Multiple Attack Detection Using SHAP and Heterogeneous Ensemble Model in the UAV's Controller Area Network

Young-Woo Hong and Dong-Young Yoo *

Posted Date: 30 April 2024

doi: 10.20944/preprints202404.2018.v1

Keywords: controller area network (CAN); shapley additive explanations (SHAP); machine learning(ML); deep learning(DL); unmanned aerial vehicles (UAVs)

Article

Multiple Attack Detection Using SHAP and Heterogeneous Ensemble Model in the UAV's Controller Area Network

Young-Woo Hong and Dong-Young Yoo *

Department of Software Convergence and Communication Engineering, Sejong Campus, Hongik University, 2639, Sejong-ro, Jochiwon-eup, Sejong, 30016, Republic of Korea; hyw1021@g.hongik.ac.kr
* Correspondence: ydy@hongik.ac.kr; Tel.: +82-44-860-2305

Abstract: Recently, methods to detect DoS and spoofing attacks that occur on in-vehicle networks using CAN Protocol are being studied through deep learning models such as CNN, RNN, and LSTM. These studies have produced significant results in the field of In-Vehicle Network attack detection using deep learning models. In addition, significant results are being achieved through research on applying time series-based deep learning models such as LSTM to detect DoS attacks and replay attacks occurring in in-drone networks by expanding them to drones using the UAVCAN protocol. In this paper, we conducted an experiment to detect in-drone network attacks through non-time series analysis using machine learning models and deep learning models, and through appropriate learning for each attack type, it can also be analyzed through non-time series analysis. The results showed that it was possible to detect attacks.

Keywords: **c**ontroller area network (CAN); shapley additive explanations (SHAP); machine learning(ML); deep learning(DL); unmanned aerial vehicles (UAVs)

1. Introduction

Recently as unmanned aerial vehicle (UAV) and internet of things (IoT) technologies develop, the use of UAVs is expanding to weather observation, agriculture, and military purposes. However not only physical signal attacks such as GPS Signal Spoofing and Jamming Signals, but also malware and malicious communication [1]. Additionally, attacks through software and communication protocols such as spoofed signals also occur. These attacks are classified as a major threat because they can also be used against unmanned aerial vehicles (UAVs). These attacks are classified as a major threat because they can also be used against unmanned aerial vehicles (UAVs). Accordingly, cyber-attack attempts targeting unmanned aerial vehicles (UAVs) are increasing, and research to detect them is actively underway [2,3]. Representative attack types include spoofing, DoS, and Replay attacks on CAN protocols against UAVs.

Research is also underway using machine learning (ML) and deep learning (DL) models to detect network intrusions occurring in the CAN protocol, but performance is not stable because the dataset's feature learning process of ML/DL models cannot be verified. There is a limitation that Researcher cannot utilize the patterns analyzed by the ML/DL model. In addition, stacking techniques that ensemble different types of models are being attempted to improve detection performance, but problems are occurring that result in lower detection performance. These limitations have recently become possible to analyze the feature importance and SHAP value of ML/DL models through Lunderberg's explainable artificial intelligence (XAI) research [5,7], and by applying the SHAP technique that can specifically analyze ML/DL models. Solving cases [8,9] are also emerging.

Therefore, we compared the binary detection performance of single models and the binary detection performance of ensemble models for each type of attack that occurs in an in-drone network, and estimated the cause of the performance difference between models through SHAP analysis.

 doi:10.20944/preprints202404.2018.v1

This paper consists of a total of 5 chapters. In chapter 2 the attack scenario that generated the related work and dataset for CAN protocol, ML/DL model, and SHAP was analyzed. The chapter 3 analyzes the dataset using pearson correlation and SHAP techniques and performs performance evaluation by learning single models for a single attack. The chapter 4 experiments with a model that performs binary classification on whether network intrusion occurs by combining models with excellent performance and analyzes the results. The chapter 5 describes the conclusions and limitations of this study.

2. Related Works

2.1. Controller Area Network (CAN) Protocol

The CAN protocol [4] is a message-oriented protocol for communication between the ECU, sensors, and control equipment in the car. Unlike the existing UART's Point to Point method, which used a 1:1 dedicated line for data communication, it uses the multi master method to reduce the weight of the car by reducing the required wiring. And the price could be reduced.

Table 1. Classification of tasks by Layer.

Layer	Description
Application	Performs vendor-defined tasks
Object (Presentation)	Performs message processing
Transfer	Performs message transmission reception and Detects signal defects and message errors
Physical	Defines how to convert physical signals such as signal level and signal optimization

To give instructions directly to the drone, UAVCAN payload according to CAN 2.0 B is used, and the structure is shown in Figure 1. And the task to be performed by the drone or status information is stored in the transfer payload, which consists of a total of 8 bytes. Therefore, Fuzzy and Relay attacks on drones occur centered on CAN payload packets.

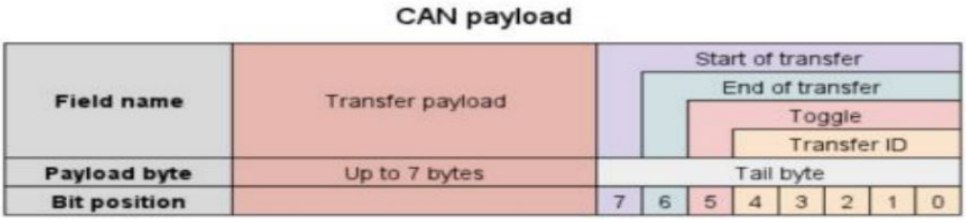


Figure 1. UAVCAN payload Frame [10].

2.2. Network Intrusion Detection Model for CAN Protocol

There are mainly five types of attacks that mainly occur in the UAVCAN protocol. [22] Denial of service (DoS) is an attack that paralyzes system resources and delays work by injecting a large amount of data into a specific system or network. This paper targets DoS of the flooding method, which injects messages into the CAN Bus at 15ms intervals.

Fuzzy is an attack in which an attacker injects a message using dictionary or brute forcing method to infer a valid CAN ID[23].

 doi:10.20944/preprints202404.2018.v1

Table 2. CAN attack Machine Learning-Based detection Research.

Model	Paper	Platform
Support Vector Machine	Tanksale, V. (2019, November) [11]	In-Vehicle Network
Random Forest	Alsoliman, A., Rigoni, G., Callegaro, D., Levorato, M., Pinotti, C. M., & Conti, M. (2023) [12],	In-Vehicle Network
	Moulahi, T., Zidi, S., Alabdulatif, A., & Atiquzzaman, M. (2021) [13]	In-Vehicle Network
Deep Neural Network	Kang, M. J., & Kang, J. W. (2016) [11]	In-Vehicle Network
CNN	Javed, A. R., Ur Rehman, S., Khan, M. U., Alazab, M., & Reddy, T. (2021) [14]	In-Vehicle Network
CNN+LSTM	Kou, L., Ding, S., Wu, T., Dong, W., & Yin, Y. (2022)[15]	In-Vehicle Network
Convolutional Neural Network	Song, H. M., Woo, J., & Kim, H. K. (2020) [16]	In-Vehicle Network
Recurrent Neural Network	Tariq, S., Lee, S., Kim, H. K., & Woo, S. S. (2020) [17]	In-Vehicle Network
LSTM (Long Short-Term Memory) Model	Qin, H., Yan, M., & Ji, H. (2021) [18]	In-Vehicle Network
	Tlili, F., Ayed, S., & CHAARI FOURATI, L. (2023, August) [19]	In-Drone Network
GAN	Seo, E., Song, H. M., & Kim, H. K. (2018, August) [20]	In-Vehicle Network
Ensemble Learing Model	Khan, M. H., Javed, A. R., Iqbal, Z., Asim, M., & Awad, A. I. (2024) [21]	In-Vehicle Network

Replay is an attack that intercepts a valid message and maliciously retransmits it, disguising it as a valid message and repeating a specific action. [24] Spoofing is an attack that disguises the CAN ID of a specific message and tricks other nodes in the CAN Bus into performing the task intended by the attacker. Impersonation is an attack that disguises itself as an appropriate node in CAN communication and performs malicious attacks or data modification on other nodes. Research on detect model for network attacks using CAN packets are shown in Figure 2.

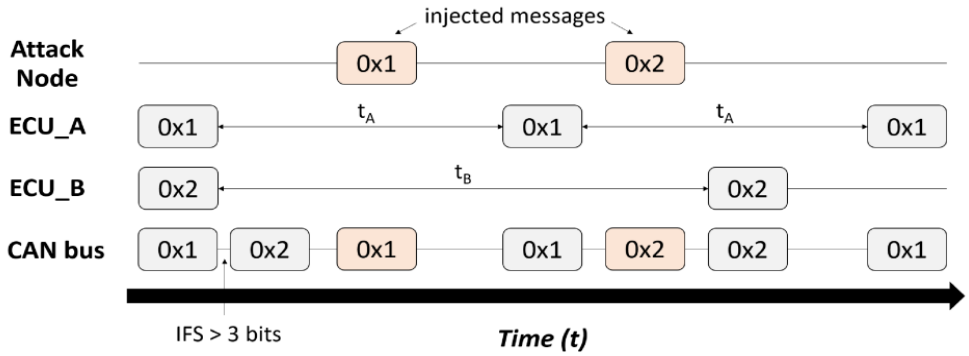


Figure 2. Diagram of attack through message injection in CAN Protocol [17].

2.3. SHAP

The SHAP (SHapley Additive exPlanations) [7] is a method of calculating SHAP values for each feature in a machine learning model, helps humans to understand the influence of features on the machine learning model. The SHAP value is the Shapley value for a feature value which is calculated using the conditional expected value function of the machine learning model. The Shapley value is a solution concept in cooperative game theory that distributes the total gain obtained through cooperation among game participants on each participant's marginal contribution.

 doi:10.20944/preprints202404.2018.v1

Shapley regression value assigns importance (importance value) to each variable based on how much it affects the model's performance when included in learning.

