

**School of Science and Technology**

Machine Learning-Based Enhancement for Intrusion Detection in IoT Networks

Student Name: Gavin Goh Kai-Wen

Student ID: 20006557

Supervisor Name: Dr Athirah Mohd Ramly

Submission Date: XX/XX/2023

BSC (HONS) INFORMATION TECHNOLOGY (COMPUTER NETWORKING AND SECURITY)

PRJ3213: CAPSTONE 1 PLANNING DOCUMENT

Project title: Machine Learning-Based Enhancement for Intrusion Detection in IoT Networks

Date : xx/xx/2023

Student : Gavin Goh Kai-Wen

Supervisor : Dr Athirah Mohd Ramly

**Abstract**

PLorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

**Table of Contents:**

1. [1.0 Introduction 4](#_Toc138353933)

[1.1 Introduction to Internet of Things (IoT) 4](#_Toc138353934)

[1.2 Current Problems of IoT 6](#_Toc138353935)

[1.3 Aim of Research 7](#_Toc138353936)

[1.4 Research Objectives 7](#_Toc138353937)

1. [2.0 Literature Review 9](#_Toc138353938)

[2.1 Overview of IoT Architecture 9](#_Toc138353939)

[2.2 Overview of Intrusion Detection System (IDS) 12](#_Toc138353940)

[2.3 Limitation of Traditional IDS for IoT Networks 14](#_Toc138353941)

[2.4 Overview of Machine Learning (ML) 15](#_Toc138353942)

[2.5 ML Techniques for IDS in IoT Network 19](#_Toc138353943)

[2.6 Related Works 22](#_Toc138353944)

[2.7 Discussion 31](#_Toc138353945)

1. [3.0 Methodology 40](#_Toc138353946)

[3.1 Proposed System 40](#_Toc138353947)

[3.2 Chosen Datasets 40](#_Toc138353948)

[3.3 Data Pre-Processing 41](#_Toc138353949)

[3.4 Feature Engineering 41](#_Toc138353950)

[3.5 Model Optimization 41](#_Toc138353951)

[3.6 Performance Matrix 41](#_Toc138353952)

[3.7 Experimental Environment 42](#_Toc138353953)

[3.7 Model Comparison 42](#_Toc138353954)

1. [4.0 Work Plan and Timeline 47](#_Toc138353955)

[4.1 Gantt Chart 47](#_Toc138353956)

1. [5.0 Reference 49](#_Toc138353957)

**List of Figures and Tables:**

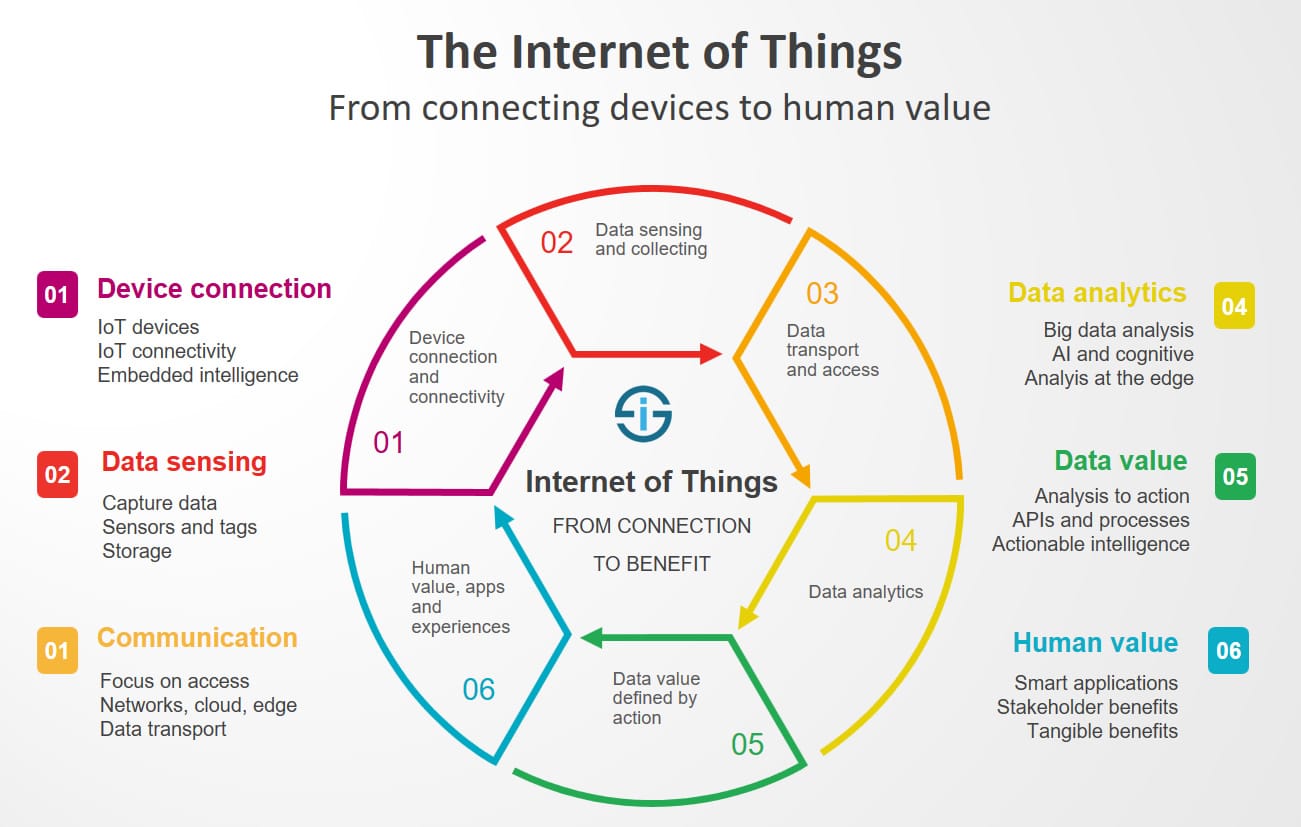
|  |  |
| --- | --- |
| **Figure No.** | **Page No.** |
| **1** | **3** |
| **2** | **4** |
| **3** | **5** |
| **4** | **9** |
| **5** | **11** |
| **6** | **12** |
| **7** | **15** |

|  |  |
| --- | --- |
| **Table No.** | **Page No.** |
| 1 | 12 |
| 2 | 27 |
| 3 | 27 |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |
| 9 |  |

**1.0 Introduction**

**1.1 Introduction to Internet of Things (IoT)**

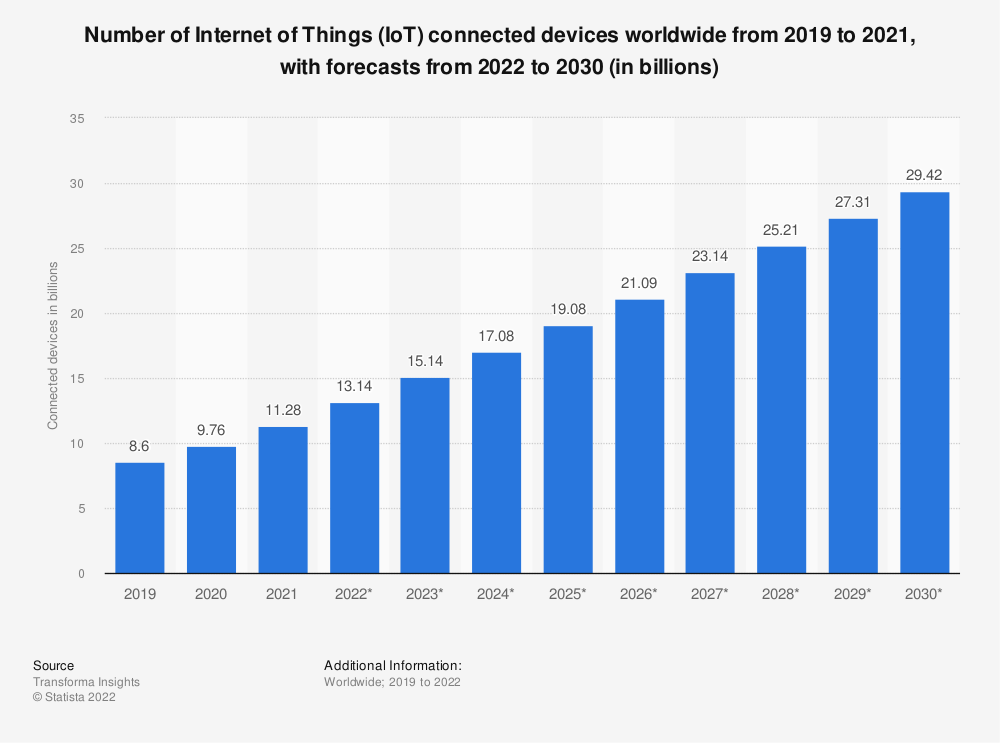
The Internet of Things or more commonly known as IoT is a network of interconnected physical objects, devices, vehicles, buildings, and even living beings, embedded with sensors, software, and connectivity capabilities. These "things" collect and exchange data over the internet, enabling real-time monitoring, control, and analysis of diverse processes and environments. By connecting the physical and digital realms, IoT empowers businesses, individuals, and communities to achieve greater efficiency, productivity, and convenience.



**Figure 1**: *IoT definitions – from connecting devices to creating value [1]*

In recent times, IoT has emerged as a revolutionary technology, transforming the way we interact with the world around us. It has rapidly gained momentum and is reshaping homes, cities, and industries, bringing upon a new era of connectivity and intelligence. IoT is not just a buzzword or a temporary trend; it represents a fundamental shift in the way devices and systems communicate, gather data, and make informed decisions. The underlying technologies that enable IoT's functionality include wireless communication protocols, cloud computing, data analytics, and artificial intelligence (AI). With the proliferation of low-cost sensors, affordable connectivity options, and advancements in data processing capabilities, the deployment of IoT devices and systems has become more accessible and scalable.

The applications of IoT span across various sectors, including healthcare, manufacturing, agriculture, transportation, energy, and smart cities. In the healthcare sector, IoT devices facilitate remote patient monitoring, wearable health trackers, and efficient healthcare delivery systems [2]. While in the manufacturing sector, IoT enables predictive maintenance, supply chain optimization, and intelligent automation [3]. On the other hand, IoT-based solutions help enhance the agriculture sector with crop monitoring, irrigation control, and livestock management [4]. These are just a few examples among the countless possibilities that IoT brings forth.

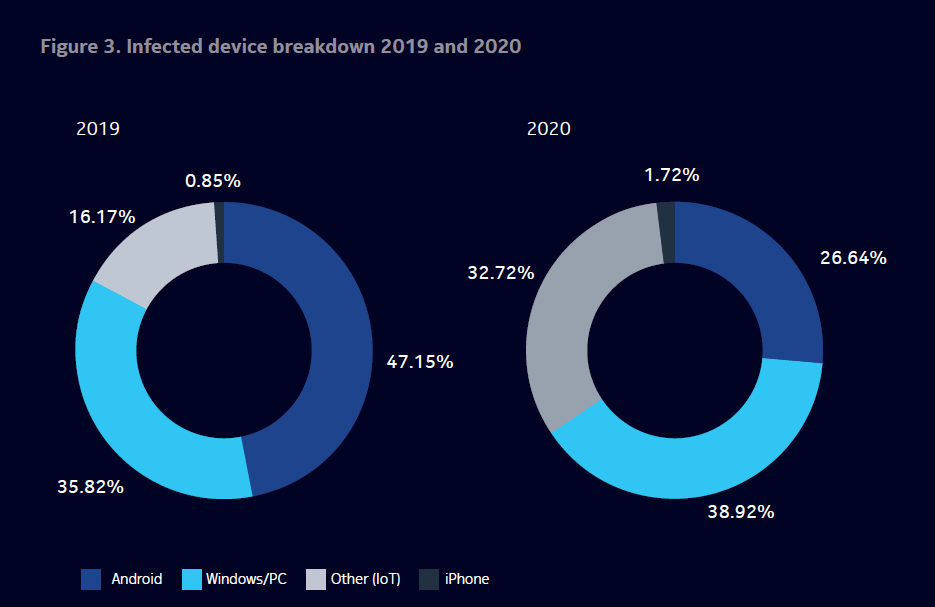
According to data from a Statista report by Lionel Sujay Vailshery [5], the number of IoT connected devices reached 13.1 billion by the end of 2022. Looking ahead, it is projected that by 2030, the number of connected devices will soar to 29.4 billion, indicating a remarkable 350% increase over the course of a decade [5]. The significant growth in the number of IoT connected devices, as shown in the report [5] showcases the accelerating adoption and integration of IoT technology across various industries and sectors.

**Figure 2**: *Graph from Statista [5], showcasing the forecast growth of IoT connected devices.*



**1.2 Current Problems of IoT**

The Internet of Things (IoT) has witnessed rapid growth and widespread adoption, bringing forth a new era of connectivity and intelligence. However, this expansion has also introduced significant security challenges. According to a study conducted by Fortinet, a leading cybersecurity company, the number of IoT-related attacks experienced a staggering 400% increase in 2020, emphasizing the growing security challenges faced by IoT networks [6]. As IoT networks expand, they become potential targets for various malicious activities, including unauthorized access, data breaches, and attacks on critical infrastructure. Ensuring the security and integrity of IoT networks is of utmost importance to protect sensitive data, maintain user privacy, and prevent disruptions in critical operations.



