# Enhancing Computational Efficiency and Time Optimization in Cloud IoT Intrusion Detection Using ANN and Hybrid Deep Learning

MSc Research Project
Cloud Computing

Swathy Menon Balachandran
Student ID: X23108568

School of Computing
National College of Ireland

Supervisor: Shaguna Gupta

**National College of Ireland**

**MSc Project Submission Sheet**

**School of Computing**

| | |
|---|---|
| **Student Name:** | …Swathy Menon Balachandran……..…………………………………………….… |
| **Student ID:** | …X23108568……………………………………………………………………… |
| **Programme:** | …MSc In Cloud Computing…………………………... **Year:** …2024…………….. |
| **Module:** | … MSc Research Project…………………………………………………….……… |
| **Supervisor:** | …Shaguna Gupta……………………………………………………………….……… |
| **Submission Due Date:** | …16/09/2024………………………………………………………………………….……… |
| **Project Title:** | Enhancing Computational Efficiency and Time Optimization in Cloud IoT Intrusion Detection Using ANN and Hybrid Deep Learning ……………………………………………………………………………….……… |
| **Word Count:** | ……8779…………………… **Page Count**……26………………………………………….…….. |

**Signature:** …Swathy Menon Balachandran……………………………………………………………….

**Date:** …16/09/2024………………………………………………………………………………………

**PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST**

| | |
|---|---|
| Attach a completed copy of this sheet to each project (including multiple copies) | □ |
| **Attach a Moodle submission receipt of the online project submission,** to each project (including multiple copies). | □ |
| **You must ensure that you retain a HARD COPY of the project**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. | □ |

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

| **Office Use Only** | |
|---|---|
| Signature: | |
| Date: | |
| Penalty Applied (if applicable): | |

# Enhancing Computational Efficiency and Time Optimization in Cloud IoT Intrusion Detection Using ANN and Hybrid Deep Learning

Swathy Menon Balachandran
X23108568

**Abstract**

Internet of Things (IoT) networks are essential across various domains in today's interconnected world including smart homes and industrial systems but they face significant security challenges due to their vulnerability to cyberattacks. This research presents a cloud-integrated intrusion detection system (IDS) specifically developed for IoT networks using advanced machine learning (ML) techniques to enhance security and efficiency. The system employs Artificial Neural Networks (ANN) for the rapid initial classification of normal and suspicious activities. Following this, a hybrid deep learning model that combines Convolutional Neural Networks (CNN) and Gated Recurrent Units (GRU) performs detailed analysis and classifies specific threats. This two-level approach optimizes resource usage by reserving computationally demanding processes for potentially malicious data. The architecture is developed to manage the substantial data volumes generated by IoT devices achieving 90.79% accuracy in initial classification and 74.32% in multiclass threat detection. It integrates with a Python flask application and deployed on Amazon Web Services (AWS) ensuring faster threat detection and response. Specifically, the AWS deployment reduced the threat detection time to approximately 0.33 seconds for normal traffic and around 1.92 seconds for attack files. This research highlights the significance of using advanced ML models that integrate with cloud-based solutions for enhance overall network resilience and reliability to address the unique security challenges in IoT environments. Future work required to validate the IDS using different datasets with regular updates to detect emerging threats effectively.

Keywords: Internet of Things, Intrusion Detection System, Cyber threats, Machine Learning, Hybrid Deep Learning

# 1 Introduction

## 1.1 Background

The increasing adoption of IoT networks across different industries, including smart homes and industrial systems, has introduced significant security vulnerabilities. Traditional IDS face limitations in effectively protecting resource constrained IoT devices and rapidly adapting to evolving cyber threats. This research aims on developing a Cloud-Integrated IDS that leverages advanced ML technologies to improve IoT network security. The proposed system focuses to accurately identify and classify potential intrusions, addressing the limitations of traditional IDS systems and providing a scalable, efficient approach to maintaining IoT network integrity. The IoT enables networks and systems to connect devices and makes simpler for both humans and the devices themselves to communicate. IoT providing real-time data across various domain including buildings, humans, animals, environment and plants by integrating physical objects into an information network. Security remains an important concern due to threats at each layer such as attacks like data forging and device destruction that may impact the perception layer, routing attacks and availability issues at the network layer and risks related to software attacks on the application layer.

IoT architecture consists of three layers (Figure 1):

- **Perception Layer:** Also referred to as Sensor Layer, it is responsible for creating a connection between devices and the IoT network by collecting and processing data.
- **Network Layer:** Also referred to as Transmission Layer, it makes use of integrated networks with technologies like Bluetooth, Wi-Fi and LTE to transmit data collected by the perception layer to IoT servers, devices and applications.
- **Application Layer:** Often referred to as the Business Layer, it uses data to provide a variety of services (Larriva-Novo et al., 2021).

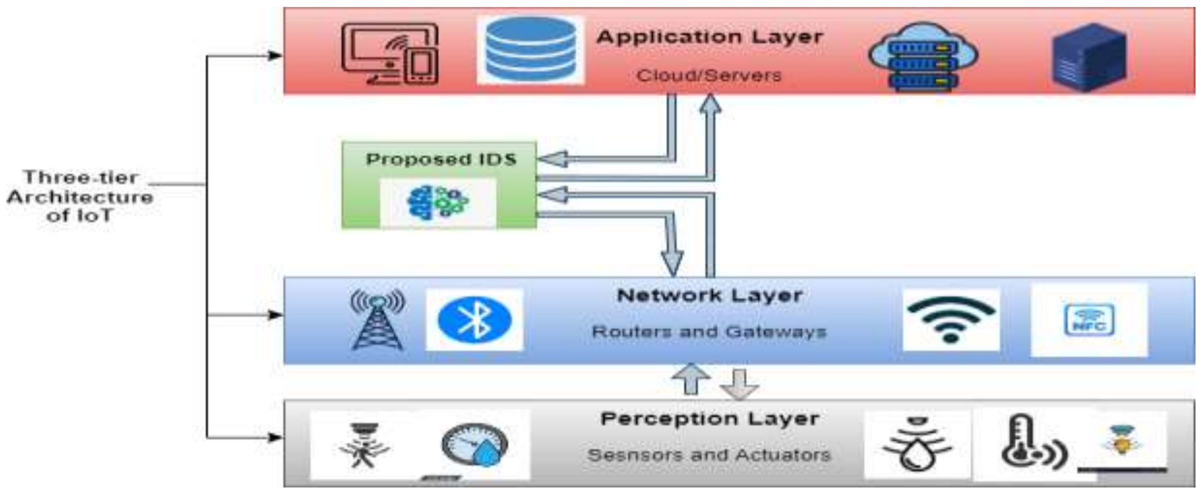


**Figure 1: Three-tier architecture of IoT and IoT IDS approach**
**(Source: Adapted from Larriva-Novo et al., 2021. Created by learner)**

Cloud-based IDS enhance IoT security by monitoring network and system activities to detect unauthorized access and attacks. There are many advantages associated with this integration particularly in terms of security and operational efficiency. Centralized monitoring and management of security threats over a large network of IoT devices is made easier by the IDS in a cloud-based IoT environment. Centralization ensures that malicious activities are swiftly addressed, reducing response time and improving defensive measures coordination. Advanced ML algorithms can be applied IDS to continually enhance threat detection accuracy and adapt to emerging security threats.

## 1.2 Problem Statement

A comprehensive review of the literature highlights the limitations of current IDS systems. Previous studies show that advanced technologies are needed to combat the ever-changing cyber threat landscape. However, challenges remain in protecting resource-constrained devices and rapidly understanding threats in IoT environments. Through the integration of a two-level classification approach, the proposed system enhances computational efficiency and scalability, making it suitable for resource constrained IoT environments. By initially filtering out normal behavior using fewer complex models, it optimizes processing time and allows computationally intensive deep learning models to focus on potential malicious activities. To enhance flexibility and scalability, this study seeks to gather expert recommendations and develop a new IDS infrastructure specifically designed for IoT hosted cloud devices. The primary research question focuses on optimizing data pre-processing and feature selection techniques to enhance the accuracy of the UNSWNB15 dataset in detecting normal behavior and various types of attacks within IoT networks including Denial of Service (DoS), Exploits, Fuzzers and Reconnaissance. One significant limitation of using this dataset is that it may not cover all possible attack types and vectors present in real-world IoT environments. Therefore, to ensure the system resilience and efficiency across various IoT scenarios, it is essential to validate it using different datasets.

## 1.3 Problem Solution

Since IoT networks are constantly evolving and dynamic, maintaining security is essential to protect sensitive data and critical infrastructures. New IDS designed for IoT environments are urgently required as sophisticated cyber threats continue to grow. With the development of a cloud-integrated IDS for IoT networks, this research seeks to address these specific challenges. The system integrates with a Python Flask application and is deployed on AWS, providing practical solutions for IoT security.

## 1.4 Research Questions

- What data pre-processing techniques can be applied to the UNSW-NB15 dataset to enhance the accuracy and efficiency of intrusion detection in IoT networks?
- Which feature selection methods are most effective in identifying relevant features from the UNSWNB15 dataset for distinguishing between normal behavior and various types of attacks such as DoS, Exploits, Fuzzers, Reconnaissance in IoT networks?
- How can a two-level classification approach be designed and implemented to optimize the detection of normal behavior and various types of attacks in IoT networks, ensuring computational efficiency and time optimization?

## 1.5 Ethical Consideration

The proposed work presents several issues regarding ethics, including:

• **Data privacy**: IoT devices collect sensitive data so in order to ensuring the protection of privacy, anonymization and secure data storage are important processes.

• **Bias in Machine Learning**: ML model biases based on the data they are trained on. It requires assessment procedures and mitigation processes to ensure fair and unbiased intrusion detection.

• **False Positives and Negatives**: Accurate identification is essential because false negatives expose the system to be actively vulnerability while false positives waste resources.it is important to balance the risks of false positives and false negatives in order to ensure well-functioning IDS.

• **Transparency and Explainability**: The complexity of deep learning decision-making processes presents another challenge. These processes are murky and more likely to cause issues, especially if the devices are falsely detected. To overcome this, model interpretability efforts will be implemented as action.

## 1.6 Structure of the Report

The main report consists of three sections which include the introduction that provide a brief background, outline the primary objectives, research questions and addresses ethical considerations which focus on enhancing the security in IoT network. This is followed by a literature review that identifies the shortfalls in existing IoT methodologies and IDS approaches. Furthermore, the research design and methodology section provide the architecture of the proposed system, integration with Python Flask and deployment on AWS.

# 2 Related Work

This literature review critically analyses the intrusion detection techniques in IoT systems, emphasizing the variety of approaches and identifying the areas for improvement. It shows the significance contemporary techniques are to improving IoT security against constantly evolving cyber threats.

## 2.1 Comprehensive Approaches to Intrusion Detection in IoT Systems

Intrusion detection in IoT systems has become an important area of research due to the unique security challenges posed by these environments. To enhance security protocols various studies have explored different methods which focusing on both systems wide and node specific strategies. The review of the literature on intrusion detection techniques in IoT systems highlights four key works that have significantly contributed to the understanding and enhance intrusion detection in autonomous distributed IoT environments.

Al-Hamadi et al. (2020) explored into the complexities of intrusion detection and defense strategies within IoT systems. Their study emphasizes the need of enhancing system resilience against different types of attacks. They analyses the interactions between different attack methods and safety measures in IoT environments by using Stochastic Petri Net (SPN) modelling. The study highlighting the need for lightweight detection methods across nodes.

In a more focused study, Nallakaruppan et al. (2024) covers the enhancement of host-based IDS specifically in IoT settings advocates for the use of ML techniques for attack classification. By concentrating on host-level security, this study provides a robust framework for classifying and responding to attacks, contrasting with the broader approach by Al-Hamadi et al. The use of ML methods in this context offers a detailed perspective on developing resilient security measures at the host level.

Kandhro et al. (2023) present a method for real-time IDS in IoT networks by leveraging deep learning techniques. Their research focus on the development of deep learning classifiers tailored for IoT driven cyber-physical systems demonstrating significant improvements in detection accuracy and efficiency. This work highlights real-time threat detection distinguishes this study from the lightweight mechanisms proposed by Al-Hamadi et al. and the attack classification approach by Nallakaruppan et al.

Laassar et al. (2024) present the Binary Search Equation Enhanced Artificial Bee Colony Algorithm (BABCN) in order to enhance network intrusion detection (NID) in cloud computing environments. Their method showed promising results in terms of efficiency and real-world applicability though it is computationally intensive. When compared to methods like Random Forest(RF) and CNN, BABCN demonstrates the importance of selecting appropriate techniques to achieve optimal accuracy and efficiency. Although it also emphasizes the need for further exploration of robustness and applicability, this study shows the potential of advanced algorithms in improving detecting capabilities.

