Received: 2 January 2024 | Revised: 5 April 2024 | Accepted: 2 May 2024

IET Smart Cities

DOI: 10.1049/smc2.12084

ORIGINAL RESEARCH

Securing smart cities through machine learning: A honeypot-driven approach to attack detection in Internet of Things ecosystems

Yussuf Ahmed | **Kehinde Beyioku** | **Mehdi Yousefi**

College of Computing, Birmingham City University, Birmingham, UK

Correspondence
Yussuf Ahmed, College of Computing, Birmingham City University, Steam House, Birmingham B4 7RQ, UK.
Email: Yussuf.Ahmed@bcu.ac.uk

Funding information
Birmingham City University

Abstract
The rapid increase and adoption of Internet of Things (IoT) devices have introduced unprecedented conveniences into modern life. However, this growth has also ushered in a wave of cyberattacks targeting these often-vulnerable systems. Smart cities, relying on interconnected sensors, are particularly susceptible to attacks due to the expanded entry points created by these devices. A security breach in such systems can compromise personal data and disrupt entire ecosystems. Traditional security measures are inadequate against the evolving sophistication of cyberattacks. The authors aim to address these challenges by leveraging honeypot data and machine learning to enhance IoT security. The research focuses on three objectives: identifying datasets from IoT-targeted honeypots, evaluating machine learning algorithms for threat detection, and proposing comprehensive security solutions. Real-world cyber-attack datasets from diverse honeypots simulating IoT devices are analysed using various machine learning and neural network algorithms. Results demonstrate significant improvement in cyber-attack detection and mitigation when integrating honeypot data into IoT security frameworks. The authors advance knowledge and provides practical insights for implementing robust security measures in diverse IoT applications, filling a crucial research gap.

KEYWORDS
artificial intelligence, computer network security, data analytics and machine learning, data structures, information security and privacy, IoT and mobile communications, networks and telematics, smart cities

1 | INTRODUCTION

Internet of Things (IoT) can be defined in many ways, but it simply refers to the connectivity of everything around us (watches, cars, houses, cities, etc.) to the internet with some level of intelligence available to these things [1]. It is a term for basic internet technologies that link low-power gadgets such as sensors and actuators (things), and it involves both specialised technology and millions of devices. In comparison to a few years ago, the IoT technology has significantly influenced the daily lives of many people, and this is because of its broad adoption not just by businesses but also by individuals. The adoption rate is exponential, and according to a report [2], it is predicted there will be 29.42 billion linked gadgets by 2030, representing a growth of almost double the installed base of IoT in 2020.

The proliferation of IoT usage in ecosystems such as Smart Cities brings numerous advantages. However, it also introduces significant concerns and challenges, including susceptibility to cyberattacks, compromising user privacy, and device hijacking, among others. In a Smart City setup, the infrastructure heavily relies on network connectivity to gather crucial data from sensors, some of which are linked to critical safety systems [3]. Hackers exploit this connectivity and the multitude of sensors to gain unauthorised access, raising concerns about various vulnerabilities inherent in Smart City features. Components such as smart streetlights and traffic systems are prime targets for threat actors, often utilised in Denial of Service (DoS) attacks [4]. These vulnerabilities persistently increase due to underlying technological factors and architecture flaws. The motivations behind cybercriminal actions may vary, but their impact affects numerous individuals reliant on the affected services within these cities, causing disruptions to their lives. Detecting these threats and mitigating their impact on the population is an urgent priority. This study delves into how combining honeypot data with machine learning techniques can enhance the detection of cyberattacks within IoT ecosystems.

1.1 | Background and rationale

Some of the most common attacks on the smart city and IoT ecosystems include Distributed Denial-of-Service (DDoS), ransomware, device exploitation, data manipulations, Mirai Botnet, Port scanning and brute-force attacks. DDoS is a technique where attackers flood the smart city network infrastructure or IoT ecosystems with a massive volume of traffic, overwhelming it and causing services to become unavailable to users [5]. Ransomware is Malicious software that is deployed to encrypt critical systems or data within the smart city infrastructure, with attackers demanding a ransom for decryption [6]. Ransomware has become the preferred choice for cybercriminals due to the return on investment [7]. Devices are exploited due to IoT vulnerabilities. Many smart city applications depend on IoT devices, which can have vulnerabilities due to poor security practices, such as default passwords or unpatched software. Attackers can exploit these vulnerabilities to gain control over devices and launch attacks on other parts of the network [8]. Manipulating data in smart city systems can lead to misinformation, confusion, or even dangerous outcomes [9]. For instance, altering traffic signal data could cause accidents or gridlock, while tampering with healthcare records could endanger lives. Mirai Botnet is a malware that exploits IoT devices. The malware exploits default credentials, forming botnets to launch massive DDoS attacks, revealing IoT security vulnerabilities [10]. Port Scanning is a technique used to identify open ports on a target system [11]. Brute force is a technique used to crack passwords or encryption keys by systematically guessing all possible combinations until the correct one is found [12].

In September 2016, one of the most significant DDoS attacks on record caused temporary outages at OVH, Krebs on Security, and Dyn. Dyn, a crucial controller of a significant segment of the Internet Domain Name System (DNS) infrastructure, was affected. This attack disrupted numerous popular services and websites, including Twitter, Cable News Network, Netflix, and various major corporations across Europe and the USA for a substantial portion of the day. According to Dyn, approximately '100,000 malicious endpoints' were involved in the attack, which was reported to have an unprecedented strength of 1.2 Tbps [13]. The source of this massive traffic surge was attributed to a botnet known as Mirai, which had compromised tens of thousands of IoT devices such as IP cameras, DVRs, printers, and others, as they were the primary targets of the Mirai botnet [14]. Remarkably, the owners of these compromised devices were unaware that their devices were partaking in a DDoS attack of such magnitude during the incident. This incident raises critical concerns about the security of IoT devices and underscores the need to safeguard control, confidentiality, integrity, and availability of these devices. While previous attacks on IoT devices had occurred, none had reached the scale witnessed with the Mirai botnet.

Traditional security methods, including firewalls and intrusion detection systems (IDSs), fall short of addressing the security concerns and challenges that IoT devices present. It is difficult to deploy complete security solutions due to the distributed and heterogeneous nature of these devices, the networks, and the resource limitations of many IoT devices. Additionally, proactive and adaptive ways to identify and reduce risks are necessary due to the increasing sophistication of cyberattacks. Figure 1 shows the predicted growth in the adoption and usage of IoT devices.

According to a report [15], that captures and presents yearly IoT malware volume across the world, the average volume of malware targeting IoT devices for 2021 was approximately 5.2 million, and in 2022, it approximately averaged 9.4 million, which represents over 80% increase from

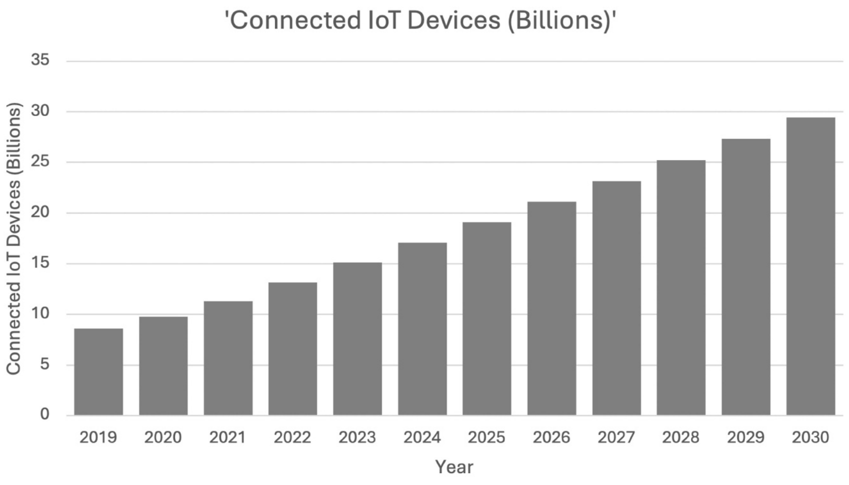


FIGURE 1 2030 predicted IoT installed base [2]. IoT, Internet of Things.

2021. The relatively high numbers year on year clearly show the increased threats posed to IoT devices and, by extension, their ecosystems or application areas. While these figures already communicate an alarming trend for IoT application areas and their security, the first half of 2023, which had been made available by the reports, further shows the need to urgently address IoT security on a large scale. January–June of 2023 averaged 13 million attacks, with the highest-ever volume recorded exceeding 25,000,000 IoT malware volume in just 1 month (May). With the trends of increased IoT malware volumes year on year, this is expected to continue. All these highlight the need for critically exploring security measures that protect IoT devices from cyberattacks.

Honeypots, which are decoy systems designed to lure in cybercriminals, are seen as an efficient way to gather threat intelligence in the broader context of cyber security [16]. They are useful tools for monitoring cyberattacks, but since more IoT devices are being connected to the Internet, they need to become more varied and interactive [17]. By simulating a variety of IoT devices and obtaining feedback from real devices, honeypots can monitor attacks aimed at susceptible IoT devices and capture malware data [18], as it is possible to improve IoT network intrusion detection capabilities by analysing data streams from IoT datasets captured [17]. This implies that research on IoT security and early threat detection can advance with the availability of IoT network traffic data gathered by high-interaction IoT honeypots [19].

The application of honeypots and machine learning algorithms is a promising strategy to improve cyber-attack detection in IoT systems. Security analysts can watch and analyse the behaviour of malicious actors by using honeypots that replicate vulnerabilities and attract attacks. Gathering important information about attack patterns, methods, and indicators of compromise is feasible by carefully placing honeypots within an IoT network. Compared to 2013, when there were just 46 malware detections in IoT devices, there were 7242 by 2017 [20]. Kaspersky Lab IoT honeypot found that there are thousands of attempts to connect to IoT devices within 24 h when they set up the various honeypots on the devices; this demonstrates the significance of honeypot applications for IoT security [21]. The enormous amount of data gathered by honeypots would be nearly impossible to process and analyse manually, but machine learning techniques can successfully analyse and process it. These methods allow for creating intelligent algorithms that can recognise patterns and abnormalities typical of cyberattacks by learning from historical attack data. IoT system administrators can improve their capacity to quickly identify and respond to emerging threats by training machine learning models on honeypot data.

This study will investigate the usage of honeypot data and machine learning algorithms to improve cyber-attack detection in IoT systems. The research aims to construct dependable and scalable models capable of precisely identifying and categorising cyber-attacks by utilising the information obtained from honeypot traffic on IoT devices. The proposed study will address several important issues, including data gathering from honeypots, feature selection, model training and evaluation, and the recommendation for IoT systems to identify cyberattacks.

1.2 | Contributions

There has been significant research on utilising honeypots as a means of cyber security, but there have been limitations regarding applying them for IoT security. These shortcomings are seen when considering (i) Lack of publicly available honeypot data specifically for IoT systems. (ii) Limited research work to evaluate the effectiveness of machine learning and neural networks on IoT datasets. (iii) Lack of practical applications to IoT security and intrusion detection utilising the IoT honeypot dataset. Therefore, this research aims to use honeypot data and machine learning, as well as neural network techniques, to improve the security of IoT systems utilising intrusion detection principles. The main contributions are as follows:

- Identify and review existing IoT honeypot datasets and select the most appropriate for this research.
- Rigorously prepare this dataset and perform 2 feature selections, which will further improve analysis by providing a basis for comparing results for effectiveness.
- Apply machine learning techniques to improve cyber-attack detection using various classification algorithms and evaluate results obtained critically, giving insights on practical applications and limitations.