This is Shapley Interaction Value of an equation:

$$\phi_i(f,x) = \sum_{z' \subseteq x'} \frac{|z'|!(M-|z'|-1)!}{M!}\left[f_x(z') - f_x(z' \backslash i)\right] \quad , \quad (1)$$

where f is the model, M is the number of x′, and x′ is simplified input that maps to the original input through a mapping function x = hx(x′).

Hx maps 1 or 0 to the original input space, where 1 indicates that the input is included in the model while 0 indicates exclusion from the model. |z′| is the number of non-zero entries in z′and z′ ⊆ x′ represents all z′vectors where the non-zero entries are a subset of the nonzero entries in x′. Feature importance through Shapley Value is calculated using the following equation (2).

This is Feature Importance of an equation:

$$I_j = \sum_{i=1}^{n} |\phi_j^{(i)}| \quad , \quad (2)$$

The Shapley value assigns an importance value to each feature that represents the effect on the model prediction. The effects of the i-th feature is computed as the difference between a model trained with the i-th feature and another model trained with the feature withheld on the current input. Since the effect of withholding a feature depends on other features in the model, the preceding differences are computed for all possible subsets z′\i. As a result, the Shapley value is the weighted average of all possible differences, a unique measure of additive feature attribution method that satisfies all three axioms (local accuracy, missingness and consistency). The SHAP value in machine learning is designed to closely align with the Shapley value, using the conditional expectations to define simplified inputs.[9] Feature importance in a linear model with multicollinearity. Although multicollinearity is a property that violates the independence assumption in linear models, it is mentioned that the Shapley regression value is a value that can be used even in linear models with multicollinearity.

2.4. Explainable Artificial Intelligence (XAI)

According to Capuano, N., Fenza, G., Loia, V., & Stanzione, C. (2022) [25], Artificial Intelligence Models such as Machine Learning and Deep Learning have important impacts such as Cyber Security due to the opacity of Internal Mechanisms. It was of limited use in areas where decisions were needed. However, if explainability is provided for the AI Model provided by techniques such as SHAP [5], LIME [26], a novel auto-encoding-based scheme for LSTM model, and sufficient framework research is followed, it can satisfy the transparency required in the cyber security area. post-analysis of AI models and reduction of workload through AI models can be expected. Additionally, in the study [27] Current As a result of reviewing related techniques, it was evaluated that XAI has the potential to develop into TAI through framework research, although there are limitations such as "post-explainability, replicability of methodology" and lack of "integrated understanding."

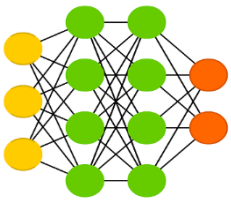
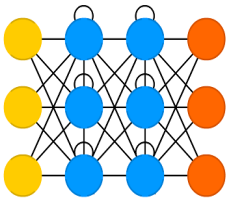
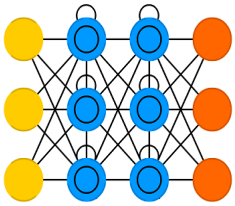
2.5. Deep Neural Network Model

The DNN (Deep Neural Network) model is a representative deep learning model and consists of several hidden layers between the input layer and the output layer. These DNN models can be applied to classification and regression problems, and a representative DNN model application study is the YouTube video recommendation algorithm [28].

On the other side, the RNN (Recurrent Neural Network) model is similar to the DNN model, but the hidden layer is composed of Recurrent Cells, so it can store past information and make predictions about the sequence. When learning long-term relationships, these RNN models have the problem that the weight of the model becomes extremely small or large, so the LSTM model that introduces Memory Cell instead of Recurrent Cell is mainly used for time series analysis.

 doi:10.20944/preprints202404.2018.v1

Table 3. Diagram of Deep Learning Models [29].

Legend of Layers	DNN	RNN	LSTM
Input Cell Hidden Cell Output Cell Recurrent Cell Memory Cell			

2.6. Attack Scenario Analysis

2.6.1. Flooding Attack Scenario Analysis

Flooding attack is a type of Denial of Service (DoS) attack that consists of two frames and injects them into the CAN bus. It is performed repeatedly in short cycles, delaying the transmission of messages on the CAN bus, and consuming the resources of the target ECU, thereby interfering with the service.

The following Figure 3 is a function written in Python 3 code for the Flooding Attack Mechanism used when creating the dataset for Scenario types 01 & 02.

```
def floodingAttack(endTime):
  frame1 = [0xA6, 0x35, 0, 0, 0, 0, 0, 0x80]
  frame2 = [0, 0, 0, 0, 0, 0, 0x60]
  idx = 0

  while True:
    frame1[-1] = (idx)%0x20 + 0x80
    frame2[-1] = (idx)%0x20 + 0x60
    try:
      msg1 = can.Message(arbitration_id=0x05040601, data=frame1)
      msg2 = can.Message(arbitration_id=0x05040601, data=frame2)
      bus.send(msg1)
      bus.send(msg2)
    except can.CanError:
      print("Message Not Send")
    time.sleep(0.005)
    idx += 1
    if time.time() > endTime:
      break
```

Figure 3. Flooding attack's Mechanism [24].

The Flooding Attack Mechanism is executed at 0.005 second intervals, generates a payload consisting of 7 bytes and 8 bytes of data, respectively, modifies the Message ID to '0x05040601', and injects it into the CAN Bus. Therefore, the Motor ECU processes a payload consisting of 8 Bytes, and there may be a delay in processing the Main ECU's message.

Flooding attack process analysis over time was performed as shown in Table 4. A flooding attack occurred between 50sec and 90sec, but the flight was normal. However, the motor stopped in the 90sec to 120sec and 130sec to 160sec sections due to flooding attack.

Table 4. The Method Proceeds of Flooding attack Scenario (Type 01 & 02).

TimeStamp (sec)	10/20	30/40/50	60/70/80/90	100	110/120	130	140/150/160	170/180
status	Booting	Take Off	Hovering	Motors Stop	Motors Stop	Hovering	Motors Stop	Landing
Label	Normal	Normal	Flooding	Normal	Flooding	Floodng	Normal	Flooding	Normal

 doi:10.20944/preprints202404.2018.v1

Table 5. Massage injection progresses according to time sequence on CAN Bus.

ECU \ Time(s)	1	2	3	4	5	6	7	8
Attacker	-	-	0x1	-	-	0x2	-	-
Main ECU	0x1	-	-	-	0x1	-	-	0x1
Motor ECU	0x2	-	-	-	-	-	0x2	-
CAN bus	0x1	0x2	0x1		0x1	0x2	0x2	0x1

The Main ECU and Motor ECU exchange messages on the CAN bus, and at this time, the attacker injects messages into the CAN bus. Therefore, the Motor ECU does not receive messages from the Main ECU, or processing delays occur, causing the UAV's motor to stop operating.

Table 6. Structure of Flooding Payload.

Byte	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]
8	08	A6	35	00	00	00	00	00	-	166	53	00	00
7	07	00	00	00	00	00	00	00	Null	00	00	00	00

There are two types of payload injected by the attacker, of which the payload with a data length of 8 bytes stores the Transfer ID in the 9th data byte. Transfer ID is an Integer value used by message type and destination node to distinguish it from other messages, and the payload uses the Transfer ID value of the previously transmitted message plus 1 as the new Transfer ID. Currently, the Transfer ID of the first Flooding Payload starts from 80.

The main features of flooding payload are three: First, the Data Length is 7 and 8. Second, the 9th Data Byte storing the Transfer ID increases by 1. Third, 6th, and 7th Data Byte is always 00.

2.6.2. Fuzzy Attack Scenario Analysis

Fuzzy attacks in Scenario Type 03 & 04 perform message injection by changing the message composition until the attacker obtains the desired information or action. For the injected message, the message id is set to '0x05040601', the data byte configuration is changed by brute forcing to a value between 0 and 255, and the message is injected at 0.001 second intervals. This fuzzy attack mechanism was written as python code in Figure 4.

```
def fuzzyAttack(endTime):
  sign = 0x217f5c87d7ec951d
  sign = sign.to_bytes(8, byteorder="little")
  tmp_crc = transCRC(sign)
  idx=0

  while True:
    payload0 = [random.randint(0, 255) for x in range(5)]
    payload1 = [random.randint(0, 255) for x in range(6)]
    frame0, frame1 = data_gen(payload0, payload1, tmp_crc)
    frame0[-1] = (idx)%0x20 + 0x80
    frame1[-1] = (idx)%0x20 + 0x60
    try:
      msg1 = can.Message(arbitration_id=0x05040601, data=frame0)
      msg2 = can.Message(arbitration_id=0x05040601, data=frame1)
      bus.send(msg1)
      bus.send(msg2)
    except can.CanError:
      print("Message Not Send")
    time.sleep(0.001)
    idx += 1
    if time.time() > endTime:
      break
```

Figure 4. Fuzzy Attack mechanism [24].

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 30 April 2024 doi:10.20944/preprints202404.2018.v1

Table 7 is written according to Timestamp of the UAV status during the Fuzzy attack Scenario experiment. A situation occurred in which the UAV's motor stopped in all sections where the fuzzy attack occurred.

Table 7. The Method Proceeds of Fuzzy attack Scenario (Type 03 & 04).

Time Stamp(sec)	10/20	30/40/50	60/70/80	90	100/110	120	130	140/150 /160	170/180
Status	Booting	Hovering	Hovering Motors stop	Hovering	Hovering Motors stop	Hovering Motors stop	Hovering	Hovering Motors stop	Landing
Label	Normal		Fuzzy	Normal	Fuzzy	Fuzzy	Normal	Fuzzy	Normal

2.6.3. Replay Attack Scenario Analysis

Replay attack is an attack in which an attacker collects normal messages in advance and repeatedly retransmits them to induce a specific action. Scenario types 05 & 06 perform this Replay attack according to the time when the message should be transmitted.

Replay attack Scenario differs from other attack scenarios in that the attack conditions for types 05 and 06 are somewhat different. Table 8 is shows the difference in attack conditions between the two scenarios.

Table 8. Dataset comparison by scenario.

