCE EVALUATION ON EACH TYPE OF UNKNOWN ATTACK OF THE CAN-INTRUSION-DATASET

Attack Type	Validation Instances	DR (%)	FAR (%)	F1
DoS	1,289,386	100.0	0.0	1.0
Fuzzy	1,193,712	73.053	0.057	0.84389
Gear Spoofing	1,299,117	100.0	0.449	0.99736
RPM Spoofing	1,356,762	100.0	0.003	0.99998
Average (MTH-IDS)	**5,138,977**	**93.740**	**0.128**	**0.96307**
Average (CL-k-means)	5,138,977	79.233	0.261	0.82643

(98.95%–99.83%), but a relatively low DR (94.10%–98.68%) and F1-score (0.9504–0.9581) due to the imbalanced data set. Other methods proposed in [11], [26], [27], [82], and [83] have also achieved high accuracy or F1-scores, but still slightly lower than our proposed model. To summarize, even though the comparison with existing approaches is not a straightforward process, the proposed system shows better performance by at least achieving a 0.279% higher F1-score than of the same-level validation configuration approaches by quantitative comparison.

Therefore, the experimental results show that the proposed IDS can efficiently separate normal and malicious network traffic data and effectively detect various types of known cyber-attacks in vehicle systems.

D. Performance Analysis of Unknown Intrusion Detection

At this stage, all the models in the anomaly-based IDS are trained for binary classification by labeling the instances of all attack types as attack and normal instances as normal. In the proposed system, after being evaluated by the signature-based IDS, all data samples of known attack types will be returned, and other normal instances will be labeled "suspicious" and passed to the anomaly-based IDS to determine whether any unknown attacks exist.

For the CAN-intrusion-dataset that represents IVN traffic data, each type of CAN attack is regarded as a new attack type in each experiment. The evaluation results of unknown CAN attack detection are shown in Table VIII. For DoS, gear spoofing, and RPM spoofing attack types, the proposed system can reach 100% DRs, very low FARs (0.0%–0.449%), and very high F1-scores (more than 0.997). However, the DR and F1 for the fuzzy attack are much lower (73.053% and 0.84389). This is because the feature values of the fuzzy attack packets are random numerical values, and certain random values can be very similar to normal packets, making it difficult for unsupervised learning algorithms to distinguish them. Moreover, the performance of the proposed anomaly-based IDS is compared to the CL-k-means model without biased classifiers. Table VIII shows that the F1-score of the proposed MTH-IDS on the CAN-intrusion-dataset can be largely improved from 0.82643 to 0.96307 by implementing the two biased classifiers after the CL-k-means model. To summarize, the proposed system can effectively detect most unknown attacks on IVNs except fuzzy attacks.

On the other hand, training on more different types of attack samples enables us to design a more comprehensive IDS that can detect more unknown attack types effectively [84]. Therefore, several experiments were conducted on the CICIDS2017 data set that contains data samples of 14 different common cyber-attacks types to illustrate potential attacks launched on external vehicular networks. In each experiment of the validation process, each type of attack is regarded as an unknown attack, and the results are shown in Table IX.

TABLE IX
PERFORMANCE EVALUATION ON EACH TYPE OF UNKNOWN ATTACK OF THE CICIDS2017 DATA SET

Attack Type	Validation Instances	DR (%)	FAR (%)	F1
Bot	3,932	63.276	21.669	0.68426
DDoS	256,054	62.697	11.698	0.71902
DoS GoldenEye	20,586	83.931	20.461	0.82127
DoS Hulk	462,146	67.440	11.806	0.75248
DoS Slow-httptest	10,998	76.687	19.094	0.78339
DoS Slowloris	11,592	83.834	7.902	0.87447
FTP-Patator	15,876	51.298	12.686	0.62564
Heartbleed	22	100.0	18.182	0.91667
Infiltration	72	72.222	5.556	0.81250
Port-Scan	317,860	98.962	17.849	0.91288
SSH-Patator	11,794	95.828	23.351	0.87443
Web Attack – Brute Force	3,014	89.516	17.319	0.86558
Web Attack – Sql Injection	42	95.238	23.810	0.86957
Web Attack – XSS	1,304	3.681	14.417	0.06233
Average (MTH-IDS)	**1,115,292**	**75.943**	**13.882**	**0.80013**
Average (CL-k-means)	1,115,292	72.682	15.357	0.77305