This study is organised as follows: Section 1 introduces the topic of this research and the rationale behind it, the research questions, and the objectives that will be achieved to answer these questions. Section 2 familiarises the reader with key concepts and the current research relating to generating, analysing, and/or utilising honeypot data to improve the detection of attacks in IoT networks. Section 3 details the methods used to select appropriate datasets, tools, and the experimental steps taken to analyse the dataset and perform machine learning and neural network algorithms on the same. Section 4 includes the results from the analysis, including the exploratory analysis of the datasets used and the initial preparation done on the dataset, and Section 5 presents a discussion of the results in the context of the research questions and the importance of these results. It also describes limitations faced during the research process. Finally, Section 6 includes the conclusion and areas of future work.

2 | RELATED WORK

This section aims to explore the similarities and differences between IoT and traditional networks, highlights the relevance of honeypots to cyber security, and summarises key research that explores how honeypot data can be utilised to enhance the security measures of IoT devices.

2.1 | IoT network versus traditional network

There has been a lot of research and dataset generation on traditional networks, but the same cannot be said of IoT networks. The question may be asked, 'Why can the datasets or traffic of traditional networks not be used for IoT research?' the answer is that while there are similarities, there are significant differences between them, which makes the adoption of one not best suited for the other. Some of these differences and similarities will be reviewed.

Traditional networks are well-known communication methods that have been applied to voice and data transmission. These networks have been the main form of communication for many years and are characterised by circuit-switched technology; they include networks such as narrow-band cellular systems, the Public Switched Telephone Network, and the Integrated Services Digital Network [22]. In terms of data transfer and communication procedures, traditional networks are slightly different from the Internet, which is a global network that links computers and other electronic devices so that they can communicate with one another [23]. However, for comparison with IoT networks, both traditional networks and the Internet will be merged. On the other hand, IoT networks are defined as physical devices, sensors, and controllers that are connected to the Internet so they may exchange data and communicate with one another. These networks are essential to the IoT ecosystem because they enable efficient data transfer and communication between physical and virtual elements [24]. While traditional networks apply to computer systems and electronic devices, IoT networks apply to physical devices, sensors, and anything connected to the Internet.

One of the major similarities is that both rely on ports for communication between devices and services. Additionally, using communication protocols and data-handling techniques is also required for both types of networks [25]. They both have security issues, with IoT networks being especially susceptible to attacks [26], and effective technologies are needed for both types of networks to identify their respective network traffic anomalies [27].

There are quite a few differences between traditional networks and IoT networks. According to ref. [28], the way these two networks are deployed is one of the biggest differences. IoT networks are frequently built on networks with slow processing, limited storage, and low power, while this is not the case with traditional networks, which require a lot more storage and power requirements. IoT devices have their own processing and communication capabilities, extending the traditional internet to a wider range of devices [29], and traditional cryptographic methods are not suited for safeguarding IoT networks due to the limitations of IoT devices which were not present in traditional network devices; this has led to the creation of lightweight cryptographic algorithms and protocols [30], and these algorithms must be implemented in such a way that it does not cause a decline in network performance while tackling unauthorised access and data-security threats vulnerabilities [31]. Additionally, a significant number of network traffic features in IoT networks are different from those in traditional networks, making the development of new traffic models a requirement for improving IoT network security [32].

2.2 | Relevance of honeypots to cyber security

Monitoring malicious traffic on IoT and quickly recognising attacks are crucial for protecting IoT devices. Malicious samples of the IoT must be gathered to analyse malicious behaviour, data characteristics, and attack characteristics. The most efficient techniques for catching malicious requests and gathering examples of malicious behaviour are honeypots. To capture attack behaviour, analyse the tools and methods used by the attackers, and deduce their motives; the honeypot arranges certain hosts, network services, or information to elicit the attack. Honeypot can improve the security protection capability of the actual system and assist the security team in understanding the security dangers clearly [33].

Honeypots have advantages and drawbacks, but they are an effective tool for cyberattack detection and provide early warning to security professionals and researchers. The study of relevant work demonstrates that a honeypot is a very effective detection tool researchers can employ in numerous contexts for cyber defence. Its most crucial feature is the capacity to provide a real-time defence system and to detect 0-day attacks, along with information about them, the tools they use, the methods they utilise, and the way they attack. Because of its effectiveness, attackers always attempt to avoid the honeypot path because they know they will be discovered [21]. Honeypots were found to be helpful in analysing malicious traffic that takes advantage of known and unknown IoT vulnerabilities. Honeypots simulate interactions between a device and an attacker. It gathers enough information for successful analysis and to prevent attacks [34]. According to another study [35], honeypot intrusion technology is the combination of honeypot technology and electronic forensics technology (SIEM logs) applied to the IDS to improve the standard of intrusion detection, decrease the amount of useful information that is missing, and gather useful data from the attacker such as the IP source of the hacker.

The paper concluded that honeypot intrusion technology has not yet developed into a more mature, systematic, or standardised system, but it has become an effective method of combating cyber-crime and has a significant research value in the field of network security. In the paper [36], they proposed a model where the honeypot was designed as the IDS for cyber-physical systems; the honeypot will keep track of all the traffic that has been sent to the cyber-physical system. Security guidelines will be set up to detect intrusions, while the honeypot continuously monitors all packets, and when an intrusion or malicious activity occurs, it recognises the activity as harmful and halts the process. Honeypots can record comprehensive information about the attackers and their strategies; thus, it was used as an IDS. This enables the user to continuously update their IDS to protect themselves from malicious threats and actions. The paper identifies the limitation of the model as it can capture intrusion for only one system per time [36]. In the implementation of our approach, the designed models have the potential to capture intrusions from multiple devices simultaneously.

In the paper [37], two types of honeypots, Kfsensor and honeyd, were used as IDSs on Windows and Linux operating systems, respectively. After the analysis of both honeypots, the proposed system was found to be suitable for network security as a level of defence to spot harmful activity whenever it happens on the network. Our study proposes that similar models can be applied to IoT devices and networks. There is currently limited research on honeypots for IoT environments, while many are built for conventional computers and mobile devices and few are for IoT platforms [38]. Thus, this study aims to add to the body of knowledge on improving network security with honeypots specifically for IoT devices.

2.3 | IoT honeypot research studies

There have been a few research studies on using honeypot data to improve the security of IoT devices using machine learning, but most of the research in this field has only proposed honeypot models specifically for IoT devices. Most of these research works do not provide datasets as the authors in a study [39] captured that there are limited available network traffic datasets gathered by honeypot systems, along with analyses and insights into these data. A few that have the dataset provided have not implemented machine learning techniques to improve the security posture of IoT networks, thus, giving room for further research which this study is focused on.

The authors in ref. [40] discussed a major limitation of traditional low-interaction honeypots for IoT as being easily detectable, prompting a need for more advanced solutions. The paper introduces the SIPHON architecture, a Scalable High-Interaction Honeypot platform for IoT. SIPHON uses physical IoT devices connected via ‘wormholes’ distributed globally, allowing a few devices (less than 10) to appear as many across different IP addresses. In a test involving 39 wormholes across 16 cities in 9 countries, less than 10 devices appeared as 85 IoT devices online, attracting considerable traffic over 2 months. This experiment revealed variances in traffic by city, over 400 brute-force login attempts with 11 successes, and even attempts using credentials from the Mirai malware. While the study analyses its results and introduces a significant concept in the application of honeypots for IoT, it does not generate a dataset nor utilise data obtained for IDS.

Another study [41] investigated the surge in Telnet-based attacks on IoT devices since 2014 and proposed a novel Telnet-emulating IoT honeypot called IoTPOT honeypot to analyse these attacks across various Central Processing Unit (CPU) architectures. The setup comprises a low-interaction front-end linked to an IoTBOX high-interaction back-end, which emulates Telnet services across a variety of IoT device profiles. This honeypot was operational for 39 days, during which time it amassed 43 distinct malware samples and detected at least five DDoS malware families specifically aimed at Telnet-enabled IoT devices, with one evolving to attack nine different CPU types. The study was very detailed in investigating and analysing Telnet-based attacks, but some limitations were identified; attacks targeting services other than Telnet can bypass this honeypot. However, it could be extended for practical applications in IDS.

These studies all prove the presence of malicious attacks on IoT networks but need to be explored further on how these data and IoT honeypots can be integrated with intrusion detection and prevention systems to improve the security of IoT networks.

The vast variety and sheer number of IoT devices make it impractical to manually create low and high-interaction honeypots. Researchers are, therefore, exploring new methods for developing smart IoT honeypots. An interesting study introduced a honeypot for IoT devices called ALLPot that uses machine learning to automatically engage with attackers as though it were a real device. Evaluations show that the system lengthens sessions with attackers and captures more attacks on the IoT network [42].

In a similar study [43], the researchers proposed an ‘intelligent interaction’ honeypot called IoTCandyJar using machine learning to learn IoT device behaviour and emulate them automatically. To increase the number and quality, the authors used a variety of machine learning approaches. The IoTCandyJar chooses from various IoT device responses gathered by scanning the Internet and replies with the one the attacker anticipates in response to the given request. Attackers think the honeypot is their target device if the chosen response matches the anticipated response, and then they send an exploit code. In yet another study [44], they introduce ThingPot, a unique IoT honeypot that mimics application protocols and the entire IoT platform. This was implemented to resemble a Philips Hue smart lighting system; ThingPot identified five types of attacks against smart devices in a 1.5-month deployment. The source code is an open source.

Another study introduced Chameleon, a versatile honeypot tailored for various IoT devices, with the goal of replicating diverse IoT configurations. Upon encountering a designated IoT device, Chameleon redirects incoming network traffic to it while ensuring security protocols are upheld. It meticulously records all exchanges between itself and the target device for later analysis, generating appropriate responses to thwart potential attackers from identifying it. Assessment results reveal that Chameleon adeptly simulates the operations of numerous IoT devices, effectively outmanoeuvring detection attempts by prominent honeypot fingerprinting tools [45]. Another study took this further by proposing a honeypot using smart firmware. They assert that current honeypot systems use gadgets with a certain firmware version installed to track cyberattacks. However, because honeypots regularly get requests aimed at hardware and firmware that differ from their own, they respond with an error message, ending the attack and monitoring. The authors suggest FirmPot, a platform that uses firmware to automatically create intelligent interactive honeypots, as a solution to this issue. The framework uses machine learning to learn the behaviour of embedded applications and features a firmware emulator that is tailored for honeypot generation. The system that keeps the produced honeypots in communication with attackers responds to an attack request with the best of the mimicked responses rather than an error response. Based on the open-source OpenWrt software, the authors experimented with embedded web applications for wireless routers. As a result, their system produced honeypots that imitated the embedded web applications of 10 different CPU architectures and eight vendors [46]. These IoT-specific intelligent interaction honeypot studies have contributed immensely to the understanding and analysis of attacks on IoT networks. However, one area of further research that they have not fully explored is the utilisation of the data and analysis generated to develop IDS specifically tailored to IoT networks.

In this paper [16], the authors posit that limited research is available on using honeypot data to gather insights on IoT networks, particularly for industrial control systems and smart grids. The paper presents results from a 6-month study of smart grid honeypots set up in various locations (USA, Germany, Canada, Brazil, and Singapore) using the Amazon cloud service; the paper explores attack patterns and provides insights useful for improving firewalls and IDSs.

