Doctoral Dissertation

Deep learning-based cyberattacks behaviour analysis for IoT ecosystem

Enkhtur Tsogbaatar

March 16, 2022

Graduate School of Information Science
Nara Institute of Science and Technology

A Doctoral Dissertation
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Enkhtur Tsogbaatar

Thesis Committee:

Professor Youki Kadobayashi	(Supervisor)
Professor Yasuhiko Nakashima	(Co-supervisor)
Associate Professor Taenaka Yuzo	(Co-supervisor)
Assistant Professor Doudou Fall	(Co-supervisor)
Associate Professor Daisuke Miyamoto	(The University of Tokyo) (Co-supervisor)
Assistant Professor Monowar Bhuyan	(Umeå University, Sweden) (Co-supervisor)

Deep learning-based cyberattacks behaviour analysis for IoT ecosystem*

Enkhtur Tsogbaatar

Abstract

The proliferation of the Internet of Things (IoT) and the evolution of technologies enable us to create multitudes of personal applications and services to make our daily life more comfortable and easier by adopting automation in daily operations. This growth boosts explosively for vulnerabilities and emerging attacks in parallel, which are for both IoT devices and applications. Most IoT devices are inherently vulnerable due to insecure design, implementation, and configuration. On the other hand, IoT applications are more vulnerable than devices during service time because attackers can exploit those vulnerabilities to compromise a secure system in milliseconds. One of such attacks is a low-rate and economic Distributed Denial of Service (DDoS) attack that targets to deteriorate users Quality of Service (QoS). Deep learning (DL) has a notable capability to assist, analyze user behaviours in complicated IoT ecosystems. DL can learn complex behavioral patterns of IoT devices and applications more effectively than conventional learning techniques, and it can detect attacks in IoT with maximum accuracy. Additionally, Software-Defined Networking (SDN) has appealing features such as flexibility, dynamicity in network operations, and resource management to support detection systems.

This thesis aims to design, develop, and implement comprehensive approaches to detect and predict cyberattacks in IoT devices and applications by integrating DL with SDN infrastructures. The major contributions of this thesis are in two folds. *Firstly*, we focus on IoT devices to address problems such as class imbalance, dynamic attack detection, and data heterogeneity. We propose a deep

*Doctoral Dissertation, Graduate School of Information Science,
Nara Institute of Science and Technology, March 16, 2022.

ensemble learning framework known as DeL-IoT to uncover and predict anomalies in IoT devices by integrating SDN. The DeL-IoT employs deep and stacked autoencoders to extract handy features for stacking into an ensemble learning model. Moreover, this framework yields efficient detection of anomalies, manages traffic flows dynamically, and forecasts both short and long-term device status for early action(s). Our experimental analysis proved that DeL-IoT performs well even in an 1% imbalanced dataset and outperforms 2%-3% when comparing F1 and MCC scores with single models. *Secondly*, several IoT applications are latency-sensitive and mission-critical when providing services over edge-clouds. However, attackers attempt to interrupt services deployed in edge-clouds by imitating legitimate behaviour of users, one such attack is Very Short Intermittent DDoS (VSI-DDoS) attack that targets services to degrade users QoS. These attacks send intermittent bursts (in tens of milliseconds) of legitimate HTTP requests to the target services for degrading users QoS. Because of the stealthy nature of VSI-DDoS, it is challenging to pinpoint the root-cause when the system resource usage remains at a moderate level. Therefore, we propose a 1D-CNN-based (Convolution Neural Network) DL method for detecting VSI-DDoS attacks in IoT applications. The experimental results on both testbed and benchmark datasets proved that our proposed method achieves maximum detection accuracy of 99.3% and 100% which gives improvement by 33.15%-0.01% detection in comparison to baseline approaches, respectively.

Keywords:

Internet of Things (IoT), anomaly detection, Software-Defined Networking (SDN), deep ensemble learning, autoencoders, Probabilistic Neural Networks (PNNs), VSI-DDoS, QoS, Convolution Neural Network (CNN)

Contents

List of Figures

1 Introduction

1.1 Motivation

The Internet of Things (IoT) comprises billions of connected devices that can share and collect data over the Internet. Business Insider's 2020 IoT report projects that the number of IoT devices is expected to reach 41 billion by 2027 [1]. In addition, IoT has now a wide range of live applications such as transportation, smart home, healthcare, industry, smart environment, smart city as well as social gaming robots and personal. Simultaneously, the Industrial Internet of Things (IIoT) is empowered by more momentum as businesses integration between Artificial Intelligence (AI), Big-data and IoT technologies [2]. Based on Business insights's report, the global IoT market is projected to grow from $381.30 billion in 2021 to $1,854.76 billion in 2028 [3]. Healthcare constitutes the majority of this IoT market or 41%, followed by industry with 33% and energy with 7%. Other sectors, such as transportation, agriculture, urban infrastructure, security, and retail have about 15% of the IoT market totally [4]. Such rapid growth of IoT allows billions of IoT devices to be connected and exchange data for various applications and it has brought significant benefits to our lives, society, and industries.

The technology used has still not matured enough to provide security for IoT devices, communications, and applications in parallel. Frequently, IoT devices are vulnerable due to limited resources, which makes them expose to cyberattacks [5]. Thus, an increase in connected IoT devices offers larger platforms for adversaries to gain faster access to IoT devices and applications and utilize them to launch large-scale attacks. For instance, Mirai Botnet caused massive Distributed Denial of Service (DDoS) attacks to a DNS server of Dyn DNS provider [6] and brought

down several sites, including GitHub*, Reddit†, Netflix‡, and Airbnb§ for several hours. Additionally, multiple zero-day attacks have been observed in IoT [7]. The applications of IoT are growing rapidly and penetrating most of the existing industries. Thus, securing IoT devices and applications, which have already been deployed or are in the process of deployment, is an increasing challenge for manufacturers, users, and service providers. Consequently, it is imperative to devise efficient and comprehensive methods for detecting cyberattacks in IoT devices and applications from small-scale to large-scale systems.

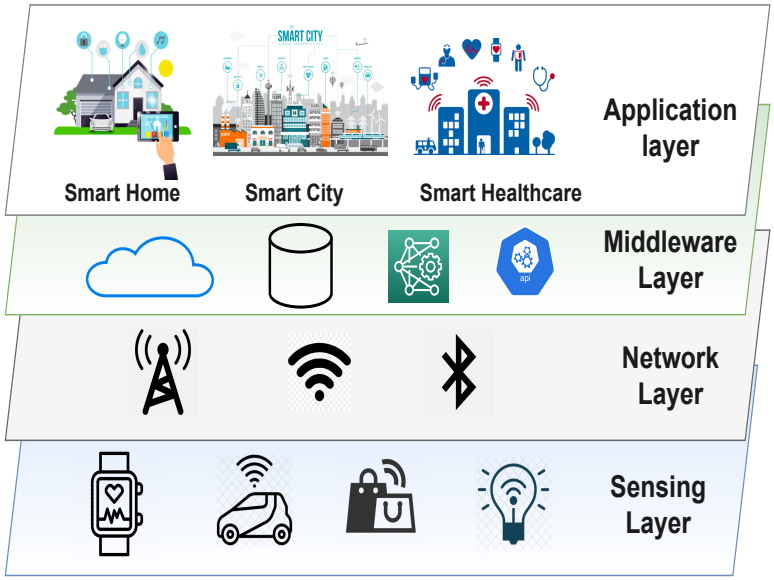


Figure 1.1: Layers in IoT ecosystem.

Figure 1.1 illustrates IoT ecosystem or environment that comprises four primary layers in it. The first layer includes the use of various sensors and actuators to perceive the data or information to accomplish various tasks. The acquired data is then transmitted via a communication network in the second layer. Most of the evolving IoT applications deploy in the third layer, called a middleware layer, to act as a bridge between the network and application layer. Finally, on the fourth layer, there are various IoT based end-to-end applications such as

*https://github.com/
†https://www.reddit.com/
‡https://www.netflix.com/
§https://www.airbnb.com/

smart home, smart city, smart transport, smart healthcare, etc. All of these four layers have security issues specific to them [8].

1.2 Problem Formulation and Assumptions

The IoT ecosystem is unique, challenging, and demands different security requirements that may not be found or solved with existing security solutions. For instance, since IoT devices are lightweight, they don't have enough computational resources such as disk space, memory, and battery by characteristics [9]. Hence, attackers employ the vulnerabilities, weaknesses, and constraints to make successful attacks within a short period of time. The following characteristics make it challenging to secure IoT devices and applications.

- *Resource constraints*: IoT devices have resource limitations that prevent having sophisticated security features and tools installed.
- *Low cost*: The worldwide IoT market is increasing exponentially with the competition. However, adding comprehensive cybersecurity features is expensive at both the software and hardware level in IoT. Thus, IoT devices are produced by manufacturers without comprehensive cybersecurity features [10].
- *Insecure devices*: IoT devices have limited resources, thus, every extra security layer requires more processing capabilities, battery power, and consumes additional energy. Hence, manufacturers do not consider an extra security layer as one of the significant factors for producing IoT devices [11].
- *Heterogeneous/diverse*: IoT ecosystems are highly heterogeneous with regard to communication medium and protocols, platforms, and devices [12]. Thus, it is challenging to devise an efficient generalized intrusion detection and prevention system (IDS/IPS) for heterogeneous IoT ecosystems [13].
- *Intrusive*: Since, some IoT devices collect highly sensitive personal information, it increases privacy concerns. For instance, in a smart home, motion sensors or security cameras also gather sensitive personal information as

well as a medical wearable device collect information of a patient's heart rate, blood pressure, location, and daily physical activity. Mainly, the data gathered by these devices is made available to the user through a smartphone application. Thus, the sensitive personal data can be leaked if the device, the application or the communication channel between the device and the application are not adequately secured [14].

- *Critical*: Mission-critical IoT applications are highly imperative such as healthcare, power systems industries and autonomous vehicles. In addition, new types of mission-critical IoT applications are also emerging in areas such as wearables, smart cities, and smart homes. Since any compromise of the IoT device or applications can lead to severe physical and human damages, all of these are always required to work as expected without any glitches [15]. For instance, the probability for attackers to take control of autonomous self-driving cars to cause intentional accidents will be a severe security issue in the near future [16].

- *Ubiquitous*: It explains the point that IoT devices have penetrated every aspect of our daily lives. In the near future, IoT devices will be everywhere around us. They will become imperative to our daily life and we will depend more and more on them. Devices will no longer need human intervention to function properly, such as a bulb connected to a motion sensor that turns on or off automatically. Most people will not even realize how to rely on IoT until a cyberattack happens. Despite its ubiquitous nature, most people regularly ignore the security implications of the IoT [13].

- *Insecure interface*: For accessing IoT services, the interfaces utilized through web, mobile, and cloud are vulnerable to different cyberattacks such as SQL injection or Cross-site Scripting (XSS) which may severely affect the data privacy [9]. Symantec reports that 10 security issues in 15 web portals are utilized to manage IoT devices, including serious issues that can lead to unauthorized access to the backend systems. Occasionally, the interface may not allow the possibility to alter the default password. A strong password recovery mechanism can also be absent. Interfaces might not implement an account lockout mechanism causing them vulnerable to brute-

force attacks. They might also be susceptible to account enumeration attacks [14].

- *Abundant*: The data generated by IoT devices are abundant. As IoT devices are increasing rapidly, they will soon generate huge amounts of data. IDC expected that there will be 55.7 billion connected IoT devices, generating 73.1 zettabytes (ZB) of data by 2025 [17].

Moreover, based on the characteristics mentioned above and the lack of general awareness of the cybersecurity of manufacturers, users, and service providers bring additional challenges and vulnerabilities when developing solutions for securing IoT devices and applications. If efficient and comprehensive cybersecurity measures were not developed and implemented in IoT devices and applications, then it is hard to expect on demand services or benefits either from each device or deployed applications. Attackers can connect easily with large bot networks. Further, a single infected IoT device can connect to a particular network that generates risk to all other vulnerable IoT devices in that network as well. The aforementioned process can cause thousands of IoT devices and applications to be compromised without anyone aware of them, and it offers power to adversaries for employing a command and control server (C&C) and launching large-scale attacks on critical systems. Although the utilization of smart IoT applications is aimed to improve the overall quality of life of the citizens, it comes with a threat to the privacy of the citizens. For instance, there are IoT applications utilizing which parents can keep track of their child. However, if such applications are compromised by attacker, then the safety of the child can come to risk [8]. Hence, it is crucial to propose comprehensive security solutions for detecting and predicting cyberattacks in the IoT ecosystem by considering the limitations of IoT devices and applications.

Artificial Intelligence (AI) is continually changing our lives and being applied in wide areas such as autonomous vehicles, IoT systems, smart home, city, etc. When developed AI algorithms play a key role to bring too much convenience to our society, they are also vulnerable to attacks at the same time. AI systems hacked by attackers may lead to leakage of user information, incorrect result of classification, property loss, wrong decision making, etc. However, machine

learning (ML) and Deep Learning (DL) methods have been applied to detect, automate, predict, and root-cause analysis of critical systems [18]. DL is extensively used in computer vision, speech recognition, robotics, natural language processing, and numerous other applications but still less in the cybersecurity domain. Compared to traditional ML techniques, DL methods have a number of advantages that are required when developing cybersecurity solutions. *First*, the utilization of multiple hidden layers within a neural network structure indicates that DL can identify complex nonlinear relationships among features. *Second*, popular DL architectures, such as convolutional neural network (CNN) long short-term memory (LSTM), autoencoders, and transformer networks have ability to extract and determine useful features directly from raw data instead of relying on handcrafted statistical features as performed in the traditional ML [4]. *Third*, DL is especially well suited for dealing with 'big data' challenges [4]. With billions of IoT devices interconnected together to sense and share information worldwide, IoT systems typically generate an enormous amount of data, typically composed of devices and applications deployed in the edge. DL has a notable capability to assist in analyzing users behaviour in complicated IoT systems. Moreover, DL could allow IoT devices to learn complex behavioural patterns more effectively than traditional learning techniques.

1.3 Research Objectives and Contributions

1.3.1 Research objectives

The objective of this research is to devise, develop efficient and comprehensive anomaly detection methods to detect and predict cyberattacks against IoT devices and applications using DL approaches when deploying services on the edge.

1. **RO1.1**: To devise and develop an efficient and comprehensive cyberattack detection and prediction approach for IoT devices that employ SDN and deep learning (DL) to address multiple problems such as data heterogeneity, data imbalance, dynamic flow management, and prediction of the future status of IoT devices.
    - To develop a dataset with diverse and emerging cyberattacks that

carries all characteristics such as data heterogeneity, data imbalance, dynamic network flow management, and predicting future status of IoT devices.

- To address those challenges, how deep learning approaches help to adapt dynamic patterns of attackers, specifically in heterogeneous scenarios, data imbalance, dynamic flow management using SDN controller and DL.
- To predict the IoT devices' future status by using DL before compromising them.
- To validate the proposed methods using testbed and benchmark datasets.

R.O1.1: The objective is to devise and develop an effective comprehensive approach to detect and predict the cyberattacks in IoT devices. First, due to lack of datasets that carries the behaviour mentioned above, we investigate to setup a testbed and prepare a dataset to meet these requirements like data heterogeneity, data imbalance, dynamic flow management, and prediction of the future status of the IoT devices. Second, we investigate the proper solution for each challenge, and we find that deep ensemble learning approach can address the data heterogeneity and data imbalance problems. In addition, our experimental results show that deep ensemble learning approaches give the highest detection accuracy for the imbalanced dataset compared with the single DL approaches. Regarding dynamic flow management of the IoT devices, we observe that SDN can (i) manage the data flow of IoT devices well, (ii) isolate the infected IoT devices using the network controller, and (iii) mitigate and managing future cyberattacks at early. The details of this work is presented in Chapter 4.

2. **RO1.2**: To design and develop effective method to detect emerging cyberattacks in IoT application using DL.

    - To generate emerging cyberattacks in IoT applications in a testbed environment in order to develop emerging cyberattacks datasets.
    - To develop an effective DL approach for detecting emerging cyberattacks in IoT applications.

- To evaluate the impact of emerging cyberattacks in IoT applications.

R.O1.2: The primary objective is to generate a dataset with the emerging attacks in IoT applications and to develop an effective approach for detecting the emerging attacks in IoT applications. First, we investigate the emerging cyberattacks and observe that there is a lack of solutions for assessing and detecting IoT applications against Very Short Intermittent Distributed Denial of Service (VSI-DDoS) attacks. Such attacks send intermittent bursts (in tens of milliseconds) of legitimate HTTP requests to the target services for degrading providers Quality of Service (QoS). Thus, first, we generate VSI-DDoS attacks data by deploying IoT application in a testbed environment using Time Series Benchmark Suite (TSBS)* and create IoT application dataset for VSI-DDoS attacks. Second, we evaluate the impact of the VSI-DDoS attacks in IoT applications. From experimental evaluation, we find that the VSI-DDoS attacks can easily bypass a state-of-art intrusion detection system and it makes a significant impact on the QoS of the legitimate users of target services in deployed IoT applications. Finally, we devise and develop an 1D-CNN based DL approach for detecting VSI-DDoS attacks in IoT applications to ensure secure services towards users. The aforementioned proposal details are given in following Chapter 5.

1.4 Contributions

To address the issues mentioned in previous Section, this thesis contributes the following as outcomes.

1.4.1 Research contributions

- We propose DeL-IoT, a deep ensemble learning approach to detect anomalies in IoT devices employing SDN.
  - It utilizes the principles of deep autoencoders and probabilistic neural networks.

*https://github.com/timescale/tsbs#appendix-i-query-types

  - It can detect dynamic attacks and handle the data imbalance problems.

- The deployment of a learned model within SDN controller makes the DeL-IoT system effective and reliable for dynamic flow management.
- DeL-IoT introduces an IoT device status prediction mechanism for forecasting anomalies and taking preventive measures before interrupting a device.
- We prepare a new VSI-DDoS IoT applications dataset with diverse attack scenarios and make it available for public use to fill the research gap.
- We propose a 1D-CNN-based deep learning approach to detect VSI-DDoS attacks early for IoT applications.
- Systematic and extensive experimental analysis using testbed and benchmark datasets, showing the proposed method is superior to competitors in terms of accuracy, $F1$, Matthews Correlation Coefficient (MCC) measures, and prediction scores.

1.5 Outline of the Thesis

The rest of the thesis is organized as follows. In Chapter 2, we present preliminaries of cyberattacks in the IoT. In addition, we discuss how to manage IoT devices' network traffic flow utilizing SDN, and how important the dataset is for developing solutions against cyberattacks. Finally, we describe common evaluation criteria to measure the performance of the proposed methods. In Chapter 3, we present the prior research on IoT cyberattacks detection and prediction methods utilizing the SDN, machine learning, and deep learning approaches. In Chapter 4, we explain our proposed deep ensemble learning approach that integrates with SDN for detecting and predicting anomalies in IoT devices. The Chapter 5 presents our proposed 1D-CNN-based approach for detecting VSI-DDoS attacks in IoT applications. The Chapter 6 concludes this thesis. Finally, in Chapter 7, we enumerate a list of dimensions as future work.

2 Background

The fundamentals of our research topic and methodologies including cyberattacks in the IoT ecosystem, datasets, software defined network (SDN), and assessment criteria, are presented in this chapter. First, we present an overview of the cyberattacks in IoT. Second, we discuss how to utilize SDN to manage the network traffic flow of IoT devices, as well as roles of datasets for devising cyberattacks detection systems. Finally, we describe how to evaluate the performance of detection methods or systems using common evaluation criteria.

2.1 Cyberattacks in the IoT ecosystem

2.1.1 A taxonomy of IoT attacks

The IoT ecosystem has been subjected to a variety of cyberattacks. The comprehensive taxonomy of DL techniques utilized in IDSs for IoT is shown in Figure 2.1. The taxonomy includes the various areas that are crucial to understand IoT security issues and their solutions. The taxonomy includes (1) IoT cyberattacks, (2) IoT architecture layers [8], (3) IDSs for IoT [19], (3) DL techniques used in the IoT IDSs [19], (4) common datasets used for evaluating DL systems, and (5) classification strategies [19]. The different areas included in the taxonomy are in various ways interconnected as root causes of IoT security vulnerabilities in IoT and solutions to counter such causes.

2.1.2 Security issues at Sensing Layer

The sensing layer primarily deals with IoT sensors and actuators. Sensors sense the physical phenomenon happening around them. On the other hand, actuators perform a specific action on the physical environment based on the sensed data.

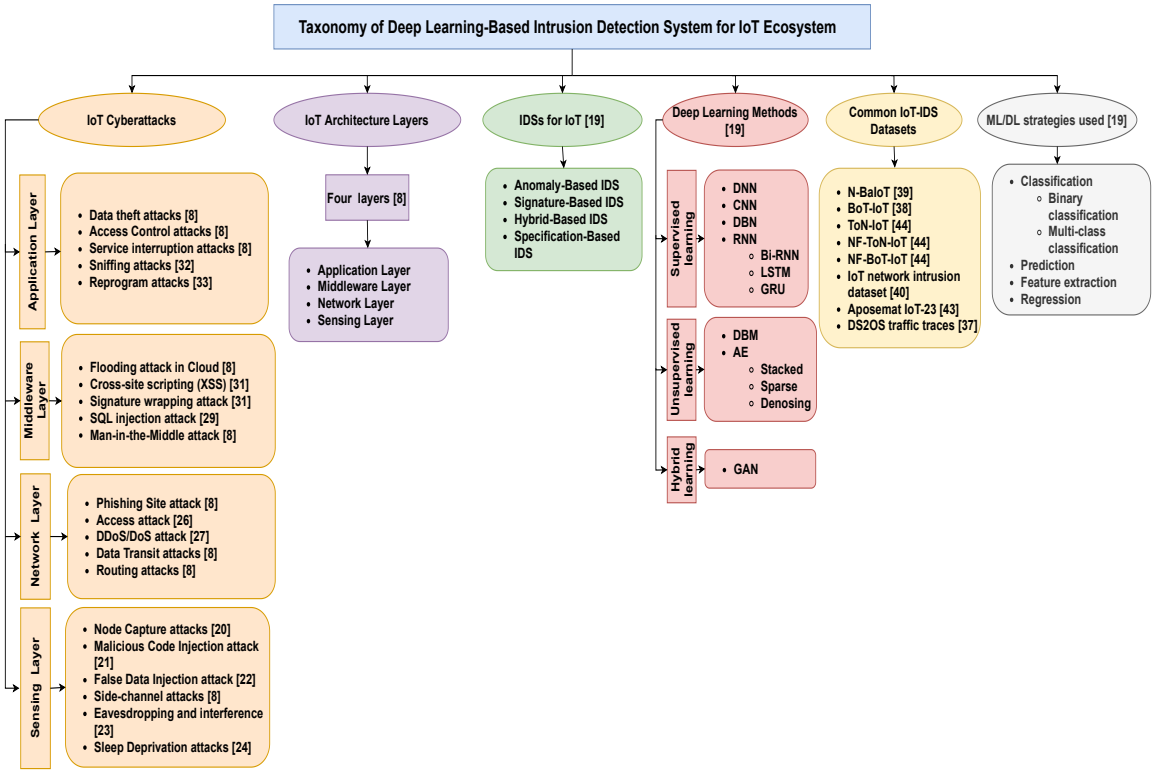


Figure 2.1: Taxonomy of deep learning-based detection system against cyberattacks for IoT.

There are various kinds of sensors for sensing different kinds of data, such as ultrasonic sensors, camera sensors, smoke detection sensors, temperature, and humidity sensors. Various sensing layer technologies are used in the different IoT applications such as RFID, GPS, WSNs, and RSNs [8]. Main security threats that can be encountered at the sensing layer are as follows.

- **Node capturing:** IoT applications comprise several low power nodes such as sensors and actuators. These nodes are vulnerable to a variety of attacks by adversaries. The attackers may try to capture or substitute the node in the IoT system with a malicious node. The new node may appear to be part of the system but is controlled by the attacker. This may lead to compromising the security of the complete IoT application [20].

- **Malicious code injection attack:** The attack involves the attacker in-

jecting some malicious code into the memory of the node. In general, the firmware or software of IoT nodes are upgraded on the air, and this provides a gateway to the attackers to inject malicious code. Utilizing such malicious code, the attackers may force the nodes to perform some unintended functions or may even try to access the complete IoT system. [21].