From Table IX, it can be seen that the proposed system exhibits different performances when applied to the experiments of different types of unknown attacks. By implementing the proposed methods, the FARs for most of the attack types are at a low level of less than 20%. The DRs for the "Heartbleed," "Port-Scan," "SSH-Patator," "Web Attack—Brute Force," and "Web Attack—Sql Injection" attacks are high (from 89.516% to 100%), while the DRs for other types of attacks are relatively lower (from 51.298% to 83.931%). The F1-scores for most of the attack types are larger than 0.80. The only type of attack that the proposed system cannot detect effectively is the "Web Attack—XSS" whose results show a very low F1-score (0.062), because their data distribution is very similar to normal data distributions. The average F1-score of the proposed MTH-IDS on all the attacks is 0.80013, which is higher than the CL-k-means model without biased classifiers (0.77305).

Thus, the proposed IDS can detect most of the previously unseen types of attacks with a relatively high DR and a relatively low FAR on both intravehicle and external vehicular networks. Nevertheless, there is still some room for improvement since effectively detecting zero-day attacks is still an unsolved research problem.

E. Vehicle-Level Model Evaluation and Discussion

The proposed IDS with all trained models are tested on Raspberry Pi 3, a vehicle-level machine, to evaluate its feasibility in vehicular environments. Moreover, implementing the proposed model on the untouched test sets can evaluate its generalizability.

TABLE X
PERFORMANCE EVALUATION ON THE UNTOUCHED TEST SET

Dataset	Acc (%)	DR (%)	FAR (%)	F1
CAN-intrusion-dataset	99.99	100.0	0.00005	0.9999
CICIDS2017	99.88	99.77	0.10	0.9988

TABLE XI
MODEL EVALUATION ON A VEHICLE-LEVEL SYSTEM

System Component	Dataset 1: Avg Test Time (ms)	Dataset 2: Avg Test Time (ms)	Dataset 1: Model Space (MB)	Dataset 2: Model Space (MB)
Z-score	0.028	0.031	-	-
KPCA	0.005	0.009	0.002	0.006
Stacking	0.389	0.297	2.02	14.36
CL-k-means	0.145	0.157	0.48	0.78
Biased Classifiers	0.007	0.015	0.11	1.06
Sum	**0.574**	**0.509**	**2.61**	**16.21**

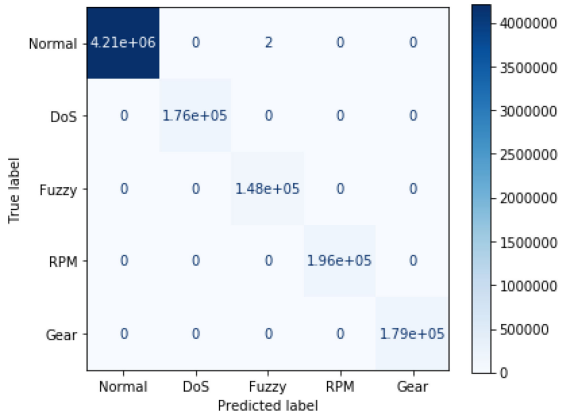

Fig. 4. Confusion matrix for the test set of the CAN-intrusion-dataset.

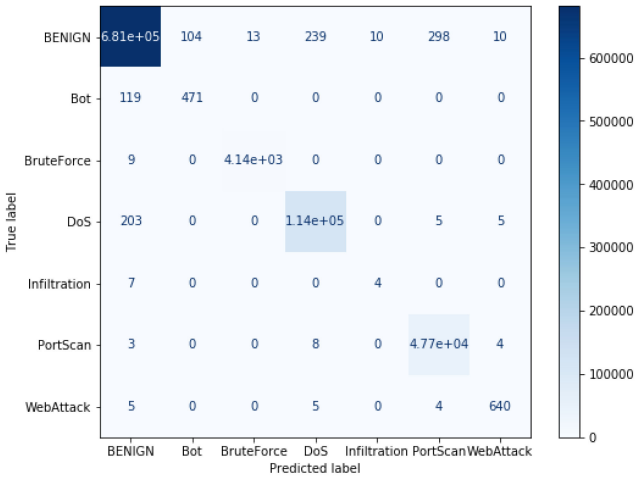

Fig. 5. Confusion matrix for the test set of the CICIDS2017 data set.

