A Study On Machine Learning-Based Network Intrusion Detection System

Final Progress Report

Artemiy Vishnyakov

CS4490Z THESIS
Department of Computer Science
Western University
July 8, 2024

Thesis supervisor:
Associate Prof. Zubair Fadlullah
Dept. of Computer Science, Western University

Course instructor:
Prof. Nazim H. Madhavji
Dept. of Computer Science, Western University

Glossary:

AI: Artificial Intelligence

ANN: Artificial Neural Network

CNN: Convoluted Neural Network

DT: Decision Tree

IoT: Internet of Things

KNN / KNNC: K-Nearest Neighbor Classifier

LR: Linear Regression

ML: Machine Learning

RF: Random Forest

RNN: Recurrent Neural Network

SGD / SGDC: Stochastic Gradient Descent Classifier

SVC / SVM: Support Vector Machine Classifier

VPN: Virtual Private Network

Abstract

Context and motivation:
Cybersecurity is an ongoing concern, particularly with networked systems connected to the Internet. Current practices to detect network intrusions and other unwanted attacks are mainly rule-based or utilise machine learning. However, attacks against encrypted protocols and zero day threats need to be detected in next-generation network systems, and these types of threats are harder to detect with current rule- or machine-based models.
Research question:
The key research question is how to adapt existing Machine Learning frameworks to detect cyber attacks and investigate how such models can be generalized based on network protocol behavior to reduce reliance on packet payload and have better performance against zero-day attacks which are typically overlooked, as well as check the applicability of such machine learning models on encrypted VPN traffic.
Principal ideas:
Key ideas to be addressed in this research include the exploration of the utilization of advanced Machine Learning techniques for classification of data in order to build a reliable and efficient system for detecting and classifying various types of network attacks. Additionally, considerations will be given to study and account for changes in communication protocols as well as reducing the reliance on packet payload to further generalize the models.
Research methodology:
The research methodology for this study consists of a combination of literature review, theoretical analysis, data collection and pre-processing, model development, evaluation and optimization, and documentation.
Result analysis: The result of this research demonstrates an optimistic utilization of Machine Learning models to detect and classify modern network traffic and intrusions. Moreover, this research demonstrates the possibility for the integration and application of machine learning models into a real-world network intrusion detection system.
Novelty analysis:
This research is not only a novel approach to the detection and classification of network intrusions, but also removes the reliance on packet payload and addresses zero-day attacks which are often overlooked. Thus contributing to the advancement of cybersecurity research.

Contents

1 Introduction **1**
- 1.1 General background and problem description . . . 1
- 1.2 Focus of the Thesis . . . 1
- 1.3 Key Research Questions Addressed . . . 1
- 1.4 Key Results . . . 1
- 1.5 Novelty and Significance . . . 2
- 1.6 Report structure . . . 2

2 Background and Related Work **2**
- 2.1 What is a Network Intrusion Detection System . . . 3
- 2.2 Network Attacks . . . 3
- 2.3 Types of Machine Learning Models . . . 3
  - 2.3.1 Supervised Learning . . . 3
  - 2.3.2 Unsupervised Learning . . . 4
- 2.4 Existing Network Intrusion Detection Systems . . . 4
  - 2.4.1 Android Malware Detection Using Deep Learning . . . 4
  - 2.4.2 Machine Learning Based Network Attacks Classification . . . 5
  - 2.4.3 Detection of Network Attacks using Machine Learning and Deep Learning Models . . . 5
  - 2.4.4 Cyber Attack Detection in Network Traffic Using Machine Learning . . . 5
  - 2.4.5 Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study . . . 6
  - 2.4.6 An unsupervised deep learning ensemble model for anomaly detection in static attributed social networks . . . 6
- 2.5 Research Gap . . . 6

3 Research Objectives **6**

4 Methodology **7**
- 4.1 Data Analytics . . . 7
- 4.2 Dataset Labeling . . . 10
- 4.3 Feature Selection . . . 12
- 4.4 Per Packet Training . . . 12
  - 4.4.1 Objective . . . 12
  - 4.4.2 Selected Features . . . 13
  - 4.4.3 Rationale . . . 13
- 4.5 Per Flow Training . . . 13

4.5.1 Objective . . . 13
4.5.2 Selected Features . . . 13
4.5.3 Rationale . . . 13
4.6 Dataset Train Test split . . . 13
4.7 Model Selection . . . 14
4.8 Environmental Factor Specifications . . . 16

5 Results **16**
5.1 Performance Metrics . . . 16
5.2 Comparison between models . . . 17
5.2.1 Context to results . . . 19
5.2.2 Analysis of results . . . 19
5.2.3 ROC and AUC . . . 19
5.2.4 Trade-off between True Positives and False Positives . . . 20
5.2.5 Prediction Timings . . . 21
5.2.6 Decision Tree and Random Forest Feature Importance . . . 21
5.3 Evaluation of results . . . 22
5.4 Effectiveness on VPN Traffic . . . 23
5.4.1 VPN Data . . . 23
5.4.2 Encrypted Traffic Dataset . . . 23
5.4.3 Results . . . 24
5.4.4 Discussion of Results for VPN Traffic . . . 24

6 Discussion **25**
6.1 Threats to the validity of results . . . 25
6.1.1 Low Sample Size . . . 25
6.1.2 Impact of Dataset Imbalance . . . 26
6.1.3 Patterns within Dataset . . . 26
6.2 Mitigation . . . 26
6.2.1 Low Sample Size . . . 26
6.2.2 Impact of Dataset Imbalance . . . 26
6.2.3 Patterns within Dataset . . . 26
6.3 Implications of results . . . 26
6.4 Limitations of results . . . 27
6.4.1 Conditions of Limitations . . . 27
6.4.2 Conditions counter measurement . . . 27
6.5 Generalizability of results . . . 28

7 Conclusions **28**

8 Future Works and Lessons Learned **29**
8.1 Future Works 29
8.2 Lessons Learnt 30

9 Acknowledgments **30**

List of Figures

1 Network Security Taxonomy [1] 4
2 Data Description for CIC IoT 2023 Dataset [2] 8
3 Dataset Methodology Topology from [2] 9
4 Dataset Counts [2] 10
5 Dataset Distribution 11
6 Performance Metrics 18
7 ROC of the models 20
8 Average Prediction Times 21
9 Feature Importance 22
10 Model Metrics for VPN Traffic 24

List of Tables

1 Metrics for PerFlow models 17
2 Metrics for PerPacket models 18
3 PerPacket Metrics for VPN Traffic 24
4 PerFlow Metrics for VPN Traffic 25

1 Introduction

1.1 General background and problem description

In the modern digital age, network security is a critical concern for organizations and individuals alike. The proliferation of cyber-attacks necessitates the development of robust methods for detecting and mitigating these threats [3]. Machine learning models have shown great promise in this field due to their ability to learn and adapt to complex patterns in data [4]. This thesis investigates the application of various machine learning models for the detection and classification of network attacks, presenting an analysis based on empirical results.

1.2 Focus of the Thesis

This research aims to investigate machine learning techniques and algorithms for the purpose of detection and classification of network intrusions. Since many current detection systems rely on the packet payload to identify whether it is an intrusion, this research will focus on the improvement of detection and classification of traffic the remove the reliance on payload content for detection.

1.3 Key Research Questions Addressed

How to utilize existing Machine Learning algorithms and frameworks to detect and classify network activity and handle the model drift problem to combat emerging, zero-day attacks in next generation networks.

1.4 Key Results

Models were trained on 2 types of data, features regarding traffic flows, and features regarding individual packets. For the flow-based training, the decision tree model achieved the highest performance with an average accuracy of 99.38%, precision of 99.39%, recall of 99.38%, and F1 score of 99.38%. This model's superior performance shows its robustness in handling the broader contextual information provided by per flow features. Close contenders were the random forest and RNN with accuracies of 90.19%. As for the packet-based training, the best model was the random forest with an average accuracy of 86.72%, precision of 86.02%, recall of 86.72%, and F1 score of 86.19%.

1.5 Novelty and Significance

The novelty of this research lies in addressing present gaps within the current cybersecurity landscape. The investigation focuses on machine learning-based systems capable of classifying network traffic into more specific classes, which enhances the granularity and accuracy of attack detection. Notably, this study emphasizes handling of network traffic exclusively using header information, reducing the reliance on payload content for classification, which is often overlooked in traditional methods. Additionally, it ensures that the models are up to date with modern attack techniques, thereby improving the detection of newer iterations of attacks, including zero-day exploits. By integrating these advanced features, this research not only offers a novel approach to intrusion detection but also significantly contributes to the advancement of cybersecurity research.

