Data Mining for Intrusion Detection: An Analysis Using the CIC-IDS2017 Dataset

Bernardo Vitorino
Universidade de Évora
m57376@alunos.uevora.com

Abstract

1 Introduction

Maintaining the security and dependability of networked systems has become increasingly difficult in recent years due to the growing complexity and scope of cyber attacks. In order to protect sensitive data's confidentiality and integrity, intrusion detection systems, or IDS, are essential for spotting and stopping harmful activity. Conventional intrusion detection systems, which depend on static rules and signatures, frequently have trouble identifying new and developing threats. As a result, data mining and machine learning approaches have emerged as viable substitutes for creating robust and adaptive IDS.

Data mining has been widely used in cybersecurity to identify patterns, anomalies, and correlations in massive databases. These methods use machine learning algorithms and statistical models to accurately detect malicious activity. The CIC-IDS2017 dataset has become a well-known resource among the datasets accessible for IDS research. This dataset, which was created by the Canadian Institute for Cybersecurity, provides an accurate representation of actual network traffic, covering both malicious and benign activity. It is especially well-suited for assessing contemporary intrusion detection techniques since it encompasses a broad range of attack types, including denial-of-service (DoS), distributed denial-of-service (DDoS), brute force, penetration, and web-based attacks.

The CIC-IDS2017 dataset has been used in a number of investigations to create and evaluate machine learning-based intrusion detection systems. Promising outcomes have been shown by deep neural networks, ensemble learning strategies, and hybrid systems that combine supervised and unsupervised techniques. Despite the advancements, problems with real-time detection, lowering false positives, and managing the enormous volume of network traffic data still persist. These difficulties highlight the necessity of persistent investigation into novel strategies and careful evaluation of their effectiveness.

The main objective of this paper is to assess the effectiveness of various machine learning algorithms for intrusion detection using the CIC-IDS2017 dataset. By using feature optimization techniques to remove redundant or irrelevant attributes, the study specifically seeks to determine which features are best for detecting intrusions. This is done in order to evaluate how feature reduction affects model performance and computational efficiency. Key performance metrics like accuracy, precision, recall, and F1-score will be compared, along with processing time before and after feature optimization.

Classification algorithms from several categories will be used in the study to offer a variety of viewpoints on their applicability for this purpose. A methodical approach involving data preprocessing and feature selection will be used to train and evaluate the models. The findings will draw attention to the compromises made between computing efficiency, detection accuracy, and feature optimization.

The remainder of this paper is organized as follows: Section 2 introduces the CIC-IDS2017 dataset and reviews its relevance and prior applications in intrusion detection research. Section 3 outlines the selected algorithms and the methodological framework for this study. Section 4 presents the experimental results, including a comparative analysis of model performance and computational efficiency before and after feature optimization. Finally, Section 5 concludes the paper and suggests potential directions for future work.

2 Data

2.1 Dataset Description

The dataset that I'll analyze in the remainder of the paper is the CIC-IDS2017 (Canadian Institute for Cybersecurity Intrusion Detection System 2017) dataset [9], a widely recognized and publicly available benchmark dataset specifically designed for intrusion detection research.

The importance of CIC-IDS2017 lies in its careful planning to rectify the flaws in previous datasets and offer a more accurate depiction of contemporary network traffic and attack scenarios. The Canadian Institute for Cybersecurity established it to provide a rich groundwork for creating and assessing intrusion detection systems and algorithms. The dataset is extremely significant for training reliable and accurate IDS models because it includes both innocuous network traffic and a wide variety of modern attacks.

The richness of features in CIC-IDS2017 is one of its main advantages. It includes five full days of network traffic that has been analyzed to extract more than 80 features from network flow data. These features capture various aspects of network traffic, including packet length, duration, protocol type, flow bytes/s, flow packets/s, and many other statistical measures. The creation of advanced intrusion detection models that can accurately distinguish between hostile and benign traffic patterns is made possible by this vast feature set, which offers a comprehensive and detailed perspective of network activity.

Also, CIC-IDS2017 provides an accurate depiction of actual attack situations. It contains a range of modern attacks, including Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet and DDoS.