**Figure 3**: *Pie chart showing the increase in Infected Devices from 2019 to 2020 [7]*

Intrusion detection plays a crucial role in safeguarding IoT networks against potential threats and attacks. However, traditional intrusion detection techniques which were primarily designed for conventional networks, may not be well-suited to handle the unique characteristics and challenges posed by IoT environments which is evident by the fact that IoT related attacks are constantly increasing yearly. These challenges include resource constraints of IoT devices, dynamic network behaviour, and the diversity of connected devices. Moreover, IoT devices often operate in heterogeneous and decentralized environments, making it difficult to deploy centralized security mechanisms [8]. As a result, there is a pressing need for enhanced intrusion detection mechanisms that can effectively address the security concerns specific to IoT networks.

**1.3 Aim of Research**

The primary objective of this research is to overcome the inherent limitations of traditional intrusion detection techniques applied to IoT networks, by proposing an innovative intrusion detection mechanism that leverages the immense potential of machine learning. With its cognitive capabilities, machine learning empowers the system to efficiently analyse and learn from the extensive data generated within IoT networks, automatically detecting intricate patterns, identifying anomalies, and dynamically adapting to evolving threats. The integration of machine learning algorithms into intrusion detection systems promises to enhance their overall resilience, precision, and effectiveness in promptly identifying and mitigating potential intrusions in IoT environments, safeguarding the integrity and security of connected devices and systems.

**1.4 Research Objectives**

The specific objectives of this research project are as follows:

* **Topic Research** - to explore the challenges and unique characteristics of intrusion detection in IoT networks.
* **Literature Review and Analysis –** to review and analyse existing machine learning techniques for intrusion detection in IoT networks.
* **Model Development -** to develop an optimized intrusion detection system using machine learning algorithms tailored specifically for IoT networks.
* **Model Evaluation -** to evaluate the performance and effectiveness of the developed intrusion detection system through extensive experimentation and analysis.
* **Model Comparison -** to compare the performance of the developed system with traditional intrusion detection approaches in IoT networks.
* **Final Insight -** to provide insights and recommendations for enhancing the security of IoT networks through machine learning-based intrusion detection.

**1.5 Project Scope**

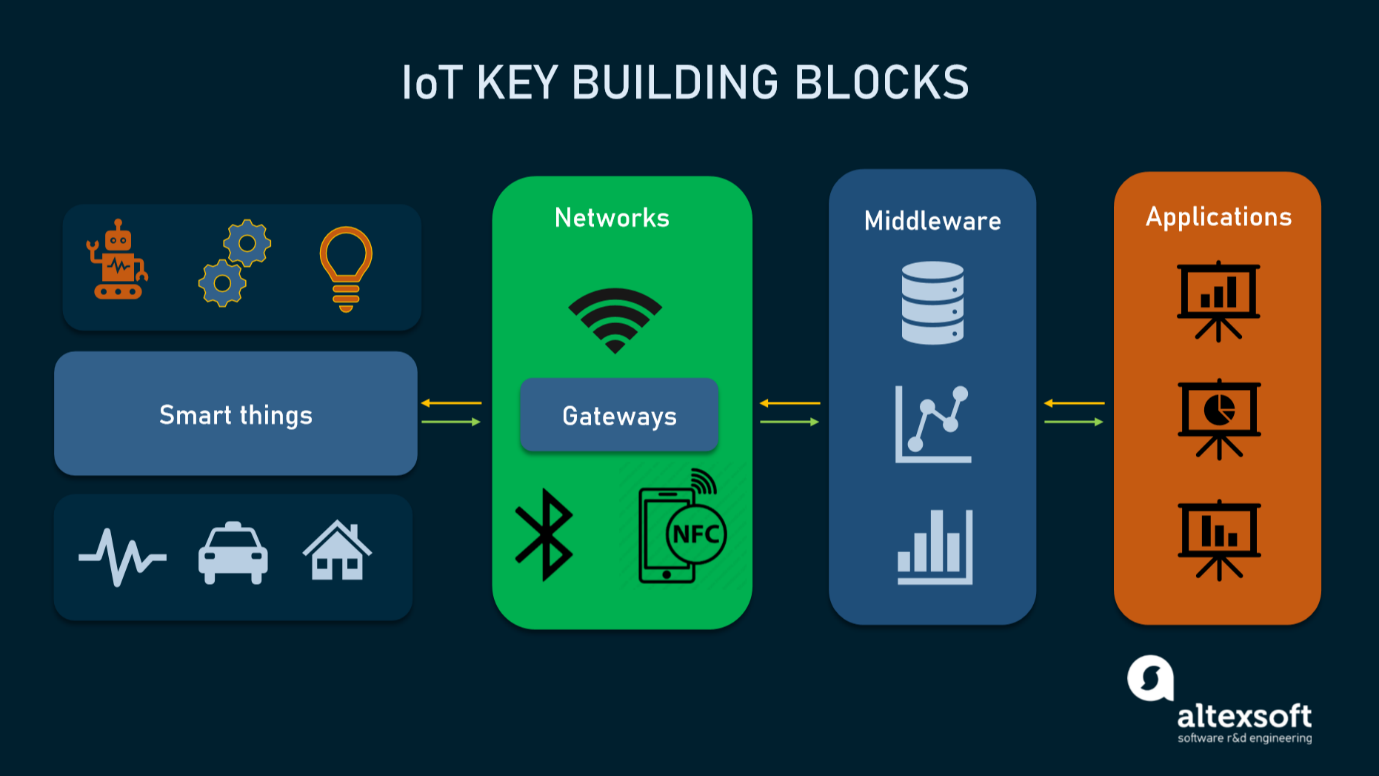
The scope of this project encompasses the development and evaluation of a machine learning-based enhancement for intrusion detection in IoT networks, with a focus on finding a balance between resource efficiency and effectiveness. The research will explore and analyze various machine learning techniques, including decision trees, random forests, support vector machines, and neural networks, aiming to develop an optimized and reliable intrusion detection system. The evaluation phase will involve extensive experimentation and analysis to assess the performance and effectiveness of the developed system while considering resource constraints. Additionally, the project will compare the performance of the machine learning-based approach with traditional intrusion detection methods in IoT networks. By considering the trade-off between resource efficiency and effectiveness, this research aims to contribute to the body of knowledge in IoT security and provide practical insights for enhancing the security and trustworthiness of IoT deployments.

**2.0 Literature Review**

This literature review section aims to provide a comprehensive understanding of the current state of research related to intrusion detection in IoT networks using machine learning techniques. It includes basic background to all related topics and provide a comprehensive review of papers relating to the research title.

**2.1 Overview of IoT Architecture**

As mentioned before, IoT Networks refers to a system of interconnected devices, sensors, and objects that communicate with each other and exchange data over the internet. These devices, known as IoT devices, are embedded with sensors, actuators, and networking capabilities that enable them to collect, transmit, and receive data. According to a blog published by Altexsoft [8], there is no single, agreed-upon architecture of IoT Networks. Instead, its architecture layers and complexity are dependent on a particular business task. But no matter the use case and number of layers, any IoT structure would always the same key building blocks as shown in Figure 4.



**Figure 4**: *Image showing the key building blocks of an IoT network [8]*

The explanation of each of the key building blocks are as follows:

* **Smart Things –** Refers to the IoT Devices, which are physical objects or devices that are equipped with sensors or actuators to collect and/or manipulate data. Examples include smart thermostats, wearable fitness trackers, industrial sensors, smart appliances, and so on.
* **Networks –** IoT devices connect to the network using various wireless or wired technologies, depending on the specific requirements of the application. Common connectivity options include Wi-Fi, Bluetooth, Zigbee, Z-Wave, RFID, cellular networks (2G, 3G, 4G, or 5G), LPWAN (Low Power Wide Area Network) technologies like LoRaWAN or NB-IoT, and Ethernet [8].
* **Gateways –** In larger IoT networks, gateways serve as intermediaries between the IoT devices and the internet [8]. They aggregate data from multiple devices, perform protocol translation if needed, and transmit the data to a central server or cloud platform. Gateways also provide additional security measures, manage device authentication, and optimize network traffic.
* **Middleware –** IoT middleware or IoT platform is the software that simplifies the connection between devices and applications on the Internet of Things. It manages data communication, security, and integration, abstracting device complexities and ensuring interoperability [8]. By transforming and aggregating data, it enables efficient communication and enhances development and management processes. IoT middleware plays a crucial role in handling data flow, simplifying interactions, and facilitating seamless communication between IoT devices and applications [8].
* **Applications –** IoT applications are the specific usages or purposes for which IoT technology is utilized [8]. They represent the practical implementation of IoT in various domains, addressing specific needs and challenges. These applications leverage IoT capabilities to enable functions such as smart homes, industrial automation, healthcare monitoring, or environmental monitoring. They provide real-world value and solutions by harnessing the power of interconnected devices and data collection in a specific context.

These elements make up the backbone of any IoT system upon which effective, multi-layered architecture can be developed. The most common IoT architecture is the five-layer architecture which is shown in Figure 5 in the next page.

A picture containing text, screenshot, font, brand

Description automatically generated

**Figure 5**: *Image showing the key building blocks of an IoT network [9]*

Each of these layers have its own specific functions which is shown in the following:

* **Business Layer –** This layer is designed to manages the whole IoT system effectively which includes applications, business and profit models, and users’ privacy [9].
* **Application Layer –** This layer is accountable for delivering application-specific functionalities directly to the user by outlining methods for implementing IoT in various contexts like smart homes or smart cars. [9].
* **Processing Layer** – This layer more widely known as the Middleware layerstores, analyses, and processes huge amount of data. This is the layer process like employ databases, cloud computing and big data processing modules are conducted [9].
* **Transport Layer –** The primary role of this layer is to transmit sensor data from the perception layer to the processing layer. It accomplishes this task by employing diverse networks like RFID, Bluetooth, and 3G. [9].
* **Perception Layer –** This is the physical layer. It has sensors to find and gather information about an environment, including the ability to identify other smart objects [9].

**A diagram of a network intrusion detection system

Description automatically generated with medium confidence2.2 Overview of Intrusion Detection System (IDS)**

**Figure 6**: *Image showing the functions of an IDS [10]*

An Intrusion Detection System (IDS) is a passive monitoring device designed to monitor and analyse network traffic, system activities, and user behaviour in order to detect and respond to potential security breaches or unauthorized access attempts [11]. It acts as an additional layer of defence alongside firewalls and other security measures. The primary goal of an IDS is to identify and alert system administrators or security personnel about any suspicious or malicious activities that may indicate an ongoing or attempted attack. It helps in identifying potential threats, collecting relevant information, and initiating appropriate response actions. IDS are designed to be deployed in different environments [11]. And like many cybersecurity solutions, an IDS can be sorted into two category which are host-based or network-based as shown in the following:

* **Host-Based IDS (HIDS) -** A host-based IDS is installed on a specific endpoint (host devices) to safeguard it from internal and external threats [11]. This type of IDS can monitor network traffic going to and from the machine, observe active processes, and examine the logs of the system. While a host-based IDS has limited visibility restricted to its host machine, which reduces the context available for making decisions, in return it possesses comprehensive visibility into the internal workings of the host computer [11].
* **Network-Based IDS (NIDS)** - A network-based IDS solution is specifically developed to monitor an entire secured network [11]. It possesses visibility into the entirety of the network traffic and makes assessments based on packet metadata and contents. This broader perspective offers greater context and the capability to identify widespread threats [11]. However, these systems do not have visibility into the internal workings of the endpoints they are protecting [11]

|  |  |  |
| --- | --- | --- |
| **Benefit** | **Host** | **Network** |
| **Deterrence** | Strong Deterrence for insiders. | Strong Deterrence for insiders. |
| **Detection** | Strong Insider Detection.  Weak Outsider Detection. | Weak Insider Detection.  Strong Outsider Detection. |
| **Response** | Weak real-time response.  Good for long-term attacks. | Strong Response Against Outsider Attacks. |
| **Damage Assessment** | Excellent for determining extend of compromise. | Very weak damage assessment capabilities. |
| **Attack Anticipation** | Good at trending and detecting suspicious behaviour patterns. | None. |
| **Prosecution Support** | Strong prosecution support capabilities | Very weak there is no Data Source Integrity. |

**Table 1**: *Comparing Network and Host based Benefits [12]*

Deploying either a Host-Based Intrusion Detection System (HIDS) or a Network-Based Intrusion Detection System (NIDS) alone does not offer complete protection to an organization's system due to the varying levels of visibility they provide [11]. To achieve more comprehensive security, organizations can opt for a unified threat management solution that combines multiple technologies into a single system [11]. Beyond their deployment location, IDS solutions also differ in how they identify potential intrusions, these solutions are shown in the following:

* **Signature Detection** - Signature-based IDS solutions rely on known threat signatures to detect and identify malicious content [11]. When malware or other threats are recognized, a signature is created and included in the IDS's list for scanning incoming content. This approach ensures a high detection rate with no false positives, as alerts are triggered only by known malicious content. However, signature-based IDS solutions have limitations, as they can only detect known threats and are unable to identify zero-day vulnerabilities.
* **Anomaly Detection -** Anomaly-based IDS solutions construct a model of the expected behaviour of the system being protected. Any deviations from this model are considered potential threats and trigger alerts [11]. This approach is capable of detecting new or previously unseen threats. However, accurately establishing the "normal" behaviour model poses a challenge, and these systems must strike a balance between false positives (incorrect alerts) and false negatives (missed detections)
* **Hybrid Detection -** A hybrid IDS combines signature-based and anomaly-based detection methods, allowing it to identify a wider range of potential attacks with improved accuracy compared to using either approach separately [11].