The researcher [47] extended and evaluated a specific honeypot called RIoTPot designed to lure and study attacks exploiting the vulnerability of the IoT and operational technology protocols. They conducted a 3-month study exposing RIoTPot deployed in lab and cloud infrastructures to real-world attacks. They gathered data on nearly 11 million attack events from over 22,000 unique IP addresses. The attacks included various types, such as brute force and poisoning. The study also examined the effectiveness of different interaction levels of the honeypot in attracting specific kinds of attacks. The data collected is made available to other researchers on request for further study. While this study provided a dataset and in-depth analysis, utilising the dataset in IoT networks would have facilitated a more comprehensive evaluation to determine its effectiveness. This paper [48] addresses the security challenges posed by the rapid growth of the IoT, particularly their use in DDoS attacks. The authors maintain that traditional machine-learning-based IDSs rely on extensive data, which is often scarce in IoT networks. To remedy this, they provide a labelled IoT dataset called MedBIoT, featuring both normal and botnet malicious network traffic from a medium-sized network of 83 IoT devices. Data was collected from three significant botnet malwares (Mirai, BashLite, and Torii). The dataset was then tested using binary and multi-class machine learning models, proving its reliability and usefulness for developing and testing botnet detection systems. The MedBIoT dataset is publicly available. It is worth noting that this dataset was not generated using a honeypot. While it contributes immensely to securing IoT networks, it focuses solely on botnet attacks, which is one of the numerous attack types faced by IoT networks.

Another study [49] addresses the security concerns raised by large-scale attacks on IoT devices, such as the Mirai malware attack in 2016. The authors emphasise the usefulness of honeypots in monitoring IoT threats but noted the lack of publicly available research data. To fill this gap, the authors developed a dataset collected from high-interaction IoT honeypots deployed for 1.5 years (2017–2018). The honeypots operated on 40 public IP addresses and directed traffic to 11 real IoT devices. The dataset, created using the Zeek tool, consists of over 81.5 million logs generated from 258,871 Packet Capture (PCAP) files. The data includes attack types and threat intelligence to encourage further research. This study focuses on working on the existing data to improve IoT network security. An area for future work in ref. [49] involves applying the dataset for IDS in IoT networks, which is a major contribution this study aims to achieve. Thus, this paper will use the dataset from ref. [49] to improve IoT network security by applying machine learning techniques proposed for cyber-attack detection in these networks.

Table 1 shows some of the relevant papers that were reviewed, their description and the research gap this study looks to bridge.

3 | RESEARCH EXPERIMENTATION

The aim of this study was to use honeypot data and machine learning techniques to improve the security of IoT systems building on previous studies. The study sets out to achieve this through three objectives which are as follows:

- Identify and review existing IoT honeypot datasets and select the most appropriate for this research.
- Review machine learning techniques and select the most appropriate technique for this research.
- Apply machine learning techniques to improve cyber-attack detection using various classification algorithms and evaluate results obtained.

This section details the approach and experimental tools, methods, and processes that were used to address these objectives.

3.1 | Overview

The methodology for carrying out this research was divided into three distinct segments namely:

- Literature review
- Dataset selection and labelling
- Machine learning application and evaluation

There was an overlap in the process of reviewing various relevant research works and the dataset selection as these studies that presented the datasets found were critically evaluated before the selection was made. The reason for the choice of iTrust, Centre for Research in Cyber Security, Singapore University of Technology and Design's IoT dataset [51] were as follows:

- The IoT honeypots were deployed in the wild for 1.5 years, thereby capturing a broad and diverse range of attacks from across the world [49].
- The honeypot deployment utilised 40 public IP addresses, with traffic forwarded to real IoT devices. Consequently, the captured attacks were not simulated; instead, they comprised genuine attack attempts [49].
- The sheer size of the dataset is significant, with 258,871 PCAP files generated, resulting in over 81.5 million logs. This ample volume of data ensures the capacity to draw accurate conclusions [49].

The dataset selected was in a raw format (.log file), and thus, conversion to a machine learning readable dataset format was required. After conversion, preprocessing and application of three (3) machine learning algorithms and two (2) neural network algorithms were done, and the results were then analysed and discussed. The end-to-end process can be seen in Figure 2.

The tool used for all Python code implementations was Google Colaboratory services (also known as 'Google Colab') because most libraries for machine learning have been pre-installed; thus, it required zero setup and only needed internet access to work. Also, the computational resource requirement which would have been taken by the local machine was transferred to it. The machine learning algorithms used were Naïve Bayes, Decision Tree and K-Nearest Neighbour (KNN), and the neural network algorithms used were Sequential Neural Network (SNN) and Long Short-Term Memory (LSTM). Details as to the choice of algorithm will be discussed in the later part of this section.

Google Colab, also known as 'Colab', is a cloud-based platform designed for prototyping machine learning models using high-performance hardware such as GPUs and TPUs. It provides an interactive development environment through a serverless Jupyter notebook setup [52]. In this study, Google Colab was configured with the maximum hardware accelerator,

TABLE 1 Related works brief description relevant to this study.

Reference	IoT honeypot proposed/deployed	Dataset generated	Machine learning application
[40]	Proposed and deployed a high-interaction honeypot architecture (SIPHON)	No	No
[41]	Proposed and deployed a Telnet-emulating IoT honeypot (IoTPOT)	No	No
[47]	Proposed and deployed RIoTPot	Yes	No
[49]	Deployed high-interaction IoT honeypots for 1.5 years	Yes	No
[50]	Proposed and deployed ThingPot (Mimics entire IoT platform)	No	No
[48]	Non-honeypot setup (83 IoT devices with simulated Botnet attacks)	Yes	Yes
[43]	Proposed an 'intelligent interaction' honeypot (IoTCandyJar)	No	No
[45]	Proposed an adaptive honeypot and deployed to simulate different IoT devices (Chameleon)	No	No
[16]	Deployed smart grid honeypots for 6-month study in varied locations	Yes	No
Our research	Utilising dataset from ref. [49]	Yes	Yes

Abbreviation: IoT, Internet of Things.

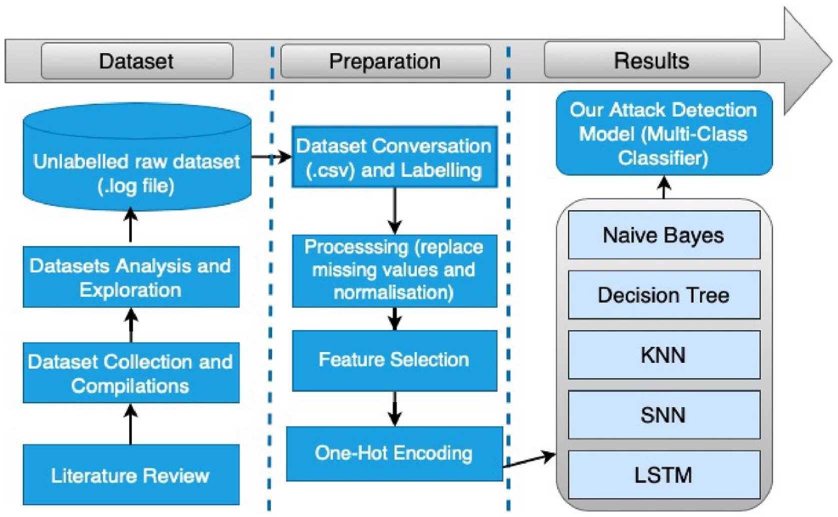

Dataset
Preparation
Results
Unlabelled raw dataset (.log file)
Datasets Analysis and Exploration
Dataset Collection and Compilations
Literature Review
Dataset Conversation (.csv) and Labelling
Processsing (replace missing values and normalisation)
Feature Selection
One-Hot Encoding
Our Attack Detection Model (Multi-Class Classifier)
Naive Bayes
Decision Tree
KNN
SNN
LSTM

FIGURE 2 Research methodology architecture.

a Tensor Processing Unit, and utilised 32 GB of RAM. Further details on dataset preprocessing are elaborated later in this chapter. The algorithms employed included decision tree, Naïve Bayes, and KNN classifiers from the scikit-learn (sklearn) library, as well as SNN and LSTM classifiers from the TensorFlow. Keras library. The SNN model was trained with parameters including 25 epochs, a batch size of 25, and a validation split of 0.2. The LSTM model was trained with parameters including 50 epochs, a batch size of 32, and a verbosity level of 2.

3.2 | Dataset selection

This study was specific to the IoT honeypot dataset, and these were the considerations when selecting a dataset. Thus, critical requirements for any dataset that this study used were that it must be for IoT networks and must have been captured using a honeypot. Two papers [39, 47] both attested that there is a significant unavailability of the public dataset that captures IoT network traffic using honeypots. Thus, an extensive search was carried out for a dataset that met the predefined criteria. Over 200 research papers and article abstracts were carefully reviewed, and while over 50 were relevant to this study, only 8 datasets were found to be relevant to this research work.

Table 2 shows the description of the 8 datasets reviewed; the choice of the dataset was based on the volume of quality raw data generated from real attack traffic, not simulated, and also these were captured over an extended period. Thus, we are confident of multiple attack types from multiple sources.

3.2.1 | Dataset description and exploratory analysis

According to the study [49], the IoT dataset this paper used generated over 81.5 million logs from 258,871 PCAP files. The set-up was to manifest the honeypots on 40 public IP addresses, and traffic to these addresses was forwarded to 11 real IoT devices. Thus, the raw data was examined, and eventually, data from just one of the anonymised IP addresses, which were presented as log files, was only needed. Upon further inspection of the log files, it was realised that different services had different logs and focused on connection log files. 282 log files were randomly selected, and features were extracted to generate a CSV file. The file generated had 108,697 instances and 19 features, as seen in Table 3.

The dataset generated was unlabelled; thus, one of the options for machine learning applications was using an unsupervised learning approach. A major limitation of this approach is that a lack of labelled data makes it difficult to train the model to recognise specific attacks, which can lower detection accuracy [57]. Another limitation is the generation of more false positives, which is very problematic for a security-sensitive domain like IoT [58]. Therefore, unsupervised learning was not selected for these reasons.

TABLE 2 Relevant datasets reviewed.

Reference	Dataset name	Dataset description
[47]	RIoTPot: a dataset of hybrid IoT/OT honeypots	Deployed RIoTPot deployed in lab and cloud infrastructures to real-world attacks for 3 months. 11 million attack events from over 22,000 unique IP addresses captured. Brute-force and poisoning attacks included, and dataset presented as pcap files (iTrust labs)
[49]	Internet of Things dataset	Deployed high-interaction IoT honeypots for 1.5 years presented through 40 IP addresses which forward traffic to 11 real IoT devices. Dataset presented as log files from 258,871 PCAP files resulting in more than 81.5 million logs
[48]	MedBIoT data set	A simulated IoT environment with 83 IoT devices and internet environment, 3 known attack binaries were introduced into the setup and a labelled dataset was the outcome
[16]	Smart grid honeypot data	6-month study of smart grid attacks from honeypots set up in varied locations using the Amazon cloud service. Presented as PCAP files
[53]	Amanzon Web Services (AWS) honeypot attack data	Deployed AWS honeypot and generated Data with 451,581 data points collected from 9:53 PM on 3 March 2013 to 5:55 AM on 8 September 2013
[54]	CyberLab honeynet dataset	Cowrie honeypot was deployed from May 2019 to February 2020 and data collected and presented as Json files as daily datasets
[55]	Canadian Institute of Cybersecurity (CIC) honeynet dataset	Tpot honeypot deployed with security onion as Security Information and Event Management (SIEM) Secure and the PCAP data are released monthly
[56]	IoTID20 dataset	The dataset was generated from another study that had presented it as PCAP files and it was unlabelled. They extracted features, generated a CSV format dataset and then labelled the dataset
Our research	Utilising dataset from ref. [49] (iTrust labs)	Same as described above

Abbreviations: AWS, Amanzon Web Services; IoT, Internet of Things; PCAP, Packet Capture.