Scenario type	Number of Attack	Interval(s)	Total time(s)	Propotion of Label(normal:attack)
05	3	0.005	210	2.5:1
06	4	0.005	280	2:1

The Replay attack mechanism used in Scenario types 05 & 06 is written as python code in Figure 5.

```
def replayAttack(endTime):
  idx = 0
  replayStart = time.time()

  while True:
    while time.time() - replayStart >= frames[idx][0]-frames[0][0]:
      try:
        msg = can.Message(arbitration_id=0x05040601, data=frames[
  idx][1])
        bus.send(msg)
      except can.CanError:
        print("Message Not Send")
      idx += 1
      if idx >= len(frames):
        return
    if time.time() > endTime:
      break
```

Figure 5. Relay Attack mechanism [24].

3. Dataset Analysis and Preprocessing

In this chapter, the dataset for each scenario type is analyzed. By comparing the SHAP analysis results of the attack scenario and the ML model, we confirmed whether the ML model effectively analyzes patterns within the dataset.

3.1. Dataset Analysis for Each Datasets with SHAP Analysis

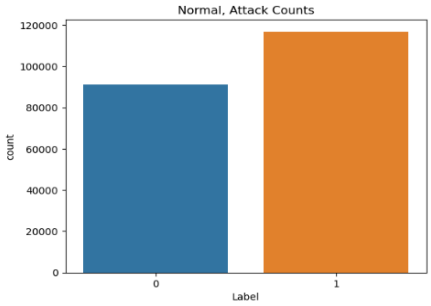
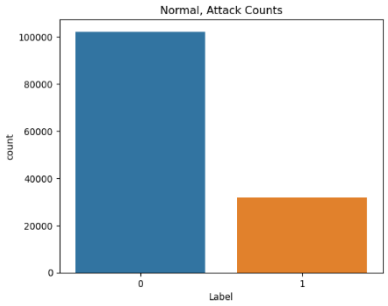
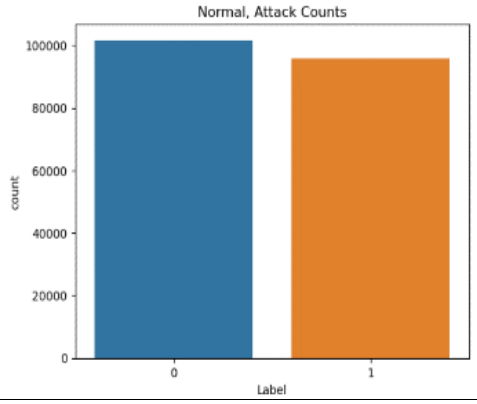
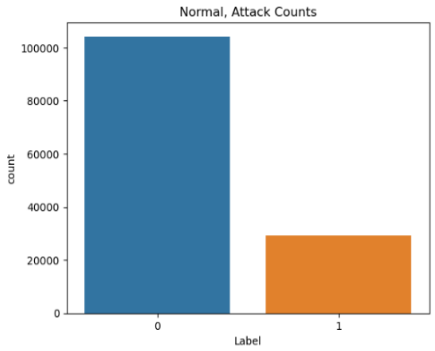
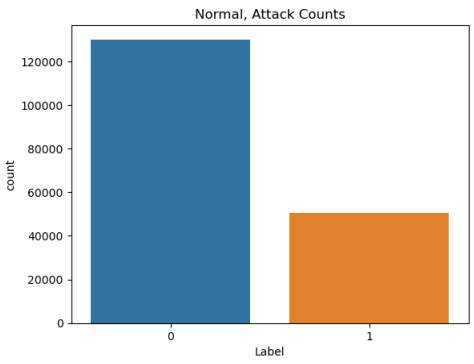
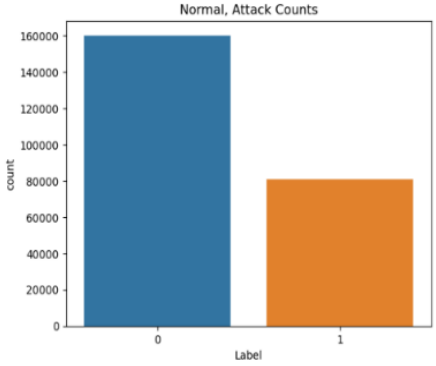
In this chapter, normal/attack data ratio, pearson correlation coefficient analysis, and SHAP value analysis with Tree Explainer (XG Boost) were performed on the dataset for each scenario type to measure the SHAP interaction value between Feature Importance and Important Features.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 30 April 2024 doi:10.20944/preprints202404.2018.v1

The analyzed dataset was preprocessed as follows. Train/Test datasets split was split at a 7:3 ratio under the condition of stratifying the label. Timestamps was included for the time series models LSTM and Simple RNN model, and data sets with timestamps rows removed were used for ML models. Additionally, in all scenario type datasets, there was an error of less than 2 seconds in timestamp between the experiment for data collection and the collected data set.

Attack dataset proportion for each scenario type is as follows Table 9.

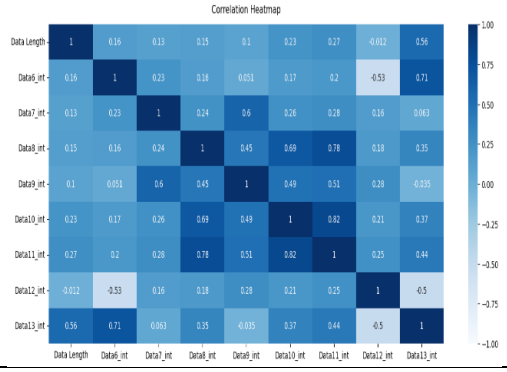
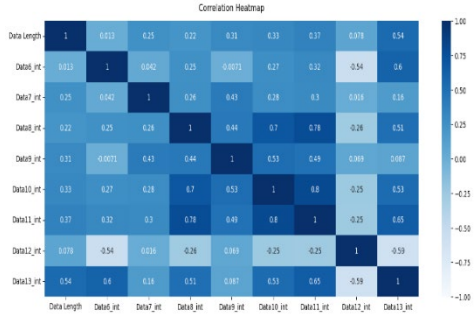
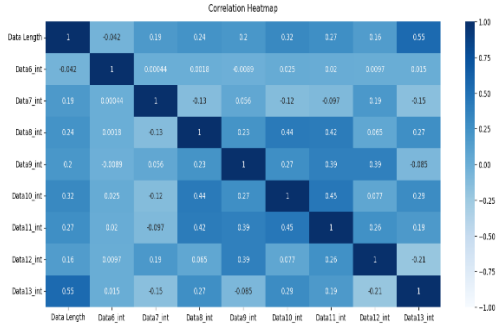
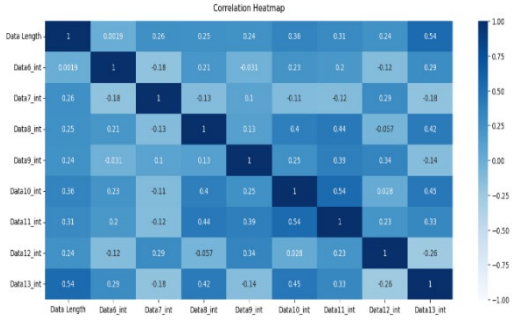
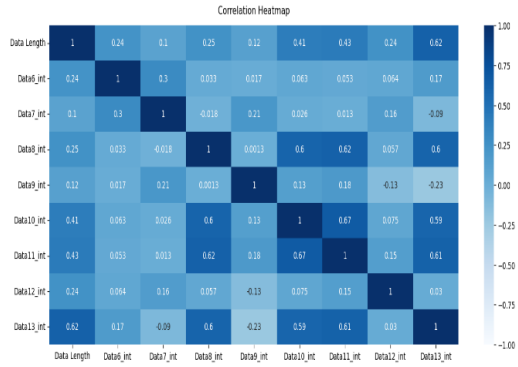
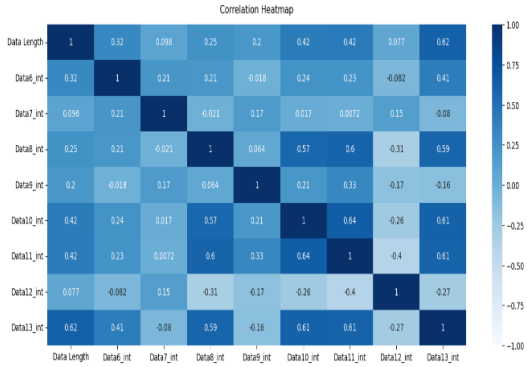
Table 9. Single Attack Dataset Proportion.

Type 01. Dataset Proportion by Label	Type 02. Dataset Proportion by Label
	
Type 03. Dataset Proportion by Label	**Type 04. Dataset Proportion by Label**
	
Type 05. Dataset Proportion by Label	**Type 06. Dataset Proportion by Label**
	

Normal is labeled 0 and attack is labeled 1. Types 01 and 03 have relatively balanced ratios, and Types 02, 04, 05, and 06 have relatively unbalanced ratios.

 doi:10.20944/preprints202404.2018.v1

Table 10. Pearson Correlation Heatmap of dataset by Scenario Type.

Pearson Correlation Heatmap (Type 01 Dataset)	Pearson Correlation Heatmap (Type 02 Dataset)
	
Pearson Correlation Heatmap (Type 03 Dataset)	**Pearson Correlation Heatmap (Type 04 Dataset)**
	
Pearson Correlation Heatmap (Type 05 Dataset)	**Pearson Correlation Heatmap (Type 06 Dataset)**
	

As a result of Pearson correlation analysis, the data length of all scenario datasets shows a high correlation coefficient with other data bits. It is assumed that the composition of the payload varies depending on the data length.

