Comparative study on Deep Learning and Machine Learning Models in Network Intrusion Detection

Mr. Emmanuel William Frimpong
School of Cyber Security and Digital Forensics
National Forensic Sciences University,
Gandhinagar, Gujarat, India.
emmanuelwilliamfrimpong@gmail.com

Mr. Khalid Nur Ali
School of Cyber Security and Digital Forensics
National Forensic Sciences University,
Gandhinagar, Gujarat, India.
khdstudy@gmail.com

Mr. Wayne Daniel Amewu
School of Cyber Security and Digital Forensics
National Forensic Sciences University,
Gandhinagar, Gujarat, India.
hoakpedw@gmail.com

Mr. Pruthvirajsinh Parmar
Centre of Excellence in Cyber Security,
National Forensic Sciences University,
Gandhinagar, Gujarat, India.
to.prpmr@gmail.com

***Abstract* - Network intrusion detection is critical for protecting computer networks from unauthorized access and malicious activities. The introduction of machine learning enhanced detection and eased the lives of network administrators. Deep learning is a new field, and this study assesses its potential in Intrusion Detection. Thus, in this study, we conducted an in-depth comparative analysis of various machine learning and deep learning algorithms. The study's results highlighted significant variations in performance across the different algorithms and how promising Deep learning is in the field of Intrusion Detection.**

Keywords – IDS, Intrusion Detection System, Machine Learning, Deep Learning, algorithm; model; training

I. INTRODUCTION

The growth of computing, interconnectivity, and interoperability have significantly impacted our daily activities. While these advancements have improved our lives, they have also raised security concerns. The widespread sharing of digital information across networks has created vulnerabilities that could harm individuals and businesses. Therefore, it is crucial to prioritize security regarding confidentiality, integrity, and availability across all aspects.

In recent years, cyber-attacks have increased targeting various sectors such as banking, healthcare, and the Internet of Things (IoT). Organizations victimized by these attacks incur substantial costs in resolving them. Cybersecurity solutions like Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) emerged to address these challenges.

A. EVOLUTION OF IDS

A Network Intrusion Detection System (NIDS) is a hardware or software application that monitors network traffic for malicious activity [1]. The first intrusion detection system was proposed in 1980 [2]. The early history of IDS can be traced back to an academic paper written by Dorothy E. Denning titled, "An Intrusion-Detection Model". This paper led to the development of the Intrusion Detection Expert System (IDES) by the Stanford Research Institute (SRI). IDS has had an evolution through a series of detection techniques. These include signature-based IDS, and Anomaly-based IDS [3][4].

- Signature-Based Detection
  Signature-based detection methods are the traditional IDS developed. These systems identified known attack patterns based on predefined signatures [5]. While it was effective against known attacks, signature-based IDS struggled with zero-day attacks. Also, the signature libraries required constant updates to stay effective.

- Anomaly-Based IDS
  To address the limitations of signature-based IDS, anomaly-based detection came into play. These systems use Machine Learning algorithms to learn behavioral patterns from network data. These patterns help to differentiate between abnormal behaviors and normal behaviors. Based on supervised and unsupervised learning methods, Machine Learning has gained popularity in network monitoring due to the increasing volume of data generated by network devices and terminals [6].

B. MACHINE LEARNING

Machine learning is a subset of artificial intelligence focused on developing algorithms that allow computers to learn from and make predictions based on data. It encompasses various techniques, including supervised and unsupervised learning. It involves using algorithms to analyze data, recognize patterns, and make decisions or predictions. Applications range from speech recognition and image classification to recommendation systems and autonomous vehicles.

The two major categories of anomaly detection techniques are supervised and unsupervised learning techniques.

- Supervised anomaly detection
  In this technique, a model is trained on a labeled dataset. Here, the model learns based on the trends labeled in the dataset. This type of technique is used for classification and regression.

- Unsupervised anomaly detection
  This approach is based on statistical methods, clustering, outlier detection [7] schemes, state machines, etc. The dataset used in this approach is not labeled; thus, the model must learn by itself to detect which data points belong to a cluster.

