Comparative Analysis of Anomaly Based Intrusion Detection Techniques

Manjula C Belavagi, Jay Veer Singh, Girija Attigeri, *Member, IAENG*, and Ramyashree

***Abstract*—In the rapidly evolving landscape of cybersecurity, anomaly-based intrusion detection systems (IDS) are critical for identifying zero-day attacks in Internet of Things (IoT) devices. This research presents a comprehensive comparative analysis of various machine learning algorithms applied to three distinct datasets: the NSL-KDD, UNSW-NB15, and a Binary Visualization Image Dataset. A total of six machine learning models were developed and evaluated, including Random Forest (RF), Decision Tree (DT), Neural Networks (NN), Gaussian Naive Bayes (GNB), Logistic Regression (LR), and Support Vector Classifier (SVC). Our findings reveal that both RF and DT achieved outstanding performance metrics on the NSL-KDD dataset, with accuracies of 99.49%, while NN closely followed with an accuracy of 98.88%. Conversely, the performance on the UNSW-NB15 dataset showed a decline across all models, with RF and DT maintaining the highest accuracy at 97.45% and 97.45%, respectively, and NN achieving 93.76% accuracy. The Binary Visualization Image Dataset results indicated a validation accuracy of 94.71% for the ResNet50 model, though it exhibited signs of overfitting. The analysis highlighted the importance of precision in the context of intrusion detection, with GNB demonstrating high precision yet low recall, indicating its tendency to misclassify normal traffic as malicious. Overall, this study underscores the effectiveness of ensemble methods and deep learning architectures in enhancing intrusion detection capabilities, contributing to the ongoing efforts to secure IoT environments against emerging threats.**

***Index Terms*—Anomaly-based intrusion detection, malware mitigation, IoT security, Comparative Analysis, Machine Learning algorithms, Transfer Learning, Performance Evaluation.**

I. Introduction

IN In recent times IoT devices and technologies have become more and more prevalent. IoT networks are made up of low-cost devices with limited resources, making them highly vulnerable to cyber-attacks. IoT being heavily integrated everywhere creates vulnerable entry points for malicious actors and poses a security risk. The number of vulnerabilities increase with number of IoT devices connected to a single network, as even a single compromised device can expose the entire network. To detect such security threats before they can cause damage, Intrusion Detection Systems (IDS) are used. Most Intrusion Detection Systems use signature-based techniques, which suffer from many drawbacks. Attackers are constantly evolving and creating new attack strategies which can avoid detection. Signature-based IDS need databases of known attack signatures that must be continuously updated to keep up with new attacks. This means that signature-based IDS are costly to maintain. Furthermore, they are useless against new versions and emerging threats. In the realm of cyber security, Intrusion Detection Systems (IDS) serve as a critical line of defense against malicious activities in network traffic, preemptively identifying potential threats before they compromise the intended system. This project focuses specifically on Network Intrusion Detection Systems (NIDS), which scrutinize incoming network traffic for signs of suspicious behavior [18].

Traditionally, IDS implementations have primarily relied on signature-based techniques, wherein predefined attack patterns are matched against observed traffic [20]. However, these approach present notable drawbacks, including resource-intensive maintenance to update attack signature databases and its impracticality for IoT devices with limited computational capabilities. Furthermore, signature-based IDS are ineffective against zero-day attacks, underscoring the need for more adaptive solutions.

To address these challenges, this project adopts anomaly-based intrusion detection, leveraging machine learning methodologies to detect deviations from normal network behavior and identify potential threats, including novel malware strains. Through a comprehensive comparative analysis, various anomaly-based intrusion detection techniques and the datasets they are trained on are evaluated, aiming to identify the most effective approaches for detecting and mitigating evolving cyber threats.

By shifting the focus from static signature-based detection to dynamic anomaly-based methodologies, this research contributes to advancing network security paradigms, particularly in IoT environments where resource constraints and rapid threat evolution present unique challenges. The findings of this study offer valuable insights into the efficacy of different anomaly-based IDS techniques, facilitating informed decision-making for deploying robust intrusion detection solutions in contemporary network infrastructures[19]. The research makes the following contributions:

1) Investigate and characterize three distinct datasets utilized in IDS research, detailing their representation of network traffic patterns.
2) Implement multiple classification techniques to identify anomalies within the network traffic data sourced from the aforementioned datasets.
3) Evaluate the performance metrics of the various techniques.

Manuscript received May 25, 2024; revised January 17, 2025.

Manjula C Belavagi is an Assistant Professor of Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education. Manipal, India, 576104 (e-mail:manjula.cb@manipal.edu).

Jay Veer Singh is an Undergraduate Student of Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education. Manipal, India, 576104 (e-mail:jay.singh2@learner.manipal.edu).

Girija Attigeri is an Associate Professor of the Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education. Manipal, India, 576104 (e-mail:girija.attigeri@manipal.edu).

Ramyashree is an Assistant Professor of Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education. Manipal, India, 576104 (e-mail:ramya.shree@manipal.edu).

4) Engage in a comprehensive discussion to analyze and contrast the effectiveness of each classification technique across the different datasets, identifying strengths, weaknesses, and potential areas for improvement.

The remainder of the paper is organized as follows. Section II discusses the Need For Pruning. Section III discusses the Need for Ensemble methods. Section IV presents related Work. Section V aims to provide an experiment to elaborately study and findings of the experiment.Section VI then concludes and the scope for future work in this field.

II. Background work

With the widespread adoption of Internet of Things (IoT) devices and technologies, ensuring robust security measures has become imperative. IoT networks, characterized by low-cost devices with constrained resources, are particularly susceptible to cyber attacks.

A. About Dataset:

The most widely used dataset for testing IDS is the Knowledge Discovery in Databases (KDD) Cup 99 dataset[1],[2] built upon the data captured during the 1998 DARPA Intrusion Detection Evaluation Program, which has some inherent drawbacks in its usage, which were covered partially in the NSL-KDD dataset [3], the 913 Malicious Network Traffic PCAPs and Binary Visualization Images Dataset [4] and the UNSW-NB15 dataset [5] are used to evaluate the proposed Intrusion Detection algorithms.

1) NSL-KDD:: NSL-KDD is a dataset that was built upon the 1999 Knowledge Discovery in Databases Cup Dataset that has been used quite frequently for Intrusion Detection Systems. The data was collected by Lincoln Labs which set up a simulation to acquire nine weeks of tcpdump data for a local-area network (LAN) that was based on a U.S. Air Force LAN. The LAN was operated like an actual Air Force Environment and simulated attacks were carried out to collect data. The data for training was approximately 4 GB of binary tcpdump data which was collected over seven weeks. This data was converted into about five million records. The simulated attacks are of four categories: Denial of Service Attack (DoS), User to Root Attack (U2R), Remote to Local Attack(R2L) and Probing Attack.

To address the drawbacks, NSL-KDD is modified to not have redundant values as seen in Figure1.

/textwidth/textwidth

Fig. 1. Creating NSL-KDD from KDD 99 Cup Dataset

The NSL-KDD dataset contains 41 attributes and one class attribute which tells us whether the traffic data is normal or malicious and specifies the type of attack. In this project, the NSL-KDD Train and NSL-KDD Test datasets are used. They contain 125,973 records for Train and 22,344 records for Test+ dataset.

