Machine Learning Models for UNSW-NB15 Dataset

Vishnusai Janjanam
Department of Computer Science
Indian Institute of Technology, Ropar
Rupnagar, India
2020csb1142@iitrpr.ac.in

Sukhmeet Singh
Department of Computer Science
Indian Institute of Technology, Ropar
Rupnagar, India
2020csb1129@iitrpr.ac.in

***Abstract*—Network security is an increasingly critical concern in the digital age, with organizations facing a multitude of cyber threats. In response to this challenge, the development of robust Intrusion Detection Systems (IDS) has become imperative. This paper presents a comprehensive study on the application of machine learning techniques to enhance network security, with a focus on the evaluation and comparison of Random Forest, Artificial Neural Network (ANN), and XGBoost models. This project explores the UNSW-NB15 dataset, a well-established benchmark in the field of network security, which includes various network-based attacks and normal traffic data. The machine learning models are applied to this dataset to detect and classify potential network intrusions, encompassing diverse attack categories, such as denial-of-service (DoS) attacks, port scans, and web application vulnerabilities. The research encompasses model development, training, and evaluation using appropriate metrics. The models' performance is assessed in terms of accuracy, precision, recall, F1-score, and computational efficiency. Additionally, the study provides insights into feature engineering and selection techniques to improve model performance. The findings of this research offer a comprehensive comparison of the three machine learning models' effectiveness in detecting network intrusions, shedding light on their strengths and weaknesses. This analysis is invaluable for both the academic community and the cybersecurity industry, providing guidance on the selection and deployment of appropriate IDS solutions.**

***Index Terms*—Network Security, Intrusion Detection System, Machine Learning**

I. Introduction

In today's interconnected world, the security of computer networks is of paramount importance. The proliferation of digital communication and data exchange has brought about a corresponding increase in cyber threats (see Fig.1), making network security a critical concern for individuals, businesses, and governments alike. The need for robust Intrusion Detection Systems (IDS) has never been more pressing, as these systems play a pivotal role in safeguarding network infrastructures against a wide array of malicious activities.

Network intrusion detection involves the continuous monitoring of network traffic to identify and respond to unauthorized and potentially harmful activities. Traditional rule-based IDS solutions have limitations in adapting to the evolving landscape of cyber threats. As a result, there has been a growing interest in harnessing the power of machine learning to enhance the efficacy of intrusion detection.

This research paper presents a comprehensive study on the application of machine learning techniques in the domain of network security. Specifically, it focuses on the evaluation and comparison of three widely employed machine learning models: Random Forest, Artificial Neural Network (ANN), and Long Short-Term Memory Recurrent Neural Network (LSTM-RNN). The objective is to assess the ability of these models to detect and classify network intrusions using the well-established UNSW-NB15 dataset, which captures various network-based attacks and normal network traffic.

The motivation behind this research is twofold. Firstly, it addresses the growing need for advanced IDS solutions that can effectively identify and respond to an ever-expanding range of cyber threats. Secondly, it seeks to provide valuable insights into the strengths and weaknesses of machine learning-based approaches for network intrusion detection. By comparing these three models, this study aims to offer guidance to the academic community and industry practitioners regarding the selection and deployment of suitable IDS solutions.

In the subsequent sections of this paper, we delve into the methodologies employed for model development, the features selected for analysis, the experimental setup, and the results obtained through extensive evaluation. The findings are expected to contribute to the ongoing discourse on network security and facilitate informed decisions in the quest to fortify network defenses.

II. Related Work & Background studies

In the ever-evolving landscape of network security, Network Intrusion Detection Systems (NIDS) serve as essential guardians against emerging cyber threats. These systems vigilantly monitor network traffic, promptly identifying and responding to anomalies or malicious activities to ensure the resilience of digital infrastructures. Over time, from early rule-based methods to contemporary machine learning-driven approaches, the evolution of NIDS underscores the continuous pursuit of adaptability and efficacy. Early work by Denning [14] laid the groundwork for anomaly-based detection, and ongoing research explores advanced machine learning models and newer datasets, such as UNSW-NB15 [1], [9], to propel the capabilities of NIDS.

Wilton et al. [7] propose new algorithms for learning from positive and unlabeled data (PU learning) using random forests. They introduce a novel perspective of decision tree learning as recursive greedy risk minimization and extend it to the PU setting.

