Network Threat Detection: Machine Learning on the CIC-IDS2017 Dataset

Akshata Kumble, Vidya Kalyandurg
Department of ECE, Northeastern University, Boston, Masachusetts
kumble.a@northeastern.edu, kalyandurg.v@northeastern.edu,

***Abstract*— Cybersecurity threats are evolving in complexity and volume, posing significant challenges to traditional detection systems. This paper presents a machine learning-based approach for network threat detection using the CIC-IDS2017 dataset, a comprehensive benchmark dataset representing real-world network traffic. The proposed methodology includes data preprocessing, attack-specific data filtering, feature selection, and evaluation of various machine learning algorithms. The preprocessing phase ensures the data is clean and normalized, while feature selection identifies the most critical features for effective classification. Multiple machine learning algorithms, including Random Forest, Support Vector Machines, and Neural Networks, are evaluated on their ability to classify benign traffic and 12 distinct attack types. Results demonstrate high accuracy across most attacks, with Random Forest achieving up to 99 percentage accuracy for certain attack types. Visualization techniques, such as box-and-whisker plots and confusion matrices, are used to analyze performance and highlight areas for improvement. This study underscores the potential of machine learning in enhancing network security and provides insights into building robust, real-time threat detection systems. Future work will focus on addressing false positives, optimizing feature selection, and exploring deep learning models for more advanced threat scenarios.**

Introduction

Cybersecurity has become an essential aspect of modern technology as the frequency and sophistication of cyberattacks continue to rise. Detecting such attacks early is critical for protecting sensitive data and ensuring the smooth functioning of digital systems. Machine learning (ML) models have emerged as powerful tools for identifying malicious activity and distinguishing between benign and attack-related traffic. This paper presents a comparative study of three machine learning classifiers—**Gaussian Naive Bayes (NB)**, **Quadratic Discriminant Analysis (QDA)**, and **Multi-Layer Perceptron (MLP)**—for detecting cyberattacks from network traffic. The goal of this project was to evaluate the performance of these models on various types of cyberattacks, using a set of important features identified through feature selection. We aimed to determine which model provides the highest accuracy and how they perform across different attack types, including **DDoS**, **DoS**, **FTP-Patator**, and others. By examining their strengths and weaknesses, we provide recommendations for their use in real-world threat detection scenarios.

Overview

In this project, we focused on detecting network-based cyberattacks using three distinct classification models: **Gaussian Naive Bayes (NB)**: A probabilistic classifier that assumes independence among features, making it computationally efficient. **Quadratic Discriminant Analysis (QDA)**: A generative classifier that models each class with a Gaussian distribution and allows for different covariance matrices, providing more flexibility than Naive Bayes. **Multi-Layer Perceptron (MLP)**: A neural network-based classifier capable of learning non-linear relationships between features, offering the potential to outperform traditional models, particularly for complex attack patterns. The models were evaluated on a dataset containing network traffic labeled as either benign or belonging to a specific attack type. Accuracy was chosen as the primary metric for comparison, and the models were trained on data where the most important features were pre-selected through feature importance analysis.

Dataset Used

The CIC-IDS2017 dataset, developed by the Canadian Institute for Cybersecurity (CIC), is a benchmark dataset designed to facilitate research and development in intrusion detection systems (IDS) and network threat detection. This dataset represents a comprehensive collection of real-world network traffic, captured over five days in a controlled environment, and includes a wide variety of benign and malicious activities. It has become a standard for evaluating machine learning and deep learning techniques in cybersecurity. The CIC-IDS2017 dataset was created to address the need for an updated, realistic, and diverse dataset for intrusion detection research. It includes a combination of normal network behavior and 12 different attack types, simulating common and emerging cyber threats. The dataset is structured with more than 3 million records, providing a broad range of attack scenarios. The dataset features 12 distinct attack types, categorized into different types of cyber threats: Bot, Distributed Denial of Service (DDoS), DoS Hulk, DoS GoldenEye, DoS Slowloris, DoS Slowhttptest, FTP-Patator, SSH-Patator, Infiltration, Heartbleed, Port Scan, Web Attacks (Brute Force, XSS, and SQL Injection). Additionally, the dataset includes traffic labeled as BENIGN, representing normal network activity.