C. DEEP LEARNING

Deep learning is a branch of machine learning based on artificial neural network architecture that mimics the intricate neural networks of the human brain, enabling computers to discover patterns autonomously and make decisions from vast amounts of unstructured data. There is an input layer and one or more hidden layers connected one after the other. Each neuron receives input from the previous layer neurons or the input layer. The output of one neuron becomes the input to other neurons in the next layer of the network. In addition to supervised and unsupervised machine learning, deep learning can be used for reinforcement machine learning. This is the machine learning technique in which an agent learns to make decisions in an environment to maximize a reward signal. Deep learning models can extract better representations from the data to create better models [8]. This study aimed to investigate the effectiveness of machine learning algorithms compared to deep learning algorithms in detecting network intrusions.

II. METHODOLOGY FOR THE STUDY

Fig. 1 below illustrates how the models were trained.

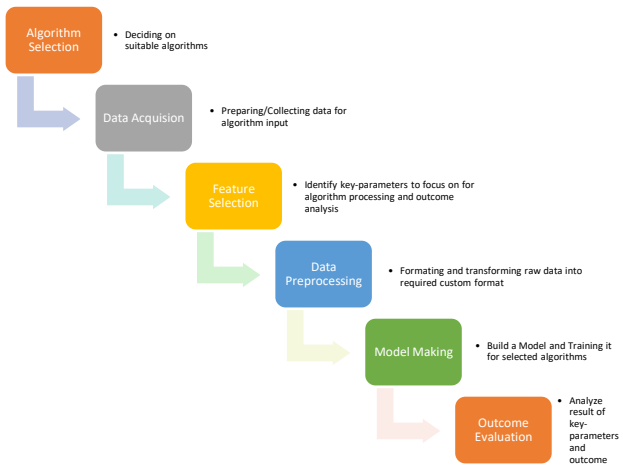

Fig. 1. Flow chart of methodology

III. ALGORITHM SELECTION

In the study, six algorithms were selected. These algorithms are;

A. Convolutional Neural Network
CNN is a type of Deep Learning neural network architecture that enables a computer to understand and interpret images and visual data [9]. And provides support for feature extraction, introduction of non-linearity into the network, dimension reduction and global information integration.

B. Recurrent Neural Network
Specifically, the Long Short-Term Neural Network was selected. This architecture is used for predicting the sequence of words [9].

C. Vanilla Artificial Intelligence model
A vanilla neural network consists of layers of neurons where each neuron is connected to every neuron in the preceding and succeeding layers. It provides a baseline for comparison and offers flexibility in model architecture.

D. Random Forest
Random Forest algorithm is a powerful tree-learning technique in Machine learning. It works by creating several Decision Trees during the training phase. Each tree is constructed using a random subset of the data set to measure a random subset of features in each partition [10].

E. Decision Tree
This algorithm is a flowchart-like structure used to make decisions or predictions. It consists of nodes representing decisions or tests on attributes, branches representing the outcome of these decisions, and leaf nodes representing outcomes or predictions [11].

F. Naïve Bayes model
Naïve Bayes algorithm is a supervised learning algorithm, which is based on the Bayes theorem and used for solving classification problems [12].

IV. TOOLS AND PLATFORMS

The study employed various tools and libraries to facilitate data analysis and model development. The Python programming language was used in Jupyterlab in the Anaconda Navigator environment. Essential libraries such as NumPy and Pandas were utilized for efficient data manipulation and preprocessing tasks.

The study relied on the Sci-kit Learn library for machine learning tasks. Additionally, TensorFlow, a powerful deep learning framework, was incorporated into the analysis to explore the potential of neural network architectures.

For visualization and interpretation of results, Matplotlib and Seaborn libraries were used. Microsoft Excel was also used for specific visualization tasks.

Together, these tools and libraries formed the foundation of the computational framework, enabling a comprehensive analysis to derive meaningful insights from the data.