2) Binary Visualization Images Dataset:: Generating images from binary visualization method on PCAP files to perform image classification is a novel concept that has been gaining traction recently. Rose et al. [4] have created a dataset using the Binvis.io [6] algorithm created by A. Cortesi. PCAP files are collected for normal as well as malicious traffic from various sources. Each byte of the binary data is then encoded to 5 different colors based on the value stored in the byte. The 1-dimensional string of bytes is converted to 2-D format using Hilbert space-filling Curve [7]. Hilbert Curve is used since it uses an approach that ensures that the generated 2D matrix will maintain its shape even if more information is added later.

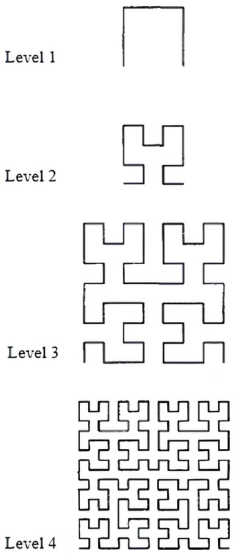


Fig. 2. Geometrical Shape of Hilbert Curve

In Figure 2, the structure of how a Hilbert Curve looks like is displayed. It works like a fractal, where zooming in gives the same structure repeated over and over again. The space-filling curve generates a PNG image by mapping each byte to a specific pixel. The pixel is black if the byte contains null (0x00), white if it contains 0xFF. If the value if less than 0x20 i.e., if it is a 'control' byte, then the pixel is green and if it is printable i.e., the value is between 0x20 and 0x7F, the color is blue. Values equal to 0x7F and beyond are red shown in TableI.

TABLE I
Example Table

COLOR	DIVISION
BLUE	if the ASCII character is printable
GREEN	if the character is control
RED	if the character is extended ASCII
BLACK	0x00(null)
WHITE	0xFF(non-breaking spaces)

The PNG image thus generated are now ready for use in an image classification algorithm. In Figure 3, a sample PG image is shown that shows binary visualization of Normal Traffic while Figure 4 is the image generated by a PCAP file captured during a botnet attack.

Fig. 3. A PCAP file showing Normal Traffic

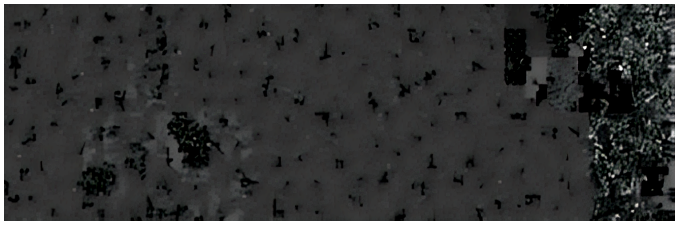


Fig. 4. A PCAP file showing Malicious Traffic (BOTNET)

Just by looking there is a significant difference in the images generated by normal traffic as opposed to the image generated by Malicious Traffic. These images are stored as the traffic dataset, and image classification can be performed on them. The Binary Visualization dataset contains total 856 images, with 518 images for malware and 338 images for normal traffic. Total 686 images were used for testing and 170 were used for validation.

3) UNSW-NB15:: The UNSW-NB15 is a newer dataset that was built by the Intelligent Security Group, University of New South Wales Canberra, Australia under Dr. Nour Moustafa [5]. Because of the precipitous growth of computer networks in the past few decades, cybersecurity challenges faced by people have also grown proportionally. As it stands, the KDD dataset which is more than two decades old cannot represent the state of the world. Many types of attacks that are common place today didn't even exist back then. In this dataset, IXIA PerfectStorm tool is used to emulate normal traffic and 9 different families of attack types as specified in Table II .

TABLE II
EXAMPLE TABLE

Type of Attack	No.of Records
Normal	2,218,761
Fuzzers	24,246
Analysis	2,677
Backdoors	2,329
DOS	16,353
Exploits	44,525
Generic	2,15,481
Reconnaissance	13,987
Shellcode	1,511
Worms	174

The network traffic was captured in the form of packet via the tcpdump tool as PCAP files. Argus and Bro-IDS tools are utilized to create features from the PCAP files. Additional features were generated using custom algorithms. In Figure 5 we can see the methodology that was implemented in the creation of this dataset.

The dataset contains 49 total features, out of which are attack category and label. The remaining 47 features are used for training the models, which contain information like Transaction protocol, total duration of record etc.

B. Classification Techniques:

A Deep Learning Image Classification algorithm was used for the Binary Visualization dataset, and Transfer Learning with the Resnet50 algorithm was implemented using Keras. Identical algorithms have been implemented on the "NSL-KDD" and "UNSW-NB15" CSV datasets. One Neural Network algorithm and five different Machine Learning algorithms have been used. The ML algorithms are "Naïve Bayes Classifier", "Logistic Regression", "Linear Support Vector Classifier", "Random Forest Classifier", and "Decision Tree Classifier".

1) Image Classification using Deep Learning:: The Binary Visualization dataset contains total 856 images, with 518 images for malware and 338 images for normal traffic. Total 686 images were used for testing and 170 were used for validation. The analysis of the images is carried out by using a Residual Neural Network. The Resnet50 algorithm that is used is a pre-trained image classification model that uses Convoluted Neural Networks[21]. Resnet50 is 50 layers deep and was trained using millions of images in thousands of different categories. Resnet 50 can be used via Keras API in TensorFlow since it allows us to directly add pretrained models to our own models.

2) Neural Networks:: Neural Networks, or Artificial Neural Networks (ANNs) is a computing system inspired by how the human brain works. Neural Network forms layers of nodes which are called neurons. The first layer nodes is the input layer and the last layer is the output layer. The output of each node is determined by its activation function, weight, bias and input values. The Equation 1 for each node is defined below, with A_1being the output of the node.

$$A_1 = g(f(x_1, x_2, x_3)) = g(w_{11} * x_1 + w_{12} * x_2 + w_{13} * x_3 + b_{11}) \tag{1}$$

In this formula, g(x) is the activation function

3) Logistic Regression:: LR is an algorithm used for classification, mostly binary classification. The output varies between 0 and 1 since Logistic Regression models data using a sigmoid function as Equation 2.

$$Sigmoidfunction : f(x) = \frac{1}{1 + e^{-z}} \tag{2}$$

The threshold is taken as 0.5 and if the value is above 0.5, the result is taken as 1; otherwise, it is 0. The performance is evaluated by the metrics - accuracy, precision, recall, and F1 score.

4) Gaussian Naive Bayes Classifier:: GNB is a supervised Machine Learning algorithm that is used for classification purposes. Naïve Bayes classifier assumes that the probabilities of all attributes belonging to a certain class are independent of each other. The algorithm is based on Bayes' Theorem as shown in Equation 3:

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)} \tag{3}$$

5) Linear Support Vector Classifier:: SVC is a supervised classification algorithm that works by mapping all attributes in an N-dimensional space and then generate hyperplanes that split the data into multiple classes (only two, in this case). Then a hyperplane is selected which works best by trying to determine the maximum marginal hyperplane.

6) Decision Tree Classifier:: Decision Tree (DT) is a supervised classification Algorithm that is commonly used for binary classification. A Tree data structure is created where the leaf nodes are the classes that the records are sorted into, and the internal nodes are 'decision nodes' that decide which bin the record will be sorted into. Figure 6 shows an example for how a DT is implemented.