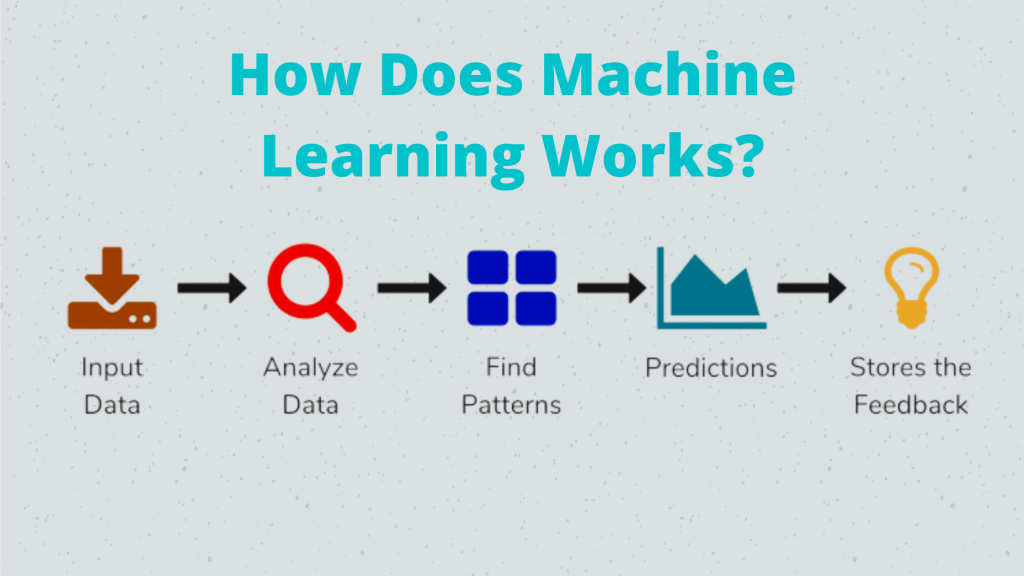
**2.3 Limitation of Traditional IDS for IoT Networks**

Traditional intrusion detection techniques, like Snort, Sucritara, McAFree, Zeek and Snorby are primarily designed for conventional networks, may not be well-suited for IoT environments, as it usually only uses signature-based IDS solutions. Signature Detection relies on predefined signatures, patterns, or rules to identify known attacks or suspicious activities. Which comes with certain limitations when applied to IoT networks if not combined with machine learning techniques as shown in the following:

* **Limited detection of unknown threats –** Signature-based detection are less effective at detecting new or previously unseen attacks, commonly known as zero-day vulnerabilities [11]. Since they rely on predefined signatures or rules, they may not recognize novel attack patterns or variations that do not match the existing signatures or rules.
* **False positives and negatives:** - Signature-Based Detection may generate false positives, which are alerts triggered for benign or non-malicious activities, leading to unnecessary disruptions or investigations [13]. Conversely, false negatives can occur when the IDS fails to detect actual attacks, posing a risk to the security of the IoT network.
* **Scalability challenges** - IoT networks typically consist of numerous interconnected devices, generating massive amounts of data and network traffic. This is troublesome as signature-based detection can be resource-intensive and slow down the network performance, as it has to scan every packet of data from the IoT network [14].
* **Difficulty in handling dynamic IoT environments**: - IoT networks often have a dynamic and evolving nature, with frequent changes in devices, configurations, and communication patterns. Traditional IDS solutions may struggle to adapt quickly to these changes, leading to decreased accuracy and potential security gaps [13].
* **Lack of context and behavioural analysis -** Traditional IDS solutions typically lack the ability to analyse the behaviour and context of IoT devices and their interactions within the network. Without this contextual understanding, the IDS may struggle to differentiate between normal and anomalous activities, potentially leading to missed detections or false alarms.

It's important to note that while traditional IDS approaches without machine learning have limitations, they still play a crucial role in providing basic security measures for IoT networks. However, integrating machine learning techniques can enhance the IDS capabilities by enabling more accurate detection of unknown threats, adaptive learning, and improved contextual analysis.

**2.4 Overview of Machine Learning (ML)**

Machine Learning (ML) is a subset of Artificial Intelligence (AI) that provides computers cognitive capabilities to quote on quote “self-learn” from data sets consisting of training data, without the need to be specifically programmed [15]. Machine learning algorithms possess the capability to recognize patterns within data and utilize that knowledge to make informed predictions. In essence, machine learning algorithms and model gain knowledge by actively learning from their experiences [15].

**Figure 7**: *Image showing the Step-by-step process of Machine Learning [16]*

Although Machine Learning is a field within computer science, it differs from traditional computational approaches [17]. In traditional computational approach, a programmer will code a series of instructions as algorithms that directs a computer on how to convert input data into wanted output [15]. These algorithms are mainly based on a conditional statement (also known as an IF-THEN statement) which refers to when a specific action that are executed only when certain conditions are met. In contrast to the traditional method, Machine learning algorithms enable computers to undergo training on input data and utilize statistical analysis to produce output values that lie within a predefined range. As a result, Machine learning empowers computers to construct models from sample data, thereby automating decision-making processes based on data inputs [17].

Although AI and ML are sometimes used interchangeably, they represent distinct concepts. AI encompasses a broader scope, referring to machines making decisions, acquiring new skills, and solving problems in a human-like manner. On the other hand, machine learning is a subset of AI that specifically enables intelligent systems to autonomously learn from data [15].

A picture containing text, circle, screenshot, font

Description automatically generated

**Figure 8**: *Types of Machine Learning [18]*

Machine learning is classified into many categories, as the type of machine learning model used is dependent on the nature of the input data. Based on the type of input data used to train the algorithm, machine learning can be split into four categories which are **Supervised, Unsupervised, Semi-Supervised** and **Reinforcement Learning** as shown in Figure 8. Each of these categories will be explained in this following section.

**Supervised Learning**

Supervised Learning is the category used when the input data provided to the machine learning model is labelled (which means the value is known). With this method, both the input and output data are known, and the goal is to find a mapping function, denoted as f, that can accurately predict the output (Y) based on the input (X). This process involves the algorithm learning from known input-output pairs and adjusting its parameters to minimize the difference between predicted values (Y') and the actual values (Y) [19]. The learning algorithm has the capability to compare its output (Y) with the correct intended output (Y), known as the ground truth label, and adjust based on identified errors, such as through back-propagation [20]. This learning process is akin to a supervisor supervising the process hence it is named Supervised Learning.

Supervised Learning can be further characterized into **classification** and **regression** cases.

* **Classification** - Classification involves predicting a category for the output variable [20], such as classifying an email as "spam" or "not spam."
* **Regression** – Regression deals with predicting a real continuous value for the output variable [20], like estimating the temperature for the next day.

Supervised learning is commonly used for a variety of tasks, such as spam email detection, sentiment analysis, stock price forecasting, customer churn prediction, image recognition, and medical diagnosis. Some frequently used supervised learning models include Linear Regression, Decision Trees, Support Vector Machines (SVM) and Random Forest [20].

**Unsupervised Learning**

Unsupervised Learning is the category used in situations where the values of input data are known, but the output and mapping function are unknown. In unsupervised learning, the algorithms aim to discover patterns or similarities among different input data instances (samples) and group them based on these similarities [19]. Since there is no pre-existing output data to guide the learning process, algorithms generate a pseudo-output or do not require any output at all. Instead, they focus on finding similarity indexes, such as distance measurements, to create a mapping function to determine the proximity or dissimilarity between data samples. Unsupervised learning is characterized by its lack of supervision and the algorithms' ability to autonomously identify patterns or structures within the input data [19].

Unsupervised Learning can be further characterized into **clustering, dimensionality reduction,** and **association** cases.

* **Clustering** In a clustering problem, the objective is to identify natural groupings within a dataset [19], such as categorizing plants based on its characteristic/feature such as colour, number of leaves, etc.
* **Association**– Association rule learning involves uncovering associations or relationships between items in a dataset [19]. For instance, it aims to discover patterns like customers who purchase product A are also likely to purchase product B.
* **Dimensionality Reduction -** Dimensionality reduction in unsupervised learning reduces the number of features in a dataset while preserving important information, addressing challenges of high-dimensional data [19]. It aids data visualization, exploration, and enhances machine learning algorithm performance.

Unsupervised learning is applied to various tasks like market segmentation, fraud detection and image segmentation. Some commonly used Unsupervised models include K-Means, Apriori Algorithm for learning association rule and Principal Component Analysis [19].

**Semi-supervised Learning**

Semi-supervised Learning is the category used in situations where some of the inputted data are known but not all which means the data is partially annotated [19]. It is a technique where training data is divided into two parts: a small portion with labelled data and a larger portion with unlabelled data [15]. By leveraging the labelled data, the model can make predictions and draw insights about the unlabelled data, leading to more accurate results compared to traditional supervised learning models. This approach is particularly advantageous for tasks involving large datasets, like image classification, as it reduces the need for a large amount of labelled data, making it faster, more cost-effective, and suitable for businesses dealing with massive data volumes. Semi-supervised learning finds applications in scenarios where labelled data is limited but unlabelled data is abundant, such as text classification, image classification, fraud detection, speech recognition, and anomaly detection.

**Reinforcement Learning**

Reinforcement learning is a category of machine learning where the software agent aims to maximize rewards by determining the best course of action in a given situation. Unlike supervised or unsupervised learning, reinforcement learning relies on trial and error without the need for pre-existing training data [15]. Through exploration and learning from mistakes, reinforcement learning models seek to identify actions that lead to optimal solutions or maximum rewards, following a self-guided and autonomous approach.

Reinforcement learning finds significant applications in the fields of robotics and gaming, where there is a clear cause-and-effect relationship between actions and outcomes [15]. Video games, in particular, provide an ideal environment for refining reinforcement learning algorithms as they offer measurable success through scoring systems. Leveraging this link, video games serve as effective tools for enhancing and fine-tuning reinforcement learning models.

**Deep Learning**

Deep learning is a subset of machine learning that focuses on training artificial neural networks with multiple layers to learn hierarchical representations of data [21]. These neural networks, called deep neural networks, are capable of automatically learning intricate patterns and features from raw data. A popular example of a deep learning model is the Convolutional Neural Network (CNN), widely used in computer vision tasks. CNNs employ specialized layers, such as convolutional and pooling layers, to extract and process visual features from images, enabling tasks like object detection and image classification. Deep learning has also shown remarkable performance in natural language processing, speech recognition, and generative modelling tasks.

**2.5 ML Techniques for IDS in IoT Network**

Traditional Intrusion Detection Systems in IoT networks often struggle to keep up with the dynamic and evolving nature of threats as mentioned before. This is where Machine Learning techniques come into play, offering a promising solution to enhance traditional IDS capabilities in IoT networks. By utilizing machine learning algorithms, IDS can analyse vast amounts of network traffic, sensor data, and device interactions to detect and mitigate potential security breaches. Machine learning brings the power of automation, pattern recognition, and anomaly detection, enabling IDS to adapt, learn, and evolve alongside emerging threats in real-time.

The most commons traditional machine learning techniques that were used by programmers in the early stages of machine learning implementation for IDS include:

1. **Support Vector Machines (SVM) -** SVM is a highly utilized supervised machine learning algorithm that can handle both classification and regression tasks [22]. Its effectiveness relies on the choice of the kernel and the parameters used. SVM has gained popularity in anomaly intrusion detection due to its ability to handle high-dimensional data and its strong generalization capabilities. The SVM algorithm identifies a single hyperplane from a set of N-planes that effectively distinguishes data points [28]. This hyperplane plays a crucial role in separating the data points distinctly. The dimension of the hyperplane is determined by the number of features in the dataset. Visualizing the hyperplane becomes more challenging as the number of features increases.

The SVM kernel is another important term of the model [28], which transforms non-separable problems into separable ones. The kernel performs extreme data conversions to facilitate the separation of data points based on their labels or defined outputs. The choice of the kernel determines how the data is separated.

1. **Decision Trees (DT) -** A decision tree is a versatile machine learning algorithm used for classification and regression tasks [22]. It creates a tree-like structure by recursively partitioning data based on selected features. Internal nodes represent attributes, and leaf nodes represent class labels or values. During training, the tree learns to make decisions by evaluating features and splitting data to minimize impurity. In the prediction phase, new data follows attribute conditions to reach a leaf node, providing the predicted class or value [22]. Decision trees handle both categorical and numerical data, but they can overfit and struggle with generalization. Techniques like pruning, which removes unnecessary branches, and ensemble methods, like Random Forest, which combine multiple decision trees, can help improve their performance and mitigate these limitations.
2. **Random Forest (RF) -** Random Forest is a powerful machine learning algorithm that is commonly used for classification and regression tasks [22]. It constructs multiple decision trees during the training process and combines their predictions to obtain the final result. The accuracy of the model improves with the number of trees. Random Forest employs the bagging method to create diverse training samples, ensuring that no two samples are identical [28]. It utilizes the CART (Classification and Regression Trees) method to build individual decision trees. Random Forest is known for its ability to handle noise and demonstrate excellent categorization performance. In Random Forest, the final output is generated by aggregating the outputs from multiple decision trees. In classification problems, this aggregation is done through a simple majority voting approach, while for regression problems, it uses the mathematical average method.
3. **k-Nearest Neighbours (k-NN) -** k-NN is widely recognized as one of the most significant and popular algorithms due to its simplicity and ease of implementation [22]. It belongs to the category of example-based learning methods and is often referred to as a lazy learning technique [28]. KNN is commonly employed for tasks such as statistical estimation and pattern recognition.

