Machine Learning for Autonomous Vehicle Network Intrusion Detection

Ajay Kumar Dhul
Department of CSE
Ganga Institute of Technology & Management, Jhajjar
Rohtak Haryana
India
ajay.dhul@gmail.com

Dr. Pramod Kumar
Assistant Professor
Department of CSE
Ganga Institute of Technology & Management, Jhajjar
Rohtak Haryana
India
hod.cse@gangainstitute.com

***Abstract*— According to this study, autonomous vehicles (AV) employ a network IDS or Intrusion Detection System with tree-structured machine learning model. Techniques such as XGBoost, additional trees, decision trees and random forests are utilised in ML. A well-known classification method that uses a divide-and-conquer approach is the DT or Decision Tree. Each leaf node and decision node in a DT represents a criterion test for a particular attribute and outcome class. The decision tree class with the most votes, according to the majority voting rule, decides the categorization outcome. This is how Random Forest, an ensemble learning-based classification algorithm, works (RF). Extra Trees (ET), a comparable ensemble model, is made up of a large number of randomly generated decision trees derived through the analysis of a large volume of dataset. To improve performance and speed, the XGBoost ensemble learning system employs gradient descent and multiple decision trees. The IDS multi classification problem is solved using machine learning algorithms. The combination of feature selection and ensemble learning in the suggested technique results in high detection rates at low processing costs. The computational efficiency of well-known supervised machine learning techniques is also considered when selecting a model. The proposed method for detecting different threats in anti-virus (AV) networks produced promising results.**

Keywords—Machine Learning ML, Intrusion Detection System IDS, VANETs, Ensemble Method, SMOTE.

1. INTRODUCTION

IoV is replacing VANETs [1]. Because more cars, devices, and infrastructures are joining the conversation. VANETs turn vehicles into wireless routers or mobile nodes that connect vehicles and equipment wirelessly [2]. Autonomous vehicle technology could reduce traffic accidents and their costs. V2X connects people, buildings, and vehicles to local and wide-area cellular networks. [3]. IT research focuses on vehicle-pedestrian, vehicle-infrastructure, vehicle-network, and vehicle-infrastructure communications. IoT requires wireless devices. Gateways and firewalls lack security [3].Autonomous vehicles are vulnerable to network attacks, which can be disastrous. This article discusses network threats. Denial-of-service (DoS) attacks flood a node with unnecessary requests or messages [4]. Adversaries use GPS spoofing to impersonate legitimate users and fool nodes into thinking they have accurate geolocation information.

Port scans can steal user and system data. Hackers attempt brute-force password decryption on automotive networks and systems. [5]. Autonomous vehicles are vulnerable to intra-vehicle communication attacks and external connectivity issues (premise 5). SQL injection and cross-site scripting can exploit computer and vehicle internet interfaces (XSS). CAN bus technology lets vehicle ECUs communicate. Maintain error detection systems for continuous transmission while reducing wire cost, weight, and complexity. [6]. Every ECU uses the CAN bus, making system security vulnerable in many ways. CAN bus nodes will carry malicious messages from insecure conversations without verifying their source. Adversaries can monitor network activity and launch attacks. CAN bus injection attacks have multiple goals. CAN bus DoS and spoofing attacks can send equipment data and RPMs. 'Fuzzy attacks, which involve sending erroneous signals to vehicles, are common. Given the risks and limitations, AV systems need a defense system to prevent intrusions into their internal and external communications. Intrusion detection systems (IDSs) analyze network traffic from cars and other connected devices to identify threats. [8] Machine learning (ML) algorithms are heavily used in intrusion detection systems (IDS). This article discusses a smart IDS that works with autonomous vehicles' CAN bus and the industry standard Internet of Vehicles (IoVs). XGBoost uses many tree-based machine learning algorithms to detect intrusions. These algorithms include decision trees, additional trees, random forests, and extreme gradient boosting. A serious intrusion detection system (IDS) needs a high detection rate and calculation cost. Ensemble learning and feature selection stacking can improve accuracy and processing speed. Information security has been challenged by autonomous vehicle and IoV technology. IoVs are replacing VANETs as more devices and infrastructures join the conversation. ITSs are turning vehicles into wireless routers or mobile nodes that connect other vehicles and equipment. V2X technology allows vehicles to communicate with people, infrastructure, networks, and other vehicles via local and wide-area cellular networks. However, these advancements pose new security threats like catastrophic network attacks. DoS attacks bombard nodes with irrelevant requests or messages to take control. Nodes can be fooled by GPS spoofing. Port scans can steal personal information from computer users and systems, while brute-force password decryption is used to break into automotive networks and systems. SQL injection and XSS can exploit computer and car Internet interfaces. Autonomous vehicles can also be attacked internally. CAN bus technology lets vehicle ECUs communicate, but it also opens the system to security breaches. Insecure CAN bus conversations can carry malicious messages without verification by network nodes. Now adversaries can monitor network traffic and attack. Fuzzy attacks involve sending vehicles false signals to affect their behavior.

