DETECTION OF NETWORK ANOMALIES IN SMART HOME INTERNET OF THINGS IN MACHINE LEARNING

GUO YACHAO

UNIVERSITI TEKNOLOGI MALAYSIA

UNIVERSITI TEKNOLOGI MALAYSIA
DECLARATION OF Choose an item.

Author's full name	:	GUO YACHAO			
Student's Matric No.	:	MCS241039	Academic Session	:	20242025-02
Date of Birth	:	23 March 1995	UTM Email	:	guoyachao@graduate.utm.my
Choose an item. Title	:	Detection Of Network Anomalies In Smart Home Internet Of Things In Machine Learning			

I declare that this Choose an item. is classified as:

☒ **OPEN ACCESS** — I agree that my report to be published as a hard copy or made available through online open access.

☐ **RESTRICTED** — Contains restricted information as specified by the organization/institution where research was done. *(The library will block access for up to three (3) years)*

☐ **CONFIDENTIAL** — Contains confidential information as specified in the Official Secret Act 1972)

(If none of the options are selected, the first option will be chosen by default)

I acknowledged the intellectual property in the Choose an item. belongs to Universiti Teknologi Malaysia, and I agree to allow this to be placed in the library under the following terms :

1. This is the property of Universiti Teknologi Malaysia
2. The Library of Universiti Teknologi Malaysia has the right to make copies for the purpose of research only.
3. The Library of Universiti Teknologi Malaysia is allowed to make copies of this Choose an item. for academic exchange.

Signature of Student:

Signature :

Full Name: GUO YACHAO
Date : 30 JUNE 2025

Approved by Supervisor(s)

Signature of Supervisor I:	Signature of Supervisor II
Full Name of Supervisor I DR SHAHIZAN BIN OTHMAN	Full Name of Supervisor II MOHD ZULI JAAFAR
Date : 30 JUNE 2025	Date :

NOTES : If the thesis is CONFIDENTIAL or RESTRICTED, please attach with the letter from the organization with period and reasons for confidentiality or restriction

"I hereby declare that I have read this thesis and in my opinion this thesis is sufficient in term of scope and quality for the award of the degree of Master in (Data Science)"

Signature	:	_____
Name of Supervisor I	:	DR SHAHIZAN BIN OTHMAN
Date	:	30 JUNE 2025

Signature	:	_____
Name of Supervisor II	:	
Date	:	

Signature	:	_____
Name of Supervisor III	:	
Date	:	

DETECTION OF NETWORK ANOMALIES IN SMART HOME INTERNET OF THINGS IN MACHINE LEARNING

GUO YACHAO

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Master in (Data Science)

School of Education
Faculty of Social Sciences and Humanities
Universiti Teknologi Malaysia

JUNE 2025

DECLARATION

I declare that this thesis entitled *"title of the thesis"* is the result of my own research except as cited in the references. The thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature	:	.....
Name	:	GUO YACHAO
Date	:	30 JUNE 2025

ACKNOWLEDGEMENT

In preparing this thesis, I was in contact with many people, researchers, academicians, and practitioners. They have contributed towards my understanding and thoughts. In particular, I wish to express my sincere appreciation to my main thesis supervisor, Professor Dr. Mohd Shahizan bin Othman, for encouragement, guidance, critics and friendship. Without their continued support and interest, this thesis would not have been the same as presented here.

I am also indebted to Universiti Teknologi Malaysia (UTM) for funding my Master study. Librarians at UTM also deserve special thanks for their assistance in supplying the relevant literatures.

My fellow postgraduate student should also be recognised for their support. My sincere appreciation also extends to all my colleagues and others who have provided assistance at various occasions. Their views and tips are useful indeed. Unfortunately, it is not possible to list all of them in this limited space. I am grateful to all my family member.

ABSTRACT

In the high-tech era of the 21st century, with the continuous advancement of the Internet and the growing prevalence of the Internet of Things (IoT), our lifestyles are changing at an unprecedented pace. Smart homes, smart scenes, and smart spaces are the most evident manifestations of this transformation. Smart homes, in particular, have brought us significant convenience and comfort. However, as smart homes become more widespread, concerns about their security have also increased. This is because smart home devices involve users 'privacy, safety, and health, making it crucial to ensure the security of these devices and protect against malicious attacks from external network environments. In this context, machine learning plays a vital role in detecting normal and abnormal traffic on IoT devices, effectively identifying malicious attacks and attempts. This study builds on previous research, using the UNSW BoT IoT dataset to test and evaluate the performance of smart home networks through new machine learning classification methods. The aim is to enhance the security of IoT devices and protect users' privacy, safety, and health by identifying malicious attacks in traffic using machine learning technology. The analysis of the experimental results shows that Logistic Regression performs relatively poorly in detecting normal traffic, with 4.55% of normal traffic being incorrectly identified as abnormal. Naive Bayes, however, performs perfectly in both normal and attack traffic recognition, achieving a 100% accuracy rate for normal traffic and no false positives for attack traffic. Therefore, the final conclusion of this experiment is that Naive Bayes is the better model for traffic anomaly detection among the three.

Keywords: Smart homes; abnormal traffic; normal l traffic; Naive Bayes; Logistic Regression; attacks; smart home networks

ABSTRAK

This project utilizes the BoT-IoT dataset from the University of New South Wales to investigate how to detect traffic anomalies in IoT devices. Through a scientific experimental process that includes data collection, exploratory data analysis, preprocessing, and the application of various machine learning algorithms, an effective machine learning method for detecting traffic anomalies in IoT devices has been developed. The results show that Logistic Regression performs relatively poorly in detecting normal traffic, with 4.55% of normal traffic being incorrectly identified as abnormal. In contrast, Naive Bayes excels in both normal and attack traffic recognition, achieving to the 100% accuracy rate with no misjudgments for normal traffic and the100% accuracy rate with no false negatives for attack traffic. Therefore, the final conclusion is that Naive Bayes is the best model among the three for detecting traffic anomalies. Additionally, the study briefly explores potential improvements to enhance the quality and accuracy of the experiment, aiming to achieve more precise analysis. Consequently, this research adopts a structured approach from data processing to model evaluation, aiming to make significant contributions to the detection of traffic anomalies in IoT devices.

Kata Kunci: BoT-IoT dataset ; machine learning; traffic anomalies; detection

TABLE OF CONTENTS

TITLE **PAGE**

CHAPTER 1

INTRODUCTION

1.1 Overview

In this era of advanced technology, the continuously evolving Internet of Things is prominently reflected in our daily lives through smart homes, intelligent scenarios, and smart spaces. IoT was often known by the "Internet of Things," which first used in the context of supply chain management in 1999 by Kevin Ashton (Ashton, 2009).IoT can communicate and interact over the Internet or other communication networks(Gubbi et al., 2013).According to statistics, the number of IOT devices worldwide is expected to nearly double from 15.9 billion in 2023 to 32.1 billion by 2030(Lionel, 2024).Smart homes equipped with a large number of continuously operating IoT devices provide us with great convenience and comfort.However,Rapid expansion of the Internet of Things brings innovation and serious security challenges, including malicious attacks and attempts by external network environments that threaten the user privacy, security, and health (Xu et al., 2014).Therefore, it is essential to ensure that smart home devices are protected from malicious attacks and attempts by external network environments, thus safeguarding user privacy, security, and health. Machine learning, which detects normal and abnormal traffic of IoT devices, plays a crucial role in identifying such malicious attacks and attempts. This study tests and evaluates the performance of smart home networks using a machine learning method for detecting abnormal device traffic based on different classifiers. The aim of this study is to enhance the security of IoT devices through machine learning in identifying malicious traffic, thereby protecting user privacy, security, and health.

1.2 Problem Background

The Internet of Things (IoT) aims to the facilitate effective communication between the real world and digital equivalents (also known as digital transformation or cyber-physical systems)(Klötzer.2017).So the rapid growth of the Internet of Things -IoT has revolutionized how we interact with our homes, leading to the rise of smart homes. These environments are equipped with a wide array of connected devices—such as smart thermostats, cameras, door locks, lighting systems, and appliances(Alam et al., 2012)—which can provide remote monitoring, automatic control of lighting and heating, and intelligent monitoring(Weber, R. H. ,2010).These devices communicate over networks to enhance convenience, energy efficiency, and security(Alam et al., 2012). However, this growing interconnectedness also introduces significant challenges, particularly in ensuring the security , reliability , and privacy of these systems. One critical challenge is the detection of network anomalies, which can indicate malicious activities, device malfunctions, or other irregularities.

1.3 Problem Statement

Smart homes are becoming increasingly popular due to their ability to automate daily tasks and improve quality of life. According to recent studies, the global smart home market is expected with growing significantly in the coming years, with billions of IoT devices being deployed worldwide. These devices rely on network connectivity to function effectively, communicating with each other and with external services (e.g., cloud platforms, mobile apps).

However, this reliance on networking introduces vulnerabilities. Unlike traditional computing devices such as laptops or servers, IoT devices often have limited computational power , minimal security features , and less stringent software update mechanisms . As a result, they are attractive targets for attackers looking to exploit weaknesses in the network.

Smart home IoT networks are increasingly targeted by cyberattacks, device malfunctions, and environmental disruptions, leading to network anomalies (unexpected or malicious deviations from normal behavior). However, existing detection methods struggle to address the heterogeneity , resource constraints , and dynamic communication patterns of IoT devices, leaving smart homes vulnerable to security breaches, privacy leaks, and service disruptions.

1.4 Research Goal

The purpose of this project is to detect abnormal and normal behavior of IoT device traffic through machine learning based on a smart home anomaly detection method,in order to identify malicious activities on the network,cluding external attacks and attempts .