Different methods for detecting intrusions in IoT systems are shown in the critical evaluation of these research. The complexity of IDS techniques in IoT systems is highlighted by this analysis which also emphasizes the need for an integrated strategy which combines several techniques in order to address dynamic threats. The research gaps identified suggest a need for developing unified IDS that leverage the strengths of different techniques to secure IoT environments against a wide range of cyberattacks, enhancing the overall robustness and reliability of IoT networks.

## 2.2 Advancements in Intrusion Detection Systems for IoT Environments

Zhou et al. (2021) present a critical evaluation of previous research projects focused on improving IoT Security specifically NID. They emphasize the rapid expansion of IoT which raises serious security threats due to the massive amount of data involved. Existing intrusion detection techniques frequently struggle due to insufficient and limited attack data used for training, leaving systems vulnerable to unknown attacks. Zhou et al. suggest an innovative approach to intrusion detection known as hierarchical adversarial attack (HAA) technology, targeting graph neural network (GNN) based intrusion detection. This technique uses a level-aware black box adversarial attack strategy to update significant feature points with minor modifications using a saliency map approach and a shadow GNN model. Its showing a significant reduction in classification precision of more than 30%. The conclusion highlights the importance of effective IDS to safeguard IoT devices as their use expands.

While addressing the challenges posed by evolving cyber threats, Fatani et al. (2021) explore an AI-based IDS proposed using deep learning and metaheuristic optimization techniques focused TSODE feature selection method that improves the transient search optimization (TSO) algorithm with differential evolution (DE) operators. It also advances the use of CNNs for feature extraction. The efficiency of this approach is validated using various publicly available datasets outperforming previous methods indicates the importance of combining deep learning and metaheuristic optimization techniques to improve intrusion detection accuracy.

Thamilarasu et al. (2020) analyze a specific field where safety issues related to IoT integration in healthcare are explored. They define a cell agent-based IDS with medical devices specifically designed for networks. The system implements a self-sufficient and hierarchical methodology, leveraging ML and regression techniques to identify anomalies presents optimal detection accuracy with limited resource usage over long experiments. In the conclusion, the significance of IDS in addressing security issues on the Internet of Medical Things domain as well as the need of selecting algorithms that balance computational efficiency and accuracy in intrusion detection are highlighted.

Vibhute and Nakum (2024) explore the challenges of maintaining cloud network security and the requirement for improved anomaly detection technologies. They suggest using modern techniques like deep learning and CNNs to quickly recognize threats. They highlight the significance of combine efficient feature selection techniques with deep learning to enhance precision and efficiency of anomaly detection. This combination improves quick threat detection and helps in handling cloud network security challenges.

Ali et al. (2024) discuss the significance of IDS for cloud computing and shows the importance of secure cloud infrastructures from evolving cyber threats. They concentrate on how deep learning systems such CNNs automatically recognize threats by using raw data. According to their research, IDS that utilize CNN can effectively identify threats while reducing false positives. However, they focus to challenges including the requirement for various datasets ongoing upgrades to handle emerging threats and further research to ensure the scalability of CNN-based IDS in various cloud settings.

This analysis highlights the requirement for an integrated strategy which includes a variety of techniques in order to address dynamic threats. The research gaps indicate the necessity for developing integrated IDS that make use of the advantages of various methodologies. By protecting IoT environments from various threats, these technologies would improve the overall robustness and reliability of IoT networks.

## 2.3 Novel Approaches in Intrusion Detection for IoT Environments

Rani et al. (2023) carried out an IoT-based study on intrusion detection in smart homes and observed that ML approaches such Light Gradient Boosting Machine, RF, Extreme Gradient Boosting, and Logistic Regression might lead to desirable outcomes. Their model ‘LGB-IDS’ showed great rapidity and precision in identifying intrusions.

Another study conducted by Maghrabi (2024) focused on intrusion detection in IoT networks using Network Intrusion Detection (NID) techniques by the UNSWNB15 dataset and the RF method to achieve good accuracy, shows the importance of proper data preparation and balancing for reliable detection. While Maghrabi aimed on accuracy and Rani et al. on speed, and both research highlighted the important role of ML in IoT security.

The efficiency of traditional intrusion detection techniques is reduced by a variety of protocols and limited resources, as highlighted by Roy et al. (2022) in their analysis of security challenges in IoT networks. They developed a lightweight intrusion detection model based on ML to meet the requirements of resource-constrained IoT environments, enhancements such as complexity reduction, sampling, and the removal of redundant data, aiming to balance performance with the inherent limitations of IoT devices.

Meanwhile, Bakhsh et al. (2023) explored deep learning algorithms for IoT security such as Feed Forward Neural Networks (FFNN), Random Neural Networks (RandNN), and Long Short-Term Memory (LSTM). However, they highlighted the need for huge computational resources and a variety of datasets. Roy et al. aimed to develop lightweight models that were adapted to IoT constraints providing various solutions for specific security issues, whereas Bakhsh et al. concentrated on improving accuracy.

Long et al. (2024) conducted research on NIDS in cloud computing environments, indicate the necessity for strong defenses due to the distributed architecture of cloud computing and the ever-changing threat landscape. Although they presented the novel use of the Transformer model, there are still limitations with computing demands and adaptability in different settings.

Arthi et al. (2024) highlight the urgent need for robust safety measures in cloud computing and using a hybrid model of RF and Deep Neural Networks (DNNs) for the significant improvements in cloud security. Further analyses and improvements are required to ensure consistent performance across diverse datasets.

A comprehensive review of previous studies shows the limitations of current intrusion detection techniques in IoT systems and underscores the significance of developing innovative approaches to address constantly evolving security threats. IoT network security and resilience against cyberattacks can be improved by combining ML and deep learning techniques with conventional methods, enhancing overall robustness and reliability.

## 2.4 Summary of Literature Review

The literature review covers a lot of detail about several intrusion detection techniques in IoT systems, pointing out the need of using a variety of methods from lightweight approaches to deep learning. They additionally bring out a variety of limitations with current intrusion detection models such as large usage of static techniques, unbalanced training data, and difficulties in resource-constrained IoT applications.

These drawbacks indicate the importance for novel approaches, especially for real-time threat detection and managing large amounts of data, that are aligned with our cloud-based IDS project's objectives. Our research involves experimenting with different ML models to find the optimal balance between accuracy and computational resources. Moreover, our approach is consistent with the literature's focus on enhancing intrusion detection techniques against emerging cyber threats.

## 2.5 Literature Review Table

Table 1: Literature Review Table

| Author | Framework | Scenario | Dataset | Tools | Metrics | Advantages | Limitations |
|---|---|---|---|---|---|---|---|
| Long et al. (2024) | Transformer-based NIDs approach | Cloud security | CIC-IDS 2018 | Programming Language Python DL Framework: Pytorch | Accuracy Precision Recall F1-score | Use Transformer model for cloud security: achieves high accuracy and adapts effectively to evolving threats. | Still potential for model to generate false positive alerts Further evaluation on larger datasets is needed to fully assess its performance and generalizability |
| Vibhute and Nakum (2024) | DL based network anomaly detection, classify network intrusions | Anomaly Detection in IoT Using Optimized CNN | CSE-CICIDS2018 | Python 3.10, Anaconda, Jupyter Notebook. | Accuracy Precision Recall F1-Score | High accuracy in detecting network anomalies, suitable for real-time Industry 4.0 systems. | Detecting network anomalies in Industry 4.0 is challenging due to high traffic, large data volumes, diverse environments, and the integration of IT and operational systems. |
| Ali et al. (2024) | CNN approach to (IDS) | Extracting valuable features from network traffic data to enhance cloud security | Network data including packet headers, payload data,communication metadata. | Primary tool used is CNN | Accuracy Precision Confusion Matrix Recall F1Score | CNNs provide advanced pattern recognition and feature extraction, offering high adaptability to evolving attacks. They reduce false positives, lessening the workload on security teams. | Addressing challenges related to data diversity, model scalability, computational efficiency. |
| Arthi et al. (2024) | Cloud based IDS using RF | Cloud based IDS | NSLKDD | Python3 Colab, Pytorch | Accuracy Precision Recall F1 Score | Scalability and adaptability evolving threats Real-time implementation and monitoring | Model is overfitting the training data Human oversight and interaction necessary for effective operation |
| Laassar et al. (2024) | Optimized feature selection-based IDS using the BABCN algorithm. | Cloud Computing IoT | NSLKDD | ML algorithms [No mentioned ] | Accuracy, True Positive, False Positive Rate, Elapsed Time | Improved of performance in terms convergence, communication speed, and IDS accuracy | Improving the computational efficiency Dealing with imbalanced multi-category traffic data. |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Maghrabi (2024) | NIDs using Random Forest algorithm | NIDs in IoT networks | UNSW-NB15 | Google Colab | Accuracy Precision Recall F1 Score | Effective in distinguishing between normal and malicious activities. Appropriate data preparation and balancing | Focus on accuracy might not address real-time performance issues Potential for false negatives and false positives |
| Nallakaruppan et al. (2024) | Enhancing Security of Host-Based IDS for the IoT: ML | IoT environments | CTU-13 dataset | Machine Learning [No specific tool mentioned] | Accuracy precision, recall, false alarm rates, F1scor e | Leverages ML algorithms to automatically classify various types of attacks Designed to protect IoT devices and networks from cyber threats. | High computational complexity of NIDS implementation |
| Kandhro et al. (2023) | Real-time IDS for IoT cybersecurity using deep learning. | IDS for detecting malicious intrusions and attacks in IoT-based cybersecurity | NSLKDD KDDCup99, UNSW-NB15 | RNN, CNN, DNN and generative models (RBN, DBN, DBM, DA) | Accuracy, True Negative Rate, Precision, Recall, F1 Score, False Positive Rate | High accuracy and efficiency in detecting various types of cyberattacks, robust against new types of intrusions, maintains confidentiality and integrity during training/testing | Requires large datasets for training, computationally intensive, potential l overfitting, performance may vary with different datasets |
| Rani et al. (2023) | Design IDS for IoT Enabled Smart Home | IoT enabled smart home security with various types of attacks. | DS2OS | LGBM Classifier | Accuracy time efficiency error rate, TPR, FNR | High accuracy in attack detection, low latency using LGBM classifier. | Dependency on labeled data for supervised learning, overfitting in high accuracy scenarios. |
| Bakhsh et al. (2023) | DL-based IDS using FFNN, LSTM, RandNN | Intrusion detection in IoT networks | CIC-IoT22 | Python, Jupyter Notebook, Keras, CICFlow Meter4.0 | Accuracy: FFNN LSTM RandNN | High accuracy in intrusion detection, robust preprocessing and feature selection, | Computational complexity, potential overfitting, dependency on dataset quality and preprocessing |
| Zhou et al. (2021) | Novel hierarchical adversarial attack generation method | Black box adversarial attacks against GNN based IoT NIDS | UNSW - SOSR2019 | CentOS 8, GTX 1070, G39030 Dual Core, 16-GRAM, Python3.6 PyTorch 1.4 | Attack effectiveness, cross-entropy loss, classification performance degradation. | Operates in a black-box scenario, efficient Node Selection for Attacks, Hierarchical Consideration | Limited to specific Dataset, require a lot of computing power and time |
| Roy et al. (2022) | Intrusion detection model using ML for IoT networks | Detecting cyberattacks, anomalies in resource constraint IoT networks | CICIDS2017, NSLKDD | ML (PCA, Boosting, Stacking) | Detection Rate, False Alarm Rate, Accuracy, Precision, Recall, F1score | High detection rate, low false alarm rate | Performance may vary with dataset |
| Fatani et al. (2021) | IoT IDS using DL and Transient Search Optimization (TSODE) | Intrusion detection in IoT systems | BoT- IoT and CICIDS-2017. | CNN and TSODE | Average Accuracy Average Recall Average Precision Performance Improvement Rate | Efficient DL feature extraction and advanced metaheuristic for improved feature selection enhance intrusion detection accuracy across IoT datasets. | Not provide details on the computational complexity and scalability |
| Al-Hamadi et al. (2020) | Analytical model based on SPN modeling techniques | Intrusion detection in autonomous distributed IoT systems | Dataset: ADIoT with 128 nodes | SPNP package to define and analytically solve the SPN mode | System lifetime: Mean Time to Failure (MTTF) | Analyzes attack-defense dynamics and optimal strategies to maximize system lifetime | Scalability of the proposed approach for large-scale ADIoTSs with thousands or millions of nodes is not addressed. |
| Thamilarasu et al. (2020) | Autonomous mobile agent-based IDS | IDS in wireless body area networks in Internet of Medical Things | Not specified | Naïve Bayes Classifier Nearest Neighbors, RF, SVM, Decision Tree | Accuracy, Cost Benefit Ratio Feedback Reliability Value, Training Time | High detection accuracy, mobile agents can increase with the network size | increased computational and communication overhead Sensitivity to network fluctuations, as the system relies on the cluster heads for agent instantiation and dispatch |