3.1.1. Flooding Scenario Dataset Analysis

As a result of SHAP analysis of flooding scenario datasets (Scenario Type 01 and Type 02), the 9th and 7th data byte and data length were derived as the features with high importability among all features. These results are consistent with the flooding payload analysis analyzed in Chapter 3.1.1, and it was derived from this that the model analyzed key features of flooding payload such as data length and transfer ID.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 30 April 2024 doi:10.20944/preprints202404.2018.v1

Table 11 is a graph of SHAP value distribution analysis between feature value, feature importance, and important features through SHAP value analysis for scenario type 01 dataset. In the feature value graph, you can see that data Length is concentrated in two values near SHAP value 0.0. This shows that the data length of the flooding payload is all 7 and 8, and feature values are concentrated in that section. Additionally, the blue dots with relatively low feature values are concentrated in the range of -0.6 to -0.2 based on the SHAP value, and you can see that most of the normal driving data is distributed with a data length other than 7 or 8.

Table 11. SHAP Analysis for Scenario Type 01.

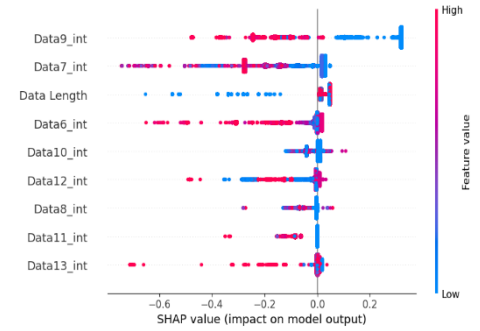
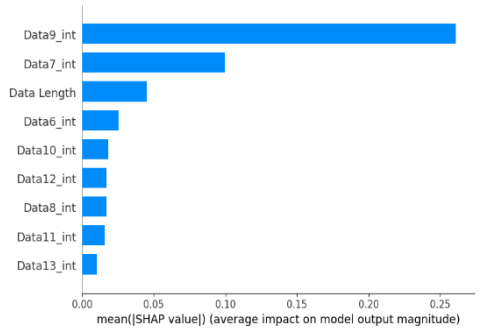
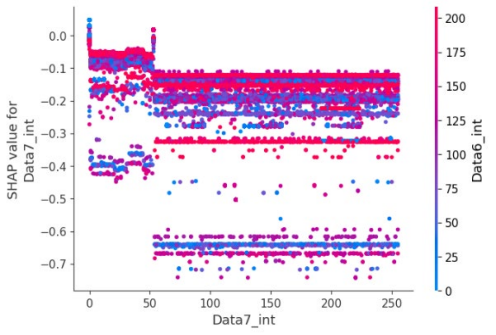
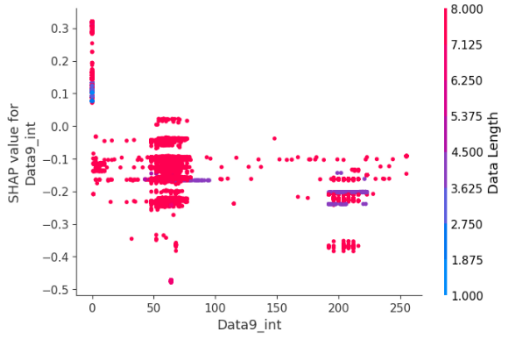
SHAP Value Distribution by Feature through Tree Explainer (Type 01 Dataset)	Feature Importance through Tree Explainer (Type 01 Dataset)
	
Comparison of SHAP Value distribution of 7th Data and 6th Data	**Comparison of SHAP Value distribution of 9th Data and Data Length**
	

Figure 6 is a SHAP force plot for 3 random rows of data. The red section is where the SHAP value is high, and the blue section is where the SHAP value is low. The 1st and 2nd graphs in Figure 6 are clearly different from the 3rd graph in the 9th data byte and data length. This means that the flooding payload has a feature for flooding attacks in the 9th data byte and is judged according to the data length. We can confirm that the evidence is being classified.

Figure 6. SHAP Force plot for 3 random rows of data.

SHAP value analysis was performed with data length, which had a high correlation coefficient with other data, and Data 9, which had the highest feature importance.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 30 April 2024 doi:10.20944/preprints202404.2018.v1

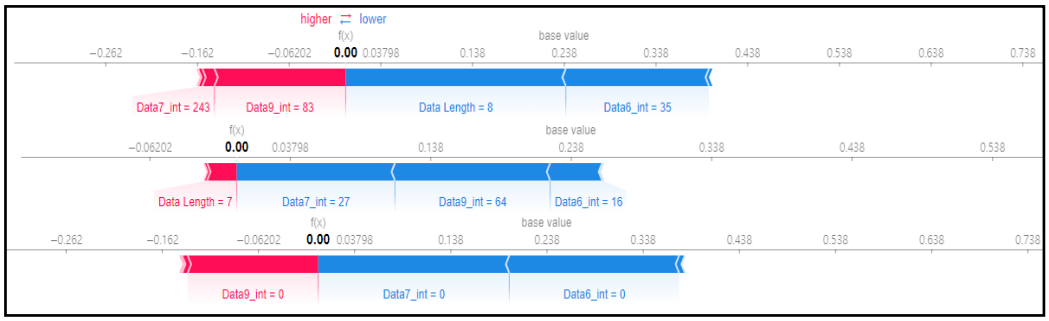
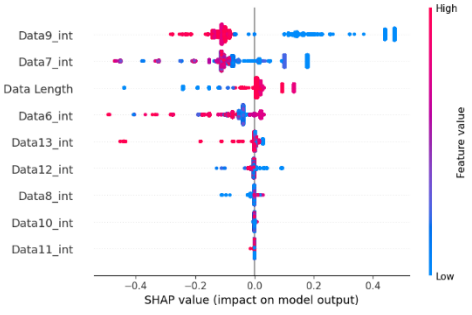
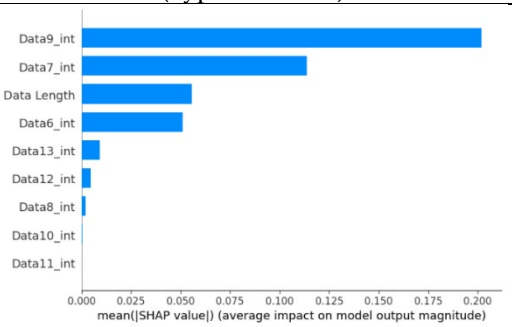
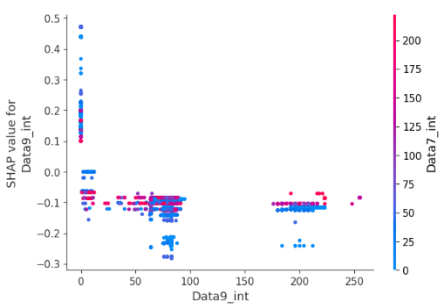
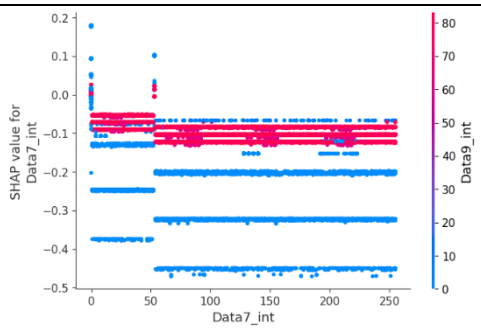


Figure 7. SHAP Force plot for 4 random rows of data (Type 02).

Scenario Datasets 01 and 02 are all data related to flooding attacks, and in the SHAP analysis results, the 6th, 7th, and 9th data bytes were commonly selected as important features. Additionally, when analyzing the SHAP value distribution of the 6th and 7th data bytes of Scenario Type 01, a specific pattern is confirmed.

Table 12 is a graph of the SHAP value distribution analysis between feature value, feature importance, and important features through SHAP value analysis for the scenario Type 02 dataset. It is an analysis of the same flooding attack dataset as the previous Table 11. Therefore, in the Feature Importance graph, the 9th Data byte, 7th data byte, and data length are selected as important features. However, in the SHAP value distribution graph, both blue and red dots show a more biased distribution, and because type 02 has a more unbalanced normal and attack ratio than type 01, SHAP analysis was performed in which the explainer model of type 02 over-fitted compared to type 01. It is estimated that.

Table 12. SHAP Analysis for Scenario Type 02.

SHAP Value Distribution by Feature through Tree Explainer (Type 02 Dataset)	Feature Importance through Tree Explainer (Type 02 Dataset)
	
Comparison of SHAP Value distribution of 9th Data and 7th Data	Comparison of SHAP Value distribution of 7th Data and 9th Data
	

The analysis of feature importance for each scenario type dataset through SHAP value analysis based on the XG boost model is as follows. Scenario Type 1 and Type 2 Datasets, in which flooding attack data, a type of DDoS attack, were collected, commonly measured the 9th, 7th, and 6th data

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 30 April 2024 doi:10.20944/preprints202404.2018.v1

bytes and data length as high Importance. This was consistently observed despite the difference in proportions between the two data sets.

3.1.2. Fuzzy Attack Scenario Dataset Analysis (Type 03 & 04)

We can check the features of the fuzzy attack through the SHAP value distribution graph in Table 13. Except for data length, all data byte items have high feature values in the section with high SHAP value, and low feature values in the section with low SHAP value. This is presumed to be a pattern caused by the attacker injecting the payload using the brute forcing method. Therefore, unlike the previous flooding dataset, the feature importance of data length was measured to be relatively low.

Table 13. SHAP Analysis for Scenario Type 03.