1.4.1 Research Objectives

The research objectives of this research are followings:

1）To collect more accurate, representative, comprehensive and widely used data sets;

2）To preprocess the data set. By cleaning, balancing and feature selection in advance, the data set can remove noise data and missing information；

3） To transform the data set into a more appropriate feature format for the classification model;

4）To choose a variety of machine learning methods to experiment for the data set to get more experimental results；

5）To analyze the experimental results in various characteristics to determine the machine learning model with the best abnormal performance of network traffic.

1.5 Research Scope

1）The data set uses a more refined and widely used UNSW BoT IoT data set;

2）The data is converted into feature vectors by using unique heat coding and label coding techniques;

3）A variety of machine learning methods are adopted, including Ada Boost, decision tree, random forest, autoencoder and artificial neural network;

4） The performance parameters of the experimental results evaluation include confusion matrix, accuracy, precision, recall and F1 score.

1.6 Expected Research Contribution

This research is a significant contribution to identify malicious patterns in traffic by using machine learning methods based on feature selection. In this way, user privacy, security, and protection can be achieved. The solution of this project shows a path to integrate secure design into existing IoT installations.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this advanced technological 21st century, as the Internet continues to evolve and the Internet of Things becomes more widespread, our lifestyles are changing at an unprecedented pace. Smart homes, intelligent scenarios, and smart spaces are the most evident proof. In particular, smart homes have provided us with tremendous convenience and comfort. However, as the use of smart homes becomes increasingly prevalent and popular, concerns over their security have also grown. This is because smart home devices involve users' privacy, safety, and health, making it crucial to ensure their security and protect against malicious attacks from external network environments.Among these, machine learning for detecting normal and abnormal traffic on IoT devices plays a crucial role in identifying such malicious attacks and attempts. This study is based on previous research, which are using the the University of New South Wales (UNSW) BoT-IoT data set to test and evaluate the performance of

smart home networks through new classification methods in machine learning for detecting abnormal device traffic. The aim of this study is to enhance the security of IoT devices and protect user privacy, safety, and health by using machine learning to identify malicious attacks in traffic.

2.2 IoT devices

With the continuous progress of the Internet, the core components of the smart home environment, the Internet of Things (IOT) devices can be more automated and intelligent control. Under the synergistic effect of sensors, communication modules and intelligent control systems, the IOT devices provide users with a more convenient, safe and energy-saving life.(Atzori et al., 2010; Gupta et al., 2020)

2.2.1 IoT devices

Security devices in IoT equipment include smart locks, smart cameras, smoke detectors/gas sensors, and infrared human presence sensors. The primary risks associated with smart locks are default passwords, unencrypted communications, and remote access vulnerabilities (Li, X., & Wang, Y., 2021) ; smart cameras face major issues such as video data leakage and man-in-the-middle attacks (Smith, J., & Zhang, L 2022),

2.2.2 Voice and communication equipment

Voice and interaction devices in IoT devices include smart speakers, home gateways/control panels, voice recognition switches, etc.; among them, smart speakers are at risk of being used for voice spoofing attacks (Kumar, R., Jain, S., & Liu, P, 2022).

2.2.3 Lighting and home appliances

Voice and interactive devices in IoT devices include smart light bulbs/switches, smart sockets, smart refrigerators/ovens, etc., among which smart sockets are faced with the risk of automatically turning on high-power appliances regularly after injecting malicious code (Meidan et al., 2018) .

2.2.4 Environmental control equipment

Environmental control devices in IoT devices include intelligent thermostat (such as thermostat), intelligent air conditioning controller, air quality monitor, etc., among which intelligent thermostat is faced with risks such as tampering with instructions or stealing user privacy data, (Roman et al.2013).

2.3 Abnormal traffic detection

When smart home devices are under cyber attacks, anomaly detection of network traffic plays a crucial role. In the field of cybersecurity, anomaly detection of network traffic is an essential method for identifying potential attack behaviors and ensuring system security. This method involves analyzing communication behavior patterns, such as packet frequency, protocol type, and flow duration, where machine learning models can distinguish between normal and abnormal traffic. This approach is particularly important in smart home environments because these devices have limited computing resources and struggle to run traditional firewall mechanisms. (Koroniotis et al., 2019) .

2.4 The current state of research

(1) Machine Learning Methods: Rotation Forest

Through machine learning methods: Rotation Forest, using PCA to rotate the feature space and then construct decision trees, the final results show that Rotation Forest outperforms RF in certain attack types (such as scanning attacks).

(2) Machine Learning Methods: Isolation Forest Unsupervised Detection

Using normal traffic to train the Isolation Forest model, as shown by experimental results; this machine learning method can effectively identify DDoS attack anomalies on Bot-IoT and UNSW BoT-IoT with shorter abnormal path lengths, though the false positive rate is slightly higher but acceptable. (Liu. et al., 2008) Isolation Forest. Proceedings of the Eighth IEEE International Conference on Data Mining

(3) Machine Learning Methods: Autoencoder Reconstruction Error Detection

Through machine learning methods, Autoencoder reconstruction error detection can be achieved by using shallow Autoencoder to learn normal traffic patterns and identify deviant behaviors. Experiments have shown that this machine learning method can effectively identify Mirai attacks, using reconstruction error (MSE) as a criterion for judgment, applicable to edge deployment (Meidan. et al., 2018). N-BaIoT—Network-based detection of IoT botnet attacks using deep autoencoders.

(4) Machine learning methods: Random Forest

Uses machine learning methods: Random Forest, to model Bot-IoT and UNSW BoT-IoT traffic experiments. The experimental results show that the machine learning Random Forest performs well in identifying DDoS attacks (F1-score> 0.9), where Dst Port, Protocol, Total Fwd Packets is proven to be the most critical feature.Towards the development of realistic botnet data set in the internet of things for network forensic analytics: Bot-IoT data set (Koroniotis. et al.2019).

(5) Machine Learning Methods: LSTM Time Series Modeling

Through machine learning methods, specifically LSTM time series modeling, this experiment models traffic as a time series to identify periodic request patterns of DDoS attacks, concluding that LSTM can capture the time dependency of attacks (such as sudden spikes in requests per second) (Perera. et al 2020). AIoT: Artificial Intelligence Meets IoT.

(6) Machine Learning Methods: XGBoost /LightGBM Classification Experiment

Using machine learning methods, specifically the Gradient Boosting Tree model, for anomaly traffic detection, experiments with this method show that XGBoost outperforms traditional Random Forest in many classification attack recognition tasks. (Zhang. et al., 2021) Unsupervised Network Anomaly Detection with Hybrid Models: A Comparative Study

(7) Machine learning approach: Transformer

Through the machine learning approach: Transformer, using the Transformer encoder to analyze long-term dependencies in traffic, this experiment ultimately concluded that the machine learning approach is suitable for complex DDoS pattern recognition, but with high memory requirements.

(Chefer. et al 2021). Transformer-based Anomaly Detection with Explainability.

2.5 Classification

Classification is a data mining technique that can be used to predict the group affiliation of data instances. There are many classification techniques for different classification purposes. (Aized and Arshad, 2017)

Machine learning is a branch of artificial intelligence that uses advanced algorithms to detect patterns in large data sets, enabling machines to learn and adapt without explicit programming. Machine learning is divided into supervised learning and unsupervised learning.

Supervised learning is a machine learning method that uses labeled datasets (labeled data) to train models. The model learns the mapping relationship between input features and output labels, enabling it to predict unknown data. Those techniques are further divided into two major categories:regression and classification. In the regression, the target variable is continuous; whereas in classification, the target variable has discrete category labels.

Unsupervised learning is a machine learning method that does not rely on labels, but models the underlying structure of the data to discover hidden patterns or anomalies. Essentially, in unsupervised learning, the model has no predefined output to predict.

2.5.1 Method of Classification

(1) Logistic regression:

Logistic regression is a classic statistical classification model. Despite its name, it is primarily used for binary classification problems and can be extended to multi-classification scenarios. Its core concept involves using a Sigmoid function to map the output of linear regression into the [0,1] range, indicating the probability that a sample belongs to a specific category.

mathematical expression ：

$$P(y=1|x) = \frac{1}{1+e^{-(w^T x+b)}}$$

Bot-IoT is a large-scale network intrusion detection dataset that includes extensive traffic records. The Logistic Regression model is known for its fast training speed and low resource consumption, making it ideal for initial modeling and rapid validation.

Many attack behaviors in Bot-IoT exhibit a certain degree of linear separability in network characteristics, which Logistic Regression can effectively capture.

This model can serve as a benchmark against more complex models like SVM and random forests, aiding in the evaluation of whether subsequent models can improve upon it

(2) SVC：

Support Vector Classification(SVC) is the application of support vector machine (SVM) in classification tasks. Its core idea is:

Find an optimal hyperplane in the feature space so that samples of different categories are separated as far apart as possible and the boundary (interval) between categories is maximized.

The Bot-IoT data set contains a large number of network traffic features (such as source IP number, destination port, protocol type, etc.), and SVC can handle these high-dimensional data well

Even if the number of training samples is small or the distribution is complex, SVC can obtain better classification results by selecting appropriate kernel functions (such as RBF).

The attack samples in the Bot-IoT data set are usually much less than the normal traffic, and SVC can mitigate the impact of imbalance by adjusting the category weight.

(3) naive Bayes：

Naive Bayes (NB) is a probabilistic classification model based on Bayes' theorem and assumes that features are independent of each other. Although the assumption of "feature independence" may not be true in reality, it still performs well in many practical tasks, especially for high-dimensional data and text classification.

Bayes' theorem formula:

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

The Naive Bayes model is simple in structure and efficient in computation, making it ideal for data sets such as Bot-IoT with large amounts of network traffic records.

IoT traffic data may be incomplete or abnormal, and Naive Bayes has a certain tolerance for such problems.

Bot-IoT contains many discrete fields (such as protocol type, service type, connection status, etc.), and MultinomialNB or BernoulliNB can be used to achieve better results.