# 3 Research Methodology

## 3.1 Research Flow

Steps of Research Methodology for Developing an IDS in IoT Networks using CRISPDM Methodology are as follows and also presented in Figure 2:

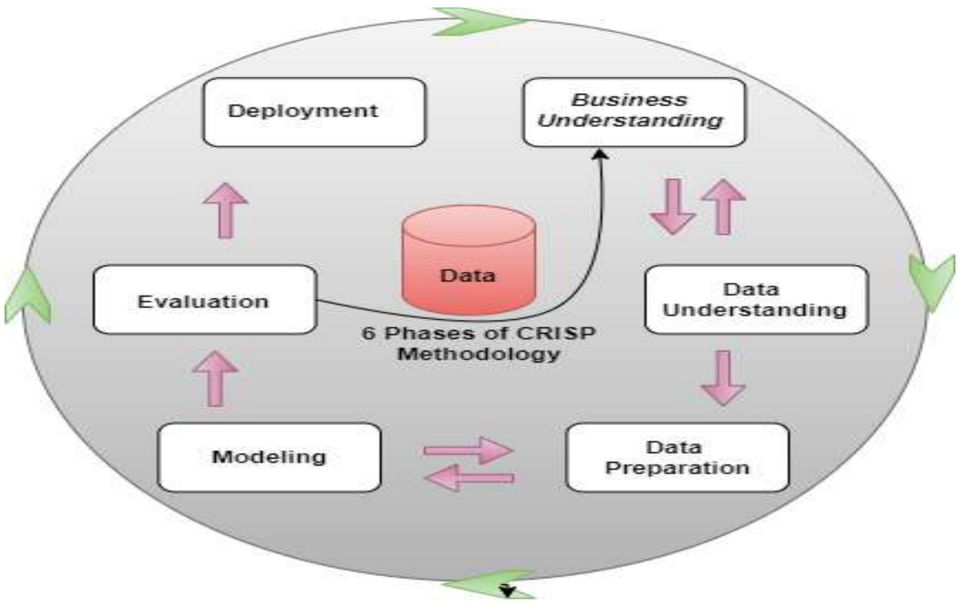


**Figure 2: CRISM-DM System Flow**
**(Source: Adapted from Talaviya (2023), Created by learner)**

- **Understanding Objectives:** The initial stage includes understanding the objectives of the project focusing on the specific needs of stakeholders and identifying the various types of attacks that are most relevant to IoT environments. It also involves setting specific, measurable objectives for the IDS like detection accuracy, response times and resource utilization limits.
- **Data Exploration**: In this stage, the structure of the UNSW-NB15 dataset is examined to get insights into its features, quality and potential issues with the data. This involves a detailed analysis of the dataset to find the importance of these features for intrusion detection. And also identified and highlighted any issues with the data quality of such as outliers, inconsistencies and missing values which could impact the performance of the IDS.
- **Data Preparation**: This includes cleaning the data to ensure a clean dataset for model training by removing or imputing missing values, correcting inconsistencies and handling outliers. It also includes selecting appropriate features for intrusion detection and scaling and normalizing the data that may include converting categorical variables to numerical ones to ensure that ML models can effectively learn from the input features.
- **Model Development**: ML and deep learning models are created to effectively detect both normal behaviour and various types of attacks in IoT networks. This includes selecting proper algorithms such as ANN, CNN and GRU. Selected models are then trained using the prepared dataset.
- **Evaluation Process:** The performance of the model is evaluated using metrics such as accuracy and precision to ensure that certain quality requirements are met. This involves computing standard metrics to evaluate model performance on a validation dataset.

Additionally, to verify that the models exhibit robust and generalizable performance, different techniques were applied.

- **Deployment Phase**: This phase focus on integrating the models into the network in order to enable real time intrusion detection capability. It also ensures that across various network conditions and attack scenarios the IDS function effectively (Talaviya, 2023).

## 3.2 Dataset Description

This research utilizes the UNSW-NB15 dataset sourced from dataworld (https://data.world/victorpuli/useful-unsw-nb15-data) to support the development and evaluation of IDS in IoT environments. The raw network packets of this dataset were created by IXIA PerfectStorm tool in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS). The attack label in this dataset provides a clear classification of network activities into normal traffic and different attack types such as Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms. By leveraging the comprehensive and well pre-processed dataset which contains 192,111 records and 45 features this research focus on to develop a robust IDS capable of effectively detect and mitigating security threats in IoT environments, addressing the dynamic and complex nature of modern cyber threats.

## 3.3 Proposed Approach and Architecture

The research leverages the CRISP-DM methodology and utilizes the UNSW-NB15 dataset to build the IDS for IoT networks. The methodology includes stages such as understanding objectives, data exploration, data preparation, model development, evaluation process and deployment phase. The systematic design and evaluation of the IDS models is ensured by this structured approach. The proposed system architecture is illustrated in Figure 3.

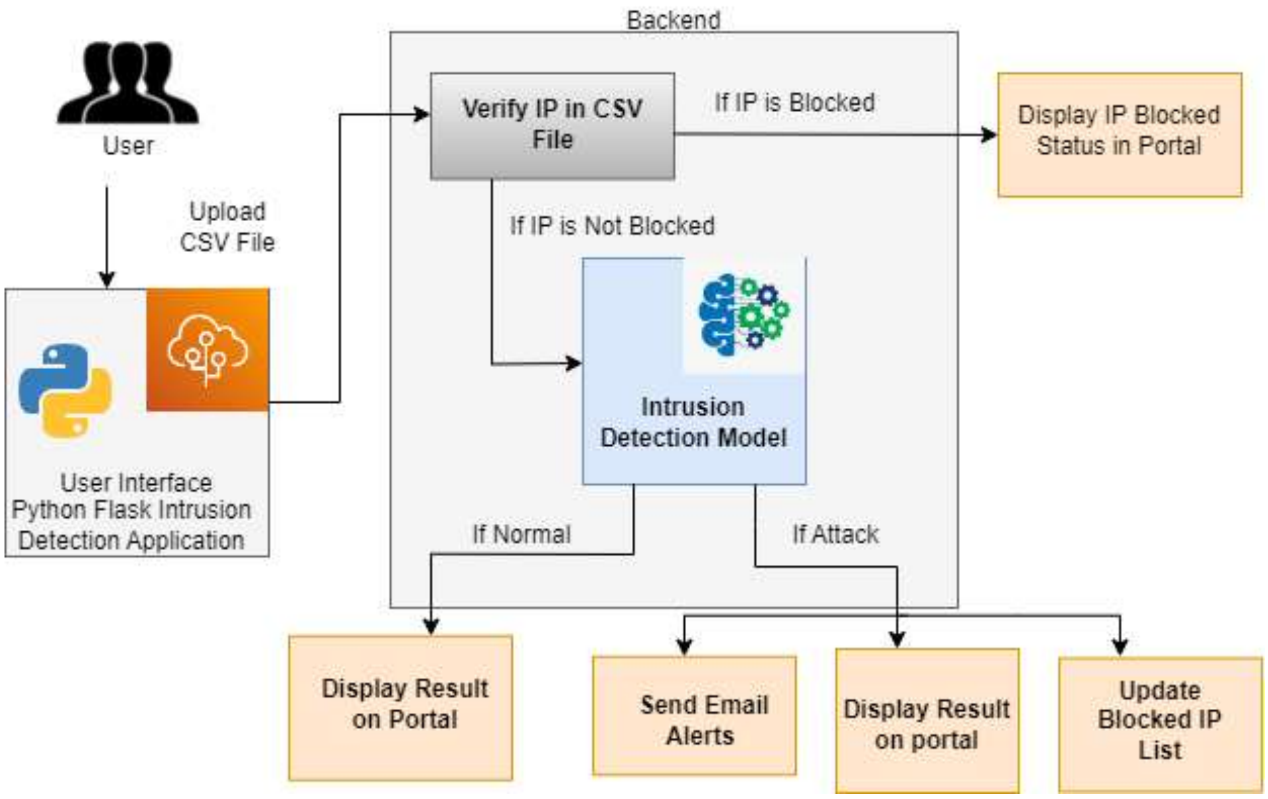


**Figure 3: Proposed System Architecture (Source: Created by learner)**

The workflow of the model is as follows:

- **Data Collection and Pre-processing**: After preprocessing, data collected from IoT devices is fed into ML models for analysis. This includes clearing irrelevant data, replacing missing values, and scaling to ensure model performance.
- **Feature Selection**: The most suitable set of features is selected from the dataset using ML algorithms or statistical methods, for training the ML models to detect intrusions.
- **Development of Machine Learning Models**: ML models such as ANN and hybrid CNNGRU are developed and trained to detect intrusions based on IoT network data. Different model architectures and hyperparameters are used to evaluate the performance of these models.
- **Model Evaluation:** The performance of the models is assessed on a separate dataset to verify their ability to identify intrusions.
- **Integration with Python Flask Application**: After the models are trained and evaluated, it integrated with a user-friendly interface Python Flask web application which allow users to upload network data and receive real-time intrusion detection results.
- **Deployment on Amazon Web Services**: By deploying the application in AWS Elastic Beanstalk (EBS), it ensures scalability and accessibility. User can access the IDS application from anywhere, making it a practical and scalable solution for IoT network security.

### 3.3.1 Machine Learning Algorithms

**i. Artificial Neural Networks**

Fundamental machine learning algorithms, well known for their ability in detecting complex patterns and connections within data and classifying them. ANN consist of an input layer, hidden layers for processing and an output layer for predictions, composed of interconnected nodes or neurons in layers. Weights on connections and activation functions add non-linearity allowing network to learn and model complex patterns (Tpoint, 2021). ANN plays a crucial role at the initial classification level to distinguish between normal and suspicious activities in IoT networks (Figure 4).

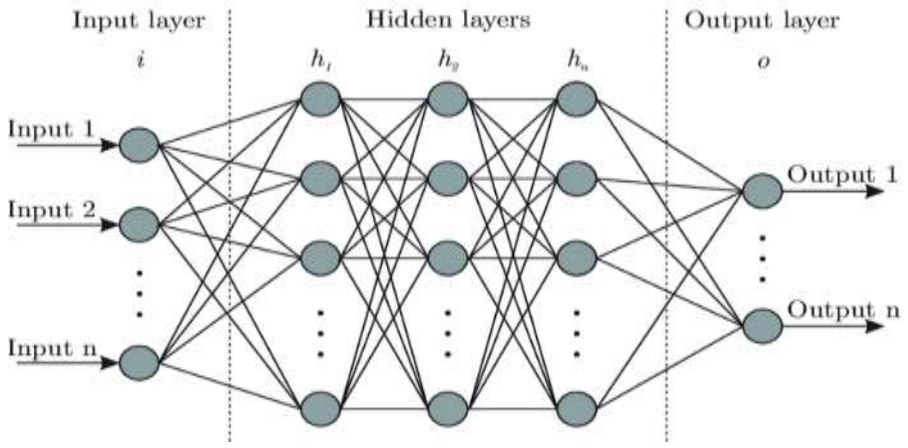


**Figure 4: ANN Architecture**
**(Source: (Tpoint, 2021))**

**ii. Convolutional Neural Networks**

In CNN, feature extraction is done by convolutional layer and pooling layer which detect patterns and reduce dimensionality. Classification is handled by fully connected layers that process these features for final predictions. CNNs analyze network traffic to detect complex patterns and identify anomalies in the IoT network security (Figure 5) (Phung and Rhee, 2019).

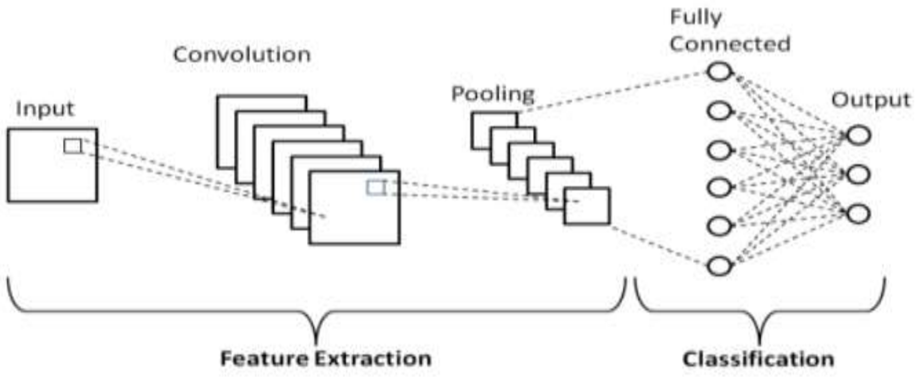


**Figure 5: CNN Architecture**
**(Source: (Phung and Rhee, 2019))**

### iii. Gated Recurrent Units

GRU are a type of recurrent neural network (RNN) that use update gate which controls how much past information is retained and reset gate which determines what to forget. it handles sequential data by addressing the vanishing gradient problem. Hidden state and Candidate hidden state which are combined by an elementwise operator to update the final hidden state. Fully connected layers with activation functions and use operations like copy and concatenate to manage information effectively. GRUs capture temporal dependencies in network traffic, complementing CNNs that focus on spatial patterns and enhancing the detection of attacks (Figure 6) (Dive into Deep Learning, 2023).