The experimental results on the test sets are shown in Table X. As shown in Table X, the F1-scores of the proposed IDS on the 30% test sets of the CAN-intrusion-dataset and CICIDS2017 data set are 99.99% and 99.88%, respectively. Moreover, the confusion matrices of evaluating the proposed method on the test sets of the CAN-intrusion-dataset and CICIDS2017 data set are shown in Figs. 4 and 5, respectively. For the CAN-intrusion-dataset, as shown in Fig. 4, the proposed method can accurately detect all the DoS, RPM spoofing, and gear spoofing attack samples, and only has two false alarms for the fuzzy attack detection. These results are in line with other similar works from the literature that used the same data set. For example, Song *et al.* [18] also achieved high accuracy of 99.93%. The main reason for achieving high accuracy is that the large difference between the attack and normal patterns in the CAN-intrusion-dataset can be obviously distinguished. For the CICIDS2017 results shown in Table X and Fig. 5, the attack patterns are more difficult to be distinguished than the CAN-intrusion-dataset, but the classification error rate is still at a very low level (0.12%), except for the infiltration attack type that has seven misclassified samples out of 11 test samples. This is because the number of data samples for the infiltration attack is only 36, which is insufficient to train an effective classifier to accurately identify this attack. Nevertheless, the proposed model can accurately detect other attacks with an overall F1-score of 99.88%. As the test sets were untouched before the hold-out validation, the high performance indicates the strong generalizability of the proposed framework on new data sets.

The proposed model can achieve high performance without overfitting mainly due to the following reasons [85].

1) It trains on large-sized data sets, as using more data samples can improve the generalizability of the proposed method.
2) It implements a comprehensive feature engineering method to improve the generalizability by removing irrelevant and misleading features that may cause overfitting.
3) It uses the stacking ensemble method to combine the results of base learners. Ensemble models often have better generalizability than single models because combining the single learners can reduce the variance of estimation and prevent overfitting.

On the other hand, to deploy the proposed IDS in real-world vehicle systems, the real-time requirements of vehicle safety services should be met. For each packet transmitted on vehicular networks, the alarm generation time should be less than 10 ms to meet the real-time requirements, as described in Section III-D. For each packet passed to the proposed IDS, it will be processed by Z-score normalization and four trained models: 1) KPCA; 2) stacking; 3) CL-k-means; and 4) a biased classifier (either FN-based or FP-based). As shown in Table XI, the average of the processing time for each packet in the CAN-intrusion-dataset (Dataset 1) and the CICIDS2017 data set (Dataset 2) is only 0.574 and 0.509 ms, respectively, which is much lower than the vehicular network security latency requirement (10 ms). Moreover, since we trained the models on the small-size subsets obtained by the proposed k-means cluster sampling method, the total size of the trained models for the IVN IDS and external network IDS is only 2.61 and 16.21 MB, respectively, which is much less than the memory limit of vehicle-level machines that often have more than 1 GB RAM, like Raspberry Pi 3. Thus, the experimental

results show the feasibility of applying the proposed system to real-time vehicle systems.

VI. Conclusion

To enhance IoV security, this work proposed an MTH-IDS model that can detect various types of known and zero-day cyber-attacks on both intravehicle and external-vehicular networks for modern vehicles. The proposed MTH-IDS consists of two traditional ML stages (data preprocessing and feature engineering) and four main tiers of learners utilizing multiple ML algorithms. Through data preprocessing and feature engineering, the quality of the input data can be significantly improved for more accurate model learning. The first tier of the proposed system consists of four tree-based supervised learners used for known attack detection, while the second tier comprises the BO-TPE and stacking models for supervised base learner optimization to achieve higher accuracy. The third tier consists of a novel CL-k-means unsupervised model used for unknown/zero-day attack detection. Finally, BO-GP and two biased classifiers are used to construct the fourth tier for unsupervised learner optimization. The four tiers of learning models enable the proposed MTH-IDS to achieve optimal performance for both known and unknown attack detection in vehicular networks.