- **False data injection attack:** Once the node is captured, the attacker may apply it to inject false data onto the IoT system. This may lead to false results and may result in malfunctioning of the IoT application [22].
- **Side-channel attacks (SCA):** Besides direct attacks on the nodes, various side-channel attacks may lead to leaking of sensitive data. The microarchitectures of processors, electromagnetic emanation, and their power consumption reveal sensitive information to adversaries. Side-channel attacks may be based on power consumption, laser-based attacks, timing attacks, or electromagnetic attacks. Modern chips take care of various countermeasures to prevent this side-channel attacks while implementing the cryptographic modules. [8]
- **Eavesdropping:** IoT applications usually comprise various nodes deployed in open environments [23]. Thus, such IoT applications are exposed to eavesdroppers. The attackers may eavesdrop and capture the data during different phases such as data transmission or authentication.
- **Sleep deprivation attacks:** During these types of attacks, the adversaries attempt to deplete the battery of the low-powered IoT edge devices. This problem leads to a denial of service from the nodes in the IoT application due to a dead battery. This attack can be carried out by running infinite loops in the edge devices applying malicious code or by artificially increasing the power consumption of the edge devices [24].

2.1.3 Security issues at Network Layer

The network layer's primary function is to send information from the sensing layer to the computational unit for processing. The main security problems encountered at the network layer are as follows.

- **Phishing site attack:** Phishing is one of the top threats that cause data breaches and it is a technique in which the attacker tries to steal a user's credentials through fraudulent attempts, for instance, by sending emails [25]. The attackers expect that at least a few of the devices will become a victim of the attack. There is a probability of encountering phishing sites in the course of users visiting web pages on the Internet. Once the user's account and password are compromised, the whole IoT ecosystem being managed by the user becomes vulnerable to cyberattacks. The network layer in IoT is highly vulnerable to phishing sites attacks [8].

- **Access attack:** The advanced persistent threat is another term for an access attack. An unauthorized adversary gains access to the IoT network in this form of attack. The intruder will remain undetected in the network for an extended period. Rather than causing network harm, the ultimate goal of such a type of attack is to steal valuable information. IoT applications receive and transmit valuable data regularly, making them particularly vulnerable to such attacks [26].

- **DDoS/DoS attack:** The attacker floods the target servers with a large number of unwanted requests. These requests hinder the target server, thereby disrupting services to legitimate users. If there are many sources utilized by the attacker to flood the target server, then such an attack is termed a DDoS. Many IoT devices in IoT applications are not strongly configured and thus easily compromised by attackers. The compromised IoT devices, bots, or zombies, are controlled by a command and control (C&C) server to perform various tasks. A bot-master, an attacker controls the bots, can use thousands or possibly millions of infected bots in its botnet to concurrently launch large-scale attacks like spam emails for monetary gains or DDoS on critical infrastructure or websites to make it unresponsive. For instance, one of the well-known large-scale IoT botnet attacks, Mirai, happened in 2016, causing high-profile websites such as Twitter, New York Times, GitHub, and Netflix to become inaccessible [27].

- **Data transit attacks:** IoT applications handle a lot of data storage and exchange. Data is valuable, and thus it is always the target of hackers.

Data that is stored in the local servers or the cloud has a security risk, but the data that is in transit or is moving from one location to another is even more vulnerable to cyber attacks. In IoT applications, there is a lot of data movement between sensors, actuators, cloud, etc. Various connection technologies are employed in such data movements, and hence IoT applications are susceptible to data breaches [8].

- **Routing attacks:** During data transit, malicious nodes in an IoT application may seek to redirect the routing. Sinkhole attacks are a particular kind of a routing attack in which an intruder compromises a node inside the network and launches an attack. Then the compromised node attempts to attract all the traffic from neighbor nodes based on the routing metric that is used in routing protocol into it [28]. A worm-hole attack is another attack which can become a serious security threat if combined with other attacks such as sinkhole attacks. A warm-hole is an out of band connection between two nodes for fast packet transfer. An attacker can generate a warm-hole between a compromised node and a device on the Internet and attempt to bypass the basic security protocols in an IoT application [8].

2.1.4 Security issues at Middleware Layer

The purpose of the middleware layer in IoT is to generate an abstraction layer between the network layer and the application layer. This layer comprises information processing systems that take automated actions based on the results of processed data and links the system with the database which provides storage capabilities to the collected data [21]. Even though the middleware layer is beneficial to provide a reliable and robust IoT application, it is also vulnerable to various attacks. These attacks can take control of the whole IoT application by infecting the middleware. Moreover, database security and cloud security are other main security challenges in the middleware layer. Several possible attacks in the middleware layer are explained as follows [8].

- **Man-in-the-middle attack:** Using the MQTT broker, which basically functions as a proxy, the MQTT protocol utilizes a publish-subscribe paradigm

of communication between clients and subscribers. This helps in disconnecting the publishing and the subscribing clients from each other and messages can be sent without the knowledge of the destination. If the attacker can control the broker and become a man-in-the-middle, then the attacker can get complete control of all communication without any knowledge of the clients [8].

- **SQL injection attack:** Middleware layer is vulnerable to SQL Injection (SQLi) attacks, which allows attackers to bypass authentication, access private information, modify data, or even destroy databases [29].
- **Flooding attack in cloud:** This attack operates almost the same as DoS attack in the cloud and affects the QoS. For exhausting cloud resources, the attackers continuously send multiple requests to a service. These attacks can have a significant impact on cloud systems by increasing the load on the cloud servers [8].
- **Signature wrapping attack:** In the web services used in the middleware, XML signatures are used [30]. XML signature wrapping attack works by intercepting and modifying XML message and re-transmitting it to a target machine in order to run tainted code [31].
- **Cross-site scripting (XSS):** In such an attack, the adversary injects malignant HTML/ JavaScript codes into the data content which may be accessed and executed by a server leading to its compromise [31].

2.1.5 Security issues at Application Layer

The application layer is mainly responsible for providing services to the end-users related to particular applications. There is a wide range of IoT applications such as smart homes, smart agriculture, smart cities, and smart healthcare. Main security issues encountered by the application layer are explained below.

- **Data thefts:** IoT applications deal with a lot of critical and private data. The data in transit is even more vulnerable to attacks than data at rest, and in IoT applications, there is a lot of data movement. The users will be

unwilling to register their private data on IoT applications if these applications are vulnerable to data theft attacks. Some of the techniques and protocols such as data encryption, data isolation, user and network authentication, privacy management are being utilized to secure IoT applications against data thefts.

- **Access control attacks:** Access control is an authorization mechanism that allows only legitimate users or processes to access the data or account. Access control attack is a severe attack in IoT applications because once the access is compromised, then the whole IoT application becomes vulnerable to attacks.

- **Service interruption attacks:** These attacks are also referred to as DDoS attacks in the existing literature. There have been various cases of such attacks on IoT applications. Such attacks are to generate a large number of requests to IoT applications by mimicking legitimate users to make target IoT applications services unavailable for legitimate users.

- **Sniffing attacks:** The attackers may utilize the sniffer applications to monitor the network traffic in IoT applications. Thus, If IoT applications do not implement the data confidential security protocols to prevent it, this may enable the attacker to obtain access to confidential user data [32].

- **Reprogram attacks:** If the programming process is not secured, then the attackers can seek to reprogram the IoT objects remotely. This may lead to the hijacking of the IoT network [33].

2.2 Managing networks - SDN

The utilization of SDN is getting high momentum also within the security research communities. SDN departs the control plane of networking devices such as router and switch from its data plane, making it possible to control, monitor, and manage a network from a centralized controller [34]. By leveraging SDN, we can provide several attractive benefits for IoT ecosystem security through controlling the flow dynamically and mitigating attacks at an early stage by rate-limiting

the flow at the SDN switches. Several significant works [35] have been proposed for anomaly detection in IoT applying SDN. Bhunia and Gurusamy [35] present a machine learning-based attack detection and mitigation method in IoT traffic using SDN. It uses the SVM algorithm at the SDN controller to monitor and learn the behavior of IoT devices over time, and to detect attacks. However, they evaluated only with a simulation environment known as Mininet, it remains to be seen if the method works in real-time environments.

2.3 Data collection, testbed setup, and benchmarking

In network intrusion detection, especially when using anomaly-based detection, it is challenging to accurately evaluate, compare, and deploy a system that is expected to detect new attacks due to the scarcity of adequate datasets. An anomaly-based network intrusion detection system must be examined and evaluated utilising the real labelled network traffic traces with a comprehensive set of cyberattacks before deploying it in any real-world environment [36]. Since not many datasets are available in IoT, it is becoming a significant challenge for developing new cyberattacks detection methods especially emerging attacks in IoT and evaluating them.

2.3.1 Existing datasets

As mentioned prior, datasets execute an essential role in the evaluation of IoT cyberattacks detection approaches or systems. There are several datasets publicly available for evaluation of IoT cyberattacks detection approaches and systems including DS2OS traffic traces [37], BoT-IoT [38], N-BaIoT [39], IoT Network Intrusion Dataset [40], ToN-IoT, NF-BoT-IoT, and NF-BoT-IoT. A taxonomy of IoT cyberattacks datasets is illustrated in Figure 2.2. We briefly explain each of them as follows.

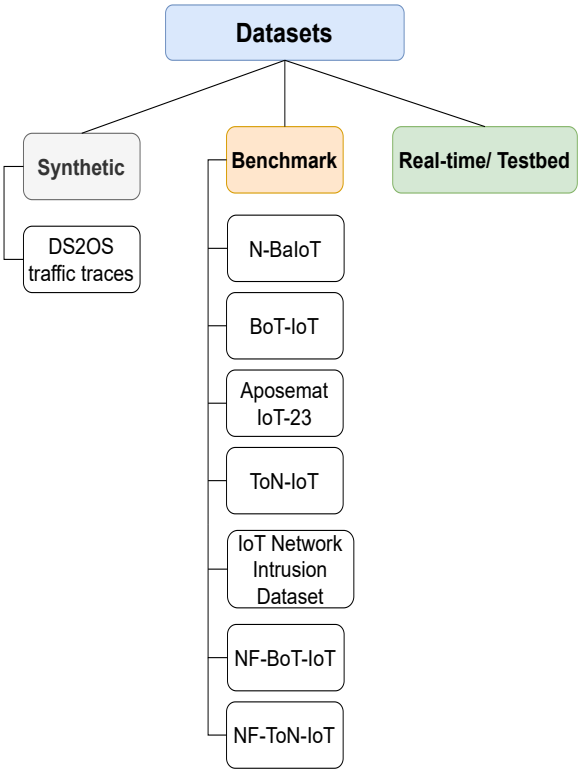


Figure 2.2: A taxonomy of IoT cyberattacks datasets

Synthetic datasets

Synthetic datasets are created to provide specific requirements or certain conditions or tests that real data satisfy. Such datasets are beneficial when devising any prototype system for theoretical analysis thus the design can be improved. As affirmed before, a synthetic dataset can be utilized to test and generate numerous different types of test scenarios. This allows researchers to create realistic behavior profiles for legitimate users and attackers based on the dataset to evaluate a proposed system. Additionally, these datasets give fundamental validation of a specific method or a system; if the results confirm to be satisfactory, the researchers then continue to evaluate a method or a system in a specific domain [41].

DS2OS traffic traces was presented in [37] and is publicly available on Kaggle [42]. It comprises synthetic data generated in a virtual IoT environment by means of a Distributed Smart Space Orchestration System (DS2OS). The system architecture of the IoT environment includes a set of microservices that commu-

nicate with each other using the Message Queuing Telemetry Transport (MQTT) protocol. The dataset comprises 13 different features gathered by monitoring connections between 7 different Virtual State Layer (VSL) service types that connect illumination controllers, movement sensors, thermostats, solar batteries, washing machines, door locks, and smartphones. It involves information about the microservice source, destination, operation, and so forth, however, this dataset does not include any NetFlow or packet data. It only comprises features that were particularly designed for detecting anomalies in the IoT traffic frequencies and the communication baseline models of the microservices [37].

Benchmark datasets

Benchmark datasets are crucial for method developers as well as for those who want to obtain the best performing tools. There are seven IoT benchmark datasets publicly available with different scenarios of attacks and legitimate instances created utilizing IoT devices and network environments. Thus, we briefly discuss each of them as follows.

BoT-IoT dataset was devised to enable botnet identification in IoT networks. An IoT network was imitated utilizing virtual machines (both normal and attacking). Kali VMs executed port scanning, DDoS, and other botnet attacks. On Ubuntu VMs, the Node-red tool was utilized for mimicking various IoT sensors including a weather station, a smart fridge, motion-activated lights, a garage door, and a thermostat. A total of 46 network features were extracted applying the Argus tool. No sensor or log information was recorded. Since the Bot-IoT dataset collects data using simulated IoT devices, no real IoT hardware is monitored, making the dataset less representative of real IoT traffic [38].

IoT network intrusion dataset presented in 2019, was generated utilizing two typical smart home devices, a SKT NUGU (NU 100) and EZVIZ Wi-Fi Camera (C2C Mini O Plus 1080P), a few laptops and a few smartphones. These devices were all connected to the same wireless network on which different attacks were simulated using tools such as Nmap. The dataset comprises 42 raw network packet files (pcap) collected at different points in time [40].

N-BaIoT dataset was prepared by using two attack generation tools, i.e., Mirai (scan, ACK flooding, SYN flooding, UDP flooding, UDPplain attacks) and

Bashlite (scan, junk, UDP flooding, TCP flooding, COMBO attacks), with 9 commercial IoT devices. There are 5 Bashlite, 5 Mirai, and 1 legitimate datasets, having 115 features in each of them [39].

Aposemat IoT-23 is a dataset of network traffic from IoT devices including a Philips HUE smart LED lamp, an Amazon Echo home intelligent personal assistant and a Somfy Smart Door Lock. Data was collected within the Stratosphere project and comprises 20 captures of malware executed in IoT devices, and 3 captures of benign IoT devices traffic. The executed attacks involve different botnets such as Mirai and Torii. Moreover, the original packet capture files, netflows generated by Zeek/Bro IDS and their labels are also available [43].

ToN-IoT A recent heterogeneous dataset published in 2020 [44] that involves telemetry data of IoT services, network traffic of IoT networks and operating system logs. The dataset consists of a large number of attack scenarios conducted in a realistic representation of a medium-scale network at the Cyber Range Lab by ACCS. Additionally, Zeek was utilized to extract the dataset's original 44 features.

NF-BoT-IoT is an IoT NetFlow-based dataset created employing the BoT-IoT dataset, named NF-BoT-IoT. The features were extracted from the publicly available pcap files and the flows were labelled with their respective attack categories. There are four attack categories in the dataset.

NF-TON-IoT was used publicly available pcaps of the TON-IoT [44] dataset to generate NetFlow records, leading to a NetFlow-based IoT network dataset. In addition, TON-IoT was presented in 2020 [44] that comprises telemetry data of IoT services, network traffic of IoT networks, and operating system logs.

Real-time/Testbed datasets

The real-time/Testbed datasets are generated by deploying several IoT devices and applications. It injects attacks and legitimate traffic and services, respectively. Later, the monitoring agents will collect data for both system Key Performance Indicators (KPIs) and traffic statistics for both IoT devices and deployed applications. However, the community lacks real-time/testbed datasets by considering or creating real environments with hundreds of IoT devices and applications with multiple services for validating developed solutions.

2.4 Exploratory Data Analysis

Exploratory data analysis (EDA) is an imperative step in any research analysis. The main aim with exploratory analysis is to analyze the data for distribution, outliers and anomalies to direct specific testing of your hypothesis. It also gives tools for hypothesis generation by visualizing and understanding the data usually through graphical representation [45].

2.5 Evaluation criteria

In this section, we describe the performance measures commonly used to evaluate network intrusion detection methods and systems. The evaluation of classifiers' performances plays an important role in development and selection of the classification model. We explain the multiple measures to complement each other and ensure efficacy measures of the proposed models. These measures include confusion matrix, precision, recall, F1-measure, Matthews Correlation Coefficient (MCC). Using these performance metrics, one can decide which model is performed well and best suited for the proposed framework when addressing multiple problems in IoT.
Confusion matrix (CM) is used to describe the performance of a classification model on a set of test data. The diagonal represents the correct classification. The confusion matrix for attack detection is defined as a 2-by-2 matrix, since there are only two classes known as anomalous and legitimate. Thus, the TNs (True Negative) and TPs (True Positive) that represent the correctly predicted cases lie on the matrix diagonal while the FNs (False Negative) and FPs (False Positive) are on the right and left sides, having wrong predictions [46]. We employ accuracy, precision, recall and F_1 measures as evaluation metrics in addition to MCC.

- True Positive (TP): The number of an actual legitimate class is correctly predicted as a legitimate class.
- False Negative (FN): The number of an actual anomaly class is incorrectly predicted as a legitimate class.

Table 2.1: Confusion matrix

		Actual class	
		Legitimate	Anomalous
Predicted class	Legitimate	TP	FN
	Anomalous	FP	TN

- False Positive (FP): The number of an actual legitimate class is incorrectly predicted as an anomaly class.
- True Negative (TN): The number of an actual anomaly class is correctly predicted as an anomaly class.

True Negative Rate (TNR) is the percentage of the anomaly traffic misclassified as legitimate.

$$TNR = \frac{TN}{FP + TN} \tag{2.1}$$

False Negative Rate(FNR) is the percentage of legitimate traffic misclassified as an anomaly.

$$FNR = \frac{FN}{TP + FN} \tag{2.2}$$

Accuracy is a metric that measures how correctly an attack detection system works, measuring the percentage of detection and failure as well as the number of false alarms that the system produces [46].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision means the positive predictive value. It is a measure of the number of true positives the model claims compared to the number of positives it claims.

$$Precision = \frac{TP}{TP + FP}$$

Recall is known as the actual positive rate which means the number of positives in the model claims compared to the actual number of positives there are throughout the data [47].

$$Recall = \frac{TP}{TP + FN}$$

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Matthews Correlation Coefficient (MCC) measures the quality of the classification, showing the correlation agreement between the observed values and the predicted values [48]. MCC is a great metric, especially in case of imbalanced datasets, it is given in Eq. 2.7.

$$N = TN + TP + FN + FP \tag{2.3}$$

$$S = \frac{TP + FN}{N} \tag{2.4}$$

$$P = \frac{TP + FP}{N} \tag{2.5}$$

$$MCC = \frac{\frac{TP}{N} - S \times P}{\sqrt{P \times S(1-S)(1-P)}} \tag{2.6}$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}} \tag{2.7}$$

2.6 Summary

This chapter gives the preliminaries of numerous kinds of critical cyberattacks in IoT. This chapter also discusses how to manage IoT devices' network traffic flow utilizing the SDN, and how important the dataset is for devising and evaluating the new cyberattacks detection and prevention approaches in IoT cyberattacks. Lastly, we present common evaluation criteria to measure the performance of network intrusion detection methods or systems.

3 Related Work

This chapter presents related works on IoT cyberattacks detection and prediction methods utilizing the machine and deep learning approaches.

3.1 Introduction

IoT Intrusion is defined as an unauthorized action or activity that harms the IoT ecosystem. In other words, an attack that results in any kind of damage to the confidentiality, integrity, or availability of information is considered an intrusion [49]. Limited computational and storage resources in IoT systems establish constraints on installing conventional security software. The two main intrusion detection techniques utilized to detect cyberattacks in IoT systems are signature-based (a.k.a. misuse-based or knowledge-based) and anomaly-based (a.k.a. behavior-based) approaches [1]. Occasionally they are combined to generate a hybrid detection system, however, it is complex to implement [50]. Signature-based techniques rely on the existing threat knowledge to classify traffic as legitimate or malicious, whereas anomaly-based systems depend on traffic patterns to detect cyberattacks [51]. Signature-based systems work excellent on existing and well-known cyberattacks. One of the challenges with the signature-based system is that constantly updating the signature database is time-consuming. As the size of the database increases, comparing the input with a large database is computational costly. This method is based on previously known attack signatures thus it cannot detect zero-day or unknown attacks [52]. Anomaly-based detection is preferred because it observes abnormal traffic patterns, generates alarms, or blocks traffic when abnormal traffic patterns are detected. One of the advantages of utilizing anomaly-based systems is that they are good at detecting zero-day and unknown attacks, however, they

may produce high false-positive results [53]. Thus, in recent years, researchers have lately utilized machine and deep learning approaches to address the high false-positive results of anomaly-based intrusion detection systems in the IoT ecosystem. There have been broad research studies performed utilizing various machine learning and deep learning models individually and as an ensemble of classifiers to detect cyberattacks in IoT ecosystem. Machine learning (ML) is applied to uncover beneficial patterns in big data. ML employs statistical methods and algorithms to learn complex and difficult patterns from the data. Over time and continuous data analysis helps ML algorithms to give improved decisions or predictive accuracy [53]. DL is an ML branch that has given promising results to detect cyberattacks in the IoT ecosystem. DL models are devised to learn the way the human brain learns. They utilize many hidden layers to form an artificial neural network (ANN). One of the advantages of DL over ML is its capability to process big data volume. Since the data size increases, the performance plateau of the ML models reaches a certain data size threshold limit, while the DL models continue to perform better and better [18]. Since IoT devices generate huge amounts of data, this makes DL approaches fit for IoT cyberattacks detection systems.

3.2 Prior surveys on IoT attack detection and prediction

Advances and growth in IoT have brought many researchers to make detailed surveys on IoT security. Thus, many types of survey and review researches [54–58] have been done in the field of intrusion detection in IoT. Mohamed et al. [54] give a comprehensive study of DL, big data, and IoT security. The study's main aim is to provide DL capabilities and big data technologies that can be combined to protect IoT devices from cyberattacks. Al-Garadi et al [55]. provide a comprehensive survey of ML and DL methods that can be used to improve security in IoT ecosystems. A detailed survey of threats analysis, including confidentiality, integrity, and availability (CIA) is presented. Authors give a survey on IoT IDS, their detection strategy, placement strategy, security threats, and the validation strategy. Each of these strategies is presented in detail [56]. Xiao et al [57] present

a detailed survey on ML-based methods to protect IoT devices, data protection, access control for privacy protection, malware detection, and ML implementation challenges. Somayye et al. [58] give a detailed survey on IDS to protect IoT systems, their advantages, disadvantages, open issues, and future trends.

3.2.1 Classical machine learning approaches

In this subsection, we discuss the common ML models including decision trees (DT), support vector machines (SVM), Bayesian algorithms, k-nearest neighbour (KNN), random forest (RF), association rule (AR) algorithms, ensemble learning, k-means clustering and principal component analysis (PCA)for the intrusion detection system in IoT ecosystem.

Decision Trees (DTs)

DT-based method utilizes a DT to create a model to learn from training samples by representing them as branches and leaves. The pretrained model is then utilized to predict the class of the new sample [59]. DT methods require large storage because of the nature of construction. Alharbi et al. [60] propose the utilization of a fog-based system to secure IoT devices. In this study, DT was utilized to analyse network traffic to detect DDoS attacks.

Support Vector Machines (SVMs)

SVM is one of the powerful classification ML techniques that give good accuracy with less computational power. SVM algorithm creates a hyperplane or decision boundaries of the number of features and then applies classification techniques to differentiate multiple classes [1]. Bhunia et al. [35] present a machine learning-based attack detection and mitigation method in IoT traffic using SDN. It uses the SVM algorithm at the SDN controller to monitor and learn the behavior of IoT devices over time, and to detect attacks. However, they evaluated only with a simulation environment known as Mininet, it remains to be seen if the method works in real-time environments.

Random Forest (RF)

RF is another popular ML algorithm based on ensemble methods, bootstrapping, and bagging techniques. In this method, several individuals and unique decision trees are trained in parallel, and their individual results are aggregated (called bagging) to find the best results [1]. Researchers have employed RF in multiple research fields, including finding anomalies and malicious network traffic in IoT networks. Mahmudul et al. [47] propose an anomaly detection method for IoT sensors in IoT sites using machine learning approaches. This method is evaluated by using an open-source dataset [61] with seven classes of attacks. They achieve 98.2%−99.4% accuracy for random forest and artificial neural networks. Chaudhary et al. [62] propose an approach to detect DDoS attacks on IoT devices utilizing ML models including Random Forest (RF), SVM, Logistic Regression (LR), and Decision Tree (DR). RF model achieves 99.17% in terms of accuracy. However, this study was evaluated the proposed approach utilizing the only testbed dataset. Chaudhary et al. [62] propose an approach to detect DDoS attacks on IoT devices employing multiple ML algorithms including SVMs, RF, DT, and logistic regression algorithms (LRA). In this study, RF outperforms other ML methods with 99.17% accuracy to detect DDoS attacks.

3.2.2 Deep learning approaches

In recent years, the applications of DL to IoT systems have become an essential research topic [63]. The most essential merit of DL over conventional ML is its superior performance in large datasets. Several significant works [38, 38, 39, 48, 64–66] have been proposed for anomaly detection in IoT using deep learning approaches.