V. DATA ACQUISITION

The dataset utilized in the research was the Network Intrusion Detection dataset, 838kb in size, which consists of labeled and unlabeled network traffic data. This dataset consists of a wide variety of intrusions simulated in a military network environment. The LAN was focused on a real environment and blasted with multiple attacks [13].

VI. FEATURE SELECTION

The dataset has 41 parameters. All of these parameters were used in the training of the model. However, the features that have the most weight on the outcome are discussed in the table below.

TABLE 1. TABLE OF FEATURES HAVING MOST WEIGHT ON OUTCOME IDENTIFIED FROM DATASET

Feature Name	Description
count-dest	Number of flows to unique destination IP addresses inside the network in the last T seconds from the same source
count-src	Number of flows from unique source IP addresses inside the network in the last T seconds to the same destination
count-serv-src	Number of flows from the source IP to the same destination port in the last T seconds
count-serv-dest	Number of flows to the destination IP address using the same source port in the last T seconds
count-dest-conn	Number of flows to unique destination IP addresses inside the network in the last N flows from the same source
count-src-conn	Number of flows from unique source IP addresses inside the network in the last N flows to the same destination
count-serv-src-conn	Number of flows from the source IP to the same destination port in the last N flows
count-serv-dest-conn	Number of flows to the destination IP address using the same source port in the last N flows

VII. PREPROCESSING STEPS

Data preprocessing is an important step in data science. This step helps to remove outliers, fill up missing data points, normalize the dataset, convert strings to numbers, and many more. The flowing paragraphs are the preprocessing steps used in preparing the dataset in this research.

A. Encoding
Before conducting any analysis or modeling, preprocessing steps were applied to ensure the dataset's quality and compatibility with the intended analytical methodologies. One crucial preprocessing technique employed was data encoding, especially for categorical variables. Categorical variables often require encoding to convert them into a numerical format that machine learning algorithms can effectively process.
In this study, the following categorical columns underwent label encoding:

- protocol type
- service
- flag
- label

Label encoding was chosen due to its simplicity and efficiency in converting categorical variables into numerical representations. Each unique category within the categorical columns was assigned a numerical label, facilitating subsequent analysis.

B. Outlier Detection

Another crucial aspect of data preprocessing is outlier detection. This step is used to avoid the skewness of data. The technique used was the Z-score outlier detection. This technique enabled the identification of data points that deviated significantly from the expected distribution or exhibited unusual behavior relative to the majority of the data. However, in the dataset selected, no outlier was identified.

VIII. MODEL MAKING

All of the algorithms used the same preprocessed dataset. The code snippet for each algorithm is shown below.

- Convolutional Neural Network

```
model = Sequential()
model.add(Conv1D(filters=64,kernel_size=5,activation='relu',inp
ut_shape=(x_train_shape[1],1)))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=128,kernel_size=3,activation='relu'))
model.add(Conv1D(filters=256,kernel_size=3,activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(256,activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(128,activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1,activation='sigmoid'))
```

- Vanilla Neural network

```
Classifier=Sequential()
Classifier.add(Dense(units=6,activation='relu'))
Classifier.add(Dense(units=6,activation='relu'))
Classifier.add(Dense(units=1,activation='sigmoid'))
```

- Recurrent Neural Network

```
model=Sequential()
model.add(LSTM(128,input_shape=(x_train.shape[1:]),activation=
'relu',return_sequences=True))
model.add(LSTM(128,activation='relu'))
model.add(Dense(32,activation='relu'))
model.add(Dense(1,activation='sigmoid'))
```

- Decision Tree

```
Dt_classifier=DecisionTreeClassifier()
Dt_classifier.fit(x_train,y_train)
```

- Random Forest

```
rf_classifier=RandomForestClassifier(n_estimators=100)
rf_classifier.fit(x_train,y_train)
```

- Naïve Bayes

```
nb_classifier=GaussianNB()
nb_classifier.fit(x_train,y_train)
```

The diversity of models allows for a comprehensive evaluation of performance across various criteria.