In the KNN algorithm, when given new or testing data, it searches for the K nearest objects among the provided training data [28]. The proximity between training and testing data is determined using the Euclidean distance formula. The data point with the lowest distance is referred to as the nearest neighbour. This approach is typically used when all attributes in the dataset are continuous. Once the K training instances closest to the unknown object are identified, the most frequently occurring classification among these K instances is selected as the basis for determination [28]. However, a challenge with this algorithm is the selection of an appropriate value for K, which is not easily determined.

1. **Naive Bayes** **(NB)** - Naive Bayes is a probabilistic classifier that applies Bayes' theorem with the assumption of feature independence. Its efficiency, scalability, and ability to handle high-dimensional data make it popular in various domains. It calculates the posterior probability of a class given the input features by combining prior knowledge and the likelihood of observing the features. Naive Bayes assumes that features are conditionally independent, which simplifies the computation. It works well even with limited training data and performs real-time classification. Naive Bayes has been successfully used in text mining, document classification, spam detection, sentiment analysis, and intrusion detection systems, among others.

Although the traditional ML technique mentioned before may seem enough for Intrusion detection systems and has shown effectiveness in improving anomaly and intrusion detection, there are still limitations and challenges that traditional ML techniques struggles to deal with due to the unique challenges posed by IoT environments. For example, IoT networks generate massive volumes of heterogeneous data, characterized by temporal dependencies and resource constraints. Traditional ML techniques struggle to scale and handle IoT data's complexity while addressing privacy and security concerns. Hence the reason why in recent times, Deep Learning models such as Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) are being incorporated into IDS, as it offers better scalability and adaptability to the data generated by IoT network. However, this comes with the problem of computing resources, as complex intrusion detection models are not suitable for the massive heterogeneous network environment of the Internet of Things. Therefore, striking a balance between model complexity and resource efficiency remains an ongoing concern in the development of IDS for the heterogeneous and resource constrained IoT environment.

Additionally, as mentioned before ensemble methods can improve the performance of an IDS by leveraging the strengths of different models and mitigating their individual weaknesses. Here are a couple of popular ensemble techniques that can be applied to IDS:

* **Bagging** - Bagging (Bootstrap Aggregating) is a technique that involves training multiple instances of the same model on different subsets of the training data [23]. In the context of IDS, multiple instances of a specific model like Random Forest or SVM can be trained on different subsets of the labelled dataset. The predictions of each model are then combined through voting or averaging to make the final decision.
* **Boosting** - Boosting is another ensemble technique that trains multiple models in sequence, where each subsequent model focuses on correcting the mistakes made by the previous models [24]. In boosting, the models are trained iteratively, and more weight is given to the misclassified instances in each iteration [24]. Gradient Boosting algorithms like XGBoost, CatBoost, LightGBM or AdaBoost are commonly used in IDS.

By combining different models through ensemble techniques, IDS can benefit from improved detection accuracy, robustness to different types of attacks, and better generalization performance. Hence in the next related work section, the most popular used machine learning and deep learning model will be recoded in order to determine which models will be best used ensembled for the purpose of this project.

**2.6 Related Works**

This section will show information regarding various related research paper and highlight the models that researchers for their research. These related works will show which models are the most commonly used after 2018 and allows better understanding of the current state of machine learning technology for IDS.