1.6 Report structure

1. Introduction
2. Background and Related Work
3. Research Objectives
4. Methodology
5. Results
6. Discussion
7. Conclusions
8. Future Work and Lessons Learnt
9. Acknowledgments
10. References

2 Background and Related Work

This section briefly summarizes some relevant research done in network intrusion systems using machine learning, as well as the recent developments in both. Furthermore, this section analyzes some of the current research gaps related to the focus of the thesis.

2.1 What is a Network Intrusion Detection System

A Network Intrusion Detection System is a cybersecurity solution designed to monitor and analyze incoming network traffic to detect patterns which may indicate malicious activity, intrusions, or cyber attacks. A network intrusion detection system can be set up to block the suspicious traffic, or only alert a network administrator. By proactively scanning the network traffic, the network intrusion detection system allows for neutralization of threats before they appear as well as rapid response to sudden threats. There are 2 main methods for detection, signature and heuristic based, which differ on how they detect. Signature based detection uses pattern matching to detect attacks, whereas heuristic based systems find deviation from the normal [5].

The proposed approach employed in this research best aligns with what is defined as a Network Security Monitor by Molina-Coronad et. al., an NSM analyzes traffic on a broadcast LAN to detect intrusive behavior, by using network standards, the NSM can monitor a heterogeneous set of hosts and operating systems simultaneously [6]. Making it more accurate to call the intention of this research a signature-based network security monitor.

2.2 Network Attacks

There are many different types of network attacks, each with their own intricacies and methodologies. The list is too long and varied to explain each type individually, but the general taxonomy is summarized in Figure 1.

2.3 Types of Machine Learning Models

According to El Naqa et. al., machine learning is an evolving branch of computational algorithms that are designed to emulate human intelligence by learning from the surrounding environment [7]. They are considered the work horse in the new era of big data. Models are categorized by their learning, with the two main types being supervised and unsupervised learning.

2.3.1 Supervised Learning

Supervised learning is based on the training data being similar to the problem it must solve, with the input data being provided alongside the expected output. While the model is being trained on the training data, it tries to find patterns to find the mapping function f from input x to output y such that $f(x)$ returns y and will be able to predict on unknown data points using said function and patterns.

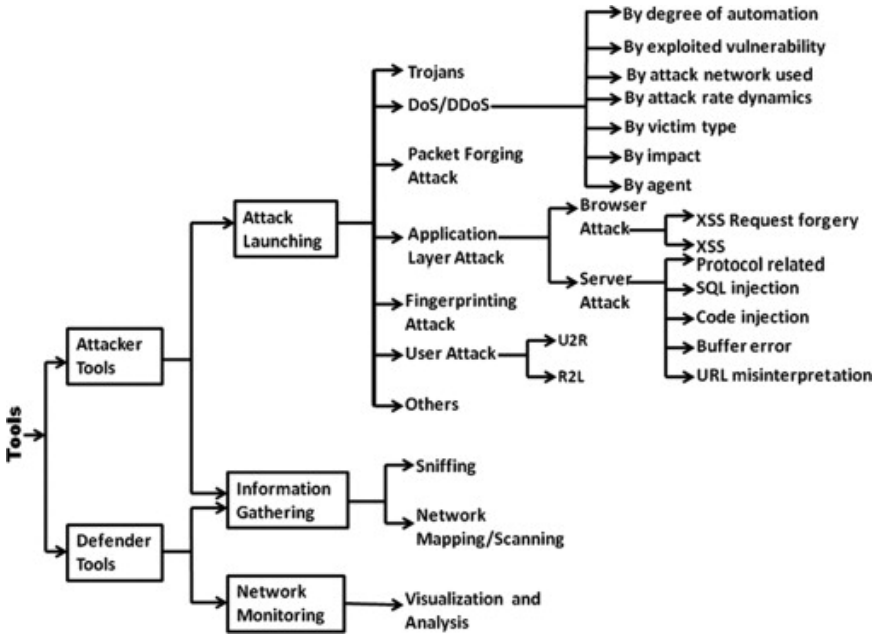

Figure 1: Network Security Taxonomy [1]

2.3.2 Unsupervised Learning

Unsupervised learning differs in the data it is trained on, which is unlabelled. The model learns from input data without expected set output values, and the training dataset does not provide definite answers to the given problem. Instead of categorization or prediction of output values, unsupervised learning algorithms focus on grouping the data based on patterns and shared characteristics, without a set single correct answer.

Given that for each instance of network activity there is a distinct and absolute classification, for the purpose of classification, supervised learning is the correct choice for learning strategy.

The types of machine learning models used in this project as well as their details will be discussed in the Methodology section.

2.4 Existing Network Intrusion Detection Systems

2.4.1 Android Malware Detection Using Deep Learning

Omar et. al. proposed a machine learning based malware detection system for android systems [8]. They utilized SVM, KNN, DT, RF, Naïve Bayse (NB) and GRU RNN, with the resulting f1 scores being in the range [93.9%, 98.0%]. The best model was

the RNN at 98.0% f1 score, followed closely by RF at 97.9%, KNN at 97.1%, DT at 96.6%, SVM at 96.1%, and NB at 93.9%. The features selected to be trained on were related to permissions and API calls from APK files.

2.4.2 Machine Learning Based Network Attacks Classification

Yuxuan Che proposed a machine learning based network intrusion detection system with utilized Naïve Bayes and SVM classifier models [9]. The results gave widely varying scores for model metrics, but the best performers based on f1 score were the SVM trained on the DARPA1999 dataset and the Naïve Bayes model trained on the KDDCup 1999 dataset. But results for many of the models were missing scores and metrics, so comparisons were limited. Though it was not stated, the dataset selection implies the NB models were trained on packet data, whereas the SVM models were trained on data about flows of traffic rather than individual packets, which makes the models unsuitable for live monitoring and detection.

2.4.3 Detection of Network Attacks using Machine Learning and Deep Learning Models

Dhanya K. A et. al. investigated the usage of machine learning models for the purpose of usage within a network intrusion detection system [10], the unique ideas and results to take note of are as follows, they opted to not utilize any feature selection and did not disclose how they prepared the data for training, making it impossible to deduce if the models were trained to predict for packets or whole flows. The best model was a DT with an accuracy of 99.05% and F1 score of 99%, the RF, AdaBoost, XGBoost, and MLP all showed similar results, and the SVM and KNN showed scores around 94%. It was seen that the chosen K value for KNN did not have a significant impact on results.

2.4.4 Cyber Attack Detection in Network Traffic Using Machine Learning

Khalid Almulla investigated various machine learning models for the purpose of detection and classification of network traffic [11]. The notable results include a C5 Decision Tree model with 99.21% accuracy, an ANN with 98.57% accuracy, and a Multilayer Perceptron with a 98.90% accuracy. The results are quite good and indicate that it is worth investigating the process behind them. The dataset used was the KDD Cup 1999 which contains individual connection data, and the CICIDS2017 and 2006+ Kyoto datasets which contains flow information. Feature selection used was {protocol_type, flag, src_bytes, dst_bytes, count, srv_count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate}.

2.4.5 Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study

According to Mohamed A. et. al. any model that can be trained on network data has been trained to at minimum decent success, but a vital takeaway from their research is the evident focus on old datasets [12]. The majority of the studies investigated used one of the old datasets such as DARPA or one of the KDD datasets, which are 25 years outdated. Additionally, it is seen that there is a lack of datasets with encrypted traffic, which exposes a research gap.

2.4.6 An unsupervised deep learning ensemble model for anomaly detection in static attributed social networks

Wasim Khan investigated the usage of unsupervised learning neural networks for the detection of anomalies within network traffic [13]. The methodology and decision to use unsupervised learning contradicts most other research into network intrusion detection, and seemingly for a good reason, as of the created models, the best one was an ensemble with a precision of 83.25%, significantly lower than other approaches.

2.5 Research Gap

It is seen that for many of the models suffer from at least one of the following problems, many have used the KDD Cup 1999 dataset, but it is quite outdated, and with the rapid evolution of network attacks it is illogical to trust that the results of the models trained on the KDD Cup 1999 dataset will be as accurate with modern traffic. Another flaw with some of the research done is the reliance on packet payload for detection and classification, though payload-based detection is still valuable, it is easily overcome through the usage of VPNs or encrypted traffic. Finally, most of the models classify traffic as either an attack or benign, though this is the minimum needed for a network intrusion detection system, it is more valuable to classify traffic into more specific classes and change behaviour and response depending on which type of attack is detected. Therefore, the research gaps this report intends to fill is an investigation into machine learning based network intrusion systems which classifies traffic into more specific classes, does not rely on payload content for classification, and is up to date with modern attacks which will help detect newer iterations of attacks.

3 Research Objectives

The research objectives of this study are as defined below.