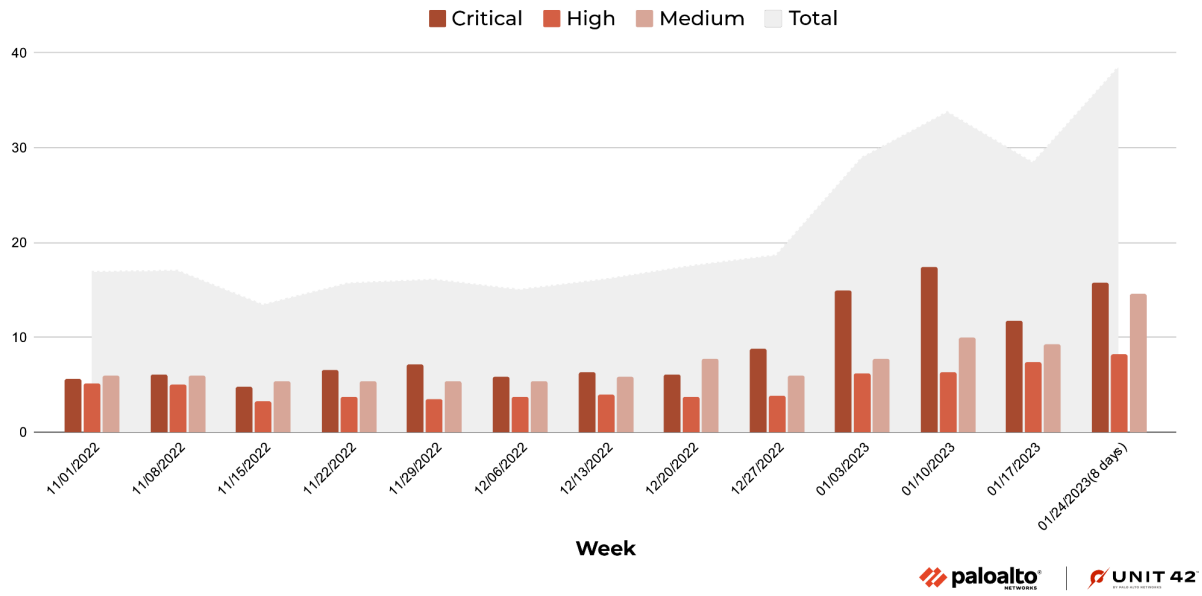

Fig. 1. **Growth of Network Attacks. [6]**

Hartl et al. [8] propose a recurrent neural network (RNN) based intrusion detection system (IDS) for network data, and evaluate its adversarial robustness, explainability, and defense methods. They show that RNNs can achieve high accuracy and detect attacks before the flow terminates, but also that they are vulnerable to adversarial attacks that manipulate packet length and Interarrival time. They introduce the Adversarial Robustness Score (ARS) as a new metric to quantify the IDS's resistance to such attacks, and demonstrate that it can be improved by adversarial training. They also apply and extend several feature importance and explainability methods to understand how the RNN makes its decisions, and reveal the influence of different features, packets, and values on the classification outcome.

Bhamare et al. [10] discuss the security issues in cloud computing and the limitations of supervised machine learning (ML) techniques for intrusion detection systems (IDS). They trained and tested four ML algorithms -SVM, Naïve Bayes, and Logistic Regression using two different datasets UNSW for training and ISOT [15] for testing obtained from simulated cloud environments. The authors found that the ML models perform well with the same dataset but poorly with a different dataset, indicating a lack of robustness and applicability to real scenarios. They suggest exploring unsupervised ML and anomaly detection techniques for cloud security.

Silivery et al. [11] propose a model for multi-attack classification to improve intrusion detection performance using deep learning approaches, such as RNN, LSTM-RNN, and DNN. The authors use three deep learning models to self-learn the features and classify the network data into different attack types, such as DoS, Probe, R2L, U2R, and Normal2. The models are trained and tested on three benchmark datasets: KDD'99, NSL-KDD, and UNSW-NB15. The authors compare the results of their models with existing shallow and deep learning methods in terms of accuracy, detection rate, false alarm rate, and other metrics. They show that their models outperform the existing methods, especially in detecting low-frequency attacks like R2L and U2R.

Sinha et al. [13] propose a deep learning model that combines a 1-D convolutional neural network (CNN) and a bidirectional long short-term memory (BiLSTM) network to detect network intrusions from time-series data. The authors use a 1-D CNN layer to learn the spatial features of the network data, followed by max pooling, batch normalization, and reshape layers1. The authors use two BiLSTM layers to learn the temporal features of the network data from both forward and backward directions, followed by dropout and dense layers.