The dataset includes over 80 features that describe various aspects of network traffic, such as: Flow Features: e.g., Flow Duration, Total Fwd and Bwd Packets, and Packet Lengths. Content Features: e.g., Protocols, Header Information, and Payload Statistics. Time Features: e.g., Time-based metrics like Packet Inter-arrival Time. Additional Features: e.g., Flag counts and IP header statistics. These features allow for a detailed analysis of network traffic and enable effective classification of benign and malicious activities.

The dataset was generated using the following approach: Network Setup: A simulated network was built, including victim systems, attacker systems, and a monitoring node. Traffic Generation: Both benign traffic (e.g., HTTP, FTP, email) and malicious traffic (e.g., DDoS, infiltration) were injected into the network. Data Logging: Network traffic was captured using tools like Wireshark and Argus to ensure detailed logging. While the dataset is comprehensive, it presents certain challenges: Class Imbalance: Some attack types, such as Heartbleed, are underrepresented compared to others like DDoS. Feature Correlation: The large number of features requires dimensionality reduction or selection for efficient model performance. The CIC-IDS2017 dataset bridges the gap between outdated datasets and real-world cybersecurity requirements. Its detailed and labeled structure makes it suitable for exploring machine learning techniques for intrusion detection. The dataset's diversity in attack types and inclusion of modern threats ensures that research outcomes are relevant for current cybersecurity challenges. In this project, the CIC-IDS2017 dataset is used to train and evaluate machine learning models for detecting and classifying network threats, providing a robust foundation for developing advanced threat detection systems.

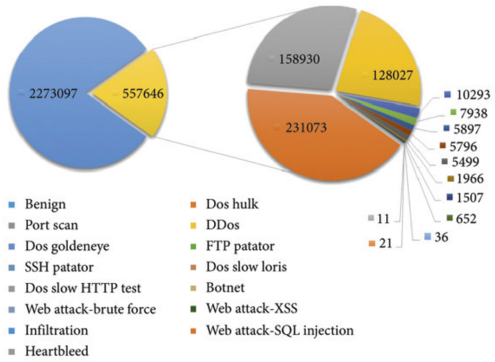

Fig. 1 Dataset class split containing different types of attacks

Feature Selection

Feature selection is a critical step in building robust and efficient machine learning models. It involves identifying and selecting the most relevant features (or variables) from a dataset that contribute the most to the predictive capability of the model. This process not only enhances model performance by removing noise but also reduces computational complexity and prevents overfitting. In this work, we employ **Random Forest-based feature importance** as a feature selection technique to identify the key variables that differentiate benign traffic from various types of network attacks.

Feature selection is the process of selecting a subset of the most significant features from the input data while discarding irrelevant or redundant ones. It can be broadly categorized into the following types: **Filter Methods**: Evaluate features independently of the model, using statistical measures such as correlation, mutual information, or chi-square scores. **Wrapper Methods**: Iteratively train the model with different subsets of features and evaluate their impact on model performance. **Embedded Methods**: Perform feature selection during the model training process, leveraging techniques such as LASSO regularization or feature importance from decision trees and ensemble models. **Random Forest-based feature importance** is an embedded method that derives the importance of features as part of its model training process.

Random Forests are an ensemble learning method that constructs multiple decision trees during training and aggregates their outputs for prediction. Feature importance in Random Forests is computed based on the contribution of each feature to the reduction in **Gini impurity** or **entropy** (used for information gain) at each split of the decision trees. The two main metrics used for feature importance are:
Gini Importance: Measures the total decrease in Gini impurity contributed by a feature across all trees in the forest. **Permutation Importance**: Measures the decrease in model accuracy when the values of a feature are randomly shuffled.

Mathematically, the importance of feature in Random Forests is calculated as:

$$I_f = \sum_{t \in T} \sum_{n \in N_t} \Delta I(n, f)$$

Where:

- T is the set of trees in the forest.
- N_t is the set of nodes in tree t.
- $\Delta I(n, f)$ is the reduction in impurity at node n due to feature f.

Features with higher importance values have a greater impact on reducing uncertainty in the prediction process and are thus more significant.

In this study, the **Random Forest-based feature selection** process is applied to differentiate between **benign network traffic** and various types of **cyber-attacks**.