Classification method	Strengths	Limitations	Reference
Logistic Regression	1. Computationally efficient and suitable for large datasets. 2. Provides interpretable results showing the impact of each feature on classification. 3. Outputs class probabilities allowing for threshold tuning and risk assessment. 4. Performs well on linearly separable data and can be extended to multi-class classification.	1. Assumes a linear relationship between features and the target, which may not capture complex patterns. 2. Sensitive to irrelevant features and multicollinearity. 3. May underperform on highly imbalanced datasets without proper weighting or resampling.	(Hosmer & Lemeshow, 2000). Applied Logistic Regression. Wiley.

Support Vector Classification (SVC)	1. Effective in high-dimensional spaces and suitable for complex decision boundaries using kernel tricks (e.g., RBF kernel). 2. Robust against overfitting with appropriate regularization and parameter tuning. 3. Works well with small to medium-sized datasets. 4. Supports multi-class classification through one-vs-one or one-vs-rest strategies.	1. Computationally expensive, especially with large datasets. 2. Sensitive to the choice of kernel and hyperparameters. 3. Interpretability is limited compared to simpler models like logistic regression.	(Cortes & Vapnik, 1995). Support-vector networks. Machine Learning"
Naive Bayes	1. Fast training and prediction speed, suitable for real-time applications. 2. Simple and effective for high-dimensional and discrete features. 3. Performs well on categorical data and text-like features. 4. Lightweight model ideal for deployment in resource-constrained environments such as IoT devices."	1. Assumes feature independence, which may not hold in real-world datasets. 2. May perform poorly if this assumption is strongly violated. 3. Not suitable for capturing complex relationships between features.	(Duda et al., 2001). Pattern Classification. Wiley-Interscience.

2.3.1 2.6 Research Gap

2.3.2 Problems / Issues	2.3.3 References	2.3.4 Algorithms/Policies Frameworks	2.3.5 Performance Parameters	2.3.6 Experimental Tools	2.3.7 Results	2.3.8 Advantages	2.3.9 Research Gap
DDoS, attack detection	Koroniotis et al.2019	Random Forest、XGBoost、LSTM、Autoencoder	1.Accuracy: overall accuracy of the model ; 2.F1-score: balance between Precision and Recall, suitable for unbalanced data; 3. AUC-ROC: measure the overall performance of the classifier	Bot-IoT data set	1.Random Forest and XGBoost perform well in DDoS detection (F1-score> 0.9); 2. LSTM can capture the time dependence of attacks, but it has high memory requirements; 3. Autoencoder can effectively identify Mirai attacks and is suitable for unsupervised scenarios	1. High precision classification; 2. Suitable for edge deployment (such as LightGBM, RF)	1. A large amount of annotated data is required; 2. LSTM requires high memory

Scan attack recognition	Meidan et al.2018	Isolation Forest、Autoencoder	1.Recall: the ability to detect real attacks ; 2.Precision: the proportion of predicted attacks in the actual attacks; 3. Reconstruction Error: a key indicator of Autoencoder	Bot-IoT data set	1.Recall: the ability to detect real attacks; 2.Precision: the proportion of predicted attacks in the actual attacks; 3. Reconstruction Error: a key indicator of Autoencoder	1. No need for labels; 2. The length of the abnormal path is short	1. The false alarm rate is slightly higher; 2. The interpretation is limited
Malicious connection detection	Zhang et al.2021	LightGBM、Random Forest	1.Accuracy: the proportion of all correct judgments; 2.F1-score: balance Precision and Recall; 3.Recall: the ability to detect real attacks	UNSWBoT-IoT data set	1.LightGBM is superior to traditional RF in multi-class attack recognition; 2. The feature importance analysis of RF shows that features such as Dst Port and Protocol are key	1.Fast reasoning speed; 2. Low memory occupancy	1. Low sensitivity to complex attack patterns; 2. Edge device resource limitations may affect performance

Cross-device horizontal penetration identification	Perera et al.2020	Transformer、Graph Neural Networks (GNN)	1.Attention Mechanism: Self-attention mechanism; 2. Edge Score: Abnormal communication score between devices; 3. Node Score: Abnormal score of a single node	Bot-IoT data set	1. Attention Mechanism: Self-attention mechanism; 2. Edge Score: Abnormal communication score between devices; 3. Node Score: Abnormal score of a single node	1.Suitable for complex attack pattern recognition; 2. Provides interpretability (Attention Map)	1.High memory demand; 2. High training complexity
The lightweight model is deployed to the home gateway	Rodrí guez et al.2006	Rotation Forest、LightGBM、Extra Trees	1.Model Size: model size; 2,.Inference Speed: reasoning speed	Raspberry Pi、ESP32	1.Rotation Forest performs better than RF in some attack types; 2. LightGBM is more suitable for edge deployment	1. Fast reasoning speed; 2. Low memory occupancy	1. Limited generalization ability; 2. Limited computing resources of edge devices

2.3.10 2.7 Summary

This chapter briefly introduces specific categories of IoT smart home devices and how to detect attacks through network traffic. It also includes a review of the latest research literature on classification and experimental analysis using various machine learning methods. The chapter further analyzes the advantages and disadvantages of different approaches. The next chapter will discuss the research methods and summarize the main strategies used in this paper.

CHAPTER 3

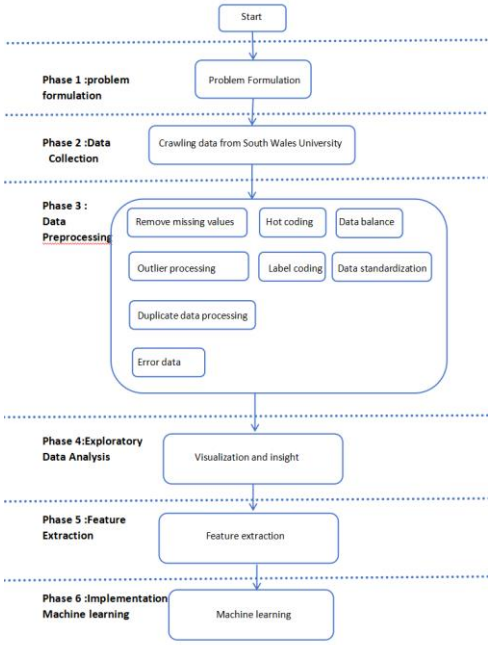
RESEARCH METHODOLOGY

3.1 Introduction

The chapter 3 explains the research methods used to analyze traffic anomaly detection in IoT devices. The methods include data collection, data preprocessing, and machine modeling; various machine learning techniques are employed for classification and detection of device traffic anomalies. The study aims to enhance the security of IoT devices and protect user privacy, safety, and health by using machine learning to identify malicious attacks in traffic.

3.2 Research Framework

1. Tools Problem Definition and Literature Review.
2. Data collection: provided by South Wales University.
3. Data preprocessing: Clean and prepare data for further analysis
4. Feature extraction: word stem extraction and vectorization techniques are applied.
5. Model construction: Build and train the model.
6. Model evaluation: The performance of the model is analyzed through accuracy (accuracy), recall (recall) and precision (precision)



3.3 Problem Formulation：

The main purpose of this study is to use machine learning to identify malicious attacks in the traffic of IoT devices, so as to improve the security of IoT devices and protect users' privacy, safety and health. However, in order to ensure the accuracy and reliability of the analysis, the following problems need to be solved:

1、Pre-data input effects:

(1) Issues of data quality (missing values, redundancy, non-numerical, and irrelevant features)

(2) Class imbalance and noisy data.

2、Impact of data input:

(1) The data type of the input data machine learning does not match the data type that machine learning accepts.

(2) The imbalance of data volume in each category leads to low efficiency.

3、After data input:

(1) The false positive and false negative rates lead to errors in the experimental results

3.4 Data Collection

3.4.1data sources：

This dataset was created by researchers who set up a real IoT experimental environment, including several common smart home devices: Smart Doorbell, Thermostat, Camera, and others. These devices are connected to the home gateway via Wi-Fi or wired connections and are controlled to interact with normal and malicious traffic. The Mirai botnet tool is used to launch DDoS, scanning, and MITM attacks on these devices, simulating attack traffic and capturing network packets during communication. The UNSW BoT-IoT dataset was obtained through this experimental process.

3.4.2 Dataset comparison:

According to the comparison in the table, the UNSW BoT-IoT dataset was published in 2019, making it relatively new. The institution releasing this dataset, the University of New South Wales (UNSW), is highly reputable. This dataset provides a comprehensive CSV label file with 45 fields and has the largest scale, containing over 7.2 million traffic records. Notably, this dataset focuses on botnets in smart home scenarios and is one of the most frequently cited IoT security datasets compared to other datasets.Therefore, this data set is selected as the experimental object.

Dataset	Date of publication	Research and development units	Feature Class	Instances	For IoT scenarios	Up-to-date Relevance	Large-scale Attack Representation	Evaluated by Recent Studies
(UNSW)BoT-IoT	2019	University of New South Wales	Provides a complete CSV label file with 45 fields	The largest scale, with more than 72 million traffic records	Focus on botnets in the smart home scenario	It is relatively new and meets the needs of modern IoT security research	Provides complete botnet lifecycle modeling (scan → control → attack)	One of the most cited IoT security data sets today
N_BaIoT	2018	Ben-Gurion University of the Negev	Only the original pcap file is provided	The scale is moderate, mainly in the form of pcap	Based on real IoT device traffic	While newer, it focuses on the Mirai type botnet and may not reflect the latest attack trends	Focus only on the Mirai type botnet	It is widely used in IoT botnet research
TON_IoT	2020	CSIRO's Data61	Provides multiple CSV files covering a variety of attack types	2.28 million records, enough for ML/DL training	Covering IoT and IIoT (industrial Internet of Things)	It has high timeliness and supports a variety of modern attack types	Provides 23 kinds of real IoT malware behavior data, suitable for practical testing	It has been used in many papers in recent years
IoT-23	2020	ThreatMon (Cybersecurity Company)	Each sample corresponds to one type of malware (a total of 23)	Millions of samples, suitable for small and medium scale experiments	Emphasis on IoT malicious behavior in live deployment	Contains the latest malware variants and is time-sensitive	Including DDoS, Ransomware, Backdoor, Crypto Mining and many other attacks	It is adopted by security vendors and practical projects

3.5 Data Pre-Processing：

The preprocessing of the dataset is to make the input machine learning data set not contain data that affects the experimental results, and the data set can be smoothly input into the machine learning model, and the machine learning model can run efficiently, so the data set needs to be pre-processed.