III. Description of Proposed Framework

This section explains how we create machine-learning and neural network models for the UNSW-NB15 dataset. This section provides a detailed overview of the procedures involved in both training and assessing the models' performance. To give a clearer picture, let's delve into the specifics.

A. Data Preprocessing

- **Loading Data :** The dataset is available in csv file format with mostly numeric columns. The training data distribution can be found in the Fig.2
- **Feature Extraction :** To capture essential characteristics, 40 columns were chosen. Columns for transportation and application layer protocols and the state of the connection were dropped.
- **Encoding Non-integer columns :** The attack classes were encoded from 0 to 9 to be fed to different machine learning and neural network models.

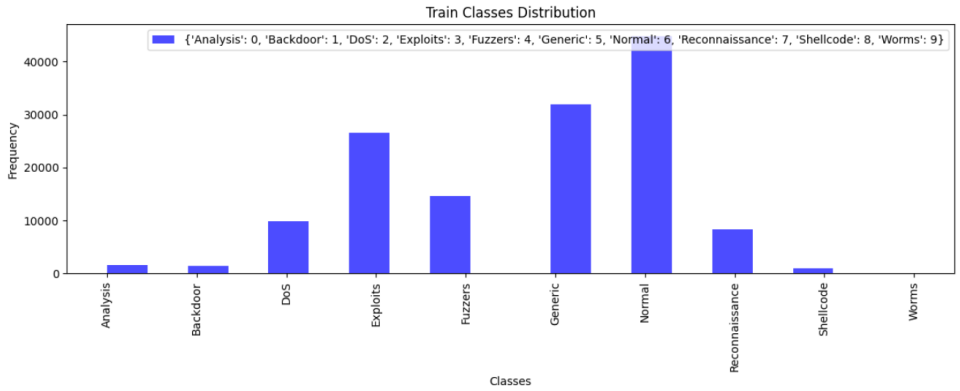

Fig. 2. **Train Data distribution of different classes**

- **Feature scaling and clamping:** The features were mostly right-skewed. So, log transformation is applied to decrease the skewness. To further reduce the skewness, the top 5% data is replaced with the 95th percentile data.
- **Standardization :** As values in numerical data can affect how the model gets trained, all the columns are standardized to prevent it.

B. Neural Network Models

1) Artificial Neural Network Architecture:

- **Input Layer :** The 40 standardized columns are used as input for this architecture.
- **Hidden Layers :** Our architecture contains three hidden layers containing 100, 75, and 50 neurons in order with the ReLU activation function. Additionally, dropout was used to prevent overfitting of the model.
- **Output Layer :** Each class has its own neuron to predict the probability. So, ten output neurons are used with a softmax activation function.

2) Long Short Term Memory Architecture:

- **Input Layer :** The 40 standardized columns are used as input for this architecture.
- **LSTM Layers:** Three LSTM layers are employed sequentially. The first LSTM layer consists of 150 units, the second layer comprises 100 units, and the third LSTM layer has 50 units. Each layers uses `tanh` activation and dropout to prevent overfitting.
- **Output Layer :** The final layer is a dense layer with ten units, one for each class, with a softmax activation function.

3) Convolutional Neural Network Architecture:

- **Input Layer :** The 40 standardized columns are used as input for this architecture.
- **Convolutional Layers:** The architecture includes three convolutional layers. The first layer has 128 filters with a kernel size of 7, and the next two have 256 filters with a kernel size of 3. Batch normalization is applied, followed by a ReLU activation for each layer.
- **MaxPooling and Flatten:** A MaxPooling1D layer is applied with a pool size of 3, and then a Flatten layer is used to flatten the output into a one-dimensional array.
- **Fully Connected (Dense) Layers:** Two fully connected (dense) layers follow with 256 units each and ReLU activation. Dropout with a rate of 0.2 is applied after each dense layer for regularization.
- **Output Layer :** The final layer is a dense layer with ten units, one for each class, with a softmax activation function.

C. Machine Learning Models

- Standard implementation of Decision Trees and Random Forest is used for the classification purpose.

IV. Experimental Results

Once the model is trained, assess its performance on the testing data by utilizing the evaluate function. This function calculates metrics like loss and accuracy to gauge how well the model performs on new, unseen data. Furthermore, it can compute additional evaluation metrics such as precision, recall, and F1-score to analyze better the model's ability to detect network attack.

The UNSW-NB15 dataset comprised 175,341 traning samples, each containing 40 extracted features. We have allocated 20% of the training data for validation when working with all the neural network models. The testing dataset consisted of 82,332 samples.