2.1.1 Dataset Characteristics

- **Number of Features**: The original dataset contains 84 features extracted from network traffic flows. However, after data cleaning and pre-processing, the number of features used for machine learning may be reduced. For example, one study removed some features due to miscalculation or redundancy, including 'Bwd PSH Flags', 'Bwd URG Flags', and several bulk-related features, as well as timestamp and flow identifier, using 71 features after PCA [6, 8]. Another study removed features such as the IP addresses and ports and also removed features that were always null [8].

- **Number of Instances**: The dataset initially contains a total of 2,830,743 instances of network traffic. However, after cleaning, the number of instances is reduced. One study removed instances with empty features, 'NaN', or 'Infinity' values [8]. Another study, using a corrected dataset called LYCOS-IDS2017, had 1,837,500 instances after dropping packets from Thursday afternoon and addressing TCP termination issues [8]. A subset of the dataset used for machine learning had 440,632 instances for training, 220,312 for cross-validation, and 220,312 for testing.

- **Number of Classes**: The dataset has 15 classes, consisting of one class for normal traffic (BENIGN) and 14 classes for different types of attacks. For some machine learning tasks, the classes may be reduced to a binary classification problem, with one class for normal traffic and another class combining all attacks. Another study focused on a 5 class output [6].

- **Number of Instances Per Class**: The dataset is highly imbalanced, meaning that the number of instances varies significantly between classes.
  - The BENIGN class makes up the majority of the dataset, with 2,273,097 instances in the original CIC-IDS2017 dataset and 1,395,659 instances in the LYCOS-IDS2017 dataset.
  - Some attack classes have a large number of instances, such as DDoS with 128,027 instances and DoS Hulk with 231,073 instances.
  - Other attack classes are represented by very few instances, such as Heartbleed with 11 instances.
  - The imbalance between classes poses a challenge for machine learning and may require techniques such as oversampling or undersampling to achieve optimal performance.

2.2 Prior work with CIC-IDS2017

Several research papers have used the CICIDS2017 dataset for network intrusion detection, employing various preprocessing and evaluation methods. Here's an analysis of their approaches, tasks, and methodologies:

2.2.1 Tasks and Objectives

- The primary task across the studies is to detect network intrusions and classify them as either normal or malicious traffic [11, 12]. This can involve binary classification (normal vs. attack) or multi-class classification (identifying specific types of attacks) [12, 14].
- Some studies aim to improve the detection rate of minority classes within the dataset, addressing the class imbalance problem.
- A key objective is to achieve high accuracy and low false positive rates in intrusion detection.
- Researchers also aim for real-time detection capabilities, which means developing models that are efficient in terms of time [10, 11, 13].

2.2.2 Preprocessing Methods

- **Digitizing Symbolic Features**: String data features (such as source IP, source port, destination IP, destination port, and protocol) are converted into numerical data [13].
- **Data Normalization**: Numerical data is often normalized to a standard scale, typically between 0 and 1, which helps improve model performance [12].
- **One-Hot Encoding**: This is used to convert categorical features into a numerical format, expanding the feature dimension [10].
- **Handling Missing or Infinite Values**: Some studies address issues such as "infinity" values in features like "Flow Bytes" and "Flow Packets" by replacing them with maximum values.
- **Feature Selection/Reduction**: Techniques like Principal Component Analysis (PCA) and autoencoders are used to reduce the dimensionality of the data [4, 10–12]. Deep Belief Networks (DBN) are also used for feature dimensionality reduction. Some studies also use feature selection algorithms to identify the most relevant features, for example, the "Highest Wins" (HW) algorithm and a modified binary grey wolf optimisation (MBGWO) feature selection algorithm.