3.5.1 Preliminary Analysis

Before preprocessing, a preliminary analysis of the dataset is necessary. This initial step is crucial for data analysis as it helps to familiarize oneself with the dataset, including its structure, format, and the types of variables it contains. The preliminary investigation can identify issues that need correction to ensure the reliability of the analysis, such as missing values, outliers, or inconsistencies.

3.5.2 Data Pre-Processing

Data preprocessing includes data cleaning, data type conversion, data balancing and data standardization.These steps are designed to make machine learning models better and more efficient at processing data, resulting in more accurate experimental data.

1. Data cleaning:

(1) Remove missing values; For missing data, replace with NaN or empty string:

```python
# (1) Handle Missing Values: Replace NaN and empty strings
def handle_missing_values(df):
    print("Handling missing values...")
    # Replace empty strings, '?', and other placeholders with NaN
    df.replace(['', ' ', '?', -1, '-1', np.inf, -np.inf], np.nan, inplace=True)

    # Fill numeric columns with mean
    num_cols = df.select_dtypes(include=[np.number]).columns
    df[num_cols] = df[num_cols].fillna(df[num_cols].mean())

    # Drop non-numeric columns with missing values (or encode later if needed)
    non_num_cols = df.select_dtypes(exclude=[np.number]).columns
    df.drop(columns=non_num_cols, inplace=True)

    print("Missing values handled.")
    return df
```

(2) Outlier processing: the values obviously deviating from the normal range in the data are deleted;

```python
# (2) Remove Outliers using Z-score
def remove_outliers(df, z_threshold=3):
    print("Removing outliers...")
    numeric_df = df.select_dtypes(include=[np.number])
    z_scores = np.abs(stats.zscore(numeric_df))
    filtered_entries = (z_scores < z_threshold).all(axis=1)
    df_cleaned = df[filtered_entries]
    print(f"Outliers removed. New shape: {df_cleaned.shape}")
    return df_cleaned
```

(3) Duplicate data processing: delete the same data that has been processed multiple times;

```python
# (3) Remove Duplicate Rows
def remove_duplicates(df):
    print("Removing duplicate rows...")
    initial_shape = df.shape
    df.drop_duplicates(inplace=True)
    duplicates_removed = initial_shape[0] - df.shape[0]
    print(f"{duplicates_removed} duplicate(s) removed.")
    return df
```

(4) Error data: the data contains data that is not required for the search, and the data is deleted;

```python
# (4) Remove Invalid Data: Example filter for invalid protocol types
def remove_invalid_data(df, valid_protocols=None):
    print("Removing invalid data...")
    if valid_protocols is not None and 'proto' in df.columns:
        # Example: Only keep rows where 'proto' is one of the valid protocols
        df = df[df['proto'].isin(valid_protocols)]
        print(f"Invalid protocol entries removed. New shape: {df.shape}")
    return df
```

2. Data type conversion:

(1) The classification features are coded with unique heat, and each attack category has a unique column, thus increasing the number of features;

```python
# (1) One-Hot Encoding for categorical features
def one_hot_encode_features(df, categorical_cols=None):
    if categorical_cols is None or len(categorical_cols) == 0:
        print("No categorical columns provided for one-hot encoding.")
        return df

    print(f"Performing one-hot encoding on columns: {categorical_cols}")
    df_encoded = pd.get_dummies(df, columns=categorical_cols)
    print(f"One-hot encoding completed. New shape: {df_encoded.shape}")
    return df_encoded
```

(2) The data is converted into feature vectors using label coding techniques, which can be converted into a format that machine learning models can receive;

```python
# (2) Label Encoding for target labels (attack types)
def label_encode_target(df, target_col='attack'):
    if target_col not in df.columns:
        raise ValueError(f"Target column '{target_col}' not found in the dataset.")

    print(f"Performing label encoding on column: '{target_col}'")
    le = LabelEncoder()
    df[target_col] = le.fit_transform(df[target_col])

    # Print mapping of original labels to encoded values
    label_mapping = dict(zip(le.classes_, le.transform(le.classes_)))
    print("Encoded label mapping:")
    for cls, idx in label_mapping.items():
        print(f"  {cls}: {idx}")

    print("Label encoding completed.")
    return df, label_mapping
```

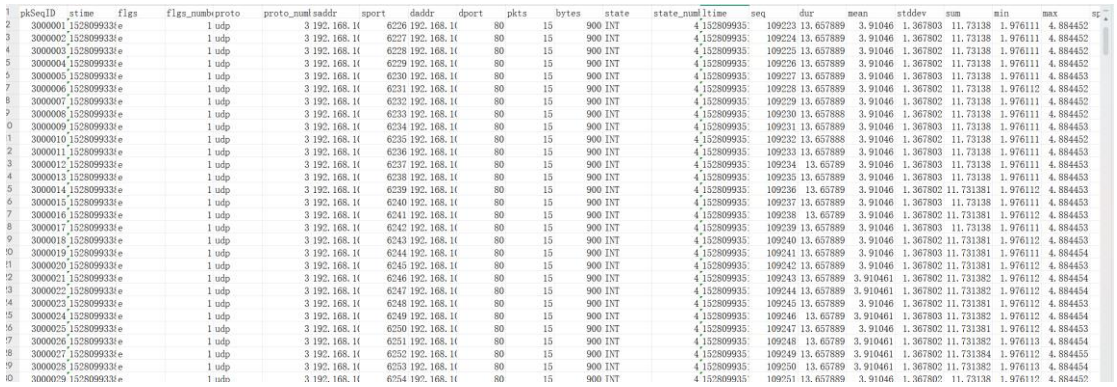
3. Data balance:Using smote over-sampling technology to balance the number of data sets in each category, so as to improve the efficiency of machine learning models processing data;

```python
# Apply SMOTE over-sampling
def apply_smote(X, y):
    print("Applying SMOTE to balance class distribution...")

    # Print original class distribution
    print("Original class distribution:", dict(Counter(y)))

    # Initialize SMOTE
    smote = SMOTE(random_state=42)

    # Apply SMOTE
    X_resampled, y_resampled = smote.fit_resample(X, y)

    # Print new class distribution
    print("Resampled class distribution:", dict(Counter(y_resampled)))
    print("✅ SMOTE applied successfully.")
    return X_resampled, y_resampled
```

4. Data standardization/normalization:Standardized data can help the model better capture the relationship between features and avoid prediction instability caused by numerical problems.

```python
# Load encoded/balanced dataset
def load_data(file_path):
    print(f"Loading data: {file_path}")
    df = pd.read_csv(file_path)
    print(f"Loaded shape: {df.shape}")
    return df

# Standardize numerical features
def standardize_data(df, feature_cols, target_col='attack'):
    print("Starting data standardization...")

    # Separate features and labels
    X = df[feature_cols]
    y = df[[target_col]]

    # Initialize StandardScaler
    scaler = StandardScaler()

    # Fit and transform the features
    X_scaled = scaler.fit_transform(X)

    # Convert back to DataFrame
    X_scaled_df = pd.DataFrame(X_scaled, columns=feature_cols, index=df.index)

    # Combine with labels
    df_scaled = pd.concat([X_scaled_df, y.reset_index(drop=True)], axis=1)

    print("✅ Standardization complete.")
    return df_scaled
```

The processed data set:



	pkSeqID	stime	flgs	flgs_numb	proto	proto_num	saddr	sport	daddr	dport	pkts	bytes	state	state_num	ltime	seq	dur	mean	stddev	sum	min	max	sp
2	3000001	1528093933	e	1	udp	3	192.168.1(	6226	192.168.1(	80	15	900	INT	4	1528099351	109223	13.657889	3.91046	1.367803	11.73138	1.976111	4.884452	
3	3000002	1528093933	e	1	udp	3	192.168.1(	6227	192.168.1(	80	15	900	INT	4	1528099351	109224	13.657889	3.91046	1.367802	11.73138	1.976111	4.884452	
4	3000003	1528093933	e	1	udp	3	192.168.1(	6228	192.168.1(	80	15	900	INT	4	1528099351	109225	13.657889	3.91046	1.367802	11.73138	1.976111	4.884452	
5	3000004	1528093933	e	1	udp	3	192.168.1(	6229	192.168.1(	80	15	900	INT	4	1528099351	109226	13.657889	3.91046	1.367802	11.73138	1.976111	4.884452	
6	3000005	1528093933	e	1	udp	3	192.168.1(	6230	192.168.1(	80	15	900	INT	4	1528099351	109227	13.657889	3.91046	1.367803	11.73138	1.976111	4.884453	
7	3000006	1528093933	e	1	udp	3	192.168.1(	6231	192.168.1(	80	15	900	INT	4	1528099351	109228	13.657889	3.91046	1.367802	11.73138	1.976112	4.884452	
8	3000007	1528093933	e	1	udp	3	192.168.1(	6232	192.168.1(	80	15	900	INT	4	1528099351	109229	13.657889	3.91046	1.367802	11.73138	1.976111	4.884452	
9	3000008	1528093933	e	1	udp	3	192.168.1(	6233	192.168.1(	80	15	900	INT	4	1528099351	109230	13.657888	3.91046	1.367802	11.73138	1.976111	4.884452	
0	3000009	1528093933	e	1	udp	3	192.168.1(	6234	192.168.1(	80	15	900	INT	4	1528099351	109231	13.657889	3.91046	1.367803	11.73138	1.976111	4.884453	
1	3000010	1528093933	e	1	udp	3	192.168.1(	6235	192.168.1(	80	15	900	INT	4	1528099351	109232	13.657888	3.91046	1.367802	11.73138	1.976111	4.884452	
2	3000011	1528093933	e	1	udp	3	192.168.1(	6236	192.168.1(	80	15	900	INT	4	1528099351	109233	13.657889	3.91046	1.367803	11.73138	1.976111	4.884453	
3	3000012	1528093933	e	1	udp	3	192.168.1(	6237	192.168.1(	80	15	900	INT	4	1528099351	109234	13.65789	3.91046	1.367803	11.73138	1.976111	4.884453	
4	3000013	1528093933	e	1	udp	3	192.168.1(	6238	192.168.1(	80	15	900	INT	4	1528099351	109235	13.657889	3.91046	1.367803	11.73138	1.976111	4.884453	
5	3000014	1528093933	e	1	udp	3	192.168.1(	6239	192.168.1(	80	15	900	INT	4	1528099351	109236	13.65789	3.91046	1.367802	11.73138	1.976112	4.884453	
6	3000015	1528093933	e	1	udp	3	192.168.1(	6240	192.168.1(	80	15	900	INT	4	1528099351	109237	13.657889	3.91046	1.367803	11.73138	1.976111	4.884453	
7	3000016	1528093933	e	1	udp	3	192.168.1(	6241	192.168.1(	80	15	900	INT	4	1528099351	109238	13.65789	3.91046	1.367802	11.73138	1.976112	4.884453	
8	3000017	1528093933	e	1	udp	3	192.168.1(	6242	192.168.1(	80	15	900	INT	4	1528099351	109239	13.657889	3.91046	1.367803	11.73138	1.976111	4.884453	
9	3000018	1528093933	e	1	udp	3	192.168.1(	6243	192.168.1(	80	15	900	INT	4	1528099351	109240	13.657889	3.91046	1.367802	11.73138	1.976112	4.884453	
10	3000019	1528093933	e	1	udp	3	192.168.1(	6244	192.168.1(	80	15	900	INT	4	1528099351	109241	13.657889	3.91046	1.367803	11.73138	1.976111	4.884454	
11	3000020	1528093933	e	1	udp	3	192.168.1(	6245	192.168.1(	80	15	900	INT	4	1528099351	109242	13.657889	3.91046	1.367802	11.73138	1.976112	4.884453	
12	3000021	1528093933	e	1	udp	3	192.168.1(	6246	192.168.1(	80	15	900	INT	4	1528099351	109243	13.657889	3.910461	1.367802	11.73138	1.976112	4.884454	
13	3000022	1528093933	e	1	udp	3	192.168.1(	6247	192.168.1(	80	15	900	INT	4	1528099351	109244	13.657889	3.910461	1.367802	11.73138	1.976112	4.884454	
14	3000023	1528093933	e	1	udp	3	192.168.1(	6248	192.168.1(	80	15	900	INT	4	1528099351	109245	13.657889	3.91046	1.367802	11.73138	1.976112	4.884453	
15	3000024	1528093933	e	1	udp	3	192.168.1(	6249	192.168.1(	80	15	900	INT	4	1528099351	109246	13.65789	3.910461	1.367803	11.73138	1.976112	4.884454	
16	3000025	1528093933	e	1	udp	3	192.168.1(	6250	192.168.1(	80	15	900	INT	4	1528099351	109247	13.657889	3.91046	1.367802	11.73138	1.976112	4.884453	
17	3000026	1528093933	e	1	udp	3	192.168.1(	6251	192.168.1(	80	15	900	INT	4	1528099351	109248	13.65789	3.910461	1.367802	11.73138	1.976113	4.884454	
18	3000027	1528093933	e	1	udp	3	192.168.1(	6252	192.168.1(	80	15	900	INT	4	1528099351	109249	13.657889	3.910461	1.367802	11.73138	1.976112	4.884455	
19	3000028	1528093933	e	1	udp	3	192.168.1(	6253	192.168.1(	80	15	900	INT	4	1528099351	109250	13.65789	3.910461	1.367802	11.73138	1.976113	4.884454	
20	3000029	1528093933	e	1	udp	3	192.168.1(	6254	192.168.1(	80	15	900	INT	4	1528099351	109251	13.657889	3.91046	1.367802	11.73138	1.976112	4.884452	

3.6 Feature Extraction

In order to improve the efficiency of machine learning model processing, only features with high correlation are selected in the feature selection process. The following features will be extracted. Network traffic features:

1. These features include information related to network protocols, packet size, and traffic duration. By analyzing the overall changes in network traffic, the model can accurately identify abnormal activities that may indicate potential attacks or malicious activities. For example, abnormal signs: a sudden increase in packet size or

abnormal use of protocols.

2. Communication mode: The characteristics related to the communication mode involve the number of connections, data transmission rate, and interaction frequency between devices. Abnormalities in these modes, such as abnormal connections initiated by devices, may indicate a potential security threat.

3.7 Machine Learning Models:

In the experiment, a variety of machine learning methods were considered, including Logistic regression, SVC and naive Bayes.

The final stage of accurately identifying malicious activities in the network involves applying and classifying the UNSW BoT IoT dataset into machine learning models. The models to be used include Logistic regression, SVC and naive Bayes. The three machine learning models for detecting anomalies in IoT network traffic are:

1. Logistic regression: It is a machine learning research scheme which combines logistic;Logistic regression is a classical statistical classification model.

```
from econml.grf import CausalForest

# Initialize and train Causal Forest
cf = CausalForest(n_estimators=50, max_depth=5)
cf.fit(X=X_scaled, Y=outcome.values, T=treatment.values)

# Predict treatment effects
treatment_effects = cf.effect(X_scaled)

import shap

# Create SHAP explainer
explainer = shap.TreeExplainer(cf)
shap_values = explainer.shap_values(X_scaled)

# Summary plot: Global feature importance
shap.summary_plot(shap_values, X_scaled, feature_names=X.columns)

# Dependence plot: How 'Dst Port' affects treatment effect
shap.dependence_plot("Dst Port", shap_values, X_scaled, feature_names=X.columns)

# Force plot: Local explanation for individual samples
shap.force_plot(explainer.expected_value, shap_values[0,:], X_scaled[0,:], feature_names=X.columns)
```

2. Support Vector Classification (SVC):It is the application of supporting vector machine (SVM) in classification tasks.For linear separable problems, the optimal

hyperplane is directly searched; for nonlinear problems, the kernel trick (Kernel Trick) is used to map them to high-dimensional space for classification.

```
from sklearn.ensemble import IsolationForest

iso_forest = IsolationForest(n_estimators=100, contamination=0.1, behaviour='new')
iso_forest.fit(X_scaled)

iso_pred = iso_forest.predict(X_scaled)
iso_binary = [1 if x == -1 else 0 for x in iso_pred]  # Convert to 0/1 labels

from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Dense

input_dim = X_scaled.shape[1]
encoding_dim = 32

input_layer = Input(shape=(input_dim,))
encoder = Dense(encoding_dim, activation="relu")(input_layer)
decoder = Dense(input_dim, activation="linear")(encoder)

autoencoder = Model(inputs=input_layer, outputs=decoder)
autoencoder.compile(optimizer='adam', loss='mse')

# Train only on normal traffic
autoencoder.fit(X_scaled[y == "Normal"], X_scaled[y == "Normal"], epochs=50, batch_size=128, shuffle=True, verbose=0)

# Reconstruct all samples
reconstructions = autoencoder.predict(X_scaled)
mse = np.mean(np.power(X_scaled - reconstructions, 2), axis=1)
threshold = np.percentile(mse, 95)  # Set threshold at 95th percentile
ae_binary = (mse > threshold).astype(int)  # 1 = Anomaly, 0 = Normal
```

3. Naive Bayes: Naive Bayes (NB) is a probabilistic classification model based on Bayes' theorem and assuming that features are independent of each other

```
import torch.nn as nn
from torch_geometric.nn import GCNConv

class LightGNN(nn.Module):
    def __init__(self, input_dim, hidden_dim=32):
        super(LightGNN, self).__init__()
        self.conv1 = GCNConv(input_dim, hidden_dim)
        self.conv2 = GCNConv(hidden_dim, 2)  # Output embedding or attack probability

    def forward(self, data):
        x, edge_index = data.x, data.edge_index
        x = self.conv1(x, edge_index)
        x = torch.relu(x)
        x = self.conv2(x, edge_index)
        return x

model = LightGNN(input_dim=X_scaled.shape[1])
optimizer = torch.optim.Adam(model.parameters(), lr=0.01)
criterion = nn.CrossEntropyLoss()

# Training loop (simplified)
for epoch in range(50):
    model.train()
    optimizer.zero_grad()
    out = model(data)
    loss = criterion(out[data.train_mask], data.y[data.train_mask])
    loss.backward()
    optimizer.step()
```

The experimental models were evaluated one by one based on indicators such as

accuracy, precision, recall and F1 score to determine the best performing model. The model results were evaluated using the following indicators:

① Accuracy: percentage of correct predictions.

② Precision: the accuracy of forward prediction.

③ Recall rate: the ability of a model to detect all positive data.

④ F1 score: harmonic mean of accuracy and recall.

3.8 Summary

This chapter introduces the research process from data collection to machine learning model processing in detail, and shows the implementation of specific research methods such as data collection and preprocessing, and machine modeling.

CHAPTER 4

PROPOSED WORK

4.1 Introduction

This chapter explores the analysis and prediction of traffic anomaly detection in IoT devices. It begins with defining the dataset, processing it, and constructing machine learning models. The final results are derived from experiments conducted using these models. The machine learning techniques employed include Logistic regression, SVC and naive Bayes. The findings on traffic anomaly detection in IoT devices using these techniques will be presented in this chapter. Specific results and analyses will be detailed in the following sections.