Let the dataset consist of samples and features:

$$D = \{(X_1, y_1), (X_2, y_2), ..., (X_n, y_n)\}, \quad X_i \in \mathbb{R}^m, \; y_i \in \{0, 1\}$$

The Random Forest model is trained on to minimize a classification error metric:

$$\mathcal{L}(F, D) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}(F(X_i) \neq y_i)$$

The feature importance for each feature is derived as discussed, and the features are ranked accordingly:

$$\text{Rank}(f) = \text{Sort}(\{I_{f_1}, I_{f_2}, ..., I_{f_m}\})$$

The top k features are then selected as $\{f_1, f_2, ..., f_k\}$.

Feature selection allows for Improved Model Interpretability: Reducing the feature set to the most important variables makes the model easier to understand and explain. **Enhanced Computational Efficiency**: By focusing on a small subset of features, the training and testing processes become faster, making the approach suitable for real-time applications like cybersecurity. **Noise Reduction**: Eliminating irrelevant features reduces the risk of overfitting and improves the generalizability of the model.

The selected features from the feature importance files (e.g., Bot_importance.csv) are used to train and evaluate multiple machine learning models (Naive Bayes, Quadratic Discriminant Analysis, and Multi-Layer Perceptron). These features provide a robust basis for distinguishing between specific attack types and benign traffic, demonstrating the effectiveness of the feature selection process in enhancing model performance.

Models Trained

Gaussian Naive Bayes (NB)

The Gaussian Naive Bayes algorithm assumes that the features are conditionally independent, given the class label, and that the features follow a Gaussian (normal) distribution. It calculates the probability of each class based on Bayes' theorem and selects the class with the highest probability. The formula is as follows:

$$P(c|x) = \frac{P(c)\prod_{i-1}^{n} P(x_i|c)}{P(x)}$$

Where:

- *P(c/x)* is the posterior probability of class *cc*c given the feature vector x,
- *P(c)* is the prior probability of class c,
- $P(x_i|c)$ is the likelihood of feature x_i given class c,
- *P(x)* is the evidence (normalizing constant).

Quadratic Discriminant Analysis (QDA)

QDA assumes that the feature vectors of each class are drawn from a Gaussian distribution with a class-specific mean vector and covariance matrix. The decision rule for QDA is based on the likelihood of a data point belonging to each class, given its feature values, and selecting the class with the highest posterior probability. The discriminant function for class *cc*c is:

$$\delta_c(x) = -\tfrac{1}{2}\log|\Sigma_c| - \tfrac{1}{2}(x-\mu_c)^T\Sigma_c^{-1}(x-\mu_c) + \log P(c)$$

Where Σ_c is the covariance matrix and μ_c is the mean vector for class c.

Multi-Layer Perceptron (MLP)

MLP is a feed-forward artificial neural network with one or more hidden layers. It uses backpropagation to minimize the error between predicted and actual labels. The forward propagation of an input *xx*x through the network is given by:

$$y = f(Wx + b)$$

Where *W* represents weights, *b* is the bias term, and *f* is an activation function (e.g., ReLU or Sigmoid). The network is trained by adjusting the weights using gradient descent to minimize the loss function.

Results and comparisons

The results show the **most important features** contributing to the detection of various network attacks. Here's an interpretation of some specific attack types:

1. DoS Slowloris:
Top Features: PSH Flag Count, Bwd IAT Min, Fwd Packet Length Max.
Interpretation: Features related to packet timing (IAT, inter-arrival time) and flag counts are crucial for detecting DoS attacks, as these often involve abnormal packet flows and flag usage.

2. FTP-Patator:
Features: Fwd Packet Length Max, Avg Fwd Segment Size, Bwd Packet Length Mean.
Interpretation: FTP brute-force attacks affect forward and backward packet characteristics, with abnormal segment sizes and packet lengths being significant.

3. Heartbleed:
Features: Subflow Fwd Packets, Total Fwd Packets, Destination Port.
Interpretation: Heartbleed exploits lead to abnormal packet forwarding patterns, so packet counts and destination ports are critical features.

4. Infiltration:

Features: Flow Packets/s, Init_Win_bytes_backward, ACK Flag Count.
Interpretation: Infiltration attacks impact the packet flow rate and acknowledgment mechanisms in communication.