Table 4 shows some of the Zeek notices, and Secure Shel (SSH) password guessing was evident. This can imply some brute-forcing activities, as confirmed by the authors, and thus, it was included in the labelled category.

From Table 5, the categorisation of the attacks in the paper is shown. Thus, this dataset has 5 label classes in total, and the names of classes with the number of instances they appear in the dataset are Port Scanning: 88,439, Brute-Force Attack: 15,704, Normal: 1711, DoS Attack: 1585, and Mirai Botnet: 1257.

The description of each attack label can be seen below:

Mirai botnet: Mirai is known for using DNS queries over (UDP) to communicate with its command-and-control servers (C2 servers). This is a common technique used by Mirai to resolve domain names of C2 servers or potential targets [59]. It is also known for attempting to exploit IoT devices through Telnet by using a list of default credentials. A rejected connection on port 23 is a strong indicator of a Mirai scan or attack attempt [60].

Port scanning: A common indicator of port scanning activity is if an originating host attempts to or connects to more than 10 unique destination ports [61]. The authors in ref. [62] posit that rejected connections on well-known ports are often signs of scanning; the authors in ref. [63] posit that rejected connections on specific and unpopular ports are usually port scanning. The line for UDP and port 1900 with connection state S0 indicates scanning for UPnP devices, a known vector for attacks [64].

DoS attack: High packet rates are a classic indicator of flooding attacks [65], and large payloads in UDP packets are often used in amplification attacks [66]. Also, port 69 is commonly used for Trivial File Transfer Protocol, and exploiting standard protocol features is a known DoS tactic [67]. Half-closed connections can be indicative of Finish flood attacks [68].

Brute-force attack: Frequent SSH and FTP login attempts are indicative of brute-force attacks [69], and the Reset Take Over connection state often indicates failed login attempts, which are common in SSH brute-force attacks. Traffic with Session Finished state, destination port 22, and high packet and byte counts as Brute-Force. These metrics can signify

TABLE 4 Zeek notices—notice types [49].

Rank	Type	# Count
1	SSH: password guessing	63,923
2	Capture loss: too much loss	133
3	Secure Socket Layer (SSL): invalid server certificate	66
4	Weird: activity	1

Abbreviation: SSH, Secure Shel.

TABLE 3 Dataset features description.

S/N	Feature	Description
1	Ts	Timestamp—the time at which the network event occurred
2	Uid	Unique ID—A unique identifier for the connection
3	id.orig_h	Origin host—the IP address of the originating host
4	id.orig_p	Origin port—the port number used by the originating host
5	id.resp_h	Response host—the IP address of the host responding to the connection
6	id.resp_p	Response port—the port number used by the responding host
7	Proto	Protocol—the network protocol used (e.g. Transmission Control Protocol [TCP], User Datagram Protocol [UDP])
8	Duration	Duration—the length of time the connection was active
9	orig_bytes	Origin bytes—the number of bytes sent by the originating host
10	resp_bytes	Response bytes—the number of bytes sent by the responding host
11	conn_state	Connection State—the state of the connection (e.g. established, rejected)
12	missed_bytes	Missed bytes—bytes that were not captured in the logs
13	History	History—a string representing the sequence of events for the connection
14	orig_pkts	Origin packets—the number of packets sent by the originating host
15	orig_ip_bytes	Origin IP bytes—the total number of IP bytes sent by the originating host
16	resp_pkts	Response packets—the number of packets sent by the responding host
17	resp_ip_bytes	Response IP bytes—the total number of IP bytes sent by the responding host
18	Service	Service—the application protocol of the connection (e.g. Hypertext Transfer Protocol (HTTP), Secure Shel (SSH)
19	ssh_freq	SSH Frequency—a measure of how frequently SSH protocol is used in the connection

TABLE 5 Network Intrusion Detecton System (NIDS)—alert categorisation summary [49].

Alert type	# Count	Activity	Category	Severity
Emerging Threat (ET)TROJAN ELF/Mirai Variant UA Inbound (Yakuza)	31	Trojan horse (Trojan) activity	Trojan	High
ET TROJAN Executable and linkable Format (ELF)/Mirai Variant User-Agent (UA) Inbound (Muhstik)	1044	Trojan activity	Trojan	High
ET SCAN (Scanning) Mirai Variant User-Agent (Inbound)	525	Attempted-Admin	Scan	High
ET EXPLOIT Linksys E-Series Device Remote Code Execution (RCE) Attempt Outbound	95	Attempted-Admin	Scan	High
ET Simple Network Managment Protocol (SNMP) Attempt to retrieve Cisco Config via TFTP (CISCO-CONFIG-COPY)	562	Policy Violation	SNMP	High
ET SCAN Potential SSH Scan OUTBOUND	21	Attempted-recon	Scan	Medium
ET DOS Possible NTP DDoS Inbound Frequent Un-Authed MON_LIST Requests IMPL 0x03	1045	Attempted-DOS	DOS	Medium
ET DOS Possible Network Time Protocol (NTP) DDoS Inbound Frequent Un-Authed MON_LIST Requests IMPL 0x02	16	Attempted-DOS	DOS	Medium

Abbreviations: SSH, Secure Shel; TFTP, Trivial File Transfer Protocol.

automated SSH login attempts as the SSH server usually listens on Transmission Control Protocol port 22 [70].

Normal: The default label was set as normal, which can also be considered a benign attack in this study. The rationale is that since honeypots are decoy systems, any traffic to them is some sort of attack, even if it is harmless.

3.2.2 | Dataset pre-processing

In addition to dataset labelling, the next step was to pre-process for machine and deep learning model applications. Firstly, the missing values were checked, and *duration*, *orig-bytes*, and *respbytes* each had 6639, *history* had 4438, and *service* had 91,116 missing values, respectively. For the features with numerical values missing, the median was used to fill in the missing values, *history* missing values were filled with the mode value, while *service* was dropped altogether as it had too many missing values. The *sshfreq* feature was generated while defining the labels but had to drop it also as it had 92,540 missing values; although before any features were dropped, the five (5) selected algorithms were performed on the entire 19 features. The disparity between the smallest and largest numbers for numerical features was too large, and thus, all the numerical columns were normalised using *MinMaxScaler*. After normalisation, feature selection was done; this will be discussed in more detail later on. Then, categorical columns were identified, and one-hot encoding was performed to make the dataset suitable for machine learning algorithms. The data was then split into training and testing sets at 70% and 30%, respectively, after which the respective machine learning and neural network algorithms were applied.

3.2.3 | Features selection

Feature selection is the process of extracting a subset of features from the original features. The selected features are chosen based on their relevance to the dataset, while the eliminated features are based on their redundancy [71]. Feature selection is very important to machine learning applications as it eliminates noise in the dataset and improves the performance and speed of the algorithm [72]. Two feature selection methods used on the dataset were Information Gain (IG) and One Rule (OneR) for comparison purposes.

- *IG*: This is an entropy-based feature selection [72] where information is derived from a random variable by observing another random variable. The information gained is then ranked based on relevance from the most relevant to the least relevant
- *OneR*: This is also a ranking-based feature selection that picks a single feature and defines one or more decision rules, which are then used to classify and rank the features [73].

In a study [74], features from OneR produced better results when compared with two other methods inclusive of IG, but that was slightly different in this study. All the features were ranked from highest to lowest, which was used to eliminate features in three (3) iterations. The selected algorithms were performed on the dataset per iteration. The base iteration was done on all features. The first iteration had a total of 13 features. The process for arriving at this was that in addition to the last two features from the OneR ranked features being dropped, the first two from the ranking were also dropped; they were the time stamp and unique ID, as they have nothing to teach the model based on the physical inspection and domain understanding of the dataset. This was similar to IG; thus, the 1st iteration for both methods had the same features. The second iteration had a total of 9 features as the bottom 4 from the first iteration were dropped, and the same process was repeated for the last iteration, which had just 5 features.

These features from each iteration were based on the ranked results obtained from OneR and IG features selection methods. The first iteration was the same for both methods; thus, results for that were the same, but the other iterations presented very diverse results which produced different results and allowed us to compare between iterations and between feature selection methods. The selected features for both OneR and IG used for the three iterations can be seen in Tables 6 and 7, respectively.

3.3 | Machine learning and neural network algorithms application

The dataset label comprises five different network traffic categories; thus, multi-class classification algorithms were required for training and testing the data. This was a strong consideration in selecting the models to apply. The paper considered models from the following classifier families: tree-based, probabilistic-based, and neural networks.

3.3.1 | Naïve Bayes

The Naïve Bayes classifier is a probabilistic classifier that belongs to the family of generative classifiers [75]. It is particularly well-suited for multi-class classification problems and has been effectively applied in IDSs [76]. This study chose Naïve Bayes as its probabilistic classifier because of its simplicity, scalability, and ability to handle large datasets efficiently with less computationally intensive compared to kernel-based classifiers, making it a preferred choice for real-time intrusion detection [75].

TABLE 6 Selected OneR features for the three iterations.

1st iteration (13 feature)	2nd iteration (9 features)	3rd iteration (5 features)
orig_ip_bytes	orig_ip_bytes	orig_ip_bytes
Duration	Duration	duration
id.orig_h	id.orig_h	id.orig_h
resp_ip_bytes	resp_ip_bytes	resp_ip_bytes
History	History	history
resp_bytes	resp_bytes	
orig_bytes	orig_bytes	
resp_pkts	resp_pkts	
orig_pkts	orig_pkts	
conn_state		
id.resp_p		
id.orig_p		
Proto		

Abbreviation: OneR, One Rule.

TABLE 7 Selected Information Gain features for the three iterations.

1st iteration (13 feature)	2nd iteration (9 features)	3rd iteration (5 features)
orig_ip_bytes	orig_ip_bytes	orig_ip_bytes
Duration	resp_ip_bytes	resp_ip_bytes
id.orig_h	History	history
resp_ip_bytes	resp_pkts	resp_pkts
History	conn_state	conn_state
resp_bytes	Duration	
orig_bytes	id.orig_h	
resp_pkts	orig_pkts	
orig_pkts	id.resp_p	
conn_state		
id.resp_p		
id.orig_p		
Proto		

3.3.2 | Decision Tree

The Decision Tree classifier is a rule-based classifier that belongs to the family of tree-based classifiers. It is commonly used for multi-class classification and has been effectively implemented in IDSs. They are usually used for IDS because they are easy to understand and interpret, making them useful for real-time intrusion detection [77]. Decision Trees inherently perform feature selection, improving the IDS accuracy and performance. This attribute makes them efficient and effective for real-time intrusion detection, setting them apart from other tree-based classifiers [78]. It was also chosen because of its ease of implementation and the small computational resource requirement.

3.3.3 | K-NN

The KNN classifier is an instance-based learning algorithm that belongs to the family of lazy learners. KNN is chosen for IDS due to its real-time detection capabilities and flexibility to adapt to new data, setting it apart from other lazy learners in its family. KNN was selected for its flexibility in adapting to new data and its real-time detection capabilities, setting it apart from other lazy learner algorithms [79]. It can adapt to new data without retraining, offering a dynamic approach to intrusion detection [80], and its learning allows for real-time intrusion detection, making it highly responsive [81]. Thus, it is widely used for multi-class classification and has been effectively implemented in IDSs.