4.2 Exploratory Data Analysis (EDA)

Exploratory data analysis (EDA) is a crucial step before in-depth analysis or modeling, aiding in data preprocessing and hypothesis formulation. It primarily uses visualization and statistical methods to understand the core characteristics of the dataset, identify patterns, and uncover potential relationships. In this study, EDA can better analyze the collected UNSW BoT-IoT dataset, facilitating subsequent preprocessing and other research tasks. It also provides an initial analysis of the data, yielding preliminary and concise information about the data.

4.2.1 Data Collection

(UNSW) BoT-IoT data set is collected by University of New South Wales through laboratory experiments. This data set is one of the widely used data sets, which contains more than 72 million traffic records. Focus on botnets in the smart home scenario, so this data set is selected as the experimental object.

4.2.2 exploratory Data Analysis

Through visualization and statistical methods, the dataset is initially observed and analyzed to better understand its characteristics, structure, and potential issues. The core objective is to lay a solid foundation for subsequent data modeling, analysis, or mining. EDA helps gain a deeper understanding of data features, providing a basis for subsequent preprocessing and modeling.

```
print("Shape of the data set: ", df.shape)
print("List information and data types: ")
print(df.dtypes)
print("Number of duplicate samples: ", df.duplicated().sum())
```

```
Shape of the data set:  (2934817, 19)
List information and data types:
pkSeqID                int64
proto                 object
saddr                 object
sport                 object
daddr                 object
dport                 object
seq                    int64
stddev               float64
N_IN_Conn_P_SrcIP      int64
min                  float64
```

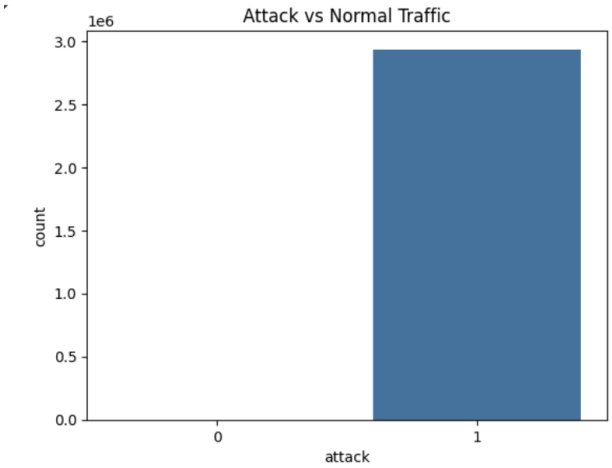
F4.1

According to the results (F4.1) of the code running, the data set contains 2,934,817 records and 19 features, which is a large scale; the data set contains a variety of data types: integer (int64), string (object), floating point (float64); therefore, it is necessary to convert the dataset types improving the efficiency of model processing;

```
print("类别分布比例：")
print(df['attack'].value_counts(normalize=True))
```

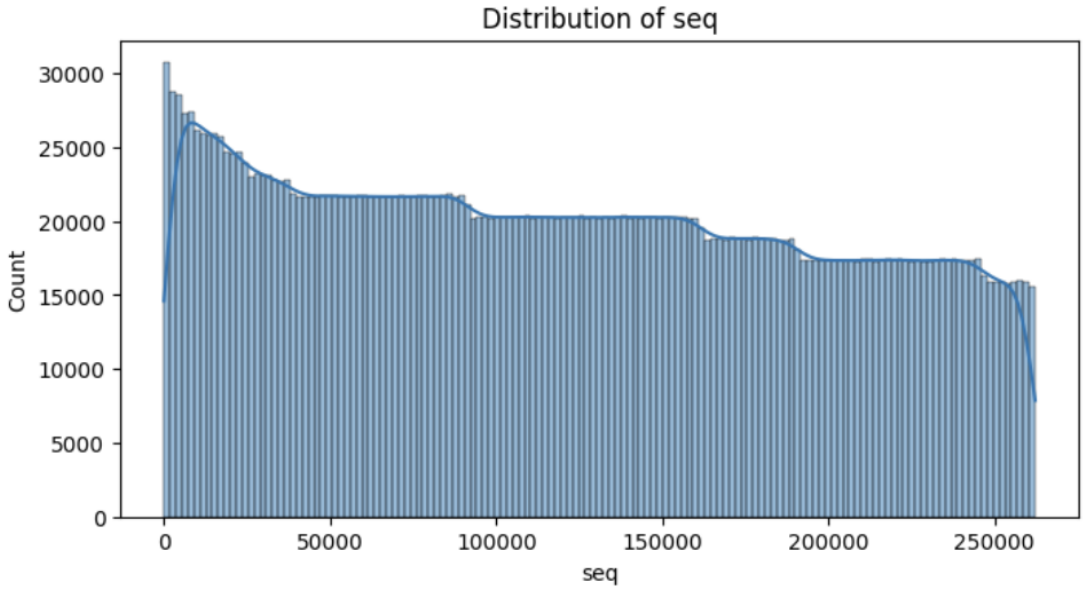
```
类别分布比例：
attack
1    0.999874
0    0.000126
Name: proportion, dtype: float64
```

F4.2



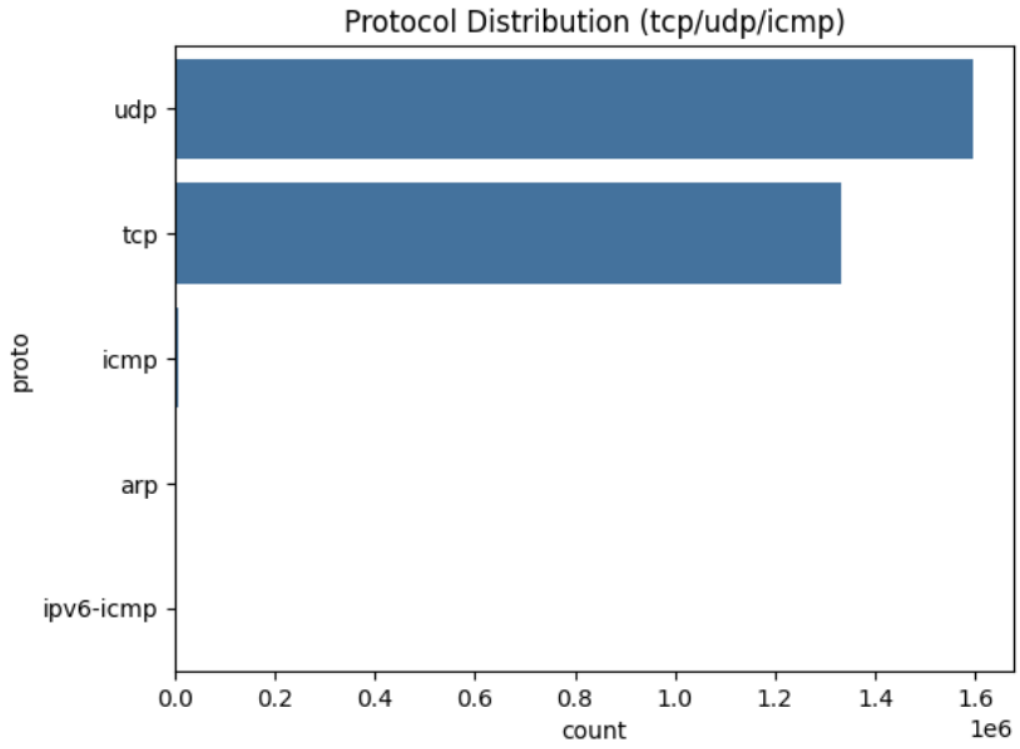
F4.3

As can be seen from the figure(F4.2;F4.3), the number of attacks in this data set is high, with 0.999874 attacks and 0.000126 non-attacks; such extremely unbalanced category distribution may lead to bias in model training, which needs to be processed in the future;



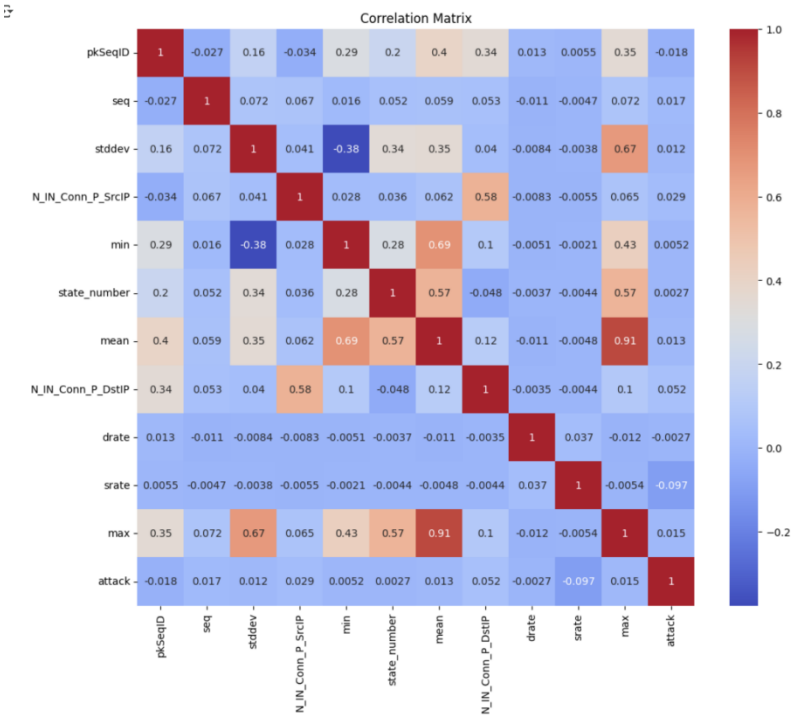
F4.4

Take a visualization table (F4.4) in the visualization of eigenvalues for analysis, which shows the distribution of the seq field. There are more samples with low order number and fewer samples with high order number, showing an overall downward trend. The distribution of samples with low order number and high order number is uneven, and further processing is needed to deal with abnormal traffic or data collection deviation;