Convolutional Neural Networks (CNN)

CNN inspired by the visual cortex of animals, is widely used for object recognition tasks [65]. As a deep learning architecture, CNN is proposed to minimize the data preprocessing requirements. The most powerful part of CNN is the learning feature hierarchies from a large amount of unlabeled data. Hence, CNN are quite promising for applications in network attack detection [67]. Jinyin et al. [65]

devise a multichannel CNN (MC-CNN) deep learning approach to detect DDoS attacks and it was evaluated employing KDDCUP99 and CICIDS2017 datasets. The experimental results illustrate that MC-CNN detects DDoS attacks with 99.18% accuracy. Recently, Imtiaz et al. [66] develop the intrusion detection system for IoT networks based on CNN and transfer learning frameworks, and it was evaluated by utilizing the BoT-IoT [38], IoT Network Intrusion, MQTT-IoT-IDS2020 [68], and IoT-23 intrusion detection [43] datasets. The experiment result demonstrates that it achieves 99.99% in terms of detection accuracy. Bambang et al. [64] present an intrusion detection method for the IoT environment using the machine and deep learning approaches with the BoT-IoT dataset [38]. Their results demonstrate that Random forests (RF) and CNN give the best result in terms of accuracy and the AUC for multiclass classification.

Recurrent neural network (RNN)

RNN is another powerful deep neural network (DNN) commonly utilized to process and find patterns such as sequential data, time series, and natural language [1]. IoT devices produce many sequential data from sources such as network traffic flows [69]. Fangyu et al. [70] have analyzed system statistics by utilizing time series methods and applied different ML models to detect anomalous behavior. RNN performs better than an ML model linear regression as well as RNN has the least mean absolute error (MAE). One of the limitations of RNN is the issue of vanishing or exploding gradients [55]. Thus, RNN's are unable to capture long-term dependencies. The timestamp's computational dependency on the previous timestamp limits parallelism abilities [71].

Long Short-Term Memory (LSTM):

One of the specialized RNN is called LSTM, which can remember information for a long period of time [72]. LSTM evolved from RNN and can detect normal and malicious traffic by learning patterns using long sequences [73]. As opposed to RNN, LSTM is widely utilized, solves the vanishing gradient problem, and can learn long-term dependencies [74]. The ability of LSTM to learn from temporal sequences and long-term dependencies gives us an effective IDS opportunity than CNN and RNN. However, LSTM takes a significantly long training time be-

cause feature computation at different timestamps cannot happen in parallel [75]. Hwang et al. [76] utilized LSTM to detect real-time malicious IoT network traffic by simply analyzing the packet header information. This proposed approach was evaluated by using the ISCX2012 [77], USTC-TFC2016 [78], and testbed datasets, and it gives 99.36-99.98% in terms of accuracy. A limitation of this study, the results of this study have not been compared with the results of other baseline models and related works. Liang et al. [79] propose an LSTM-based approach to analyze only the packet header information to detect large-scale attacks. Their proposed method performs better than the conventional ML and simple ANN model. [80] propose a hybrid CNN-LSTM model and evaluate it using the CICIDS2017 dataset [81] for detecting DDoS attacks and, resulting in an accuracy of 97.16%.

Autoencoders (AE)

AE is one of the powerful unsupervised neural network techniques [82]. AE is composed of three layers including the input layer, the hidden encoding layer, and the decoded output layer. Input and output layers are comprised of the same dimension. The dimensionality and size of input data are reduced after it is passed to an AE, encoding, and compressing the input. The output layer takes the compressed data as input and converts it back into the same dimensions as the original input data was. AE's learn and classify output automatically without the need for a labeled dataset [83]. AE's have been utilized in IDS to detect anomalies in network traffic. For instance, Yair et al. [39] come up with a network-based anomaly detection method that uses deep autoencoders and to detect compromised traffic of IoT devices. This method is validated by using nine different commercial IoT devices in a laboratory environment with two famous IoT-based botnets, i.e., Mirai and BASHLITE. This method lacks variation in attack scenarios. In addition, researchers have been integrating AE's with many other ML and DL algorithms such as probabilistic neural network (PNN) and RNN, LSTM, CNN to achieve better classification results. For instance, Debastrita et al. [84] report an ensemble learning method for outlier detection to handle problems in imbalanced datasets. This method employs stacked autoencoder (SAE) to extract deep features and input them to an ensemble of probabilistic

neural network models for single and multiple outliers detection. The use of an autoencoder makes the method's performance better and more stable. Hwang et al. [85] propose an unsupervised IDS based on CNN and AE to detect malicious network traffic. Their approach utilizes CNN to auto-learn the traffic features by looking at the first few raw traffic bytes. AE's were employed to build the benign traffic profile in an unsupervised manner. Their model detected malicious traffic with nearly 100% accuracy and less than 1% FNR and FPR. Their solution shows a gap in performing only stateful traffic inspection and misses stateless traffic analysis.

Deep Ensemble Learning

Ensemble learning uses multiple algorithms to get better predictive performance than any single one of its constituent algorithms could [80, 84, 86]. There are several evidences that prove that ensemble learning model can handle data heterogeneity or class imbalance problems in anomaly detection [87]. Timcenko and Gajin [88] present an anomaly detection method for the IoT environment using single and ensemble learning approaches and achieved 99% accuracy using the UNSW-NB15 dataset. However, the method lacks testbed experiments and dynamic attack scenarios. Shanzeb et al. [86] propose a deep ensemble CNN framework to detect DDoS attacks in SDNs, and they evaluate it using the flow-based CICIDS2017 dataset. The proposed approach achieves 99.45% in terms of detection accuracy.

3.3 Observations and summary

Most studies do not focus on a comprehensive approach to detect and prevent cyberattacks in the IoT ecosystem. The data imbalance problem can decrease the ML and DL models performance. We observe in previous studies that DL models, especially the DL method, perform better than ML methods in detecting cyberattacks in the IoT ecosystem by addressing data imbalance and heterogeneity problems in the IoT ecosystem. However, there are very few studies that address the issue of data imbalance for detecting cyberattacks in the IoT ecosystem. A small number of prior studies have mentioned that how the proposed ML and DL

methods deploy on IoT devices with the constraint resource. Thus, still, several opportunities are left to address IoT security problems in small and large-scale industries with a comprehensive approach in either centralized or distributed environments.

4 Deep Ensemble Learning-Based Approach for Detecting Anomalies in IoT Devices

This chapter presents a comprehensive approach to detect and predict cyberattacks in IoT devices by leveraging DL and SDN.

4.1 Introduction

The rapid evolution of the Internet of Things (IoT) has brought billions of internet-enabled devices into our daily life to make it smarter by bridging the gap between the physical and the virtual world. Frost and Sullivan [89] have predicted that the number of connected devices will increase up to 45.41 billion by 2023. Autonomous decision making, information to end-users, machine-to-machine and machine-to-user interaction have boosted the acceptance of IoT as a critical asset in the service chain. IoT systems open up several opportunities in areas of autonomous transportation and industrial automation [90]. Manufacturers hastily produce new IoT devices without basic security and privacy checks; thus, allowing attackers to easily and swiftly identify vulnerabilities that allow them to evade, manipulate, and take over IoT networks [91]. The failure to implement proper security and privacy measures have already resulted in dire consequences for certain IoT manufacturers and service providers in terms of reputation and financial penalties [92]. Hence, security is becoming crucial to protect IoT devices and applications from cyberattacks in both small and large-scale networks comprised

of physical and virtual infrastructures.

The rapid growth of IoT-related vulnerabilities, which is proportional to the exponential production of IoT devices and applications, has created new categories of attacks and malware. For example, Mirai Botnet caused massive Distributed Denial of Service (DDoS) attacks to a DNS server of Dyn DNS provider [6] and brought down several sites including GitHub*, Reddit†, Netflix‡, and Airbnb§. Multiple zero-day attacks have been observed in complex networks [7]. It is imperative to devise efficient methods for detecting attacks in IoT networks. Traditionally, there are two primary categories of attack detection methods: signature-based and anomaly-based. The signature-based method needs to frequently update the attack signature for known attacks from released signatures of Intrusion Detection System (IDS) vendors [6]. The anomaly-based detection method employs legitimate behavior patterns to detect unknown attacks based on deviation from them [93]. These detection problems intensify in the context of IoT in terms of the following points. *First*, IoT devices do not generate unusual traffic due to limited communication, such as status update and sensor data reading. *Second*, developing a dynamic model that considers device heterogeneity opens up another challenge. *Third*, IoT devices have resource limitations that complicate the deployment of a detection system at the device level. However, the class imbalance problem is one of the main bottlenecks of attack detection algorithms because attack classes are rarer than legitimate classes [94]. For instance, in most of the cases, the number of samples from the anomaly (outlier) class is below 10% of the total number of samples in the entire training set [84]. This data imbalance problem introduces a bias in the machine learning models that degrades the performance substantially.

SDN has appealing features such as flexibility, dynamicity in network operations and resource management; nevertheless, it has limited scalability [95]. By leveraging SDN, we can provide several attractive benefits for IoT security through controlling the flow dynamically and mitigating attacks at an early stage by rate-limiting the flow at the SDN switches. Hence, developing a deep ensem-

*https://github.com/
†https://www.reddit.com/
‡https://www.netflix.com/
§https://www.airbnb.com/

ble learning model as a detection module and deploying it in the SDN framework provide: (i) an isolation of compromised devices, (ii) early detection and mitigation, (iii) a reduction of resource wastage, (iv) an improvement of the accuracy by deploying sophisticated algorithms. Mostly, detection models suffer from class imbalance problems that incur significant performance loss for detecting attacks in IoT. This work takes the benefits of the SDN controller at switches by grabbing the features of data back-and-forth into IoT devices.

Ensemble learning uses multiple algorithms to get better predictive performance than any single one of its constituent algorithms could [96–98]. There are several evidences that prove that ensemble learning model can handle data heterogeneity or class imbalance problems in anomaly detection [87]. Thus, we are motivated to develop a deep ensemble learning model by utilizing deep feature extraction with deep or stacked autoencoder, and feed them to ensemble of Probabilistic Neural Networks (PNNs) for detecting anomalies in IoT. To address the above challenges, we leverage our recent work [99] to present DeL-IoT, an SDN-enabled deep ensemble learning framework for IoT anomaly detection, dynamic flow management, and prediction that utilizes the appealing features of deep autoencoders, probabilistic neural networks and long short-term memory (LSTM). Our proposal uses both device and network switching level features to build an efficient detection and prediction system, which will ensure the increase of IoT device uptime, performance, and forecast device status. We make the following contributions.

4.2 Prior Research

Several significant works [35, 39, 47, 88, 100, 101] have been proposed for anomaly detection in IoT using machine learning approaches with and without SDN. However, most of them were focused to protect either device level or network level, and they did not address the problems together with data imbalance problem, flow management, data heterogeneity, and device status prediction. Bhunia and Gurusamy [35] present a machine learning-based attack detection and mitigation method in IoT traffic using SDN. It uses the SVM algorithm at the SDN controller to monitor and learn the behavior of IoT devices over time, and to detect

attacks. However, they evaluated only with a simulation environment known as Mininet, it remains to be seen if the method works in real-time environments.

Table 4.1: Comparison of existing IoT anomaly detection methods

Author and Year	**Detection method**	**Botnets**	**Dynamic attack detection**	**Deployment level**	**Datasets**	**Performance**	**Class imbalance problem**
Bhunia et al. [35], 2017	SVM	unknown	Yes	SDN controller	Mininet emulator	Precision-98% Recall-94%	No
Timcenko et al. [88], 2018	LADTree, Random Forest, REPTree, MultiBoost, SMO	unknown	No	Networks	UNSW-NB15	ROC-55%-99%	No
Maede et al. [48], 2018	ANN with SMOTE	unknown	No	Networks	Testbed IIoT control system	MCC-19%-99.86%	Yes but partially
Yair et al. [39], 2018	Autoencoder and other machine learning algorithms	Mirai, Bashlite	Yes	Network	N-BaIoT	TPR-75%-100%	No
Nickolas et al. [38], 2019	LSTM, RNN, SVM	unknown	No	Networks	Bot-IoT	Accuracy-88%-99%	No
Thien et al. [102], 2019	A federated self-learning Gated Recurrent Units (GRUs)	Mirai	No	Networks	Testbed IoT devices DĬoT	TPR-95.6%	No
Mahmudul et al. [47], 2019	machine learning algorithms	unknown	No	Networks	DS2OS traffic traces	Accuracy-98.2%-99.4%	No
Hwang et al. [85], 2020	CNN and autoencoder	Mirai	No	Networks	USTC-TFC2016, Mirai-RGU, Mirai-CCU	Accuracy-99.39%-100%	No
Proposed approach	deep ensemble learning	Mirai, Bashite, Bonesi	Yes	SDN controller, Networks	N-BaIoT, Testbed IoT devices data	F_1 Score 99.5-99.9%, MCC 91.04%-99.95%	Yes

The anomaly detection studies [84, 88, 96] show that ensemble learning models have been used to improve the performance of anomaly detection. Because the ensemble learning uses multiple algorithms to get better predictive performance than any single one of its constituent algorithms could [96]. Timcenko and Gajin [88] present an anomaly detection method for the IoT environment

using single and ensemble learning approaches and achieved 99% accuracy using the UNSW-NB15 dataset. However, the method lacks testbed experiments and dynamic attack scenarios. Debastrita et al. [84] report an ensemble learning method for outlier detection to handle problems in imbalanced datasets. This method employs stacked autoencoder (SAE) to extract deep features and input them to an ensemble of probabilistic neural network model for single and multiple outliers detection. The use of an autoencoder makes the method's performance better and more stable.

Unfortunately, only few studies address the data imbalance problem in IoT anomaly detection [48]. For instance, Zolanvari et al. [48] evaluate the efficiency of Artificial Neural Network (ANN) for anomaly detection in an imbalanced Industrial IoT (IIoT) testbed dataset. This method uses Synthetic Minority Over-Sampling Technique (SMOTE) for improving the performance of ANN in imbalanced IIoT dataset. Also, they analyze the limitations of machine learning-based intrusion detection solutions.

In recent years, advances in deep learning have transformed many areas of data-driven modeling [38,39,47,85,102]. Yair et al. [39] come up with a network-based anomaly detection method that uses deep autoencoders and to detect compromised traffic of IoT devices. This method is validated by using nine different commercial IoT devices in a laboratory environment with two famous IoT-based botnets, i.e., Mirai and BASHLITE. This method lacks variation in attack scenarios. To combat this, Nickolas et al. [38] present a BoT-IoT dataset, comprising legitimate and simulated IoT network traffic along with various types of attacks. Each existing datasets assessed against diversity, complexity, and experimental environments for evaluation of IoT attack detection algorithms. They validate the reliability of the Bot-IoT dataset by using statistical and machine learning algorithms. Thien et al. [102] present DÏoT, an autonomous self-learning distributed system for detecting compromised IoT devices using GRU (Gated Recurrent Units). DÏoT system creates device specific profiles without human intervention nor labeled data. The results enumerated that DÏoT detects IoT attacks in 257 milliseconds on average with a 95.6% True Positive Rate (TPR). Mahmudul et al. [47] propose an anomaly detection method for IoT sensors in IoT sites using machine learning approaches. This method is evaluated by using an

open-source dataset [61] with seven classes of attacks. They achieve 98.2%−99.4% accuracy for random forest and artificial neural networks. In a more recent work, Hwang et al. [85] present an unsupervised deep learning model for early network traffic anomaly detection in IoT using Convolutional Neural Network (CNN) and autoencoder known as D-PACK. D-PACK detects malicious traffic with nearly 100% accuracy with less than 1% false positive rate where it examines two packets per flow. Still, several opportunities are left to address IoT security problems in small and large-scale industries with a comprehensive focus in either centralized or distributed environments. Some problems include the increased uptime of devices that are deployed in the edge of the network, data or device heterogeneity, reliability, device status prediction, dynamic flow management, and dynamic attack detection. A comparison of the existing IoT anomaly detection methods is summarized in Table 4.1.

4.3 System Model

We present DeL-IoT, a deep ensemble learning framework to uncover IoT anomalies using SDN, primarily to detect anomalies and dynamic attacks for increasing device uptime, detection efficiency, switch-level dynamic flow management, and device status prediction. This framework aims to provide security for IoT devices by monitoring traffic and system metrics together in SDN switches. Also, it can handle the data imbalance problem where attack classes are rarer than legitimate classes. An architecture of the DeL-IoT system is given in Figure 4.1.

This framework has three primary components: including SDN controllers, SDN switches, and IoT devices. Further, the proposed framework has data collection and preprocessing, a learning module, a detection module, a flow management module, and the maintenance of a status table for forecasting device status. The network operators employ the SDN-enabled framework to isolate the services, increase reach-ability and improve service-oriented policies at the switch-level. The SDN controller disintegrates policies into service-specific rules and colonizes into flow tables of SDN-switches through the standard channel like OpenFlow [103]. Each packet is forwarded based on the enabled rules in the flow table. Each rule has three most common fields including *matching field*, *actions*,

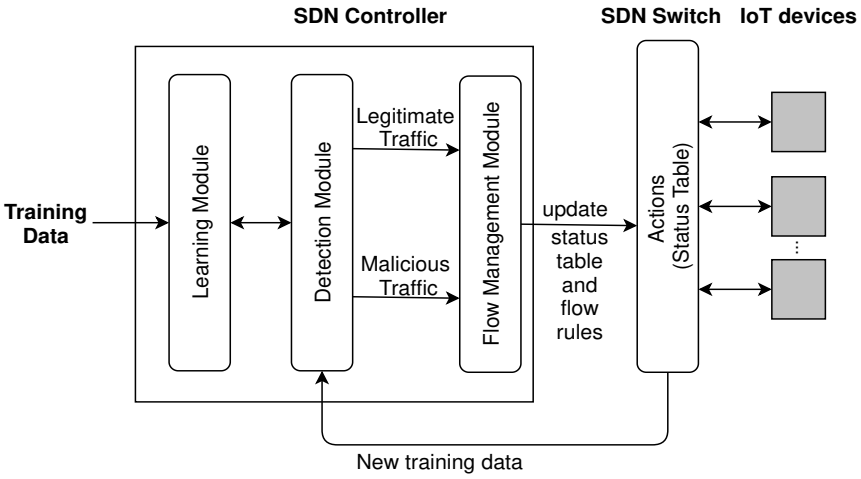


Figure 4.1: DeL-IoT system architecture.

and *flow counters*. If a packet header matches a rule, then the controller must take actions (e.g., forwarding to a specific device) and update the counters immediately. We assume that the controller has complete information of network topology and can make a request for each counter rule from switches [104]. A new rule is installed or updated reactively when new flows come to the network without any matching rules. In the following subsections, we discuss the anomaly detection, flow management and maintenance of the status table for forecasting device status.

4.3.1 Anomaly detection

The DeL-IoT framework aims to uncover anomalies in IoT based on the dynamic observation of both packet and flow level traffic instances that pass through SDN switches as well as system metrics. We deploy the detection module that can monitor traffic and system metrics of deployed devices as well as applications for anomaly detection.

Learning module

Anomaly detection models have a vital requirement to have aggregated data either at an endpoint or from multiple sources. The model adopts packet level, flow level, and system metrics data to detect anomalies in IoT. Because the attackers

primarily target different performance or system metrics. We validate this model by using both testbed and benchmark datasets. For testbed data, we set up a testbed comprising multilevel hierarchical architecture from physical infrastructures to IoT devices. The preprocessing function extracts significant features that the attackers typically utilize and labels them based on the behavioral analysis. In addition, the feature extraction procedure of testbed data is shown in Algorithm 1. For the benchmark data, we use a recent dataset, called N-BaIoT [39] for our experimentation. We provide extended explanations of each dataset in Section 4.4.

To make an efficient and automated representation of features, we use autoencoder and deep feature representation by a non-linear transformation of features set before feeding data into the learning model. This module employs both legitimate and anomalous features or system metrics to learn the model for anomaly detection in IoT. Let's assume that $X_n = \{x_1, x_2, \cdots, x_n\}$ is the input data, $n \in$, $X'_n = \{x'_1, x'_2, \cdots, x'_n\}$ is the encoded output, $F_n = \{f_1, f_2, \cdots, f_n\}$ is the features set, and h_n is the set of hidden layers.

Autoencoder and deep feature representation are multilayer neural networks having multiple hidden layers, h, to encode the input and to reconstruct the output as similar as possible to the input.The network has two parts: an encoder and a decoder. An encoder is defined as $E_n = f(w_1 X_n + b_1)$, where f is the encoding function with w_1 as weight vector, and b_1 is the bias. A decoder is defined as $X'_n = g(w_2 h_n + b_2)$, where g is the decoding function with weight matrix w_2, and bias b_2 [82]. Further, each parameter of the autoencoder is optimized to minimize the reconstruction error. We employ two categories of autoencoders, stacked autoencoder (SAE) and deep autoencoder (DAE) to extract and represent the features set obtained from the preprocessing function. The primary advantage of using deep autoencoder is having multiple hidden layers as shown in Figure 4.2. More hidden layers incur better feature representation, which is advantageous for an anomaly detection model [105].

Additionally, stacked autoencoder learns from the initial input data and obtained features make input to the next layer for reduced and compact features set (see Figure 4.3). The under-complete autoencoders have a lower number of nodes in hidden layers.

Algorithm 1 Flow Feature Extraction

Input: Raw flow parameters

Output: Extracted flow features

```
1: function MAIN(path)                                          ▷ get raw flows
2:     filter=[duration, protocol, srcIP, desIP, srcPort, desPort, packets, bytes,
   tos, idle_age]
3:     flowDic=OPENFILE(path, filter)    ▷ extract raw parameters and store in
   csv format
4: end function
5: function OPENFILE(path, filter)
6:     for line to file do
7:         if line.startswith "NXST_FLOW" in line:
8:           continue
9:         dic = PARSEF(line, filter)
10:        flowDic.append(dic)
11:    end for
12:    return flowDic
13: end function
14: function PARSEF(line, filter)
15:    for F in lists do                                  ▷ feature F in the list
16:      if (l > 1) then                                  ▷ if it's not a protocol
17:        estimate T = (t_2 − t_1) and F_n = {f_1, f_2, ··· f_n}    ▷ estimate interflow
   time, T, and extract all features, F
18:        flowDic.append(T, F)
19:      else
20:        flowDic.append(P)                              ▷ if it's a protocol
21:      endif
22:    endfor
23:    return dic
24: end function
```

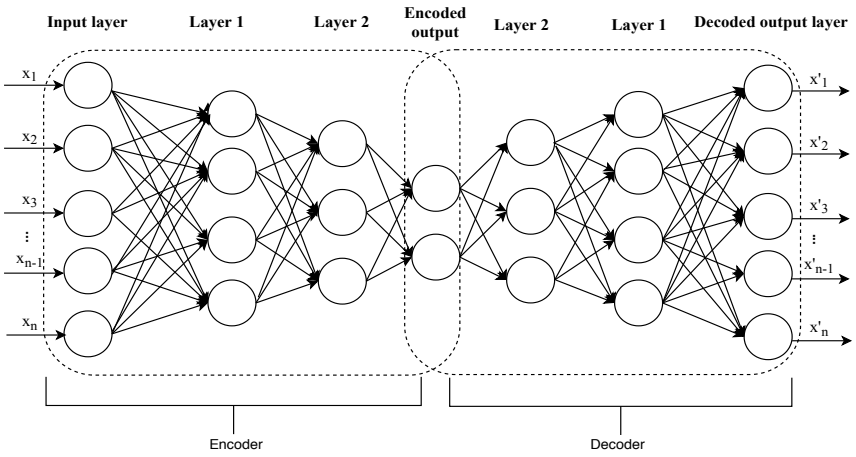


Figure 4.2: Deep autoencoder

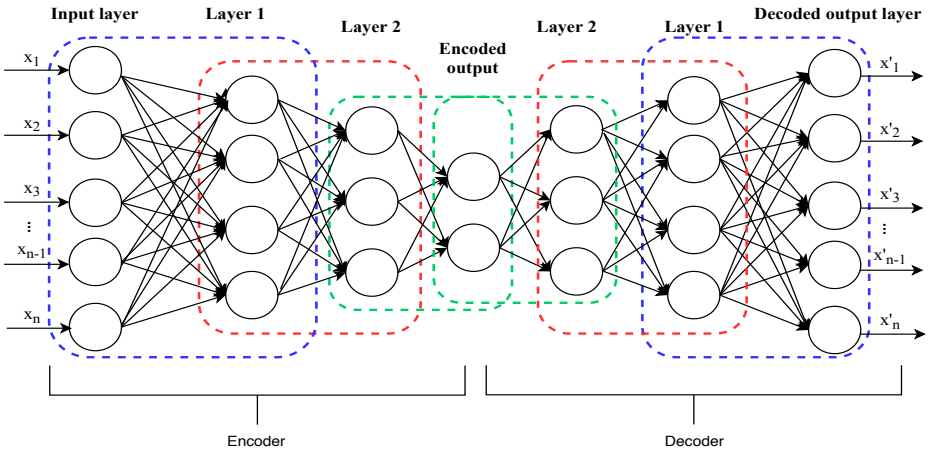


Figure 4.3: Stacked autoencoder

Detection module

This module employs the outcome of a learning module for detecting anomalies in IoT within an SDN-enabled framework. We explain the components of the detection module below.