To evaluate the performance of our model, we calculated the proportion of correctly labeled samples. We implemented the Adam optimizer and categorical cross-entropy for the feed-forward neural network as the loss function. With a batch size 2048, the model underwent training for 30 epochs. On the test dataset, it had an accuracy of 68.95%. You can observe the training and validation loss trends in Fig.3, which decrease and stabilize towards the end of training. Refer Table I for other metrics.

For the Convolutional Neural Network (CNN), we used a Stochastic Gradient Descent (SGD) optimizer and categorical

Fig. 3. **Training Loss Curves for neural network models**

TABLE I
PRECISION, RECALL, AND F1 SCORE FOR EACH CLASS IN THE FEED FORWARD NEURAL NETWORK.

Class	Precision	Recall	F1 Score
Analysis	0.00	0.00	0.00
Backdoor	0.00	0.00	0.00
DoS	0.00	0.00	0.00
Exploits	0.48	0.96	0.64
Fuzzers	0.23	0.68	0.34
Generic	1.00	0.95	0.97
Normal	1.00	0.60	0.75
Reconnaissance	0.92	0.48	0.63
Shellcode	0.00	0.00	0.00
Worms	0.00	0.00	0.00

TABLE II
PRECISION, RECALL, AND F1 SCORE FOR EACH CLASS IN THE CONVOLUTIONAL NEURAL NETWORK.

Class	Precision	Recall	F1-score
Analysis	0.00	0.00	0.00
Backdoor	0.00	0.00	0.00
DoS	0.31	0.18	0.23
Exploits	0.50	0.89	0.64
Fuzzers	0.26	0.42	0.32
Generic	1.00	0.96	0.98
Normal	0.97	0.72	0.83
Reconnaissance	0.62	0.72	0.67
Shellcode	0.58	0.10	0.16
Worms	0.00	0.00	0.00

cross-entropy as the loss function. The accuracy for CNN is 77.32%. The number of epochs is 30, and the batch size is 2048. Precision, recall, and F1 score can be found in the Table II.

For the Long Short Term Memory (LSTM) model, we used a Adam optimizer and categorical cross-entropy as the loss function. The number of epochs is 5 as LSTM is heavy to train, and the batch size is 2048. The accuracy for LSTM is 54.52%. Precision, recall, and F1 scores can be found in Table III.

The accuracy of the Decision Tree is 70.31%. The precision, recall, and F1 scores for Decision Tree can be viewed in Table IV and accuracy for Random Forest is 71.02% and precision, recall and F1 scores can be viewed in Table V.

TABLE III
PRECISION, RECALL, AND F1 SCORE FOR EACH CLASS IN THE LSTM.

Class	Precision	Recall	F1-score
Analysis	0.00	0.00	0.00
Backdoor	0.00	0.00	0.00
DoS	0.00	0.00	0.00
Exploits	0.33	0.80	0.47
Fuzzers	0.16	0.27	0.20
Generic	0.80	0.95	0.87
Normal	0.94	0.48	0.64
Reconnaissance	0.03	0.04	0.04
Shellcode	0.00	0.00	0.00
Worms	0.00	0.00	0.00

TABLE IV
PRECISION, RECALL, AND F1 SCORE FOR EACH CLASS IN THE DECISION TREE.

Class	Precision	Recall	F1-score
Analysis	0.00	0.00	0.00
Backdoor	0.00	0.00	0.00
DoS	0.31	0.18	0.23
Exploits	0.50	0.89	0.64
Fuzzers	0.26	0.42	0.32
Generic	1.00	0.96	0.98
Normal	0.97	0.72	0.83
Reconnaissance	0.62	0.72	0.67
Shellcode	0.58	0.10	0.16
Worms	0.00	0.00	0.00

V. CONCLUSION

Based on our analysis and evaluation of the network attack detection model using the UNSW-NB15 dataset, specifically employing a Convolutional Neural Network (CNN) architecture with 40 selected columns, we can draw the following conclusions: The CNN model we implemented demonstrates promising performance in accurately categorizing network attacks and normal network traffic. It achieves commendable levels of accuracy, precision, recall, and F1-score, highlighting its proficiency in recognizing instances of malicious activities. The selected features, including various network-related attributes, prove valuable in distinguishing between network attacks and benign network behavior. These features play a pivotal role in enabling the model to make precise classifications. The interpretability of the CNN model is a noteworthy aspect of our

TABLE V
PRECISION, RECALL, AND F1 SCORE FOR EACH CLASS IN THE RANDOM FOREST.