IX. OUTCOME ANALYSIS

The evaluation of the model is based on the accuracies that the models produce in predicting the dataset. Some parameters such as precision, recall, and F1-Score were used to measure the accuracy. These are explained further below.

- Accuracy measures the overall correctness of the model's predictions.
- Precision focuses on the accuracy of positive predictions.
- Recall measures the model's ability to identify all positive instances correctly.
- F1-score provides a balanced assessment of precision and recall.

In addition, for this research, a confidence interval was implemented. This was to provide a range of which the actual data of measurement would fall. A confidence level of 95% was selected.

$$CI = SP \pm Z \times \sqrt{\frac{SP \times (1-SP)}{n}} \quad (1)$$

Where,

CI = Confidence Interval,

SP = Sample Proportion,

Z = Z-Score corresponding to the desired confidence level,

N = Sample.

The margin for the parameters was calculated using eqn. (2) below.

$$M = UL - LL \quad (2)$$

Where,

M = Margin size,

UL = Upper limit,

LL = Lower limit.

By comparing performance across these metrics, we gain valuable insights into the classification performance of each model and its suitability for network intrusion detection tasks.

X. RESULTS

The sample size which is 20% of the data points (25193) was 5039. The results of the experiment are shown in the following tables.

TABLE 2. TABLE OF INTERVAL MARGINS OF THE ACCURACIES OF MACHINE LEARNING MODELS

Machine Learning Models	Accuracy	Margin Size
Decision Tree (DT)	0.99524 ± 0.00190	0.0038
Random Forest (RF)	0.99762 ± 0.00135	0.0027
Naïve Bayes (NB)	0.89978 ± 0.00829	0.005

TABLE 3. PERFORMANCE SUMMARY OF MACHINE LEARNING MODELS

Machine learning Models	Precision	Recall	F1-Score
Decision Tree (DT)	0.99699 ± 0.00151	0.99402 ± 0.00213	0.99551 ± 0.00185
Random Forest (RF)	0.99664 ± 0.00159	0.99888 ± 0.00092	0.99776 ± 0.00131
Naïve Bayes (NB)	0.89739 ± 0.00837	0.91586 ± 0.00766	0.90653 ± 0.00804

TABLE 4. TABLE OF INTERVAL MARGINS OF THE ACCURACIES OF DEEP LEARNING MODELS

Deep Learning Models	Accuracy	Margin Size
Vanilla AI (VA)	0.98909 ± 0.00287	0.00574
CNN (CN)	0.99484 ± 0.00198	0.00396
RNN (RN)	0.53066 ± 0.01378	0.02756

TABLE 5. PERFORMANCE SUMMARY OF DEEP LEARNING MODELS

Deep Learning Models	Precision	Recall	F1-Score
Vanilla AI (VA)	0.98844 ± 0.0029	0.99102 ± 0.00260	0.98973 ± 0.00278
CNN (CN)	0.99588 ± 0.00177	0.99439 ± 0.00206	0.99513 ± 0.00192
RNN (RN)	0.53066 ± 0.01378	1.00000 ± 0.00000	0.69337 ± 0.01273

Fig. 2 shows the summary of the accuracies for each of the models used in the research.

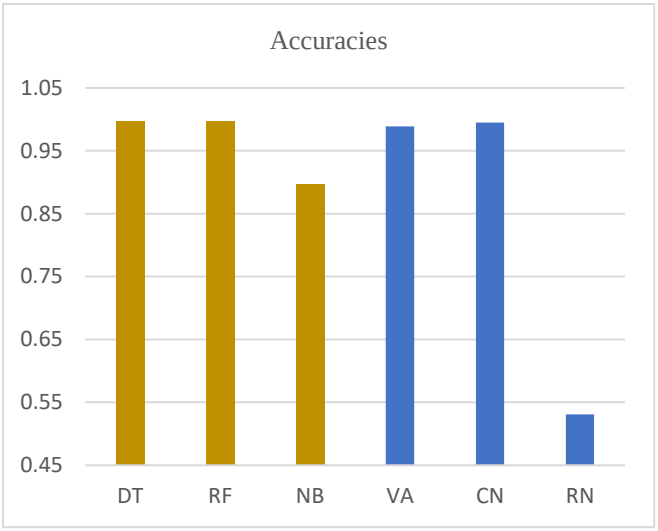

Fig. 2. Bar chart of accuracies of models

XI. CONCLUSION

Based on the experiment conducted on the six selected algorithms, the following insights were made.