Probabilistic Neural Networks (PNN) is a multilayered feedforward network with four primary layers, including input, pattern, summation, and output layers, as shown in Figure 4.4. The PNN is represented as a Kernel Discriminant Analysis (KDA), which is the generalization of Linear Discriminant Analysis (LDA), to

find the linear combination of features that separate classes. A PNN consists of several sub-networks that estimate the Parzen window probability density function of each class using the samples of the training set. Each node of the network calculates the probability density function for input x, training sample x_i, and class c_k according to the Eq. (4.1).

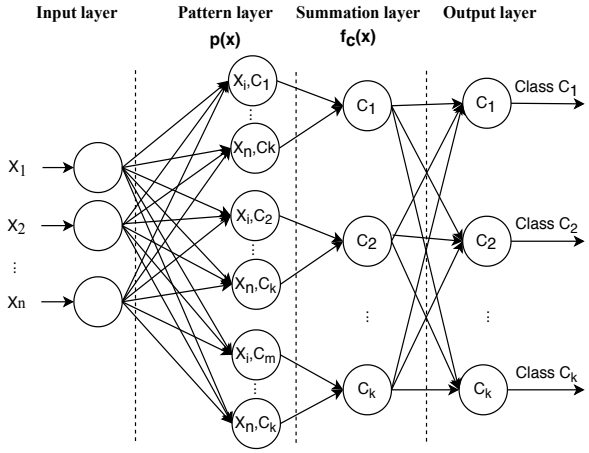


Figure 4.4: Architecture of Probabilistic Neural Networks (PNN)

$$p(x|x_i, c_k) = \frac{1}{\sigma}\omega\left(\frac{x - x_i}{\sigma}\right) \tag{4.1}$$

where x_i is the i^{th} sample and x is the input instance (unknown), $\omega()$ is the weighting function and σ is the smoothing parameter. The nodes are grouped according to the classes of the training sample in the pattern layer, and each group sums up for the next layer to get the class-wise probability. In the summation layer, the c_k^{th} nodes aggregate the values from the pattern layer of c_k^{th} classes. This summation is estimated based on a mixed Gaussian or Parzen window estimator as defined in Eq. (4.2).

$$f_{c_k}(x|x_i, c_k) = \frac{1}{n\sigma}\sum_{i=1}^{n_{c_k}}\omega\left(\frac{x - x_i}{\sigma}\right). \tag{4.2}$$

where n_{c_k} is the number of samples in c_k^{th} classes. Hence, the summation layer maps the c_k^{th} nodes to the c_k^{th} classes. For new samples, $f_{c_k}(x)$ can be estimated without retraining. The PNN needs more samples to achieve a high probability of mapping score from input instances to underlying classes where $f_{c_k}(x)$ has a maximum posterior probability of a class.

The weight function $\omega()$ is chosen as a kernel function (e.g., Radial Basis Function (RBF)) to compute the distance between the known and unknown sample points. If the distance is nearest, then it has more influence on the end class. The use of PNN provides multiple benefits, including insensitiveness to an outlier in the data, new input patterns stores in the network, and the smoothing parameter σ. Additionally, it reduces the retraining of the network if the training samples become large. Each sub-network of PNN implies a Parzen density estimator for a particular class. These features boost the detection of exceptional events in the data. However, it has been observed that PNN alone cannot provide better results. Because PNN, as a system, has a large storage requirement. Hence, we have used to handle such a high storage requirement by dimensional reduction using deep and stacked autoencoders and integration with ensemble probabilistic neural networks to detect anomalies in IoT.

Deep autoencoder and stacked autoencoder ensemble probabilistic neural networks (DAE-EPNN and SAE-EPNN) are autoencoders integrated with ensemble probabilistic neural networks. They encode the input by using multilayer neural networks instead of stacked layers. In anomaly detection, anomalous classes are rare, whereas legitimate classes are frequent. Hence, binary classifiers get more biased performance [94]. Such classifiers can be used to refine the decision boundary between the rare and frequent classes. In the proposed method, we have used deep and stacked encoders PNN for encoding the input and feeding them into PNN for classification. These inputs include samples from both rare and frequent classes. In anomaly detection, we have to make the trade-off between generalization and specialization to refine the decision boundary for achieving high accuracy. Most anomaly detection models are not specialized except just giving a bias to rare classes. The proposed model mitigates these drawbacks and gains substantial performance improvement thanks to deep ensemble learning.

In PNN, the smoothing parameter σ determines the spread of RBF when it

reaches a peak in the center of weighting. Selecting optimal values of σ implies a better spread of RBF in PNN. A shallow value of σ causes the model to over-fit whereas an extremely high value may cause the model to under-fit. However, both factors are essential to address the data imbalance problem during anomaly detection in IoT. These problems motivate us to develop a deep ensemble learning model, which we explain below.

Deep Ensemble Learning employs multiple PNNs with multiple layers as weak classifiers to address the biases by fine-tuning the smoothing parameter σ. This model takes inputs from the encoded features of the deep autoencoder PNN to construct inputs for the next layer. Let's assume that $A = \{x_{a1}, x_{a2}, \cdots, x_{an}\}$ is the set of anomalous instances, $L = \{x_{l1}, x_{l2}, \cdots, x_{ln}\}$ is the set of legitimate instances, D is the training dataset, Y is the test dataset, tr and te indicate training and testing instances.

As the legitimate instances are more than attack instances, we divided the legitimate instances into N subsets and we kept anomalous instances as one class for training. We used N number of PNNs with multiple layers for deep ensemble learning, where i^{th} PNN is trained with i^{th} subset of datasets. Hence, we chose i^{th} PNN for training with $(i+1)^{th}$ PNN for binary classification. However, we chose N^{th} PNNs for the majority of the classes and one more PNN for an anomalous class, specifically for the multiclass problem. For this, we used deep encoder, since we know that encoder provides the non-linear transformation of input data to reduce features set. A pictorial representation of the proposed model is given in Figure 4.5.

Let g be a non-linear activation function with weight w and bias b then the deep encoder is formulated as in Eq. (4.3). The choice of parameters is explained in Section 4.4.

$$\begin{aligned} \varepsilon &= \Big(g(wx+b)\Big) \\ DAE(x) &= \Big(\varepsilon_1(\varepsilon_2(\varepsilon_3(\cdots \varepsilon_h(x))))\Big) \end{aligned} \tag{4.3}$$

where $\varepsilon()$ is the encoding function whereas $\varepsilon_i()$ is the i^{th} deep encoder, h is the number of hidden layers, each feature vector x is transformed to $\hat{x}$ using DAE(x)

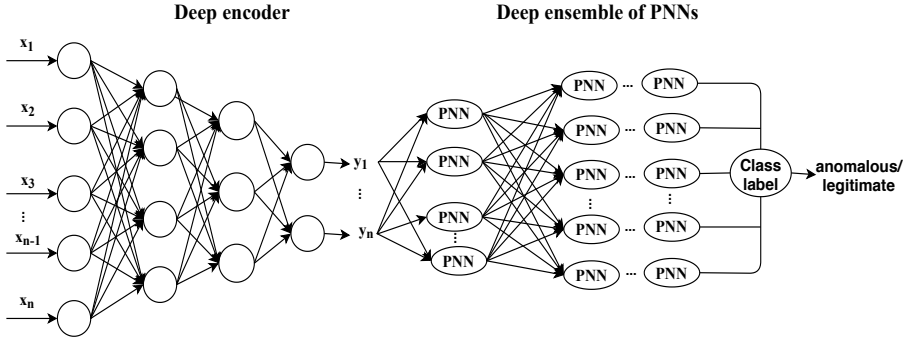


Figure 4.5: The proposed method: an integration of deep encoder and deep ensemble of PNNs

defined in Eq. (4.4). The encoded features set, $\hat{x}_1, \hat{x}_2, \cdots \hat{x}_{F'}$ is embedded to deep ensemble learning model.

$$z_{c_k}^n(\hat{x}) = f_{c_k}^n\Big(DAE(x)\Big) \tag{4.4}$$

Each input instance $\hat{x}$ is assigned to c_k classes based on intermediate RBF score received from the n^{th} PNNs, where f estimates intermediate RBF scores based on encoded features set to decide belonginess of a class. However, if there are N-PNNs for the ensemble, then each instance $\hat{x}$ is assigned to c_k classes based on the Eq. (4.5). The classification score is computed using Eq. (4.6) for each instance $\hat{x}$ that belongs to a specific class and where n represents batch size.

$$z_{c_k}(\hat{x}) = \sum_{n=1}^{N}\Big(z_{c_k}^n(\hat{x})\Big) \tag{4.5}$$

$$s_{c_k}(\hat{x}_1^n) = \frac{z_{c_k}(\hat{x}_1^n) - min(z_{c_k}(\hat{x}_1^n))}{max(z_{c_k}(\hat{x}_1^n)) - min(z_{c_k}(\hat{x}_1^n))} \tag{4.6}$$

Once we get the classification score for each instance, we label the unknown instance $\hat{x}$ as anomalous or legitimate based on the node's maximum probability, $p_{c_k}(\hat{x}) = max(s_{c_k}(\hat{x}))$. Each layer of deep ensemble learning employs majority voting to classify anomalies in IoT. This process repeats for the next layers of the deep PNNs to improve overall performance. The major steps in deep ensemble learning to uncover anomalies are defined in Algorithm 2.

Algorithm 2 DeL-IoT : a deep ensemble learning

Input: IoT dataset, D

Output: Uncover D as legitimate or anomalous

normalize original dataset D using MinMaxScalar, and to get D_1
train SAE or DAE on D_1 with hyper-parameters to obtain D_2 ▷ D_2 represents deep and compact features set.
if $(D_2 := s_a)$ then ▷ s_a indicates single attack
  chose *adam* optimizer and *binary_crossentropy* as a loss function
else
  chose *adam* optimizer and *categorical_cross_entropy* as a loss function
endif
construct feature set, D_2, by *relu* activation values of hidden layers in SAE or DAE using Eq. (4.3-4.4)
Split D_2 for training dataset D_{tr} and for testing dataset D_{ts}
for $j = 1$ to number of ensemble PNNs **do**
    train PNN on D_{tr} using Eq. (4.1-4.2)
    for $i = 1$ to number of D_{ts} **do**
        compute maximum class probability score using Eq. (4.6)
        classify $D_{ts}[i]$ samples using Eq. (4.5)
    end for
end for

4.3.2 Flow management

This module is enabled to prepare appropriate rules for each attack and to dynamically update the rules in the SDN switch as a set of actions. IoT devices are connected to the SDN switches. We employ the features of the POX framework [34] with OpenFlow protocol [104] to enable flow control and management at the SDN switch. In addition, SDN switches continuously monitor traffic flows of IoT devices and provide statistical information to the SDN controller.

We consider three scenarios for dynamic flow management in the DeL-IoT framework that leverage SDN-enabled the POX controller [34]. First, if the input flow is legitimate then it passes through the switches immediately without any interruptions. Further, if the flow is unknown then the SDN switch sends it to the deployed detection module for further investigation and, for the time being, applies rate-limiting to the traffic to control intended malicious flow within the networks. For instance, this unknown flow may end-up with an intended attack that trims the overall network performance. Second, if the input flow is detected as anomalous then a set of rules are applied to control them. At the beginning, the flow is dropped immediately and the deployed detection module investigates the source of the attacks to blacklist them. Hence, the attackers cannot reach any IoT devices to damage the entire network. Further, if a number of devices are under attack, then network-wide rules will be updated and applied to maintain network performance. Third, if the IoT devices are compromised then two actions are usually taken. The flow of traffic will be immediately blocked and investigated further for verifying the kind of malicious flows: malware or physical attacks.

The SDN enabled POX controller for managing dynamic traffic flow and generating a status table for IoT devices is shown in Figure 4.6. Based on the IoT device profile, this module inserts an entry into the status table while considering the status for a time period. It enters 0 for legitimate status and 1 for anomalous status. This status information is further utilized to forecast the device status for short and long-term using sequence modelling algorithm.

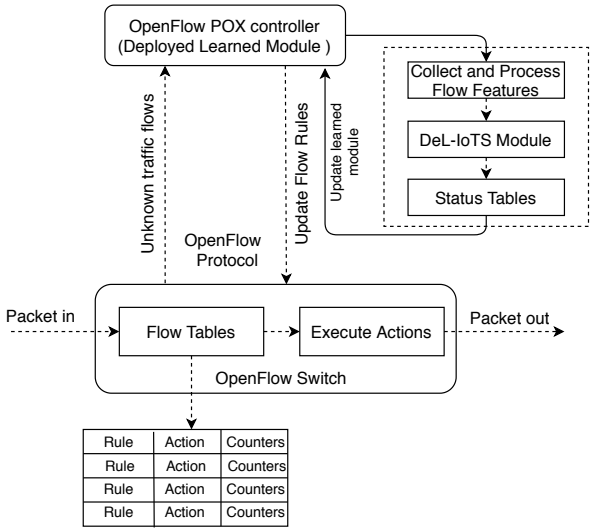


Figure 4.6: Reactive flow management architecture

4.3.3 Forecasting IoT device status

This module takes input from the flow management module and generates profiles for each IoT device with the outcome of metrics such as packet, flow, and system. This table is comprised of three columns, including time, device ID, and status, as shown in Table 4.2. The status of each IoT device is marked either as 0 (legitimate) or as 1 (anomalous) for each time point. Based on the status table information, we will predict device future status using a sequence modelling algorithm based on LSTM, steps are shown in Algorithm 3. The system manager utilizes this feature to easily handle large-scale IoT devices as well as their services to the end-users.

Table 4.2: The proposed IoT device status table

Device ID	Time	Status
dl_source	system_time	1-Anomalous 0-Legitimate

Algorithm 3 LSTM-based IoT device status forecasting

Input: IoT device status information

Output: Forecasting IoT device status

1: initialize time, t and next time step, $t+1$, loop_back=1, dataset D
2: Split D for training dataset D_{tr} and for testing dataset D_{ts}
3: D_{tr} and D_{ts} convert the status array into data matrix, Dx_{tr}=t, Dy_{tr}=t+1, Dx_{ts}=t and Dy_{ts}=t+1
4: Dx_{tr},Dy_{tr}=CREATE_DATASET(D_{tr},loop_back) and Dx_{ts},Dy_{ts}=CREATE_DATASET(D_{ts},loop_back)
5: trainPredict = lstm.predict(Dx_{tr}) and testPredict = lstm.predict(Dx_{ts}) ▷ device status prediction
6: **function** CREATE_DATASET(dataset, loop_back=1)
7: dataX, dataY = [], []
8: **for** i in range(len(dataset)-look_back-1) **do**
9: a = dataset[i:(i+look_back), 0]
10: dataX.append(a)
11: dataY.append(dataset[i + look_back, 0])
12: **end for**
13: **return** dataX, dataY
14: **end function**

4.4 Performance Evaluation

This section reports and explains the intensive experimental results obtained from a testbed and benchmark datasets. We begin with the dataset's description and proceed with experimental results.

4.4.1 Datasets

The DeL-IoT framework is evaluated using two datasets: (i) testbed data, and (ii) benchmark data. The testbed data is generated with a significant amount of attacks on IoT devices including applications in the physical infrastructures.

Testbed data: The experiment is performed in a virtualized environment with a hierarchical deployment of IoT devices to physical servers in the testbed. Figure

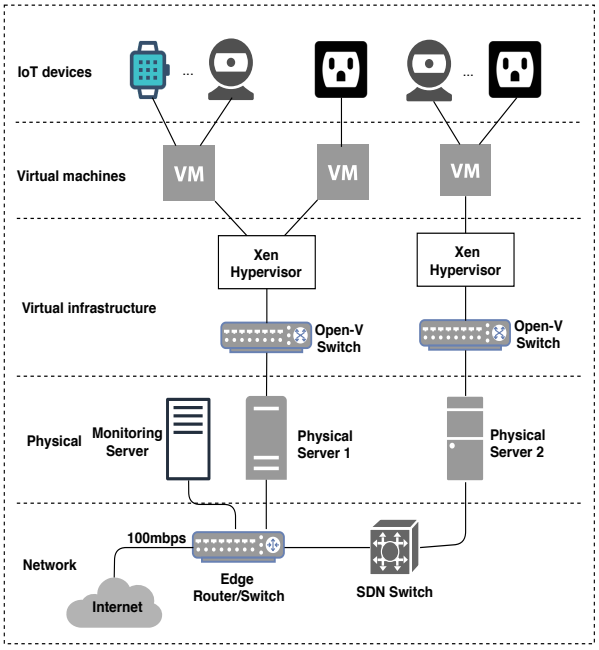
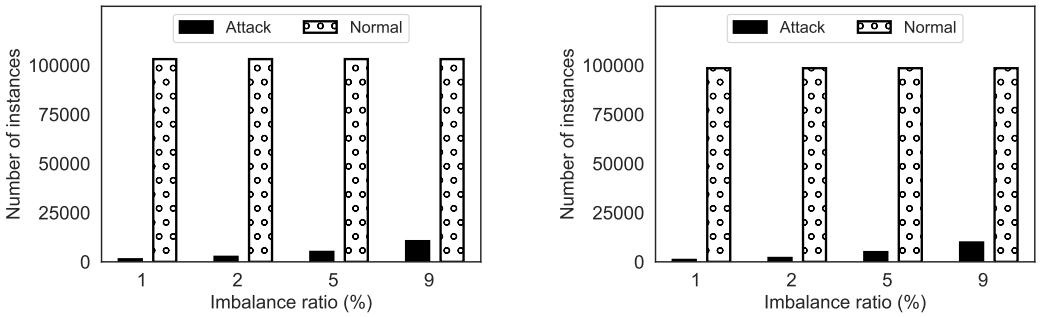


Figure 4.7: Testbed setup

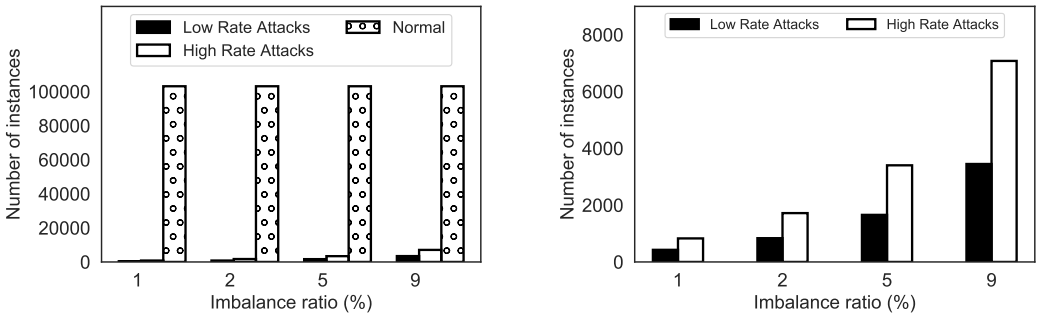
4.7 illustrates the architecture of the testbed setup. The testbed is comprised of multiple servers and applications at a physical level, virtualized level, and IoT devices. We consider multiple VMs, one of the VMs is a target of attackers. We generate both Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks using the Targa2[¶] attack generator, the D-ITG internet traffic generator [106], the BoNeSi botnet simulator[‖], and the stress-ng[**] system resources load generator tools. We generate multiscale attacks [107] to the IoT devices including applications in the physical infrastructures when they are deployed in the virtualized environment. We collect multiple metrics (e.g., packet, flow, system metrics (Key Performance Indicators), device status, etc.) from devices to physical infrastructures for learning and deploying the proposed model. The number of instances of the testbed dataset is given in Figures 4.8-4.9 where we consider 1% to 9% attack instances for data imbalance scenarios in our experi-

[¶]http://packetstormsecurity.com/
[‖]https://github.com/Markus-Go/bonesi
[**]https://kernel.ubuntu.com/~cking/stress-ng/



(a) Testbed data (b) Benchmark data

Figure 4.8: Single attack imbalanced datasets



(a) With normal instances (b) Without normal instances

Figure 4.9: Multiple attack imbalanced testbed datasets

ments. POX[††] is an OpenFlow controller used to deploy the learning algorithm to detect anomalies in IoT devices based on dynamic policy updating within the SDN framework and also for short and long term forecasting of IoT device status.

Benchmark data: Due to the non-availability of benchmark datasets, we used the N-BaIoT [39] dataset for our experiments. This dataset was prepared using two attack generation tools, i.e., Mirai (scan, ACK flooding, SYN flooding, UDP flooding, UDPplain attacks) and Bashlite (scan, junk, UDP flooding, TCP flooding, COMBO attacks), with 9 commercial IoT devices. There are 5 Bashlite, 5 Mirai, and 1 legitimate datasets, having 115 features in each of them. We created an imbalanced dataset by separating 98514 legitimate and 9850 anomalous

[††]https://github.com/noxrepo/pox

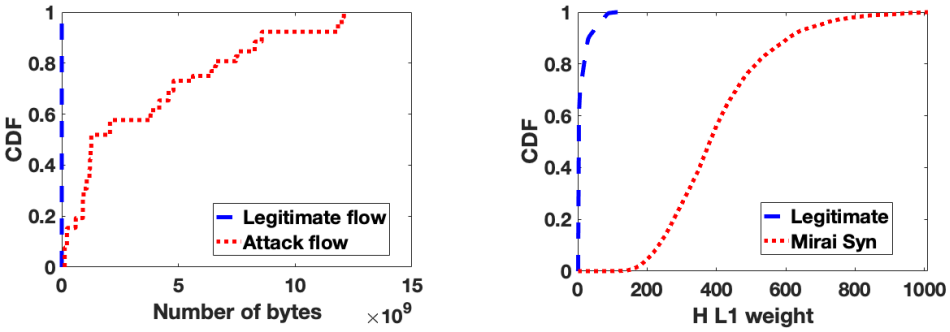
instances with a combination of 10 different kinds of Mirai and Bashlite attacks. In addition to dynamic attack detection in IoT, we also address the data imbalance problems. Hence, we have created imbalanced datasets with variation of dynamic attacks from N-BaIoT dataset. Figures 4.8-4.9 illustrate the multiple scenarios of imbalanced datasets, where we consider 1% to 9% attack instances.

4.4.2 Results

The proposed framework is evaluated using both testbed and benchmark datasets. We present the results based on testbed datasets that consider multilevel metrics such as packet, flow, and system metrics in synchronization for validation. In the single attack experiment, we used deep and stacked autoencoders with an ensemble of PNNs with the 'adam' optimizer and 'binary-crossentropy' as a loss function. Likewise, in the multiple attacks, we used deep and stacked autoencoders with an ensemble of PNNs with the 'adam' optimizer and 'categorical cross-entropy'as a loss function. First of all, we do some experiments as to how our proposed model results vary with values of σ and the depth of the DAE or SAE network used. Second, based on the experimental results, the hyperparameters were chosen by optimizing the validation set. Finally, the model was pre-trained with 100 epoch, 100 batch size, and the 'relu' activation function by considering four hidden layers with the compositions 17-16-16-15 for testbed data and 115-100-50-25 for benchmark data. Regarding the testbed and benchmark datasets, we have used non-overlapping datasets: 70% for training and 30% for testing, so that the class imbalance problem remains the same in both the training and the test set. However, full data was considered for training of the deep or stacked autoencoders and then we fed them to the deep ensemble of PNNs.