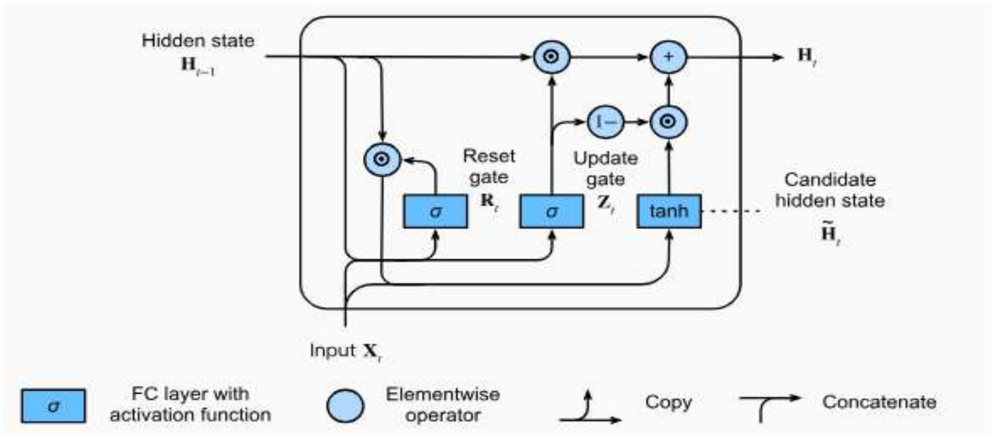

Hidden state $\mathbf{H}_{t-1}$
$\mathbf{H}_t$
Reset gate $\mathbf{R}_t$
Update gate $\mathbf{Z}_t$
Candidate hidden state $\tilde{\mathbf{H}}_t$
σ
σ
tanh
1−
Input $\mathbf{X}_t$
FC layer with activation function
Elementwise operator
Copy
Concatenate

**Figure 6: GRU Architecture**
**(Source: (Dive into Deep Learning, 2023))**

### iv. CNN-GRU Hybrid Approach

The proposed system leverages the strengths of both CNN and GRU by combining the algorithms. CNNs are good at extracting spatial features and GRU capture temporal dependencies. This hybrid method makes the model better at detecting complex attack patterns over time and space. Since IoT networks have diverse data including both spatial and temporal the CNN-GRU hybrid approach is tailored to handle such complexities and ensuring comprehensive intrusion detection across various attack scenarios. IDS effectively detect a wide range of cyber threats in IoT networks using both algorithms.

### v. Comparison Between ANN and CGRU in Intrusion Detection

Pattern recognition and classification tasks are particularly suitable for ANN which offer a robust foundation for different applications. This is enhanced by integrating the spatial feature extraction of CNNs and sequential pattern handling of GRU by combining CNN CGRU. By using a hybrid method, CGRU can provide a comprehensive intrusion detection solution for IoT networks by efficiently identifying complex and evolving attack patterns.

## 4 Design Specification

The architecture and specifications for developing an IDS for IoT networks are explained in the design specification. This system aims to detect and prevent a variety of cyber threats including DoS, exploits, fuzzers and reconnaissance activities. It describes the methods, framework, architecture and necessary requirements. An initial step includes careful data preprocessing such as checking missing values, data quality verification and feature engineering for datasets analysis. For accurate threat detection in IoT networks, the proposed architecture uses ML and deep learning models specifically ANN and hybrid CNN-GRU model (Figure 7).

The structure of the model is divided into multiple stage each of which focus on one particular aspect of data and identify threat. ANN are used to rapidly detect both normal and attack activity which enhances resource efficiency by removing unnecessary traffic and reduces the processing load on subsequent models. A hybrid deep learning model which combines CNN and GRU is used for detailed analysis and classification of various threats. By combining these models, system leverages the strengths to detect a complex of cyberattacks in IoT networks. The proposed IDS for IoT networks is evaluated based on different performance criteria like accuracy, precision, recall, and F1score. These metrics are essential for evaluating the IDS's reliability and efficiency in detecting a variety of cyber threats in IoT environments.

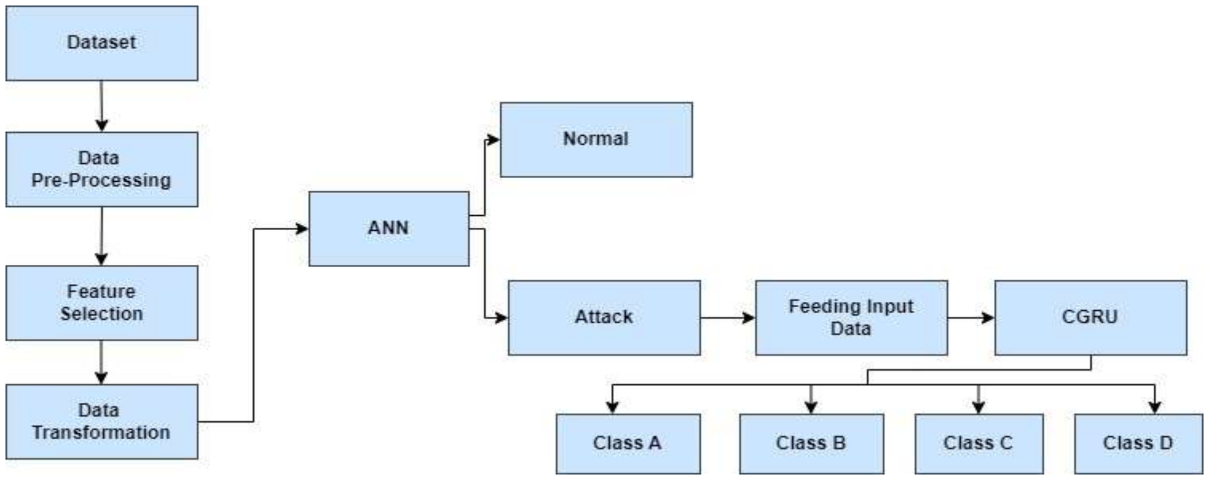


**Figure 7: Two-Level Intrusion Detection Model Workflow (Source: Created by learner)**

## 5 Implementation

### 5.1 Data Preprocessing

Preprocessing the UNSW-NB15 dataset includes dataset loading, handling missing, infinite values, exploring and visualizing data and encoding categorical features

- **Data Loading**: At the initial phase of the preprocessing, the necessary libraries for data manipulation, numerical operations and visualization were imported. The dataset was successfully imported for analysis by uploaded from a specific file location and have error handling in place to handle any issues with file availability.

- **Data Cleaning**: The 'id' column was removed as the initial stage in the data cleaning process. After that, the dataset was reviewed for null values and counts for each feature were collected. The dataset was also examined for infinite values and any features with such values were identified and recorded.
- **Exploratory Data Analysis**:Analysis included a detailed review of the structure of dataset and summary statistics. Categorical features were identified based on their data types. The distribution of attack categories was analyzed and visualized using bar charts and pie charts to understand the frequency and proportion of each category. Similarly, the distribution of service categories was visualized with bar charts to depict the count of each service type.
- **Data Transformation**: Data transformation applied to replace the missing values in the 'service' category with a placeholder value. For additional analysis encoding categorical features into numerical values and this encoding mapped each unique category to a corresponding integer.
- **Analysis of Feature Importance and Selection**: Correlation Coefficient method was most effective to identify relevant features for both binary and multiclass classification tasks. Features that significantly impact classification outcomes were identified and ensures that features having negative or missing correlation values being removed. The relevant features are then visualized using a line chart providing a clear representation of the key contributors to the classification.
- **Binary and Multiclass Oversampling**: For binary classification, the dataset is separated into feature matrices binary_X and target labels binary_y. Similarly, for multiclass classification, the dataset is split into feature matrices multiclass_X and target labels multiclass_y. In the multiclass dataset, for addressing class imbalance the SMOTE technique is applied to generate synthetic samples for underrepresented classes, resulting in a balanced dataset. The resampled data is then reassembled with the updated features and labels

## 5.2 Experimental Setup

- **Model Selection and Data Splitting**: Selected the most suitable ML algorithm like ANN, CNN and GRU based on the problem requirements. To train the model X_train and y_train are created using the dataset and X_test and y_test used to test the model. This split allocates 80% for training and 20% of the data for testing.
- **Model Training**: Initially develops ANN using TensorFlow Keras include architecture, compilation and training using batch normalization, L2 regularization and callbacks. After that, a CGRU neural network includes features Conv1D layers, residual connections, BatchNormalization, MaxPooling and Bidirectional GRUs is developed.
- **Model Evaluation**: The model performance is evaluated using performance metrics such as accuracy, precision, recall and F1-score. Training and validation accuracy and loss are visualized and performance across different classes is presented using a confusion matrix heatmap.
- **Intrusion Detection Application:** Python Flask application in which users can upload CSV files for analysis and it will classify the data into either normal or suspicious of an attack. Through a user-friendly interface, the application provides quick and accurate intrusion detection by using the trained ML models.

## 5.3 Tools Used

### I. Development Environment

- **Google Colab**: Utilizing Google Colab, a cloud-based Jupyter notebook environment that enables interactive coding and collaboration. An ideal platform for data preprocessing processes that provides both the computational resources and a user friendly interface to execute Python code.

- **Google Drive**: Cloud storage service that allows to store, access and share files online. Dataset is managed and accessed through Google Drive

**II. Programming Language**

**Python**

- Python (Version: 3.10.12): The primary programming language used for this project.

**III. Libraries and Packages**

- **Pandas** (Version: 2.1.4): Used for data manipulation and analysis. It included the data structures and functions required to clean and prepare the dataset.
- **NumPy** (Version: 1.26.4): For numerical computations, particularly for handling arrays and matrices.
- **Matplotlib and Seaborn** (Version: 3.7.1 and 0.13.1)**:** Used for data visualization. These libraries helped in generating plots and charts to explore and present the data visually.
- **Warnings**: During code execution, the warnings module was used to suppress unnecessary warnings.
- **Scikit-learn (MinMaxScaler)** (Version: 1.2.3)**:** Provided tools for scaling features to a range, which is an important step of making the data to run ML
- **TensorFlow/Keras** (Version:2.17.0)**:** An open-source library for building and training ML models

**IV. Intrusion Detection Application Development**

- **Python Flask** (Version: 2.0.3): Provides a lightweight framework for creating web applications and integrating with the ML models.

# 6 Evaluation

This section covers a summary of results of our two-level IDS, highlighting some performance metrics. This study shows model performance at different detection stages and the improvements in computational efficiency and time optimization. The results underscore the advantages of our approach in cloud-based IoT environments, highlighting significant enhancements in detection speed and accuracy.

## 6.1 Performance Metrics

- **Accuracy**: Accuracy provides an overall effectiveness measure of the IDS model reflects its capacity to accurately identify both normal and anomalous behaviours. The model accuracy shows how effectively it can differentiate between normal and malicious activities in the case of IoT intrusion (Zuccarelli, 2021).

$$\textbf{Accuracy} = \frac{\boldsymbol{True\ Positives+True\ Negatives}}{\boldsymbol{Total\ Population}}$$

Total Population refers to all instances in the dataset including True Positives, True Negatives, False Positives and False Negatives.

- **Precision:** It measures the proportion of true positive predictions among all the positive predictions by the model. High precision shows a low false positive rate which is crucial in IoT networks in order to reduce unnecessary alerts and ensure that that true activities are not mistakenly classified as threats (Zuccarelli, 2021).

$$\textbf{Precision} = \frac{\textit{True Positives}}{\textit{True Positives}+\textit{False Positives}}$$

- **Recall**: it measures the proportion of actual positive instances that were correctly identified by the model. High recall is essential for ensuring that most malicious activities are detected. In IoT networks, where the consequences of undetected attacks can be severe, high recall helps in identifying and mitigating potential security threats promptly (Zuccarelli, 2021).