Through the performance evaluation of the proposed IDS on the two public data sets that represent intravehicle and external vehicular network data, the proposed system can effectively detect various types of known attacks with accuracies of 99.99% and 99.88% on the CAN-intrusion-dataset and CICIDS2017 data set, respectively. Moreover, the proposed system can detect various types of unknown attacks with average F1-scores of 0.963 and 0.800 on the CAN-intrusion-dataset and CICIDS2017 data set, respectively. The experimental results on a vehicle-level machine also show the feasibility of the proposed system in real-time environments. In future work, the proposed anomaly-based IDS framework can be further improved by doing research on other unsupervised learning and online learning methods.

References

[1] H. Liang *et al.*, "Network and system level security in connected vehicle applications," in *IEEE/ACM Int. Conf. Comput.-Aided Design Dig. Tech. Papers (ICCAD)*, San Diego, CA, USA, 2018, pp. 1–7.

[2] M. Gmiden, M. H. Gmiden, and H. Trabelsi, "An intrusion detection method for securing in-vehicle CAN bus," in *Proc. 17th Int. Conf. Sci. Techn. Autom. Control Comput. Eng. (STA)*, Sousse, Tunisia, 2017, pp. 176–180.

[3] J. Liu, S. Zhang, W. Sun, and Y. Shi, "In-vehicle network attacks and countermeasures: Challenges and future directions," *IEEE Netw.*, vol. 31, no. 5, pp. 50–58, Sep. 2017.

[4] O. Y. Al-Jarrah, C. Maple, M. Dianati, D. Oxtoby, and A. Mouzakitis, "Intrusion detection systems for intra-vehicle networks: A review," *IEEE Access*, vol. 7, pp. 21266–21289, 2019.

[5] L. Yang, "Comprehensive visibility indicator algorithm for adaptable speed limit control in intelligent transportation systems," M.S. thesis, Dept. School Eng., Univ. Guelph, Guelph, ON, Canada, 2018.

[6] J. Golson, *Jeep Hackers at it Again, This Time Taking Control of Steering and Braking Systems*, Verge, Washington, DC, USA, Aug. 2016. Accessed: Nov. 11, 2020. [Online]. Available: https://www.theverge.com/2016/8/2/12353186/car-hack-jeep-cherokee-vulnerability-miller-valasek

[7] L. Yang, A. Moubayed, I. Hamieh, and A. Shami, "Tree-based intelligent intrusion detection system in Internet of vehicles," in *Proc. IEEE Global Commun. Conf.*, Waikoloa, HI, USA, 2019, pp. 1–6.

[8] Q. Wang, Y. Qian, Z. Lu, Y. Shoukry, and G. Qu, "A delay based plug-in-monitor for intrusion detection in controller area network," in *Proc. Asian Hardw. Orient. Security Trust Symp. (AsianHOST)*, Hong Kong, 2019, pp. 86–91.

[9] E. Seo, H. M. Song, and H. K. Kim, "GIDS: GAN based intrusion detection system for in-vehicle network," in *Proc. 16th Annu. Conf. Privacy Security Trust (PST)*, Belfast, Ireland, 2018, pp. 1–6.

[10] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Machine learning towards intelligent systems: Applications, challenges, and opportunities," *Artif. Intell. Rev.*, vol. 54, pp. 3299–3348, Jan. 2021.

[11] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. Int. Conf. Inf. Syst. Security Privacy*, 2018, pp. 108–116.

[12] A. Alshammari, M. A. Zohdy, D. Debnath, and G. Corser, "Classification approach for intrusion detection in vehicle systems," *Wireless Eng. Technol.*, vol. 9, no. 4, pp. 79–94, 2018.

[13] V. S. Barletta, D. Caivano, A. Nannavecchia, and M. Scalera, "A Kohonen SOM architecture for intrusion detection on in-vehicle communication networks," *Appl. Sci.*, vol. 10, p. 15, Jul. 2020.

[14] H. Olufowobi, C. Young, J. Zambreno, and G. Bloom, "SAIDuCANT: Specification-based automotive intrusion detection using controller area network (CAN) timing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 1484–1494, Feb. 2020.

[15] H. Olufowobi *et al.*, "Anomaly detection approach using adaptive cumulative sum algorithm for controller area network," in *Proc. ACM Workshop Autom. Cybersecurity*, 2019, pp. 25–30.