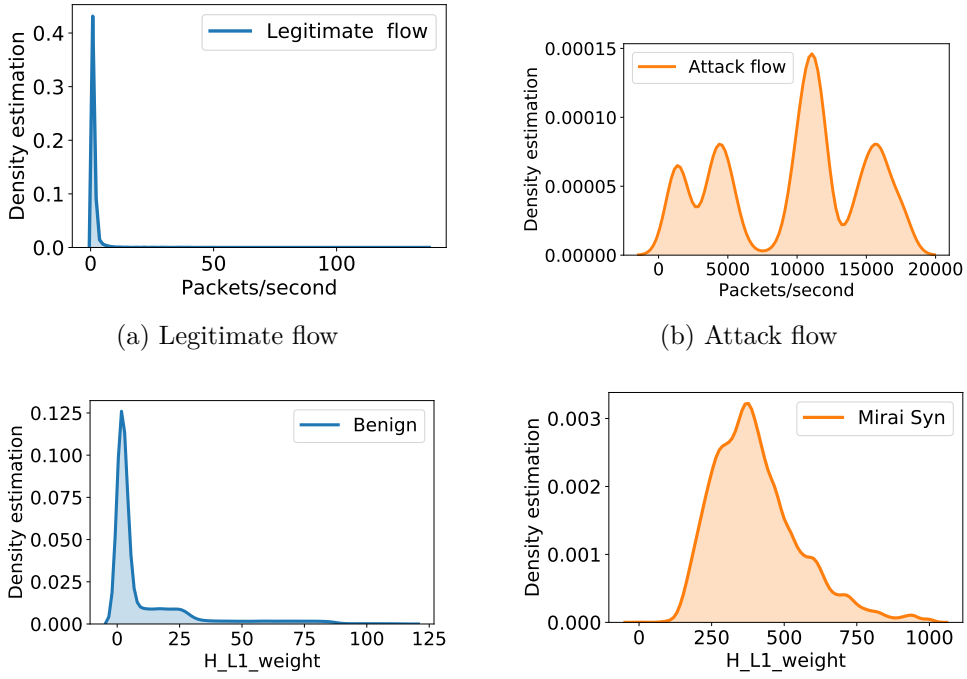
Characterization of data

To observe the behaviour of both testbed and benchmark datasets, we estimate the cumulative distribution function and kernel density for legitimate and attack instances shown in Figures 4.10 and 4.11, respectively. From the Figures 4.10 and 4.11, it is clear that the distribution of legitimate instances differs from attack instances.



(a) Testbed data (b) Benchmark data

Figure 4.10: Characterization of data: cumulative distribution function for legitimate vs. attack instances over feature set



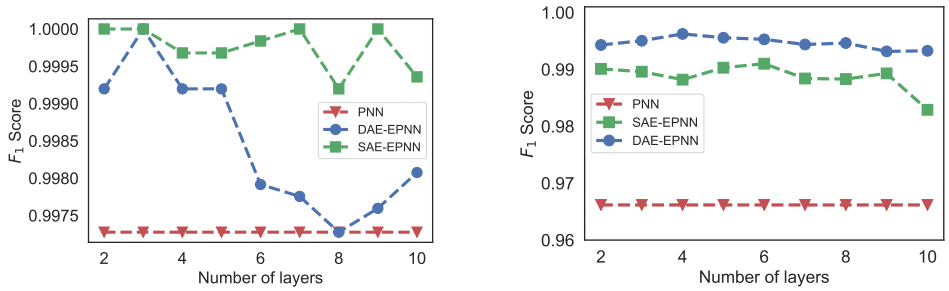
(a) Legitimate flow (b) Attack flow

(c) Benign (d) Mirai Syn Flood

Figure 4.11: Characterization of data: Kernel density estimation of legitimate vs. attack instances with testbed (a,b) and benchmark (c,d) datasets

Table 4.3: An example of status table

Device ID	Time	Status
10:0e:7e:c9:cb:f0	13:49	0
10:0e:7e:c9:cb:f0	13:50	0
10:0e:7e:c9:cb:f0	13:51	0
10:0e:7e:c9:cb:f0	13:52	0
98:f2:b3:f3:2a:38	13:52	1
98:f2:b3:f3:2a:38	13:53	1

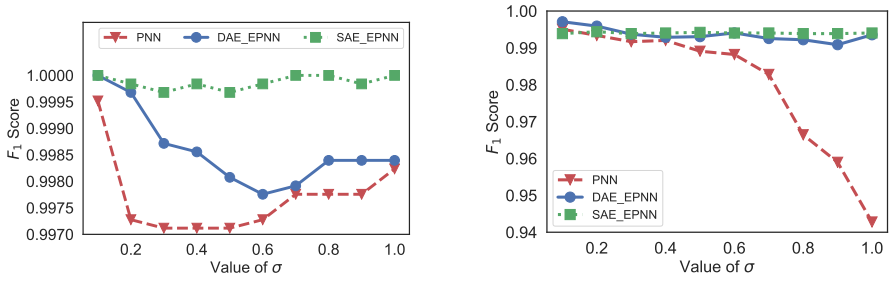


(a) Testbed (b) Benchmark

Figure 4.12: Performance variation of F_1 score with respect to the number of deep layers

Dependency analysis on deep layers

This section investigates the performance of the proposed framework based on the number of deep layers. We have used stacked and deep autoencoders to extract features from both testbed and benchmark datasets. Truly, it is noted that an increasing number of hidden layers in the autoencoder excels the performance of the framework. However, finding the optimal depth of hidden layers for a specific domain and addressing the data imbalance problem in parallel is still tricky. Figure 4.12 provides the empirical investigation on the number of deep layers (3 for the testbed and 4 for the benchmark datasets) to acquire the best performance of the model. Our model recommends that deep neural networks improve the model performance significantly, but the extremely deeper number of layers may overfit the model.



(a) Testbed (b) Benchmark

Figure 4.13: Performance variation of F_1 score σ of PNN

Dependency analysis on σ

The importance of dependency analysis on σ lies on smoothing the parameter of PNN. The σ is the spread of the Gaussian function or the width parameter which can take σ value between 0 and 1 [108]. The proposed method is evaluated on the dependency of σ for achieving a high detection rate. Notably, the lower values of σ provide a lower false-negative rate and higher values of σ yield a lower false-positive rate. However, we heuristically identify the values of σ as $0.7 - 0.8$ for testbed data and $0.1 - 0.2$ for benchmark data, respectively, when considering imbalanced datasets. Figure 4.13 illustrates the performance in the testbed and benchmark datasets, respectively. As shown in Figure 4.13, in the benchmark dataset our proposed method's detection rate is above 99%, but single PNN's detection rate decreases from 99% to 94%. Hence, we observe that our proposed method's detection rate is more stable than a single PNN model and a single PNN detection rate decreases when the sigma value increases.

Performance on single attacks

To validate the efficiency of the proposed method on the single attack scenarios, we have used balanced and imbalanced N-BaIoT datasets. First of all, Table 4.5 shows the accuracy of the proposed method in the balanced N-BaIoT [39] datasets. Second, these sets of experiments address the data imbalance problem by considering a single attack and legitimate datasets. Hence, we created from the N-BaIoT [39] dataset's Provision-PT-838 Security Camera 5 kinds of imbal-

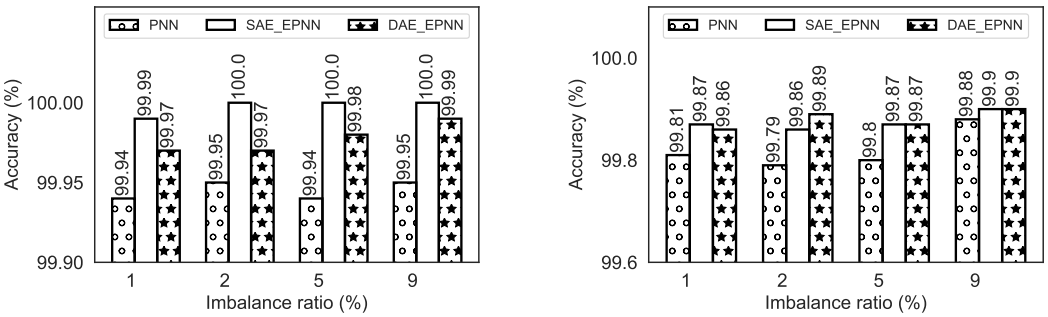
Table 4.4: The detection accuracy for single attack scenarios when running with the imbalanced N-BaIoT dataset

No	IoT devices	Benign	Attacks	PNN	DAE-EPNN	SAE-EPNN
1	SimpleHome-XCS7-1003-WHT Security Camera	19528	1950	0.9933	**0.9959**	0.9944
2	Ecobee-Thermostat	13113	1310	0.9953	**0.9979**	0.9967
3	Ennio-Doorbell	39100	3910	0.9899	**0.9960**	0.9958
4	Provision-PT-838 Security Camera	98514	9850	0.9901	**0.9987**	0.9983
5	Danmini Doorbell	49548	4950	0.9962	0.9970	**0.9971**
6	Samsung-SNH-1011-N Webcam	52150	5215	0.9976	**0.9988**	0.9985
7	SimpleHome-XCS7-1002-WHT Security Camera	46585	4650	0.9918	**0.9951**	0.9944
8	Provision-PT-737E Security Camera	62154	6210	0.9864	**0.9973**	0.9966
9	Philips-B120N10 Baby Monitor	175240	17520	0.9925	0.9960	**0.9988**

anced datasets shown in Figure 4.8. From Tables 4.5-4.4 and Figures 4.14-4.16, we observe that the integration of deep and stacked autoencoders with the deep ensemble of PNNs allowed the improved performance of the model while addressing the data imbalance issues and detecting attacks in IoT. As shown in Figure 4.14, there are slight differences in accuracy performance. However, it is not true. In anomaly detection scenarios with the imbalanced dataset, accuracy is not the representative best metric to evaluate the performance. Since a large portion of training data is legitimate traffic, the algorithms are biased toward estimating all the data as legitimate and ignoring the small portion of the attack instances [48]. Hence, MCC and F_1 score measures can evaluate the performance better in spite of having imbalanced datasets. In addition, the class imbalance problem introduces a bias in the machine learning models that degrades performance. For instance, from Figures 4.15-4.16, we observe that the machine learning models' MCC and F_1 score decrease when the dataset's imbalanced ratio decrease.

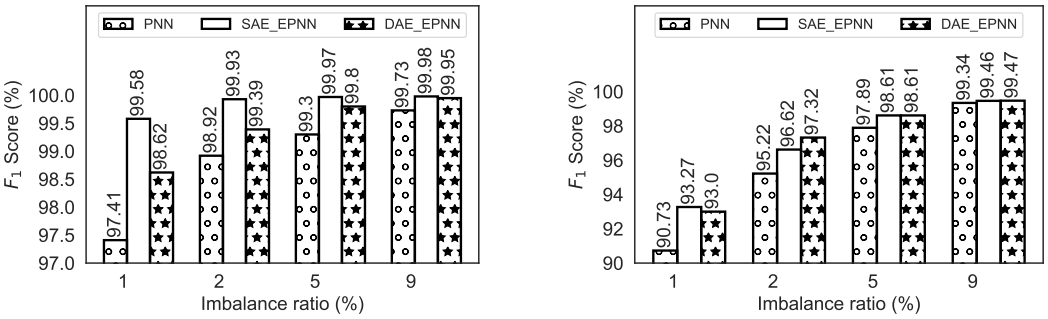
As shown in Figures 4.15-4.16, we can also observe that in the 1%, 2%, and 5% imbalanced datasets, our proposed method's MCC and F_1 performances are 1%-3% better than a single model.

Finally, since 1% is our experiment's lowest imbalanced dataset, ten-fold cross-validation was performed on the 1% imbalanced testbed and benchmark datasets using a single model and the proposed methods. Figure 4.17 shows the improved performance results of our proposed framework after ten-fold cross-validation.



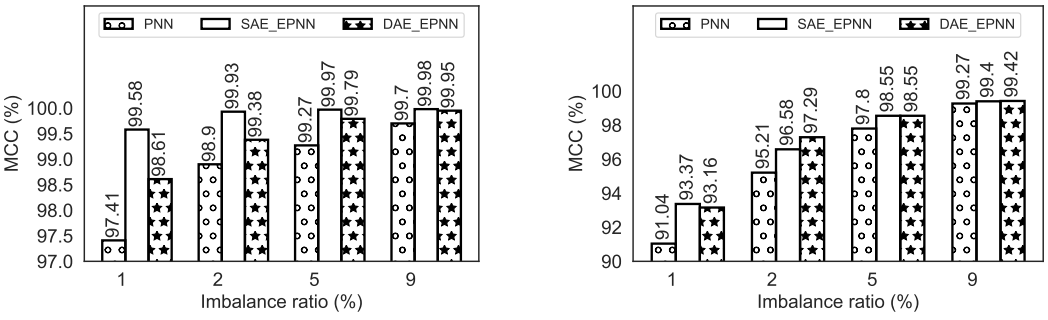
(a) Testbed (b) Benchmark

Figure 4.14: Accuracy variation for single attacks imbalanced datasets



(a) Testbed (b) Benchmark

Figure 4.15: F_1 score comparison for single attacks imbalanced datasets



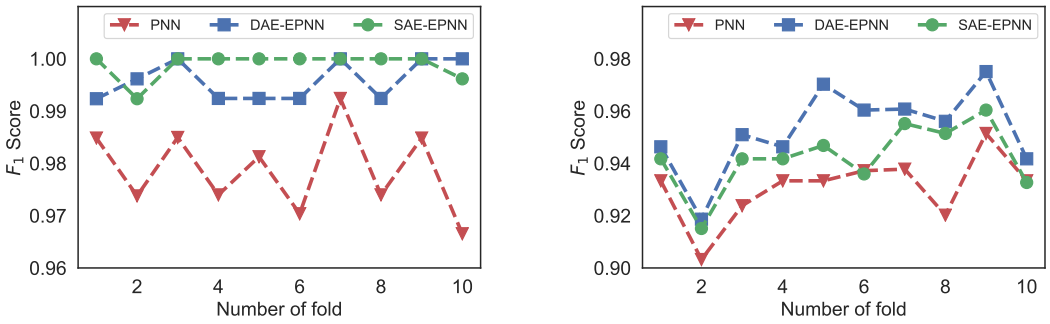
(a) Testbed (b) Benchmark

Figure 4.16: MCC score variations for single attacks imbalanced datasets

Table 4.5: The detection accuracy for single attack scenarios when running with the balanced N-BaIoT dataset

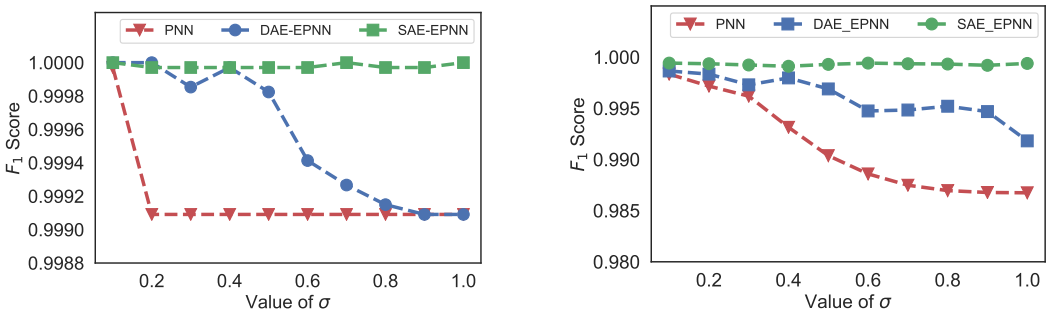
No	IoT devices	Bashlite	PNN	DAE-EPNN	SAE-EPNN	Mirai	PNN	DAE-EPNN	SAE-EPNN
1	SimpleHome-XCS7-1003-WHT Security Camera	Scan	0.9943	0.9993	**0.9994**	Scan	0.9976	**0.9997**	0.9996
		TCP	0.9985	**0.9986**	**0.9986**	Syn	**0.9999**	**0.9999**	**0.9999**
		Junk	0.9990	0.9997	**0.9998**	UDP-plain	0.9996	0.9998	**1.0000**
		UDP	0.9984	**0.9986**	0.9985		0.9997	**0.9999**	**0.9999**
		Combo	0.9979	**0.9998**	**0.9998**	Ack UDP	0.9992	0.9999	**1.0000**
2	Ecobee-Thermostat	Scan	0.9947	**0.9988**	0.9933	Scan	0.9989	**0.9993**	0.9950
		TCP	0.9992	0.9992	**0.9988**	Syn	0.9999	**1.0000**	0.9998
		Junk	0.9987	**0.9996**	0.9975	UDP-plain	0.9970	**1.0000**	0.9967
		UDP	0.9990	**0.9991**	0.9986		0.9992	0.9999	**1.0000**
		Combo	0.9984	**0.9996**	0.9823	Ack UDP	0.9983	**0.9999**	0.8034
3	Ennio-Doorbell	Scan	0.9880	0.9974	**0.9998**	No Mirai Attacks	-	-	-
		TCP	0.9986	0.9986	**0.9996**				
		Junk	0.9992	**0.9999**	0.9998				
		UDP	0.9985	0.9986	**0.9993**				
		Combo	0.9967	0.9998	**1.0000**				
4	Provision-PT-838 Security Camera	Scan	0.9973	0.9974	**0.9998**	Scan	0.9988	0.9998	**1.0000**
		TCP	0.9987	**0.9988**	0.9990	Syn	0.9969	0.9993	**1.0000**
		Junk	0.9996	**0.9999**	**0.9999**	UDP-plain	0.9999	**1.0000**	**1.0000**
		UDP	0.9988	0.9990	**0.9991**		0.9997	0.9997	**0.9999**
		Combo	0.9988	**0.9999**	**0.9999**	Ack UDP	**0.9999**	**0.9999**	**0.9999**
5	Danmini Doorbell	Scan	0.9937	0.9994	**0.9995**	Scan	0.9993	**0.9999**	0.9992
		TCP	0.9987	0.9992	**0.9996**	Syn	0.9998	**1.0000**	0.9425
		Junk	0.9960	0.9993	**0.9997**	UDP-plain	**0.9999**	**0.9999**	**0.9999**
		UDP	0.9987	0.9990	**0.9993**		**1.0000**	0.9999	0.9803
		Combo	0.9948	0.9998	**0.9999**	Ack UDP	**1.0000**	**1.0000**	0.9996
6	Samsung-SNH-1011-N Webcam	Scan	0.9969	0.9989	**0.9995**	No Mirai Attacks	-	-	-
		TCP	0.9987	0.9992	**0.9993**				
		Junk	0.9994	**0.9998**	**0.9998**				
		UDP	0.9986	0.9989	**0.9994**				
		Combo	0.9992	**0.9999**	**0.9999**				
7	SimpleHome-XCS7-1002-WHT Security Camera	Scan	0.9986	**0.9974**	0.9994	Scan	0.9986	0.9995	**0.9997**
		TCP	0.9959	**0.9987**	0.9986	Syn	0.9959	**0.9999**	**0.9999**
		Junk	0.9999	**0.9996**	0.9998	UDP-plain	0.9999	**1.0000**	**1.0000**
		UDP	0.9986	**0.9988**	0.9985		0.9999	**1.0000**	0.9999
		Combo	0.9990	**0.9998**	0.9998	Ack UDP	0.9999	0.9999	**1.0000**
8	Provision-PT-737E Security Camera	Scan	0.9965	**0.9993**	0.9991	Scan	0.9968	**1.0000**	**1.0000**
		TCP	0.9978	0.9989	**0.9990**	Syn	0.9850	0.9994	**0.9999**
		Junk	0.9967	**0.9998**	**0.9998**	UDP-plain	**0.9999**	**0.9999**	**0.9999**
		UDP	0.9977	**0.9988**	**0.9988**		**0.9999**	**0.9999**	**0.9999**
		Combo	0.9987	0.9999	**1.0000**	Ack UDP	**1.0000**	0.9999	**1.0000**
9	Philips-B120N10 Baby Monitor	Scan	0.9971	0.9992	**1.0000**	Scan	0.9995	**1.0000**	**1.0000**
		TCP	0.9993	**0.9995**	0.9994	Syn	0.9999	**1.0000**	**1.0000**
		Junk	0.9995	**0.9999**	**0.9999**	UDP-plain	0.9998	**1.0000**	**1.0000**
		UDP	0.9993	0.9994	**0.9995**		0.9999	**1.0000**	**1.0000**
		Combo	0.9985	**0.9999**	**0.9999**	Ack UDP	0.9995	**1.0000**	**1.0000**

The k-fold cross validation is sensitive to poor partition of data. As a result, the accuracy of the baseline model varies due to poor partition of highly imbalanced data.



(a) Testbed (b) Benchmark

Figure 4.17: F_1 score of models after 10 fold cross validation for detecting single attacks

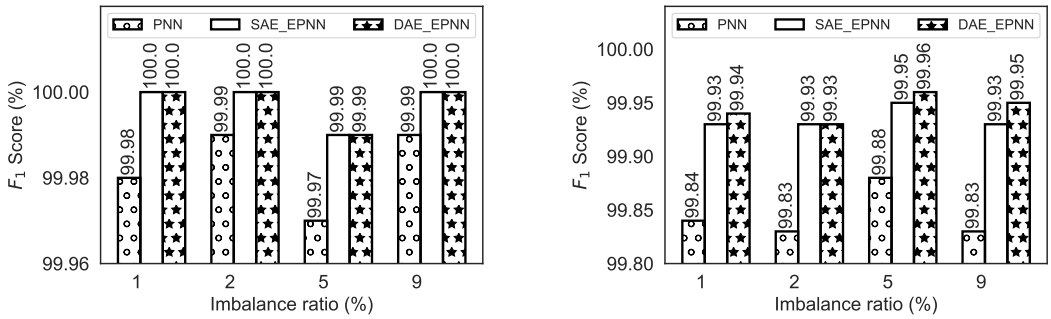


(a) Testbed (b) Benchmark

Figure 4.18: Multiple attacks performance variation of F_1 score σ of PNN

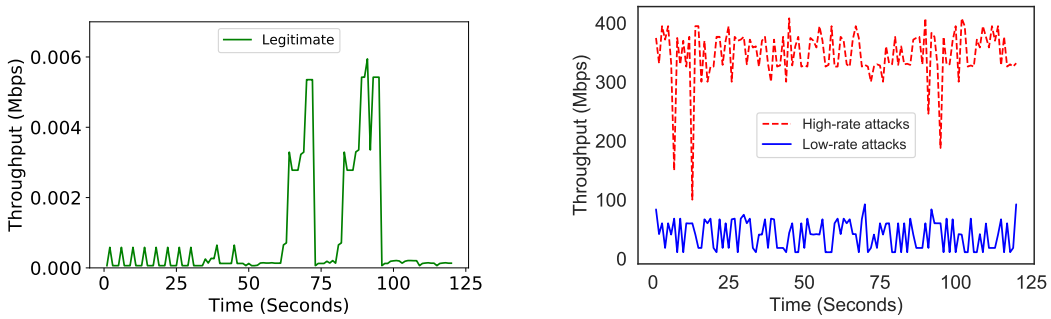
Performance on multiple attacks

The proposed framework is evaluated further for multiple attacks using testbed and benchmark datasets. In the testbed data, we consider multiscale attacks (i.e., DoS, DDoS) with data imbalance scenarios. As shown in Figure **??**, we have found multiple attacks for our experiment from the benchmark datasets. Figures 4.18-4.19 illustrate the improved performance of the proposed framework while detecting multiple attacks in IoT .



(a) Testbed (b) Benchmark

Figure 4.19: Multiple attacks F_1 score for imbalanced datasets



(a) Legitimate (b) Attacks

Figure 4.20: Throughput - (a) legitimate flows, and (b) attack flows

Performance on dynamic flow management

To observe the performance of dynamic flow management, we consider two different cases with and without attacks in the testbed setup environment. Figure 4.20 illustrates the flow management by examining the throughput in the absence and presence of attacks (low-rate and high-rate), respectively. Figure 4.21 shows the dynamic management of flows that pass through the software-enabled switch. From the Figures 4.20-4.21, we observe that the traffic flows are managed well in both periods with and without attacks.

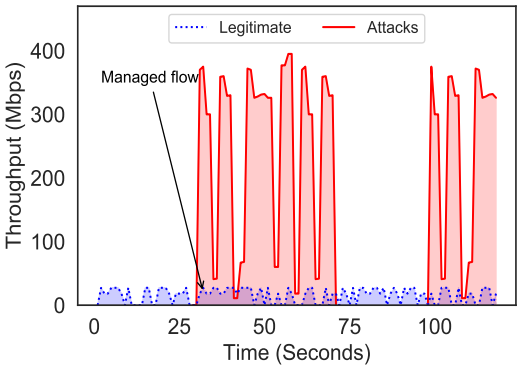


Figure 4.21: Throughput - managing flows dynamically

Performance on device status forecasting

The status table is obtained from the flow management module based on the status of each IoT device in the testbed setup. Table 4.3 is an example status table obtained by examining the time period 13:49 to 13:53 within the testbed environment. DeL-IoT offers additional features that can predict both short and long-term anomalies in IoT devices. We exploit the Long Short-Term Memory (LSTM) model for the prediction of anomalies as shown in Figure 4.22. In addition, our proposed LSTM model achieved an accuracy of 81.48%. This prediction increases the effectiveness to protect the IoT devices against emerging attacks. In this experiment, we used 84 minutes of data for training and 36 minutes of data for testing with an LSTM-based sequential model for predicting the status of IoT devices. Based on empirical evaluation, we found that the model with 64 hidden neurons has lower mean square error (MSE) than the models with 4, 32, 128 hidden neurons. For this reason, we determine that the number of hidden neurons of LSTM is 64. Hence, the network has a visible layer with 1 input (a device status), a hidden layer with 64 LSTM blocks or neurons, and an output layer that makes a single value prediction. The default sigmoid activation function is used for the LSTM blocks and the network is trained for 100 epochs with batch size 1.