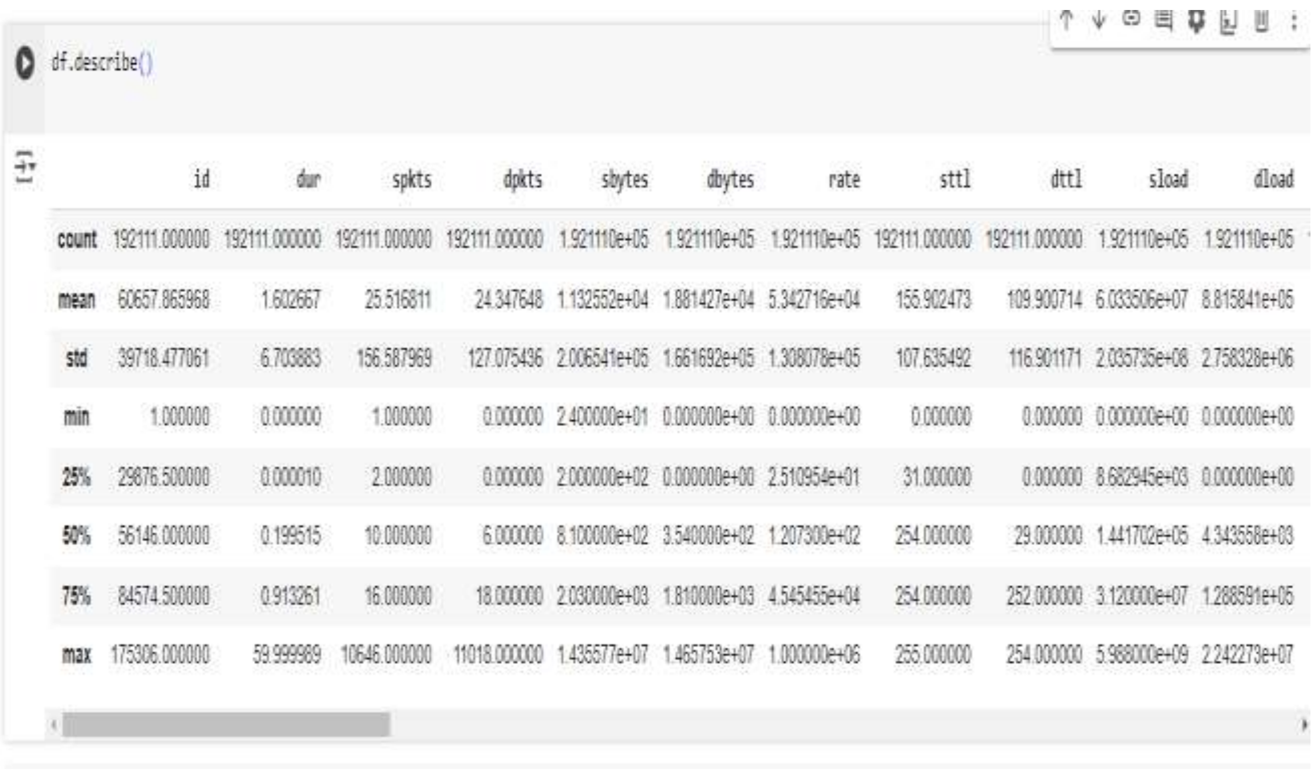
$$\textbf{Recall} = \frac{\textit{True Positives}}{\textit{True Positives}+\textit{False Negatives}}$$

- **F1-Score**: It is particularly utilized when there is an uneven class distribution. High F1-score shows that the model performs well in detecting attacks while also minimizing false positives. This metric is valuable for assessing the overall performance of IDS models in IoT networks, where both accurate detection and minimal false alarms are crucial for maintaining security and operational efficiency (Zuccarelli, 2021).

$$\textbf{F1-Score} = 2 \times \frac{\textit{Precision}\times\textit{Recall}}{\textit{Precision}+\textit{Recall}}$$

## 6.2 Dataset Analysis

The Figure 8 shows statistical summary of the numerical columns of dataset, including key metrics such as count, mean, standard deviation, minimum, median, and maximum.



```
df.describe()
```

| | id | dur | spkts | dpkts | sbytes | dbytes | rate | sttl | dttl | sload | dload |
|---|---|---|---|---|---|---|---|---|---|---|---|
| count | 192111.000000 | 192111.000000 | 192111.000000 | 192111.000000 | 1.921110e+05 | 1.921110e+05 | 1.921110e+05 | 192111.000000 | 192111.000000 | 1.921110e+05 | 1.921110e+05 |
| mean | 60657.865968 | 1.602667 | 25.516811 | 24.347648 | 1.132552e+04 | 1.881427e+04 | 5.342716e+04 | 156.902473 | 109.900714 | 6.033506e+07 | 8.815841e+05 |
| std | 39718.477061 | 6.703883 | 156.587969 | 127.075436 | 2.006541e+05 | 1.661692e+05 | 1.308078e+05 | 107.635492 | 116.901171 | 2.035735e+08 | 2.758328e+06 |
| min | 1.000000 | 0.000000 | 1.000000 | 0.000000 | 2.400000e+01 | 0.000000e+00 | 0.000000e+00 | 0.000000 | 0.000000 | 0.000000e+00 | 0.000000e+00 |
| 25% | 29876.500000 | 0.000010 | 2.000000 | 0.000000 | 2.000000e+02 | 0.000000e+00 | 2.510954e+01 | 31.000000 | 0.000000 | 8.682945e+03 | 0.000000e+00 |
| 50% | 56146.000000 | 0.199515 | 10.000000 | 6.000000 | 8.100000e+02 | 3.540000e+02 | 1.207300e+02 | 254.000000 | 29.000000 | 1.441702e+05 | 4.343558e+03 |
| 75% | 84574.500000 | 0.913261 | 16.000000 | 18.000000 | 2.030000e+03 | 1.810000e+03 | 4.545455e+04 | 254.000000 | 252.000000 | 3.120000e+07 | 1.288591e+05 |
| max | 175306.000000 | 59.999989 | 10646.000000 | 11018.000000 | 1.435577e+07 | 1.465753e+07 | 1.000000e+06 | 255.000000 | 254.000000 | 5.988000e+09 | 2.242273e+07 |

**Figure 8: Dataset Analysis (Source: Generated using Google Colab)**

## 6.3 Data Visualization

The Figure 9 displays the distribution of different attack types and Figure 10 shows the distribution of different service categories in the dataset.

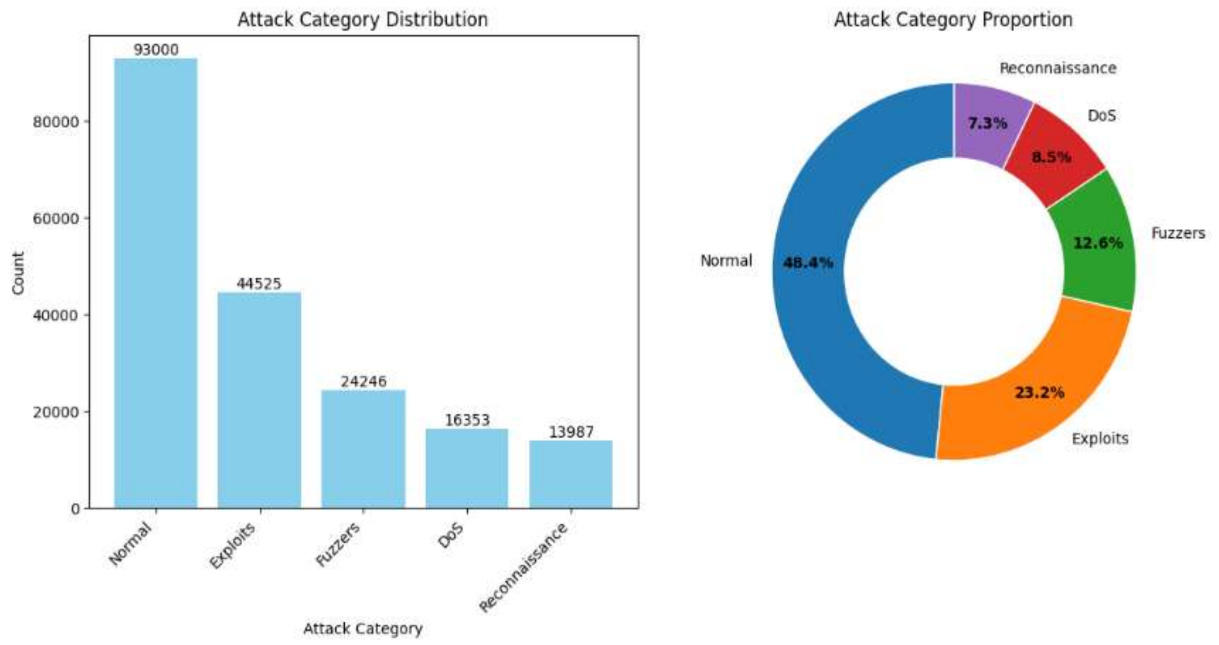


**Figure 9: Attack Category Distribution (Source: Generated using Google Colab)**

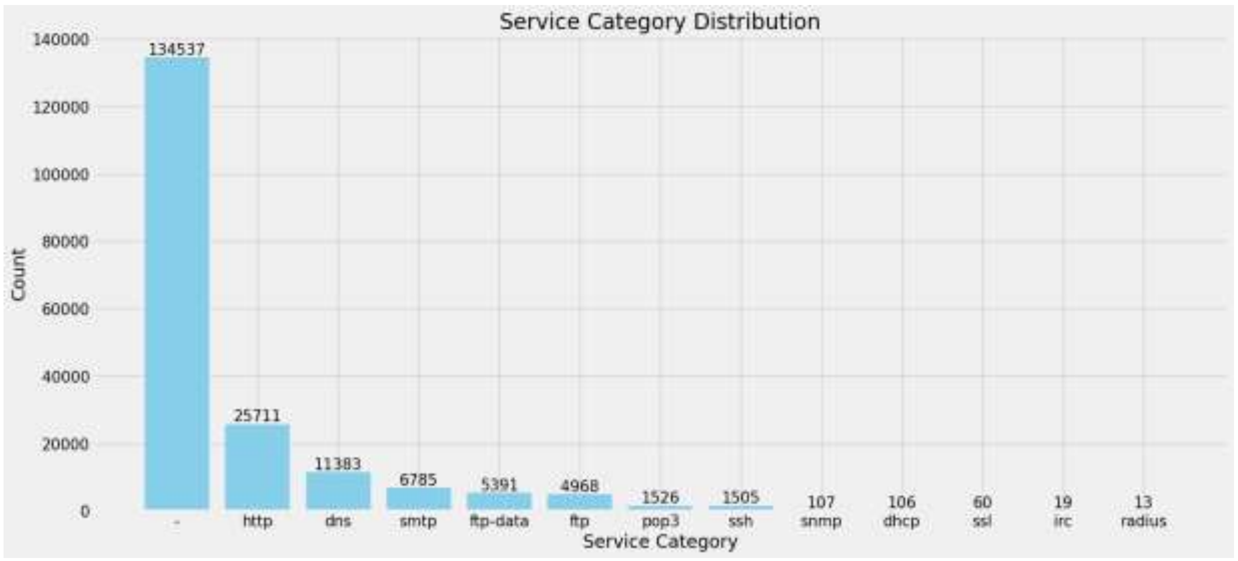


**Figure 10: Service Category Distribution (Source: Generated using Google Colab)**

## 6.4 Binary Classification Results

### 6.4.1 Performance Metrics for Binary Classification

The ANN model achieve validation accuracy is 90.79% in the binary classification. This metric indicates that the model effectiveness in distinguishing between normal and anomalous activities. It suggests that the model has strong performance in accurately classifying data into these two categories (Figure 11).

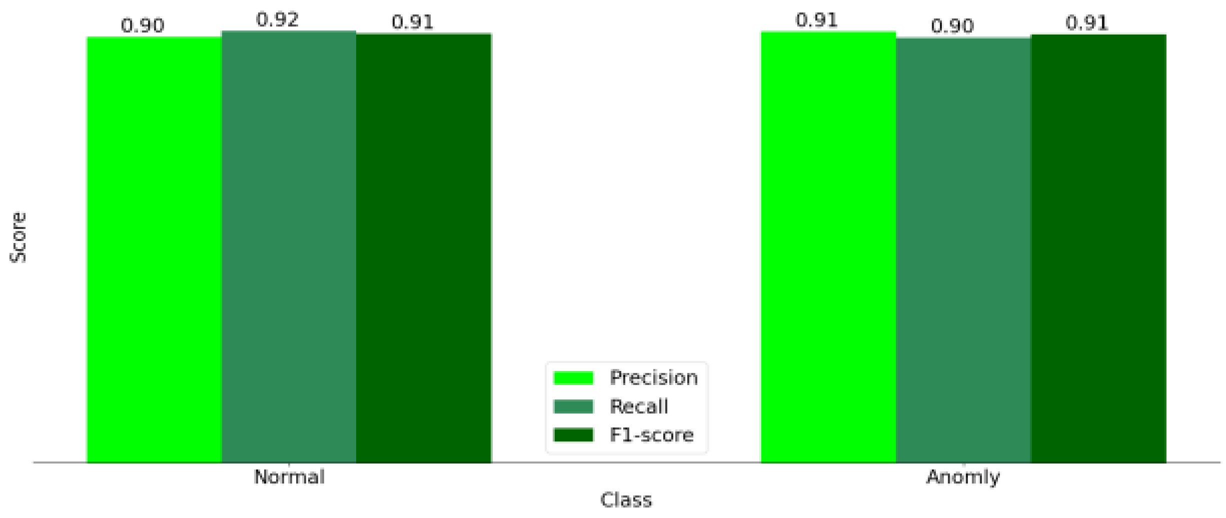


**Figure 11: Performance Metrics for Binary Classification**
**(Source: Generated using Google Colab)**

### 6.4.2 Confusion Matrix for Binary Classification

In binary classification ANN model confusion matrix shows strong performance, with 17,856 True Positives and 18,139 True Negatives. This shows that the model identifies both Normal and Anomaly activities effectively. However, it also reflects some misclassifications with 1,683 False Positives and 1,967 False Negatives. The Figure 12 illustrates the confusion matrix.