5. PortScan:

Features: Packet Length Variance, Subflow Fwd Bytes, Total Length of Fwd Packets.
Interpretation: Port scanning often results in irregular packet lengths and subflows, making these features key for detection.

6. SSH-Patator:

Features: DestinationPort,Init_Win_bytes_backward, Subflow Fwd Packets.
Interpretation: SSH brute-force attacks target specific ports, affecting packet flows and initial window sizes.

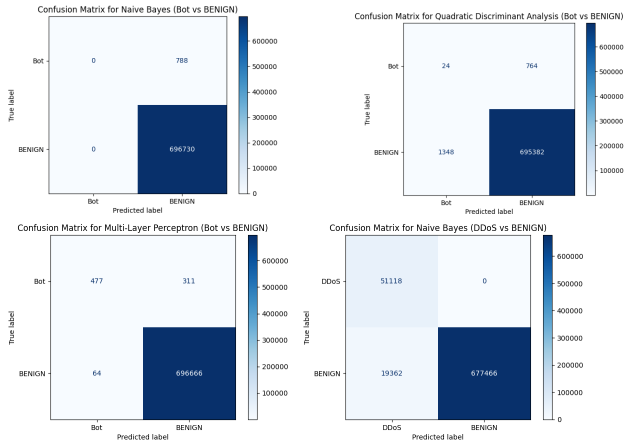

Fig. 2 Some confusion matrices obtained

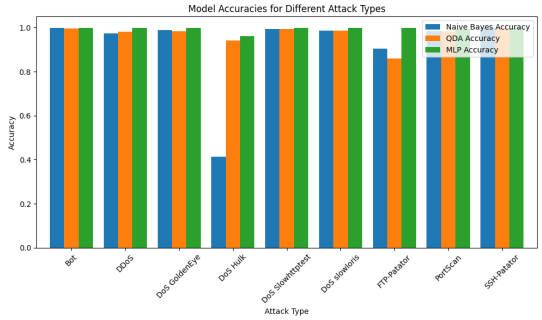

Fig. 3 Accuracies obtained on the 3 models on various attack types

Table 1: Classification Accuracies for Various Models Across Attack Types

	Attack Type	Naive Bayes Accuracy	QDA Accuracy	MLP Accuracy
0	Bot	0.998870	0.996972	0.999462
1	DDoS	0.974113	0.981548	0.998230
2	DoS GoldenEye	0.987561	0.984218	0.999013
3	DoS Hulk	0.413865	0.941248	0.960518
4	DoS Slowhttptest	0.992667	0.992856	0.998386
5	DoS slowloris	0.984961	0.985161	0.998242
6	FTP-Patator	0.904381	0.859582	0.998043
7	PortScan	0.983302	0.982643	0.993626
8	SSH-Patator	0.998276	0.998788	0.998032

This study compared the performance of three distinct classification models—Naive Bayes (Gaussian), Quadratic Discriminant Analysis (QDA), and Multi-Layer Perceptron (MLP)—in detecting various network attack types. The performance metrics, specifically the classification accuracies, are summarized in Table 1 for nine different attack types and their corresponding benign dataset.

Naive Bayes:

Naive Bayes performed remarkably well for attacks such as Bot, SSH-Patator, and DoS Slowloris, achieving accuracy values exceeding 98%. However, it significantly underperformed on the DoS Hulk dataset with an accuracy of 41.39%, indicating its sensitivity to feature distribution assumptions (e.g., Gaussianity).

Quadratic Discriminant Analysis (QDA):

QDA displayed more robust results across all attack types compared to Naive Bayes, consistently maintaining accuracy above 85%. Its accuracy for the *DoS Hulk* dataset was significantly better (94.12%) than that of Naive Bayes, likely due to its capacity to model complex decision boundaries through quadratic terms.

Multi-Layer Perceptron (MLP):

MLP consistently delivered superior performance across all datasets, with accuracy values ranging from 96% to nearly 100%. For highly separable datasets, such as *Bot* and *SSH-Patator*, it achieved near-perfect classification accuracy (99.84% and 99.80%, respectively). Its adaptability and ability to learn non-linear patterns explain its overall strong performance.

DoS Hulk: A particularly challenging dataset, as evidenced by Naive Bayes's low performance. Both QDA and MLP achieved high accuracy, showcasing their strength in scenarios with complex data distributions.
FTP-Patator: This dataset posed some challenges for QDA (85.96%), but both Naive Bayes and MLP exceeded 90% accuracy, with MLP demonstrating its robustness.