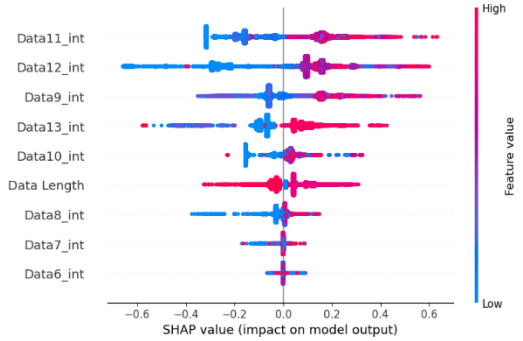
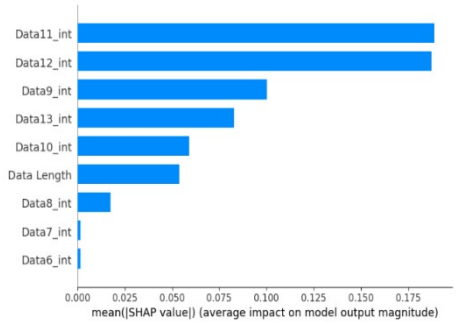
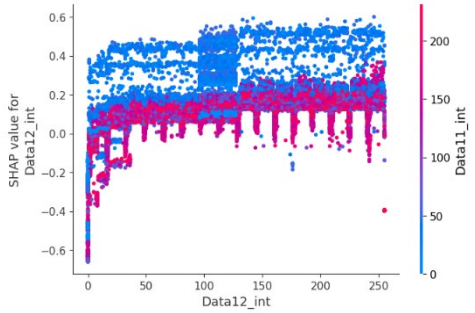
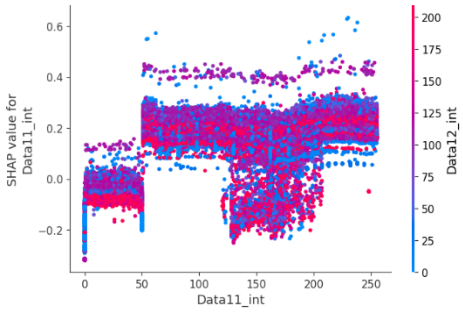
SHAP Value Distribution by Feature through Tree Explainer (Type 03 Dataset)	Feature Importance through Tree Explainer (Type 03 Dataset)
	
Comparison of SHAP Value distribution of 12th Data and 11th Data	**Comparison of SHAP Value distribution of 11th Data and 12th Data**
	

Figure 8 is the SHAP force plot for three rows of the scenario dataset. We can check the pattern that matches the previous SHAP analysis. Unlike the flooding dataset, a different data byte was selected for each row and the SHAP value was analyzed separately, and the characteristic of the fuzzy attack being brute forcing for each data byte is observed.

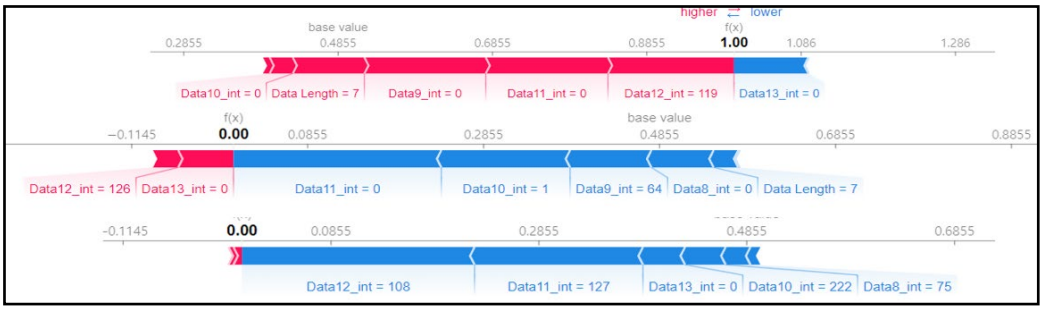


Figure 8. SHAP Force plot for 3 random rows of data (Type 03).

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 30 April 2024 doi:10.20944/preprints202404.2018.v1

Table 14 is a dataset for the same fuzzy attack as Scenario type 03, but it is unbalanced due to the low rate of attack data. However, in the section where the SHAP value was high, the feature value was high, the feature importance of the data length was measured to be low, and the SHAP value distribution between features was observed to be clustered in blue and red.

Table 14. SHAP Analysis for Scenario Type 04.

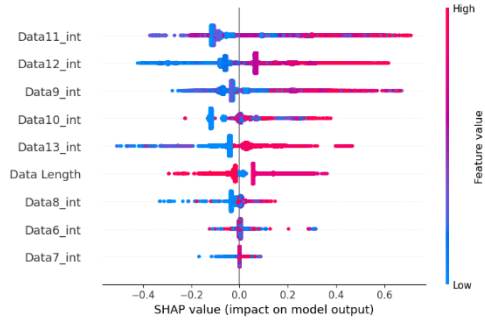
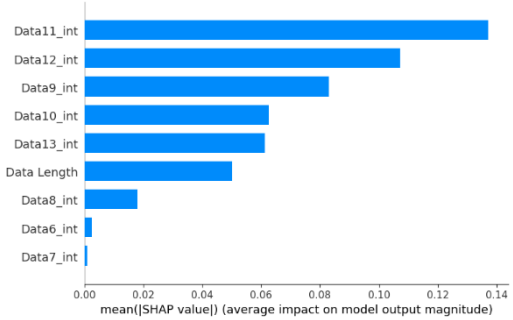
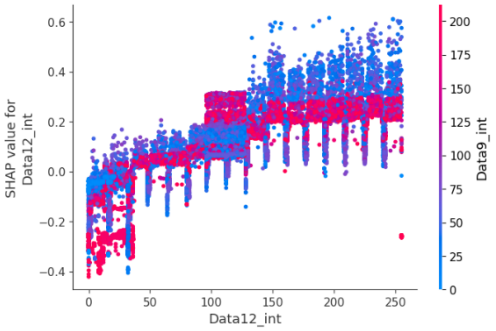
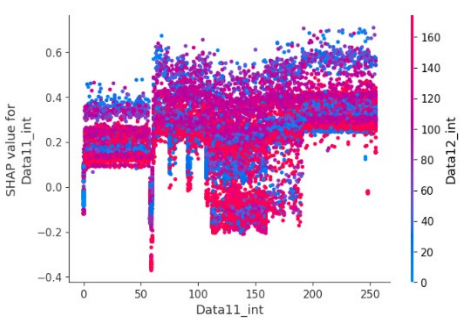
SHAP Value Distribution by Feature through Tree Explainer (Type 04 Dataset)	Feature Importance through Tree Explainer (Type 04 Dataset)
	
Comparison of SHAP Value distribution of 12th Data and 9th Data	**Comparison of SHAP Value distribution of 11th Data and 12th Data**
	

Figure 9. SHAP Force plot for 3 random rows of data (Type 04).

The above pattern can be confirmed in the SHAP force plot for three random rows. In all three rows, the SHAP value was high in the section with a low base value, and the SHAP value was measured to be low in the section with a high base value, and the target Features are also different for each row.

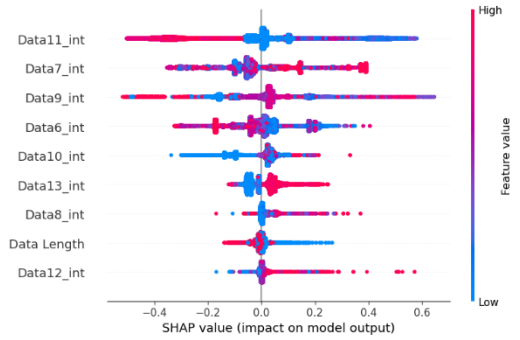
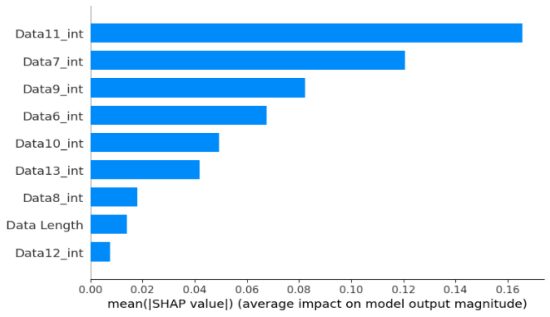
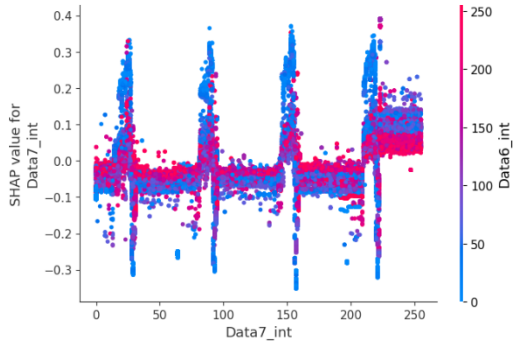
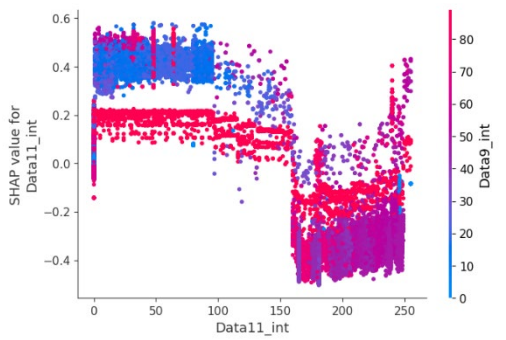
3.1.3. Replay Attack Scenario Dataset Analysis (Type 05, Type 06)

Table 15 is a graph of the SHAP value analysis results for scenario type 05, which collected replay attacks. In the SHAP value distribution plot for the 6th and 7th data bytes and the 11th and 9th SHAP value distribution plot, the distribution of blue and red dots is more clustered than other previous datasets. This is presumed to be a result of the Replay attack's principle of injecting the same payload

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 30 April 2024 doi:10.20944/preprints202404.2018.v1

repeatedly. Therefore, normal driving data using various forms can be classified into blue dots with low SHAP values and clustered in a straight line.

Table 15. SHAP Analysis for Scenario Type 05.