1. Study and Examine the various frameworks for creating and training Machine Learning models.

2. Design and Train a model to effectively detect and categorize network traffic.

3. Generalize the model such that it can successfully detect new unseen types of attacks.

4 Methodology

4.1 Data Analytics

The dataset used for experimentation was obtained from [2], which was created for the development of security analytics applications in real IoT operations. It contains 33 types of attacks which were executed in an IoT topology composed of 105 devices, the dataset is composed of 46686579 rows and 46 columns. It was acquired from the University of New Brunswick who created and host the dataset.
Figure 2 describes each feature in the dataset, for the reader to understand training data better.

The data was gathered using the topology found in Figure 3.

#	Feature	Description
1	ts	Timestamp
2	flow duration	Duration of the packet's flow
3	Header Length	Header Length
4	Protocol Type	IP, UDP, TCP, IGMP, ICMP, Unknown (Integers)
5	Duration	Time-to-Live (ttl)
6	Rate	Rate of packet transmission in a flow
7	Srate	Rate of outbound packets transmission in a flow
8	Drate,	Rate of inbound packets transmission in a flow
9	fin flag number	Fin flag value
10	syn flag number	Syn flag value
11	rst flag number	Rst flag value
12	psh flag numbe	Psh flag value
13	ack flag number	Ack flag value
14	ece flag numbe	Ece flag value
15	cwr flag number	Cwr flag value
16	ack count	Number of packets with ack flag set in the same flow
17	syn count	Number of packets with syn flag set in the same flow
18	fin count	Number of packets with fin flag set in the same flow
19	urg coun	Number of packets with urg flag set in the same flow
20	rst count	Number of packets with rst flag set in the same flow
21	HTTP	Indicates if the application layer protocol is HTTP
22	HTTPS	Indicates if the application layer protocol is HTTPS
23	DNS	Indicates if the application layer protocol is DNS
24	Telnet	Indicates if the application layer protocol is Telnet
25	SMTP	Indicates if the application layer protocol is SMTP
26	SSH	Indicates if the application layer protocol is SSH
27	IRC	Indicates if the application layer protocol is IRC
28	TCP	Indicates if the transport layer protocol is TCP
29	UDP	Indicates if the transport layer protocol is UDP
30	DHCP	Indicates if the application layer protocol is DHCP
31	ARP	Indicates if the link layer protocol is ARP
32	ICMP	Indicates if the network layer protocol is ICMP
33	IPv	Indicates if the network layer protocol is IP
34	LLC	Indicates if the link layer protocol is LLC
35	Tot sum	Summation of packets lengths in flow
36	Min	Minimum packet length in the flow
37	Max	Maximumpacket length in the flow
38	AVG	Average packet length in the flow
39	Std	Standard deviation of packet length in the flow
40	Tot size	Packet's length
41	IAT	The time difference with the previous packet
42	Number	The number of packets in the flow
43	Magnitue	(Average of the lengths of incoming packets in the flow + Average of the lengths of outgoing packets in the flow) ** 0.5
44	Radius	(Variance of the lengths of incoming packets in the flow + Variance of the lengths of outgoing packets in the flow) ** 0.5
45	Covariance	Covariance of the lengths of incoming and outgoing packets
46	Variance	Variance of the lengths of incoming packets in the flow / The variance of the lengths of outgoing packets in the flow
47	Weight	Number of incoming packets * Number of outgoing packets

Figure 2: Data Description for CIC IoT 2023 Dataset [2]

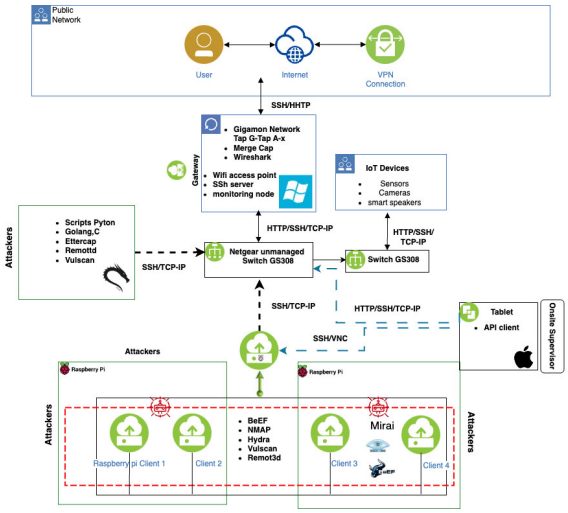

Figure 3: Dataset Methodology Topology from [2]

The data distribution is as seen in Figures 4 and 5. It is seen that DDoS and DoS compose the majority of the dataset, which follows the nature of the attacks, with DoS and DDoS relying on quantity to overwhelm the network.

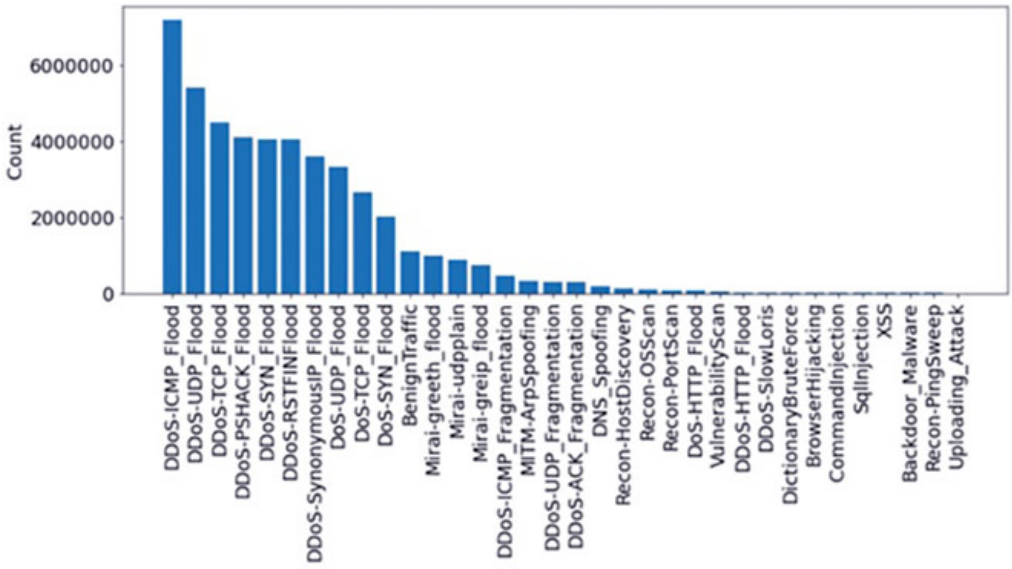

Figure 4: Dataset Counts [2]

4.2 Dataset Labeling

The database labeled network activity as one of 32 types of attacks or benign traffic. However, many of the attacks share similar characteristics and can be grouped together, the 33 types of activity were categorized into 8 classes, 7 types of attacks and benign traffic.

Network attacks come in various forms, the ones investigated in this study are: Distributed Denial of Service (DDoS), Denial of Service (DoS), Mirai botnet attacks, reconnaissance (Recon) attacks, spoofing, web attacks, and brute force attacks.

- **DoS**: A DoS attack aims to make a network service unavailable to its intended users by overwhelming the resources of the target with a flood of illegitimate requests. This causes the system to slow down or crash, disrupting access for legitimate users [14].
- **DDoS**: A DDoS attack is similar to a DoS attack but is launched from multiple compromised computers distributed across various locations. These coordinated attacks can be more challenging to mitigate due to their scale and the diverse origins of the attack traffic [14].
- **Mirai**: Mirai is a type of malware that infects IoT devices and turns them into a botnet used to conduct large-scale DDoS attacks. It exploits weak security practices in IoT devices, such as default usernames and passwords [15].

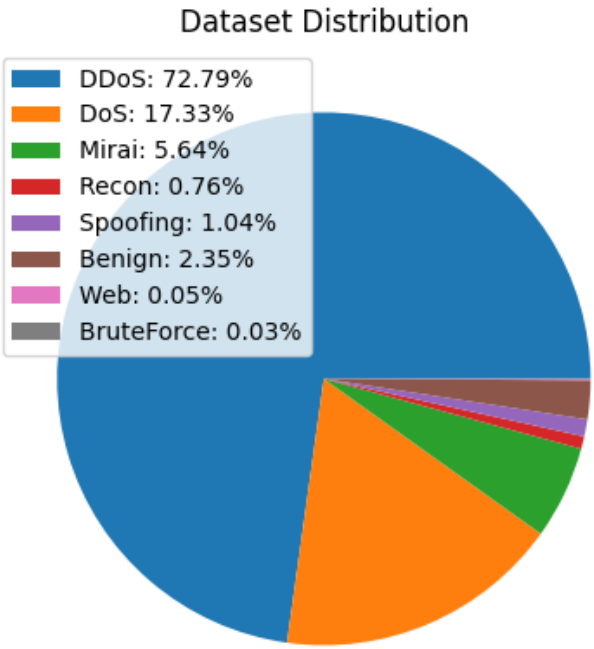

Figure 5: Dataset Distribution

- **Recon**: Recon attacks involve gathering information about a target network to identify vulnerabilities and entry points. Techniques include network scanning, port scanning, and ping sweeps to map out network infrastructure and potential weaknesses [16].