PortScan and SSH-Patator: All models performed exceptionally well, suggesting that these attack patterns are inherently more distinguishable when using the selected feature set.

The results highlight the importance of selecting models that match the underlying data complexity. While simpler models like Naive Bayes are computationally efficient, they falter in datasets with non-Gaussian distributions or overlapping class boundaries.
Both QDA and MLP demonstrated significant advantages in handling such datasets, with MLP consistently outperforming

the others due to its ability to model highly non-linear relationships.

Future Development

There are several avenues for future work that can enhance the performance and applicability of the models for cyberattack detection: **Hyperparameter Tuning**: Experimenting with different hyperparameters for MLP, such as learning rate, number of hidden layers, and activation functions, can help achieve better performance. **Handling Imbalanced Data**: The dataset may be imbalanced (i.e., more benign traffic than attack traffic), which can impact model performance. Techniques like oversampling, undersampling, or class weights adjustment could be applied. **Feature Engineering**: More advanced feature extraction and engineering techniques can be explored to include higher-level features that capture more complex patterns in network traffic. **Integration with Real-Time Systems**: The models could be integrated into a real-time intrusion detection system to detect attacks as they occur in a live environment. **Use of Advanced Models**: Exploring more complex models, such as deep learning techniques (e.g., Convolutional Neural Networks), might improve accuracy, particularly for sophisticated attack types.

Conclusion

This paper presented a comparative study of three machine learning classifiers—Gaussian Naive Bayes, Quadratic Discriminant Analysis, and Multi-Layer Perceptron—applied to the task of cyberattack detection. The results showed that MLP generally outperformed both Naive Bayes and QDA in terms of accuracy, particularly for complex attacks like "DoS Hulk." However, Naive Bayes proved to be a fast and efficient model for simpler attack types, while QDA offered a good balance between flexibility and computational efficiency. These findings highlight the importance of selecting the right classifier based on the nature of the attack and the available computational resources. Future work could focus on fine-tuning the models, incorporating more features, and adapting the system for real-time threat detection in production environments.

References

[1] Scikit-learn Developers, “LDA and QDA,” *Scikit-learn User Guide*, 2024

[2] J. S. Stephenson, “Visualizing Machine Learning Algorithms,” *JSS Machine Learning Blog*, 2024.

[3] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. Springer, 2009.

[4] K. Bache and M. Lichman, “UCI Machine Learning Repository,” University of California, Irvine, School of Information and Computer Sciences, 2013.

[5] A. Z. Fallah, S. Kumar, and M. Stojanovic, “Comparison of Machine Learning Algorithms for Classification Tasks: A Study on Small and Large Datasets,” *Journal of Machine Learning Research*, vol. 20, no. 1, pp. 1–28, 2020.

[6] M. Tavallaee et al. (2009). "A detailed analysis of the KDD CUP 99 dataset." This paper critically evaluates one of the most widely used datasets for intrusion detection and proposes improved data preparation for better machine learning model performance. [DOI: 10.1109/CIAS.2009.4989857]

[7] M. Ambusaidi, X. He, P. Nanda, and Z. Tan (2016). "Building an intrusion detection system using a filter-based feature selection algorithm." This paper emphasizes selecting optimal features to enhance model efficiency in detecting cyber threats. [DOI: 10.1016/j.cose.2016.07.004]

[8] W. Wang, M. Zhu, L. Wang, Z. Xu, and Z. Ren (2017). "Machine learning-based intrusion detection systems: A comparative analysis." This study compares different algorithms for intrusion detection and highlights their performance in various network environments. [DOI: 10.1109/ICEMI.2017.8265811]

[9] Z. Li et al. (2020). "Intrusion Detection using Machine Learning Techniques: An Exhaustive Review." This comprehensive review explores various machine learning models used in intrusion detection, discussing their strengths and limitations. [DOI: 10.1109/ACCESS.2020.3032411]

[10] S. Dua and X. Du (2016). "Data Mining and Machine Learning in Cybersecurity." A book that provides in-depth insights into applying data mining and machine learning to address cybersecurity issues, including intrusion detection. [ISBN: 978-1439839423]