Class	Precision	Recall	F1-score
Analysis	0.04	0.83	0.08
Backdoor	0.01	0.04	0.02
DoS	0.84	0.06	0.12
Exploits	0.80	0.64	0.71
Fuzzers	0.28	0.43	0.34
Generic	1.00	0.96	0.98
Normal	0.98	0.72	0.83
Reconnaissance	0.94	0.78	0.86
Shellcode	0.59	0.44	0.50
Worms	0.67	0.09	0.16

analysis. It provides insights into the significance of different features in the classification process, contributing to a better understanding of the characteristics of network attacks. Unlike the paper's approach, we did not reduce the feature set, utilizing all 40 columns for our analysis. Despite the success of the CNN model, there is still room for improvement in future work. Further refining the model's design and fine-tuning its parameters hold the potential to enhance its effectiveness in detecting network attacks. Experimentation with alternative network structures, exploration of dropout rates, or investigation into advanced methods such as recurrent neural networks (RNNs) or transformers could be beneficial.

REFERENCES

[1] Moustafa, Nour, and Jill Slay. "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)." Military Communications and Information Systems Conference (MilCIS), 2015. IEEE, 2015.

[2] Moustafa, Nour, and Jill Slay. "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 dataset and the comparison with the KDD99 dataset." Information Security Journal: A Global Perspective (2016): 1-14.

[3] Moustafa, Nour, et al. "Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks." IEEE Transactions on Big Data (2017).

[4] Moustafa, Nour, et al. "Big data analytics for intrusion detection system: statistical decision-making using finite Dirichlet mixture models." Data Analytics and Decision Support for Cybersecurity. Springer, Cham, 2017. 127-156.

[5] Sarhan, Mohanad, Siamak Layeghy, Nour Moustafa, and Marius Portmann. "NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems." In Big Data Technologies and Applications: 10th EAI International Conference, BDTA 2020, and 13th EAI International Conference on Wireless Internet, WiCON 2020, Virtual Event, December 11, 2020, Proceedings. Springer Nature.

[6] Palo Alto Networks https://unit42.paloaltonetworks.com/network-security-trends-nov-jan/

[7] Wilton, J., Koay, A. M. Y., Ko, R. K. L., Xu, M., & Ye, N. (2022). Positive-Unlabeled Learning using Random Forests via Recursive Greedy Risk Minimization. arXiv preprint arXiv:2210.08461

[8] Hartl, A., Bachl, M., Fabini, J., & Zseby, T. (2020). Explainability and Adversarial Robustness for RNNs. In 2020 IEEE Sixth International Conference on Big Data Computing Service and Applications (BigDataService) (pp. 141-150). IEEE. arXiv preprint arXiv:1912.09855

[9] Divekar, A., Parekh, M., Savla, V., Mishra, R., & Shirole, M. (2018). Benchmarking datasets for Anomaly-based Network Intrusion Detection: KDD CUP 99 alternatives. arXiv preprint arXiv:1811.05372.

[10] Bhamare, D., Salman, T., Samaka, M., Erbad, A., & Jain, R. (2018). Feasibility of Supervised Machine Learning for Cloud Security. In 2016 International Conference on Information Science and Security (ICISS) (pp. 1-6). IEEE.

[11] Silivery, A.K., Kovvur, R.M.R., Solleti, R., Kumar, L.S., & Madhu, B. (2023). A model for multi-attack classification to improve intrusion detection performance using deep learning approaches. Measurement: Sensors, 30, 100924. https://doi.org/10.1016/j.measen.2023.100924

[12] Ahmad, M., Riaz, Q., Zeeshan, M., Tahir, H., Haider, S. A., & Khan, M. S. (2021). Intrusion detection in internet of things using supervised machine learning based on 'application and transport layer features using UNSW-NB15 data-set. EURASIP Journal on Wireless Communications and Networking, 2021. https://doi.org/10.1186/s13638-021-01893-8

[13] Sinha, J., & Manollas, M. (2020). Efficient Deep CNN-BiLSTM Model for Network Intrusion Detection. In AIPR'20: Proceedings of the 2020 International Conference on Artificial Intelligence in Information and Communication (pp. 223-231). ACM https://doi.org/10.1145/3430199.3430224

[14] Denning, Dorothy E. "An Intrusion-Detection Model." ACM Transactions on Computer Systems (TOCS), 1986.

[15]] S. Sherif, I. Traore, A. Ghorbani, B. Sayed, D. Zhao, W. Lu, J. Felix, P. Hakimian, "Detecting P2P botnets through network behavior analysis and machine learning," Proceedings of 9th Annual Conference on Privacy, Security and Trust (PST2011), 2011