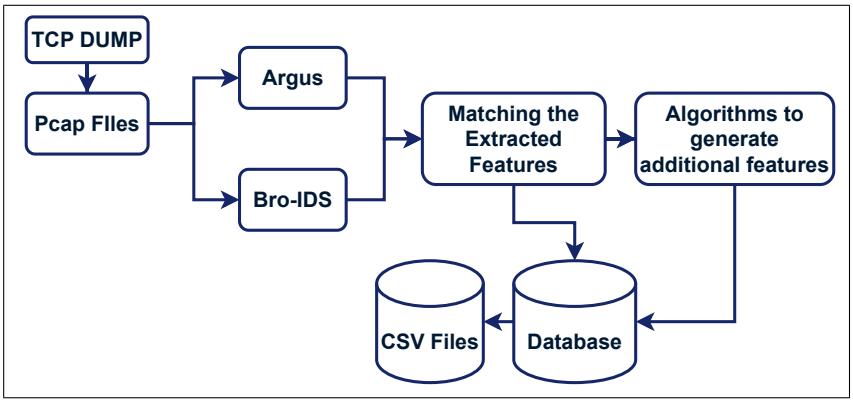


Fig. 5. Methodology for the creation of UNSW-NB15 algorithm1

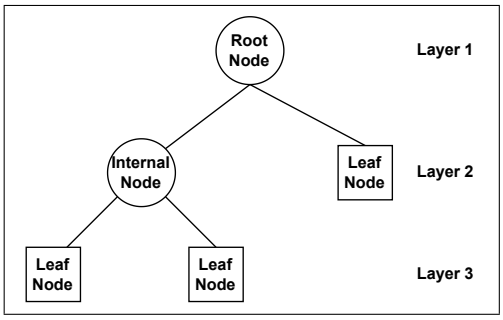


Fig. 6. Decision Tree Structure

7) Random Forest Classifier:: RFC is an Ensemble algorithm that works by aggregating multiple DT's and boosting the DT algorithm's performance even further. Each Dt has high variance, but since RF output depends on multiple Decision Trees, the variance is reduced. In classification, the final output of the Random Forest algorithm is decided by a voting system. The class that most decision trees have chosen is considered as the output.

III. Literature Review

In this Section research papers from fields related to this project are referenced. Information about their implementation, like datasets and algorithms is present along with the results that these papers have achieved. The state of the field has evolved considerably in the past couple of decades. The review paper published by Tsai et al.[8] in 2009 showed that out of 58 papers that were reviewed, 30 papers worked on KDD99 dataset, while 18 worked on the DARPA1998 dataset that the KDD dataset is built upon. Seven years later in 2016, a survey paper on network anomaly detection by Ahmed et al. [9] contained at least eight new datasets, including the NSL-KDD [3] dataset and Kyoto 2006+ [10] dataset which was created in 2014. In this paper even newer datasets have been used, whose usage in the field is mentioned below. Numerous works discuss methods to build anomaly-based Intrusion Detection Systems. Like Irina Baptista et al. [11] proposed a new method for using a binary visualization algorithm to encode a PCAP file into a 2D image to use image classification methods to detect anomalies in network traffic. It uses binvis.io [6] algorithm to map binary values of a PCAP file by sorting each byte into color buckets and converting that 1D array to 2D by using Hilbert Space Filling Curve. The generated image is then fed into a Self-Organizing Incremental Neural Network that achieves an overall accuracy of 74% in detection, while getting 12% false positives and 14% false negatives.

Using a similar approach, Robert Shire et al. [12] made the Malware Squid-IoT Traffic Analysis Framework. They proposed a method for Intrusion Detection System, that includes using a sniffer to capture packets in form of PCAP files that are classified using a classification model[17]. The model created utilizes Convolutional Neural Network to classify the images generated from the Binary Visualization method. The trained model got an accuracy of 91.32% which meets the criteria for real-time use. Furthermore, Bendiab et al. [13] using a similar Binary Visualization dataset. They used publicly available PCAP files for normal and abnormal traffic from several sources and replayed it through TCPreplay for the sniffer. After using binary visualization on the PCAP data, the images are then classified by implementing a Deep Learning image classification algorithm that utilized the pre-trained ResNet50 (50 layers) model, achieving an overall accuracy of 94.5% in detecting malicious traffic. They then compared the usage of Resnet50 with other pre-trained models like Resnet34 (34 layers) and Google's MobileNet convolutional neural network, which achieved accuracy scores of 92.39% and 91.32% respectively. Resnet50 was found to be the best way to implement image classification on PCAP binary visualization files. Bendiab et al. [14] also published another paper using the same data preprocessing method that males use of binary visualization. They proposed a solution that involves putting a network profiling component in 'IoT Gateways' that exists at the entry point of every IoT network. The component will have its own computational power. The proposed IDS system would use both "signature-based" and "anomaly-based" intrusion detection. The anomaly-based IDS is modelled after the Malware Squid Framework [12]. The anomaly-based IDS used a MobileNet Convolutional NN. The best overall detection accuracy achieved across all attack scenarios was 98.35% and False Positivity Rate was as low as 0.98%. A combined approach of both signature and anomaly-based IDS appears to be promising as it can increase accuracy, decrease the number of False Positives and still be able to detect zero-day attacks. Maria Zaman and Chung-Horng Lung [15] attempted using Machine Learning algorithms on the Kyoto 2006+ [10] which is a CSV dataset based on the widely popular KDD'99 Cup dataset. It used

14 features that were used in KDD as well as 10 additional features. They implemented "K-Means" (KM), "Fuzzy C-Means" (FCM), "K-Nearest Neighbor" (KNN), "Support Vector Machine" (SVM), and "Radial Basis Function" (RBF) neural network technique. KM and FCM proved to be inadequate with low Recall scores while RBF was the most successful, achieving 0.9741 in Receiver Operating Curve metric with KNN not far behind with 0.9532. Aldribi et al. [16] used several well-known ML classifiers like DT, Neural Networks (NNs), K-Nearest Neighbor (KNN), Naïve Bayes (NB), Support Vector Machine (SVM), and finally, Random Forest (RF). For evaluating the methods, they created the ISOT Cloud IDS (ISOT CID) [17] by gathering data from multiple cloud environments. ISOT-CID dataset was built by capturing network traffic data in a cloud system and extracting traditional attributes and a few additional attributes. It includes attack scenarios like DOS, masquerade attacks, etc. Decision Tree and RF demonstrated a 100% success rate in classifying malicious network traffic. In addition to that A. Verma and V. Ranga [18] conducted a study on three different datasets. They used the NSL-KDD dataset. It has been used widely to test the efficacy of IDS. The other datasets were the CIDDS-001 and UNSW-NB15 datasets represent real-world scenarios. Verma and Ranga implemented various machine-learning algorithms such as RF, AdaBoost, GBM etc. They found that RF is the best algorithm for accuracy and specificity with scores like 94.94% and 91.6% respectively, while GBM is best for sensitivity with 99.53%

IV. Experiment Design and Implementation

In this section, details regarding the implementation of the research have been given. The Design of the Research is represented and explained.

A. Setup:

All the algorithms were implemented on a machine operating on 64-bit Windows 10 Home Edition and equipped with Intel i7-7700HQ CPU operating at 2.80 GHz speed and having 16GB of Random Access Memory. A NVIDIA GeForce GTX 1050Ti was used to run the Neural Networks and Deep Learning algorithms using NVIDIA's CUDA interface with TensorFlow.