3.3.4 | Sequential Neural Network

A SNN is a type of neural network that consists of a stack of neural layers where each layer has one input and one output [82]. In deep learning, the Keras framework is predominately used [83] and was implemented using the same framework in this study. SNNs have shown enhanced performance results in comparison to other machine learning models utilised beforehand in the realm of IDSs as outlined by ref. [84]. The SNNs outperform most Recurrent Neural Networks (RNNs) as they are better at learning long term-dependencies [84]. The authors in ref. [85] sequentially arranged neural networks using a correction-based arrangement and found neural weights to outperform other neural networks typically. Thus making it a very effective model for multi-class classification.

3.3.5 | Long Short-Term Memory

The LSTM network model is a type of RNN that is particularly effective in solving sequence prediction problems. It is also suitable for IDSs, has low computational requirements and is optimised especially for IoT networks. This optimised LSTM approach outperforms other classical learning models with low computational complexity, making it ideal for IoT networks with limited resources [86]. This paper selected the optimised LSTM model for its balance between high prediction performance and low computational requirements, making it a preferred choice for IDS, especially in resource-constrained environments such as IoT networks [87].

3.4 | Evaluation criteria

To evaluate the performance of machine learning and deep learning algorithms in IDS, it is crucial to use appropriate evaluation metrics. This project focuses on four key metrics: Accuracy, Precision, Recall, and F1 Score, which are vital for assessing the effectiveness of IDS in real-world applications [88]. These metrics are defined below:

- *Accuracy*: Measures the proportion of correct identifications, essential for IDS reliability [89].
- *Precision*: Indicates the accuracy of positive identifications, minimising false positives [90].
- *Recall*: Measures the detection of actual attacks, ensuring most attacks are identified [91].
- *F1 score*: Balances Precision and Recall, ideal for a reliable and efficient IDS [56].

In IoT networks, higher accuracy and lower false positives are essential for effective intrusion detection. These metrics ensure that the IDS is both reliable and efficient, making it suitable for real-world applications where an immediate response is required [92].

4 | RESULTS

This section will show results that were obtained in the course of this study. A brief observation from all these results will be shared, but in-depth analysis and discussion will be carried out in the next section.

Several pre-processing steps were taken on the dataset, from dropping columns with large missing values to filling other columns with missing values. Normalisation and one-hot encoding were also performed. Another very important pre-processing step was feature selection, as this allowed us to see the improvement of results across most of the algorithms used in this paper. Tables 8 and 9 present the results for the two ranking-based feature selections.

The choice of IG and OneR features selection techniques was based on both being ranking-based feature selection. Thus, features can be added or dropped for each iteration based on the required features. The differences between the ranks of each feature based on the two different methods utilised also introduce very good dynamics for comparing results in a later section.

The application of the selected machine learning and neural network-based algorithm was done initially on 19 features, which was the baseline, and then it was further done in three iterations of 13, 9, and 5 features, respectively. For each iteration, Naive Bayes, Decision tree, KNN, SNN, and LSTM were performed on the dataset, and accuracy, precision, recall, and f1-score evaluation metrics were used to measure effectiveness. The baseline result can be seen in Figure 3, while subsequent figures in this section will compare the results of each model utilised against the two feature selection methods used.

TABLE 8 One Rule features selection results.

Feature number	Feature name	One Rule
1	uid	0
0	ts	6.44E-05
14	orig_ip_bytes	0.013993155
7	duration	0.017783543
2	id.orig_h	0.022641128
16	resp_ip_bytes	0.025364319
12	history	0.030479502
9	resp_bytes	0.034021491
8	orig_bytes	0.03437109
15	resp_pkts	0.035677486
13	orig_pkts	0.038455877
10	conn_state	0.04070987
5	id.resp_p	0.04381026
3	id.orig_p	0.049302642
6	proto	0.182794215
4	id.resp_h	0.183557813
11	missed_bytes	0.186354604

Abbreviation: OneR, One Rule.

TABLE 9 Information Gain features selection results.

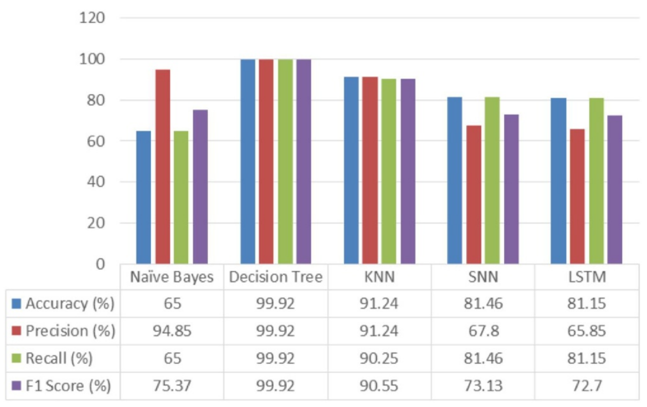
Feature number	Feature name	Info gain
14	orig_ip_bytes	0.562793
16	resp_ip_bytes	0.556161
12	history	0.549013
15	resp_pkts	0.524427
10	conn_state	0.503636
7	duration	0.49139
2	id.orig_h	0.478732
13	orig_pkts	0.478225
5	id.resp_p	0.478137
9	resp_bytes	0.454957
8	orig_bytes	0.452569
3	id.orig_p	0.300413
0	ts	0.215093
6	proto	0.084361
4	id.resp_h	0.041635
11	missed_bytes	0.002255
1	uid	0

From the baseline model performance, it is evident the Decision tree has a near-perfect performance across all evaluation criteria while Naïve Bayes had the worst performance; each model performance for the three iterations will be focused on to see the overall performance and the impact of feature selection on the dataset to their performance.

Figure 4 shows the improvement of performance in Naïve Bayes with IG across all iterations; although the OneR results show an irregular pattern, there was a sharp decline in the second iteration, and then performance significantly improved for the third. The model was clearly improved compared to the Baseline results, except for the anomaly with the OneR second iteration.

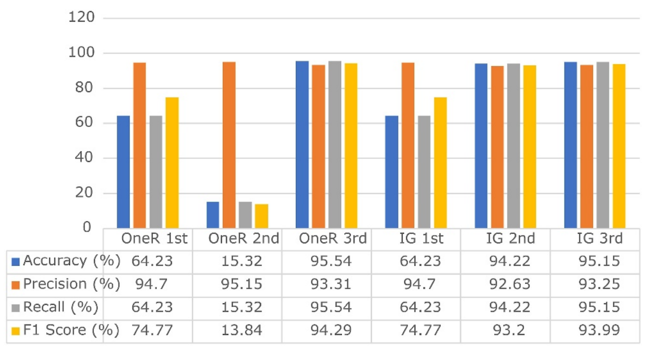
Figure 5 shows a near-perfect Decision tree performance regardless of the iteration or feature selection method used. All the results were over 99%, although there is a slight decline with the selected features across the iterations. This points to the model's ability to deal effectively with noisy data.

KNN shows a very similar pattern with both OneR and IG feature selection, as shown in Figure 6. Its result improved on the second iteration but slightly declined on the third. Overall, its performance was very good across all iterations.



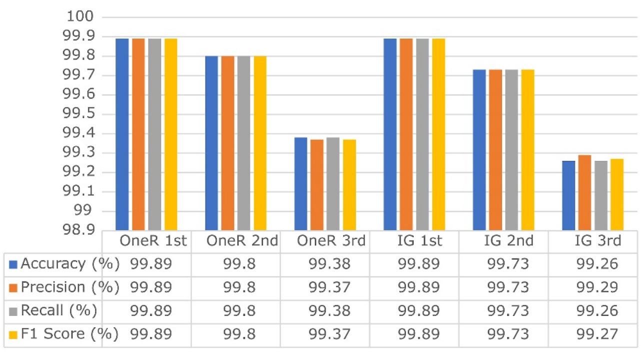
	Naïve Bayes	Decision Tree	KNN	SNN	LSTM
Accuracy (%)	65	99.92	91.24	81.46	81.15
Precision (%)	94.85	99.92	91.24	67.8	65.85
Recall (%)	65	99.92	90.25	81.46	81.15
F1 Score (%)	75.37	99.92	90.55	73.13	72.7

FIGURE 3 Baseline algorithm performance.



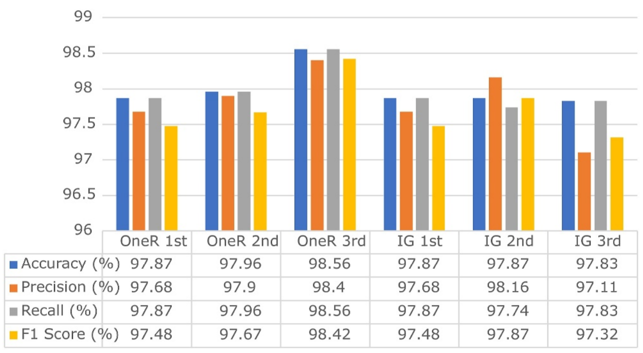
	OneR 1st	OneR 2nd	OneR 3rd	IG 1st	IG 2nd	IG 3rd
Accuracy (%)	64.23	15.32	95.54	64.23	94.22	95.15
Precision (%)	94.7	95.15	93.31	94.7	92.63	93.25
Recall (%)	64.23	15.32	95.54	64.23	94.22	95.15
F1 Score (%)	74.77	13.84	94.29	74.77	93.2	93.99

FIGURE 4 Naïve Bayes performance comparing Information Gain and OneR across the three iterations. OneR, One Rule.



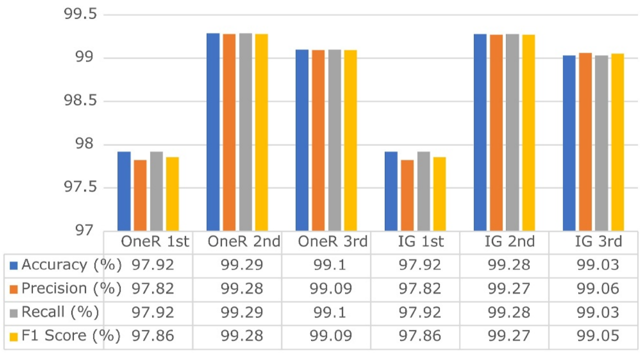
	OneR 1st	OneR 2nd	OneR 3rd	IG 1st	IG 2nd	IG 3rd
🟦 Accuracy (%)	99.89	99.8	99.38	99.89	99.73	99.26
🟧 Precision (%)	99.89	99.8	99.37	99.89	99.73	99.29
⬜ Recall (%)	99.89	99.8	99.38	99.89	99.73	99.26
🟨 F1 Score (%)	99.89	99.8	99.37	99.89	99.73	99.27

FIGURE 5 Decision tree performance comparing Information Gain and OneR across the three iterations. OneR, One Rule.



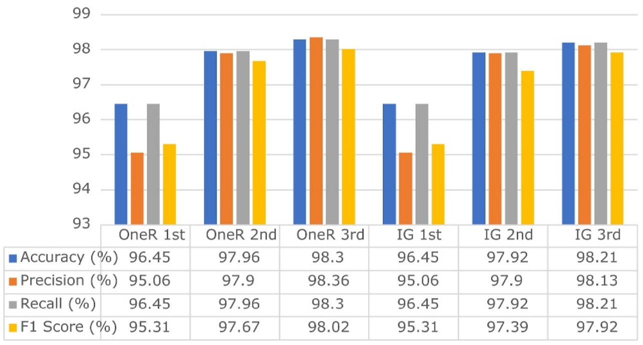
	OneR 1st	OneR 2nd	OneR 3rd	IG 1st	IG 2nd	IG 3rd
🟦 Accuracy (%)	97.87	97.96	98.56	97.87	97.87	97.83
🟧 Precision (%)	97.68	97.9	98.4	97.68	98.16	97.11
⬜ Recall (%)	97.87	97.96	98.56	97.87	97.74	97.83
🟨 F1 Score (%)	97.48	97.67	98.42	97.48	97.87	97.32

FIGURE 7 SNN performance comparing Information Gain and OneR across the three iterations. IG, Information Gain; OneR, One Rule; SNN, Sequential Neural Network.