2.2.3 Methodologies and Algorithms

- **Machine Learning (ML) Algorithms**
  - Various ML algorithms such as Naive Bayes [11], Support Vector Machines (SVM) [4, 13], Random Forest [11, 14], Decision Trees [11, 14], K-Nearest Neighbors (KNN) [4, 11] and Logistic Regression have been applied for intrusion detection.
  - Ensemble methods like AdaBoost are also used to enhance the classification performance [11].
- **Deep Learning (DL) Techniques**:
  - **Deep Neural Networks (DNN)**: Multilayer Perceptrons (MLP), and other fully connected neural network architectures [12, 14].
  - **Recurrent Neural Networks (RNN)**: Including Long Short-Term Memory (LSTM) and Bidirectional LSTM (Bi-LSTM) [10]. These are useful for modelling sequential data.
  - **Convolutional Neural Networks (CNN)**: Used for feature extraction and intrusion detection [4, 14].
  - **Deep Belief Networks (DBN)**: Often used for feature extraction, dimensionality reduction, and classification [4, 12, 13].
  - **Transformers**: Time-aware transformer models have been explored to improve intrusion detection performance, which capture the time dependency of network traffic [4].
- **Hybrid Approaches**:
  - Combining different ML or DL techniques in a multi-stage or ensemble approach, for example combining DBN and LSTM [4].
  - Integrating feature selection or dimensionality reduction methods with classifiers [12].
  - **Two-Phase IDS**: Some papers use a two-phase approach, where data is classified in two steps: 1) data categorisation based on data types and Naive Bayes classification, 2) unsupervised elliptic envelope method is used for normal traffic [11].
- **Class Imbalance Techniques**:
  - **SMOTE (Synthetic Minority Over-sampling Technique)**: Used to oversample minority classes to balance the data [14].
  - **Gaussian Mixture Model (GMM)**: Used for clustering based under-sampling on the majority class [14].
  - **Weight Initialization**: Some studies use weight initialization methods during model training to address the class imbalance problem [11].

2.2.4 Evaluation Metrics

- **Accuracy**: The overall correctness of classification.
- **Precision**: The ratio of true positives to all predicted positives.
- **Recall (True Positive Rate)**: The ratio of true positives to all actual positives.
- **False Positive Rate**: The ratio of false positives to all actual negatives.

- **F1-Score**: The harmonic mean of precision and recall.
- **Detection Rate (DR)**: The ability of the model to identify attack instances.
- **Confusion Matrix**: Provides a detailed breakdown of classification results (True Positives, False Positives, True Negatives, False Negatives).
- **Training and Testing Time**: Measured to assess the efficiency of the model.

2.2.5 Key Findings and Comparisons

- Studies show deep learning models often perform better than traditional machine learning methods, especially with large datasets [12, 14].
- Hybrid approaches that combine different techniques tend to yield better results [11].
- Class imbalance processing techniques are critical for improving detection of minority class attacks [14].
- Some models achieve high accuracy on the CICIDS2017 dataset such as 98.59%, 99.85% and 98.48%. However, some research does not achieve such high accuracy due to some limitations, such as not considering the temporal features of the network traffic or not taking into account all of the features in the dataset.

In summary, research using the CICIDS2017 dataset is focused on improving the accuracy and efficiency of network intrusion detection. The approaches involve careful preprocessing, feature engineering, and advanced machine learning and deep learning techniques with some novel and hybrid approaches to improve the results and address limitations.

3 Selected Algorithms and Methodological Framework

This section outlines the methodological framework adopted in this study, focusing on the preprocessing of data and the implementation of selected algorithms. The methodological design ensures a rigorous and reproducible approach to classifying network traffic using the CICIDS2017 dataset.

3.1 Preprocessing Framework

The preprocessing pipeline for this study was meticulously designed to handle the challenges posed by the CICIDS2017 dataset. These challenges include imbalanced class distributions, feature redundancy, and noisy data. A uniform preprocessing protocol was implemented for all algorithms to ensure comparability.

3.1.1 Data Cleaning

- **Zero Variance Features**: Columns exhibiting zero variance were removed as they provide no discriminatory information for classification.
- **Infinite and Missing Values**: Instances containing infinite values or missing data were identified and removed. This step ensured the integrity of the dataset and avoided computational errors during model training.

- **Duplicate Rows**: Duplicate entries were eliminated to prevent over representation of certain patterns.
- **Label Normalization**: Non-readable or inconsistent label values were standardized. For instance, labels containing non-printable characters were replaced with appropriate, readable equivalents.