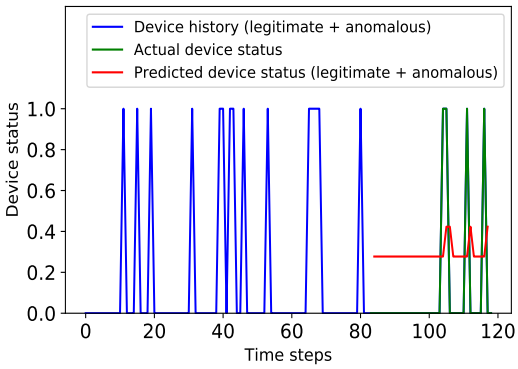
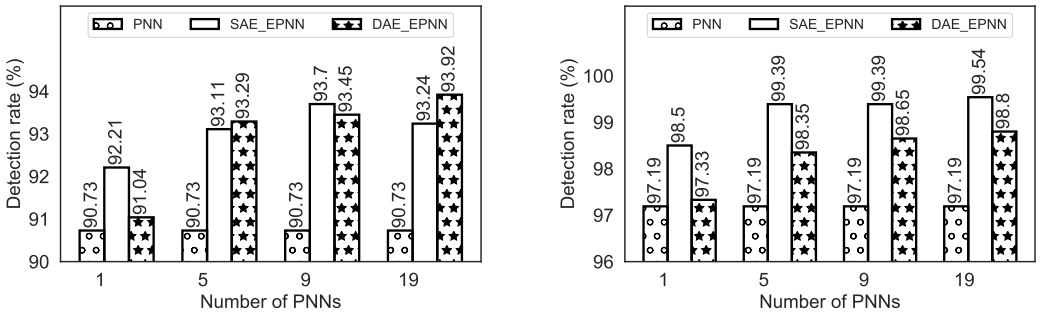


Figure 4.22: Forecasting IoT device status to uncover anomalies

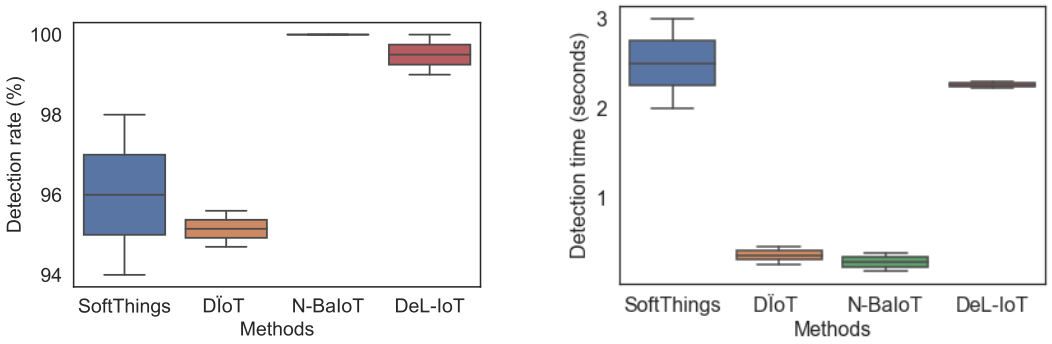


(a) Benchmark (b) Testbed

Figure 4.23: Performance variation of detection rate with respect to the number of PNNs in the 1% imbalanced dataset

4.5 Comparison with Competing Methods

To assess the efficiency of the proposed framework for detecting anomalies in IoT, we compared it with existing methods including SoftThings [35], network-based IoT anomaly detection [39], and DÏoT [102]. SoftThings [35] can detect and mitigate dynamic attacks in IoT using SDN with 98% precision. It employs linear and non-linear Support Vector Machine (SVM) with default hyper-parameters when detecting IoT attacks at the SDN controller in a Mininet simulation environment. The network-based IoT anomaly detection [39] employs the deep autoencoders with four hidden layers to detect anomalies in IoT. The system has optimized hyperparameters such as learning rate, number of epochs, anomaly threshold,



(a) Detection rate

(b) Detection time

Figure 4.24: Comparison of detection rate and time with competing methods

and windows size of autoencoders for each IoT device. For instance, in their experiment, the Danmini doorbell, that is one of the IoT devices, is used with the optimized hyperparameters of the autoencoders such as learning rate 0.012, the number of epochs 800, anomaly threshold 0.042, and window size 82. The system is evaluated using testbed datasets which have nine commercial IoT devices and it achieves a 100% detection rate. The DÏoT [102] reports an autonomous self-learning anomaly detection system for IoT. A GRU network with three hidden layers of size 128 neurons each was used in the system. The system provides evidence to detect anomalies with 95.6% detection rate. Although the existing methods have positive performance, the majority of existing methods or systems were evaluated either as testbed data or in simulated environments. Additionally, our framework shows its superiority in terms of the following points: (a) detects anomalies with 99.8% detection rate for testbed and 99.9% detection rate for benchmark datasets, (b) addresses the data imbalance problem by using a deep ensemble learning, (c) demonstrates dynamic flow management in the presence of attacks, (d) increases device uptime, and (e) forecasts IoT device status for anomalies in short and long term. To observe the performance of deep ensemble learning when chosing different number of PNNs in each layer, we found numerous distinctions in detection rate using 1% imbalanced benchmark and testbed datasets to uncover anomalies in IoT. The hyperparameters used in our experiment the explained in Section 4.3. Figure 4.23 illustrates how the deep ensemble

learning of PNNs improves the detection rate to uncover anomalies in IoT. Figure 4.24 shows the comparison of DeL-IoT with competing methods.

4.6 Summary and Limitations

In this chapter, we presented an efficient and comprehensive approach to detect and predict cyberattacks in IoT devices by integrating DL with SDN. First, we introduced deep ensemble DL approaches that addresses the data heterogeneity and data imbalance problems. Second, we propose a novel mechanism for managing dynamic network flow in presence of attacks utilizing SDN. Finally, our proposed approach predicts IoT devices' future status based on the status table generated by individual devices using LSTM. The main limitations of the proposed approach are the computational overhead while training a deep ensemble learning approach. Our testbed was developed by considering a controlled environment with limited resources. In addition, we carried out our experiments in offline mode after labelling datasets, hence, we are concerned about the performance of the proposed approach in online mode and also about the performance of detecting zero-day attacks. Moreover, we primarily employ a pre-trained model, thus, the testing time is very less with high detection accuracy. However, the proposed model mayn't perform as expected in new environments but can be tuned to generalize the model. With consideration of control and availability of resources for large-scale experiments, we employ a single server but deploy many virtual machines. These experiments can be extended to verify scalability with several servers.

5 A 1D-CNN Based Approach to Detect VSI-DDoS Attacks in IoT Applications

This chapter discusses the impact of VSI-DDoS attacks in applications and presents a 1D-CNN-based deep learning approach to detect VSI-DDoS attacks in IoT applications. The remainder of the chapter is organized as follows. Section 5.2 briefly reviews existing works. Section 5.3 describes our proposed deep learning approach. Section 5.4 presents the experimental results and the analysis of our proposed deep and machine learning models on the testbed and benchmark datasets. Finally, we summarize our findings and suggest possible directions for future work in Section 5.5.

5.1 Introduction

The rise of the Internet of Things (IoT) increases the daily use of devices and applications for making life more comfortable. IoT has already transformed several domains by offering a wide range of services, e.g., creating an intelligent smart grid, developing smart car parking, and personal health monitoring. IoT devices and cloud-deployed applications are vital to achieving particular objectives, such as healthcare monitoring. These broad-spectrum uses of IoT devices and applications open up multiple security issues to hinder legitimate services to the end-users. The existing IoT applications cannot reach on-demand services with expected Quality of Service (QoS) due to the lack of secure IoT ecosystems. One of the critical challenges facing IoT applications is the degrading of the Quality of Service (QoS) due to multiscale distributed denial of service (DDoS)

attacks. The DDoS attacks have been increasing despite numerous DDoS defense mechanisms deployed at a different level in the IoT ecosystems. For instance, Kaspersky Lab's "DDoS attacks in Q1 2020" reports that the number of DDoS attacks doubled against the previous reporting period, and by 80% against Q1 2019 [109]. One of the primary reasons for the growing number of DDoS attacks is the ever-evolving new types of DDoS attacks that can bypass state-of-the-art defense mechanisms. For instance, one of the most significant DDoS attack trends Radware observed in 2017 was an increase in short-burst attacks, becoming more complex, frequent, and persistent. In addition, 42% of organizations in Radware's investigation experienced this type of DDoS attack in 2017 [110].

Moreover, Huasong et al. [111] presented a new burst of low-rate DDoS attack which is known as very short intermittent (VSI) DDoS attacks. This attack is difficult to detect by existing security defense systems since it mimics the legitimate user's behaviour; however, it significantly degrades the QoS for end-users. To evaluate the impact of VSI-DDoS attacks on the QoS of end-users, they carry out experiments using the RUBBoS benchmark web application. Furthermore, they find that the proposed VSI-DDoS attacks can successfully cause the benchmark website's long-tail latency problem while bypassing the DDoS defense systems. Hence, efficient early detection and prevention of VSI-DDoS attacks remain a challenge to address. We are the first to come up with a deep learning-based solution for detecting VSI-DDoS attacks in IoT applications, to the best of our knowledge. However, there are deep learning methods to detect and prevent classical DDoS attacks [38,39,64,80,112] for IoT ecosystems. Also, existing studies insufficiently evaluate VSI-DDoS attacks' impact on QoS implications in IoT applications. Hence, we propose a 1D-CNN deep learning approach to detect VSI-DDoS attacks in IoT applications. The 1D-CNN combines feature extraction and learning models with a cheaper computational cost that results in high detection accuracy. The main contributions of this paper are as follows:

- We prepared a new VSI-DDoS IoT applications dataset with diverse attack scenarios and made it available for public use to fill the research gap.
- We propose a 1D-CNN deep learning approach to detect VSI-DDoS attacks early for IoT applications.

- The experimental evaluation illustrates the performance of the proposed approach using testbed datasets. We carried out experiments on benchmark datasets as well to compare with baseline models.

5.2 Existing Research

VSI-DDoS is an application-layer attack with lower traceability and higher stealthiness to make the attack smarter and easily bypass the security systems. VSI-DDoS attacks use legitimate HTTP requests for service malfunctioning or QoS degradation in the long run, which is quite challenging to investigate and advantageous for attackers to evade security systems. Suppose such attacks are not detected and resolved at an early stage. In that case, the system may experience different complications, such as long response times that eventually damage the target system's long-term business goal for end-users. However, the system appears to be operating within normal conditions. A few works [111, 113] on the impact of the VSI-DDoS attacks have been presented. Shan et al. [111] recently present the VSI-DDoS attack, which is difficult to detect in existing intrusion detection systems but can significantly degrade the QoS of the legitimate users. Jeman et al. [113] develop an approach that influences synchronization in the botnet used to launch the attacks. They illustrate that even with tiny synchronization inaccuracy under about 90 ms, the attackers can still impact the target system. A number of works [64, 65, 80, 86] have been proposed for DDoS attack detection.

Deep learning approaches have been applied successfully for detecting DDoS attacks and they became popular with significant results. Bambang et al. [64] present an intrusion detection method for the IoT environment using the machine and deep learning approaches with the BoT-IoT dataset [38]. Tarun et al. [80] propose deep learning model and evaluate it using the CICIDS2017 dataset [81] for detecting DDoS attack and, resulting in an accuracy of 97.16%. Shanzeb et al. [86] present DDoS attack detection mechanism for the Software-Defined Networks (SDN) using a deep CNN ensemble framework and achieved 99.45% accuracy using the CICIDS2017 dataset. Recently, Jinyin et al. [65] present a multi-channel CNN (MC-CNN) deep learning approach to detect DDoS attacks

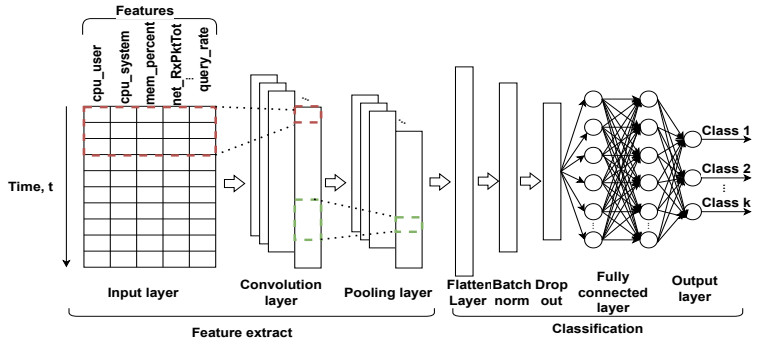


Figure 5.1: A system architecture for 1D CNN-based VSI-DDoS detection

using KDDCUP99 and CICIDS2017 datasets. The experimental results illustrate that MC-CNN detects DDoS attacks with 99.18% accuracy.

Based on this limited literature review, we contend that there are no known solutions to tackle VSI-DDoS attack's problems in IoT applications. Specifically, mission-critical IoT applications have the highest requirements of low latency services with excellent QoS during end-users services.

5.3 System Model

In this section, we first introduce the problem we aim to study, then we elaborate the proposed 1D-CNN-based deep learning for detecting VSI-DDoS attacks in IoT applications to combat QoS degradation of services towards users. A system architecture of 1D CNN-based VSI-DDoS detection is shown in Figure 5.1.

5.3.1 Problem statement

Given the data of n th different time-series with length T, i.e., $x = (x_1, x_2, x_3, \cdots, x_n)^T$, and collected data from multiple IoT applications in the presence and absence of VSI-DDoS attacks. We aim to achieve the following three goals:

- **VSI-DDoS datasets**, i.e., generating a VSI-DDoS IoT applications dataset with diverse attack scenarios and making it public.
- **VSI-DDoS detection**, i.e., detecting VSI-DDoS attacks in IoT applications to alleviate QoS interruption.

- **Experimental analysis**, i.e., carry out exhaustive experimental analysis using both testbed and benchmark datasets with diverse attack scenarios.

5.3.2 VSI-DDoS attacks

Goal and operation. The VSI-DDoS attack is a new form of application-layer short-burst low-rate DDoS attack with the aim to degrade the users QoS. Specifically, IoT applications are mostly latency-sensitive where even a little degradation of performance at the user is not acceptable either for the short or long run. Many service providers such as Google and Amazon have put a lot of effort to lessen the tail latency that creates inconvenience to users. If a user experiences such weariness of service from a service provider then they immediately switch to another provider for preventing the target services from being used. Unlike the classical DDoS attacks that aim to exhaust servers' resources, VSI-DDoS attacks cause transient saturation of resources and increase tail latency of legitimate requests. Because the number of requests is made in a short period of time, like within a millisecond, and it exceeds the server's queue capacity. As a result, the transmission of a legitimate user's request is delayed significantly and the delay is materialized in the TCP retransmission. These repeated retransmissions aggravate the user's experience because of serious packet drops. To alleviate the impact of VSI-DDoS attacks, we generated VSI-DDoS attacks towards IoT applications deployed in the edge cloud as reported in Figure 5.5.

Detection adversity. Requests in VSI-DDoS attacks are similar to legitimate users' requests, but they exhaust a server's queues in milliseconds. Hence, adverse effects happen to the detection systems when they use second-level monitoring systems such as sar[†], vmstat[‡], and top[§]. VSI-DDoS attacks can easily bypass the state-of-art detection systems and make significant impact on the QoS of the legitimate users of the target services in IoT applications with tail latency. Therefore, we propose a deep learning-based approach to detect VSI-DDoS attacks in IoT applications.

[†]https://en.wikipedia.org/wiki/Sar_(Unix)
[‡]https://linux.die.net/man/8/vmstat
[§]https://en.wikipedia.org/wiki/Top_(software)

5.3.3 Proposed CNN-based detection model

Convolutional Neural Network (CNN), inspired by the visual cortex of animals, is widely used for object recognition tasks [65]. As a deep learning architecture, CNN is proposed to minimize the data preprocessing requirements. The most powerful part of CNN is the learning feature hierarchies from a large amount of unlabeled data. Thus, CNN are quite promising for applications in network attack detection [67]. The basic structure of CNN is composed of input and output layers and multiple hidden layers which include convolution layer, pooling layer, and fully-connected layer [114].

Convolution layer.

The convolution layers are the core of the CNN and useful for extracting dominant features which are rotational and positional invariants, thus maintaining the process of effectively training the model. The preprocessing required in a CNN is much lower as compared to other classification algorithms where convolution and pooling layers of the CNN perform a role as feature extraction. Typically, the previous layers of convolution layers map features are convolved with multiple convolutional filters. The output of the convolution operators are added by a bias and put through the activation function to form the feature map for the next layer. The convolution is a linear operation and thus limits the ability to learn complex nonlinear behavior of the input. In order to introduce nonlinearity, the intermediate feature maps of a convolution layer are put through an activation function to form the formal feature maps [115].

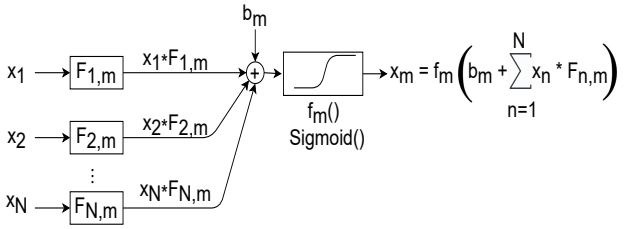


Figure 5.2: Typical architecture of a 1D-CNN unit cell.

The basic cell of a 1D-CNN layer is given in Figure 5.2. A hidden layer output

vector $\overline{x}$ of a 1D-CNN cell consists of M hidden units x_m.

$$\overline{x} = [x_1, x_2, x_m, ...x_M]^T \tag{5.1}$$

where an independent unit is an outcome of a nonlinear transformation applied on a transformation of the N differently filtered input features x. f_m represents the activation function that performs the nonlinear transformation, $F_{n,m}$ are the coefficients of the N filters and b_m is the bias.

$$x_m^l = f_m \left(b_m^l + \sum_{n=1}^{N} x_n^{l-1} * F_{n,m}^l \right) \tag{5.2}$$

where x_m^l is the output in layer l, b_m^l is the bias in layer l, x_n^{l-1} is the input of the filter n in layer $l-1$, $F_{n,m}^l$ is the m^{th} neuron of the filter n in layer l. The output size of the feature maps is calculated as:

$$O = \frac{I - F + P_{start} + P_{end}}{S} + 1 \tag{5.3}$$

where I is the length of the input, F is the length of the filter, P is the amount of zero padding, S is the stride, and O is the output size of the feature map. For instance, for the testbed dataset of this study, the I=9, the input layer convolved with a 3 filter with stride 1, $O = 7$. Each filter contains its own weight with the defined kernel size, considering the length of the input matrix [116]. In the convolution layer of the 1D-CNN, we use 3 filters to extract 7 handy different features for the testbed dataset and we use a sigmoid activation function for the testbed dataset and a relu activation function for the benchmark dataset. The sigmoid and relu activation functions are defined as in Equation (5.4) and Equation (5.5), respectively.

$$f_m = \frac{1}{1 + e^{-\hat{x}_n}} \tag{5.4}$$

and

$$f_m = max(0, \hat{x}_n) \tag{5.5}$$

where e is Euler's number, $\hat{x}_n$ is expressed as:

$$\hat{x}_n = b_m^l + \sum_{n=1}^{N} x_n^{l-1} * F_{n,m}^l \tag{5.6}$$

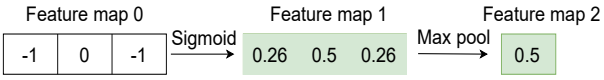


Figure 5.3: Sigmoid and max pool operation - an example.

Pooling.

The pooling layer aims to decrease the computational power required to process the data through dimensionality reduction [117]. This layer is useful for extracting dominant features which are rotational and positional invariants and effective for model training. There are two types of pooling operations: max pooling [118] and average pooling [119].

$$x_m^l = x_m^{l-1} * P^l \tag{5.7}$$

where x_m^{l-1} is the convolved features in layer $l-1$, P^l is the pooling operator in layer l. The output size of the feature maps in a pooling layer is estimated as in Equation (5.8).

$$O_{pool} = \frac{O - P_{pool}}{S_{pool}} + 1 \tag{5.8}$$

where P_{pool} is the length of the pooling operator, S_{pool} is the stride and O is the size of the feature maps of the previous convolution layer. In the pooling layer, we use 1D max pooling with pooling size 2 and stride size 1. The 1D max pooling returns the maximum value that is estimated from the portion under filter. A simple example of pooling operation is depicted in Figure 5.3. After the pooling layer, we use a dropout to avoid the overfitting of 1D-CNN [120].

Detection module.

The detection module consists of two layers: the fully connected layers and the softmax layer. We embed two fully connected layers to perform the classification at the end. In the fully connected layer, neurons from the previous layer are re-shaped as 1D layers in regular networks, and have full connection to all activation in the previous layer estimated as in Equation (5.9).

$$x_j^l = f\left(\sum_{m=1}^{M} x_m^{l-1} W^l + b_j^l\right) \tag{5.9}$$

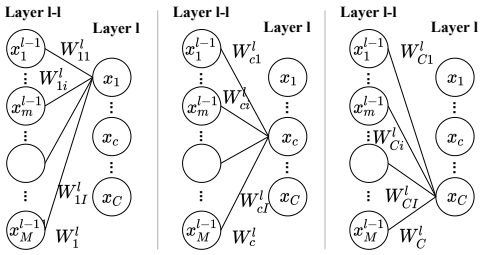


Figure 5.4: Demonstration of neurons and weights assignment in the detection module.

where x_j^l is the j-th neuron in a fully connected layer and M is the layer size. x_m^{l-1} is a neuron in the previous feature map $l-1$ (whether the feature map is the output of convolution or a pooling operation). W^l is the corresponding weight with x_m^{l-1} between layer $l-1$ and layer l, b_j^l is the bias in the input of x_j^l and $f(.)$ denotes an activation function. We have chosen the $relu$ activation function for our experiments. A demonstration of neurons and weights location in the detection module is shown in Figure 5.4. Next, the high-level features vectors from the full connection layer are fed into the softmax layer, which is defined as in Equation (5.10).

$$\begin{aligned} y_c = [P(prediction = c|x^{l-1}; W_c^l)] &= \frac{e^{W_c^l x^{l-1}}}{\sum_{m=1}^{C} e^{W_m^l x^{l-1}}} \\ x^{l-1} &= [x_1^{l-1}, \cdots x_m^{l-1}, \cdots x_M^{l-1}]^T \\ W_c^l = [W_{c1}^l, \cdots W_{ci}^l, \cdots W_{cI}^l], W_l &= [W_1^l, \cdots W_c^l, \cdots W_C^l]^T \end{aligned} \tag{5.10}$$

Let's assume that l is the softmax layer, y_c is the output probability for a particular class c, and C is the number of classes. Hence, the total probability is given by Equation (5.11).

$$\sum_{c=1}^{C} y_c = 1 \tag{5.11}$$

x_{l-1} is the feature vector of size $M \times 1$, W_c^l is the c-th row weight vector of size $1 \times M$. The detection accuracy of the networks is evaluated by measuring the error between the discrete probability distributions of real classes and predicted classes. The cross-entropy is applied as the objective function to optimize the learning rate and the overall detection accuracy.

5.4 Performance Evaluation

This section reports and explains the intensive experimental results obtained from a testbed and benchmark datasets. We begin with the dataset's description and proceed with experimental results.

5.4.1 Datasets

We evaluate our proposed 1D-CNN deep learning approach using two datasets: (i) testbed data, and (ii) benchmark data. The testbed data is generated by emulating VSI-DDoS attacks when deploying IoT applications in edge cloud.
Testbed data: We use Time Series Benchmark Suite (TSBS)[¶], which simulates data streaming from a set of trucks belonging to a fictional trucking company in IoT applications. As shown in Figure 5.5, our system consists of the IoT applications for a fictional trucking company, an IoT applications database server, bots, and legitimate users.