[16] H. Lee, S. H. Jeong, and H. K. Kim, "OTIDS: A novel intrusion detection system for in-vehicle network by using remote frame," in *Proc. 15th Annu. Conf. Privacy Security Trust (PST)*, Calgary, AB, Canada, 2018, pp. 57–66.

[17] S. F. Lokman *et al.*, "Stacked sparse autoencoders-based outlier discovery for in-vehicle controller area network (CAN)," *Int. J. Eng. Technol.*, vol. 7, pp. 375–380, Aug. 2019.

[18] 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.

[19] 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.*, early access, Sep. 16, 2020, doi: 10.1109/TITS.2020.3017882.

[20] K. M. A. Alheeti and K. Mc Donald-Maier, "Intelligent intrusion detection in external communication systems for autonomous vehicles," *Syst. Sci. Control Eng.*, vol. 6, no. 1, pp. 48–56, 2018.

[21] A. Rosay, F. Carlier, and P. Leroux, "Feed-forward neural network for Network Intrusion Detection," in *Proc. IEEE 91st Veh. Technol. Conf. (VTC-Spring)*, Antwerp, Belgium, May 2020, pp. 1–6.

[22] K. Aswal, D. C. Dobhal, and H. Pathak, "Comparative analysis of machine learning algorithms for identification of BOT attack on the Internet of Vehicles (IoV)," in *Proc. Int. Conf. Inventive Comput. Technol. (ICICT)*, Coimbatore, India, Feb. 2020, pp. 312–317.

[23] M. Aloqaily, S. Otoum, I. Al Ridhawi, and Y. Jararweh, "An intrusion detection system for connected vehicles in smart cities," *Ad Hoc Netw.*, vol. 90, Jul. 2019, Art. no. 101842.

[24] Y. Gao, H. Wu, B. Song, Y. Jin, X. Luo, and X. Zeng, "A distributed network intrusion detection system for distributed denial of service attacks in vehicular ad hoc network," *IEEE Access*, vol. 7, pp. 154560–154571, 2019.

[25] D. A. Schmidt, M. S. Khan, and B. T. Bennett, "Spline-based intrusion detection for VANET utilizing knot flow classification," *Internet Technol. Lett.*, vol. 3, no. 3, pp. 2–7, 2020.

[26] E. Min, J. Long, Q. Liu, J. Cui, Z. Cai, and J. Ma, "SU-IDS: A semi-supervised and unsupervised framework for network intrusion detection" in *Cloud Computing and Security*, 2018, pp. 322–334.

[27] Y. Yao, L. Su, Z. Lu, and B. Liu, "STDeepGraph: Spatial-temporal deep learning on communication graphs for long-term network attack detection," in *Proc. 18th IEEE Int. Conf. Trust Security Privacy Comput. Commun. 13th IEEE Int. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE)*, 2019, pp. 120–127.

[28] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Multi-stage optimized machine learning framework for network intrusion detection," *IEEE Trans. Netw. Service Manag.*, early access, Aug. 7, 2020, doi: 10.1109/TNSM.2020.3014929.

[29] K. Zdeněk and S. Jiří, "Simulation of CAN bus physical layer using SPICE," in *Proc. Int. Conf. Appl. Electron.*, 2013, pp. 8–11.

[30] S.-F. Lokman, A. T. Othman, and M.-H. Abu-Bakar, "Intrusion detection system for automotive controller area network (CAN) bus system: A review," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, no. 1, p. 184, 2019.

[31] L. Yang, R. Muresan, A. Al-Dweik, and L. J. Hadjileontiadis, "Image-based visibility estimation algorithm for intelligent transportation systems," *IEEE Access*, vol. 6, pp. 76728–76740, 2018.

[32] Y. Fraiji, L. B. Azzouz, W. Trojet, and L. A. Saidane, "Cyber security issues of Internet of Electric Vehicles," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, vol. 2018. Barcelona, Spain, Apr. 2018, pp. 1–6.

[33] B. Groza and P.-S. Murvay, "Efficient intrusion detection with bloom filtering in controller area networks," *IEEE Trans. Inf. Forensics Security*, vol. 14, pp. 1037–1051, 2019.

[34] U. E. Larson, D. K. Nilsson, and E. Jonsson, "An approach to specification-based attack detection for in-vehicle networks," in *Proc. IEEE Intell. Veh. Symp.*, Eindhoven, The Netherlands, 2008, pp. 220–225.