SHAP Value Distribution by Feature through Tree Explainer (Type 05 Dataset)	Feature Importance through Tree Explainer (Type 05 Dataset)
	
Comparison of SHAP Value distribution of 7th Data and 6th Data	**Comparison of SHAP Value distribution of 11th Data and 9th Data**
	

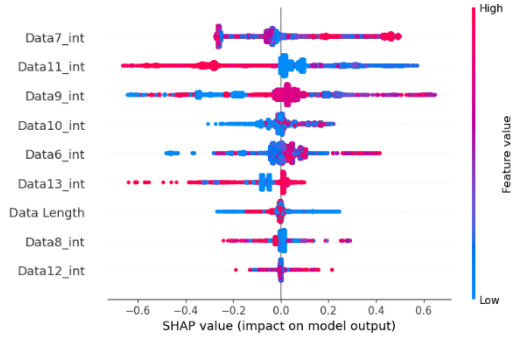
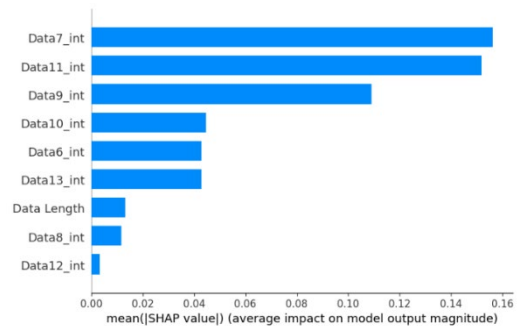
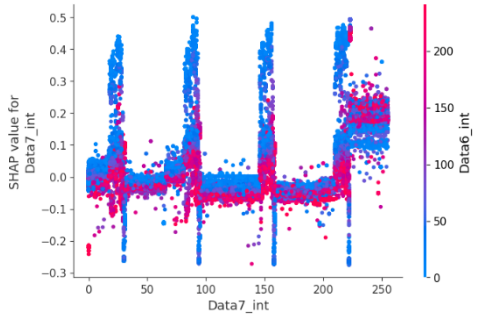
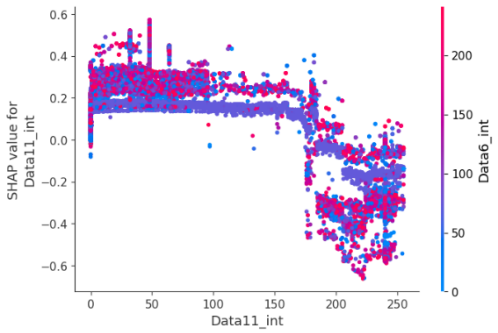
The pattern of the above replay attack can be seen in the SHAP Force plot in Figure 10. It is observed that the SHAP value for some features is not high due to repeated injection of replay's payload, and the SHAP value is evenly distributed for several features. This model is assumed to analyze whether it is a replay payload through the data bytes of the entire message rather than finding patterns for attacks in some features.

Figure 10. SHAP Force plot for 3 random rows of data (Type 05).

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 30 April 2024 doi:10.20944/preprints202404.2018.v1

Table 16 consists of a SHAP value distribution plot, feature importance plot, and distribution graph between SHAP value and feature value for each feature for scenario type 06, which collected replay attack data as in Table 15. We can see the clustering of blue and red dots in the SHAP value distribution plot of the 6th and 7th data bytes.

Table 16. SHAP Analysis for Scenario Type 06.

SHAP Value Distribution by Feature through Tree Explainer (Type 06 Dataset)	Feature Importance through Tree Explainer (Type 06 Dataset)
	
Comparison of SHAP Value distribution of 7th Data and 6th Data	**Comparison of SHAP Value distribution of 11th Data and 6th Data**
	

In Figure 11 unlike the previous Figure 10, features with high SHAP values are observed, which suggests that scenario type 06 collected more diverse normal driving data than scenario type 05. This is because the dataset of Scenario type 06 consists of more rows of data, and the rate of attack data is also higher.

Figure 11. SHAP Force plot for 3 random rows of data (Type 06).

 doi:10.20944/preprints202404.2018.v1

3.2. Single Model Performance Evaluation by Scenario Type

3.2.1. Experiment Environment

Table 18. Hardware and Software Environments Configuration.

Division	Description
Memory	32 GB
CPU	AMD Ryzen 7 5700X
GPU	NVIDIA GeForce RTX 3060
OS	Microsoft Windows 11 Pro 64bit

The criteria used to evaluate the performance of each model are as follows. The elements of each matrix, TP, FP, TN, and FN, follow the definition of confusion matrix.

3.2.2. Performance Evaluation Metrics

In this Experiment, the relationship between the model's detection results and the labels of the actual data was defined according to the confusion matrix in Table 19. TP (True Positive) when an actual attack is normally detected, FN (False Negative) when an actual attack is not detected, FP (False Positive) when a normal attack is detected, and TN (True Negative) when a normal is classified as normal.

Table 19. Confusion Matrix.

		Label	
		Positive	**Negative**
Prediction	Positive	TP (True Positive)	FP (False Positive)
	Negative	FP (False Negative)	TN (True Negative)

TP, FP, TN, and FN classified according to the definition in Table 19 are substituted into the evaluation matrix definitions in Table 20 to verify the model.

Table 20. Evaluation Matrix Definitions.

$Accuarcy = \frac{TP+TN}{TP+TN+FP+FN} * 100$	$F1-score = 2 * \frac{Precision * Recall}{Precision + Recall} * 100$
$Precision = \frac{TP}{TP+FP} * 100$	$Recall = \frac{TP}{TP+FN} * 100$

Accuracy is an indicator of how well a model classifies actual data, and is a commonly used indicator for model evaluation. However, the bias problem caused by model over-fit can be overlooked, so it is used together with the F1-Score indicator, which consists of precision and recall.

3.2.3. Performance Evaluation Results

Accuracy and F1-Score in Table 14 were used as indicators for model performance evaluation, and the performance evaluation results for each attack are shown in Table 15.

Table 21. Evaluation Results for each base models.

< Scenario Type 01 >			**< Scenario Type 02 Accuracy >**		
Model	**Accuracy**	**F1-Score**	**Model**	**Accuracy**	**F1-Score**
Logistic Regression	98.028%	98.028%	Logistic Regression	97.771%	97.771%
K-Neighbors Classifier	98.635%	98.635%			
Decision Tree #1	98.440%	98.440%			

 doi:10.20944/preprints202404.2018.v1

Decision Tree #2	98.632%	98.798%	K-Neighbors Classifier	98.519%	98.519%
Random Forest Classifier	98.637%	98.801%	Decision Tree #1	98.005%	98.005%
DNN #1	98.549%	99.00%	Decision Tree #2	96.977%	98.517%
DNN #2	98.400%	71.959	Random Forest Classifier	96.982%	98.519%
RNN	98.786%	98.788%	DNN #1	98.295%	97.921%
LSTM	98.621%	98.786%	DNN #2	98.253%	98.241%
			RNN	98.519%	96.983%
			LSTM	96.983 %	96.983%

< Scenario Type 03 Accuracy >

Model	Accuracy	F1-Score
Logistic Regression	93.383%	93.383
K-Neighbors Classifier	97.932%	97.932
Decision Tree #1	98.391%	98.367%
Decision Tree #2	98.391%	99.518%
Random Forest Classifier	99.617%	99.650%
DNN #1	95.086%	95.086%
DNN #2	96.644%	96.944%
RNN	99.180%	99.151%
LSTM	98.953%	98.922%

< Scenario Type 04 Accuracy >

Model	Accuracy	F1-Score
Logistic Regression	79.415%	79.415%
K-Neighbors Classifier	94.487%	94.487%
Decision Tree #1	95.514%	95.514%
Decision Tree #2	98.691%	98.691%
Random Forest Classifier	98.867%	98.867%
DNN #1	90.045%	73.345%
DNN #2	94.272%	87.673%
RNN	98.995%	97.678%
LSTM	99.025%	97.743%

< Scenario Type 05 Accuracy >

Model	Accuracy	F1-Score
Logistic Regression	79.769%	79.769%
K-Neighbors Classifier	95.890%	95.889%
Decision Tree #1	92.208%	84.422%
Decision Tree #2	97.414%	97.414%
Random Forest Classifier	97.840%	97.840%
DNN #1	82.697%	82.698%
DNN #2	84.118%	84.119%
RNN	89.458%	89.986%
LSTM	94.487%	89.458%

Scenario Type 06 Accuracy >

Model	Accuracy	F1-Score
Logistic Regression	86.929%	86.929%
K-Neighbors Classifier	98.778%	98.185%
Decision Tree #1	92.68%	92.679%
Decision Tree #2	99.465%	99.465%
Random Forest Classifier	99.601%	99.743%
DNN #1	84.347%	78.306%
DNN #2	90.886%	84.477%
RNN	98.338%	97.539%
LSTM	98.512%	97.800%

Scenario type 1 and scenario type 2 are flooding attack datasets, a type of DoS attack. Despite the conditions in which Timestamp and data order are shuffled, time series analysis models RNN and LSTM models show excellent performance of over 95% in accuracy and F1-Score indicators is showed. In addition, scenario type 3 and scenario type 4, which are fuzzy attack datasets, and scenario type 6, which are relay attack datasets, showed excellent performance of over 95% in both Accuracy and F1-Score indicators. However, in scenario type 5's dataset, the f1-score of both models fell below 90%. On the other hand, the LSTM model that performed time series analysis shows accuracy of over 99% [22]

Decision Tree single model, Random Forest, K-Neighbors Classifier, and dual classification DNN model showed excellent performance with accuracy and F1-Score indicators of over 95% in all datasets, and Logistic Regression model's Accuracy varied between 79% ~ 97%. The F1-Score of ML model is 97%, which is like the existing model. Over-fit due to data imbalance was not observed.

 doi:10.20944/preprints202404.2018.v1

4. Experiments

In Performance Evaluation in Chapter 3.2, models with excellent detection performance for each attack type are selected as base models to design a stacking-based ensemble model.

The base models independently detect flooding, fuzzy, and replay attacks from the input CAN Traffic Data and deliver these detection results to the Meta model. The decision tree-based the Meta model synthesizes the detection results of each base model to determine whether an attack has occurred in a binary sense. Perform classification.