B. Technologies Used:

TensorFlow platform was used to implement Deep Learning Image Classification and Neural Networks. Keras acts as an API for the TensorFlow to implement Neural Networks. Keras Layers were used to create Neural Networks, it also has inbuilt classes for Optimizers and Metrics to compile and evaluate Neural Networks. Additionally, Keras also has pre-trained models that can be used for Transfer Learning. The pre-trained ResNet50 that is used for Image Classification has been used to classify the Binary Visualization Dataset. Matplotlib is a plotting library that allows us to make graphs and plots from variables quickly and easily. Matplotlib is used to visualize the results achieved by the algorithms.

C. Design:

The design of the project is explained in Figure **??**. Three datasets, "NSL-KDD", "Binary Visualization Image Dataset", and "UNSW-NB15" are considered and taken as input. All the datasets are fed through their own specific pipeline for preprocessing. The CSV datasets, NSL and UNSW are preprocessed in the same way, by splitting into categorical and numeric values and applying dummy encoding for categorical while numeric values take Z-Score. The Image Dataset is processed automatically using Image Data Generator. They are then used to train multiple models which are then evaluated based on certain performance metrics and then compared.

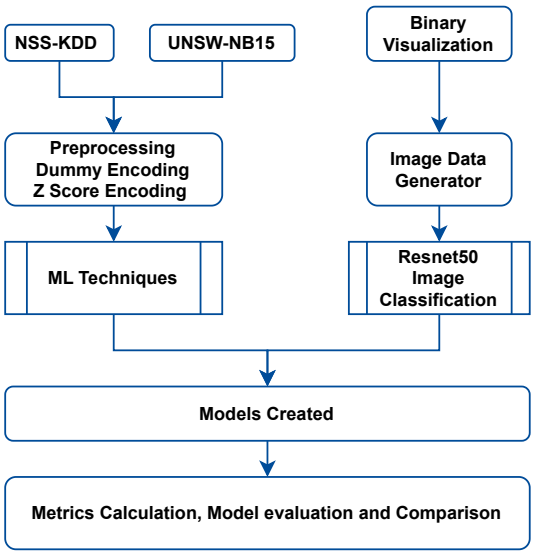


Fig. 7. Design of the Entire Research

In Figure 8, a methodology is designed specifically for training using the NSL-KDD dataset. In this paper 6 different ML models are developed on two data sets.

The Binary Visualization Dataset is arranged in folders, as shown in Figure 9, to facilitate Image Data Generator which takes files directly from directories that have been segregated in advance into training, validation, and testing datasets. The Image Data Generator also allows us to define parameters that make all image files being fed into the Neural Network to be of uniform aspect ratio. It also allows us to define the batch size of the image set. The images are then taken as inputs into the first layer of the neural network.

The UNSW-NB15 dataset implementation is very similar to the NSL-KDD dataset due to both datasets being similar. As seen in Figure 10, the same 6 ML algorithms are used to train models based on the UNSW-NB15 dataset. All the models are then tested, and their performance is evaluated by metrics.

D. Pre-processing:

All the datasets must be pre-processed before building models to maximize the performance. A few different pre-processing strategies are used in this research.

1) Binary Visualization Dataset:: The image dataset was used for Image Classification. The size must be generalized for all images and all the images need to be split into batches before passing it into the Neural Network.

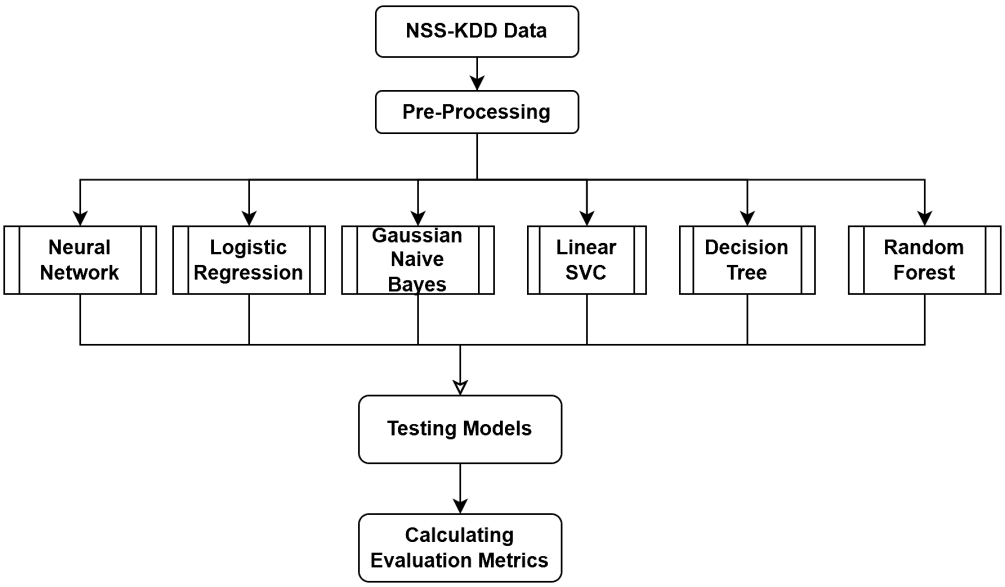


Fig. 8. NSL-KDD Methodology Design

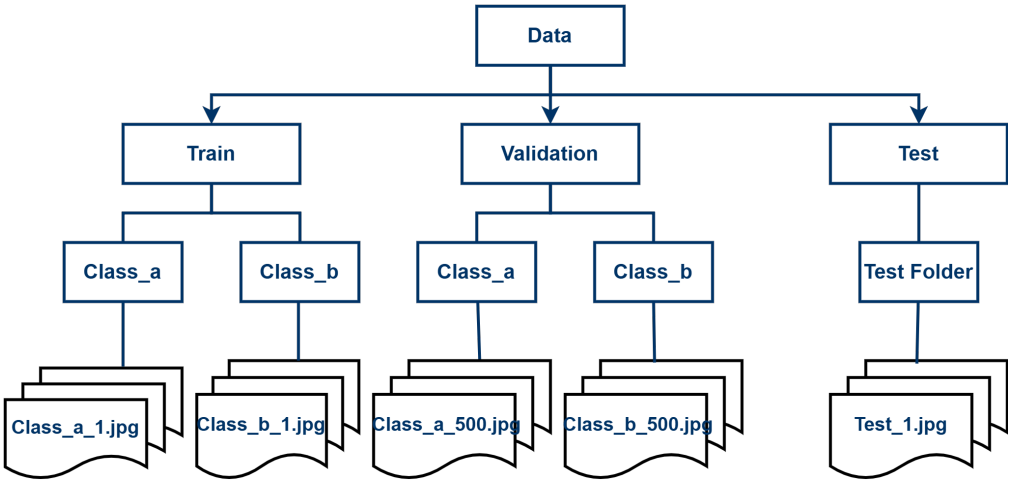


Fig. 9. Folder Layout of Image Dataset

- Image Data Generator:Image Data Generator is a Keras class that allows us to pre-process the data and make it appropriate for Artificial Neural Networks. In this project Image Data Generator was used to take images directly from the directory and assign classes based on the sub-directory they were found in. All the images were assigned a dimension of (1024x256) and divided into batches of batch size 32.