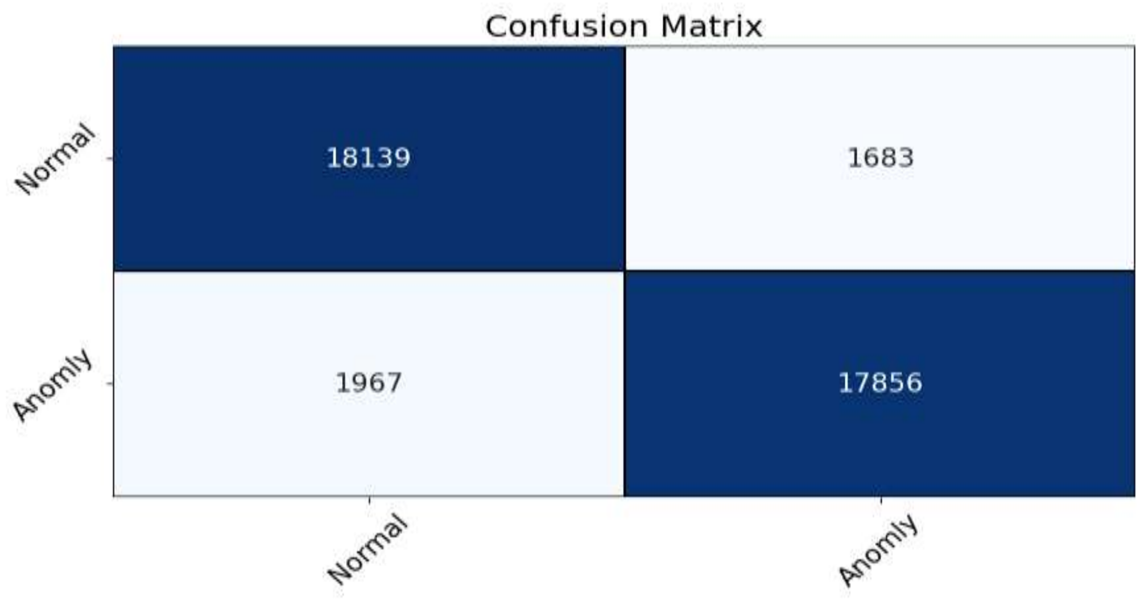


**Figure 12: Confusion Matrix of Binary Classification**
**(Source: Generated using Google Colab)**

## 6.5 Multiclass Classification Results

### 6.5.1 Performance Metrics for Multiclass Classification

The hybrid CGRU model achieved validation accuracy of 74.32% in the multiclass classification, showing the proportion of effectively classified instances out of the total instances. model classifies data into four categories like DoS, Exploits, Reconnaissance, and Fuzzers. The Table 2 illustrates specific performance metrics of each class in detail:

| Class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| DoS | 0.76 | 0.46 | 0.57 | 6679 |
| Exploits | 0.92 | 0.88 | 0.90 | 6679 |
| Reconnaissance | 0.55 | 0.84 | 0.67 | 6678 |
| Fuzzers | 0.86 | 0.81 | 0.83 | 6679 |
| **Accuracy** | | | 0.74 | 26715 |
| **Macro Avg** | 0.77 | 0.74 | 0.74 | 26715 |
| **Weighted Avg** | 0.77 | 0.74 | 0.74 | 26715 |

**Table 2: Performance Metrics of Multiclass Classification (Source: Created by learner)**

An overall view of the model's performance, regardless of class distribution, is provided by the macro average of precision, recall and F1-score for all classes which are 0.77, 0.74 and 0.74, respectively. These values show the consistent performance of model across various attack types. These results correspond with the weighted average considering the support of each class that further validating the robustness of the model. Overall, the CGRU model shows an effective ability to classify different kinds of attacks.

### 6.5.2 Confusion Matrix for Multiclass Classification

The CGRU model confusion matrix in multiclass classification illustrates its effectiveness in identifying different attacks such as DoS, exploits, reconnaissance and Fuzzers. High True Positive counts and relatively low False Positive and False Negative counts in these categories show the model notable strengths in accurately identifying Exploits and Fuzzers. However, there remains some challenges such as the frequent misclassifications between DoS and Reconnaissance. The Figure 13 below illustrates the confusion matrix.

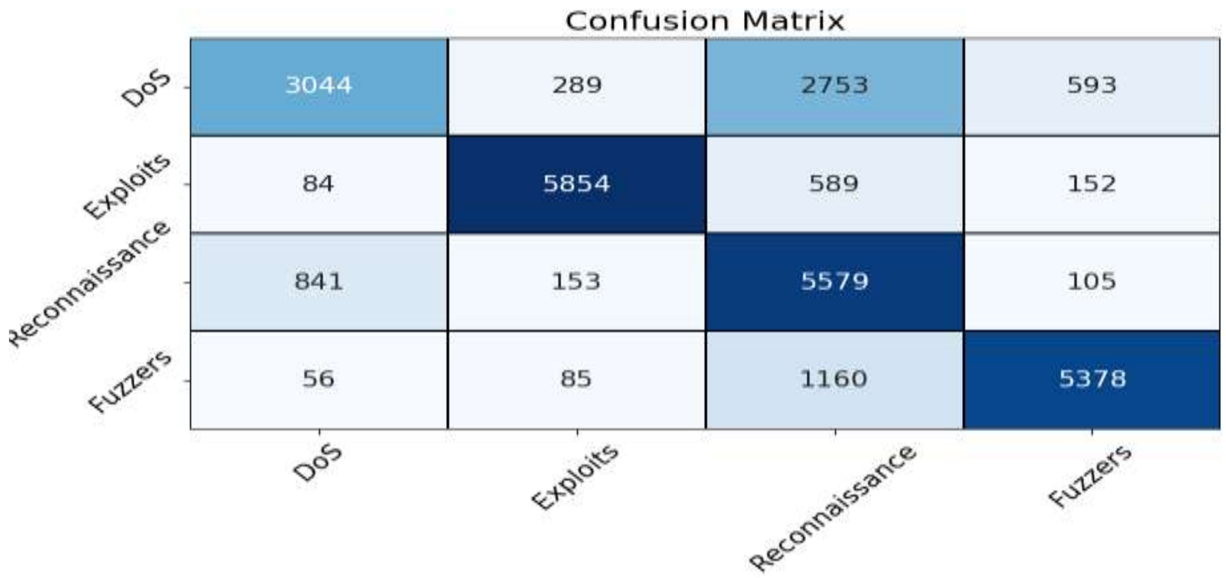


**Figure 13: Confusion Matrix of Multiclass Classification (Source: Generated using Google Colab)**

## 6.6 Feature Importance Visualization

### 6.6.1 Binary Classification

The feature importance visualization highlights the most important features, indicating their crucial role in in binary classification by identifying between normal and anomalous activities. The feature importance for binary classification is shown in the line chart below (Figure 14).

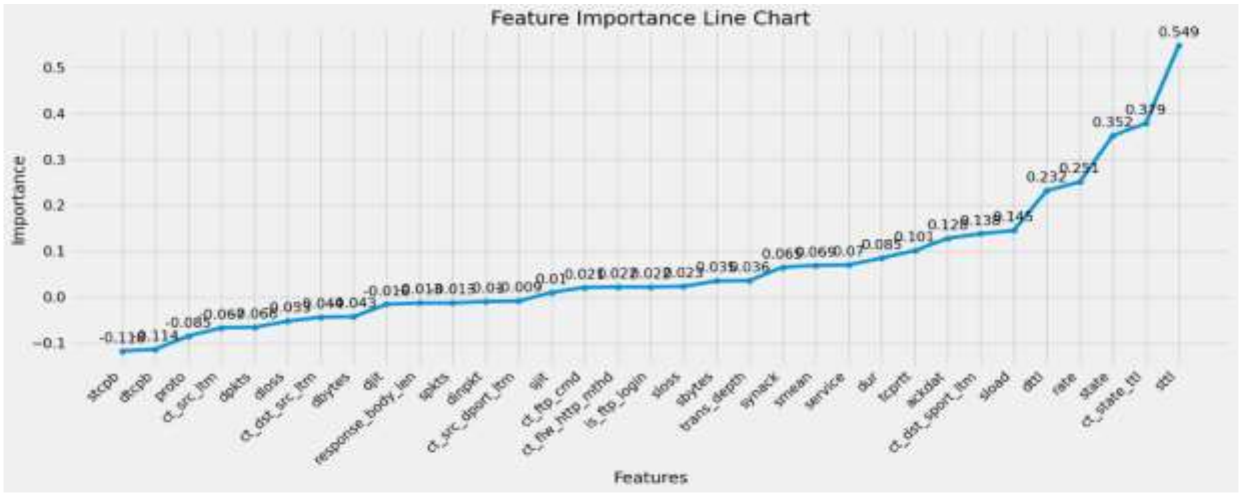


**Figure 14: Feature Importance for Binary Classification**
**(Source: Generated using Google Colab)**

### 6.6.2 Multiclass Classification

Feature Importance visualization for multiclass classification task shows the significance of various features in identifying different attack types. This feature importance is illustrated in the line chart below (Figure 15).

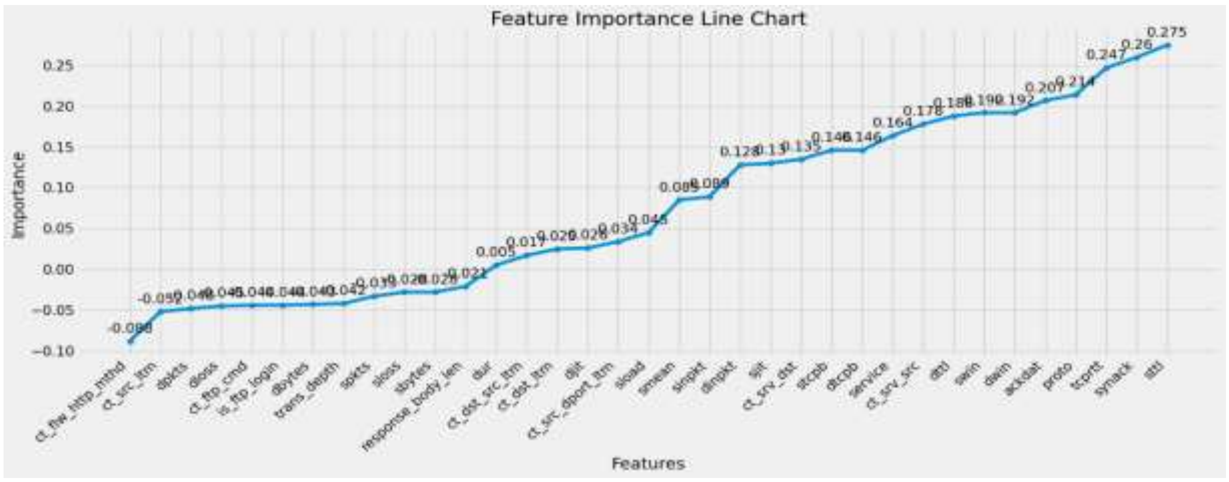


**Figure 15: Feature Importance for Multiclass Classification**
**(Source: Generated using Google Colab)**

## 6.7 Training and Validation Loss & Accuracy Visualization

### 6.7.1 Artificial Neural Networks Model

The model is performing effectively and achieving excellent outcomes overall, indicated by the accuracy graph (Figure 16), which shows steady improvements in training accuracy as well as validation accuracy. The loss graph (Figure 17) indicates occasional overfitting when training loss consistently decreases while validation loss changes. In the end of training, both losses eventually stabilize.

The ANN includes dense layers with units ranging from 64 to 256, utilizing ReLU activation functions and BatchNormalization to enhance learning stability. L2 regularization with a small penalty (l2=0.0001) is applied to control overfitting and balance model complexity. The model is trained over 50 epochs with a batch size of 32, which allows for effective weight updates while managing computational demands. The learning rate starts at

0.001 and is adjusted dynamically using the ReduceLROnPlateau callback which reduces the learning rate if validation accuracy does not improve.