	OneR 1st	OneR 2nd	OneR 3rd	IG 1st	IG 2nd	IG 3rd
🟦 Accuracy (%)	97.92	99.29	99.1	97.92	99.28	99.03
🟧 Precision (%)	97.82	99.28	99.09	97.82	99.27	99.06
⬜ Recall (%)	97.92	99.29	99.1	97.92	99.28	99.03
🟨 F1 Score (%)	97.86	99.28	99.09	97.86	99.27	99.05

FIGURE 6 KNN performance comparing IG and OneR across the three iterations. IG, Information Gain; KNN, K-Nearest Neighbours; OneR, One Rule.



	OneR 1st	OneR 2nd	OneR 3rd	IG 1st	IG 2nd	IG 3rd
🟦 Accuracy (%)	96.45	97.96	98.3	96.45	97.92	98.21
🟧 Precision (%)	95.06	97.9	98.36	95.06	97.9	98.13
⬜ Recall (%)	96.45	97.96	98.3	96.45	97.92	98.21
🟨 F1 Score (%)	95.31	97.67	98.02	95.31	97.39	97.92

FIGURE 8 LSTM performance comparing Information Gain and OneR across the three iterations. IG, Information Gain; LSTM, Long Short-Term Memory; OneR, One Rule.

SNN accuracy can be seen to improve from iteration to iteration with OneR feature selection, while with IG in Figure 7, it had the same approximate accuracy across all iterations.

LSTM showed very good performance and also improved across all iterations with both OneR and IG as shown in Figure 8.

These results would all be discussed in more details in the next section.

4.1 | Observations

It can be observed that there was a mix of similarities and disparities between the IG and the one-rule feature selection techniques. They both had 3 (*proto*, $id_{r}esp_{b}$, and $missed_{b}ytes$) of the last 5 labels as the same, but only 1 ($orig_{i}p_{b}ytes$) of the top 5 was the same. This clearly demonstrates that the application of these feature selections could have a very different impact on the performance of the models. Naïve Bayes gave the most inconsistent results and also the worst performance among all the models, while the decision tree remained consistent across all the iterations for both feature selection methods. It was also observed that the second iteration with 9 features had the best results for both OneR and IG across all the models, with one exception of Naïve Bayes with OneR.

The algorithm results will be discussed in more detail in Section 5.

5 | DISCUSSION

This study aimed to utilise honeypot data to improve cyber-attack detection in IoT systems. The raw dataset is saved as log files to generate the dataset into a CSV file. The dataset was normalised, missing values were filled in, and feature selection was performed on the dataset to rank the relevance of the features, as seen in Tables 6 and 7. Thirteen (13) features were first used on all the machine learning models, and the neural network models, then nine (9) features were used, and finally, five (5) features were used across all selected models. The performance of the neural network models improved with reduced features, but there was no defined pattern for the machine learning algorithms. The focus of this discussion is on the result of the selected machine learning and neural networks algorithm; while other results in this study are important, they are not necessary for further analysis in demonstrating how the use of honeypot data can improve cyber-attack detection and overall security of IoT networks.

5.1 | Insights from results

It can be observed from the result that Naïve Bayes had the lowest score in all the evaluation criteria across all iterations. In the first iteration of OneR, its precision was high, but recall and accuracy were low. This suggests that while the algorithm correctly identifies attacks when it says there is one, it misses a lot of actual attacks, which could be detrimental to the IoT device. The second iteration with OneR has extremely low accuracy and recall, suggesting that it is not a good fit for IDS detection for IoT network traffic, but this was not the case with IG where it performed well. The third iteration presents the best results, with all metrics returning high values. With the most variable scores, ranging from 15.32% to 95.54% in accuracy, this suggests that it is highly sensitive to the dataset or conditions under which it is applied, but with the right features selected, its performance is more stable and reliable. Overall, Naïve Bayes performance was the most erratic with feature selection, and in real-time scenarios where traffic data input is not clearly defined and pruned, the model will not be ideal for such a scenario.

On the other end, the Decision Tree consistently ranked highest on all metrics across all iterations of both Feature Selection methods. This suggests that the Decision Tree algorithm is very effective at identifying attacks in the given IoT network traffic regardless of what features are available; this implies the ability to perform well even in noisy data. A gradual decline was observed across all metrics as the features were reduced. This suggests that in real time, the decision tree will perform well in identifying attacks on IoT networks. Although, with the remarkably high scores across all metrics, nearly touching 100% in most cases, this could indicate a perfect or near-perfect model and raise concerns about overfitting, as decision trees are known for that.

After the decision tree, KNN performed the second best with high performance across all metrics, iterations, and similar results with both feature selection methods. The best performance was the second iteration, and its worst was the baseline, suggesting that it performs better without noisy data. Overall, this indicates good generalisation and effective detection capabilities, and it seemed to be a bit more balanced and devoid of overfitting, as with the possibility of such with a decision tree.

SNN also consistently performed highly across all metrics but slightly lower than Decision Trees and KNN. The model improves progressively as features are reduced, which suggests it performs better without noisy data. Its high performance also indicates its effectiveness. However, it was a lot more computationally intensive as running a single iteration took far longer, even after the epoch and batch size were reduced to 25 each. This suggests that while it would be a great tool for post-attack analysis, it might not be so effective for real-time applications.

LSTM performance was also similar to SNN, which improved progressively with fewer features. It had lower accuracy in the first iteration but improved in subsequent ones, such as SNN; this could indicate that these neural network models may require more sophisticated setups or hyperparameter tuning. LSTM was also computationally intensive, although it took less time and resources than SNN. It also took more time than KNN and decision trees; thus, the application for real-time detection might be less resource-effective when compared to the machine learning models.

5.2 | Validating the model with MedBIoT dataset

Our models were applied to the MedBIoT dataset to evaluate the performance and compared with the work by ref. [48] for both binary and multi-class classifications. The rationale for using this dataset is that it is targeted at the IoT dataset and is one of the popular datasets researchers use. Machine learning was then applied to the dataset. Three algorithms were used for both binary and multi-class classifications; Decision tree, KNN, and Random Forest (RF). The same four metrics of accuracy, precision, recall, and f1 score were used to evaluate the performance in this study. The same parameter in Section 3 was applied to aid with the comparison.

The study found that Support Vector Machine showed low results across all the metrics, and so it was not included in the results.

Machine learning algorithms were applied to the MedBIoT dataset. RF performed best in the study by ref. [48] as seen in Table 10; while our study did not utilise RF, the other two algorithms were utilised, thus providing a basis for comparison.

As observed in Table 11, both KNN and Decision Tree models from this study outperformed the same algorithms when compared to results from the MedBIoT dataset in Table 10 for the binary classification. The multi-class

TABLE 10 Binary classification results on MedBIoT dataset [48].

Model	Accuracy	Precision	Recall	F1 score
KNN	0.9025	0.9082	0.9025	0.9001
DT	0.9315	0.9448	0.9315	0.9293
RF	0.9532	0.958	0.9532	0.9481

Abbreviations: DT, Decision Tree; KNN, K-Nearest Neighbours; RF, Random Forest.

TABLE 11 Binary classification results on MedBIoT dataset.

Model	Accuracy	Precision	Recall	F1 score
KNN	0.9672	0.9673	0.9672	0.9673
DT	0.9723	0.9723	0.9723	0.9723
Naïve Bayes	0.6023	0.6793	0.6023	0.5373
SNN	0.9598	0.9601	0.9598	0.9598
LSTM	0.893	0.8942	0.893	0.8927

Abbreviations: DT, Decision Tree; KNN, K-Nearest Neighbours; LSTM, Long Short-Term Memory; SNN, Sequential Neural Network.

classification was further done in the study, which was the same classification type the models in this research utilised. Thus, it is imperative to compare the model results further using the multi-classification criteria.

From the results in Table 12, it is evident that RF had the highest performance in comparison to the other two. While our study did not utilise the RF model, the performance results of KNN and Decision tree as seen in Table 13 were higher when compared with Table 12.

These results further show the effectiveness of the models especially the decision tree and highlights demonstrate its ability for efficient intrusion detection.

A lot of interesting conclusions can be drawn from this research regarding the effectiveness of machine learning and neural network techniques in detecting cyberattacks in IoT systems. The implications of these lessons will be highlighted for IoT Network Security:

- *Real-time detection*: Decision Trees and KNN offer the best real-time attack detection based on their high scores across all metrics. SNN and LSTM also offer high performance across all metrics, but the computational requirement and length of processing make them not the best of options for real-time applications.
- *False alarms*: Naïve Bayes, in its first iteration, had high precision but low recall, which means fewer false alarms but more missed detections. Again, Decision Tree and KNN were the best-performing algorithms, with SNN and LSTM also showing high performance across all iterations.
- *Computational cost*: Neural networks (SNN, LSTM) require more computational resources, and thus if integrated with IoT devices, they might not be ideal for IoT devices with limited computational capabilities.
- *Feature selection*: The performance varied significantly in different iterations for Naïve Bayes and slightly for LSTM and SNN, indicating the importance of feature selection in improving performance according to the model, specific types of attacks, or network conditions. The decision tree also demonstrated its effectiveness with or without feature selection, demonstrating its ability to work with noisy data.

In summary, Decision Trees and KNN show the most promise for general real-time use cases for cyber-attack detection in IoT systems. The neural network also showed improvement in performance across all metrics with feature selection and thus could be highly effective where real-time performance is not required. In practical applications, an implementation of an IoT-specific honeypot and decision tree-based IDS placed between the IoT network and the IoT devices can be extremely beneficial in detecting cyber-attacks and, hence, improving security in IoT devices.

From these results, it can be concluded that the application of relevant machine learning algorithms can effectively improve the detection of cyberattacks in IoT systems and, thus, significantly reduce intrusion, especially for critical application areas of IoT systems such as smart cities, smart grids, autonomous vehicles etc.

5.3 | Limitations

While this study offers valuable insights into improving security in Smart Cities through machine learning approaches, a notable limitation is the absence of real-life deployment of the proposed solution on existing IoT devices and networks. This aspect of the work will be explored in future research.

5.4 | Future application in edge devices with federated learning approach

Federated learning is a distributed machine learning approach where models are trained under centralised control on endpoints, end devices, organisations, or individuals without necessitating the sharing of their local datasets. This ensures data privacy throughout the training process. An edge server periodically gathers the trained parameters to construct and enhance a more precise and superior model. Subsequently, the updated model is transmitted back to the edge devices for local training [93].

In practical applications, our models can be deployed on edge devices using a federated learning approach, which could significantly enhance attack detection in IoT networks and devices. Federated learning was not utilised in this study since the models were implemented as a proof of concept. However, there is a plan to deploy them in a real-time IoT network in the future. Additionally, incorporating federated learning approaches on edge devices will be considered as part of future work.

TABLE 12 Multi-class classification results on MedBIoT dataset [48].

Model	Accuracy	Precision	Recall	F1 score
KNN	0.8706	0.8849	0.8706	0.8505
DT	0.9516	0.9584	0.9516	0.9499
RF	0.9766	0.9824	0.9766	0.9657

Abbreviations: DT, Decision Tree; KNN, K-Nearest Neighbours; RF, Random Forest.

TABLE 13 Multi-class classification.