IDSs scan network traffic for threats and malware. XGBoost uses decision trees, additional trees, random forests, and extreme gradient boosting. Intrusion detection systems use ML algorithms. Ensemble learning and feature selection stacking can speed up and improve precision. The XGBoost IDS can detect industry-standard IoVs and autonomous vehicles' CAN buses.

2. SYSTEM DESIGN

A. Problem statement

AV systems are sensitive to a wide variety of network threats via numerous communication pathways, it is recommended that both internal and external communication networks implement comprehensive IDS systems. All IoT devices within the IoV, as well as the actual vehicle components, have been strengthened. The proposed intrusion detection system must be capable of detecting a wide range of CAN bus and external network intrusions. This research will concentrate mostly on denial-of-service attacks directed against external and internal vehicle communication networks, fuzzy and spoofing attacks directed against the CAN-bus, and sniffer, brute-force, and web assaults directed against external networks. A high detection rate is required of the intrusion detection system in order to facilitate the speedy and precise identification of the great majority of potential dangers.

B. The IDS system's architecture and an overview

The proposed IDS is integrated throughout AV system to protect both internal and external connections. To evaluate each broadcasted message and guarantee the nodes' security by identifying and thwarting threats, an IDS can be added to the CAN bus [9]. Figure 1 can be secured by installing the recommended intrusion detection system (IDS) within the gateway. The suggested AV architecture with IDS security is shown in Figure 2. External communication networks are incorporated into the proposed IDS framework [10]. The framework for IDS deployment on automotive systems is shown in Figure 1. The IDS detect this change by routing the message through the CANH signal line on the CAN bus. Any message with an ID and data field that a node receives from another CAN bus-connected device is routed through the IDS. A node delivers messages containing an ID and a data field from other CAN bus-connected devices through the IDS.

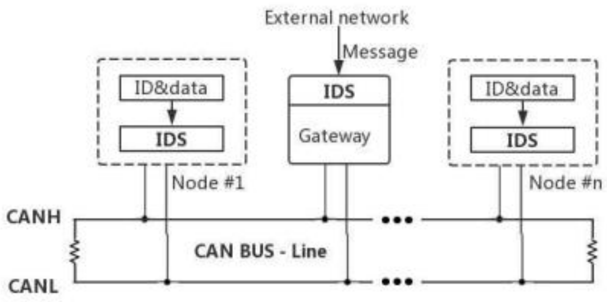


Fig. 1. AV Architecture for protected IDS System

The suggested AV architecture with IDS security is shown in Figure 2. External communication networks are incorporated into the proposed IDS framework [10]. The framework for IDS deployment on automotive systems is shown in Figure 1. The IDS detects this change by routing the message through the CANH signal line on the CAN bus. The suggested model's workflow diagram is shown below. The first step is to collect network traffic data. The second reason for oversampling is to compensate for the dataset's uneven class distribution. The following step involves selecting features based on their average relevance in order to reduce processing costs. The stacking ensemble model is then fed four basic models. After that, the final data categorization model is built.