- **Spoofing**: Spoofing involves impersonating another device or user on a network to gain unauthorized access to resources or to spread malware. Common types include IP spoofing, where the attacker disguises their IP address, MAC spoofing where the attacker alters their perceived factory-assigned Media Access Control (MAC) address, DNS Spoofing where the attacker makes a Domain Name System (DNS) entry to point to another IP than it would be supposed to point to, and email spoofing, where attacker's emails appear to come from trusted sources [17].

- **Web**: These attacks target web applications to exploit vulnerabilities in code or architecture. Common types include SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF), which can lead to data breaches, unauthorized access, or service disruptions [18].

- **Brute Force**: A brute force attack attempts to gain unauthorized access to a system by systematically trying all possible combinations of passwords or encryption keys. This method relies on computational power and can be time-consuming but is often effective if strong passwords are not used [19].

4.3 Feature Selection

Feature selection is a crucial step in building effective machine learning models, as it involves identifying the most relevant features that contribute to the model's predictive power [20]. In the experimentation two distinct feature selection approaches to handle different types of training data were used: per packet training and per flow training.

4.4 Per Packet Training

4.4.1 Objective

To classify individual packets based on their features, so that each incoming packet can be evaluated and categorized immediately.

4.4.2 Selected Features

For per packet training, features that are directly associated with individual packets were selected. These features include, but are not limited to header length, time-to-live (ttl), protocol type, and flags.

4.4.3 Rationale

The selection of these features is based on the need to capture the characteristics of each packet independently. By focusing on individual packets, the model can make real-time decisions based on the immediate packet data.

4.5 Per Flow Training

4.5.1 Objective

To classify flows of packets that belong to the same session or connection.

4.5.2 Selected Features

For per flow training, the features were aggregated over flows, where a flow is defined as a sequence of packets sharing common attributes such as source and destination IP addresses, ports, and protocol type within a specific time window. The selected features include flow duration, counters such as ACK count, total number of packets, total bytes transferred, duration of the flow, average packet size, flow start and end times, and flow inter-arrival times.

4.5.3 Rationale

The aggregation of features over flows helps in capturing the behavior of network traffic over time. This approach provides a broader context for the model to understand patterns associated with prolonged sessions or connections, which is particularly useful for detecting certain types of attacks that manifest over multiple packets such as DoS.

4.6 Dataset Train Test split

In order to detect issues such as overfitting and ensure that the model's evaluation is unbiased and provides a realistic assessment of its generalization capabilities, a 70:30 train test split was utilized. Overfitting occurs when a model learns the training data too well, including its noise and outliers, leading to poor performance on new, unseen data. By splitting the dataset in this ratio, we allocate 70% of the data for training,

allowing the model to learn from a substantial amount of data, while reserving 30% for testing to evaluate its performance. This split strikes a balance, ensuring that there is enough data to train the model effectively while also maintaining a sufficient and representative test set to assess the model's ability to generalize to new data. This method provides a robust mechanism to gauge the model's predictive power and ensures that the evaluation metrics reflect its true performance, rather than being skewed by the specific characteristics of the training data.

4.7 Model Selection

It is vital to select the correct model for the task, as each one has its own strengths, weaknesses, and nuisances. The models evaluated in this study include:

- **Artificial Neural Network (ANN)**: ANNs consist of multiple layers of interconnected neurons that process input features and learn patterns through back-propagation, if there is a hidden layer between the input layer and output layer then it is considered Deep Learning [21]. The utilized model had 3 total layers, input, output, and 1 hidden layer, making is a deep learning model.

- **Convoluted Neural Network (CNN)**: CNNs are a specialized type of ANN designed to process structured grid data. CNNs utilize convolutional layers that apply filters to input data to capture spatial hierarchies of features automatically. Though most frequently used for images, videos, and other data which is visual or can be expressed as a grid, they can be used for many other functions such as categorization of 1-dimensional data [22]. The utilized CNN had 2 convolution layers and 1 regular hidden layer.

- **Recurrent Neural Network (RNN)**: RNNs are designed to handle sequential data by maintaining a memory of previous inputs, making them ideal for tasks involving time series or ordered data, there are different types of RNNs such as Long Short-Term Memory (LSTM), gated recurrent unit (GRU) and many others [23]. This may prove useful for attacks which are repeated such as DoS or DDoS or multi-stage attacks. The used RNN has 2 LSTM layers and 1 regular hidden layer.

- **Decision Tree**: A Decision Tree splits the data into subsets based on the value of input features, creating a tree-like model of decisions. Each node in the tree represents a feature, each branch represents a decision rule, and each leaf node represents an outcome or class label. The primary objective is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features [24].

- **Random Forest**: A Random Forest is an ensemble learning method, which means it utilizes multiple models simultaneously. A Random Forest operates by constructing multiple decision trees during training and outputting the mode of the classes of the individual trees. By averaging the results of many decision trees, Random Forests improve predictive accuracy and control overfitting, making them more robust and accurate than individual decision trees at the cost of higher computational requirements [24].

- **K-Nearest Neighbors (KNN)**: Classifies data points based on the majority class among its k-nearest neighbors. The distance metric used for this research was Euclidean and is used to find the nearest neighbors in the feature space. This method is non-parametric and makes no assumptions about the underlying data classification. [25].

- **Support Vector Machine (Linear-SVC)**: Support Vector Machines are categorized by their usage of kernels, which allow the model to find the optimal boundary, or hyperplane, which separates different classes in the dataset. In the case of a linear kernel, as was used in this case, the model seeks to find a linearly defined hyperplane that best divides the data into distinct classes [26]. The model was constructed following the advice and results found in [27].

- **Stochastic Gradient Descent (SGD) Classifier**: SGD is an optimization algorithm used to minimize the loss function in machine learning models. Unlike batch gradient descent, which computes the gradient using the entire dataset, SGD updates the model parameters iteratively using only one or a small batch of data points at a time. This makes SGD particularly efficient for large and sparse datasets, as well as providing superior generalizability [28]. However, it is unlikely that the full capabilities of SGD will be seen in this context due to the dataset already being split into chunks with the models being trained on only one chunk at a time, leading to SGD methodology already being implemented in them to some extent. The SGD model used a logistic regression probabilistic classifier.

- **Logistic Regression**: An LR model models the probability of class membership using a logistic function which outputs values between 0 and 1. The primary goal of logistic regression is to find the best-fitting model to describe the relationship between the binary dependent variable and one or more independent variables. However, it is quite simple and prone to overfitting, and not all relationships can be expressed logistically [29].

Each model was trained and tested on a dataset comprising various types of network attacks and benign traffic. The performance of these models was evaluated

using metrics such as accuracy, precision, recall, and F1 score. The impact of dataset imbalance was also examined, as indicated by the support metric for each attack type.

4.8 Environmental Factor Specifications

The network intrusion detection system was implemented and tested on Windows 10 Pro with the Intel core 10th generation i7 processor, 16GB RAM, and 512GB SSD. The machine learning and deep learning models are implemented in python 3.10 using Jupyter Notebook with SciKit-Learn version 1.4.2, Keras 3.3.3, and TensorFlow 2.16.1 libraries.

5 Results

Due to time constraints, there was only sufficient time to train and test

- 3 ANN models
- 3 CNN models
- 1 RNN model
- 3 RF models
- 1 DT model
- 3 KNN models
- 1 SVM model
- 1 SGD model

5.1 Performance Metrics

The performance of machine learning models can be evaluated using different metrics, mainly using counts for True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN).

- **Accuracy**: Accuracy is defined by the formula

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

  It describes what proportion of predictions were correct out of all predictions made [30].

Model	avg Accuracy (%)	avg Precision (%)	avg Recall (%)	avg f1 score (%)
ANN	95.93	96.93	95.93	96.10
CNN	96.66	97.04	96.66	96.51
RNN	99.19	99.19	99.19	99.15
DT	99.38	99.39	99.38	99.38
RF	99.19	99.15	99.19	99.10
KNN	94.71	94.60	94.71	94.61
SVM	82.51	81.06	82.51	77.25
LR	83.17	82.20	83.17	78.54
SGD	84.65	85.45	84.65	84.72

Table 1: Metrics for PerFlow models

- **Precision**: Precision is defined by the formula

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

  It is the proportion of true positive predictions among all positive predictions, showing how accurate the positive predictions are [30].

- **Recall**: Precision is defined by the formula

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

  It is the proportion of true positive predictions among all actual positive instances. It measures the classifier's ability to identify positive instances correctly [30].

- **F1 Score**: F1-Score is defined by the formula

$$\frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} = \frac{2 * Precision * Recall}{Precision + Recall}$$

  The F1-score is the harmonic mean of precision and recall, providing a metric that considers false positives and false negatives [30].