4.1. Experiment Model

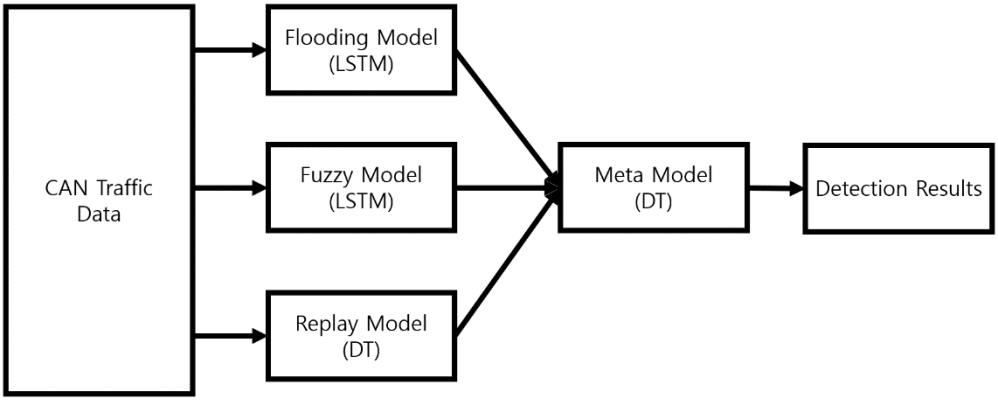
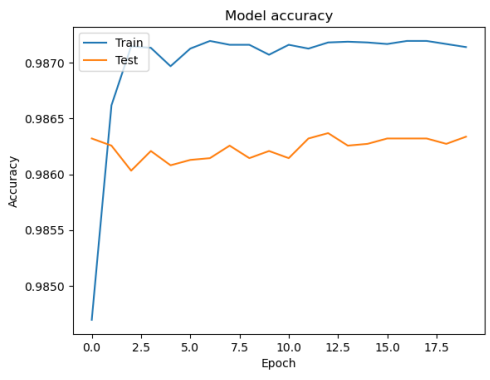
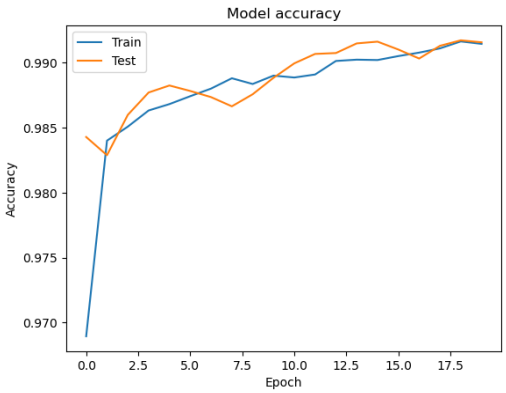


Figure 12. Structure of Experiment model.

The base model was constructed according to the performance evaluation results in chapter 3. Flooding detection used the LSTM model, fuzzy detection used the LSTM model, and replay detection used the Decision Tree (DT) model, all under the same conditions as the model learned during performance evaluation.

The Flood model and Fuzzy model were learned with 20 epochs through the LSTM model, and the accuracy of the two models is over 98% during the train and test process. Accuracy changes according to the learning process are shown in Table 22.

Table 22. Experiment Model's Optimization Process History Plot for Flood, Fuzzy model.

Flood	Fuzzy
	

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 30 April 2024 doi:10.20944/preprints202404.2018.v1

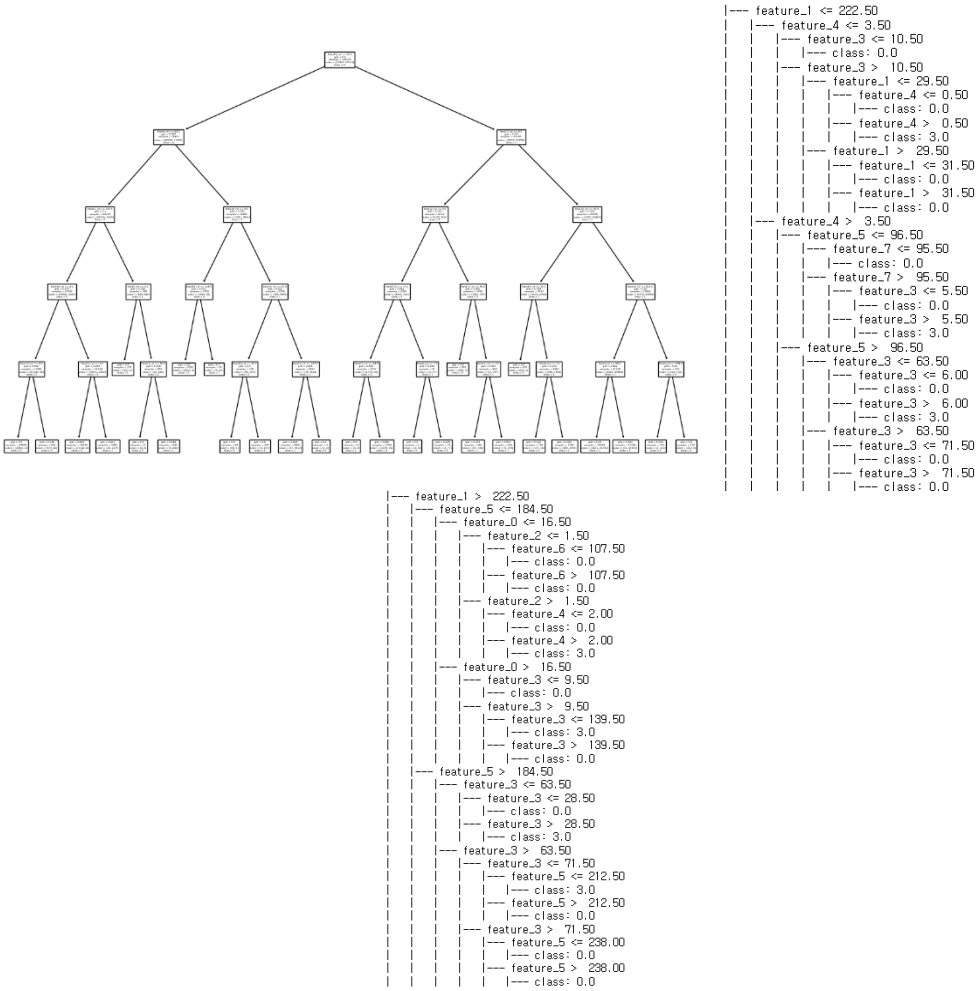


Figure 13. Structure of Relay Attacks model.

Unlike the previous two models, the Replay model was learned through a decision tree model with max depth set to 5, and in the performance evaluation in chapter 3. Replay attacks were detected with an Accuracy and F1-Score of over 97%. Through tree visualization, it was confirmed that the model underwent balanced training.

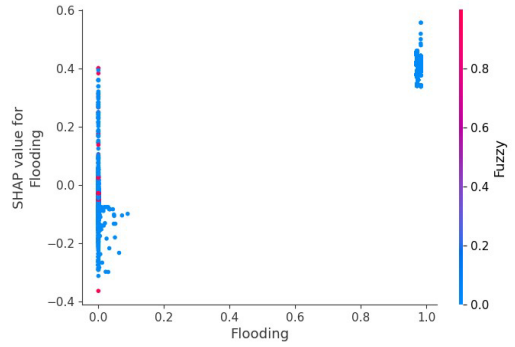
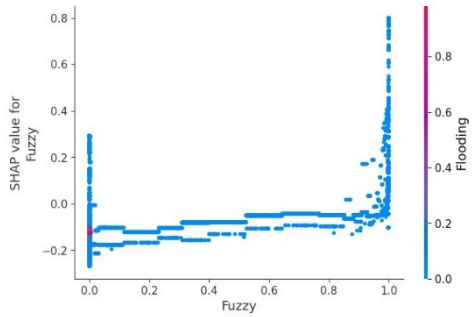
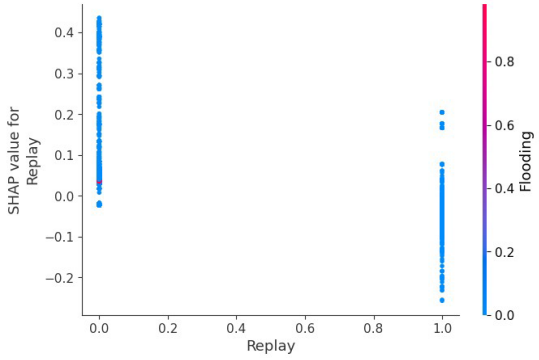
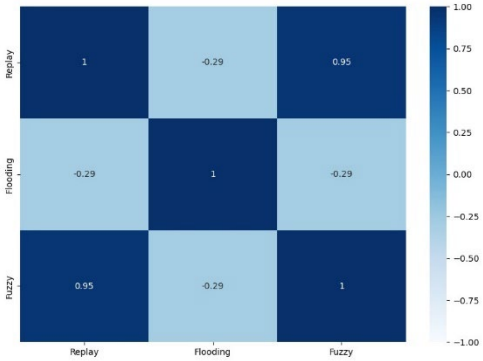
The model that assembled these base models had an Accuracy of 83%, Precision of 62%, and F1-Score of 68%. Therefore, the cause of lower performance than the existing base model was analyzed through SHAP analysis in Chapter 4.2.

4.2. Base Model's Results Analysis

As a result of performing SHAP value interaction analysis and Pearson correlation analysis on the detection results for each base model, it is estimated that the correlation and similarity between the detection results of the Replay model and the Fuzzy model are high.

 doi:10.20944/preprints202404.2018.v1

Table 23. Analysis Results of each model.

Flooding-fuzz	Fuzz-flooding
	
Replay-flooding	**Heatmap of Pearson correlation**
	

4.3. Experiment Analysis

In this experiment, time stamps were removed from DL models that perform time series analysis, such as RNN, LSTM model, and ML models, and experiments were conducted to detect fuzzing, flooding, and relay attacks by learning data with shuffled data sequence, and all models showed significant detection performance.

In addition, the results of analyzing this detection performance through SHAP analysis show that the models learned the data byte associated with the algorithm used in each attack as a meaningful feature.

Although it was effective in binary classification of a single attack through a single model, performance was low in binary classification of multiple attack types through a stacking model. Looking at the SHAP analysis and correlation analysis results, it is assumed that the Replay model and the Fuzzy model have a high correlation in the detection results, causing overfitting in the Ensemble model through stacking, adversely affecting performance.