The results show that intrusions can be detected with high precision. To evaluate the effectiveness of the proposed intrusion detection system, several well-known open-source data sets are used.

Here are some examples of the contributions:

● The threats and vulnerabilities that CAN and AV networks face are examined in this article.

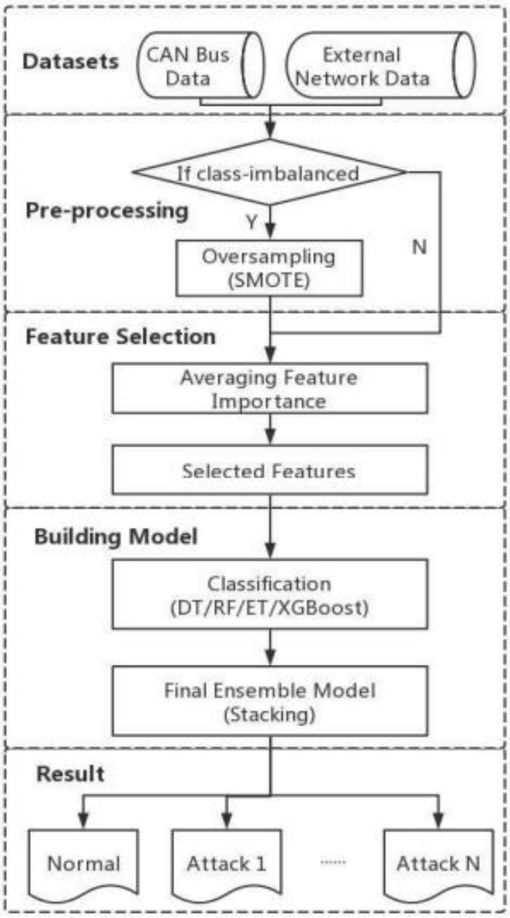


Fig. 2. Framework of the proposed IDS

- The threats and vulnerabilities that CAN and AV networks face are examined in this report.
- We provide a machine learning and ensemble learning method based on tree designs for both specialized and general networks for intelligent intrusion detection systems (IDS).
- A comprehensive architecture for gathering network traffic data is outlined before constructing an IDS.

The structure of this paper is as follows: The suggested intrusion detection system's overview and architecture are provided in the second section. Section III contains additional information about the suggested IDS framework. Section IV contains a summary of the findings, performance analyses, and feature rankings. Section V is the final section.

III. SUGGESTED IDS FRAMEWORK

A. Data pre-processing

Gathering enough network traffic data under normal and abnormal attack conditions is the first step to developing an IDS. Packet sniffers cannot build an intrusion detection system (IDS) because they lack network characteristics and pre-defined network components. An IDS for CAN bus intrusion must capture CAN messages and frames. Attacks involve frame data fields and CAN IDs. [7]

Before developing an IDS that can detect multiple attacks, network properties must be better understood because external networks are a subset of general networks and exposed to several widespread network threats. Most network indicators—TCP flag counts, segment size, active/idle state, packet size, segment length, data transfer rate, and throughput—must be considered. However, high data dimensionality may increase the computational cost of the proposed IDS. Thus, external network feature analysis is needed. Several processes would improve network data for IDS development. One-hot vector encoding, which uses a threshold to distinguish normal and abnormal data, is useful [6]. Normalized data performs better in machine learning training [3]. Each value after normalization:

$$x_n = \frac{x - \min}{\max - \min} \quad (1)$$

The initial value is x, and the maximum and minimum parameters are max and min. Network data is class-imbalanced because real-world networks are usually normal and attack-label instances are often inadequate. SMOTE and random oversampling can provide minority classes with insufficient data with low anomaly detection [11]. Random oversampling duplicates samples to increase underrepresented group sample sizes. Due to data specificity, random oversampling may overfit. SMOTE analyses minority groups and creates new samples based on their characteristics using K nearest neighbours. SMOTE provides high-quality samples for underrepresented populations.