1. M. Alani [25] evaluated four different ML models Decision Tree, Gaussian Naïve-Bayes, Logistic Regression and Random Forests with ToN IoT Datasets (Network Flow) in order to find out which model is best suited for IDS for IoT Network. Results shows that Random Forests achieved the best result followed by Decision Tree. The result of Random Trees was 99.971% accuracy, with 0% false-negative, and only 0.29% false-positive.
2. L. Manocchio, et al. [26] aim to propose a high-performance ML-based NIDS, capable of running on resource-constrained edge IoT devices. The authors choose Decision Trees with Depth 5-12 as the ML model due to its low complexity and high classification performance while primarily using the datasets of ToN-IoT, BoT-IoT, and MQTT datasets (Network Flow). Results shows 99%+ Accuracy on all datasets at Depth 12, higher than other approaches on a more powerful device while having a low memory footprint which only occupied 10% program space in a ESP8266.
3. M. Ge, et al. [27] developed a feed-forward neural networks model using BoT-IoT datasets (Packet Capture) for binary and multi-class classification including denial of service, distributed denial of service, reconnaissance, and information theft attacks against IoT devices. Results for Binary Classification were 0.99+ F1-Score for all reconnaissance attacks, 0.6 False Negative and 0.3 False Positive for service scan attacks, above 0.9 precision for Information Theft Attacks. While the accuracy of multi-class classification was 98.1% for Normal 99.4% for DDOS 98.4% for Reconnaissance and 88.9% for Information Theft. The lower accuracy for information theft of both classifications could be attributed by lower sample size from the dataset. When compared to a SVM model, FFNN model were more time efficient for classification over large datasets.
4. D. Chauhan, et al. [28] compares the results obtained by applying Support Vector Classifier, Decision Tree Classifier and Random Forest Classifier on the CICIDS -17 (Data Flow). Results shows that both DT and RF Classifier performed better with a few differences than SVM, with RF scoring the best cross validation score of 0.9610 out of the two and DT scoring F-1 score of 1.0 for Macro Average and Weighted Average in Training. Both RF and DT shown 96% Accuracy in testing.
5. T. Saba, et al [29] proposed a two-stage hybrid model Two-stage hybrid method with Genetic Algorithm for feature selection and ML Classifier like SVM, DF, ensemble classifier (Bagged Tree) using NSL-KDD Datasets. A higher result of classification in terms of accuracy has been observed on the NSLKDD dataset through the ensemble classifier using 10- fold cross-validation technique displaying 99.8% accuracy.
6. M. Alani, et al. [30] present a deep learning-based intrusion detection system using Multi-Layer Perceptron (MLP) classifier for industrial IoT. The proposed system was trained and tested using the WUSTL-IIOT-2021 dataset (Network Flow). Testing results showed accuracy exceeding 99% with minimally low false-positive, and false-negative rates.
7. S. Li, et al. [31] conducted a comparative analysis of 15 machine learning techniques for both binary classification and multi-classification tasks using NSL-KDD and CIC-IDS2017 datasets. The evaluation was based on their performance in handling IoT traffic data, considering indicators related to both temporal and spatial aspects. The Tree Algorithms (DT, RF, BG) performed the best in both Binary and Multi-Class Classification achieving 0.99 Accuracy for both classifications. KNN also shown to be performing just a tiny behind the Tree algorithms, and Clustering Technique like Cluster K Mean was suggested that it should be used in conjunction with other supervised algorithms to achieve best result. The conclusion was Decision Tree had the best result with the least time cost.
8. D. Priya, et al. [32] compared 7 popular models (Logistic Regression, Decision Tree, K Nearest Neighbour, Multi-Layer Perception, Random Forest, Naïve Bayes and Support Vector Machines) on three data sets: UNSW-NB15, KDD Cup 99, and NSL-KDD. Both DT and KNN algorithms provided the highest levels of accuracy out of the 7 algorithms, but DT took the lesser time hence it is more efficient and effective.
9. R. Kale, et al. [33] introduce a network intrusion attack detection framework that utilizes deep learning and consists of three stages: unsupervised learning using K-means clustering, semi-supervised learning using GANomaly, and supervised learning using CNN. The performance of the framework is then evaluated on three benchmark datasets: NSL-KDD, CIC-IDS2018, and TON IoT. The proposed framework achieved better FPR with comparable TPR performance when evaluated against state-of-the-art methods such as KMeans, GANomaly and One Class Support Vector Machine.
10. J. Ashraf, et al. [34] proposed an SDN-based IoT Anomaly detection system design comprising of SVM, KNN and MLP which was evaluated using UNSW-NB15 and ISCX dataset. The detection results achieved by the design is either on par or outperform other IDS systems that rely on different ML techniques. These results have been validated using two benchmarked datasets, demonstrating the effectiveness and competitiveness of the design created by the authors.
11. E. Ciklabakkal, et al. [35] developed an IDS for MQQT for IoT Network named ARTEMIS using Python Outlier Detection (PYOD). A comparative analysis was conducted for Autoencoder, Single-Objective Generative Adversarial Active Learning (SO GAAL), Random Forest, Isolation Forest, OCSVM and K Means Clustering. The data collection was done by using aDHT11 sensor connected to a raspberry PI. The experiment shown that ML based IDS can perform well even when not trained with a previously known attack.
12. L. Yang et al. [36] presented a hybrid IDS model called MTH-IDS, which combines a Signature-based IDS using tree-based models (DT, RF, ET, XGBoost) with an Anomaly Detection-based IDS using Cluster K-Means (CL-K) for detecting zero-day attacks. The authors incorporated various data pre-processing algorithms, such as K-Means Clustering Sampling, SMOTE, and Z-Score, to enhance training efficiency and model accuracy. While also applying various algorithms along with the training such as Stacking was performed, and BO-TPE was applied to improve the accuracy of known attacks, while BO-GP and Two Biased Classifier were applied to improve detection rate for unknown attacks. The algorithms show massive difference, with accuracy increasing from 99.861% to 99.879%, and a 70.2% decrease in execution time. MTH-IDS demonstrated high effectiveness in detecting different known attacks, achieving accuracies of 99.99% and 99.88% on the CAN-intrusion-dataset and CICIDS2017 dataset, respectively. Additionally, the system exhibited the capability to detect various unknown attacks, with average F1-scores of 0.963 and 0.800 on the CAN-intrusion dataset and CICIDS2017 dataset, respectively.
13. R. K. Vigneswaran, et al. [37] utilized Deep Neural Network (DNN) to predict the attacks on N-IDS using KDD CUP 99 dataset. Through testing, it was determined DNN with 3 layers is the most efficient. When compared to other popular ML techniques such as DT and RF, DNN-3 Layers performed the best with the F-1 Score of 0.955.
14. B. Xue, et al. [38] proposed an intrusion detection approach based on heterogeneous transfer learning (HTL) consisting of an Autoencoder architecture and CNN for unsupervised domain adaptation, which was tested using CICIDS2017 as datasets and Kitsune and IoTID as target domain.
15. X. Li, et al. [39] proposed an IDS that is using an CNN that was improved by introducing Inception module for optimal intrusion feature extraction based on the traditional convolution neural network; tested using NSL-KDD dataset. Results shows that normal CNN have a test set accuracy of 74.22% while the improved CNN model have 95.17%
16. X. Cheng, et al. [40] proposed hybrid model of swam intelligence fusion of various ML techniques (DT, RF, GB, Perceptron) and QBSO-FS which was tested using NSL-KDD datasets. The results show that each ML performance has improved.
17. G. Guo, et al. [41] utilized the recently published TON\_IoT network dataset to construct an IoT Intrusion Detection System (IDS). Through a thorough evaluation of ten machine learning algorithms and careful deliberation, the XGBoost method was selected as the final classifier. Experimental results demonstrate that the model achieves exceptional performance, with an MCC (Matthews Correlation Coefficient) of 99.84% in binary classification and 99.17% in multiclass classification.
18. L. Yang et al. [42] introduced a new ensemble IDS framework called LCCDE (Leader Class and Confidence Decision Ensemble). In this framework, the best-performing machine learning model from three advanced algorithms (XGBoost, LightGBM, and CatBoost) is trained on the CAN and CICIDS17 datasets and their result chosen for each specific class or type of attack. This framework does not require feature engineering as the algorithms are able to feature important score and selection during training. Notably, LightGBM shown the best result with the least time taken in most of the test. The results shows that the framework achieved 99.9997% and 99.811% on CAN and CICIDS, respectively.
19. S.M. Kasongo [43] IDS that incorporates an XGBoost-based feature extraction method along with multiple machine learning algorithms. The XGBoost algorithm, known for its ensemble-tree approach, was applied to reduce the number of attributes in the UNSW-NB15 dataset. Among the classifiers used, the LR (Logistic Regression) method was included. Experimental findings showcased that the XGBoost-LR combination achieved an accuracy of 75.51% for binary classification and 72.53% for multiclass classification. To address the issue of class imbalance in the UNSW-NB15 dataset, the authors recommended the use of oversampling techniques.
20. S. Singh, et al. [44] proposed a Multi Classifier Intrusion Detection system (MCIDS) using CNN which was trained and tested using UNSW-NB15 dataset. The paper conclude that CNN give higher and better accuracy with low false positive and negative compared to other deep learning models.
21. N. Chaabouni, et al. [45] conducted a survey which analyzed and compared state-of-the-art NIDS proposals in the IoT context. Out of all the state-of-the-art proposal they reviewed, the MLP-based model performed the best. The MLP model used a three-layer feed-forward and backward Neural Network with unipolar sigmoid transfer functions in the hidden and output layers. It employed a stochastic learning algorithm with a mean square error function. The NIDS was tested in a simulation with four client nodes and a server relay node, where DoS and DDoS attacks were launched. The MLP-based NIDS achieved an overall attack detection accuracy of 99.4% with a 0.6% false positive rate, demonstrating network stability.
22. I. Ullah, et al. [46] proposed a CNN-based IDS which was validated using the BoT-IoT, IoT Network Intrusion, MQTT-IoT-IDS2020, and IoT-23 datasets. Additionally, Transfer learning is used to implement binary and multiclass classification using a CNN multiclass pre-trained model. The binary and multiclass classification models proposed demonstrated excellent accuracy, precision, recall, and F1 score, surpassing existing classification strategies and recent deep learning implementations. The CNN1D model achieved a minimum detection rate of 99.74%, the CNN2D model achieved 99.42%, and the CNN3D model achieved 99.03%
23. G. Guo, et al. [47] introduces an IDS framework for IoT systems that utilizes machine learning techniques. The researchers evaluate the performance of ten different models using the TON\_IoT network dataset. The experimental findings reveal that the stacking-ensemble model (DT, EF) performs exceptionally well as a classifier. It achieves high MCC scores of 0.9971 for binary classification and 0.9909 for multiclass classification.
24. R. Elsayed, et al. [48] present a secure automatic two-level intrusion detection system (SATIDS) that combines the minimum redundancy maximum relevance (MRMR) feature selection technique with a recurrent neural network (RNN), which was tested using ToN-IoT and InSDN datasets. The goal of SATIDS is to improve the accuracy of detecting traffic anomalies while reducing the detection time. The proposed system shown improve accuracy with results F1-score of 98.05% on the ToN-IoT dataset. On the InSDN dataset, F1-score of 99.3% was obtained.
25. S. Fenanir, et al. [49] proposed a lightweight intrusion detection model that utilises three Feature Selection algorithms (Filter Methods, Wrapper Methods, and embedded Methods). Multiple ML was tested using model using KDD99, NSL-KDD, and UNSW-NB15 datasets. After performing testing, DT and KNN was shown to have the best result followed by MLP. However, KNN takes too much time compared to DT, which meant DT had the best result.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Authors** | **Years** | **Title** | **Methodology** | **Advantage** | **Limitation** | **Results** |
| M. Alani [25] | 2022 | IoTProtect: A Machine-Learning Based IoT Intrusion Detection System | Decision Tree, Gaussian Naïve-Bayes, Logisitic Regresstion,  Random Forest Using ToN-IoT Datasets | Random Forest using the Ton IoT Datasets were able to reach an accuracy of 99.999% | Limited Dataset, Weak Performance Metrics  Individual Classifier | Random Forests achieved 99.971% Detection Accuracy with 0% False Negative and only 0.29% false-positive. |
| L. Manocchio et al. [26] | 2022 | Network Intrusion Detection System in a Light Bulb | Decision Tree Depth 5-12, Using ToN-IoT, BoT-IoT, and MQTT datasets | Higher accuracy, Low Memory Footprint | Weak to Zero Day Attacks due to not having unsupervised models. Weak Performance Metrics | Model were able to achieve 99%+ Accuracy on all datasets while only occupying 10% program space of ESP8266 |
| M. Ge, et al [27] | 2019 | Deep Learning-Based Intrusion Detection for IoT Networks | Feed-forward neural networks model using BoT-IoT datasets | Higher Accuracy, More time efficient for classification over Large Datasets. | Low Precision in Information Theft, Classifier Confusion, Small Sample size for some attacks, only works in Batch Mode | Binary Classification were 0.99+ F1-Score for all reconnaissance attacks, 0.6 False Negative and 0.3 False Positive for service scan attacks, 90% precision for Information Theft Attacks.  While the accuracy of multi-class classification was 98.1% for Normal 99.4% for DDOS 98.4% for Reconnaissance and 88.9% for Information Theft. |
| D. Chauhan, et al. [28] | 2023 | A Novel Intrusion Detection System based on Machine Learning for Internet of Things (IoT) Devices | SVM, DT, RF  using CICIDS-17 | Random Forest and Decision Tree are shown to have better accuracy than Support Vector Classifier | Limited Datasets, Imbalanced Datasets | Results shows that both DT and RF Classifier performed better with a few differences than SVM, with RF scoring the best cross validation score of 0.9610 out of the two and DT scoring F-1 score of 1.0 for Macro Average and Weighted Average in Training. Both RF and DT shown 96% Accuracy in testing. |
| T. Saba, et al. [29] | 2021 | Intrusion Detection System Through Advance Machine Learning for the Internet of Things Networks | Two-stage hybrid method using Genetic Algorithm and using ML Classifier like SVM, DF, EN (Bagged Tree) using NSL-KDD Datasets | Higher result of classification in terms of accuracy | Limited Datasets, Not IoT focused, Weak Performance Metric | Results shows that Ensembled Classifier with the pre-set of Bagged Tree have an accuracy of 99.8%, the highest of out the three classifiers. |
| M. Alani, et al. [30] | 2022 | DeepIIoT: An Explainable Deep Learning Based Intrusion Detection System for Industrial IOT, | Multi-Layer Perceptron (MLP) classifier using WUSTL-IIOT-2021 dataset | High Accuracy, Able to handle imbalance issue in the dataset | Limited Dataset. Limited Feature Set, Individual Classifier | Testing results showed accuracy exceeding 99% with minimally low false-positive, and false-negative rates. |
| S. Li, et al. [31] | 2022 | Can We Half the Work with Double Results: Rethinking Machine Learning Algorithms for Network Intrusion Detection System | 14 traditional statistical learning algorithms and 1 neural network algorithms to train using NSL-KDD and CIC-IDS2017 | Multiple Datasets and Classifier Used. Decision Tree has the best result with the least time cost | Not enough Deep Learning Classifier Tested. Not tested in a real IoT environment | Tree Algorithms (DT, RF, BG) performed the best in both Binary and Multi-Class Classification achieving 0.99 Accuracy. Followed by that KNN also performed well |
| D. Priya, et al. [32] | 2022 | Lightweight Intrusion Detection System(L-IDS) for the Internet of Things | Comparing 7 popular models (LR, DT, KNN, MLP, RF, NB, SVM) on three data sets: UNSW-NB15, KDD Cup 99, and  NSL-KDD | Multiple Datasets and Classifier Used. Decision Tree had the best Result | Individual Classifier. Ensemble Method not used | DT and KNN algorithms provided the highest levels of accuracy, but DT took the lesser time hence it is more efficient and effective. |
| R. Kale, et al. [33] | 2022 | A Hybrid Deep Learning Anomaly Detection Framework for Intrusion Detection | Framework consists of three methods: K-means clustering, GANomaly, and CNN. Which was evaluated on three benchmark datasets: NSL-KDD, CIC-IDS2018, and TON IoT. | Higher accuracy when compared to other methods in most areas | Performed worse than OCSVM when evaluated using NSL-KDD datasets | The proposed framework achieved better FPR with comparable TPR performance when evaluated against state-of-the-art methods such as KMeans, GANomaly and OCSVM. |
| J. Ashraf, et al. [34] | 2021 | Intrusion Detection System for SDN-enabled IoT Networks using Machine Learning Techniques | Design comprising of SVM, KNN and MLB which was evaluated using UNSW-NB15 and ISCX dataset. | Higher accuracy, Multiple Classifier used which decrease bias | Not IoT Focused Datasets | The detection results achieved by design is either on par or outperform other IDS systems that rely on different ML techniques |
| E. Ciklabakkal, et al [35] | 2019 | ARTEMIS: An Intrusion Detection System for MQTT Attacks in Internet of Things | Python Outlier Detection (PYOD). With multiple techniques and self-collected datasets using DHT11 Sensor | Good result even when dealing with a unknown attack. | Data Set not used, the design is not explained well | The experiment shown that ML based IDS can perform well even when not trained with a previously known attack. |
| L. Yang, et al. [36] | 2022 | MTH-IDS: A Multitiered Hybrid Intrusion Detection System for Internet of Vehicles | Signature based IDS consisting of tree-based models (DT. RF, ET, XGABoost) and an Anomaly Detection Based IDS using K-Mean Clustering (CLK). This is tested using CAN and CICIDS2017 dataset | Better Generalization, Better Optimization, Lower Time, High Accuracy, Hybrid Model,  Parallel Execution | Complicated might be hard to implement into an IoT device which has resource constraints. data distribution modelling limitation due to low complexity of CLK | The proposed system demonstrates high effectiveness in detecting different known attacks, achieving accuracies of 99.99% and 99.88% for CAN and CICIDS respectively. While the result for unknown attacks, were F1-scores of 0.963 and 0.800 for CAN and CICIDS respectively. |
| R. K. Vigneswaran, et al. [37] | 2018 | Evaluating Shallow and Deep Neural Networks for Network Intrusion Detection Systems in Cyber Security | DNN using KDD Cup 99 Datasets | Deep Learning, Performed Better than Deep Learning | Old Benchmark Dataset, Limited Datasets | Through testing, it was determined DNN with 3 layers is the most efficient. When compared to other popular ML techniques DNN-3 Layers performed the best with the F-1 Score of 0.955. |
| B. Xue, et al. [38] | 2022 | Deep Transfer Learning for IoT Intrusion Detection | Autoencoder architecture and CNN for unsupervised domain adaptation, which was tested using CICIDS2017 as datasets and Kitsune and IoTID as target domain | Multiple Datasets, Higher Detection Rate | Binary Classification | The proposed approach significantly improves cross-domain network intrusion detection compared to the baseline method without domain adaptation, outperforming other deep domain adaptation techniques. |
| X. Li, et al. [39] | 2022 | Intrusion Detection System Using Improved Convolution Neural Network | CNN improved with Inception Module using NSL-KDD Datasets | Improved CNN shown better result than normal CNN | Limited Datasets, Weak Performance Matrix, only one other Model Compared | Results shows that normal CNN have a test set accuracy of 74.22% while the improved CNN model have 95.17% |
| X. Cheng, et al. [40] | 2020 | Intrusion detection system based on QBSO-FS | Hybrid model of swam intelligence fusion of various ML techniques (DT, RF, GB, Perceptron) and QBSO-FS | Improve ML performance by Feature Selection | Lack of Performance Matrix | Each Corresponding Machine Learning methods has improved performance |
| G. Guo, et al. [41] | 2022 | An Intrusion Detection System for the Internet of Things Using Machine Learning Models | XGBOOST using ToN-IoT datasets | XGBoost performed the best among 10 other classifiers | Non-Standardised Performance Matrix, Individual Classifiers, Limited Datasets | The model achieves exceptional performance, with an MCC (Matthews Correlation Coefficient) of 99.84% in binary classification and 99.17% in multiclass classification. |
| L. Yang et al. [42] | 2020 | LCCDE: A Decision-Based Ensemble Framework  for Intrusion Detection in The Internet of Vehicles | LCCDE using XGBoost, CatBoost, and LightGBM. CAN and CICIDS dataset | Best Performed is chosen for each attack. No Additionally Feature Engineering is Required | Only Supervised Models. | The results shows that the framework achieved 99.99997% and 99.811% on CAN and CICIDS, respectively. |
| S. M. Kasongo [43] | 2021 | An Advanced Intrusion Detection System for IIoT Based on GA and Tree Based Algorithms | XGBoost combined with LR using UNSW-NB15 | Ensemble Method used | Poor Performance, Data Imbalance | XGBoost-LR combination achieved an accuracy of 75.51% for binary classification and 72.53% for multiclass classification |
| S. Singh, et al. [44] | 2021 | MCIDS-Multi Classifier Intrusion Detection system for IoT Cyber Attack using Deep Learning algorithm | CNN using UNSWB15 | Higher Accuracy | Lack of comparison, Weak Performance Matrix | CNN  gave higher and better accuracy with low false positive and low false negative rates. |
| N. Chaabouni, et al. [45] | 2019 | Network Intrusion Detection for IoT Security Based on Learning Techniques | MLP-based NIDS tested using simulation | Best performance among state-of-the-art proposals in the IoT context. | Requires a training dataset for model building. | The MLP-based NIDS achieved an overall attack detection accuracy of 99.4% with a 0.6% false positive rate, demonstrating network stability. |
| I. Ullah, et al. [46] | 2021 | Design and Development of a Deep Learning-Based Model for Anomaly Detection in IoT Networks | CNN, tested using the BoT-IoT, IoT Network Intrusion, MQTT-IoT-IDS2020, and IoT-23 dataset | Various Dataset used, Higher Accuracy | Individual Classifier, Resource Intensive | The binary and multiclass classification models proposed demonstrated excellent accuracy, precision, recall, and F1 score, surpassing existing classification strategies and recent deep learning implementations. The CNN1D model achieved a minimum detection rate of 99.74%, the CNN2D model achieved 99.42%, and the CNN3D model achieved 99.03% |
| G. Guo, et al. [47] | 2023 | An IoT Intrusion Detection System Based on TON IoT Network Dataset | A Stacking-Ensemble Model consisting of XGBoost, DT, EF, tested using ToN-IoT dataset. | High Accuracy, Ensemble Method Used | Non-Standardised Performance Matrix Limited Datasets | The proposed model achieves high MCC scores of 0.9971 for binary classification and 0.9909 for multiclass classification. |
| R. Elsayed, et al. [48] | 2023 | A Hierarchical Deep Learning-Based Intrusion Detection Architecture for Clustered Internet of Things. | MRMR Feature Selection with RNN model tested using ToN-IoT and InSDN datasets | Two datasets used, Increased Accuracy, Decreased Detection Time | Individual Classifier | The proposed system shown improve accuracy with results F1-score of 98.05% on the ToN-IoT dataset. On the InSDN dataset, F1-score of 99.3% was obtained. |
| S. Fenanir, et al. [49] | 2019 | A Machine Learning-Based Lightweight Intrusion Detection System for the Internet of Things | Feature Selection algorithms (Filter Methods, Wrapper Methods, and embedded Methods) using KDD99, NSL-KDD, UNSW-NB15 datasets | DT shown the best result when it comes to accuracy to time efficiency. | Individual Classifier | After performing testing, DT and KNN was shown to have the best result followed by MLP. However, KNN takes too much time compared to DT, which meant DT had the best result. |