5.2 Comparison between models

The performance metrics of the various machine learning models are shown below in Figure 6 as well as Tables 1 and 2.

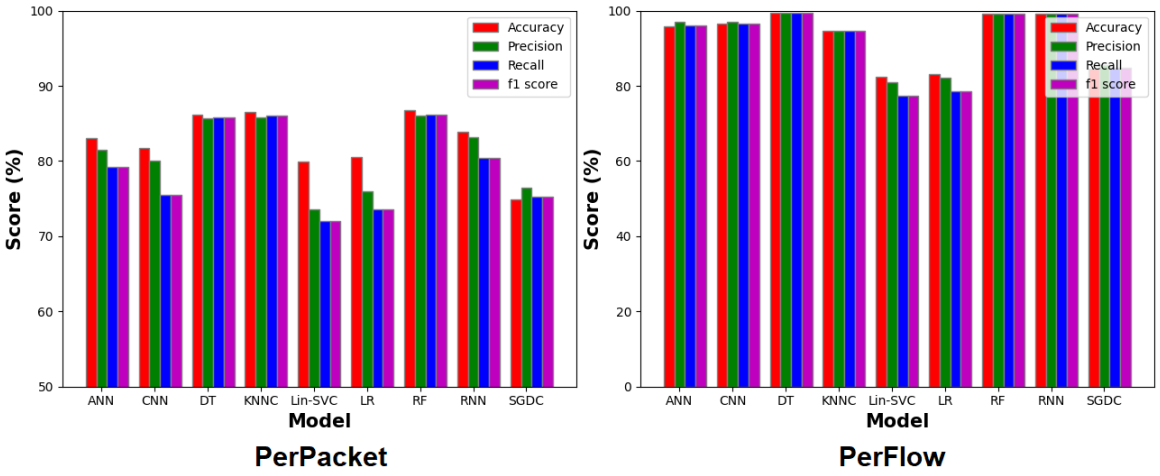

Figure 6: Performance Metrics

Model	avg Accuracy (%)	avg Precision (%)	avg Recall (%)	avg f1 score (%)
ANN	83.05	81.46	83.05	79.15
CNN	81.68	80.01	81.68	75.55
RNN	83.91	83.21	83.91	80.40
DT	86.18	85.64	86.18	85.82
RF	86.72	86.02	86.72	86.19
KNN	86.53	85.82	86.53	86.03
SVM	79.90	73.60	79.90	72.05
LR	80.54	75.97	80.54	73.55
SGD	74.92	76.51	74.92	75.23

Table 2: Metrics for PerPacket models

5.2.1 Context to results

The results were taken from the categorization of 30% of the dataset on which the models were not trained. The data was categorized to 8 possible classes, benign traffic and 7 types of attacks.

5.2.2 Analysis of results

PerFlow feature selection gives noticeably better results compared to PerPacket, with the best models in both being the ANN, CNN, RNN, DT, RF, and KNN. With the given results, the Decision Tree was the best for PerFlow feature selection, with a precision of 99.39% and f1 score of 99.38%, and the best model for PerPacket feature selection was the Random Forest with a precision of 86.02% and f1 score of 86.19%. However, it cannot be decided which approach is best due to the random nature of some of the models in combination with the low sample size, but the general trends can be seen. When compared to the existing research, the SVM did not perform as well as the one present in [9], but RNN, DT, and RF give superior results compared to the best SVMs found in their research. Additionally, comparing to the results found in [10] to the PerFlow results, comparatively, some of the models created in this research indicate equal or superior classification, with the DT and RF both having an f1 score around 99%, and KNN at 94%. However, their overall best model was the DT with an accuracy score of 99.05% and F1 score of 0.99, but the DT model created for this research has superior scores to it in every metric, implying that it is slightly superior at classification compared to what was created by Dhanya K. A et. al..

5.2.3 ROC and AUC

The Receiver Operating Characteristic (ROC) curve is a graphical plot used to assess the performance of a binary classifier. It illustrates the trade-off between the True Positive Rate (TPR) and the False Positive Rate (FPR) across different threshold settings. The True Positive Rate measures the proportion of actual positives correctly identified, while the False Positive Rate measures the proportion of actual negatives incorrectly identified as positives. The Area Under the ROC Curve (AUC) quantifies the overall ability of the classifier to distinguish between classes. An AUC of 0.5 suggests no discriminative power, equivalent to random guessing, while an AUC of 1.0 indicates perfect classification [31].

To convert the 8 Class classification to a binary classification, the models' predictions were processed as prediction==Benign, making the models' predictions binary.

As is seen in Figure 7, the ROC curves for both feature selections and all models show very high discriminative power, with the highest being the CNN for PerFlow

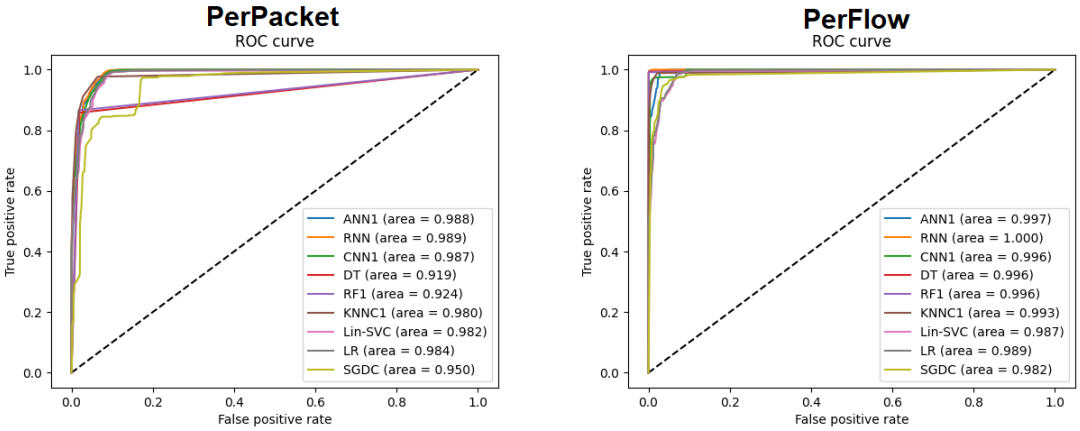

Figure 7: ROC of the models

feature selection with an AUC of 0.987, meaning that 98.7% of the time the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one. The ANN, RNN, CNN, KNN, Linear SVM, and LR all show very good results in the range of [0.980, 0.987], the RF, DT, and SGD models show decent results but have much more room for improvement. As for the PerFlow feature selection, the RNN gave an AUC of 1.0 which implies perfect classification, but is more likely due to rounding. Excluding RF, the best PerFlow model is the ANN with an AUC of 0.997, all other models also show very close results in the range [0.982, 0.996], showing that they are all very accurate in their predictions. It is seen that the PerFlow feature selection gives substantially better results with this metric, especially when comparing the DTs (0.919 for PerPacket compared to 0.996 for PerFlow) and RFs (0.924 for PerPacket compared to 0.996 for PerFlow).

5.2.4 Trade-off between True Positives and False Positives

When traffic is being classified by the model as benign or not, a true positive is benign traffic that is labeled as benign, a true negative is when benign traffic is labeled as an attack, a false positive is when an attack is labeled as benign, and a false negative is when benign traffic is labeled as an attack. It is obvious that it is better to have higher true positive and true negative rates as that shows correct classification. However, when comparing the significance of them all, the most important one is the false positive. Given that the objective of a network intrusion detection system is to stop all possible intrusions, and that a single attack let through is more devastating to a network that a benign packet being rejected, it is vital to minimize false positives, arguably at the expense of true positives.

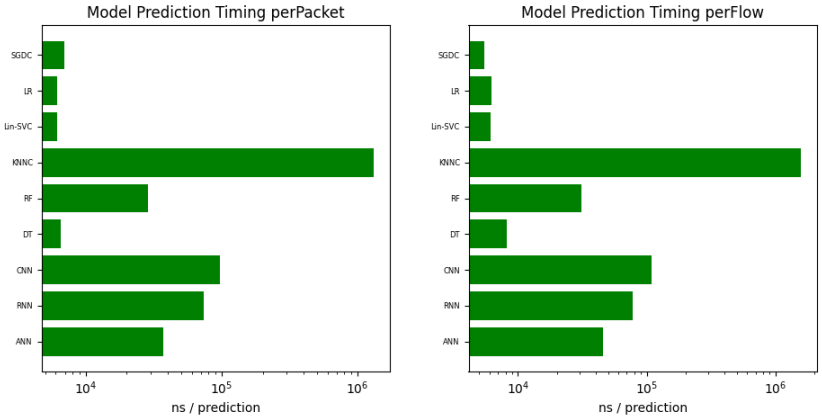

Figure 8: Average Prediction Times

5.2.5 Prediction Timings

If a model is to be used in a live network intrusion detection system, it must be able to classify the incoming traffic in a reasonable amount of time. Though the exact prediction times will vary from the experimental results due to external factors such as hardware and available computational resources, it is still valuable to compare prediction times between models. As seen in figure 8 (note that the scale is logarithmic), the timings between feature selections are similar, as are the rankings, the fastest is SGD, for PerPacket, relative to SDG time, SVM is 11% slower, LR is 14% slower, DT is 48% slower, RF is 465% slower, ANN is 740% slower, RNN is 1311% slower, CNN is 1900% slower and KNN is 28763% slower. Therefore, in the case that computational resources are limited or there is a high amount of traffic to categorize, it makes sense to peruse a faster model. However, time is not an issue unless computational resources are limited or fast prediction timing is crucial.