B. The Proposed Machine Learning Approach

The proposed system's IDS uses machine learning to detect cyberattacks. The goal is to classify each network packet as normal or one of several attack classes. Decision trees, random forests, extra trees, and XGBoost are used for such problems. Decision trees divide and conquer data to classify it. Decision and leaf nodes form a decision tree. The leaf node represents the outcome class, and the decision node represents a feature selection test. Random Forest classifies better by combining multiple decision trees. Each decision tree uses a random subset of features and training samples. It selects the most supported classification outcome by aggregating all decision tree predictions.

The ensemble method Extra Trees generates multiple randomized decision trees from different data subsets. It randomly chooses features and training samples for each decision tree, like Random Forest. Instead of picking the best threshold based on data, it picks random thresholds for each feature. XGBoost, another ensemble learning method, uses gradient descent and multiple decision trees to improve performance and speed. It boosts each decision tree to correct the previous tree's errors. To avoid overfitting, a regularization term penalizes large weights.

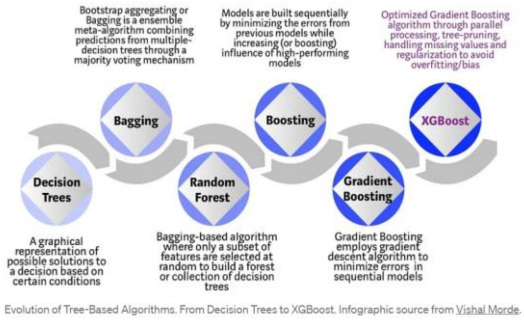


Fig. 3. Tree-Based Algorithms Evolution

Tree-based machine learning can detect network traffic cyberattacks. Random Forest, Extra Trees, and XGBoost improve classification and generalization by combining multiple decision trees. These algorithms can handle high-dimensional and noisy data, making them suitable for real-world intrusion detection.

C. Metrics for validation

Accuracy, precision, recall, and F1 score are the most important metrics used to evaluate the suggested approach [22]. Accuracy is measured by data classification accuracy. Despite excellent classification accuracy for normal data, assault detection rates may be low due to class imbalance. Evaluation uses detection rate. Divide known attack data by all unusual data to calculate the detection rate. To detect most attacks, an IDS must have high recall. The harmonic mean of recall and precision is calculated using the F1 score to evaluate strategy effectiveness. Training models also affects computer system execution.

IV. PERFORMANCE EVALUATION

A. Datasets have been condensed

The first dataset, dubbed "Car-Hacking Dataset" or "CAN-intrusion Dataset," was released in 2018 [6] to aid in the development of CAN intrusion detection systems. This study develops a comprehensive IDS that is also efficient in external communication networks using "CICIDS2017," a typical IDS data collection that includes the most recent attack scenarios. This IDS was created with "CICIDS2017." Section 3 requirements are met by the two selected data sets. Simple operations such as combining the data, removing any missing values, and reassigning the labels were performed on both datasets to improve their suitability for the creation of an IDS. The updated dataset descriptions can be found in Tables I and II, respectively.