3.1.2 Feature Reduction and Selection

- **Domain Knowledge-Based Reduction**: Features deemed irrelevant for network intrusion detection (e.g., ”Flow ID,” ”Source IP,” and ”Timestamp”) were removed.
- **Correlation Analysis**: Highly correlated features were identified using a Pearson correlation matrix. Features with correlation coefficients exceeding 0.95 were systematically reduced to minimize redundancy.
- **Feature Importance Evaluation**: A Random Forest classifier was employed to quantify the importance of features. Features contributing minimally to the classification task were excluded based on a predefined threshold.

3.1.3 Class Imbalance Handling

- **Under-Sampling**: To address the severe class imbalance, an initial random under-sampling strategy was applied, balancing the dataset while reducing computational overhead.
- **Over-Sampling with SMOTE**: Synthetic Minority Over-sampling Technique (SMOTE) [1] was employed during model training to generate synthetic samples for minority classes, achieving a balanced distribution. Additionally, small classes with extremely low sample sizes were handled through manual replication to ensure representation.

3.1.4 Standardization and Encoding

- Feature Standardization: A quantile transformer [7] was employed to standardize features, mapping them to a Gaussian-like distribution with zero mean and unit variance, facilitating gradient-based optimization.
- **Label Encoding**: The categorical “Label” column was transformed into a numerical format using one-hot encoding, enabling compatibility with machine learning algorithms.

3.2 Data Partitioning

The dataset was divided into training (70%), validation (15%), and test (15%) subsets. An additional split within the training set was used for hyperparameter tuning.

3.3 Selected Algorithms

This study leverages three distinct algorithms—Long Short-Term Memory (LSTM) neural networks, AdaBoost, and Naïve Bayes—for multiclass classification of the CICIDS2017 dataset. The rationale behind selecting these algorithms lies in their diverse methodological approaches and their proven effectiveness in solving classification problems in intrusion detection systems (IDS). Each algorithm is detailed below: LSTM multiclass classification guide

3.3.1 Long Short-Term Memory (LSTM) Neural Networks

LSTM is a specialized form of Recurrent Neural Network (RNN) designed to capture long-term dependencies in sequential data by addressing the vanishing gradient problem inherent in traditional RNNs. This property is particularly advantageous for analyzing temporal relationships within network traffic data, where the temporal order of events can be critical for identifying malicious patterns [2].

The model architecture consisted of stacked LSTM layers with 128 and 64 units, followed by dense layers with ReLU activation and a softmax output layer for multiclass classification. Dropout layers were incorporated to prevent overfitting.

The model was trained using backpropagation through time (BPTT) and optimized with the Adam optimizer, combining momentum and adaptive learning rates. Early stopping was employed to monitor validation loss.

3.3.2 AdaBoost

Adaptive Boosting (AdaBoost) is an ensemble learning technique that constructs a robust classifier by iteratively combining multiple weak classifiers, typically decision trees. Each weak classifier focuses on samples misclassified by its predecessor, assigning higher weights to these samples, thereby improving overall accuracy [3].

Its capacity to adjust dynamically to the dataset's complexities makes it particularly suitable for distinguishing between various types of network traffic in the multiclass CICIDS2017 dataset. Parameter tuning is conducted to optimize its performance under imbalanced class distributions.

3.3.3 Naïve Bayes

The Naïve Bayes classifier operates on the principles of Bayes' theorem, assuming conditional independence among features. Despite its simplicity, this algorithm often performs well in high-dimensional datasets [5].

The Gaussian variant of Naïve Bayes was selected for its computational efficiency and effectiveness in handling continuous features, such as those in the CICIDS2017 dataset. It serves as a benchmark for comparing the efficacy of more complex algorithms.

3.4 Experimental Protocol

The following steps outline the experimental protocol used to evaluate the algorithms:

3.4.1 Training and Validation

Each algorithm was trained on the preprocessed training set, with hyperparameter optimization performed on the validation set. LSTM models were trained on reshaped three-dimensional data to capture temporal features. Validation loss was monitored to guide training and avoid overfitting. AdaBoost and Naïve Bayes were applied directly to the training data after preprocessing, with cross-validation used where feasible to ensure robustness.