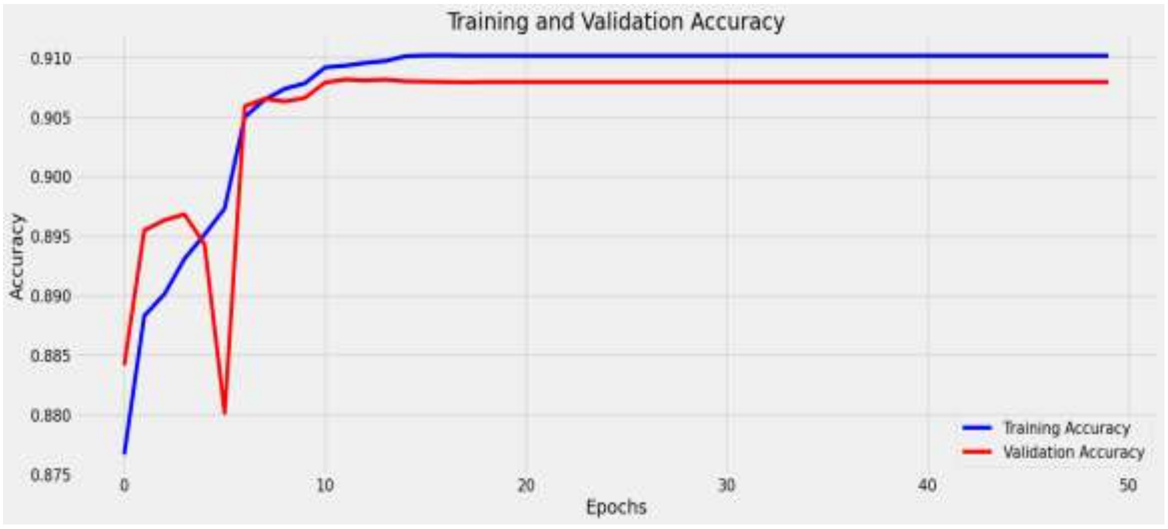


**Figure 16: Training and Validation Accuracy Visualization (Source: Generated using Google Colab)**

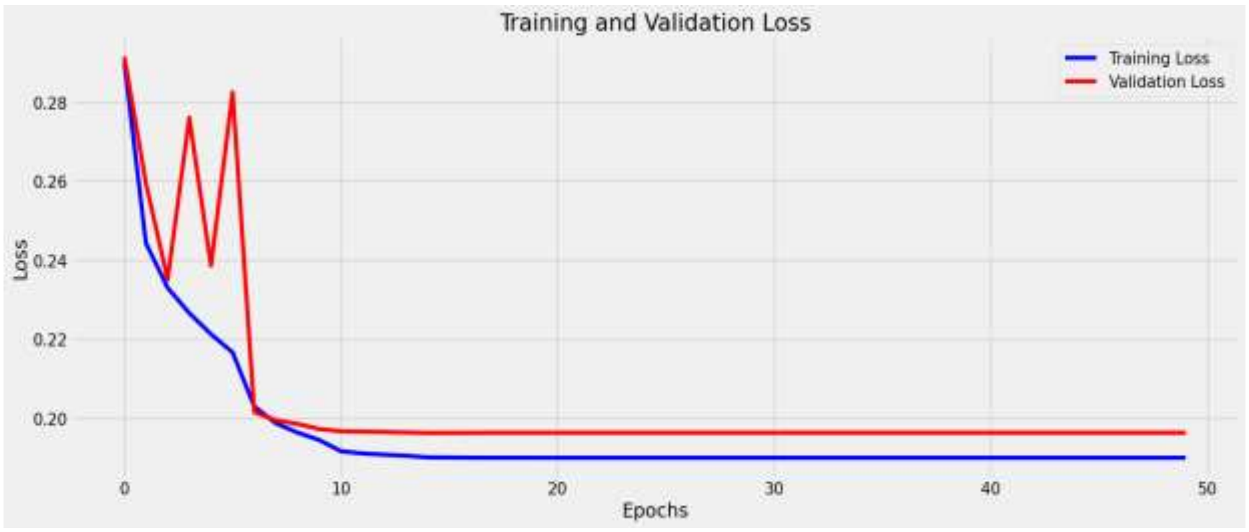


**Figure 17: Training and Validation Loss Visualization (Source: Generated using Google Colab)**

### 6.7.2 CNN-GRU Hybrid Model

The training loss on the loss graph (Figure 18) steadily decreases, while the validation loss varies before stabilizing. This may indicate occasional instability or overfitting. At the end, both the losses remain stable with validation loss shows minimal variations. Both training and validation accuracy indicate consistent increases, according to the accuracy graph (Figure 19). By the end of training, validation accuracy reflects robust performance, whereas training accuracy improves more slowly.

The use of Conv1D layers followed by GRU layers and a Global Average Pooling layer enhances computational efficiency. The model is trained for 20 epochs with a batch size of 32, balancing memory usage and computational load. The learning rate starts at 0.001 and is dynamically adjusted using the ReduceLROnPlateau callback. Dropout (0.4) and BatchNormalization help prevent overfitting and stabilize training, leading to faster and more efficient learning. The Adam optimizer efficiently adjusts the learning rates during training will improves training efficiency

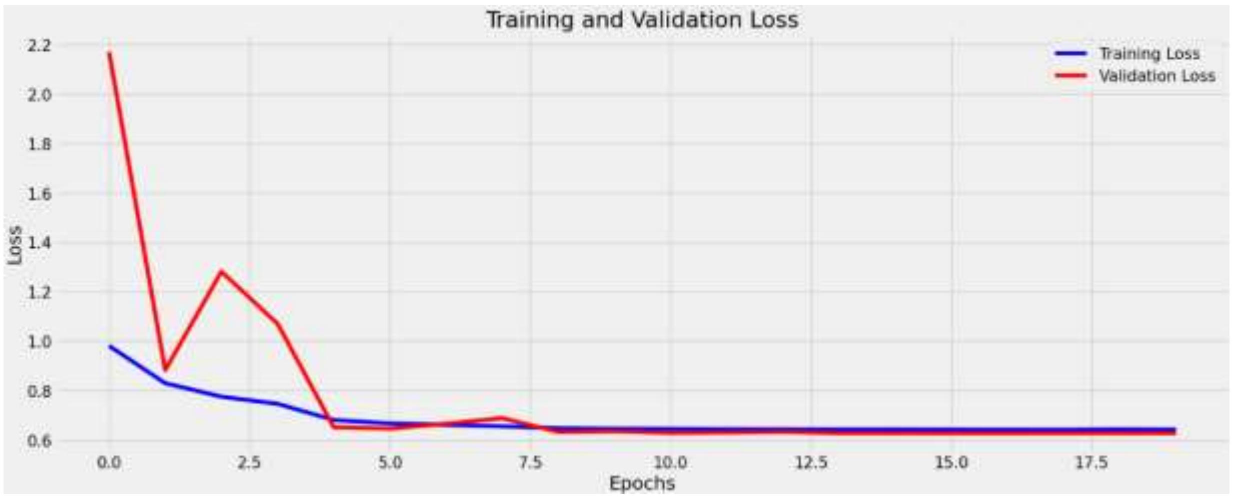


**Figure 18: Training and Validation Loss Visualization**
**(Source: Generated using Google Colab)**

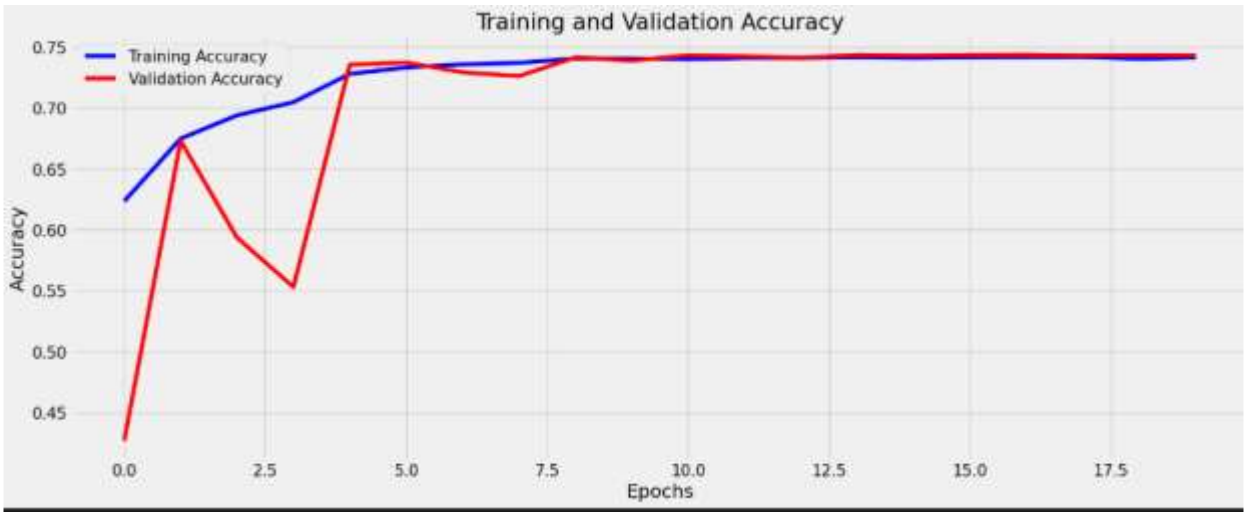
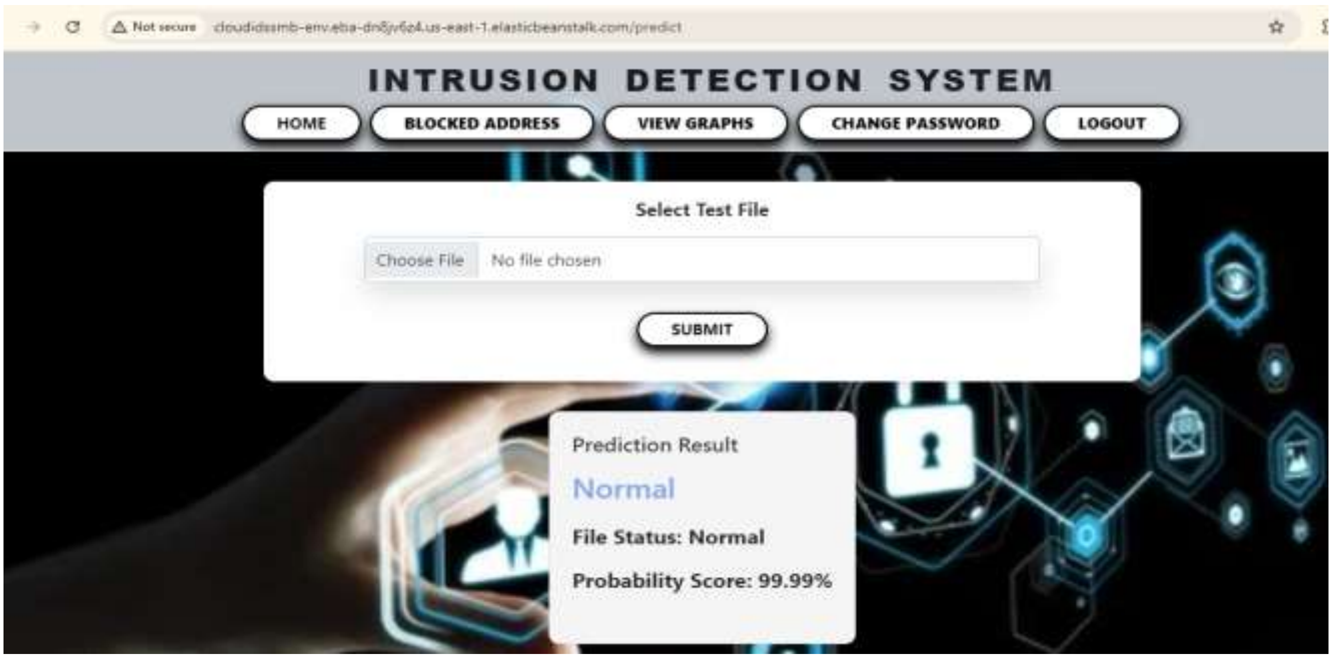


**Figure 19: Training and Validation Accuracy Visualization**
**(Source: Generated using Google Colab)**

## 6.8 AWS Cloud-Enabled Intrusion Detection Application

- **User Interface and Workflow:**
- **User Interface:** A user-friendly Python Flask web interface application deployed in AWS Elastic Beanstalk where the user accesses the application using successful signup and signin, and can upload a CSV file containing network traffic data.
- **Check IP Address:** Initially, the application checks whether the IP address in the uploaded data is already included in the blocked IP list.
  - If the IP address is blocked: It informs the user that the IP is blocked.
  - If the IP address is not blocked: The system proceeds to the next phase.
- **Prediction:** The system checks whether the data is normal or an attack using a ML model integrated with the application in the backend.
- **Result:** On the user interface, the result will be displayed whether it is found to be normal (Figure 20) or an attack (Figure 21). If an attack is detected an alert email will have sent to the user with the CSV details and also IP address is added to the blocked list.