[35] D. Zhou, Z. Yan, Y. Fu, and Z. Yao, "A survey on network data collection," *J. Netw. Comput. Appl.*, vol. 116, pp. 9–23, Aug. 2018.

[36] A. Moubayed, A. Refaey, and A. Shami, "Software-defined perimeter (SDP): State of the art secure solution for modern networks," *IEEE Netw.*, vol. 33, no. 5, pp. 226–233, Sep./Oct. 2019.

[37] P. Kumar, A. Moubayed, A. Refaey, A. Shami, and J. Koilpillai, "Performance analysis of SDP for secure internal enterprises," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, vol. 2019. Marrakesh, Morocco, Apr. 2019, pp. 1–6.

[38] A. Moubayed and A. Shami, "Softwarization, virtualization, & machine learning for intelligent & effective V2X communications," *IEEE Intell. Transp. Syst. Mag.*, early access, Sep. 29, 2020, doi: 10.1109/MITS.2020.3014124.

[39] M. Y. Abualhoul, O. Shagdar, and F. Nashashibi, "Visible Light inter-vehicle Communication for platooning of autonomous vehicles," in *Proc. IEEE Intell. Veh. Symp.*, vol. 2016. Gothenburg, Sweden, 2016, pp. 508–513.

[40] A. Moubayed, A. Shami, P. Heidari, A. Larabi, and R. Brunner, "Edge-enabled V2X service placement for intelligent transportation systems," *IEEE Trans. Mobile Comput.*, vol. 20, no. 4, pp. 1380–1392, Apr. 2021.

[41] K. M. Faraoun and A. Boukelif, "Neural networks learning improvement using the K-means clustering algorithm to detect network intrusions," *INFOCOMP J. Comput. Sci.*, vol. 5, no. 3, pp. 28–36, 2006.

[42] S. Na, L. Xumin, and G. Yong, "Research on k-means clustering algorithm: An improved k-means clustering algorithm," in *Proc. 3rd Int. Symp. Intell. Inf. Technol. Security Informat. (IITSI)*, Jian, China, 2010, pp. 63–67.

[43] A. Moubayed, M. Injadat, A. Shami, and H. Lutfiyya, "Student engagement level in an e-Learning environment: Clustering using K-means," *Amer. J. Distance Educ.*, vol. 34, no. 2, pp. 137–156, 2020.

[44] A. Moubayed, M. Injadat, A. Shami, and H. Lutfiyya, "DNS typo-squatting domain detection: A data analytics & machine learning based approach," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Abu Dhabi, UAE, 2018, pp. 1–7.

[45] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, Nov. 2020.

[46] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Multi-split optimized bagging ensemble model selection for multi-class educational data mining," *Springer's Appl. Intell.*, vol. 50, pp. 4506–4528, Jul. 2020.

[47] M. Injadat, F. Salo, A. B. Nassif, A. Essex, and A. Shami, "Bayesian optimization with machine learning algorithms towards anomaly detection," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2018, pp. 1–6.

[48] Z. Chen *et al.*, "Machine learning based mobile malware detection using highly imbalanced network traffic," *Inf. Sci.*, vols. 433–434, pp. 346–364, Apr. 2018.

[49] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002.

[50] F. Salo, A. B. Nassif, and A. Essex, "Dimensionality reduction with IG-PCA and ensemble classifier for network intrusion detection," *Comput. Netw.*, vol. 148, pp. 164–175, Jan. 2019.

[51] L. Yu and H. Liu, "Efficiently handling feature redundancy in high-dimensional data," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2003, pp. 685–690.

[52] L. Yu and H. Liu, "Feature selection for high-dimensional data: A fast correlation-based filter solution," in *Proc. 20th Int. Conf. Mach. Learn.*, vol. 2, 2003, pp. 856–863.

[53] S. Egea, A. R. Mañez, B. Carro, A. Sánchez-Esguevillas, and J. Lloret, "Intelligent IoT traffic classification using novel search strategy for fast-based-correlation feature selection in industrial environments," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1616–1624, Jun. 2018.

[54] M. N. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Systematic ensemble model selection approach for educational data mining," *Knowl. Based Syst.*, vol. 200, Jul. 2020, Art. no. 105992.