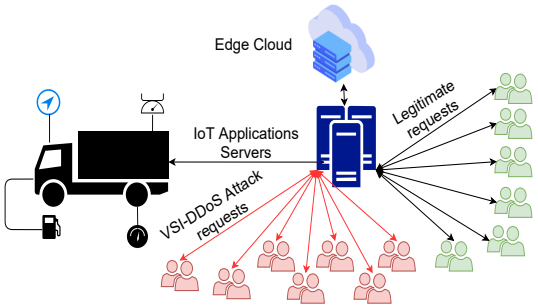


Figure 5.5: Experimental testbed setup and topology

IoT applications server. We have created multiple IoT applications servers with the specification of a 1.8 GHz two core virtual CPU and a 2 GB of Random Access Memory (RAM) for each. This setup was used to generate, collect and process data for our learning models for the detection of VSI-DDoS attacks.

Legitimate users. The behavior of a total of 1000 legitimate users is imitated using the query "last-loc" to get the real-time location of each truck of the TSBS setup. The legitimate users continuously send requests to the IoT applications servers for real-time location information of the trucks.

[¶]https://github.com/timescale/tsbs#appendix-i-query-types

Bot for VSI-DDoS attacks. First, we create 100 queries to retrieve real-time locations from 1000 trucks using the *tsbs_generate_queries* command of the TSBS setup. The execution time of the 100 queries is 50ms. Second, every 2 seconds, we send the created queries to the IoT application servers using the *watch* command. Finally, we increase the number of bots from 2 to 24 as a requirement for generating sets of VSI-DDoS attacks. We observed that most responses for the IoT application's requests are quickly returned to the users within 100ms in absence of VSI-DDoS attacks. However, some of the responses are delayed for more than 1 second due to VSI-DDoS attacks and we observe more delayed responses based on cumulative distribution function (cdf) analysis when the number of botnets increases from 2 to 24 as shown in Figure 5.6.

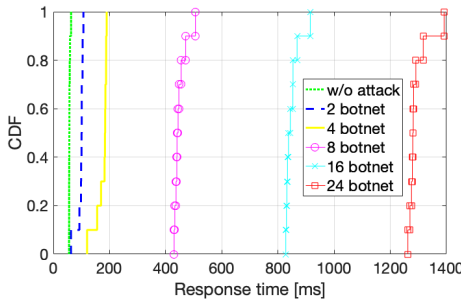


Figure 5.6: Cumulative distribution analysis of response time.

Benchmark data: Since there is a non-availability of benchmark datasets for VSI-DDoS attacks, we use the N-BaIoT [39] dataset for initial empirical validation of our proposed method. This dataset was prepared by using two attack generation tools, i.e., Mirai (scan, ACK flooding, SYN flooding, UDP flooding, UDPplain attacks) and Bashlite (scan, junk, UDP flooding, TCP flooding, COMBO attacks), with 9 commercial IoT devices such as Provision-PT-838 Security Camera, Ecobee thermostat, and Danmini doorbell. There are 5 Bashlite, 5 Mirai, and 1 legitimate datasets, having 115 features in each of them. As baseline models validation, we employ 1% of the imbalanced multiple attacks data of Provision-PT-838 Security Camera to detect DDoS attacks. This experiment uses seven classes, including benign, Mirai (scan, SYN flooding, ACK flooding) and Bashlite (TCP flooding, junk, UDP flooding). The detailed dataset statistics for both benchmark and testbed are given in Table 5.1.

Table 5.1: Dataset statistics - testbed and benchmark.

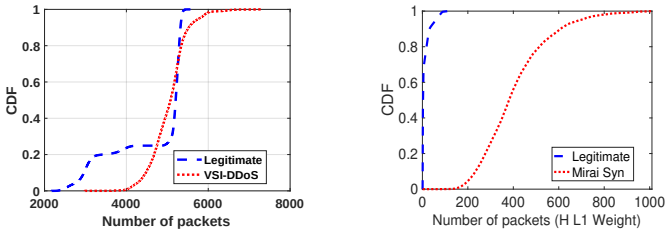
Dataset	number of features	number of legitimate instances	number of anomalous instances
Testbed	9	276868	90503
Benchmark	115	99494	918

5.4.2 Results

In this section, we start with the characterization of data and report experimental results.

Characterization of data.

To observe the behaviour of both testbed and benchmark datasets, we estimate the cumulative distribution function for legitimate and attack instances shown in Figure 5.7. In the benchmark dataset, the distribution of legitimate instances differs from the distribution of the attack instances as clearly illustrated in Figure 5.7 (b). However, in the testbed dataset, we can observe from the Figure 5.7 (a) that the distributions of legitimate instances and attack instances are almost the same due to the stealthy behavior of the VSI-DDoS attacks.



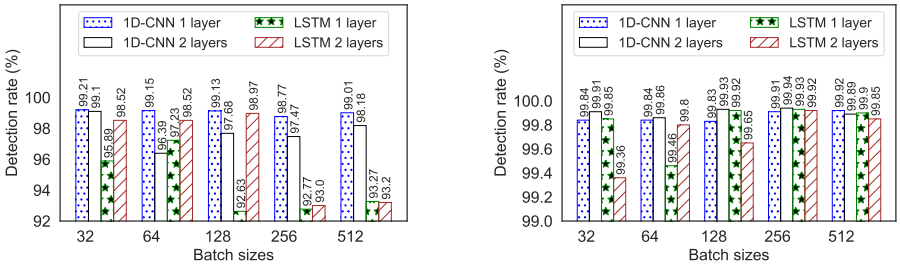
(a) Testbed data (b) Benchmark data

Figure 5.7: Data characterization - cdf for legitimate vs. attack instances over features set.

Hyper-parameters tuning.

The hyper-parameters of the CNN and LSTM models are tuned based on the data for configuring optimal parameters. Batch sizes, numbers of layers, learning rates, and types of loss functions are some of the hyper-parameters tuned by employing a grid search algorithm (GSA) to improve the performance of the model. The proposed 1D-CNN-based deep learning model was modified based on these optimal hyper-parameters and the GSA was run again to find a suitable optimizer.

(a) Experiments with different batch sizes and number of layers: The batch size impacts how quickly a model learns and the stability of the learning process. As it plays a vital role in deep learning model performance, we tuned the parameters with respect to our problem [121]. We start with tuning the number of layers and batch sizes at the same time. For the testbed dataset, Figure 5.8 illustrates our empirical evaluation results with optimal batch sizes of 32 for the 1D-CNN and 128 for the LSTM, while the optimal number of layers is 1 for the 1D-CNN and 2 for the LSTM. Besides, the benchmark experiment results is obtained as shown in Figure 5.8 by choosing optimal batch size 256 for both the 1D-CNN and LSTM, while the optimal number of layers is 2 for the 1D-CNN and 1 for the LSTM. As shown in Figure 5.8, we observe that the detection of the VSI-DDoS attacks is a more difficult task than the detection of the classical DDoS attacks in IoT.

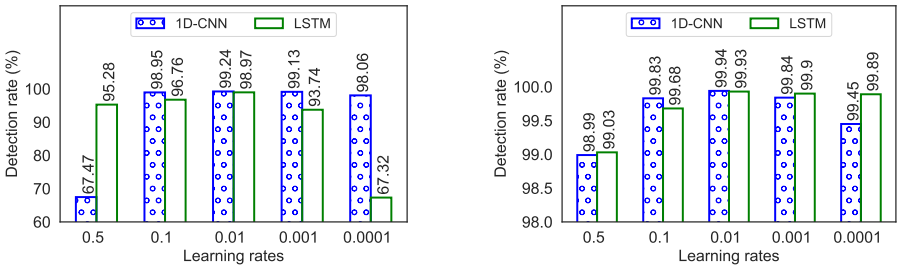


(a) Testbed dataset (b) Benchmark dataset

Figure 5.8: Choosing and analysing optimal batch size and number of layers using GSA.

(b) Experiments with different learning rates: As the learning rate controls how

quickly a model adapts to the problem, it is the most important hyperparameter [82]. Therefore, we perform experiments on 1D-CNN and LSTM models with varied learning rates including 0.5, 0.1, 0.01, 0.001, and 0.0001 for obtaining optimal values. Figure 5.9 illustrates the experimental results based on the different learning rates. According to the results, we found that 0.01 learning rate gives the best detection rate in the testbed and benchmark datasets, respectively. Hence, we run all trails of experiments with the 0.01 learning rate till 100 epochs.



(a) Testbed dataset

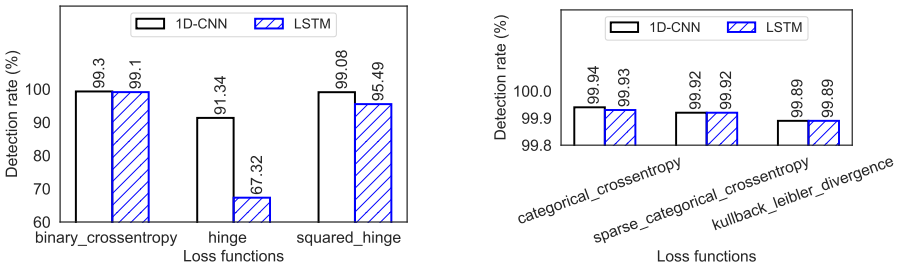
(b) Benchmark dataset

Figure 5.9: Choosing and analysing of optimal learning rate for the testbed and benchmark datasets.

(c) Experiments with different loss functions: As our problem is classification centric, we investigated three binary classification loss functions, *binary_crossentropy*, *hinge*, and *squared_hinge* for the testbed dataset. In addition, we investigated three multi-class classification loss functions, *categorical_crossentropy*, *sparse_categorical_crossentropy* and *kullback_leibler_divergence* for the benchmark dataset. Figure 5.10 shows that *binary_crossentropy* loss function performs best for both CNN and LSTM for the testbed dataset, while the *categorical_crossentropy* loss function performs best for both CNN and LSTM for the benchmark dataset.

5.4.3 Comparison with competing methods

We use LSTM [72], Support Vector Machines (SVM) [64], and Naive Bayes (NB) [122] as our baselines. In this study, we consider the classification problem, thus we chose classification-based deep and machine learning methods as our baseline methods. We use classification accuracy to show the performance compared with



(a) Testbed dataset (b) Benchmark dataset

Figure 5.10: Experimental results - 1 layer 1D-CNN and 2 layers LSTM with different loss functions on the testbed and benchmark datasets.

baselines. Figure 5.11 shows the accuracy of each model. As results of the hyper-parameters tuning for the benchmark dataset, 1 layer 1D-CNN provides the highest detection accuracy of 100%, with 256 batch size, relu activation function, categorical_crossentropy loss function and the Adam optimizer with a learning rate of 0.01. The experiment results show that our 1D-CNN model has the highest accuracy compared with LSTM and the other baseline machine learning methods.

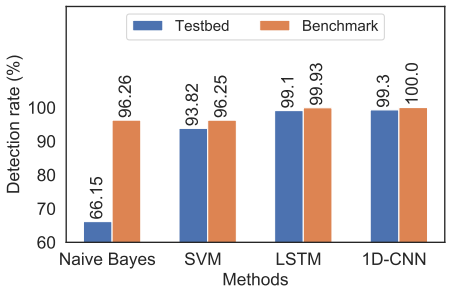


Figure 5.11: Comparison of the proposed method with baseline models

5.5 Summary and Observations

In this chapter, firstly, we generated a new VSI-DDoS attacks IoT application dataset. A detailed description of designing the testbed configuration is presented and collected data by deploying multiple services in presence and absence of attacks. Second, we evaluate the impact of VSI-DDoS attacks on the IoT

application over many different services. Finally, we presented a 1D-CNN deep learning method for detecting VSI-DDoS attacks in IoT applications. The major limitations of the proposed approach are that it can't deal with finite context length, computational inefficiency, and handling time stretching, however, our tasks weren't focused to address these problems rather tuned the model for an efficient detection performance to combat VSI-DDoS attacks in IoT applications.

6 Conclusions

In this research, our main aim is to devise and develop comprehensive approaches to detect, prevent, and predict cyberattacks in IoT devices and applications by integrating DL with SDN. To begin with, we investigate challenges in IoT cyberattacks detection and prevention approaches, and then we find challenges including data heterogeneity, data imbalance, dynamic flow management, and prediction of the future status of the IoT devices. We investigate the proper solution for each challenge, and we find that deep ensemble DL approaches can address the data heterogeneity and data imbalance problem. Thus, firstly we present a novel deep ensemble learning model based framework called DeL-IoT for IoT anomaly detection using SDN primarily to detect anomalies or dynamic attacks for increasing detection performance, switch-level dynamic flow management, and forecasting short and long term device status. We suggest the deep and stacked auto-encoders to extract features for stacking into an ensemble of PNNs learning model for performance improvement while addressing the data imbalance problem. Additionally, we propose a novel mechanism for dynamic flow management in presence of attacks and forecasting device status based on the status table. The system manager can utilize this forecasting features for early action against dynamic attacks from the device to physical infrastructures. We have demonstrated the improved performance in the testbed and benchmark datasets with 99.8% and 99.9% detection rates, respectively, better than the existing methods. Our proposed DeL-IoT method results show that in the 1% imbalanced datasets for both single and multi-class anomalies F1 and MCC measures are around 2%-3% better than a single model. Finally, we conclude that the use of the deep or stacked autoencoder in combination with ensemble PNN improves the performance of IoT anomaly detection in data imbalance problem and also can forecast device status efficiently.

Second, we propose a 1D-CNN deep learning method for VSI-DDoS attacks detection with a vector of time series data. Because 1D-CNN lends benefits for extracting fine-grained features together with learning model with low-cost computation but achieves high accuracy in detecting VSI-DDoS attacks. In addition, we design a new IoT application VSI-DDoS attacks dataset and give a detailed description of designing the testbed configuration and emulated IoT application. We evaluate the impact and detect VSI-DDoS attacks on the IoT application. Using concrete experimental results we show that VSI-DDoS attacks are effective and stealthy towards a target service because they cause QoS issues of the target IoT application while the average usage rate of all the system resources is at a moderate level. We perform experiments on our created testbed and benchmark datasets. The results show that our proposed approach achieved maximum detection rate in contrast to LSTM and other machine learning approaches. We comprehensively investigate the hyper-parameters, and we select the optimal hyper-parameter values to obtain maximum detection accuracy. Based on our exhaustive experimental analysis using testbed and benchmark datasets, the proposed approaches reach the maximum accuracy of 99.3% and 100% which gives improvement by 33.15% and 0.01% detection accuracy in comparison to baseline models, respectively.

7 Future Directions

7.1 Boost difficulties with resource constraints nature in IoT

In recent years, cyberattacks have become sophisticated, advanced and are growing at a very fast pace. DL models self-learn from past cyberattacks by investigating available data and performing user behavior analysis to uncover hidden data patterns to detect cyberattacks. As we aforementioned in this study, a more scalable and efficient approach to detect zero-day and emerging cyberattacks in the IoT ecosystem is DL models. However, the resource constraint nature of IoT devices is one main limitation that makes the challenge the adoption of DL-based IDS in IoT ecosystems. Hence, in this study, we propose to install DL models on the SDN controller to detect cyberattacks in the IoT ecosystem due to the resource limitation of IoT devices. In addition, edge and fog computing are both extensions of cloud computing which is widely used by various organizations. Cisco predicts that by 2021, the organizations will move fifty percent of their workloads and costly computational processes to the cloud or network edge [123]. Thus, in the future, since IoT devices have resource limitations and DL models are computationally expensive, researchers may propose to move the computational processes of deep learning models to edge/fog nodes. This will help security through the firewall, complex models training, and IDSs and give redundancy and failover advantages if a critical site gets hit by a large-scale cyberattack.

7.2 Unknown cyberattacks detection

The unique characteristics of the IoT ecosystem require careful model designing and training for IDS. Therefore, it is crucial to train ML and DL models on a

dataset that is IoT specific and comprises various cyberattacks traffic. There are several datasets publicly available for evaluation of IoT cyberattacks detection approaches and systems including DS2OS traffic traces [37], BoT-IoT [38], N-BaIoT [39], IoT Network Intrusion Dataset [40], ToN-IoT [44], NF-BoT-IoT, and NF-BoT-IoT. However, these datasets mainly include botnets, DoS, DDoS attacks data and do not contain other new types of cyberattacks. Moreover, lack of real-life datasets in the IoT cybersecurity ecosystem. Thus, it is imperative to devise real-life datasets that include emerging cyberattacks in the IoT ecosystem. An essential future research direction is the use of crowd-sourcing methods for generating datasets related to IoT threats and cyberattacks. Rich datasets that include nearly all cyberattack patterns should be generated for training ML and DL models.

7.3 Lack of cross-layer real-time/testbed datasets

The IoT ecosystems consist of several components including devices, applications, proxies or interfaces, and edge servers. The existing IoT benchmark datasets are mostly on IoT devices. Here, cross-layer implies how we collect data together in devices, applications, and infrastructures (physical and virtual). In large-scale IoT systems, each component plays a key role when ensuring secure services and protecting devices during communication in diverse environments. There are no such datasets in the existing literature, we like to create one that can be used to validate methods and systems for detecting and protecting cross-layer attacks.

7.4 Root-cause analysis of cyberattacks

Root-cause detection of attacks in large systems can be a black box and white box perspective. From the black box perspective, graph-based algorithms are commonly used to learn the dependencies between different parts of a system using unsupervised learning to analyze kernel and userspace traces in system logs. The root-cause analysis brings several benefits such as cost-effective mitigation, auto-

mated detection, and mitigation, and recommends specific action. The existing methods didn't implement in cross-layer scenarios against emerging cyberattacks, specifically low-rate DDoS attacks. By applying DL and probabilistic-reasoning approaches, we like to design and develop a method to detect and identify root causes at an early stage that saves enormous cost and also ensures the security of devices and applications.

7.5 Blockchain-enabled SDN controller

Blockchain is an emerging technology that employs cryptography to secure transactions within a network [124]. The blockchain delivers a decentralized database of transactions, of which each node on the network is aware [125]. The decentralized nature of blockchain supports secure distributed computing through the distributed trust concept. IoT devices and SDN controller servers can safely share data using blockchain. Thus, a secure blockchain-enabled architecture of DL-based SDN controllers for IoT networks is still required.

Dedication

This thesis is dedicated to the memory of my beloved mother, Daariimaa Tunjinbaatar. For her endless love, encouragement, and support.

Acknowledgements

I would like to express my sincere gratitude and appreciation to everyone who has been involved in my PhD journey in the past three years. This thesis would not have been possible without the help, support and guidance of a large number of people.

First of all, I would like to express my sincere gratitude to my supervisor, Prof. Youki Kadobayashi, for giving me the opportunity to do my PhD at Cyber Resilience Laboratory in a positive and welcoming environment and his guidance and encouragement. I also wholeheartedly thanks Assistant Prof. Monowar Bhuyan (Umeå University) for his patient guidance, support and motivation to me since the beginning of my doctoral research journey. I am deeply grateful to Assistant Prof. Doudou Fall, Associate Prof. Yuzo Taenaka, Assistant Prof. Md Delwar Hossain, Prof. Yasuhiko Nakashima, Associate Prof. Daisuke Miyamoto (The University of Tokyo), Associate Prof. Khishigjargal Gonchigsumlaa (Mongolian University of Science and Technology), and Prof. Erik Elmroth (Umeå University) for their valuable comments and suggestions. I also thank all the past and present members of the Cyber Resilience laboratory, especially Fatou Williams and Top for their advice and support. I also thank secretaries Kuniko Komano, Haruna Okumura, Yoko Inada, and Ayana Ryono for their kind support and assistance in administration works. I am also really thankful to my friends Khulan, Dulguun, Solongomandakh, and Saruul-Undrakh for their friendship, support and encouragement. Importantly, I would like to express my deepest gratitude to my beloved son and my husband for their endless love, encouragement and support. I would like to express sincere thanks to my parents and younger brothers for their endless love, support, and encouragement.

I sincerely appreciate the financial support received towards my PhD from Mongolia-Japan Higher Engineering Education Development (MJEED) project. I

am also grateful to the Industrial Cyber Security Center for Excellence (ICS-CoE) for supporting my research. Additional support was provided by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by Knut and Alice Wallenberg Foundation, Sweden.

References

[1] R. Ahmad and I. Alsmadi, "Machine learning approaches to IoT security: A systematic literature review," *Internet of Things*, vol. 14, p. 100365, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2542660521000093

[2] S. Al-Sarawi, M. Anbar, R. Abdullah, and A. B. Al Hawari, "Internet of things market analysis forecasts, 2020–2030," in *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, 2020, pp. 449–453.

[3] F. B. Insights. IoT Market Size, Share COVID-19 Impact Analysis, By Component (Platform, Solution Services), By End-Use Industry (BFSI, Retail, Government, Healthcare, Manufacturing, Agriculture, Sustainable Energy, Transportation, IT Telecom, Others), and Regional Forecast, 2021-2028, Last Accessed: Nov 12, 2021. [Online]. Available: https://www.fortunebusinessinsights.com/industry-reports/internet-of-things-iot-market-100307

[4] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep learning for iot big data and streaming analytics: A survey," *IEEE Communications Surveys Tutorials*, vol. 20, no. 4, pp. 2923–2960, 2018.

[5] Y. Yue, S. Li, P. Legg, and F. Li, "Deep learning-based security behaviour analysis in iot environments: A survey," *Security and Communication Networks*, vol. 2021, p. 8873195, Jan 2021. [Online]. Available: https://doi.org/10.1155/2021/8873195

[6] R. Doshi, N. Apthorpe, and N. Feamster, "Machine Learning DDoS Detection for Consumer Internet of Things Devices," in *2018 IEEE Security and*

Privacy Workshops (SPW). San Francisco, CA, USA: IEEE, May 2018, pp. 29–35.

[7] V. Sharma, J. Kim, S. Kwon, I. You, K. Lee, and K. Yim, "A framework for mitigating zero-day attacks in IoT," in *Information Security and Cryptography (CISC-SÍ7)*, Sinchang-Asan, South Korea, 2018, pp. 1–6.

[8] V. Hassija, V. Chamola, V. Saxena, D. Jain, P. Goyal, and B. Sikdar, "A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures," *IEEE Access*, vol. 7, pp. 82 721–82 743, 2019.

[9] M. A. Khan and K. Salah, "Iot security: Review, blockchain solutions, and open challenges," *Future Generation Computer Systems*, vol. 82, pp. 395–411, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X17315765

[10] T. M. Attia, "Challenges and opportunities in the future applications of iot technology," ser. 2nd Europe - Middle East - North African Regional Conference of the International Telecommunications Society (ITS): "Leveraging Technologies For Growth", Aswan, Egypt, 18th-21st February, 2019. Calgary: International Telecommunications Society (ITS), 2019. [Online]. Available: http://hdl.handle.net/10419/201752

[11] B. D. Davis, J. C. Mason, and M. Anwar, "Vulnerability Studies and Security Postures of IoT Devices: A Smart Home Case Study," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10 102–10 110, 2020.

[12] E. Bertino and N. Islam, "Botnets and Internet of Things Security," *Computer*, vol. 50, no. 2, pp. 76–79, 2017.

[13] W. Zhou, Y. Jia, A. Peng, Y. Zhang, and P. Liu, "The Effect of IoT New Features on Security and Privacy: New Threats, Existing Solutions, and Challenges Yet to Be Solved," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1606–1616, 2019.

[14] M. R. Shahid, "Deep Learning for Internet of Things (IoT) Network Security," Ph.D. dissertation, 03 2021.

[15] S. Ahmad, F. Mehmood, and D.-H. Kim, "A DIY Approach for the Design of Mission-Planning Architecture Using Autonomous Task–Object Mapping and the Deployment Model in Mission-Critical IoT Systems," *Sustainability*, vol. 11, no. 13, 2019. [Online]. Available: https://www.mdpi.com/2071-1050/11/13/3647

[16] A. Chowdhury, G. Karmakar, J. Kamruzzaman, A. Jolfaei, and R. Das, "Attacks on Self-Driving Cars and Their Countermeasures: A Survey," *IEEE Access*, vol. 8, pp. 207 308–207 342, 2020.

[17] IDC. IoT Growth Demands Rethink of Long-Term Storage Strategies, Last accessed: Nov 12, 2021. [Online]. Available: https://www.idc.com/getdoc.jsp?containerId=prAP46737220

[18] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, no. 11, p. e00938, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405844018332067

[19] K. Albulayhi, A. A. Smadi, F. T. Sheldon, and R. K. Abercrombie, "IoT Intrusion Detection Taxonomy, Reference Architecture, and Analyses," *Sensors*, vol. 21, no. 19, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/19/6432

[20] S. K. K, S. Sahoo, A. Mahapatra, A. K. Swain, and K. Mahapatra, "Security Enhancements to System on Chip Devices for IoT Perception Layer," in *2017 IEEE International Symposium on Nanoelectronic and Information Systems (iNIS)*, 2017, pp. 151–156.