**Table 2**: Literature Review Matrix

*ac*

**2.7 Discussion**

After conducting a thorough review of the Related Work section, valuable insights have been gathered regarding the ideal ML model architecture for an IoT environment. It has been emphasized that the proposed IDS should prioritize lightweight and less resource-intensive characteristics, considering the processing capabilities of individual IoT nodes. One prominent finding, as indicated by [49] and [32], is that the limited processing capacity and power consumption of IoT nodes make it unfeasible to deploy an active intrusion detection agent in each node. As a solution, a centralized IDS architecture has been adopted to overcome these limitations while addressing the challenges of limited capacity and peripheral heterogeneity. In this approach, the IDS is implemented at the network layer, positioned above the Gateway component. To further optimize resource usage, NIDS have been recognized as more suitable for this centralized architecture. NIDS exhibits the advantage of requiring fewer resources compared to HIDS, which would impose excessive demands on the limited resources of IoT nodes.

|  |  |  |  |
| --- | --- | --- | --- |
| **Machine Learning Models** | | | |
| **Name** | **Category** | **Frequency** | **Percentage** |
| **Decision Tree (DT)** | Supervised | 9/41 | 21.95% |
| **Random Forest (RF)** | Supervised | 6/41 | 14.63% |
| **Convolutional Neural Network (CNN)** | Supervised | 5/41 | 12.20% |
| **XGBoost (XGB)** | Supervised | 4/41 | 9.76% |
| **K-Nearest Neighbors (KNN)** | Supervised | 3/41 | 7.32% |
| **Multi-Layer Perceptron (MLP)** | Supervised | 3/41 | 7.32% |
| **K-Means Clustering (CL-K)** | Unsupervised | 2/41 | 4.88% |
| **Bagged Tree (BG)** | Supervised | 2/41 | 4.88% |
| **GANomaly** | Unsupervised | 1/41 | 2.44% |
| **Gradient Boosting (GB)** | Supervised | 1/41 | 2.44% |
| **Perceptron** | Supervised | 1/41 | 2.44% |
| **Deep Neural Network (DNN)** | Semi-Supervised | 1/41 | 2.44% |
| **CatBoost** | Supervised | 1/41 | 2.44% |
| **LightGBM** | Supervised | 1/41 | 2.44% |
| **Support Vector Machine (SVM)** | Supervised | 1/41 | 2.44% |

**Table 3**: Frequency of ML

*ac*

Table 2 presents the Machine Learning models utilized in the papers from the Related Work section. The table includes the name, category, frequency, and percentage for each ML model. Notably, the following ML models were used more than once: DT, RF, CNN, XGB, MLP, and BG. Therefore, a comprehensive evaluation will be conducted for these algorithms, excluding ML models that were only used once. Bagged Trees has also been omitted from the evaluation since it is considered an inferior version of RF, rendering it redundant. Additionally, Catboost, and LightGBM will be evaluated along with XGBoost, as they are all tree based gradient boosting models feature similar benefits and limitations.

|  |  |  |
| --- | --- | --- |
| **Evaluation of Most Used ML Algorithm** | | |
| **Algorithm** | **Advantage** | **Limitation** |
| **Decision Tree (DT)** | * High Accuracy * Resource and Time Efficient * Lightweight | * Struggle with Complex IoT Data * Prone to Overfitting |
| **Random Forest (RF)** | * Robust and effective for handling diverse IoT data with high accuracy * Provides ensemble learning capabilities | * Resource Intensive for Large-Scale Deployment * Longer Time Needed |
| **XGBoost**  **(XGB)** | * High accuracy and good performance on IoT data * Deals well with imbalance datasets. * Built in Feature Engineering | * Requires careful tuning of hyperparameter |
| **Convolutional Neural network (CNN)** | * Effective in analysing IoT sensor data * Capable of detecting patterns and anomalies | * Requires high computational power which is not suitable for IoT environment. |
| **Multi-Layer Perceptron**  **(MLP)** | * Able to capture non-linear relationships in IoT data. * High Accuracy (Better than Tree Models) * Lightweight in terms of DL | * Requires careful tuning and regularization of the model. * Hard to implement due to complexity of DL |
| **K-Nearest Neighbour (KNN)** | * High Accuracy * Simple and Intuitive for IoT data analysis | * Slow Prediction Phase * Does not scale well. |
| **K-Mean Clustering (CLK)** | * Unsupervised Learning to detect zero-day attacks. * Low Resource Requirement * Lightweight | * Require Setup due to needing to manually choose the number of clusters. |

**Table 4**: ML Evaluation

*ac*

Table 3 provides a comprehensive evaluation of the algorithms used in the IDS context. It reveals that tree-based models, namely Decision Tree, Random Forest, and XGBoost, have demonstrated superior performance in IDS according to references [25, 26, 28, 29, 31, 32, 36, 41, 43, 47]. Reference [31] and [32] highlighted that Decision Tree stands out as the most efficient model due to its high accuracy, low complexity, and low time cost, as highlighted in [31] and [32]. Furthermore, [26] suggests that a Decision Tree with a depth of 12 achieves the highest accuracy among the evaluated models. Considering the findings, it is concluded that Decision Tree, either alone or in combination with RF, and Tree gradient models as shown in [36], would serve as the most suitable base model as it fits the lightweight and efficient requirement for IDS in IoT network.

As for deep learning model such as CNN and MLP, it is stated by [31] that Deep learning would be too resource intensive to work in an IoT network, which is especially true for CNN as seen by its limitation in Table 3. For MLP however, due to its lightweight nature and good result in some references like [32, 45, 49], which in some cases performed even better than tree models, could be considered for the proposed IDS, but further testing is needed to test the resource usage.

As for KNN, although its precision has always been shown to be on par with DT as shown in reference [31, 32, 49], a common concern is its slow prediction phase when compared to the tree models and its lack of scalability, which is why it would not be considered in the proposed IDS. K-Mean Clustering has shown to be used in multiple references like [31, 33, 36] due to its unsupervised nature which deals with the biggest problem of Signature based models like the Tree Models and MLP which is the detection of Zero-Days attack as stated in. Authors of [31] suggested that clustering method such as CLK can be used in conjunction with supervised model to achieve the best result. As shown in [36], biased classifier can also be applied to after the CL-K model to increase accuracy as evident from the increase from 0.826 F-1 score to 0.96307 F-1 score. Additionally, CL-K has the advantage of being lightweight due to its low complexity which is suitable for the IoT Network’s requirement. In summary, to avoid bias, overfitting, and improve performance, it is advisable to create an ensemble classifier instead of relying on an individual classifier. This recommendation has been supported by several references, including [29, 33, 34, 36, 47]. Therefore, a hybrid or ensemble model combining supervised models like Tree Models and MLP with an unsupervised model like CL-K could be the recipe for a robust IDS.

Additionally, there are two types of classification: binary classification, which categorizes a packet as either an attack or normal, and multi-class classification, which not only identifies whether a packet is an attack but also specifies the type of attack. Binary classification is less resource-intensive, while multi-class classification provides a more comprehensive analysis. Both types of models will be created separately in testing to offer variation for the proposed IDS.

Moving on, datasets serve as a foundation for training a machine learning model, which is why it is important to choose a dataset that suits the need of a ML-based IDS for an IoT network.

|  |  |  |
| --- | --- | --- |
| **Training Datasets** | | |
| **Name** | **Frequency** | **Percentage** |
| **NSL-KDD Dataset** | 7/36 | 19.44% |
| **ToN-IoT Dataset** | 6/36 | 16.67% |
| **CICIDS2017 Dataset** | 5/36 | 13.89% |
| **UNSW-NB15 Dataset** | 4/36 | 11.11% |
| **Bot-IoT Dataset** | 3/36 | 8.33% |
| **KDD Cup 1999 Dataset** | 2/36 | 5.56% |
| **MQTT Dataset** | 2/36 | 5.56% |
| **IoT Intrusion Detection Dataset** | 2/36 | 5.56% |
| **WUSTL-IIOT-2021** | 1/36 | 2.78% |
| **Invasive Network Traffic Dataset** | 1/36 | 2.78% |
| **ISCX IDS 2012 Dataset** | 1/36 | 2.78% |
| **IoT-23 Dataset** | 1/36 | 2.78% |
| **KITSUNE Dataset** | 1/36 | 2.78% |

**Table 5:** Frequency of Datasets

*ac*

Tables 4 displays all the datasets used to train machine learning models in the related work section. Notably, datasets such as NSL-KDD, ToN-IoT, CICIDS2017, UNSW-NB15, BoT-IoT all shown at least 3 or more usages in the related section. Thus, these datasets are chosen to be evaluated, while excluding all the datasets that have 2 or less datasets due to their low usages.

|  |  |  |
| --- | --- | --- |
| **Evaluation of Most Used Datasets** | | |
| **Algorithm** | **Advantages** | **Limitations** |
| **NSL-KDD** | * Popular and the most used * Based on 41 Features for each connection with the class label * Provide Network Traffic * No Duplicated Records | * Based on an old dataset from 1999 * Lack of modern low footprint attack scenarios. * Not for IoT Systems |
| **ToN-IoT** | * Represents real-world IoT Network Traffic with 126 Features * Modern (Released in 2020) * It has various normal and attack events for different IoT/IIoT services. * Contains heterogeneous data sources | * Relatively New * Only 7 Attack Types |
| **CICIDS2017** | * Comprehensive labelled network flow with 78 Features * For ML and DL purposes * Covers multiple attacks | * Not Public * Not For IoT Systems |
| **UNSW-NB15** | * Provide Modern normal activities and synthetic contemporary attack behaviour. * Covers multiple attacks | * More Complex due to similarity of modern normal and attack activities * Not for IoT Systems |
| **Bot-IoT** | * Focus on IoT Botnet Attacks: * Labelled Botnet network with 46 Features * Realistic Attack Scenario in IoT environment | * Limited Variety of Attacks * Small Scale * Specific to Botnet Attack which means it does not help train against other type of attacks. * Synthetic IoT Traffic |

**Table 6:** Datasets Evaluation

*ac*

Table 5 presents the evaluation of the most frequently used datasets from the related work section, along with their advantages and limitations. Based on the evaluations, ToN-IoT and CICIDS2017 appear to be the most suitable choices due to their respective advantages. ToN-IoT stands out for providing realistic IoT network traffic, consisting of 126 features, making it a perfect fit for an IoT-based ML IDS. Moreover, it is a relatively new dataset released in 2020 and contains heterogeneous data sources, addressing a primary concern in developing an IDS for an IoT network.

On the other hand, although CICIDS2017 is not explicitly focused on IoT, it does include some IoT-based attacks and covers a wide range of attack types. This diversity is beneficial for training unsupervised models to detect zero-day attacks.

NSL-KDD, UNSW-NB15, and BoT-IoT are excluded from consideration due to their limitations. NSL-KDD lacks modern attack scenarios, UNSW-NB15 suffers from similarity among records, and BoT-IoT has a limited variety of attacks. In summary, ToN-IoT and CICIDS17 emerge as the most favourable options for an IoT-based ML IDS, offering distinct advantages that address the specific requirements of the system. Thus, both of these datasets would be use as the benchmark to train the proposed IDS for this research. Additionally, the reason for choosing multiple datasets over only using one is because it is a common practice as shown in reference [26, 31, 32, 33, 34, 36, 46, 48, 49] and also provide advantage provides advantages such as enhanced generalization, increased robustness, reduced overfitting, improved coverage, and mitigation of dataset biases.

Other than ML models and datasets. There are two important steps in Machine learning, namely: Data Pre-Processing and Feature Engineering. Data Pre-processing turns data into a format that is suitable for machine learning models. Decisions regarding pre-processing must consider the data format, as well as the type of model being used [26]. While Feature Engineering helps improve the quality of datasets for more accurate and efficient model learning by removing irrelevant, redundant, and noisy features through Feature Selection. It also creates new features through Feature extraction methods to enhance interpretability of the datasets [36].