5.2.6 Decision Tree and Random Forest Feature Importance

Feature importances can be extracted from trained Decision Trees and Random Forests to provide significant insights into the factors most influential in predicting the target variable. These importances are quantified based on the contribution of each feature to reducing impurity or improving the model's accuracy. High feature importance indicates that a feature significantly enhances the model's predictive power, suggesting its strong influence on the decision-making process. This understanding can be leveraged to identify the key drivers within the dataset, offering a deeper comprehension of the underlying patterns and relationships [32].

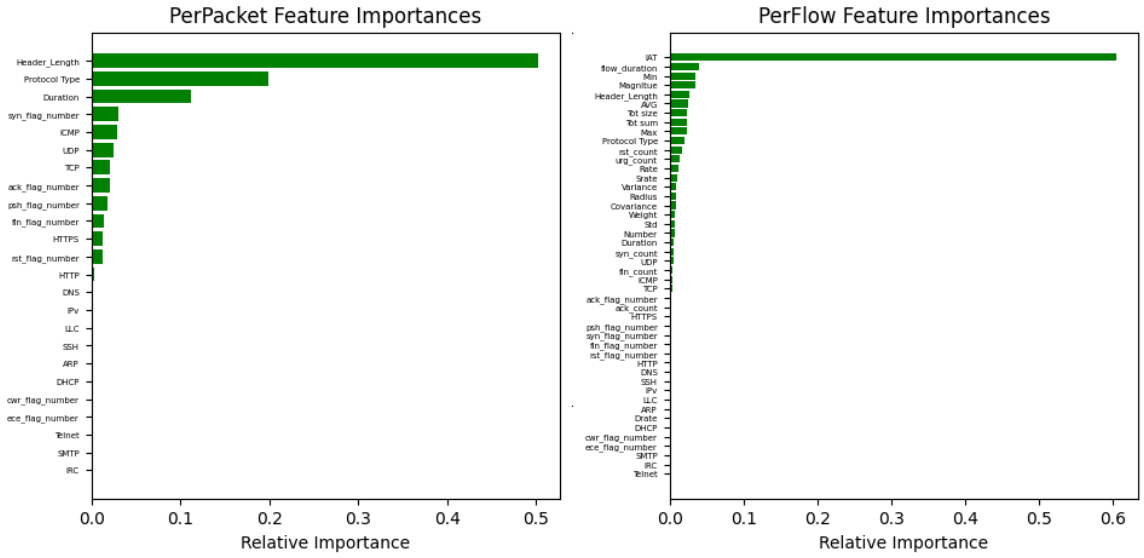

Figure 9: Feature Importance

Figure 9 displays the average feature importances over the 3 RF models and 1 DT model. It is seen that the PerFlow models heavily rely on IAT, flow duration, and other features which relate to the flow rather than individual packet contribute to the majority of the prediction, which is likely one of the reasons behind the superior performance of the PerFlow feature selection. The PerPacket is heavily reliant on header length, protocol type, and duration, which have some significance in the PerFlow feature selection but are overshadowed heavily. From this information, it can be drawn that the nature of the features significantly impacts the performance and reliability of the model. Specifically, features that provide broader contextual information about the flow tend to offer better predictive capabilities, emphasizing the importance of comprehensive data representation in improving detection accuracy.

5.3 Evaluation of results

For PerFlow feature selection, the Decision Tree model achieved the highest performance with an average accuracy of 99.38%, precision of 99.39%, recall of 99.38%, and F1 score of 99.38%. The Random Forest and Recurrent Neural Network models also performed exceptionally well, with accuracies of 99.19% and 99.19%, respectively. These high scores indicate that the models based on PerFlow features are highly effective in classifying network traffic with minimal misclassifications. Notably, the Decision Tree's model's superior performance underscores its robustness in handling the broader contextual information provided by PerFlow features.

In contrast, the PerPacket feature selection models showed comparatively lower performance. The best-performing model, the Random Forest, achieved an average accuracy of 86.72%, precision of 86.02%, recall of 86.72%, and F1 score of 86.19%. While these results are still strong, they are noticeably lower than those of the PerFlow models. This disparity suggests that PerPacket features, which focus more on individual packet characteristics such as header length and protocol type, may not capture the comprehensive patterns necessary for optimal classification as effectively as PerFlow features.

The Stochastic Gradient Descent model was the fastest, making it suitable for high-speed networks with limited computational resources. In contrast, the more complex models despite their slower prediction times, offer higher accuracy and robustness. This trade-off between speed and accuracy must be considered when selecting models for real-time intrusion detection. However, the Decision Tree provided superior results in both metrics, with the best results for PerFlow and third best for PerPacket, as well as being only 48% slower than Stochastic Gradient Descent, with its only flaw being how simple it is which makes the patters it searches for more predictable.

5.4 Effectiveness on VPN Traffic

5.4.1 VPN Data

A Virtual Private Network (VPN) creates a secure, encrypted connection over an unsecured network. VPNs encrypt the data to be sent, use tunneling protocols to encapsulate and transmit the data packets, and act as a middleman between the client and destination server. The primary difference between unencrypted and encrypted data is the payload, which is not used for this classification in this case, other differences include compression of the payload and standardization of packet size, which alters the length of the packet [33], as well as well as standardization of timing intervals and behaviours to reduce difference in patterns between different activities [34]. Additionally, due to how VPNs work, a major difference would be the source ip, instead of the actual server for the data, it is the VPN server which acts as the middleman.

5.4.2 Encrypted Traffic Dataset

The models were tested on data taken from [35] which contains raw pcap files for benign and attack traffic and was created to fill the gap of datasets for VPN traffic [36]. From the pcap files the expected features for testing were extracted, and the data was labeled as "Benign" or "Attack".

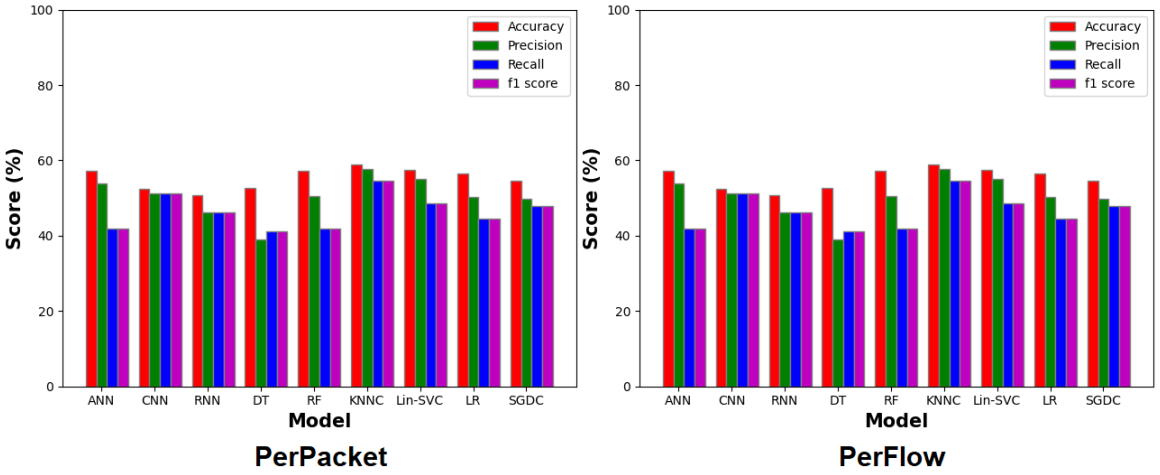

Figure 10: Model Metrics for VPN Traffic

Model	avg Accuracy	avg Precision	avg Recall	avg f1 score
ANN	59.78	59.58	59.78	53.33
CNN	58.94	57.58	58.94	55.70
RNN	55.42	51.52	55.42	48.82
DT	51.40	45.76	51.40	45.52
RF	56.80	53.39	56.80	47.32
KNN	52.78	46.95	52.78	45.99
SVM	57.65	56.15	57.65	55.42
LR	59.87	59.68	59.87	53.74
SGD	60.60	60.10	60.60	56.33

Table 3: PerPacket Metrics for VPN Traffic

5.4.3 Results

The models were tested on the encrypted traffic, and predictions were binarized to "Benign" or "Attack". The resulting metrics can be seen in Figure 10 as well as Tables 3 and 4.