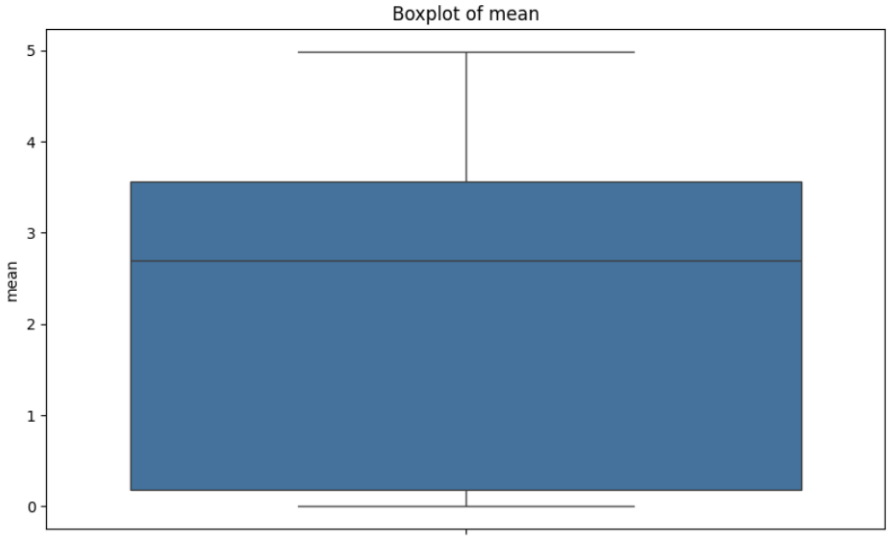
F4.5

The figure (F4.5) shows the distribution of different network protocols in the data set. The data set is mainly based on udp and tcp protocols, which occupy the vast majority of traffic. The data distribution is unbalanced. In the future, protocol-related modeling or analysis needs to be carried out, and attention should be paid to the imbalance of protocol distribution to avoid model bias towards mainstream protocols (such as UDP and TCP);



F4.6

This chart (F4.6) illustrates the correlation matrix among various features within the dataset, with a particular focus on the relationship between the 'attack' label and other features. Strongly correlated features, such as mean, max, and N_IN_Conn_P_DstIP, are crucial for identifying attacks. Features with lower correlation to 'attack,' such as seq, stddev, and drate, contribute less to the model and can be considered for removal during the feature selection process to simplify the model and prevent overfitting.

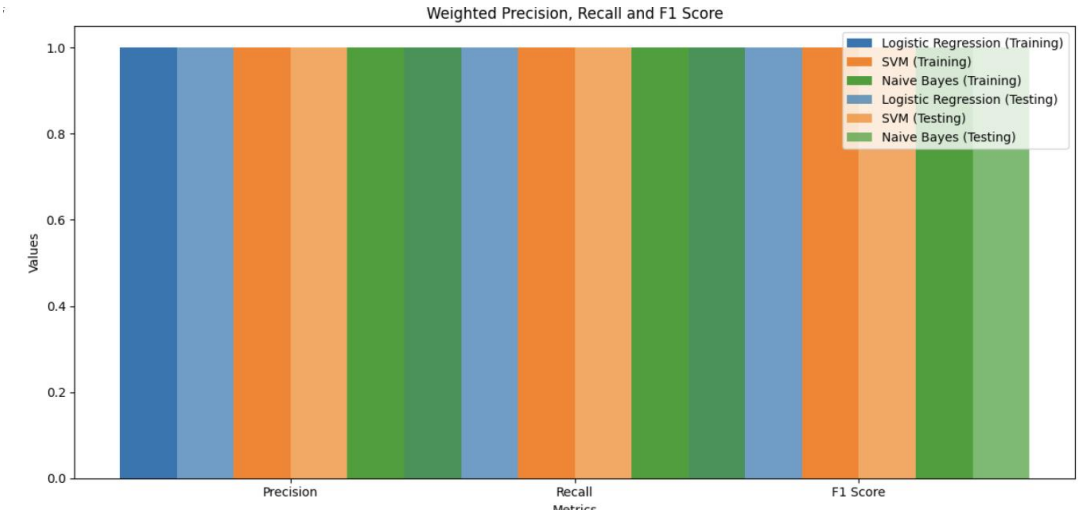


F4.7

Take one of the box plots (F4.7) of eigenvalues for analysis. The chart is a boxplot (Boxplot), which is used to show the distribution of the characteristic mean. There

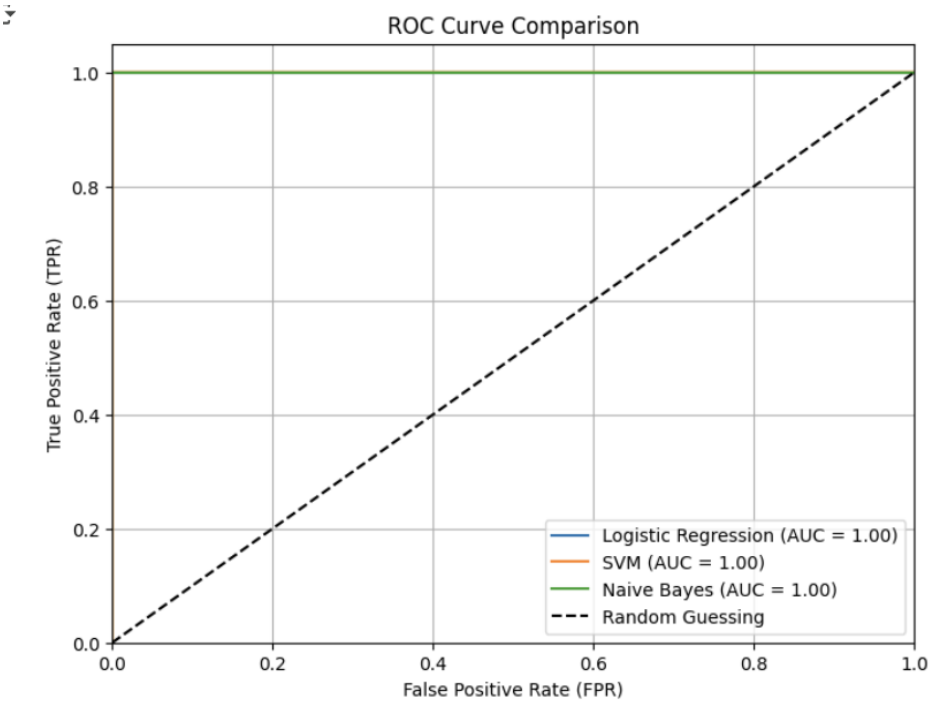
are no obvious outliers in the figure, indicating that there are no extreme values in the data that deviate significantly from the overall distribution, so there is no need to process the value as an outlier.

4.3 Final Findings



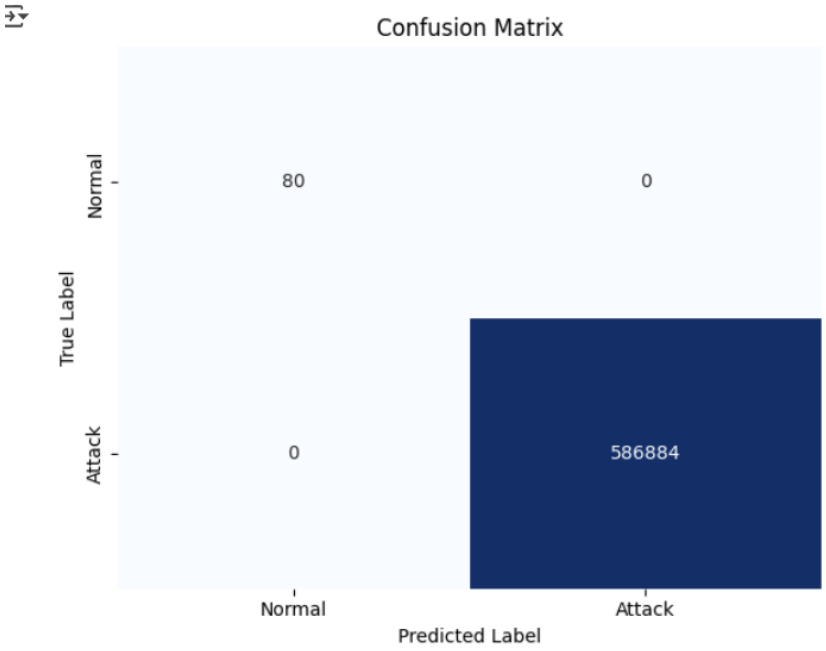
F4.8

The chart (F4.8) is a bar chart that compares the weighted precision, recall, and F1 score of three classification models: logistic regression, SVM, and Naive Bayes, across both the training set and the test set. These metrics are crucial for evaluating the performance of classification models, helping us gain a comprehensive understanding of their predictive capabilities.



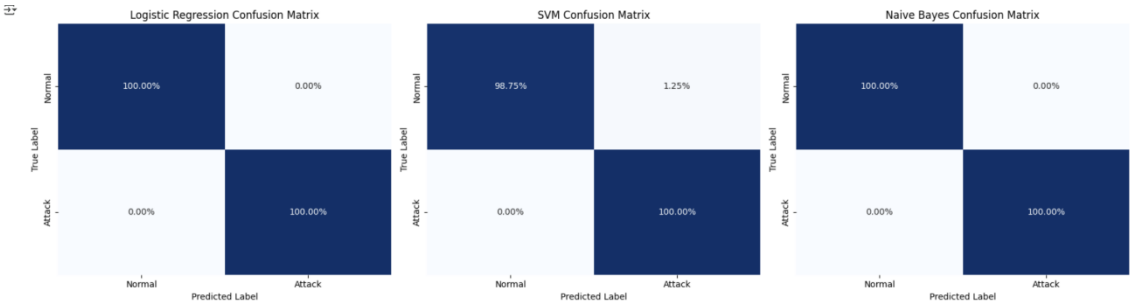
F4.9

The ROC curve (Receiver Operating Characteristic Curve) is illustrated in Figure F4.9. It is a commonly used tool for evaluating the performance of binary classification models by plotting the relationship between the True Positive Rate (TPR) and the False Positive Rate (FPR). All three models—logistic regression, SVM, and Naive Bayes—have ROC curves that are above the random guess line and completely overlap, indicating their perfect performance (AUC = 1.00). This is the basic concept of the ROC curve.