- Among all the positive outcomes identified as normal traffic, the Decision Tree algorithm achieved the highest precision score. Also, the Random Forest algorithm excelled in recall and F1-score. This means Random Forest was the most effective at identifying actual positives within the dataset, as indicated by its recall score. Additionally, it demonstrated the best balance in classification, reflected in its higher F1-Score. Consequently, the Random Forest algorithm had the highest overall accuracy compared to the other machine learning algorithms.

- For the deep learning algorithms, CNN achieved the highest accuracy score. Because margin size is inversely proportional to accuracy, CNN also had the smallest margin size. Conversely, RNN had the lowest accuracy score but the highest recall score, indicating that RNN could classify all normal traffic (positives) and some anomalous traffic as normal. This is reflected in its lower F1-Score compared to the other algorithms. Overall, CNN excelled in all the measurement parameters.

- The deep learning algorithms performed similarly to the machine learning algorithms, as shown in the graph, except for the RNN algorithm. Despite deep learning being a relatively new field that requires further exploration, this suggests its promising potential in intrusion detection.

After the study, the outcomes for both ML and DL algorithms were similar, effectively differentiating normal traffic from anomalous traffic. However, the Random Forest and CNN algorithms excelled in their respective fields. Considering the promising potential of deep learning in Network Intrusion Detection, choosing a model, whether machine learning-based or deep learning-based, would depend on the user, the dataset, and the parameters used for training and classification.

XII. REFERENCES

[1] Lirim, A., & Cihan, D. (2021). Network Intrusion Detection System using Deep Learning. *ScienceDirect*, 16-18.

[2] Ander, J. P. (1980). Computer Security Threat Monitoring and Surveillance. *Technical Report*.

[3] Raman, B. N. (2022). Network Intrusion Detection System (NIDS). *First International Conference on Emerging Trends in Engineering and Technology.* Nitin Shivsharan.

[4] Daniel, B., Ningning, W., & Sushil, J. (2001). Detecting novel network intrusion using Bayes estimators. *First SIAM Conference.* Chicago.

[5] Khraisat A, G. I. (2018). An anomaly intrusion detection system using a C5 decision tree classifier. In: Trends and applications in knowledge discovery and data mining. *Springer International Publishing*, 149–155. From https://link.springer.com/chapter/10.1007/978-3-030-04503-6_14

[6] Pavol, M., & Pedro, C. (2018). Stream-based Machine Learning for Network Security and Anomaly Detection. *COMM*, 1-7.

[7] Markus, B., Hans-Perter, K., Raymond, T. N., & J"org, S. (2000). Lof: Identifying density-based local outliers. *ACM SIGMOD Conference.* Dallas: TX.

[8] Chuanlong, Y., Yuefei, Z., Jinlong, F., & Xinzheng, H. (2017). A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Network. *IEEE Access*, 21954-21961. doi:10.1109/ACCESS.2017.2762418

[9] *Introduction to Convolution Neural Network*. From https://www.geeksforgeeks.org/introduction-convolution-neural-network/

[10] *Random Forest Algorithm in Machine Learning*. From https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/

[11] *Decision Tree*. From https://www.geeksforgeeks.org/decision-tree/

[12] *Naïve Bayes Classifier Algorithm*. From https://www.javatpoint.com/machine-learning-naive-bayes-classifier

[13] Sampada, B. (2018). Network Intrusion Detection. *Kaggle*. Retrieved May 5, 2024, from https://www.kaggle.com/datasets/sampadab17/network-intrusion-detection