According to [43], data pre-processing involves several essential steps to prepare the data for analysis, modelling, or machine learning tasks. These steps include:

* **Removing duplicates** - Identifying and removing duplicate records from the dataset to ensure data integrity and avoid bias in subsequent analyses.
* **Replacing missing data** - Handling missing values by imputing or replacing them with appropriate values to avoid data loss and maintain the integrity of the dataset.
* **Fixing structural errors** - Correcting any structural errors in the data, such as inconsistencies in formatting, data types, or naming conventions, to ensure consistency and accuracy.
* **Removing unwanted observation -** Eliminating irrelevant or potentially noisy observations that may adversely affect the analysis or model performance.

Data imbalance can also impact the accuracy of machine learning models [43]. To address this issue, oversampling is often conducted to obtain a balanced dataset, where the minority class is over-represented to achieve better model performance.

|  |  |  |
| --- | --- | --- |
| **Pre-Processing Methods** | | |
| **Methods** | **Type** | **Function** |
| **K-Means Clustering** | Clustering and Sampling | Divides data points into K clusters, allowing for grouping or subsampling based on similarity. |
| **Random Sampling** | Sampling | Randomly selects a subset of data points from a larger dataset for analysis or training. |
| **Random Oversampling** | Oversampling | Increases the representation of minority class samples by duplicating them randomly. |
| **SMOTE** | Oversampling | Synthetic Minority Over-sampling Technique: Generates synthetic samples for the minority class to balance class distribution. |
| **Z-Score Normalization** | Normalization | Standardizes data by subtracting the mean and dividing by the standard deviation. |
| **Min-Max Normalization** | Normalization | Rescales data to a specified range (e.g., 0 to 1) by subtracting the minimum and dividing by the range. |

**Table 7:** Pre-Processing Methods

*ac*

In addition to handling these data quality aspects, other important functions including oversampling is shown in Table 7 and the purpose of each of the function is as follows:

* **Clustering -** Identifying natural groupings or patterns within the data using clustering techniques. This helps in further analysis, targeted sampling, or creating new features based on these groupings.
* **Sampling -** Selecting representative subsets from the dataset to reduce computational costs, improve efficiency, or create balanced training and testing sets.
* **Normalization -** Applying normalization techniques like Z-Score and Min-Max normalization to scale the data appropriately and make it comparable across different features or variables.

By performing these data pre-processing steps, the data is refined, ensuring its quality, usability, and addressing issues such as duplicates, missing data, structural errors, unwanted observations, and data imbalance. This prepares the data for subsequent analysis, modelling, or machine learning tasks. However, according to [26] pre-processing considerations for tree-based models are less critical compared to neural networks, as tree models are inherently robust to unprocessed categorical and unscaled numerical data. Experimental results in [26] demonstrate comparable performance between pre-processed and non-pre-processed tree-based models.

The next step after data pre-processing is feature engineering which there are two primary functions: Feature Extraction and Selection.

* **Feature selection -** This involves choosing a subset of relevant features from the available set, aiming to reduce dimensionality, eliminate noise, and enhance model efficiency.
* **Feature extraction** - This focuses on creating new features or transforming existing ones to capture underlying patterns and relationships, often using techniques like principal component analysis (PCA) or deep learning.

Both techniques are crucial for optimizing the input data and ensuring that models can effectively learn from the most informative aspects of the data, leading to improved accuracy, generalization, and understanding of the underlying data characteristics. All of the Feature Engineering methods under the Related Work Section can be found in Table 8

|  |  |  |
| --- | --- | --- |
| **Feature Engineering Methods** | | |
| **Methods** | **Type** | **Function** |
| **Genetic Algorithm (GA)** | Selection | Uses evolutionary principles to select the best subset of features by optimizing a fitness function. |
| **Filter Method** | Selection | Ranks and selects features based on their individual relevance to the target variable using statistical measures or heuristics. |
| **Wrapper Method** | Selection | Selects features by training and evaluating a machine learning model with different subsets of features to find the best subset. |
| **Embedded Method** | Selection | Incorporates feature selection as part of the model training process itself, leveraging algorithms that perform feature selection during model building. |
| **Information Gain (IG)** | Selection | Measures the amount of information that a feature provides about the target variable, often used in decision tree-based algorithms. |
| **Fast Correlation Based Filter** | Selection | Selects features based on their correlation with the target variable, considering both individual feature relevance and redundancy between features. |
| **Network Flow Feature Extraction** | Extraction | Extracts features by analyzing the flow of information in a network, identifying important patterns or structural characteristics. |
| **Statistical Feature Extraction** | Extraction | Extracts features by applying statistical operations to the data, such as mean, standard deviation, or other summary statistics. |
| **Kernel Principal Component Analysis** | Extraction | Performs dimensionality reduction by projecting the data into a higher-dimensional space using kernel functions, capturing complex relationships and patterns. |

**Table 8:** Feature Engineering Methods

*ac*

However, according to [42], the use of gradient boosting algorithms like CatBoost, XGBoost, and LightGBM suggest that feature engineering may not be necessary for framework. The reason being that these algorithms have the capability to automatically generate feature importance scores and select features during their training process. This automated feature selection saves both time and resources, eliminating the need for additional feature engineering which is greatly beneficial for a IoT based IDS as it saves computational cost, thus these models will be taken in consideration towards the proposed IDS.

Additionally, [36] has shown the usage of Hyper-parameter optimization (HPO) is the process of building an optimized ML model for a specific problem or dataset using an optimization algorithm. This is shown to be used in both [36] and [42] where Bayesian optimization with tree Parzen estimator (BO-TPE) was used for supervised learning optimization, while a Bayesian optimization with Gaussian process (BO-GP) was used for unsupervised learning optimization. These Bayesian optimizations help tuned the best suited hyper-parameters which in turn improves the optimization for the classifier. This is evident in the result of [36] where the implementation of HPO and Feature Selection has improved the accuracy of the model from 99.861% to 99.879%. Additionally, the execution time has also been decreased by a large margin of 70.2%. In summary, by tuning the hyperparameters with HPO along with Feature Engineer, the performance and efficiency of the model can be greatly improved. This is especially valuable in a centralized IDS structure as resource are more constraint thus model optimization is crucial.

In conclusion, although creating a centralized IDS architecture could help lessen the burden for the resource limitation of an edge IoT devices, it is still a challenge that need to be addressed. The architecture faces problem such as constrained computational capabilities, memory, and network bandwidth pose significant challenges. Efficiently addressing these limitations becomes crucial for the effective implementation of intrusion detection in IoT environments. Thus to address these limitations, several strategies can be implemented:

1. **ML Selection** - The selection of an appropriate machine learning model is crucial for effective intrusion detection in IoT environments. Given the unique characteristics of IoT data, such as its volume, velocity, and heterogeneity, careful evaluation and comparison of different models is necessary Models that are efficient and suitable for resource constrained IoT environments, are the Tree Models (Decision Tree, Random Forest, XGBoost, CatBoost, LightGBM), CL-K or lightweight neural networks (MLP), should be prioritized. Through comprehensive model evaluation and testing, an optimal choice can be made to strike a balance between accuracy and resource utilization.
2. **Datasets Selection** - Selecting a dataset that accurately represents the characteristics and challenges of IoT network traffic is crucial for training an effective intrusion detection model. Datasets such as ToN-IoT and CICIDS2017 fill these requirements as mentioned before.
3. **Data pre-processing** - Pre-processing IoT data is crucial to handle its diverse nature, including noise, missing values, and inconsistencies. Techniques like cleaning, normalization, transformation, and sampling (including oversampling for addressing class imbalance) should be applied to ensure data quality, consistency, and enhance the model's ability to detect rare intrusions. By tailoring these pre-processing techniques to the specific requirements of IoT data, accurate and efficient analysis can be achieved.
4. **Feature engineering -** Feature extraction and selection are crucial for extracting and choosing the most valuable insights from IoT data. These techniques involve transforming and selecting the most informative features that capture intrusion patterns effectively. By leveraging dimensionality reduction, time-series analysis, and domain-specific feature extraction, the intrusion detection system's performance can be enhanced, reducing computational and memory requirements while improving accuracy.
5. **Model optimization -** Once a machine learning model is selected, optimizing its hyperparameters can further enhance its performance and resource utilization. Experiment with different hyperparameter settings to find the optimal configuration that balances model accuracy and resource usage. Fine-tuning hyperparameters can help reduce model complexity and improve efficiency in resource constrained IoT environments.

.

1. **Performance testing** - It is crucial to conduct performance testing of the IoT NIDS system to evaluate its effectiveness and resource utilization. Use representative datasets and realistic intrusion scenarios to assess the system's detection capabilities, computational requirements, and memory usage. Performance testing allows for identifying potential bottlenecks, optimizing system parameters, and ensuring that the IoT NIDS operates efficiently and effectively within the available resources.

By following this systematic approach, starting with choosing the appropriate machine learning model, selecting a relevant dataset, data pre-processing, feature engineering, model optimization, and performance testing, the resource utilization and overall efficiency of the centralized IoT NIDS architecture can be significantly improved allowing it to be better suited for an IoT environment. These strategies enable effective intrusion detection while efficiently managing the resource limitations inherent in IoT environments.

3.0 Methodology

**3.1 Proposed System**

Proposed System –

Model 1: LightGBM + CL-K

Model 2: LightGBM x Decision Tree + CL-K

Model 3: LightGBM x MLP + CL-K

Model 4: LightGBM x DT x MLP + CL-K

If Hybrid Model is proven to be too hard CL-K can be removed

Optimizations include.

**3.2 Chosen Datasets**

The performance of the proposed framework will contain two datasets which would serve as the benchmark for the algorithms, these being ToN-IoT and CICIDS2017.

* **CIC-IDS2017 Dataset -** The CIC-IDS2017 dataset consists of system logs and network traffic captures from various sources. It contains a total of 78 features extracted from the captured network traffic. The dataset does not include the timestamp feature, as it is not considered relevant for anomaly detection or attack category identification. The dataset covers multiple attack types, providing a comprehensive representation of network security threats.
* **ToN-IoT Dataset -** The ToN IoT dataset focuses on network traffic captured from various IoT and IIoT sensors. It includes Linux and Windows system trace datasets collected from hosts running the corresponding operating systems. Specifically, the Windows 10 dataset was obtained using the Performance Monitor Tool on Windows 10 systems. The dataset encompasses activities related to the desk, process, processor, memory, and network on Windows 10 systems. It offers a total of 124 features, excluding the timestamp feature for anomaly detection and attack categorization.

**3.3 Data Pre-Processing**

K-Mean Clustering, Random Sampling, SMOTE, Z-score normalization

**3.4 Feature Engineering**

K-Mean Clustering, Random Sampling, SMOTE, Z-score normalization

**3.5 Model Optimization**

HPO,   
BO-TPE for supervised.

BO-GP for unsupervised

Biased Classifier for CL-K

**3.6 Performance Matrix**

|  |  |  |
| --- | --- | --- |
| **Binary Classification** | | |
| **Metrics** | **Formula** | **Function** |
| **Accuracy** |  | Accuracy calculates the ratio of correctly predicted instances by dividing the sum of TP and TN by the total number of instances. |
| **Precision** |  | Precision measures the accuracy of positive predictions and is computed by dividing TP by the sum of TP and FP. |
| **Recall** |  | Recall evaluates the ratio of correctly identified positive instances by dividing TP by the sum of TP and FN. |
| **F1-Score** |  | F1-score is the harmonic mean of precision and recall. It is calculated using the equations provided, considering TP, FN, and FP. |

In this section, we analyse the outcomes of both binary classification and multi-class classification tasks. In the context of binary classification, four fundamental performance metrics are employed: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). These measures are combined to construct the confusion matrix. Our research focuses on evaluating the performance using these metrics. The key performance metrics listed in Table 9.

**Table 9: Definition of Metrics**

*ac*

These metrics provide valuable insights into the performance of the binary classifier, allowing for a comprehensive evaluation of its effectiveness in terms of accurate predictions, precision, recall, and a balance between precision and recall.

**3.7 Experimental Environment**

All experiments were implemented on a computer with the following specifications:

* **CPU** – AMD Ryzen 9 6900hs
* **GPU** – AMD Radeon RX 6800s
* **RAM** – 32 GB DDR5 RAM
* **OS** – Window 11 Home
* **Tools** – Python 2.1,

**3.7 Model Comparison**

RF, DT, SVM, KNN

Notes:  
(DT Hybrid with CL-K so that It can deal with zero day attacks)

Finding a suitable machine learning model, which can satisfy the requirements of an efficient and lightweight NIDS at the IoT edge, is critical. While Deep Learning (DL) models have been shown to be very successful in the implementation of ML-based NIDSs in general networks [5], they usually need a large amount of memory and compute resources. As such, we considered several Shallow Learning approaches that are known to be less resource intensive, and we decided to use a decision tree model based on its gen

Signature

Anomaly Needs

(Candidate so far (Feed Forwarded Network, MLB) / (Decision Tree, RandomForest, XGBOOST)

Experimental Setup

A picture containing text, screenshot, font

Description automatically generated

A picture containing text, diagram, screenshot, plan

Description automatically generated

A screenshot of a graph

Description automatically generated with low confidenceA picture containing text, diagram, plan, technical drawing

Description automatically generated

A diagram of a flowchart

Description automatically generated with low confidence

A picture containing text, screenshot, font, number

Description automatically generated

A diagram of a data processing model

Description automatically generated with low confidence

**Improvement Points**

* State Traditional IDS Names and Explanation of what it does state that these are used for normal computers not iot devices
* Delve deeper into ML Explanation with charts shown.
* Explore more about the resource constraints of iot devices and the balance that is needed to be made between deep learning and shallow learning. Talk about individual classifier
* Give examples of deep learning
* Raspberry pi for honeypot

4.0 Work Plan and Timeline

**4.1 Gantt Chart**

5.0 Reference

[1] i-SCOOP, What is Iot? the internet of things defined and explained, https://www.i-scoop.eu/internet-of-things-iot/internet-of-things-what-definition/ (accessed May 22, 2023).