Model	Accuracy	Precision	Recall	F1 score
KNN	0.9492	0.9495	0.9492	0.9492
DT	0.9654	0.9654	0.9654	0.9654
Naïve Bayes	0.6081	0.6339	0.6081	0.5576
SNN	0.9275	0.929	0.9275	0.9275
LSTM	0.8501	0.8569	0.8501	0.8505

Abbreviations: DT, Decision Tree; KNN, K-Nearest Neighbours; LSTM, Long Short-Term Memory; SNN, Sequential Neural Network.

6 | CONCLUSION AND FUTURE WORK

This study makes significant strides in advancing the field of IoT security using honeypot raw data to generate a labelled dataset and machine learning and deep learning algorithms. Machine learning and neural network algorithms were then applied to this dataset. The robustness of this entire research approach is underscored by the promising results, which indicate substantial practical benefits in threat detection capabilities. Moreover, the research findings are not merely academic exercises but have real-world applicability in improving the security posture of IoT deployments.

The decision tree was the best-performing algorithm, and it maintained a consistent level of performance with the highest results ranging from 99.38% to 99.92% across the baseline in the first, second and third iterations with OneR feature selection on the IoT Dataset. The accuracy results ranged from 99.26% to 99.92% with IG feature selection when the same iterations were applied to the dataset. This consistency was maintained and validated with the MedBIoT dataset, achieving results of 97.23% and 96.54% for the binary and multi-class classifications, respectively, demonstrating the model's effectiveness.

While the current study provides valuable insights, there are several avenues for future research. The dataset used was derived from raw data traffic, and acquiring more IoT honeypot-labelled datasets could significantly benefit both research and industry. Extending similar research to practical IoT applications such as healthcare, smart-grid, and smart homes, is another area for future exploration. Additionally, the scalability of the proposed security solutions could be examined to ensure they can be integrated into larger, more complex IoT ecosystems without compromising performance. Finally, the real-time applicability of the findings is an important future consideration; future research could focus on developing real-time threat detection and mitigation systems based on the data and insights generated by this study.

AUTHOR CONTRIBUTIONS

Yussuf Ahmed: Conceptualisation; data curation; formal analysis; funding acquisition; investigation; methodology; project administration; resources; software; supervision; validation; visualisation; writing—original draft; writing—review and editing. **Kehinde Beyioku**: Data curation; formal analysis; investigation; methodology; validation; writing—original draft; writing—review and editing. **Mehdi Yousefi**: Methodology; project administration.

ACKNOWLEDGEMENTS

This research was supported in part by Birmingham City University.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflicts of interest.

DATA AVAILABILITY STATEMENT

Restrictions apply to the availability of the dataset from iTrust Centre for Research in Cyber Security, Singapore University of Technology and Design data. The MedBIoT data can be accessed online at https://cs.taltech.ee/research/data/medbiot. We are grateful to the dataset providers.

ORCID

Yussuf Ahmed https://orcid.org/0000-0003-4079-9243

REFERENCES

1. Hosseinzadeh, M., Hemmati, A., Rahmani, A.M.: Clustering for smart cities in the internet of things: a review. Cluster Comput. 25(6), 4097–4127 (2022). https://doi.org/10.1007/s10586-022-03646-8
2. Sujay Vailshery, L.: Number of internet of things (IoT) connected devices worldwide from 2019 to 2023, with forecasts from 2022 to 2030. https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/ (2023). Accessed 27 Jul 2023
3. Chai, K.T.: Security for smart cities. IET Smart Cities 2(2), 95–104 (2020). https://doi.org/10.1049/iet-smc.2020.0001
4. Mahoor, M., et al.: State-of-the-art in smart streetlight systems: a review. IET Smart Cities 2(1), 24–33 (2020). https://doi.org/10.1049/iet-smc.2019.0029
5. Ying, W., et al.: Distributed consensus tracking of networked agent systems under denial-of-service attacks. IEEE Trans. Syst. Man Cybern. Syst. 51(10), 6183–6196 (2020). https://doi.org/10.1109/tsmc.2019.2960301
6. Roy, K.C., Chen, Q.: DeepRan: attention-based BiLSTM and CRF for ransomware early detection and classification. Inf. Syst. Front 23(2), 299–315 (2021). https://doi.org/10.1007/s10796-020-10017-4
7. Ahmed, Y., Naqvi, S., Josephs, M.: Cybersecurity metrics for enhanced protection of healthcare IT systems. In: 2019 13th International Symposium on Medical Information and Communication Technology (ISMICT), pp. 1–9. IEEE, Oslo (2019)
8. Li, Y., et al.: Deep learning in security of internet of things. IEEE Internet Things J. 9(22), 22133–22146 (2021). https://doi.org/10.1109/jiot.2021.3106898
9. Qureshi, K.N., et al.: A novel and secure attacks detection framework for smart cities industrial internet of things. Sustain. Cities Soc. 61, 102343 (2020). https://doi.org/10.1016/j.scs.2020.102343
10. Affinito, A., et al.: The evolution of Mirai botnet scans over a six-year period. J. Inf. Secur. Appl. 79, 103629 (2023). https://doi.org/10.1016/j.jisa.2023.103629
11. Nisa, M.U., Kifayat, K.: Detection of slow port scanning attacks. In: 2020 International Conference on Cyber Warfare and Security (ICCWS), pp. 1–7. IEEE, Islamabad (2020)
12. Stiawan, D., et al.: Investigating brute force attack patterns in IoT network. J. Electr. Comput. Eng. 2019, 1–13 (2019). https://doi.org/10.1155/2019/4568368
13. Woolf, N.: DDoS attack that disrupted internet was largest of its kind in history, experts say. https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet (2016). Accessed 26 Oct 2016
14. Antonakakis, M., et al.: Understanding the Mirai botnet. In: Proceedings of the 26th USENIX Security Symposium, pp. 1093–1110. USENIX Association, Vancouver (2017)
15. SonicWall: Mid year update: 2023 sonicwall cyber threat report | tracking cybercriminals into the shadows. https://www.sonicwall.com/medialibrary/en/white-paper/mid-year-2023-cyber-threat-report.pdf (2023). Accessed 12 Dec 2023
16. Mashima, D., Yuan, Li, Chen, B.: Who's scanning our smart grid? Empirical study on honeypot data. In: 2019 IEEE Global Communications Conference (GLOBECOM), pp. 1–6. IEEE, Waikiloa (2019)
17. Kato, S., et al.: Adaptive observation of emerging cyber attacks targeting various IoT devices. In: 2021 IFIP/IEEE International Symposium on

Integrated Network Management (IM), pp. 143–151. IEEE, Bordeaux (2021)
18. Trajanovski, T., Zhang, N.: An automated and comprehensive framework for IoT botnet detection and analysis (IoT-BDA). IEEE Access 9, 124360–124383 (2021). https://doi.org/10.1109/access.2021.3110188
19. Xu, Y., et al.: Brief industry paper: catching IoT malware in the wild using HoneyIoT. In: 2021 IEEE 27th Real-Time and Embedded Technology and Applications Symposium (RTAS), pp. 433–436. IEEE, Nashville (2021)
20. Tran, N.P., et al.: CFDVex: a novel feature extraction method for detecting cross-architecture IoT malware. In: Proceedings of the 10th International Symposium on Information and Communication Technology, pp. 248–254, ACM, Hanoi (2019)
21. Reddy Kondra, J., et al.: Honeypot-based intrusion detection system: a performance analysis. In: 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), pp. 2347–2351. Bharati Vidyapeeth's Institute of Computer Applicatoins and Management, Delhi (2016)
22. Lutz, E., Werner, M., Jahn, A.: Multimedia Communications in Satellite Systems, pp. 273–298. Springer Berlin Heidelberg, Berlin (2000)
23. Bhandari, S.: Internet of things vs internet: difference and comparison. https://askanydifference.com/difference-between-internet-of-things-and-internet/ (2023). Accessed 13 Jul 2023
24. Singh, M., Kaur Jhajj, N., Goraya, A.: IoT-enabled wireless mobile ad-hoc networks: introduction, challenges, applications: review chapter. Internet Things, 1, 121–134 (2022)
25. Milenkovic, M.: Internet of things: concepts and system design. In: Internet of Things: Concepts and System Design, Springer, Cham (2020). https://doi.org/10.1007/978-3-030-41346-0
26. Dange, S., Chatterjee, M.: IoT botnet: the largest threat to the IoT network. In: Jain, L.C., et al. (eds.) Data Communication and Networks, pp. 137–157. Springer, Singapore (2020)
27. Dang, H.H., Nguyen, H.D.: Detecting anomalous network traffic in IoT networks. In: 2019 21st International Conference on Advanced Communication Technology (ICACT), pp. 1143–1152. IEEE, Pyeong-Chang (2019)
28. Aqeel-ur-Rehman, S.U.R., et al.: Security and privacy issues in IoT. Int. J. Commun. Network. Inf. Secur. 8(3), 147–157 (2016). https://doi.org/10.17762/ijcnis.v8i3.2074
29. Srinidhi, N., et al.: Ensuring fault tolerant connectivity in IoT networks. Lect. Notes Network Syst., 391–400 (2021). https://doi.org/10.1007/978-981-16-0980-0_36
30. Rana, M., Mamun, Q., Islam, R.: Lightweight cryptography in IoT networks: a survey. Future Generat. Comput. Syst. 129, 77–89 (2022). https://doi.org/10.1016/j.future.2021.11.011
31. Mahlake, N., et al.: A hybrid algorithm to enhance wireless sensor networks security on the IoT. ArXiv, abs/2303.14445 (2023). https://doi.org/10.48550/arXiv.2303.14445
32. Li, Y., Tu, W.: Traffic modelling for IoT networks: a survey. In: Proceedings of the 10th International Conference on Information Communication and Management. ACM, New York (2020)
33. Zhang, W., et al.: An IoT honeynet based on multiport honeypots for capturing IoT attacks. IEEE Internet Things J. 7(5), 3991–3999 (2020). https://doi.org/10.1109/jiot.2019.2956173
34. Šemić, H., Mrdovic, S.: IoT honeypot: a multi-component solution for handling manual and Mirai-based attacks. In: 2017 25th Telecommunication Forum (TELFOR), pp. 1–4. IEEE, Belgrade (2017)
35. Suo, X., Han, X., Gao, Y.: Research on the application of honeypot technology in intrusion detection system. In: 2014 IEEE Workshop on Advanced Research and Technology in Industry Applications (WARTIA), pp. 1030–1032, IEEE, Ottawa (2014)
36. Kingsle Edwin, G., et al.: Honeypot based intrusion detection system for cyber physical system. In: 2022 International Conference on Augmented Intelligence and Sustainable Systems (ICAISS), pp. 958–962. IEEE, Trichy (2022)
37. Bhagat, N., Arora, B.: Intrusion detection using honeypots. In: 2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC), pp. 412–417. IEEE, Solan (2018)
38. Faiz Razali, M., et al.: IoT honeypot: a review from researcher's perspective. In: 2018 IEEE Conference on Application, Information and Network Security (AINS), pp. 93–98. IEEE, Landgkawi (2018)
39. Fachkha, C., et al.: Internet-scale probing of CPS: inference, characterization and orchestration analysis. In: 24th Annual Network and Distributed System Security Symposium (NDSS), pp. 1–15. The Internet Society, San Diego (2017)
40. David Guarnizo, J., et al.: Siphon: towards scalable high-interaction physical honeypots. In: Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security, pp. 57–68. ACM, New York (2017). https://doi.org/10.1145/3055186.3055192
41. Yin, M.P.P., et al.: A novel honeypot for revealing current IoT threats. J. Inf. Process. 24, 522–533 (2016)
42. Saphir Mfogo, V., et al.: AIIPot: adaptive intelligent-interaction honeypot for IoT devices. In: 2023 IEEE 34th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), pp. 1–6. IEEE, Toronto (2023)
43. Luo, T., et al.: Iotcandyjar: towards an intelligent-interaction honeypot for IoT devices. https://api.semanticscholar.org/CorpusID:33436536 (2017). Accessed 12 May 2023
44. Liu, Y.-H., Tian, D.-X., Wang, A.-M.: ANNIDS: intrusion detection system based on artificial neural network. In: Proceedings of the 2003 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.03EX693), vol. 3, pp. 1337–1342. IEEE, Xian (2003)
45. Zhou, Y.: Chameleon: towards adaptive honeypot for internet of things. In: Proceedings of the ACM Turing Celebration Conference-China, pp. 1–5. ACM, Chengdu (2019)
46. Yamamoto, M., Kakei, S., Saito, S.: FirmPot: a framework for intelligent-interaction honeypots using firmware of IoT devices. In: 2021 Ninth International Symposium on Computing and Networking Workshops (CANDARW), pp. 405–411. IEEE, Matsue (2021)
47. Srinivasa, S., Pedersen, J.M., Vasilomanolakis, E.: Interaction matters: a comprehensive analysis and a dataset of hybrid IoT/OT honeypots. In: Proceedings of the 38th Annual Computer Security Applications Conference. pp. 742–755. ACM, Austin (2022)
48. Guerra-Manzanares, A., et al.: MedbIoT: generation of an IoT botnet dataset in a medium-sized IoT network. In: International Conference on Information Systems Security and Privacy, pp. 207–2018. SciTePress, Valletta (2020)
49. Lin Aung, Y., et al.: Scalable VPN-forwarded honeypots: dataset and threat intelligence insights. In: Sixth Annual Industrial Control System Security (ICSS) Workshop, ACM, Austin (2020)
50. Wang, M., Santillan, J., Kuipers, F.A.: ThingPot: an interactive Internet-of-Things honeypot. ArXiv, abs/1807.04114, arXiv, 11 July 2018. arXiv.org, http://arxiv.org/abs/1807.04114
51. Singapore University of Technology "iTrust, Centre for Research in Cyber Security and Design". IoT Datasets. https://itrust.sutd.edu.sg/itrust-labs_datasets/ (2018). Accessed 19 Nov 2023
52. Bisong, E.: Google Colaboratory, pp. 59–64. Apress, Berkeley (2019)
53. Jay, J., Bob, R.: AWS honeypot attack data — kaggle.com. https://www.kaggle.com/datasets/casimian2000/aws-honeypot-attack-data (2013). Accessed 15 May 2024
54. Sedlar, U., et al.: Cyberlab Honeynet Dataset (2020)
55. CIC Honeynet (2017). www.honeynetproject.com. Accessed 23 May 2024
56. Churcher, A., et al.: An experimental analysis of attack classification using machine learning in IoT networks. CoRR, abs/2101.12270 21(2), 446 (2021). https://doi.org/10.3390/s21020446
57. Sadek, F.S., Abouaissa, A., Lorenz, P.: The limitations of unsupervised machine learning for identifying malicious nodes in IoT networks. In: GLOBECOM 2022 – 2022 IEEE Global Communications Conference, pp. 1984–1989. IEEE, Rio de Janeiro (2022)
58. Dalal, K.R.: Analysing the role of supervised and unsupervised machine learning in IoT. In: 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), pp. 75–79. IEEE, Coimbatore (2020)
59. April, T., et al.: Understanding the Mirai Botnet Manos Antonakakis (2018). https://api.semanticscholar.org/CorpusID:140067524. Accessed 15 Aug 2023