5. Conclusions

In this study, the features analyzed by ML models in the payload of flooding, fuzzy, and replay attacks occurring in the UAVCAN Protocol were confirmed through SHAP Analysis using the Tree Explainer model. This can be inferred that the ML model is learning by deriving valid features from the payload for each attack type.

For the dataset for each scenario type, SHAP value - feature value distribution comparison, feature importance analysis, SHAP value distribution analysis between important features, and force plot analysis for each row of the dataset were performed. In all scenario datasets, the more imbalanced the data ratio according to label, the more biased the SHAP analysis results were. Although this bias is not effective in the model's performance, it is easier to see how the explainer model performs analysis on the features in the dataset, and it can be assumed that effective analysis is being performed according to the characteristics of each attack type.

 doi:10.20944/preprints202404.2018.v1

We confirmed that the LSTM model that removed the timestamp and learned the shuffled dataset produced an Accuracy of over 96% and an F1-Score of over 96% in all scenarios except type 5. These results show the performance of the LSTM model that learned time series data. Although it is lower, it can be assumed that detection is possible through non-Timelines analysis. As a result, the cause of poor performance in the heterogeneous ensemble model was estimated through SHAP analysis, which can be used in research on applying the heterogeneous ensemble model to detecting UAVCAN network attacks.

In the future, the performance of the detection model can be improved by creating an ensemble model composed of base models learned to enable multi classification.

Funding: This work was supported by the Hongik University new faculty research support fund.

References

1. Altawy, R., & Youssef, A. M. (2016). Security, privacy, and safety aspects of civilian drones: A survey. ACM Transactions on Cyber-Physical Systems, 1(2), 1-25.
2. Liu, J., Yin, T., Yue, D., Karimi, H. R., & Cao, J. (2020). Event-based secure leader-following consensus control for multiagent systems with multiple cyber attacks. IEEE Transactions on Cybernetics, 51(1), 162-173.
3. Cao, J., Ding, D., Liu, J., Tian, E., Hu, S., & Xie, X. (2021). Hybrid-triggered-based security controller design for networked control system under multiple cyber attacks. Information Sciences, 548, 69-84.
4. Robert Bosch GmbH, (1991). CAN Specification version 2.0, Postfach 30 02 40, D-70442 Stuttgart
5. Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. Advances in neural information processing systems, 30.
6. Lundberg, S. M., Erion, G. G., & Lee, S. I. (2018). Consistent individualized feature attribution for tree ensembles. arXiv preprint arXiv:1802.03888.
7. Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B. & Lee, S. I. (2020). From local explanations to global understanding with explainable AI for trees. Nature machine intelligence, 2(1), 56-67.
8. Li, J., Guo, Y., Li, L., Liu, X., & Wang, R. (2023, August). Using LightGBM with SHAP for predicting and analyzing traffic accidents severity. In 2023 7th International Conference on Transportation Information and Safety (ICTIS) (pp. 2150-2155). IEEE.
9. Lee, Y. G., Oh, J. Y., Kim, D., & Kim, G. (2023). Shap value-based feature importance analysis for short-term load forecasting. Journal of Electrical Engineering & Technology, 18(1), 579-588.
10. OpenCyphal. Available online: https://legacy.uavcan.org/ (accessed on 26 april 2024)
11. Tanksale, V. (2019, November). Intrusion detection for controller area network using support vector machines. In 2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems Workshops (MASSW) (pp. 121-126). IEEE.Shrestha, B. (2015).
12. Kou, L., Ding, S., Wu, T., Dong, W., & Yin, Y. (2022). An intrusion detection model for drone communication network in sdn environment. Drones, 6(11), 342.
13. Moulahi, T., Zidi, S., Alabdulatif, A., & Atiquzzaman, M. (2021). Comparative performance evaluation of intrusion detection based on machine learning in in-vehicle controller area network bus. IEEE Access, 9, 99595-99605.
14. Javed, A. R., Ur Rehman, S., Khan, M. U., Alazab, M., & Reddy, T. (2021). CANintelliIDS: Detecting in-vehicle intrusion attacks on a controller area network using CNN and attention-based GRU. IEEE transactions on network science and engineering, 8(2), 1456-1466.
15. Alsoliman, A., Rigoni, G., Callegaro, D., Levorato, M., Pinotti, C. M., & Conti, M. (2023). Intrusion Detection Framework for Invasive FPV Drones Using Video Streaming Characteristics. ACM Transactions on Cyber-Physical Systems, 7(2), 1-29.Kang, M. J., & Kang, J. W. (2016). Intrusion detection system using deep neural network for in-vehicle network security. PloS one, 11(6), e0155781.
16. Song, H. M., Woo, J., & Kim, H. K. (2020). In-vehicle network intrusion detection using deep convolutional neural network. Vehicular Communications, 21, 100198.
17. Tariq, S., Lee, S., Kim, H. K., & Woo, S. S. (2020). CAN-ADF: The controller area network attack detection framework. Computers & Security, 94, 101857.
18. Qin, H., Yan, M., & Ji, H. (2021). Application of controller area network (CAN) bus anomaly detection based on time series prediction. Vehicular Communications, 27, 100291.
19. Seo, E., Song, H. M., & Kim, H. K. (2018, August). GIDS: GAN based intrusion detection system for in-vehicle network. In 2018 16th Annual Conference on Privacy, Security and Trust (PST) (pp. 1-6).
20. Khan, M. H., Javed, A. R., Iqbal, Z., Asim, M., & Awad, A. I. (2024). DivaCAN: Detecting in-vehicle intrusion attacks on a controller area network using ensemble learning. Computers & Security, 103712.

 doi:10.20944/preprints202404.2018.v1

21. Tlili, F., Ayed, S., & CHAARI FOURATI, L. (2023, August). Dynamic Intrusion Detection Framework for UAVCAN Protocol Using AI. In Proceedings of the 18th International Conference on Availability, Reliability and Security (pp. 1-10).
22. Islam, R., Refat, R. U. D., Yerram, S. M., & Malik, H. (2020). Graph-based intrusion detection system for controller area networks. IEEE Transactions on Intelligent Transportation Systems, 23(3), 1727-1736.
23. Kim, Dongsung, et al. "Uavcan dataset description." arXiv preprint arXiv:2212.09268 (2022).
24. Capuano, N., Fenza, G., Loia, V., & Stanzione, C. (2022). Explainable artificial intelligence in cybersecurity: A survey. IEEE Access, 10, 93575-93600.
25. Ribeiro, M. T., Singh, S., & Guestrin, C. (2016, August). " Why should i trust you?" Explaining the predictions of any classifier. In Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (pp. 1135-1144).
26. Chamola, V., Hassija, V., Sulthana, A. R., Ghosh, D., Dhingra, D., & Sikdar, B. (2023). A review of trustworthy and explainable artificial intelligence (xai). IEEE Access.
27. Covington, P., Adams, J., & Sargin, E. (2016, September). Deep neural networks for youtube recommendations. InProceedings of the 10th ACM conference on recommender systems(pp. 191-198).
28. The Asimov Institute. Available online: https://www.asimovinstitute.org/neural-network-zoo/ (Acceses on 8 March 2024)
29. OpenCyphal. DS-015 UAVCAN Drone Standard v1.0.1, (2021)
30. Sikora, Riyaz. "A modified stacking ensemble machine learning algorithm using genetic algorithms." Handbook of research on organizational transformations through big data analytics. IGi Global, 2015. 43-53.Kwon, H., Park, J., & Lee, Y. (2019). Stacking ensemble technique for classifying breast cancer. Healthcare informatics research, 25(4), 283-288.
31. Charoenkwan, P., Chiangjong, W., Nantasenamat, C., Hasan, M. M., Manavalan, B., & Shoombuatong, W. (2021). StackIL6: a stacking ensemble model for improving the prediction of IL-6 inducing peptides. Briefings in bioinformatics, 22(6), bbab172.
32. Akyol, K. (2020). Stacking ensemble based deep neural networks modeling for effective epileptic seizure detection. Expert Systems with Applications, 148, 113239.
33. Rashid, M., Kamruzzaman, J., Imam, T., Wibowo, S., & Gordon, S. (2022). A tree-based stacking ensemble technique with feature selection for network intrusion detection. Applied Intelligence, 52(9), 9768-9781.
34. Shrestha, R., Omidkar, A., Roudi, S. A., Abbas, R., & Kim, S. (2021). Machine-learning-enabled intrusion detection system for cellular connected UAV networks. Electronics, 10(13), 1549.
35. Hartmann, K., & Steup, C. (2013, June). The vulnerability of UAVs to cyber attacks-An approach to the risk assessment. In 2013 5th international conference on cyber conflict (CYCON 2013) (pp. 1-23).
36. Kim, J., Kim, S., Ju, C., & Son, H. I. (2019). Unmanned aerial vehicles in agriculture: A review of perspective of platform, control, and applications. Ieee Access, 7, 105100-105115.
37. Mademlis, I., Nikolaidis, N., Tefas, A., Pitas, I., Wagner, T., & Messina, A. (2018). Autonomous unmanned aerial vehicles filming in dynamic unstructured outdoor environments. IEEE Signal Processing Magazine, 36(1), 147-153.
38. Gargalakos, M. (2021). The role of unmanned aerial vehicles in military communications: application scenarios, current trends, and beyond. The Journal of Defense Modeling and Simulation, 15485129211031668.
39. Elman, J. L. (1990). Finding structure in time.Cognitive science,14(2), 179-211.
40. Hochreiter, Sepp, and Jürgen Schmidhuber. “Long short-term memory.” Neural computation 9.8 (1997): 1735-1780.
41. Hyung-Hoon Kim, Yeon-Seon Jeong, Won-Seok Choi, and Hyo-Jin Cho. 2022. Evaluation framework for automotive intrusion detection systems using AI, The Korea Institute of Information Security and Cryptology, 32, 4, (2022), 7-17.