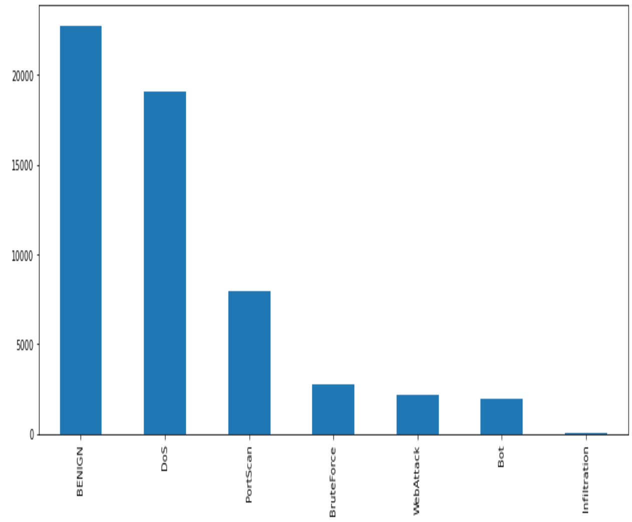


Fig 4 Plot for the count of different features of the dataset

Table 1. Raw Dataset Description:

Class_Label	**No._of_ In-stances**	**Class_ Label**	**No._of_ In-stances**
BENIGN	22731	DOS	19035
PORT SCAN	7946	BRUTE FORCE	2767
WEB AT-TACK	2180	BOT	1966
INFILTRA-TION	36	--	--

After oversampling by SMOTE (when performing train-test split values):

Table 2. Dataset Description after oversampling:

Class_Label	No._of_ Instances	Class_Label	No._of_ Instances
Benign	18184	Dos	15228
Port scan	6357	Brute force	2213
Web attack	1744	Bot	1573
Infiltration	1500	--	--

B. IDS Performance Analysis

Tables III and IV show the outcomes of tests performed on the CICIDS2017 data set to compare various strategies. Based on their performance on the test data set, the tree-based algorithms have been used in our proposed system, such as ET, RF, Decision tress as well as the XGBoost, have a high level of accuracy, as shown in Table III. Stacking, which is enabled by DT, RF, ET, and XGBoost, reduces execution time. Bagging Classifier was chosen as the second-layer meta-classifier, and the stacking ensemble model was limited to the classes ET, RF, Decision tress as well as the XGBoost. The slowest and least accurate of the 4 tree-based machine learning models is the Adaptive Booster Classifier. The precision, recall rate, and F1 score are all close to 99.65% because virtually all taught attacks can be identified using a combination of tree-based models.

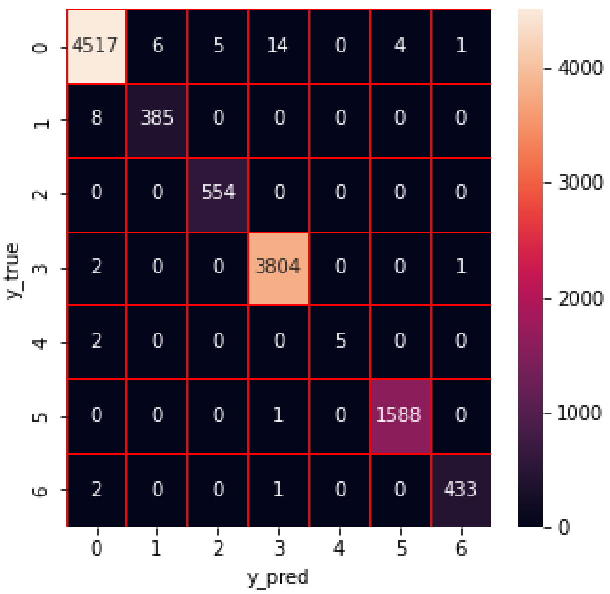


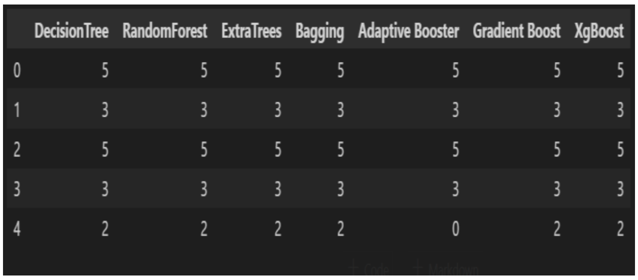
Fig 5 Confusion Matrix Heatmap of Model Predictions on Test Data