60. Kolias, C., et al.: DDoS in the IoT: Mirai and other botnets. Computer 50(7), 80–84 (2017). https://doi.org/10.1109/mc.2017.201
61. Bhuyan, M.H., Bhattacharyya, D.K., Kalita, J.K.: Network anomaly detection: methods, systems and tools. IEEE Commun. Surv. Tutorials 16(1), 303–336 (2014). https://doi.org/10.1109/surv.2013.052213.00046
62. Zhang, J., et al.: On the Mismanagement and Maliciousness of Networks (2014)
63. Durumeric, Z., Wustrow, E., Alex Halderman, J.: ZMap: fast internet-wide scanning and its security applications. In: 22nd USENIX Security Symposium (USENIX Security 13), pp. 605–620. USENIX Association, Washington (2013)
64. Golam, K., Mahmud, H., Jamie, P.: An Overview of UPnP-Based IoT Security: Threats, Vulnerabilities, and Prospective Solutions (2020). https://doi.org/10.48550/arXiv.2011.02587. Accessed 14 May 2024
65. Mirkovic, J., Martin, J., Reiher, P.L.: A Taxonomy of DDoS Attacks and DDoS Defense Mechanisms (2001)
66. Krämer, L., et al.: AmpPot: monitoring and defending against amplification DDoS attacks. In: Bos, H., Monrose, F., Blanc, G. (eds.) Research in Attacks, Intrusions, and Defenses, pp. 615–636. Springer International Publishing, Cham (2015)
67. Douligeris, C., Mitrokotsa, A.: DDoS attacks and defense mechanisms: classification and state-of-the-art. Comput. Network. 44(5), 643–666 (2004). https://doi.org/10.1016/j.comnet.2003.10.003
68. Taghavi Zargar, S., Joshi, J.B.D., Tipper, D.: A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks. IEEE Commun. Surv. Tutorials 15(4), 2046–2069 (2013). https://doi.org/10.1109/surv.2013.031413.00127
69. Ali Hamza, A., Al-Janabi, J.S.U.: Detecting brute force attacks on SSH and FTP protocol using machine learning: a survey. J Al-Qadisiyah Comput. Sci. Math. 16(1) (2024). https://doi.org/10.29304/jqcsm.2024.16.11432
70. Owens, J., Matthews, J.N.: A Study of Passwords and Methods Used in Brute-Force SSH Attacks (2008)
71. Venkatesh, B., Anuradha, J.: A review of feature selection and its methods (2019). https://www.researchgate.net/publication/331864360_A_Review_of_Feature_Selection_and_Its_Methods. Accessed 29 Feb 2024
72. Irmina Prasetiyowati, M., Ulfa Maulidevi, N., Surendro, K.: Determining threshold value on information gain feature selection to increase speed and prediction accuracy of random forest. J Big Data 8(1), 84 (2021). https://doi.org/10.1186/s40537-021-00472-4
73. Kumar, K., Kumar, G., Kumar, Y.: Feature selection approach for intrusion detection system. Int. J. Adv. Trends Comput. Sci. Eng. 2(5), 47–53 (2013)
74. Ahmed, Y., Asyhari, A.T., Rahman, M.A.: A cyber kill chain approach for detecting advanced persistent threats. Comput. Mater. Continua (CMC) 67(2), 2497–2513 (2021). https://doi.org/10.32604/cmc.2021.014223
75. Mandal, L., Jana, N.D.: A comparative study of Naive Bayes and k-NN algorithm for multi-class drug molecule classification. In: 2019 IEEE 16th India Council International Conference (INDICON), pp. 1–4. IEEE, Rajkot (2019)
76. Wahba, Y., ElSalamouny, E., ElTaweel, G.: Improving the performance of multi-class intrusion detection systems using feature reduction. ArXiv, abs/1507.06692 (2015)
77. Adhikari, S., Chaudhary, S.: Data mining: a bagged decision tree classifier algorithm for ids intrusion detection system based attacks classification. Des. Eng., 1826–1839 (2021). https://api.semanticscholar.org/CorpusID:236412179. Accessed 17 Sep 2023
78. Sarikaya, A., Kılıç, B.G.: A class-specific intrusion detection model: hierarchical multi-class ids model. SN Comput. Sci. 1(4), 1–11 (2020). https://doi.org/10.1007/s42979-020-00213-z
79. Ahmed, I.S., Talaat, F.M., Labib, L.M.: A hybrid intrusion detection system (HIDS) based on prioritized k-nearest neighbors and optimized SVM classifiers. Artif. Intell. Rev. 51(3), 403–443 (2019). https://doi.org/10.1007/s10462-017-9567-1
80. Xu, B., et al.: Incremental k-NN SVM method in intrusion detection. In: 2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS), pp. 712–717. IEEE, Beijing (2017)
81. Meng, W., Li, W., Kwok, L.: Design of intelligent KNN-based alarm filter using knowledge-based alert verification in intrusion detection. Secur. Commun. Network. 8(18), 3883–3895 (2015). https://doi.org/10.1002/sec.1307
82. Keras Team: Keras documentation: the Sequential model — keras.io. https://keras.io/guides/sequential_model/ (2023). Accessed 28 Feb 2024
83. Aggarwal, S., et al.: Optimized sequential model for plant recognition in keras. IOP Conf. Ser. Mater. Sci. Eng. 1022(1), 012118 (2021). https://doi.org/10.1088/1757-899x/1022/1/012118
84. Joana, R., et al.: Sequential models for endoluminal image classification — ncbi.nlm.nih.gov. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8871077/ (2022). Accessed 28 Feb 2024
85. Rahul, J., Bandaranayake Thusitha, S.: Analysis of optimizing neural networks and artificial intelligent models for guidance, control, and navigation systems. https://www.researchgate.net/publication/350567223_ANALYSIS_OF_OPTIMIZING_NEURAL_NETWORKS_AND_ARTIFICIAL_INTELLIGENT_MODELS_FOR_GUIDANCE_CONTROL_AND_NAVIGATION_SYSTEMS (2021)
86. Saad Alqahtani, A.: FSO-LSTM IDS: hybrid optimized and ensembled deep-learning network-based intrusion detection system for smart networks. J. Supercomput. 78, 9438–9455 (2022). https://doi.org/10.1007/s11227-024-05975-4
87. Aldallal, A.: Toward efficient intrusion detection system using hybrid deep learning approach. Symmetry 14(9), 1916 (2022). https://doi.org/10.3390/sym14091916
88. Zi Wei, Y., et al.: Comparing malware attack detection using machine learning techniques in IoT network traffic. Int. J. Integrated Care 13(1), 21–27 (2023). https://doi.org/10.11113/ijic.v13n1.384
89. Tasnim, A., et al.: Experimental analysis of classification for different internet of things (IoT) network attacks using machine learning and deep learning. In: 2022 International Conference on Decision Aid Sciences and Applications (DASA), pp. 406–410. IEEE, Chiangrai (2022)
90. Mosaiyebzadeh, F., et al.: A network intrusion detection system using deep learning against MQTT attacks in IoT. In: 2021 IEEE Latin-American Conference on Communications (LATINCOM), pp. 1–6. IEEE, Santo Domingo (2021)
91. Widiyasono, N., et al.: Detection of Mirai malware attacks in IoT environments using random forest algorithms. TEM J., 1209–1219 (2021). https://doi.org/10.18421/tem103-27
92. Aldhyani, T.H.H., Alkahtani, H.: Cyber security for detecting distributed denial of service attacks in agriculture 4.0: deep learning model. Mathematics 11(1), 233 (2023). https://doi.org/10.3390/math11010233
93. Abreha Haftay, G., Mohammad, H., Serhani Mohamed, A.: Federated learning in edge computing: a systematic survey — ncbi.nlm.nih.gov. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8780479/. (2022). Accessed 27 Feb 2024

How to cite this article: Ahmed, Y., Beyioku, K., Yousefi, M.: Securing smart cities through machine learning: a honeypot-driven approach to attack detection in Internet of Things ecosystems. IET Smart Cities. 1–19 (2024). https://doi.org/10.1049/smc2.12084