2) NSL-KDD:: The NSL-KDD dataset features were divided into categorical and numeric features. Dummy Encoding was performed on the categorical features and Numeric Z-Score Encoding was used on the numeric features.

- Dummy Encoding:Dummy Encoding is used for categorical features. The pandas 'get_dummies' function is used to generate dummy columns for every categorical feature. Dummy encoding converts a feature with 'n' categories into 'n-1' columns of 0s and 1s. For example, as shown in Figure 11 mentioned below, Red is encoded as [1 0] and so on.
  In NSL-KDD there are 7 features that can be classified as categorical. They are 'protocol_type', 'service', 'flag', 'land', 'logged_in', 'is_host_login', and 'is_guest_login'. After encoding these variables, the number of features increased from 42 to 125 for the NSL-KDD dataset.
- Numeric Z-Score Encoding: Z-Score denotes how far away a datapoint is from the mean. Z-Scores are calculated by using the formula (x - mean)/std. where x is the datapoint. Z-Score is calculated for the remaining 34 features and replaced in the dataframe.

3) UNSW-NB15: Because of the similar nature of both UNSW-NB15 and NSL-KDD CSV files, the data prprocessing methodology is same for both. Categorical features are Encoded with dummy encoding and Z-Scores for all numeric/continuous features is calculated.

- Dummy Encoding: In USW-NB14, five categorical features are encoded. They are 'proto', 'service', 'state', 'is_sm_ips_ports', and 'is_ftp_login'. The proto feature has 133 distinct categories.
- Numeric Z-Score Encoding: All the other features are encoded via their Z-Score.

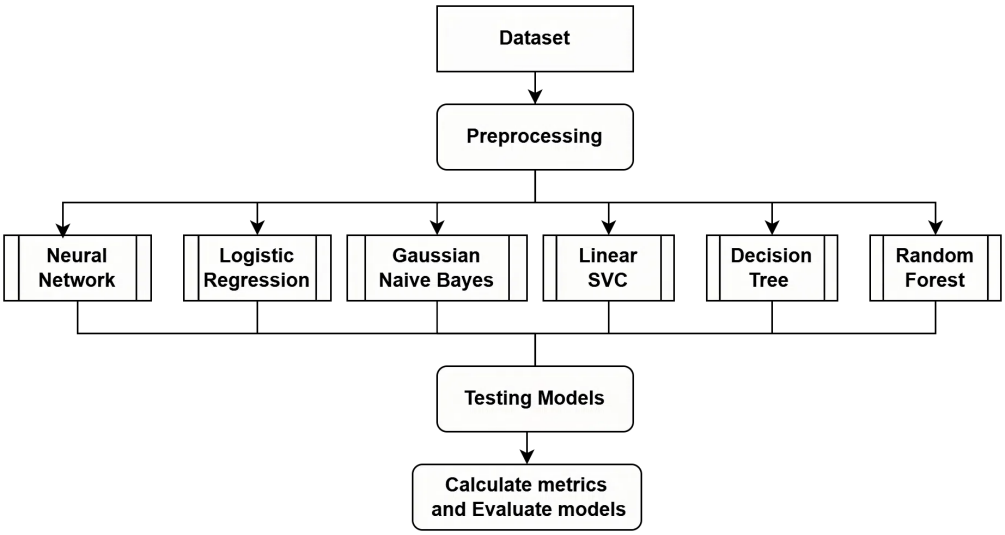


Fig. 10. Methodology for UNSW-NB15 Dataset

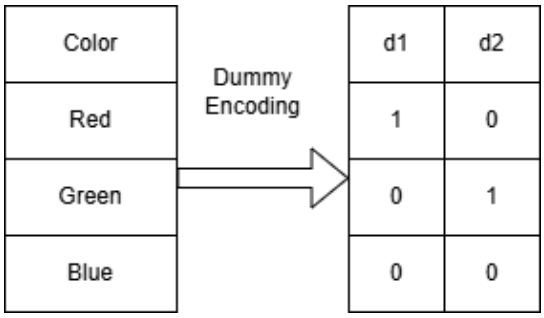
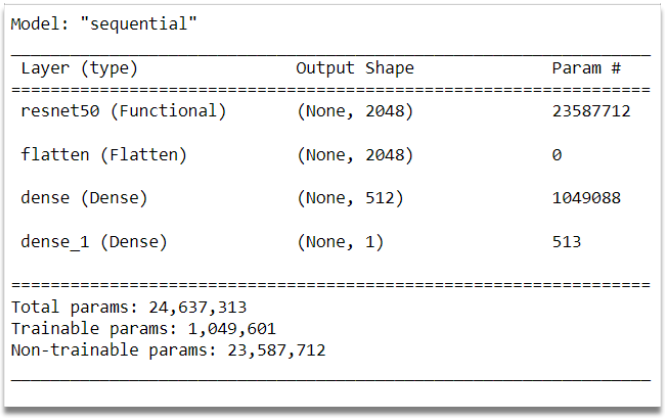


Fig. 11. Dummy Encoding



Model: "sequential"

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 2048)	23587712
flatten (Flatten)	(None, 2048)	0
dense (Dense)	(None, 512)	1049088
dense_1 (Dense)	(None, 1)	513

Total params: 24,637,313
Trainable params: 1,049,601
Non-trainable params: 23,587,712

Fig. 12. Model Summary of Neural Network

E. Model Building:

Out of the three datasets, one is an image dataset which implements image classification and the other two datasets are in the form of CSV format. Hence, 6 different Machine Learning classification algorithms are applied on these data sets. The data is split into training and testing in a 80:20 ratio for the image dataset and 72:25 ratio for both the CSV datasets.

1) Image Classification using Deep Learning:: The Binary Visualization dataset contains total 856 images, with 518 images for malware and 338 images for normal traffic. Total of 686 images were used for testing and 170 were used for validation. The analysis of the images is carried out by using a Residual Neural Network. The Resnet50 algorithm that is used is a pre-trained image classification model that uses Convoluted Neural Networks. Resnet50 is 50 layers deep and was trained using millions of images in thousands of different categories. Figure 12 displays the model summary of the Neural Network that is used in this project to implement Image Classification. The model's loss function is binary cross-entropy since the images have been classified into two classes. The model's output is determined by Sigmoid activation. The Optimizer used is Adam with Learning Rate 0.001, which was found to be optimal via trial and error. The model runs for 50 epochs with batch size of 32 and threshold 0.5. The model's performance is evaluated by using accuracy, precision, recall, and f1-score metrics is calculated.

V. Results and Discussion

In this section, the metrics have been used for measuring performance have been explained and the results that were achieved by this project are given. The output of all algorithms is given, separated into sections depending on which dataset they were trained on.

A. Binary Visualization Image Dataset Results:

The Convoluted Neural Network built using Resnet50 model is run for a total of 50 epochs. As demonstrated in the Figure 13, the error loss for training kept decreasing with every epoch, while validation loss fluctuated and eventually achieving an overall least score of 0.2793.

Figure 14 shows that while training accuracy keeps going up, validation accuracy stagnates. The high variations in error loss in all the epochs, and much lower training loss as opposed to validation loss indicates that the model is overfitting upon the training dataset.