The majority of tree-based algorithms used in the CICIDS2017 data set improved in precision, detection rates, and F1 scores, with the exception of ET, as shown in Table IV. As a result, the recommended stacking method was modified to include DT, RF, and XGBoost, with the latter serving as the meta-classifier for the stacking model. Stacking achieves the highest level of accuracy (99.98%), although it is slower than tree-based models. Phase III is implemented following the training of data sets with each characteristic. As shown in Tables III and IV, the RF and XGBoost models performed the best in relation to the data sets. After selecting the features, we examined these two single-base models because they execute faster than ensemble models. The "Stacking Algorithm (XGB)" and stacking outcomes for the feature-selected data set are presented in Table III. After the selection of the required features, the Random Forest as well as the stacking models reduced execution time by 99.98% while maintaining excellent accuracy. Although the accuracy of the XGBoost has been reduced by 0.08% and for stacking models, reduced by 0.04%, their execution times were reduced by 39.2% and 38.2%, respectively. The CICIDS2017 data collection contains 36 of the 78 qualifying characteristics. To increase execution speed without sacrificing precision, the IDS selects common properties from trees.

Table 3. Performance Evaluation on Raw Datasets.

Method	Train Accuracy (%)	Test Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Decision Tree Classifier	**99.98**	**99.58**	**99.59**	**99.58**	**99.58**
Random Forest Classifier	**99.84**	**99.58**	**99.58**	**99.58**	**99.58**
Extra Trees Classifier	**99.98**	**99.28**	**99.28**	**99.28**	**99.28**
Bagging Classifier	**99.98**	**99.65**	**99.65**	**99.65**	**99.65**
Adaptive Booster Classifier	**56.78**	**56.76**	**67.23**	**55.76**	**60.00**
Gradient Boost Classifier	**99.73**	**99.54**	**99.53**	**99.54**	**99.53**
XGB Classifier	**99.48**	**99.42**	**99.42**	**99.42**	**99.42**

After applying all the above 7 algorithms I have tried another method called stacking method which is nothing but a combination of all the 7 algorithms together.



	DecisionTree	RandomForest	ExtraTrees	Bagging	Adaptive Booster	Gradient Boost	XgBoost
0	5	5	5	5	5	5	5
1	3	3	3	3	3	3	3
2	5	5	5	5	5	5	5
3	3	3	3	3	3	3	3
4	2	2	2	2	0	2	2

Fig. 6 Predictions of 7 base models on the training data used for constructing a new ensemble model

Table 4. Performance Evaluation on Raw Dataset after Stacking.

Method	Train Accuracy (%)	Test Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Stacking Algorithm (XGB Classifier)	99.98	99.57	99.57	99.57	99.57

We checked that the labels are highly imbalanced in nature, while label values in the dataset is around 18,000 and the other label value is just 29. So, we have to balance the datasets, hence we applied an oversampling method. Specifically, we can say that we applied an oversampling method using SMOTE. After applying oversampling, we just add around 1500 labels to the least.

Table 5. Performance Evaluation on Dataset after Oversampling.

Method	Train Accuracy (%)	Test Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Decision Tree Classifier	99.96	99.52	99.52	99.52	99.52
Random Forest Classifier	99.94	99.59	99.59	99.59	99.59
Extra Trees Classifier	99.97	99.57	99.57	99.57	99.57
Bagging Classifier	99.96	99.52	99.52	99.52	99.52
Adaptive Booster Classifier	56.86	55.74	67.25	55.74	59.98

Gradient Boost Classifier	99.63	99.49	99.49	99.49	99.49
XGB Classifier	99.40	99.32	99.32	99.32	99.32

After applying all the above 7 algorithms I have tried another method calling stacking method which is nothing but a combination of all the 7 algorithms together.