- The URL of the Intrusion Detection Application deployed on AWS Elastic Beanstalk is ***http://cloudidssmb-env.eba-dn8jv6z4.us-east-1.elasticbeanstalk.com/***



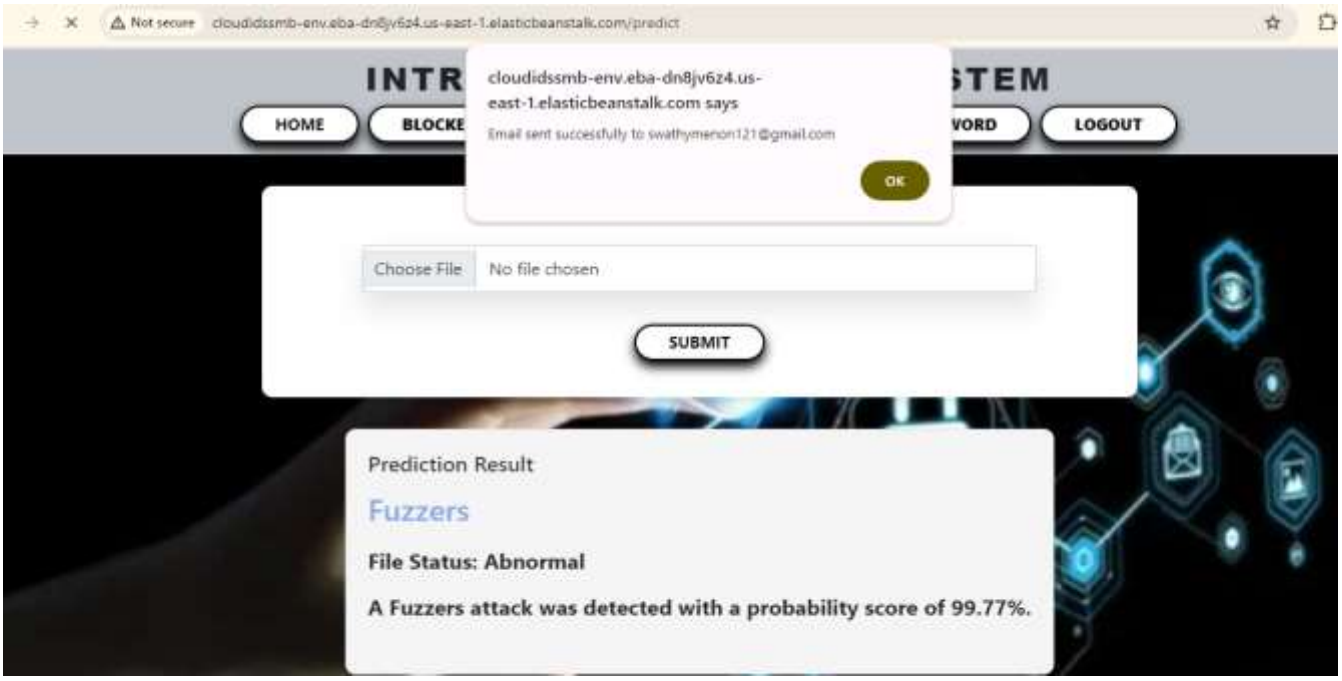
Prediction Result

Normal

File Status: Normal

Probability Score: 99.99%

**Figure 20: Classification Result Display on User Interface – Normal (Source: Application deployed on AWS EBS)**



cloudidssmb-env.eba-dn8jv6z4.us-east-1.elasticbeanstalk.com says

Email sent successfully to swathymenon121@gmail.com

Prediction Result

Fuzzers

File Status: Abnormal

A Fuzzers attack was detected with a probability score of 99.77%.

**Figure 21: Classification Result Display on User Interface – Attack (Source: Application deployed on AWS EBS)**

- **Time Analysis of Traffic Detection in a Flask Application:**

Times based on Flask application and its logs shows corresponding time required for a user to determine if some traffic is normal or an attack, from collecting the packets to obtaining the result.

i. **Login into the System:** Based on the logs, login and redirection to the home page is handled quickly, typically in less than a second.

ii. **Uploading the File**: This step takes approximately 0.01 seconds indicates a very fast file upload process.

iii. **Prediction:**

   a. Normal: This is shown as approximately 0.32 seconds for normal files

   b. Attack: This is shown as approximately 1.91 seconds for attack files

iv. **Send Email**: For attack detection it shows 2.20 seconds to send the email.

**Total duration**

- For normal files (file upload +prediction) ~0.33 seconds
- For attack detection (file upload +prediction +email send) ~ 4.12 seconds

## 6.9 Discussion

This section discusses the development and specific contributions of this research compared to prior studies in the field of IoT IDS. A significant limitation observed in previous studies is the primary focus on binary classification which is identifying whether an intrusion occurs or not without deeply entering into the specific types of attacks (Lin et al. (2021)). While some studies have explored multiclass classification, but it is not common. This research utilizes the inherent structure of the dataset that support two-level classification approach. The proposed system basically performs feature selection for binary classification to distinguish between normal and attack activity using ANN. It then uses feature selection for multiclass classification to identify specific attack types by applying hybrid CNN-GRU model (Torabi et al. (2021)).

When comparing this research with Maghrabi (2024), it presents one level classification approach in which entire dataset processes in one go, that results slower processing times and difficulties in real-time scenarios. In contrast, this research implements a two-level classification system that uses ANN for initial filtering. Due to its binary classification structure the ANN efficiently processes data with minimal computational resources. This is followed by a CNN-GRU hybrid model for detailed attack classification which optimizes resource allocation and reduces computational load. This method results processing time 0.33 seconds for normal traffic and 1.92 seconds for attacks significantly optimizing speed and efficiency. Furthermore, while Maghrabi (2024) uses traditional ML methods like Random Forest which is computationally demanding and less effective for complex attacks, this research makes use of an advanced CNN-GRU model for in-depth analysis and a lightweight ANN for initial classification. This combination improves the system ability for large scale IoT systems while also enhancing computational efficiency.

This research provides scalability, faster deployment and real-time processing by developing a python flask based IDS application deployed on AWS Elastic Beanstalk. On the other hand, Maghrabi's approach were not use cloud-based deployment but this research enhances detection accuracy and efficiency, resolves issues with scalability and real-time response and offers a more resilient and reliable IoT network solution

# 7 Conclusion and Future Work

In conclusion, a two-level classification approach is developed by this research using ANN and a hybrid deep learning model to enhance computational efficiency and time optimization in Cloud IoT IDS. In the initial classification the proposed system utilizes ANN to classify between normal and attack activities and for detailed threat analysis a hybrid model by combining CNN and GRU were implemented. This approach significantly enhances IoT network security by optimizing resource usage, providing accurate threat detection through increased computational efficiency and reduced processing times. The python flask based IDS web application that deployed on AWS Elastic Beanstalk further enhance the system scalability and ensure the system can handle large volumes of data with minimal latency in real world scenarios.

However, further study is needed in several areas such as in order ensure the robustness of IDS across various IoT scenarios it should evaluate using diverse datasets. Additionally, it is essential to update and enhance detection methods regularly to adapt new attack types. Integrating this IDS into a three-tier cloud architecture will further improves the scalability, efficiency and overall security in IoT environments. Threat detection will be improved and ensure rapid responses to potential threats through this integration. We can improve IoT security and offer adaptable solutions against evolving landscape of cyber threats by addressing these areas.

## 8 Video Presentation Demo

- **Video Presentation Link: *https://youtu.be/s97o4kOf2x8***

## References

Al-Hamadi, H., Chen, R., Wang, D.C. and Almashan, M., 2020. Attack and defense strategies for intrusion detection in autonomous distributed IoT systems. *IEEE Access*, *8*, pp.168994-169009.

Ali, S.Y., Farooq, U., Anum, L., Mian, N.A., Asim, M. and Alyas, T., 2024. Securing cloud environments: A Convolutional Neural Network (CNN) approach to intrusion detection system. *Journal of Computing & Biomedical Informatics*, *6*(02), pp.295-308.

Arthi, R., Das, U., AK, D.R. and Balachandar, N., 2024, March. Cloud-based Intrusion Detection System using Various Machine Learning Techniques. In *2024 5th International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)* (pp. 413-420). IEEE.

Bakhsh, S.A., Khan, M.A., Ahmed, F., Alshehri, M.S., Ali, H. and Ahmad, J., 2023. Enhancing IoT network security through deep learning-powered Intrusion Detection System. *Internet of Things*, 24, p.100936.

Dive into Deep Learning (2023) 'Gated recurrent units (GRU)', *Dive into Deep Learning 1.0.3 Documentation*. Available at: https://d2l.ai/chapter_recurrent-modern/gru.html#fig-gru-1 [Accessed: 30 June 2024].

Fatani, A., Abd Elaziz, M., Dahou, A., Al-Qaness, M.A. and Lu, S., 2021. IoT intrusion detection system using deep learning and enhanced transient search optimization. *IEEE Access*, *9*, pp.123448-123464.

Kandhro, I.A., Alanazi, S.M., Ali, F., Kehar, A., Fatima, K., Uddin, M. and Karuppayah, S., 2023. Detection of real-time malicious intrusions and attacks in IoT empowered cybersecurity infrastructures. *IEEE Access*, *11*, pp.9136-9148.

Laassar, I., Hadi, M.Y., Arifullah, H.R. and Khan, F.S., 2024. Proposed algorithm base optimisation plan for feature selection-based intrusion detection in cloud computing. *Indonesian Journal of Electrical Engineering and Computer Science*, 33(2), pp.1140-1149.

Larriva-Novo, X., Villagrá, V.A., Vega-Barbas, M., Rivera, D. and Sanz Rodrigo, M., 2021. An IoT-focused intrusion detection system approach based on preprocessing characterization for cybersecurity datasets. *Sensors*, *21*(2), p.656.

Lin, X.X., Lin, P. and Yeh, E.H., 2020. Anomaly detection/prediction for the internet of things: State of the art and the future. *IEEE Network, 35*(1), pp.212-218.

Long, Z., Yan, H., Shen, G., Zhang, X., He, H. and Cheng, L., 2024. A Transformer-based network intrusion detection approach for cloud security. *Journal of Cloud Computing*, 13(1), p.5.

Maghrabi, L.A., 2024. Automated Network Intrusion Detection for Internet of Things Security Enhancements. *IEEE Access.*

Nallakaruppan, M.K., Somayaji, S.R.K., Fuladi, S., Benedetto, F., Ulaganathan, S.K. and Yenduri, G., 2024. Enhancing Security of Host-based Intrusion Detection Systems for the Internet of Things. *IEEE Access.*

Phung, V.H. and Rhee, E.J., 2019. A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets. *Applied Sciences*, 9(21), p.4500.

Rani, D., Gill, N.S., Gulia, P., Arena, F. and Pau, G., 2023. Design of an intrusion detection model for IoT-enabled smart home. *IEEE Access, 11*, pp.52509-52526.

Roy, S., Li, J., Choi, B.J. and Bai, Y., 2022. A lightweight supervised intrusion detection mechanism for IoT networks. *Future Generation Computer Systems, 127,* pp.276-285.

Talaviya, A. (2023) *CRISP-DM Framework: A Foundational Data Mining Process Model*. *Medium*. Available at: https://medium.com/@avikumart_/crisp-dm-framework-a-foundational-data-mining-process-model-86fe642da18c [Accessed: 24 August 2024].

Thamilarasu, G., Odesile, A. and Hoang, A., 2020. An intrusion detection system for internet of medical things. *IEEE Access, 8*, pp.181560-181576.

Tpoint (2021) 'Artificial neural network tutorial', *Javatpoint.* Available at: https://www.javatpoint.com/artificial-neural-network [Accessed: 30 June 2024].

Torabi, M., Udzir, N.I., Abdullah, M.T. and Yaakob, R., 2021. A review on feature selection and ensemble techniques for intrusion detection system. *International Journal of Advanced Computer Science and Applications, 12*(5).

Vibhute, A.D. and Nakum, V., 2024. Deep learning-based network anomaly detection and classification in an imbalanced cloud environment. *Procedia Computer Science*, *232*, pp.1636-1645.

Zhou, X., Liang, W., Li, W., Yan, K., Shimizu, S., Kevin, I. and Wang, K., 2021. Hierarchical adversarial attacks against graph-neural-network-based IoT network intrusion detection system. *IEEE Internet of Things Journal, 9*(12), pp.9310-9319

Zuccarelli, E. ‘Jay’. (2021) Performance metrics in machine learning — Part 1: Classification. *Medium*. Available at: https://towardsdatascience.com/performance-metrics-in-machine-learning-part-1-classification-6c6b8d8a8c92 (Accessed: 10 August 2024).