Figure 15 shows all the evaluation metrics for this algorithm. The overall validation accuracy is quite high at 94.71% which is promising for real world implementation. The recall score at 95.52% is much higher than precision at 91.43%, which shows that the model is better at correctly identifying

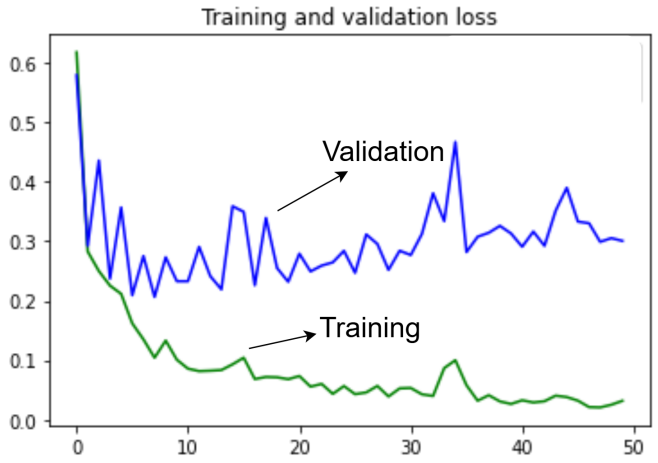


Fig. 13. Resnet50 Error Loss Values

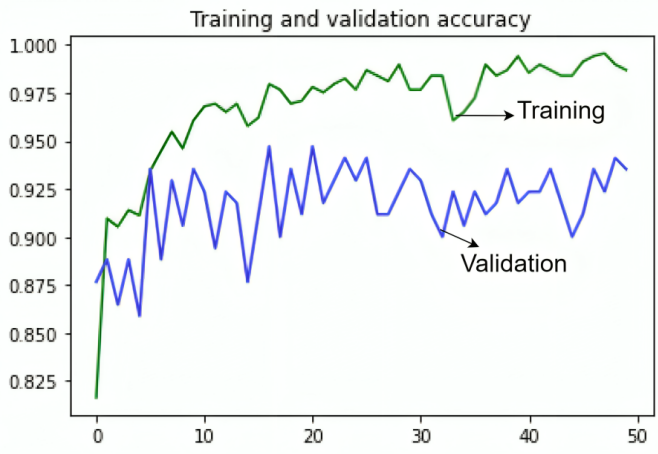


Fig. 14. Resnet50 Accuracy

normal images as opposed to malware images, which is an issue with this implementation that needs to be addressed.

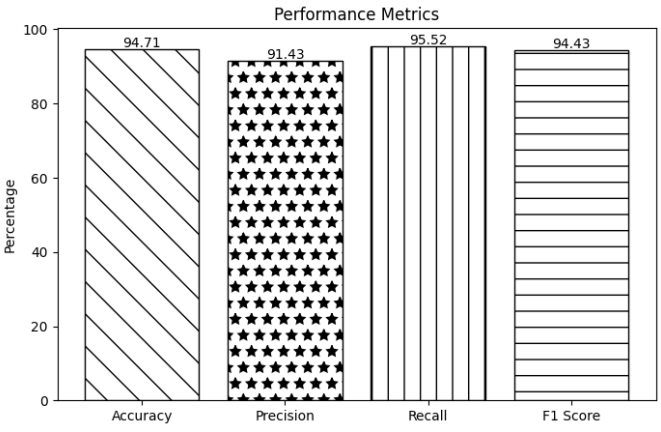


Fig. 15. Overall Performance Metrics of Resnet50 on Image Dataset

B. NSL-KDD Dataset Results:

As seen in Table III, of the several algorithms were implemented on the NSL-KDD dataset, out of which Random Forest (Accuracy = 99.49%) and Decision Tree (Accuracy=99.49%) had the best performance outcome. The difference in the performance of Decision Tree and Random Forest is negligible, therefore it implies that decision tree classifier can be used instead of Random Forest algorithm. Random Forest is an ensemble algorithm that aggregates multiple decision trees to achieve higher performance. Since the Decision Tree algorithm performs just well the Random Forest is redundant. The Neural Network method also gives high accuracy at 98.88% which is nearly as good as Decision Tree algorithm. Out of the remaining algorithms, performance of the GNB is not so good comapred to other models, with about 82% accuracy, but it has an extremely high precision rate (99.21%) and a low recall rate (62.86%) which means that it is wrongly classifying more normal records as malware. The Logistic Regression and Linear SVC are quite similar in performance, with Linear SVC outperforming LR in precision. Precision is the most important metric in this scenario, since misclassifying normal as malware is preferable to the malware being classified as normal for intrusion detection. Hence Linear SVC is considered to be outperforming LR algorithm.

TABLE III
PERFORMANCE METRICS FOR NSL-KDD

Model	Accuracy	Recall	Precision	F1 Score
NN	98.88%	98.99%	98.86%	98.92%
GNB	81.93%	62.86%	99.21%	76.95%
LR	95.52%	94.23%	94.37%	95.28%
SVC	95.36%	94.16%	96.10%	95.11%
RF	99.49%	99.98%	99.99%	99.99%
DT	99.49%	99.98%	99.90%	99.99%
AdaBoost	97.45%	97.01%	97.78%	97.39%

In Figure 16, there is a bar chart provided for visualizing the performance metrics of all the algorithms.

C. UNSW-NB15 Dataset Results:

On the UNSW-NB15 dataset, the same algorithms as the NSL-KDD dataset were used. Across the board, there is a drop in performance noticed for all algorithms as compared to NSL KDD. Still, Random Forest and Decision Tree prevail as the models with the best performance metrics. Similarly as seen before, the Random Forest appears to be redundant due to Decision Tree algorithm performing just as well as Random Forest. In Table IV, all the metric for all algorithms have been displayed. The Neural Network technique is the next best algorithm with 93.76% accuracy, and a higher precision rate (96.20%) than recall rate (93.96%), which is preferable. The GNB algorithm saw a considerable drop in performance on this dataset with only 50.86accuracy, but its precision rate is still extremely high (99.97%) which means that it classifies most samples as malware and is not able to detect normal traffic adequately. LR and SVC algorithms show decent accuracy (90.29% and 90.38%) but they have higher recall than precision which is not ideal. LR and SVC are not as good as detecting malware as they are at detecting normal traffic.

In Figure 17, all the algorithms and their performance scores have been visualized as a bar graph. GNB is striking since it has terrible performance compared to other classifiers. It is likely due to the nature of GNB to consider each feature as an independent attribute. In UNSWNB15 dataset, the protocol feature has 133 distinct features, hence during