[21] M. Farooq, M. Waseem, A. Khairi, and P. Mazhar, "A Critical Analysis on the Security Concerns of Internet of Things (IoT)," *International Journal of Computer Applications*, vol. 111, pp. 1–6, 02 2015.

[22] M. M. N. Aboelwafa, K. G. Seddik, M. H. Eldefrawy, Y. Gadallah, and M. Gidlund, "A Machine-Learning-Based Technique for False Data Injection Attacks Detection in Industrial IoT," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8462–8471, 2020.

[23] C.-H. Liao, H.-H. Shuai, and L.-C. Wang, "Eavesdropping prevention for heterogeneous Internet of Things systems," in *2018 15th IEEE Annual Consumer Communications Networking Conference (CCNC)*, 2018, pp. 1–2.

[24] M. Pirretti, S. Zhu, V. Narayanan, P. McDaniel, M. Kandemir, and R. Brooks, "The Sleep Deprivation Attack in Sensor Networks: Analysis and Methods of Defense," *IJDSN*, vol. 2, pp. 267–287, 09 2006.

[25] S. G. Abbas, I. Vaccari, F. Hussain, S. Zahid, U. U. Fayyaz, G. A. Shah, T. Bakhshi, and E. Cambiaso, "Identifying and Mitigating Phishing Attack Threats in IoT Use Cases Using a Threat Modelling Approach," *Sensors*, vol. 21, no. 14, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/14/4816

[26] R. R. Krishna, A. Priyadarshini, A. V. Jha, B. Appasani, A. Srinivasulu, and N. Bizon, "State-of-the-Art Review on IoT Threats and Attacks: Taxonomy, Challenges and Solutions," *Sustainability*, vol. 13, no. 16, 2021. [Online]. Available: https://www.mdpi.com/2071-1050/13/16/9463

[27] C. Kelly, N. Pitropakis, S. McKeown, and C. Lambrinoudakis, "Testing And Hardening IoT Devices Against the Mirai Botnet," in *2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, 2020, pp. 1–8.

[28] G. Kibirige and C. Sanga, "A Survey on Detection of Sinkhole Attack in Wireless Sensor Network," *International Journal of Computer Science and Information Security*, vol. 13, pp. 1–9, 05 2015.

[29] T.-Y. Wu, C.-M. Chen, X. Sun, S. Liu, and C.-W. Lin, "A Countermeasure to SQL Injection Attack for Cloud Environment," *Wireless Personal Communications*, vol. 96, 10 2017.

[30] J. Kumar, B. Rajendran, B. S. Bindhumadhava, and N. S. Chandra Babu, "XML wrapping attack mitigation using positional token," in *2017 International Conference on Public Key Infrastructure and its Applications (PKIA)*, 2017, pp. 36–42.

[31] M. S. Ansari, S. H. Alsamhi, Y. Qiao, Y. Ye, and B. Lee, *Security of Distributed Intelligence in Edge Computing: Threats and Countermeasures*. Cham: Springer International Publishing, 2020, pp. 95–122. [Online]. Available: https://doi.org/10.1007/978-3-030-41110-7_6

[32] S. N. Swamy, D. Jadhav, and N. Kulkarni, "Security threats in the application layer in IOT applications," in *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, 2017, pp. 477–480.

[33] H. A. Abdul-Ghani, D. Konstantas, and M. Mahyoub, "A Comprehensive IoT Attacks Survey based on a Building-blocked Reference Model," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 3, 2018. [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2018.090349

[34] S. Kaur, J. Singh, and N. S. Ghumman, "Network programmability using POX controller," in *ICCCS International Conference on Communication, Computing & Systems, IEEE*, vol. 138, Punjab , India, 2014, pp. 134–138.

[35] S. S. Bhunia and M. Gurusamy, "Dynamic attack detection and mitigation in IoT using SDN," in *2017 27th International Telecommunication Networks and Applications Conference (ITNAC)*, Melbourne, VIC, Australia, Nov 2017, pp. 1–6.

[36] M. Bhuyan, D. K. Bhattacharyya, and J. Kalita, "Towards Generating Real-life Datasets for Network Intrusion Detection," *International Journal of Network Security*, vol. 17, pp. 675–693, 11 2015.

[37] M.-O. Pahl and F.-X. Aubet, "All Eyes on You: Distributed Multi-Dimensional IoT Microservice Anomaly Detection," in *2018 14th International Conference on Network and Service Management (CNSM)*, 2018, pp. 72–80.

[38] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network

forensic analytics: Bot-IoT dataset," *Future Generation Computer Systems*, vol. 100, pp. 779–796, November 2019.

[39] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, and Y. Elovici, "N-BaIoT:Network-based Detection of IoT Botnet Attacks Using Deep Autoencoders," *IEEE Pervasive computing*, vol. 17, no. 3, pp. 12–22, 2018.

[40] H. Kang, D. H. Ahn, G. M. Lee, J. D. Yoo, K. H. Park, and H. K. Kim, "IoT network intrusion dataset," 2019. [Online]. Available: https://dx.doi.org/10.21227/q70p-q449

[41] M. Bhuyan, D. K. Bhattacharyya, and J. Kalita, *Network Traffic Anomaly Detection and Prevention: Concepts, Techniques, and Tools*, 01 2017.

[42] M.-O. Pahl and F.-X. Aubet. 'DS2OS traffic traces, IoT traffic traces gathered in a the DS2OS IoT environment'. (2021, Nov 19). [Online]. Available: https://www.kaggle.com/francoisxa/ds2ostraffictraces

[43] S. Garcia, A. Parmisano, and M. J. Erquiaga, "IoT-23: A labeled dataset with malicious and benign IoT network traffic," Jan. 2020, More details here https://www.stratosphereips.org /datasets-iot23. [Online]. Available: https://doi.org/10.5281/zenodo.4743746

[44] T. Booij, I. Chiscop, E. Meeuwissen, N. Moustafa, and F. den Hartog, "ToN_IoT: The Role of Heterogeneity and the Need for Standardization of Features and Attack Types in IoT Network Intrusion Datasets," *IEEE Internet of Things Journal*, vol. PP, pp. 1–1, 05 2021.

[45] M. Komorowski, D. C. Marshall, J. D. Salciccioli, and Y. Crutain, *Exploratory Data Analysis*. Cham: Springer International Publishing, 2016, pp. 185–203. [Online]. Available: https://doi.org/10.1007/978-3-319-43742-2_15

[46] M. H. Bhuyan, D. K. Bhtaacharyya, and J. K. Kalita, "Network anomaly detection: Methods, systems and tools," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 1, pp. 303–336, First 2014.

[47] M. Hasan, M. M. Islam, M. I. I. Zarif, and M. Hashem, "Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches," *Internet of Things; Engineering Cyber Physical Human Systems*, vol. 7, 2019.

[48] M. Zolanvari, M. A. Teixeira, and R. Jain, "Effect of Imbalanced Datasets on Security of Industrial IoT Using Machine Learning," in *2018 IEEE International Conference on Intelligence and Security Informatics (ISI)*, Miami, FL, USA, Nov 2018, pp. 112–117.

[49] A. Khraisat and A. Alazab, "A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges," *Cybersecurity*, vol. 4, no. 1, p. 18, Mar 2021. [Online]. Available: https://doi.org/10.1186/s42400-021-00077-7

[50] I. H. Sarker, A. S. M. Kayes, S. Badsha, H. Alqahtani, P. Watters, and A. Ng, "Cybersecurity data science: an overview from machine learning perspective," *Journal of Big Data*, vol. 7, no. 1, Jul 2020.

[51] Y. N. Soe, P. I. Santosa, and R. Hartanto, "DDoS Attack Detection Based on Simple ANN with SMOTE for IoT Environment," in *2019 Fourth International Conference on Informatics and Computing (ICIC)*, 2019, pp. 1–5.

[52] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network intrusion detection for iot security based on learning techniques," *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2671–2701, 2019.

[53] R. Al-amri, R. K. Murugesan, M. Man, A. F. Abdulateef, M. A. Al-Sharafi, and A. A. Alkahtani, "A Review of Machine Learning and Deep Learning Techniques for Anomaly Detection in IoT Data," *Applied Sciences*, vol. 11, no. 12, 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/12/5320

[54] M. A. Amanullah, R. A. A. Habeeb, F. H. Nasaruddin, A. Gani, E. Ahmed, A. S. M. Nainar, N. M. Akim, and M. Imran, "Deep learning and big data technologies for IoT security," *Computer Communications*, vol. 151, pp. 495–517, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366419315361

[55] M. A. Al-Garadi, A. Mohamed, A. K. Al-Ali, X. Du, I. Ali, and M. Guizani, "A Survey of Machine and Deep Learning Methods for Internet of Things (IoT) Security," *IEEE Communications Surveys Tutorials*, vol. 22, no. 3, pp. 1646–1685, 2020.

[56] B. B. Zarpelão, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in Internet of Things," *Journal of Network and Computer Applications*, vol. 84, pp. 25–37, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804517300802

[57] L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "IoT Security Techniques Based on Machine Learning: How Do IoT Devices Use AI to Enhance Security?" *IEEE Signal Processing Magazine*, vol. 35, no. 5, pp. 41–49, 2018.

[58] S. Hajiheidari, K. Wakil, M. Badri, and N. J. Navimipour, "Intrusion detection systems in the internet of things: A comprehensive investigation," *Computer Networks*, vol. 160, pp. 165–191, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128619306267

[59] B. Jijo and A. Mohsin Abdulazeez, "Classification Based on Decision Tree Algorithm for Machine Learning," *Journal of Applied Science and Technology Trends*, vol. 2, pp. 20–28, 01 2021.

[60] S. Alharbi, P. Rodriguez, R. Maharaja, P. Iyer, N. Subaschandrabose, and Z. Ye, "Secure the internet of things with challenge response authentication in fog computing," in *2017 IEEE 36th International Performance Computing and Communications Conference (IPCCC)*, 2017, pp. 1–2.

[61] M.-O. Pahl, F.-X. Aubet. DS2OS traffic traces. (2019, Sep 2). [Online]. Available: https://www.kaggle.com/francoisxa/ds2ostraffictraces

[62] P. Chaudhary and B. B. Gupta, "DDoS Detection Framework in Resource Constrained Internet of Things Domain," in *2019 IEEE 8th Global Conference on Consumer Electronics (GCCE)*, 2019, pp. 675–678.

[63] H. Li, K. Ota, and M. Dong, "Learning IoT in Edge: Deep Learning for the Internet of Things with Edge Computing," *IEEE Network*, vol. 32, no. 1, pp. 96–101, 2018.

[64] B. Susilo and R. Sari, "Intrusion Detection in IoT Networks Using Deep Learning Algorithm," *Information*, vol. 11, p. 279, 05 2020.

[65] C. Jinyin, Y. Yang, K. Hu, H. Zheng, and Z. Wang, "DAD-MCNN: DDoS Attack Detection via Multi-channel CNN," in *Proc. 11th International Conference on Machine Learning and Computing*, 02 2019, pp. 484–488.

[66] I. Ullah and Q. H. Mahmoud, "A Deep Learning Based Framework for Cyberattack Detection in IoT Networks," *IEEE Access*, pp. 1–1, 2021.

[67] Y. Xin, L. Kong, Z. Liu, Y. Chen, Y. Li, H. Zhu, M. Gao, H. Hou, and C. Wang, "Machine learning and deep learning methods for cybersecurity," *IEEE Access*, vol. 6, pp. 35 365–35 381, 2018.

[68] H. Hindy, C. Tachtatzis, R. Atkinson, E. Bayne, and X. Bellekens, "MQTT-IoT-IDS2020: MQTT Internet of Things Intrusion Detection Dataset," 2020. [Online]. Available: https://dx.doi.org/10.21227/bhxy-ep04

[69] C.-H. Liao, H.-H. Shuai, and L.-C. Wang, "RNN-Assisted Network Coding for Secure Heterogeneous Internet of Things With Unreliable Storage," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 7608–7622, 2019.

[70] F. Li, A. Shinde, Y. Shi, J. Ye, X.-Y. Li, and W. Song, "System Statistics Learning-Based IoT Security: Feasibility and Suitability," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6396–6403, 2019.

[71] N. Al-Taleb, N. A. Saqib, A. ur Rahman, and S. Dash, "Cyber Threat Intelligence for Secure Smart City," 2020.

[72] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, p. 1735–1780, Nov 1997.

[73] M. Roopak, G. Yun Tian, and J. Chambers, "Deep Learning Models for Cyber Security in IoT Networks," in *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, 2019, pp. 0452–0457.

[74] M. Roopak, G. Y. Tian, and J. Chambers, "An Intrusion Detection System Against DDoS Attacks in IoT Networks," in *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*, 2020, pp. 0562–0567.

[75] F. Bolelli, L. Baraldi, F. Pollastri, and C. Grana, "A Hierarchical Quasi-Recurrent approach to Video Captioning," in *2018 IEEE International Conference on Image Processing, Applications and Systems (IPAS)*, 2018, pp. 162–167.

[76] R.-H. Hwang, M.-C. Peng, V.-L. Nguyen, and Y.-L. Chang, "An LSTM-Based Deep Learning Approach for Classifying Malicious Traffic at the Packet Level," *Applied Sciences*, vol. 9, no. 16, 2019. [Online]. Available: https://www.mdpi.com/2076-3417/9/16/3414

[77] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Computers Security*, vol. 31, no. 3, pp. 357–374, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404811001672

[78] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "Malware traffic classification using convolutional neural network for representation learning," in *2017 International Conference on Information Networking (ICOIN)*, 2017, pp. 712–717.

[79] X. Liang and T. Znati, "A Long Short-Term Memory Enabled Framework for DDoS Detection," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6.

[80] T. D. Diwan, S. Choubey, and H. Hota, "A Novel Hybrıd Approach for Cyber Securıty ın IoT Inetwork usıng Deep Learnıng Technıques," *Int. Journal of Advanced Science and Technology*, vol. 29, no. 6s, pp. 4169 – 4179, Jun. 2020.

[81] I. Sharafaldin, A. Habibi L., and A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," 01 2018, pp. 108–116.

[82] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2017, pp. 493–495.

[83] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, p. e4150, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.4150

[84] D. Chakraborty, V. Narayanan, and A. Ghosh, "Integration of deep feature extraction an ensemble learning for outlier detection," *Pattern recognition*, vol. 89, pp. 161–171, May 2019.

[85] R.-H. Hwang, M.-C. Peng, C.-W. Huang, P.-C. Lin, and V.-L. Nguyen, "An Unsupervised Deep Learning Model for Early Network Traffic Anomaly Detection," *IEEE Access*, vol. 8, pp. 1–1, 02 2020.

[86] S. Haider, A. Akhunzada, I. Mustafa, T. B. Patel, A. Fernandez, K. R. Choo, and J. Iqbal, "A Deep CNN Ensemble Framework for Efficient DDoS Attack Detection in Software Defined Networks," *IEEE Access*, vol. 8, pp. 53 972–53 983, 2020.

[87] O. Sagi and L. Rokach, "Ensemble learning: A survey," *WIREs Data Mining and Knowledge Discovery*, vol. 8, no. 4, p. e1249, 2018.

[88] V. Timčenko and S. Gajin, "Machine Learning based Network Anomaly Detection for IoT environments," in *8th International Conference on Information Society and Technology (ICIST 2018)*, Belgrade, Serbia, 03 2018, pp. 196–201.

[89] Global Digital Transformation Research Team at Frost & Sullivan, "IoT Security Market Watch-Key Market Needs and Solution Providers in the IoT Landscape," p. 4, June 2017.

[90] A. H. M. Aman, E. Yadegaridehkordi, Z. S. Attarbashi, R. Hassan, and Y.-J. Park, "A Survey on Trend and Classification of Internet of Things Reviews," *IEEE Access*, vol. 8, pp. 111 763–111 782, 2020.

[91] M. Stoyanova, Y. Nikoloudakis, S. Panagiotakis, E. Pallis, and E. K. Markakis, "A Survey on the Internet of Things (IoT) Forensics: Challenges, Approaches, and Open Issues," *IEEE Communications Surveys Tutorials*, vol. 22, no. 2, pp. 1191–1221, Secondquarter 2020.

[92] I. Farris, T. Taleb, Y. Khettab, and J. Song, "A survey on emerging and NFV security mechanisms for IoT systems," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 812–837, 2018.

[93] C. Krügel, T. Toth, and E. Kirda, "Service Specific Anomaly Detection for Network Intrusion Detection," in *Proceedings of the 2002 ACM Symposium on Applied Computing*, ser. SAC '02, New York, NY, USA, 2002, p. 201–208.

[94] A. Ghosh, "Big Data and Its Utility," *Consult. Ahead*, vol. 10, no. 1, pp. 52–68, 2016.

[95] M. He, A. M. Alba, A. Basta, A. Blenk, and W. Kellerer, "Flexibility in Softwarized Networks: Classifications and Research Challenges," *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2600–2636, thirdquarter 2019.

[96] N. An, H. Ding, J. Yang, R. Au, and T. F. A. Ang, "Deep ensemble learning for Alzheimer's disease classification," *Journal of Biomedical Informatics*, vol. 105, p. 103411, May 2020.

[97] A. Al-Abassi, H. Karimipour, A. Dehghantanha, and R. M. Parizi, "An Ensemble Deep Learning-Based Cyber-Attack Detection in Industrial Control System," *IEEE Access*, vol. 8, pp. 83 965–83 973, 2020.

[98] F. Jia, S. Li, H. Zuo, and J. Shen, "Deep Neural Network Ensemble for the Intelligent Fault Diagnosis of Machines Under Imbalanced Data," *IEEE Access*, vol. 8, pp. 120 974–120 982, 2020.

[99] E. Tsogbaatar, M. H. Bhuyan, Y. Taenaka, D. Fall, K. Gonchigsumlaa, E. Elmroth, and Y. Kadobayashi, "SDN-enabled IoT Anomaly Detection using Ensemble Learning ," in *Proceedings of the 16th International Conference on Artificial Intelligence Applications and Innovations (AIAI)*, Halkidiki, Greece, June 2020, pp. 268–280.

[100] A. Hamza, H. H. Gharakheili, T. A. Benson, and V. Sivaraman, "Detecting Volumetric Attacks on IoT Devices via SDN-Based Monitoring of MUD Activity," in *Proceedings of the 2019 ACM Symposium on SDN Research*, ser. SOSR '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 36–48.

[101] H. Yao, P. Gao, J. Wang, P. Zhang, C. Jiang, and Z. Han, "Capsule Network Assisted IoT Traffic Classification Mechanism for Smart Cities," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 7515–7525, 2019.

[102] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "DIoT: A Federated Self-learning Anomaly Detection System for IoT," in *IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, Dallas, Texas, USA, July 7-9 2019, pp. 756–767.

[103] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008.

[104] O. N. Foundation, "Openflow switch specification," Open Networking Foundation, Report ONF TS-023, 2015.

[105] D. S. Berman, A. L. Buczak, J. S. Chavis, and C. L. Corbett, "A Survey of Deep Learning Methods for Cyber Security," *Information*, vol. 10, 2019.

[106] A. Botta, A. Dainotti, and A. Pescapè, "A tool for the generation of realistic network workload for emerging networking scenarios," *Computer Networks*, vol. 56, no. 15, pp. 3531–3547, 2012.

[107] M. H. Bhuyan and E. Elmroth, "Multi-Scale Low-Rate DDoS Attack Detection Using the Generalized Total Variation Metric," in *17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Orlando, Florida, USA, December 17-20 2018, pp. 1040–1047.

[108] A. Mehran and A. Hojjat, "Enhanced Probabilistic Neural Network with Local Decision Circles: A Robust Classifier," *Integr. Comput.-Aided Eng.*, vol. 17, no. 3, p. 197–210, Aug. 2010.

[109] Kaspersky. DDoS attacks in Q1 2020, Last Accessed: Oct 10, 2021. [Online]. Available: https://securelist.com/ddos-attacks-in-q1-2020/96837/

[110] Cisco. Cisco 2018 Annual Cybersecurity Report, Last Accessed: Oct 10, 2021. [Online]. Available: https://www.cisco.com/c/dam/m/hu_hu/campaigns/security-hub/pdf/acr-2018.pdf

[111] H. Shan, Q. Wang, and Q. Yan, "Very Short Intermittent DDoS Attacks in an Unsaturated System," in *Security and Privacy in Communication Networks*. Springer, 2018, pp. 45–66.

[112] T. Enkhtur, M. Bhuyan, Y. Taenaka, D. Fall, G. Khishigjargal, E. Elmroth, and Y. Kadobayashi, "SDN-Enabled IoT Anomaly Detection Using Ensemble Learning," in *Proc. AIAI*. Springer, 2020, pp. 268–280.

[113] J. Park, M. Mohaisen, D. Nyang, and A. Mohaisen, "Assessing the effectiveness of pulsing denial of service attacks under realistic network synchronization assumptions," *Computer Networks*, vol. 173, p. 107146, 2020.

[114] Y. Wu, D. Wei, and J. Feng, "Network attacks detection methods based on deep learning techniques: A survey," *Sec. and Comm. Net.*, vol. 2020, pp. 1–17, 08 2020.

[115] Z. Tang, Z. Chen, Y. Bao, and H. Li, "Convolutional neural network-based data anomaly detection method using multiple information for structural

health monitoring," *Structural Control and Health Monitoring*, vol. 26, no. 1, p. e2296, 2019.

[116] M. Azizjon, A. Jumabek, and W. Kim, "1D CNN based network intrusion detection with normalization on imbalanced data," in *Proc. Int. Conference on Artificial Intelligence in Information and Communication*, Feb 2020, pp. 218–224.

[117] G. Jiuxiang, W. Zhenhua, K. Jason, M. Lianyang, S. Amir, S. Bing, L. Ting, W. Xingxing, W. Gang, C. Jianfei, and C. Tsuhan, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 77, pp. 354 – 377, 2018.

[118] J. B. Yang, M. N. Nguyen, P. P. San, X. L. Li, and S. Krishnaswamy, "Deep Convolutional Neural Networks on Multichannel Time Series for Human Activity Recognition," in *Proc. 24th IJCAI.* AAAI Press, 2015, p. 3995–4001.

[119] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng, "End-to-end text recognition with convolutional neural networks," in *Proc. 21st ICPR*, 2012, pp. 3304–3308.

[120] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, p. 1929–1958, Jan 2014.

[121] D. Masters and C. Luschi, "Revisiting Small Batch Training for Deep Neural Networks, arXiv:1804.07612," 2018.

[122] R. Sharma, H. K. Kalita, and P. Borah, "Analysis of Machine Learning Techniques Based Intrusion Detection Systems," in *Proc. 3rd International Conference on Advanced Computing, Networking and Informatics*, vol. 44, 06 2015, pp. 485–493.

[123] Cisco. Cisco Annual Internet Report(2018–2023), Last Accessed: Oct 10, 2021. 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.pdf

[124] G. Zyskind, O. Nathan, and A. S. Pentland, "Decentralizing privacy: Using blockchain to protect personal data," in *2015 IEEE Security and Privacy Workshops*, 2015, pp. 180–184.

[125] A. S. M. S. Hosen, S. Singh, P. K. Sharma, U. Ghosh, J. Wang, I.-H. Ra, and G. H. Cho, "Blockchain-based transaction validation protocol for a secure distributed iot network," *IEEE Access*, vol. 8, pp. 117 266–117 277, 2020.

Publication List

Journals

1. **Enkhtur Tsogbaatar**, Monowar H. Bhuyan, Yuzo Taenaka, Doudou Fall, Khishigjargal Gonchigsumlaa, Erik Elmroth, Youki Kadobayashi, "DeL-IoT: A Deep Ensemble Learning Approach to Uncover Anomalies in IoT", *Internet of Things*, Vol. 14, pp. 100391, 2021, Elsevier.

International Conferences

2. **Enkhtur Tsogbaatar**, Monowar H. Bhuyan, Doudou Fall, Yuzo Taenaka, Khishigjargal Gonchigsumlaa, Erik Elmroth, Youki Kadobayashi, "A 1D-CNN Based Deep Learning for Detecting VSI-DDoS Attacks in IoT Applications", In *Proceedings of the 34th International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems (IEA/AIE 2021)*, July 2021.

3. **Enkhtur Tsogbaatar**, Monowar H. Bhuyan, Yuzo Taenaka, Doudou Fall, Khishigjargal Gonchigsumlaa, Erik Elmroth, Youki Kadobayashi, "SDN-enabled IoT Anomaly Detection using Ensemble Learning", In *Proceedings of the 16th International Conference on Artificial Intelligence Applications and Innovations (AIAI)*, June 2020.