Table 6. Performance Evaluation of Stacking on Dataset after Oversampling.

Method	**Train Accuracy (%)**	**Test Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
Stacking Algorithm (XGB Classifier)	**99.98**	**99.54**	**99.54**	**99.54**	**99.54**

C. Feature analysis

To analyze the characteristics, subsets of each assault were subjected to the suggested method for feature selection.

Table V shows the relative importance of the three most important factors for each assault. The target port may indicate a DoS, brute force, online, or botnet attack, according to Table V. Another important factor to consider is the size of the box. The average packet size, for example, would indicate a DoS, port scan, or web attack. Forward packet length is used to represent DoS, web, and botnet attacks, while reverse packet length is used to represent port scan, brute force, and intrusion attempts. A port scan or brute-force attack is indicated by the number of pushing flags and the difference in packet length between forward and reverse. IDSs can be created for a variety of purposes, such as detecting a specific type of attack, by selecting relevant features from a list after obtaining the feature priority list for each attack. Network administrators can still monitor the situation and identify the most critical elements. An attack is likely to be detected if the attributes of a target change in an unusual way.

Table 7. Importance of each feature by attack

Label	**Feature**	**Weight**
DoS	$BwdPacketLengthStd$	0.1723
	$AveragePacket$	0.1211
	$DestinationPort$	0.0785
Port Scan	$TotalLengthofFwdPacket$	0.3020
	$AveragePacket$	0.1045
	$PSHFlagCount$	0.1019
Brute-Force	$DestinationPort$	0.3728
	$FwdPacketLengthMin$	0.1022
	$PacketLengthVariance$	0.0859
	$InitWinbytesbackward$	0.2643

Web-Attack	*AveragePacket*	0.1650
	DestinationPort	0.0616
Botnet	*DestinationPort*	0.2364
	BwdPacketLengthMean	0.1240
	AvgBwdSegment	0.1104
Infiltration	*TotalLengthofFwdPacket*	0.2298
	SubflowFwdBytes	0.1345
	DestinationPort	0.1149

V. CONCLUSION

Despite the fact IDS are one of the most efficient ways to safeguard automotive networks and detecting newly launched network attacks, connected and autonomous vehicles are still vulnerable to a variety of threats. To detect external network vulnerabilities, CAN buses are outfitted with an intrusion detection system (IDS). This system employs tree-based machine learning techniques. Features were chosen to reduce class imbalance and computational cost using tree-based averaging and SMOTE oversampling. The proposed method outperforms previously published methods by 2% to 3% in terms of precision, detection rate, and F1 score. Unlike previous methods that relied solely on a single data set, our research develops an IDS capable of detecting a variety of attacks during each run. The trains were 99.98% accurate in both the pre- and post-oversampling datasets. The performance of the suggested IDS on the CICIDS2017 data set can be significantly improved by fine-tuning the hyper-parameters of various approaches, such as particle swarm optimization and Bayesian optimization.

References

[1] Awang, K. Husain, N. Kamel, and S. Assa, "Routing in Vehicular Ad-hoc Networks: A Survey on Single- and Cross-Layer Design Techniques and Perspectives," in IEEE Access, vol. 5, pp. 9497-9517, 2017.

[2] F. Yang, S. Wang, J. Li, Z. Liu, and Q. Sun, "An overview of Internet of Vehicles," China Commun., vol. 11, no. 10, pp. 1-15, 2014.

[3] K. M. Ali 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.

[4] H. P. Dai Nguyen and R. Zoltn, "The Current Security Challenges of Vehicle Communication in the Future Transportation System," SISY 2018 - IEEE 16th Int. Symp. Intell. Syst. Informatics, Proc, Subotica, pp. 161-166, 2018.

[5] Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," no. Cic, pp. 108-116, 2018.

[6] E. Seo, H. M. Song, and H. K. Kim, "GIDS: GAN based Intrusion Detection System for In-Vehicle Network," 2018 16th Annu. Conf. Privacy, Secur. Trust, pp. 1-6, 2018.