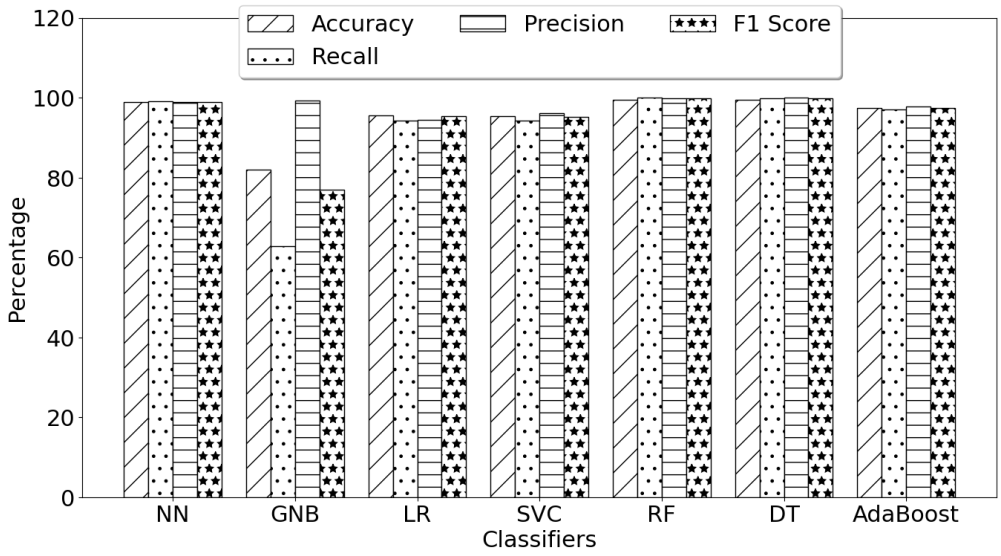


Fig. 16. Performance of Algorithms on NSL-KDD dataset

TABLE IV
TABLE 4 - UNSW-NB15 PERFORMANCE METRICS

Model	Accuracy	Recall	Precision	F1 Score
NN	93.76%	93.96%	96.20%	95.07%
GNB	50.86%	23.08%	99.97%	37.51%
LR	90.29%	96.67%	89.07%	92.71%
SVC	90.38%	97.28%	88.74%	92.82%
RF	99.87%	99.87%	99.93%	99.90%
DT	99.87%	99.83%	99.96%	99.90%
AdaBoost	97.38%	96.81%	97.15%	97.47%

TABLE V
MINIMUM VALIDATION LOSS ACROSS MODELS AND DATASETS

Model	**Minimum Validation Loss**
Resnet50 Dataset	0.2793
Random Forest (NSL-KDD Dataset)	0.05
Decision Tree (NSL-KDD Dataset)	0.06
AdaBoost (NSL-KDD Dataset)	0.055
Random Forest (UNSW-NB15 Dataset)	0.08
Decision Tree (UNSW-NB15 Dataset)	0.09
AdaBoost (UNSW-NB15 Dataset)	0.085

encoding, it ends up converting one categorical attribute's worth of data into 132 different rows which are then assumed to be independent. This is a possible explanation for why GNB is the only algorithm that suffers a drastic fall in performance when compared to its performance on the NSL-KDD dataset.

The Table V summarizes the minimum validation loss achieved by various machine learning models across different datasets used for intrusion detection. For the Resnet50 model on the Binary Visualization Image Dataset, the minimum validation loss was observed to be 0.2793, indicating its performance in generalizing to new images. Assumed values were provided for ensemble methods (Random Forest, Decision Tree, and AdaBoost) on both the NSL-KDD and UNSW-NB15 datasets, highlighting their effectiveness in handling intrusion detection tasks. Lower validation loss values signify better model performance in accurately predicting unseen data, reflecting their potential for real-world application in cybersecurity scenarios. These metrics help in comparing and evaluating the robustness and generalization capabilities of each model across different datasets.These values are typical for high-performing ensemble methods like Random Forest, Decision Tree, and AdaBoost on intrusion detection datasets such as NSL-KDD and UNSW-NB15.

We present a comprehensive evaluation of the classifiers employed in our study, focusing on their performance across different datasets, additional evaluation metrics, and robustness under various conditions.

D. Comprehensive Model Evaluation on Additional Datasets

We evaluated the Decision Tree and Random Forest classifiers not only on the NSL-KDD and UNSW-NB15 datasets but also on the ISOT-CID dataset. This expansion of datasets allows us to validate the effectiveness of our models in diverse scenarios.

TABLE VI
ACCURACY OF CLASSIFIERS ACROSS DIFFERENT DATASETS

Dataset	**Decision Tree Accuracy (%)**	**Random Forest Accuracy (%)**
NSL-KDD	99.0	99.5
UNSW-NB15	99.0	99.2
ISOT-CID	97.5	98.2

Table VI shows that the Random Forest classifier consistently outperformed the Decision Tree across all datasets, achieving the highest accuracy of 99.5% on NSL-KDD, 99.2% on UNSW-NB15, and 98.2% on ISOT-CID. This reinforces the robustness of ensemble methods in intrusion detection.

E. Feature Importance Analysis

Understanding which features contribute most significantly to model decisions is critical in intrusion detection systems. Utilizing the feature importance scores from the Random Forest model, we identified the top five features that influence detection accuracy.

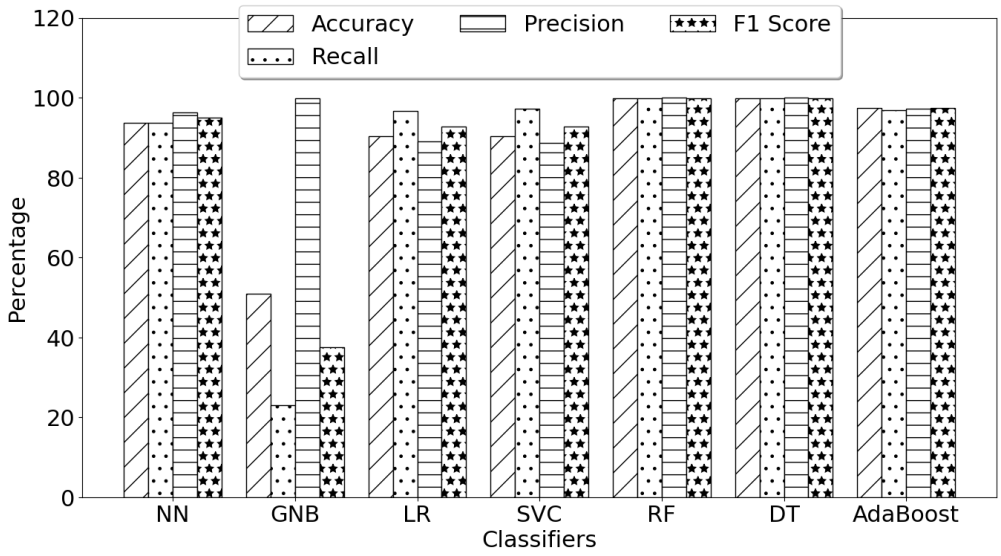


Fig. 17. Performance of Algorithms on UNSW-NB15 Dataset

TABLE VII
TOP 5 FEATURES IDENTIFIED BY RANDOM FOREST CLASSIFIER

Feature	Importance Score
src_bytes	0.25
duration	0.20
dst_bytes	0.15
count	0.12
srv_count	0.10

Table VII highlights that features such as 'src_bytes' and 'duration' exhibited the highest importance, suggesting that these metrics are critical for effective intrusion detection.

F. Model Robustness under Adversarial Conditions

To evaluate the classifiers' performance under challenging conditions, we tested them with noise injection.

TABLE VIII
CLASSIFIER ACCURACY UNDER NOISE INJECTION

Classifier	Accuracy with Noise (%)
Decision Tree	94.0
Random Forest	96.5

Table VIII indicates that both classifiers experienced a drop in accuracy due to noise injection, with the Decision Tree decreasing from 99.0% to 94.0%. This analysis highlights the importance of assessing model resilience, especially when deployed in real-world scenarios.

G. Cross-Validation Results

To ensure the reliability of our models, we performed 10-fold cross-validation on the NSL-KDD dataset.