[2] A. Bahga and V. Madisetti, "Internet of Things: A Hands-On Approach," VPT, 2014. DOI: 10.1016/C2013-0-16340-2.

[3] D. Uckelmann, M. Harrison, and F. Michahelles, "Architecting the Internet of Things," in Architecting the Internet of Things, 1st ed. Waltham, MA, USA: Morgan Kaufmann, 2011, pp. 1-13. DOI: 10.1016/B978-0-12-375165-2.00001-5.

[4] M. Kranz, "Building the Internet of Things: Implement New Business Models, Disrupt Competitors, and Transform Your Industry," McGraw-Hill Education, 2015. DOI: 10.1036/9780071835247.

[5] Lionel Sujay Vailshery. “Number of IoT connected devices worldwide 2019-2021, with forecasts to 2030”, 2022, url: https://www.statista.com/statistics/1183457/iotconnected-devices-worldwide/.

[6] Fortinet, “Global threat landscape report, 1H 2021”, 2021, url: https://www.fortinet.com/content/dam/fortinet/assets/threat-reports/report-threat-landscape-2021.pdf (accessed May 31, 2023).

[7] F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, "Internet of Things security: A survey," *Journal of Network and Computer Applications,* vol. 88, pp. 10-28, 2017/06/15/ 2017, doi: <https://doi.org/10.1016/j.jnca.2017.04.002>.

[8] AltexSoft, “IOT architecture: The pathway from physical signals to business decisions,” 2020, url: https://www.altexsoft.com/blog/iot-architecture-layers-components/ (accessed Jun. 2, 2023).

[9] E. Kuzmenko, “Three layer architecture in the internet of things. an examination of specifications and security threatsE,” KITRUM, 2023/05/18 2023, url: https://kitrum.com/blog/three-layer-architecture-in-the-internet-of-things/ (accessed Jun. 3, 2023).

[10] BPI, “Securing the internet of things with Intrusion Detection Systems,” BPI,2020/5/13 2020, url: https://www.businessprocessincubator.com/content/securing-the-internet-of-things-with-intrusion-detection-systems/ (accessed Jun. 6, 2023).

[11] Check Point Software, “What is an intrusion detection system (IDS)?,” Check Point Software, 2022/7/15 2022, url: https://www.checkpoint.com/cyber-hub/network-security/what-is-an-intrusion-detection-system-ids/#:~:text=An%20Intrusion%20Detection%20System%20(IDS)%20is%20a%20monitoring%20system%20that,actions%20to%20remediate%20the%20threat. (accessed Jun. 6, 2023).

[12] H. Kozushko, "Intrusion Detection: Host-Based and Network-Based Intrusion Detection Systems," 2003.

[13] A. Khraisat, I. Gondal, P. Vamplewand J. Kamruzzaman, “Survey of intrusion detection systems: techniques, datasets and challenges”, Cybersecurity, vol. 2, no. 1, 2019, doi: 10.1186/s42400-019-0038-7.

[14] Linkedin, “What are the pros and cons of signature-based vs. anomaly-based detection?,” Signature-Based vs. Anomaly-Based Detection: Pros and Cons, url: https://www.linkedin.com/advice/0/what-pros-cons-signature-based-vs-anomaly-based#:~:text=However%2C%20signature%2Dbased%20detection%20has,is%20mistaken%20for%20an%20attack. (accessed Jun. 7, 2023).

[15] MonkeyLearn, "An introduction to machine learning”, MonkeyLearn, url: https://monkeylearn.com/machine-learning/#:~:text=Machine%20learning%20(ML)%20is%20a,to%20make%20their%20own%20predictions. (accessed Jun. 8, 2023).

[16] DataFlair, “Machine learning tutorial - all the essential concepts in single tutorial,” DataFlair, url: https://data-flair.training/blogs/machine-learning-tutorial/ (accessed Jun. 9, 2023).

[17] L. Tagliaferri, “An introduction to machine learning,” DigitalOcean, 2022, url: https://www.digitalocean.com/community/tutorials/an-introduction-to-machine-learning (accessed Jun. 8, 2023).

[18] V. A. Kanade, “What is machine learning? understanding types & applications,” Spiceworks, 2022, url: https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-ml/ (accessed Jun. 7, 2023).

[19] R. Raj, “Supervised, unsupervised and semi-supervised learning with real-life USECASE,” enjoyalgorithms, url: https://www.enjoyalgorithms.com/blogs/supervised-unsupervised-and-semisupervised-learning (accessed Jun. 11, 2023).

[20] S. Loukas, “What is machine learning: A short note on supervised, unsupervised, semi-supervised and...,” Medium, 2023, url: https://towardsdatascience.com/what-is-machine-learning-a-short-note-on-supervised-unsupervised-semi-supervised-and-aed1573ae9bb (accessed Jun. 11, 2023).

[21] IBM, “What is deep learning?”, IBM, url: https://www.ibm.com/topics/deep-learning (accessed Jun. 13, 2023).

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

[23] A. Biswal, “What is bagging in machine learning and how to perform bagging,” Simplilearn.com, 2023, url: https://www.simplilearn.com/tutorials/machine-learning-tutorial/bagging-in-machine-learning (accessed Jun. 15, 2023).

[24] GeeksforGeeks, “Boosting in machine learning: Boosting and AdaBoost,” GeeksforGeeks, 2023, url: https://www.geeksforgeeks.org/boosting-in-machine-learning-boosting-and-adaboost/ (accessed Jun. 15, 2023).

[25] M. Alani, "IoTProtect: A Machine-Learning Based IoT Intrusion Detection System," in 2022 6th International Conference on Cryptography, Security and Privacy (CSP), Tianjin, China, 2022 pp. 61-65. doi: 10.1109/CSP55486.2022.00020

[26] L. Manocchio, S. Layeghy and M. Portmann, "Network Intrusion Detection System in a Light Bulb," in 2022 32nd International Telecommunication Networks and Applications Conference (ITNAC), Wellington, New Zealand, 2022 pp. 1-8. doi: 10.1109/ITNAC55475.2022.9998371

[27] M. Ge, et al., "Deep Learning-Based Intrusion Detection for IoT Networks," in 2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC), Kyoto, Japan, 2019 pp. 256-265. doi: 10.1109/PRDC47002.2019.00056

[28] D. Chauhan, M. Shah and H. Joshi, "A Novel Intrusion Detection System based on Machine Learning for Internet of Things (IoT) Devices," in 2023 3rd International Conference on Smart Data Intelligence (ICSMDI), Trichy, India, 2023 pp. 427-434. doi: 10.1109/ICSMDI57622.2023.00081

[29] T. Saba, T. Sadad, A. Rehman, Z. Mehmood and Q. Javaid, "Intrusion Detection System Through Advance Machine Learning for the Internet of Things Networks" in IT Professional, vol. 23, no. 02, pp. 58-64, 2021. doi: 10.1109/MITP.2020.2992710

[30] M. Alani, E. Damiani and U. Ghosh, "DeepIIoT: An Explainable Deep Learning Based Intrusion Detection System for Industrial IOT," in 2022 IEEE 42nd International Conference on Distributed Computing Systems Workshops (ICDCSW), Bologna, Italy, 2022 pp. 169-174. doi: 10.1109/ICDCSW56584.2022.00040

[31] S. Li, Y. Lu and J. Li, "Can We Half the Work with Double Results: Rethinking Machine Learning Algorithms for Network Intrusion Detection System," in 2021 Ninth International Conference on Advanced Cloud and Big Data (CBD), Xi'an, China, 2022 pp. 242-247. doi: 10.1109/CBD54617.2021.00049

[32] D. Priya, A. Kiran and P. Purushotham, "Lightweight Intrusion Detection System(L-IDS) for the Internet of Things," in 2022 International Conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC), Bhubaneswar, India, 2022 pp. 1-4. doi: 10.1109/ASSIC55218.2022.10088328

[33] R. Kale, Z. Lu, K. Fok and V. Thing, "A Hybrid Deep Learning Anomaly Detection Framework for Intrusion Detection," in 2022 IEEE 8th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS), Jinan, China, 2022 pp. 137-142.doi: 10.1109/BigDataSecurityHPSCIDS54978.2022.0003

[34] J. Ashraf, N. Moustafa, A. Bukhshi and A. Javed, "Intrusion Detection System for SDN-enabled IoT Networks using Machine Learning Techniques," in 2021 IEEE 25th International Enterprise Distributed Object Computing Workshop (EDOCW), Gold Coast, Australia, 2021 pp. 46-52. doi: 10.1109/EDOCW52865.2021.00031

[35] E. Ciklabakkal, et al., "ARTEMIS: An Intrusion Detection System for MQTT Attacks in Internet of Things," in 2019 38th Symposium on Reliable Distributed Systems (SRDS), Lyon, France, 2019 pp. 369-3692. doi: 10.1109/SRDS47363.2019.00053

[36] L. Yang, A. Moubayed and A. Shami, "MTH-IDS: A Multitiered Hybrid Intrusion Detection System for Internet of Vehicles," in IEEE Internet of Things Journal, vol. 9, no. 1, pp. 616-632, 1 Jan.1, 2022, doi: 10.1109/JIOT.2021.3084796.

[37] R. K. Vigneswaran, R. Vinayakumar, K. P. Soman and P. Poornachandran, "Evaluating Shallow and Deep Neural Networks for Network Intrusion Detection Systems in Cyber Security," 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Bengaluru, India, 2018, pp. 1-6, doi: 10.1109/ICCCNT.2018.8494096.

[38] B. Xue, H. Zhao and W. Yao, "Deep Transfer Learning for IoT Intrusion Detection," in 2022 3rd International Conference on Computing, Networks and Internet of Things (CNIOT), Qingdao, China, 2022 pp. 88-94. doi: 10.1109/CNIOT55862.2022.00023

[39] X. Li, R. Tang and W. Song, "Intrusion Detection System Using Improved Convolution Neural Network," in 2022 11th International Conference of Information and Communication Technology (ICTech), Wuhan, China, 2022 pp. 97-100. doi: 10.1109/ICTech55460.2022.00028

[40] X. Cheng, W. Li, Z. Xiao and T. Zhao, "Intrusion detection system based on QBSO-FS," in 2020 International Conference on Artificial Intelligence and Computer Engineering (ICAICE), Beijing, China, 2020 pp. 372-377. doi: 10.1109/ICAICE51518.2020.00078

[41] G. Guo, "An Intrusion Detection System for the Internet of Things Using Machine Learning Models," 2022 3rd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE), Xi’an, China, 2022, pp. 332-335, doi: 10.1109/ICBAIE56435.2022.9985800.

[42] L. Yang, A. Shami, G. Stevens, and S. DeRusett, “LCCDE: A Decision-Based Ensemble Framework for Intrusion Detection in The Internet of Vehicles," in 2022 IEEE Global Communications Conference (GLOBECOM), 2022, pp. 1-6, doi: 10.1109/GLOBECOM48099.2022.10001280

[43] S. M. Kasongo, "An Advanced Intrusion Detection System for IIoT Based on GA and Tree Based Algorithms," in IEEE Access, vol. 9, pp. 113199-113212, 2021, doi: 10.1109/ACCESS.2021.3104113.

[44] Singh, S., Fernandes, S. V., Padmanabha, V., & Rubini, P. (2021). MCIDS-Multi Classifier Intrusion Detection system for IoT Cyber Attack using Deep Learning algorithm. 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV). doi:10.1109/icicv50876.2021.9388579

[45] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac and P. Faruki, "Network Intrusion Detection for IoT Security Based on Learning Techniques," in IEEE Communications Surveys & Tutorials, vol. 21, no. 3, pp. 2671-2701, thirdquarter 2019, doi: 10.1109/COMST.2019.2896380.

[46] I. Ullah and Q. H. Mahmoud, "Design and Development of a Deep Learning-Based Model for Anomaly Detection in IoT Networks," in IEEE Access, vol. 9, pp. 103906-103926, 2021, doi: 10.1109/ACCESS.2021.3094024.

[47] G. Guo, X. Pan, H. Liu, F. Li, L. Pei and K. Hu, "An IoT Intrusion Detection System Based on TON IoT Network Dataset," 2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2023, pp. 0333-0338, doi: 10.1109/CCWC57344.2023.10099144.

[48] Elsayed, R., Hamada, R., Hammoudeh, M., Abdalla, M., & Elsaid, S. A., “A Hierarchical Deep Learning-Based Intrusion Detection Architecture for Clustered Internet of Things.” Journal of Sensor and Actuator Networks, 2023. doi: 10.3390/jsan12010003

[49] Fenanir, S., Semchedine, F., Baadache, A., “A machine learning-based lightweight intrusion detection system for the internet of things.” Revue d'Intelligence Artificielle, Vol. 33, No. 3, pp. 203-211., 2019, doi: 10.18280/ria.330306