5.4.4 Discussion of Results for VPN Traffic

It is seen that the best-performing model for PerPacket attack detection was the SGD model with an average accuracy of 60.60%, precision of 60.10%, recall of 60.60%, and f1 score of 56.33%, and for PerFlow detection was the KNN model which achieved an accuracy of 59.01%, precision of 57.73%, recall of 59.01%, and f1 score of 54.49%.

Model	avg Accuracy	avg Precision	avg Recall	avg f1 score
ANN	57.27	53.98	57.27	41.96
CNN	52.48	51.28	52.48	51.34
RNN	50.81	46.30	50.81	46.29
DT	52.65	38.88	52.65	41.24
RF	57.23	50.55	57.23	41.87
KNN	59.01	57.73	59.01	54.49
SVM	57.42	54.98	57.42	48.70
LR	56.39	50.27	56.39	44.48
SGD	54.52	49.87	54.52	47.76

Table 4: PerFlow Metrics for VPN Traffic

Though overall the models are better than random guessing, the results show a clear loss of predictive power when tested on encrypted and unseen traffic compared to the original dataset. This is expected, as a differing dataset is bound to have different patterns that the models were not exposed to during training. As was seen in Feature Importance earlier, the PerPacket models rely heavily on Header Length and Duration, which would be changed during encryption. This difference highlights the need for models to generalize well across various traffic patterns and not be overly reliant on specific training data. This experimentation proves it is difficult to have both the granularity of specific attack type classification, and the generalized detection of unseen, unknown attacks, and a tradeoff might need to be made between accurately detecting attacks by learning general patterns and knowing what the exact attacks are by increasing the number of classes, but reducing impact of shared patterns between attacks.

6 Discussion

6.1 Threats to the validity of results

6.1.1 Low Sample Size

Due to the computational intensity of machine learning alongside technological and time constraints, there was insufficient time to train a large amount of models, making the results more prone to be affected by outliers and deviating from the true results.

6.1.2 Impact of Dataset Imbalance

The imbalance in the dataset, as shown by the support metrics, affects the performance of the models, particularly for less frequent attack types such as Recon, Spoofing, Web, and Brute Force. For instance, this is seen in the ANN model which showed reduced recall and F1 scores for these categories. This is indicative of the models' difficulty in accurately detecting less common attacks, likely due to their limited representation in the training data.

6.1.3 Patterns within Dataset

It can be seen in the Decision Tree Feature Importance subsubsection of Results that the Decision Tree models are heavily reliant of very few variables, which can be extended to the other models as well. There may exist patterns within this dataset which the models picked up on and learned, which are not present in other datasets or real-world scenarios.

6.2 Mitigation

6.2.1 Low Sample Size

The low sample size for reach type of machine learning model only affects the numerical model performance metrics, experimentation can be continued in order for the averages to approach the true values, but it does not impact the real-world viability.

6.2.2 Impact of Dataset Imbalance

The purpose of the models is to detect network intrusions, the classification is valuable and would be better to correctly classify all traffic, but the models have proven to be able to classify traffic as Benign or Attack, and fundamentally that is the minimal viable functionality for them.

6.2.3 Patterns within Dataset

To combat patters exclusive to the dataset, the models can be further trained on different datasets or real-world data. The problem can be solved simply by training the models more and more, thus diluting old patterns.

6.3 Implications of results

The successful utilization of Machine Learning algorithms for detection and classification of network intrusions, using only the header and metadata for the classification,

contributes to the viability of a machine learning based network intrusion detection system which is not reliant on packet payload and is less vulnerable to zero-day attacks. This opens up opportunities for future research, development, and optimization of a network intrusion detection system which is sufficiently viable to be used in real-world applications.

6.4 Limitations of results

6.4.1 Conditions of Limitations

- **Over dependence on Header Information**: By focusing solely on the header, the models could misclassify or fail to detect attacks which are discernible using the payload, which could reduce the system's ability to detect attacks that appear to have a benign header.
- **Dynamic and evolving threats**: Though focusing on header information allows the models to be more generalized in their classification of traffic and detection zero-day attacks, the models are still ultimately trained using historical data and might not be sufficiently generalized to detect new attacks which follow different patterns.
- **Analysis and Understanding of Models**: Many of the machine learning models used, such as the CNN, ANN, and RNN, act as a blackbox when making predictions, making it difficult to properly understand the behaviour or find flaws and vulnerabilities.
- **Reliance on Patterns**: Since the models seek to find the classification function, and were trained that certain patterns indicate benign traffic, if an attacker learns these patterns, and sets the headers of their attacks to follow the benign pattern the models were trained on, then the model will be easily bypassed.
- **Real Time Classification and Hardware**: For the models to be implemented as a component to a practical real world network intrusion detection system, they must be able to handle large amounts of predictions very fast, depending on time requirements, amount of data, available hardware, and computing resources, the model can very easily become a bottleneck in the intrusion detection system

6.4.2 Conditions counter measurement

- **Over dependence on Header Information**: If a model struggles with detecting attacks which are detectable using payload, then it can be combined

with other models which look at other information to predict as an ensemble.

- **Dynamic and evolving threats**: Not much can be done to mitigate completely new threats, however, by constantly training the model with new traffic information, it will learn relatively new attack patterns and be able to quickly adapt to changes and developments in the environment.
- **Analysis and Understanding of Models**: If the model acting like a blackbox is a problem, then more transparent models can be used. As seen in the Results section, the Decision Tree Feature Importance, it is possible to analyze how a decision tree or forest of decision trees classify the data, allowing for much better transparency and detection of vulnerabilities such as over-reliance on few variables [32].
- **Reliance on Patterns**: By combining multiple models that were trained differently in an ensemble, it becomes much more difficult to reverse engineer what benign traffic looked like in training data for each model in the ensemble.
- **Real Time Classification and Hardware**: The computational resource problem can be solved by simply increasing computational resources. By utilizing newer end technology, GPUs, and selecting a model which won't exceed allocated resources, the computational bottleneck will not form at the model's prediction stage.

6.5 Generalizability of results

The generalizability of this report is rooted in its comprehensive exploration of machine learning models applied to network intrusion detection systems. By classifying network traffic into specific attack types, the research addresses the critical need for precise and actionable threat identification. This study also emphasizes the removed reliance on packet payload for classification of traffic. Additionally, by incorporating modern attack patterns and being better fitted for zero-day threats, the findings remain relevant and adaptive to the ever-evolving cybersecurity landscape.

7 Conclusions

In conclusion, this research presents a significant advancement in the field of network intrusion detection systems by leveraging machine learning models to enhance the accuracy and robustness of attack classification. The study addresses critical gaps in existing research by focusing on three key areas: the detailed classification of

network traffic into specific attack types, the removed reliance on packet payload, and the inclusion of modern threats which help with detection of zero-day attacks in the detection framework. Through rigorous experimentation with various machine learning models, including ANN, CNN, RNN, Decision Trees, Random Forest, and others, we have demonstrated the effectiveness of these approaches in accurately identifying and classifying network intrusions.

The results show that a machine learning-based network intrusion detection system can provide high accuracy and precision, even with complex and evolving attack vectors. By using header information for detection, the system is particularly effective in environments where payload data may be encrypted or obfuscated, making traditional detection methods less viable. Moreover, the system's ability to generalize to new attack patterns ensures it remains resilient against the latest cyber threats, thus offering a future-proof solution for network security.

This research not only fills the existing gaps by providing a more detailed and effective intrusion detection mechanism but also sets a foundation for further advancements in network intrusion detection systems. Future work can build upon this framework to explore even more sophisticated machine learning techniques and extend the system's capabilities to encompass broader aspects of network security. Overall, this study represents a pivotal step towards developing more intelligent, adaptive, and comprehensive network intrusion detection systems.

8 Future Works and Lessons Learned

This work is an introduction to the utilization of machine learning for the purpose of detection and classification of network traffic. It can serve as the first step for the creation or improvement of existing network intrusion detection systems.

8.1 Future Works

- **Expanding on Encrypted Data and VPNs**: The models rely exclusively on header information for the training and prediction, making them more effective for handling encrypted traffic. However, to properly address encrypted traffic and VPNs, the training would need to include more diverse traffic data for various forms of encryption including VPN traffic.

- **Integrate live packet processing**: By monitoring live network activity, processing the data, and making prediction for the classification, it can be approximated how well the model would function in a real-world network intrusion detection system.