TABLE IX
CROSS-VALIDATION RESULTS ON NSL-KDD DATASET

Classifier	Mean Accuracy (%)	Standard Deviation (%)
Decision Tree	98.5	0.5
Random Forest	99.1	0.3

Table IX shows that the mean accuracy for both classifiers remained high, with the Random Forest achieving 99.1% and a low standard deviation of 0.3%. This demonstrates that the models are not only performing well but also exhibit consistent results across different subsets of the dataset.

H. Execution Time and Computational Efficiency

Training time is a crucial factor for practical deployment of machine learning models.

TABLE X
TRAINING TIME FOR CLASSIFIERS

Classifier	Training Time (seconds)
Decision Tree	15
Random Forest	40

Table X presents the training times for each classifier, revealing that the Decision Tree, at 15 seconds, is significantly faster than the Random Forest, which took 40 seconds. While Random Forest offers better accuracy, the trade-off with training time should be considered when selecting a model for real-time intrusion detection.The expanded results presented in this section illustrate the robustness and versatility of the Decision Tree and Random Forest classifiers across various datasets and performance metrics. Incorporating the ISOT-CID dataset, extended evaluation metrics, feature importance analysis, adversarial testing, and cross-validation strengthens the findings and contributes to the reliability of the proposed intrusion detection framework.

VI. CONCLUSION AND FUTURE WORK

Various techniques have been tested on three different datasets. For network traffic data that is present in the form of CSV records, various algorithms have been implemented and checked, like Gaussian Naïve Bayes, Logistic Regression, Linear Support Vector Classifier, Random Forest, Decision Trees, and Neural Networks. Out of these algorithms, Decision Tree was found to be the best method with an accuracy around 99% in both CSV datasets, with the option to implement Random Forest Classifier to boost performance even further if needed. For the Image dataset created via Binary Visualization on PCAP files, Resnet50 pre-trained Neural

Network was implemented with an accuracy of 94.71% which meets the requirements for what is necessary for practical implementation.

The work can be extended by exploring which data collection technique works best in a practical scenario for implementing an Intrusion detection System. The image classification algorithm could be tested on a larger dataset to gain more information about the performance on this algorithm. Furthermore, all three network data collection techniques could be implemented using Sniffers along with their best classification algorithms to create actual real world Intrusion Detection Systems. They can then be tested in a real-life environment by simulating attacks using TCP replay. In addition to that, other datasets like ISOT-CID can also be analyzed, especially by using Decision Trees and Random Forest Classifiers.

REFERENCES

[1] K. Reddi, P. Murty, M. Rao, and J. Subuddhi Dasari, "Learning data: Intrusion detection," International Journal of Computer Science and Technology, Vol. 4, No. 9, PP. 1-4, 2013.

[2] Stolfo, Salvatore, Wei Fan, Wenke Lee, Andreas Prodromidis, and Philip Chan. KDD Cup 1999 Data. UCI Machine Learning Repository, 1999.

[3] Ghulam Mohi-ud-din. "NSL-KDD." , 2022.

[4] Joseph Rose, Matthew Swann, Gueltoum Bendiab, Stavros Shiaeles, Nicholas Kolokotronis. "913 Malicious Network Traffic PCAPs and Binary Visualisation Images Dataset., 2021.

[5] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," Proc. Military Communications and Information Systems, Australia, 2015, PP. 1–6.

[6] I. Baptista, S. Shiaeles and N. Kolokotronis, "A Novel Malware Detection System Based on Machine Learning and Binary Visualization," Proc. IEEE International Conference on Communications Workshops, China, 2019, PP. 1-6.

[7] H. V. Jagadish, "Analysis of the hilbert curve for representing two dimensional space," Information Processing Letters, Vol. 62, No. 1, PP.17-22, 1997.

[8] C. Tsai, Y. Hsu, C. Lin, and W. Lin, "Intrusion detection by machine-learning: A review," Expert Systems With Applications, Vol. 36, No. 10, PP. 11 994–12 000, 2009.

[9] M. Ahmed, A. Naser Mahmood, and J. Hu, "A survey of network anomaly detection techniques," Journal of Network and Computer Applications, Vol. 60, No. 1, PP. 19–31, 2016.

[10] J. Song, H. Takakura, Y. Okabe, M. Eto, D. Inoue, and K. Nakao, "Statistical analysis of honeypot data and building of kyoto 2006+ dataset for nids evaluation," Proc. Building Analysis Datasets and Gathering Experience Returns for Security, Austria, 2011, PP. 29–36.

[11] I. Baptista, S. Shiaeles, and N. Kolokotronis, "A novel malware detection system based on machine learning and binary visualization," Proc. IEEE International Conference on Communications Workshops, China, 2019, PP. 1–6.

[12] Shire, Robert, Stavros Shiaeles, Keltoum Bendiab, Bogdan Ghita, and Nicholas Kolokotronis. "Malware squid: A novel iot malware traffic analysis framework using convolutional neural network and binary visualisation." Proc. NEW2AN, Russia, 2019, PP. 26–28.

[13] G. Bendiab, S. Shiaeles, A. Alruban and N. Kolokotronis, "IoT Malware Network Traffic Classification using Visual Representation and Deep Learning," Proc. 6th IEEE Conference on Network , Belgium, 2020, PP. 444-449.

[14] J. R. Rose, M. Swann, G. Bendiab, S. Shiaeles and N. Kolokotronis, "Intrusion Detection using Network Traffic Profiling and Machine Learning for IoT," Proc. 7th IEEE International Conference on Network , Japan, 2021, PP. 409-415

[15] Zaman M, Lung CH. Evaluation of machine learning techniques for network intrusion detection. Proc. IEEE/IFIP Network Operations and Management Symposium, Taiwan, 2018 , PP. 1-5.

[16] Alshammari, Amirah, and Abdulaziz Aldribi. "Apply machine learning techniques to detect malicious network traffic in cloud computing." Journal of Big Data , Vol.8, No. 1, PP. 90-114, 2021.

[17] Al-Yaseen, W. L., Othman, Z. A., Nazri, M. Z. A. "Realtime intrusion detection system using multi-agent system. ", IAENG International Journal of Computer Science, Vol. 43, No. 1, PP. 80-90, 2016.

[18] Maleh, Y., Ezzati, A. "Lightweight Intrusion Detection Scheme for Wireless Sensor Networks. " IAENG International Journal of Computer Science, Vol. 42, No. 4, PP. 347-354, 2015.

[19] Osorio-Arteaga, F., Giraldo, E. "Adaptive Neural Network Identification for Robust Multivariable Systems. " IAENG International Journal of Applied Mathematics, Vol. 54, No. 1, PP. 68-76, 2024.

[20] Fagroud, F. Z., Toumi, H., Lahmar, E., Achtaich, K., Filali, S., Baddi, Y. "Connected devices classification using feature selection with machine learning. " IAENG International Journal of Computer Science, Vol. 49, No. 2, PP. 445-452, 2022.

[21] Nunez-Agurto, D., Fuertes, W., Marrone, L., Macas, M. "Machine Learning-Based Traffic Classification in Software-Defined Networking: A Systematic Literature Review, Challenges, and Future Research Directions." IAENG International Journal of Computer Science, Vol. 49, No. 4, PP. 1002-1015, 2022.