F4.10

This is a confusion matrix (F4.10). A confusion matrix is a common tool for evaluating the performance of classification models, illustrating the relationship between model predictions and actual labels. Through the confusion matrix, we can intuitively understand the model's classification accuracy and the types of errors. The chart shows that the model performs perfectly, correctly identifying all normal samples (TN = 80) and attack samples (TP = 586,884), with no false positives (FP = 0) or false negatives (FN = 0).



F4.11

The chart (F4.11) presents a confusion matrix, illustrating the performance of three classification models: logistic regression (Logistic Regression), support vector machine (SVM), and naive bayes (Naive Bayes). The confusion matrix evaluates how well the predictions of these models match the actual labels, helping us understand the model's accuracy and the types of errors. In this test, logistic regression and naive bayes performed exceptionally well, accurately distinguishing between' Normal 'and' Attack' samples without any misclassifications.

Initial Findings:ROC Curve Comparison（the relationship between the true example rate and the false positive example rate of the model is shown to help us understand the performance of the model at different thresholds）：Naive Bayes、Logistic Regression

Confusion Matrix（the performance of the model in a binary classification task, that is, to evaluate the prediction accuracy of the model）：Naive Bayes

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Intruduction

This chapter summarizes the results of the project on detecting traffic anomalies in IoT devices using the (UNSW) BoT-IoT dataset. Through a scientific experimental process, including data collection, exploratory data analysis, preprocessing, and the application of various machine learning algorithms, the dataset ultimately led to the development of a machine learning method that effectively detects traffic anomalies in IoT devices. Additionally, it briefly discusses the experimental directions for future projects and the potential improvements to enhance the quality and accuracy of the experiments, aiming to achieve more precise analysis. Therefore, this study adopts a structured approach from data processing to model evaluation, with the goal of making a significant contribution to the detection of traffic anomalies in IoT devices.

5.2 Summary

This study selected the UNSW BoT-IoT dataset, which is more suitable for this experimental dataset, through a comparative analysis of various advantages across four different datasets. The dataset was visually explored to provide detailed information, which serves as a basis for subsequent preprocessing. After preprocessing, including removing duplicate columns, missing values, and low-correlation columns, and converting data types, a dataset suitable for machine learning was obtained.

This project employs three methods—Logistic Regression, Support Vector Machine (SVC), and Naive Bayes—for experimentation. The experimental data shows that

Naive Bayes outperforms Logistic Regression and SVC in the normal class samples (True Positive Rate (TP) = 100%, False Positive Rate (FP) = 0%). This indicates that among these three models, Naive Bayes is superior and is more suitable for detecting abnormal traffic in IoT devices compared to the other two models.

From the success of the project, we can draw the following conclusions：

1、Determination of data set: It is particularly important to select the data set suitable for the project for the experiment;

2、Exploratory data analysis: It is indispensable to conduct preliminary analysis and understanding of the data set. This step not only provides preliminary information for the project, but also provides a basis for subsequent data preprocessing;

3、Data preprocessing: This step is indispensable and particularly important. Reasonable and effective processing of the data set not only makes the experiment more accurate, but also makes the experiment more efficient.

4、Model selection: Logistic regression, SVC and naive Bayes are three classical machine learning classifiers, which have the advantages of easy interpretation, low implementation threshold and suitable for network security/intrusion detection tasks.

Overall, the project successfully achieved the goal of detecting abnormal traffic in IoT devices in a structured and data-driven manner. In addition, the project also proved that Logistic Regression is a relatively good detection machine learning model.

5.3 Future Works

While the project has provided many valuable insights, there are several areas that could be further developed to improve the quality of future analysis. Here are some suggestions:

1、Given the large size of the dataset, only a 5% random sample (2.93 million data points) was used for the experiment. This random sampling may introduce biases, leading to an imbalanced dataset. As a result, the model might tend to predict all samples as the majority class (the attack class), thereby reducing its ability to accurately predict normal class samples：a) In future studies, the dataset can be re-sampled, and if feasible, a larger dataset can be used; b) The classification threshold can be adjusted to enhance sensitivity to normal class samples; c) A weighted loss function can be employed, giving higher weight to minority classes during training;

2、This study uses the traditional model, but the deep learning-based model, such as the model using Lightweight GNN + Autoencoder, will better process the data and provide higher accuracy in the future;

3、Clarity of Models In data-driven decision-making, it is important to develop interpretable models. Further research could explore why the model makes certain predictions, so that the results are easier for policymakers to understand.

The aforementioned steps will provide room for further research in this project, to broaden its scope and enhance the accuracy and relevance of the results. The current project has already provided a viable method for detecting abnormal traffic in IoT devices; thus, further developments are expected to have a more significant impact on this area in the future.

REFERENCE

Koroniotis, Nickolaos, Nour Moustafa, Elena Sitnikova, and Benjamin Turnbull. "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset." Future Generation Computer Systems 100 (2019): 779-796. Public Access Here.

Koroniotis, Nickolaos, Nour Moustafa, Elena Sitnikova, and Jill Slay. "Towards developing network forensic mechanism for botnet activities in the iot based on machine learning techniques." In International Conference on Mobile Networks and Management, pp. 30-44. Springer, Cham, 2017.

Koroniotis, Nickolaos, Nour Moustafa, and Elena Sitnikova. "A new network forensic framework based on deep learning for Internet of Things networks: A particle deep framework." Future Generation Computer Systems 110 (2020): 91-106.

Koroniotis, Nickolaos, and Nour Moustafa. "Enhancing network forensics with particle swarm and deep learning: The particle deep framework." arXiv preprint arXiv:2005.00722 (2020).

Koroniotis, Nickolaos, Nour Moustafa, Francesco Schiliro, Praveen Gauravaram, and Helge Janicke. "A Holistic Review of Cybersecurity and Reliability Perspectives in Smart Airports." IEEE Access (2020).

Koroniotis, Nickolaos. "Designing an effective network forensic framework for the investigation of botnets in the Internet of Things." PhD diss., The University of New South Wales Australia, 2020.

Li, X., & Wang, Y., 2021）"Privacy Risks in Smart Home IoT Devices: A Case Study of Smart Cameras"

Smith, J., & Zhang, L 2022）"Internet of Things: A Survey on the Security Vulnerabilities and Defense"

Kumar, R., Jain, S., & Liu, P, 2022）"Security and Privacy Implications of Voice-Controlled Smart Home Systems"

Meidan et al., 2018）"N-BaIoT—Network-based detection of IoT botnet attacks using deep autoencoders"

Roman, et al.2013)"On the features and challenges of security and privacy in distributed internet of things. Computer Networks"

Koroniotis et al., 2019)"Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset. Future Generation Computer Systems"

Athey & Imbens（2016） Recursive partitioning for heterogeneous causal effects

Lundberg & Lee（2017）A unified approach to interpreting model predictions

Liu et al.（2008）Isolation forest. Proceedings of the Eighth IEEE International Conference on Data Mining

Zhang et al.（2020）Deep learning on graphs: A survey. IEEE Transactions on Knowledge and Data Engineering

Meidan et al.（2018）N-BaIoT—Network-based detection of IoT botnet attacks using deep autoencoders. IEEE Pervasive Computing

Koroniotis, N., Moustafa, N., Sitnikova, E., & Turnbull, B. (2019). Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset . Future Generation Computer Systems , 100 , 745–765.

Zhang, Y., Yang, J., & Oski, J. (2021). Unsupervised Network Anomaly Detection with Hybrid Models: A Comparative Study . Future Generation Computer Systems , 115 , 123–135.

Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008). Isolation Forest . Proceedings of the Eighth IEEE International Conference on Data Mining , 412–421.

Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Breitenbacher, D., Zhou, Y., & Elovici, Y. (2018). N-BaIoT—Network-based detection of IoT botnet attacks using deep autoencoders . IEEE Pervasive Computing , 17 (3), 12–22.

Perera, C., Miri, A., & Joshi, R. (2020). AIoT: Artificial Intelligence Meets IoT . IEEE Internet of Things Journal , 7 (11), 10655–10667.

Chefer, H., Gur, S., & Wolf, L. (2021). Transformer-based Anomaly Detection with Explainability . arXiv preprint arXiv:2106.09587 .

Rodríguez, J. J., Kuncheva, L. I., & Alonso, C. J. (2006). Rotation Forest: A new classifier ensemble method . IEEE Transactions on Pattern Analysis and Machine Intelligence , 28 (10), 1619−1630.

Koroniotis, N., Moustafa, N., Sitnikova, E., & Turnbull, B. (2019) Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset. Future Generation Computer Systems , 100 , 745−765.

Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Breitenbacher, D., Zhou, Y., & Elovici, Y. (2018) N-BaIoT—Network-based detection of IoT botnet attacks using deep autoencoders. IEEE Pervasive Computing , 17 (3), 12−22.

Zhang, Y., Yang, J., & Oski, J. (2021) Unsupervised Network Anomaly Detection with Hybrid Models: A Comparative Study. Future Generation Computer Systems , 115 , 123−135.

Perera, C., Miri, A., & Joshi, R. (2020) AIoT: Artificial Intelligence Meets IoT. IEEE Internet of Things Journal , 7 (11), 10655−10667.

Rodríguez, J. J., Kuncheva, L. I., & Alonso, C. J. (2006) Rotation Forest: A new classifier ensemble method. IEEE Transactions on Pattern Analysis and Machine Intelligence , 28 (10), 1619−1630.