[55] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *Proc. Int. Conf. Artif. Neural Netw.*, 1997, pp. 583–588.

[56] A. Garg and P. Maheshwari, "Performance analysis of snort-based intrusion detection system," in *Proc. 3rd Int. Conf. Adv. Comput. Commun. Syst. (ICACCS)*, vol. 1. Coimbatore, India, 2016, pp. 1–5.

[57] K. Leung and C. Leckie, "Unsupervised anomaly detection in network intrusion detection using clusters," in *Proc. 28th Aust. Conf. Comput. Sci.*, vol. 38, 2005, pp. 333–342.

[58] S. Sahu and B. M. Mehtre, "Network intrusion detection system using J48 decision tree," in *Proc. Int. Conf. Adv. Comput. Commun. Informat. (ICACC)*, 2015, pp. 2023–2026.

[59] A. Tesfahun and D. L. Bhaskari, "Intrusion detection using random forests classifier with SMOTE and feature reduction," in *Proc. Int. Conf. Cloud Ubiquitous Comput. Emerg. Technol. (CUBE)*, Pune, India, 2013, pp. 127–132.

[60] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Mach. Learn.*, vol. 63, no. 1, pp. 3–42, 2006.

[61] Y. Xia, C. Liu, Y. Y. Li, and N. Liu, "A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring," *Expert Syst. Appl.*, vol. 78, pp. 225–241, Jul. 2017.

[62] L. Yang and A. Shami, "A lightweight concept drift detection and adaptation framework for IoT data streams," *IEEE Internet Things Mag.*, early access, May 10, 2021, doi: 10.1109/IOTM.0001.2100012.

[63] M. Mohammed, H. Mwambi, B. Omolo, and M. K. Elbashir, "Using stacking ensemble for microarray-based cancer classification," in *Proc. Int. Conf. Comput. Control Elect. Electron. Eng. (ICCCEEE)*, Khartoum, Sudan, 2018, pp. 1–8.

[64] A. Feizollah, N. B. Anuar, R. Salleh, and F. Amalina, "Comparative study of k-means and mini batch k-means clustering algorithms in android malware detection using network traffic analysis," in *Proc. Int. Symp. Biometrics Security Technol. (ISBAST)*, Kuala Lumpur, Malaysia, 2015, pp. 193–197.

[65] A. Vikram and Mohana, "Anomaly detection in network traffic using unsupervised machine learning approach," in *Proc. 5th Int. Conf. Commun. Electron. Syst. (ICCES)*, Coimbatore, India, Jun. 2020, pp. 476–479.

[66] M. J. Kearns, *The Computational Complexity of Machine Learning*. Cambridge, MA, USA: MIT Press, 1990.

[67] N. Shi, X. Liu, and Y. Guan, "Research on k-means clustering algorithm: An improved k-means clustering algorithm," in *Proc. 3rd Int. Symp. Intell. Inf. Technol. Security Informat. (IITSI)*, Jian, China, 2010, pp. 63–67.

[68] D. Kim, Y. Tao, S. Kim, and A. Zeller, "Where should we fix this bug? A two-phase recommendation model," *IEEE Trans. Softw. Eng.*, vol. 39, no. 11, pp. 1597–1610, Nov. 2013.

[69] S. Yadav and S. Shukla, "Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification," in *Proc. 6th Int. Adv. Comput. Conf. (IACC)*, Bhimavaram, India, 2016, pp. 78–83.

[70] The Pandas Development Team, *pandas-dev/pandas: Pandas*, Zenodo, Geneva, Switzerland, Feb. 2020.

[71] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Nov. 2011. [Online]. Available: http://scikit-learn.sourceforge.net

[72] T. Chen and T. He, *xgboost: eXtreme Gradient Boosting, Version 0.4-2*, R Package, 2015, pp. 1–4.

[73] T. Head *et al.*, *scikitoptimize/Scikit-Optimize: V0.5.2*, Zenodo, Geneva, Switzerland, 2018.

[74] B. Komer, J. Bergstra, and C. Eliasmith, "Hyperopt-sklearn: Automatic hyperparameter configuration for scikit-learn," in *Proc. 13th Python Sci. Conf.*, 2014, pp. 32–37.