[7] H. Lee, S. H. Jeong, and H. K. Kim, "OTIDS: A novel intrusion detection system for in-vehicle network by using remote frame," Proc. 15th Annu. Conf. Privacy, Secur. Trust. PST 2017, pp. 57-66, 2017.

[8] Alshammari, M. A. Zohdy, D. Debnath, and G. Corser, "Classification Approach for Intrusion

Detection in Vehicle Systems," Wirel. Eng. Technol., vol. 09, no. 04, pp. 79-94, 2018.

[9] A. Groza and P. Murvay, "Efficient Intrusion Detection With Bloom Filtering in Controller Area Networks," IEEE Trans. Inf. Forensics Security, vol. 14, no. 4, pp. 1037-1051, 2019.

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

[11] N.V. Chawla, K.W. Bowyer, L.O. Hall, and W.P. Kegelmeyer, "SMOTE: Synthetic Minority Over-Sampling Technique," J. Artificial Intelligence Research, vol. 16, pp. 321-357, 2002.

[12] A. Moubayed, M. Injadat, A. Shami and H. Lutfiyya, "DNS TypoSquatting Domain Detection: A Data Analytics & Machine Learning Based Approach," 2018 IEEE Glob. Commun Conf., Abu Dhabi, United Arab Emirates, pp. 1-7, Dec. 2018.

[13] D. M. Manias, M. Jammal, H. Hawilo, A. Shami, et. al., "Machine Learning for Performance-Aware Virtual Network Function Placement," 2019 IEEE Glob. Commun. Conf., Waikolao, HI, USA, Dec. 2019.

[14] Y. Ping, "Hybrid fuzzy SVM model using CART and MARS for credit scoring," 2009 Int. Conf. Intell. Human-Machine Syst. Cybern. IHMSC 2009 , vol. 2, pp. 392-395, 2009.

[15] M. Injadat, F. Salo, A. B. Nassif, A. Essex, and A. Shami, "Bayesian Optimization with Machine Learning Algorithms Towards Anomaly Detection," 2018 IEEE Glob. Commun Conf., pp. 1-6, 2018.

[16] K. Arjunan and C. N. Modi, "An enhanced intrusion detection framework for securing network layer of cloud computing," ISEA Asia Secur. Priv. Conf. 2017, ISEASP 2017, pp. 1-10, 2017.

[17] D. Zhang, L. Qian, B. Mao, C. Huang, B. Huang, and Y. Si, "A DataDriven Design for Fault Detection of Wind Turbines Using Random Forests and XGboost," in IEEE Access, vol. 6, pp. 21020-21031, 2018.

[18] L. Yang, R. Muresan, A. Al-Dweik and L. J. Hadjileontiadis, "ImageBased Visibility Estimation Algorithm for Intelligent Transportation Systems," in IEEE Access, vol. 6, pp. 76728-76740, 2018.

[19] MJ. Kearns, "The computational complexity of machine learning," MIT press, 1990.

[20] Pattawaro and C. Polprasert, "Anomaly-Based Network Intrusion Detection System through Feature Selection and Hybrid Machine Leaming Technique," 2018 16th Int. Conf. ICT&KE, Bangkok, pp. 1-6, 2018.

[21] M. Mohammed, H. Mwambi, B. Omolo, and M. K. Elbashir, "Using stacking ensemble for

[22] F. Salo, M. Injadat, A. B. Nassif, A. Shami, and A. Essex, "Data mining techniques in intrusion detection systems: A systematic literature review," in IEEE Access, vol. 6, pp. 56046-56058, 2018.

[23] 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.

[24] S. U. Jan, S. Ahmed, V. Shakhov, and I. Koo, "Toward a Lightweight Intrusion Detection System for the Internet of Things," in IEEE Access, vol. 7, pp. 42450-42471, 2019.