- **Further improve models**: Though the results are overall adequate, they can be further improved, and models can be generalized by training on additional data which will contain differing patterns, and further experimenting on feature selection.
- **Ensemble based approach**: To further increase effectiveness of the network intrusion detection system, the models present here can be combined with existing or new models for with the objective that the models will compensate for each other's weaker areas. For example, the models present here which classify based on header information can be combined with models or algorithms which classify based on the payload for a more thorough detection system.
- **Countering model reverse engineering**: A threat to any network intrusion detection system is reverse engineering, if an attacker knows how the system works and what patterns are present within the system's classification then it becomes much more trivial to outsmart it. For this purpose, a more complex model which is closer to a blackbox can be used, at the cost of administrator understanding of the model, or techniques such as adding noise can be employed at the cost of the models' predictive power.

8.2 Lessons Learnt

This research further demonstrates the possibility for the usage of Machine Learning for the purpose of Network Intrusion Detection, by itself or in combination with other algorithms or models.

9 Acknowledgments

All help received in this research is from my supervisor:

Associate Prof. Zubair Fadlullah,
Dept. of Computer Science, Western University.

References

[1] N. Hoque, M. H. Bhuyan, R. Baishya, D. Bhattacharyya, and J. Kalita, "Network attacks: Taxonomy, tools and systems," *Journal of Network and Computer Applications*, vol. 40, pp. 307–324, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804513001756

[2] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, A. A. Ghorbani., ""CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment"," *Sensor (2023) – (submitted to Journal of Sensors)*, 2023.

[3] T. Fadziso, U. R. Thaduri, S. Dekkati, V. Ballamudi, and H. Desamsetti, "Evolution of the cyber security threat: An overview of the scale of cyber threat," vol. 3, pp. 1–12, 09 2023.

[4] E. Hodo, X. Bellekens, A. Hamilton, P.-L. Dubouilh, E. Iorkyase, C. Tachtatzis, and R. Atkinson, "Threat analysis of iot networks using artificial neural network intrusion detection system," in *2016 International Symposium on Networks, Computers and Communications (ISNCC)*, 2016, pp. 1–6.

[5] M. Garuba, C. Liu, and D. Fraites, "Intrusion techniques: Comparative study of network intrusion detection systems," in *Fifth International Conference on Information Technology: New Generations (itng 2008)*, 2008, pp. 592–598.

[6] B. Molina-Coronado, U. Mori, A. Mendiburu, and J. Miguel-Alonso, "Survey of network intrusion detection methods from the perspective of the knowledge discovery in databases process," *IEEE Transactions on Network and Service Management*, vol. 17, no. 4, pp. 2451–2479, 2020.

[7] I. El Naqa and M. J. Murphy, *What Is Machine Learning?* Cham: Springer International Publishing, 2015, pp. 3–11. [Online]. Available: https://doi.org/10.1007/978-3-319-18305-3_1

[8] O. N. Elayan and A. M. Mustafa, "Android malware detection using deep learning," *Procedia Computer Science*, vol. 184, pp. 847–852, 2021, the 12th International Conference on Ambient Systems, Networks and Technologies (ANT) / The 4th International Conference on Emerging Data and Industry 4.0 (EDI40) / Affiliated Workshops. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050921007481

[9] Y. Che, "Machine learning based network attacks classification," in *2023 IEEE 3rd International Conference on Power, Electronics and Computer Applications (ICPECA)*, 2023, pp. 1198–1203.

[10] K. Dhanya, S. Vajipayajula, K. Srinivasan, A. Tibrewal, T. S. Kumar, and T. G. Kumar, "Detection of network attacks using machine learning and deep learning models," *Procedia Computer Science*, vol. 218, pp. 57–66, 2023, international Conference on Machine Learning and Data Engineering. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050922024942

[11] K. Almulla and S. Modak, "Cyber-attack detection in network traffic using machine learning," 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:259940186

[12] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, p. 102419, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214212619305046

[13] W. Khan and M. Haroon, "An unsupervised deep learning ensemble model for anomaly detection in static attributed social networks," *International Journal of Cognitive Computing in Engineering*, vol. 3, pp. 153–160, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S266630742200016X

[14] Cloudflare, "Cloudflare, how to ddos — dos and ddos attack tools," available online, Last accessed on 2024-07-02. [Online]. Available: https://www.cloudflare.com/learning/ddos/ddos-attack-tools/how-to-ddos/

[15] ——, "What is the mirai botnet?" available online, Last accessed on 2024-07-02. [Online]. Available: https://www.cloudflare.com/learning/ddos/glossary/mirai-botnet/

[16] Sai, "What is reconnaissance in cyber security?" 2022, available online, Last accessed on 2024-07-02. [Online]. Available: https://www.netsecurity.com/what-is-reconnaissance-in-cyber-security/

[17] K. Jindal, S. Dalal, and K. K. Sharma, "Analyzing spoofing attacks in wireless networks," in *2014 Fourth International Conference on Advanced Computing & Communication Technologies*, 2014, pp. 398–402.

[18] K. Thompson, "The 10 most common website security attacks (and how to protect yourself)," 2024, available online, Last accessed on 2024-07-02. [Online]. Available: https://www.tripwire.com/state-of-security/most-common-website-security-attacks-and-how-to-protect-yourself

[19] G. R. T. K. D. h3lix. Andrew Smith. Jenjava1762. Mtesauro. kingthorin, "Brute force attack," 2024, available online, Last accessed on 2024-07-02. [Online]. Available: https://owasp.org/www-community/attacks/Brute_force_attack

[20] S. Raschka, "Model evaluation, model selection, and algorithm selection in machine learning," *arXiv preprint arXiv:1811.12808*, 2018.

[21] R. Moraes, J. F. Valiati, and W. P. Gavião Neto, "Document-level sentiment classification: An empirical comparison between svm and ann," *Expert Systems with Applications*, vol. 40, no. 2, pp. 621–633, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417412009153

[22] E. U. H. Qazi, A. Almorjan, and T. Zia, "A one-dimensional convolutional neural network (1d-cnn) based deep learning system for network intrusion detection," *Applied Sciences*, vol. 12, no. 16, 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/16/7986

[23] J. Zhu, H. Chen, and W. Ye, "Classification of human activities based on radar signals using 1d-cnn and lstm," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020, pp. 1–5.

[24] J. Ali, R. Khan, N. Ahmad, and I. Maqsood, "Random forests and decision trees," *International Journal of Computer Science Issues(IJCSI)*, vol. 9, pp. 272–274, 09 2012.

[25] G. H. Chen and D. Shah, 2018.

[26] V. R. Jakkula, "Tutorial on support vector machine (svm)," 2011. [Online]. Available: https://api.semanticscholar.org/CorpusID:15115403

[27] Y. Yang, J. Li, and Y. Yang, "The research of the fast svm classifier method," in *2015 12th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, 2015, pp. 121–124.

[28] P. Nakkiran, G. Kaplun, D. Kalimeris, T. Yang, B. L. Edelman, F. Zhang, and B. Barak, "Sgd on neural networks learns functions of increasing complexity," *arXiv preprint arXiv:1905.11604*, 2019.

[29] A. Zaidi and A. Al Luhayb, "Two statistical approaches to justify the use of the logistic function in binary logistic regression," *Mathematical Problems in Engineering*, vol. 2023, pp. 1–11, 04 2023.

[30] D. Shah, "Top performance metrics in machine learning: A comprehensive guide," 2023, available online, Last accessed on 2024-07-02. [Online]. Available: www.v7labs.com/blog/performance-metrics-in-machine-learning

[31] J. Muschelli, "Roc and auc with a binary predictor: a potentially misleading metric," *Journal of Classification*, vol. 37, p. 696–708, 2020. [Online]. Available: https://doi.org/10.1007/s00357-019-09345-1

[32] Z. Zhou and G. Hooker, "Unbiased measurement of feature importance in tree-based methods," *ACM Trans. Knowl. Discov. Data*, vol. 15, no. 2, jan 2021. [Online]. Available: https://doi.org/10.1145/3429445

[33] Y. Okada, S. Ata, N. Nakamura, Y. Nakahira, and I. Oka, "Comparisons of machine learning algorithms for application identification of encrypted traffic," in *2011 10th International Conference on Machine Learning and Applications and Workshops*, vol. 2, 2011, pp. 358–361.

[34] M. Conti, L. V. Mancini, R. Spolaor, and N. V. Verde, "Analyzing android encrypted network traffic to identify user actions," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 1, pp. 114–125, 2016.

[35] Hamza, A., Habibi Gharakheili, H., Benson, T. A., and Sivaraman, V., ""Detecting Volumetric Attacks on LoT Devices via SDN-Based Monitoring of MUD Activity"," *Proc. ACM SOSR, San Jose, CA, USA*, 2023.

[36] A. Hamza, H. H. Gharakheili, T. A. Benson, and V. Sivaraman, "Detecting volumetric attacks on lot devices via sdn-based monitoring of mud activity," in *Proceedings of the 2019 ACM Symposium on SDN Research*, 2019, pp. 36–48.