[75] N. Kaja, "Artificial intelligence and cybersecurity: Building an automotive cybersecurity framework using machine learning algorithms," Ph.D. dissertation, Dept. Elect. Comput. Eng., Univ. Michigan-Dearborn, Dearborn, MI, USA, 2019.
[76] O. Y. Al-Jarrah, C. Maple, M. Dianati, D. Oxtoby, and A. Mouzakitis, "Intrusion detection systems for intra-vehicle networks: A review," *IEEE Access*, vol. 7, pp. 21266–21289, 2019.
[77] A. Rosay, F. Carlier, and P. Leroux, "MLP4NIDS: An efficient MLP-based network intrusion detection for CICIDS2017 dataset," in *Machine Learning for Networking*, 2020, pp. 240–254.
[78] R. Panigrahi and S. Borah, "A detailed analysis of CICIDS2017 dataset for designing intrusion detection systems," *Int. J. Eng. Technol.*, vol. 7, no. 3.24, pp. 479–482, 2018.
[79] W. Elmasry, A. Akbulut, and A. H. Zaim, "Evolving deep learning architectures for network intrusion detection using a double PSO metaheuristic," *Comput. Netw.*, vol. 168, Feb. 2020, Art. no. 107042.
[80] J. H. Lee and K. H. Park, "GAN-based imbalanced data intrusion detection system," *Pers. Ubiquitous Comput.*, vol. 25, pp. 121–128, Nov. 2019.
[81] M. A. Ferrag and L. Maglaras, "DeepCoin: A novel deep learning and blockchain-based energy exchange framework for smart grids," *IEEE Trans. Eng. Manag.*, vol. 67, no. 4, pp. 1285–1297, Nov. 2020.
[82] R. Vijayanand, D. Devaraj, and B. Kannapiran, "Intrusion detection system for wireless mesh network using multiple support vector machine classifiers with genetic-algorithm-based feature selection," *Comput. Security*, vol. 77, pp. 304–314, Aug. 2018.
[83] R. Abdulhammed, M. Faezipour, H. Musafer, and A. Abuzneid, "Efficient network intrusion detection using PCA-based dimensionality reduction of features," in *Proc. Int. Symp. Netw. Comput. Commun. (ISNCC)*, 2019, pp. 1–6.
[84] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, p. 20, 2019.
[85] B. Ghojogh and M. Crowley, "The theory behind overfitting, cross validation, regularization, bagging, and boosting: Tutorial," 2019. [Online]. Available: https://arxiv.org/abs/1905.12787.

Li Yang (Member, IEEE) received the B.E. degree in computer science from Wuhan University of Science and Technology, Wuhan, China, in 2016, and the M.A.Sc. degree in engineering from the University of Guelph, Guelph, ON, Canada, in 2018. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, Western University, London, ON, Canada.

His research interests include cybersecurity, machine learning, time-series data analytics, and intelligent transportation systems.

Abdallah Moubayed (Member, IEEE) received the B.E. degree in electrical engineering from Lebanese American University, Beirut, Lebanon, in 2012, the M.Sc. degree in electrical engineering from the King Abdullah University of Science and Technology, Thuwal, Saudi Arabia, in 2014, and the Ph.D. degree in electrical and computer engineering from Western University, London, ON, Canada, in August 2018.

He is currently a Postdoctoral Associate with the Optimized Computing and Communications Laboratory, Western University. His research interests include wireless communication, resource allocation, wireless network virtualization, performance and optimization modeling, machine learning and data analytics, computer network security, cloud computing, and e-learning.

Abdallah Shami (Senior Member, IEEE) received the B.E. degree in electrical and computer engineering from the Lebanese University, Beirut, Lebanon, in 1997, and the Ph.D. degree in electrical engineering from the Graduate School and University Center, City University of New York, New York, NY, USA, in 2003.

He is a Professor with the ECE Department, Western University, London, ON, Canada, where he is also the Director of the Optimized Computing and Communications Laboratory.

Prof. Shami is currently an Associate Editor of IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE NETWORK, and IEEE COMMUNICATIONS SURVEYS AND TUTORIALS. He has chaired key symposia for IEEE GLOBECOM, IEEE ICC, IEEE ICNC, and ICCIT. He was the Elected Chair of the IEEE Communications Society Technical Committee on Communications Software from 2016 to 2017 and the IEEE London Ontario Section Chair from 2016 to 2018.