3.4.2 Performance Evaluation

AdaBoost and Nayve Bayes Models were evaluated on the test set using accuracy, precision, recall, and F1-score. LSTM Neural Network was evaluated using just accuracy initially but after the models was stored other metrics were analysed. Given the class imbalance, special emphasis was placed on recall for minority classes to assess detection efficacy. Comparative analysis was conducted to determine the strengths and weaknesses of each algorithm, providing insights into their suitability for network intrusion detection.

4 Experimental Results and Comparative Analysis

4.1 Overview of Experiments

This section presents the experimental results obtained from the three selected algorithms: Long Short-Term Memory (LSTM), AdaBoost, and Naïve Bayes. The results include evaluations of model performance, computational efficiency, and the impact of feature optimization. A comparative analysis highlights the strengths and weaknesses of each algorithm in addressing the challenges of multiclass classification in network intrusion detection.

4.2 Evaluation Metrics

The models were evaluated using the following metrics:

- **Accuracy**: Overall percentage of correctly classified samples.
- **Precision**: The proportion of true positives among predicted positives.
- **Recall**: The ability to detect true positives, particularly for minority classes.
- **F1-Score**: The harmonic mean of precision and recall, balancing both metrics.

4.3 Experimental Results

4.4 Evaluation Metrics

The evaluation metrics used to assess model performance are:

- **Accuracy:** The overall percentage of correctly classified instances.
- **Precision:** The proportion of true positives among all predicted positives.
- **Recall:** The ability to identify all true positive instances, particularly for minority classes.
- **F1-score:** The harmonic mean of precision and recall, balancing their trade-offs.

4.5 Results and Analysis

The experimental results for each model are detailed below.

4.5.1 LSTM Neural Network

- **Accuracy:** The LSTM achieved an overall accuracy of **99.7%**, indicating exceptional performance in classifying the dataset.
- **Macro Average:** A precision of **87%**, recall of **93%**, and F1-score of **88%**, highlighting strong performance even for minority classes.
- **Class-Specific Analysis:**

  - Class 1 (*Web Attack Brute Force*) showed a high recall of 93%, demonstrating the model's effectiveness in identifying these attacks, albeit with lower precision (87%).
  - Rare classes, such as Class 10, achieved mixed results, with precision of 43% and recall of 60%.

- **Strengths:** The LSTM excelled in capturing sequential dependencies, resulting in high precision and recall across most classes.
- **Weaknesses:** Computationally intensive and required substantial memory and time to train.

4.5.2 AdaBoost

- **Accuracy:** AdaBoost achieved an accuracy of **79%**, reflecting moderate overall performance.
- **Macro Average:** Precision of **23%**, recall of **17%**, and F1-score of **17%**, indicating significant struggles with minority class detection.
- **Class-Specific Analysis:**
  - Class 0 (*Benign Traffic*) was classified with high accuracy (precision: 73%, recall: 99%), contributing substantially to overall performance.
  - Minority classes, such as Class 13 (*SQL Injection*), were not detected, with precision, recall, and F1-scores of 0.
- **Strengths:** Robust to overfitting and performed well on the majority class.
- **Weaknesses:** Struggled with imbalanced data, requiring more advanced resampling or algorithmic adjustments for minority classes.

4.5.3 Naïve Bayes

- **Accuracy:** Naïve Bayes achieved an accuracy of **90%**, surpassing AdaBoost in overall classification performance.
- **Macro Average:** Precision of **75%**, recall of **91%**, and F1-score of **77%**, demonstrating better handling of minority classes than AdaBoost.
- **Class-Specific Analysis:**
  - Class 1 (*Web Attack Brute Force*) showed significant improvement in recall (100%) but suffered from low precision (5%).
  - Rare classes, such as Class 10, were identified with high precision (100%) and recall (90%).
- **Strengths:** Computationally efficient and capable of capturing patterns in high-dimensional data.
- **Weaknesses:** Assumption of feature independence limited its performance in classes with complex dependencies.

4.6 Comparative Analysis

Model	Accuracy	Macro Avg F1	Strengths/Weaknesses
LSTM	99.7%	88%	High precision, computationally expensive
AdaBoost	79%	17%	Handles majority classes well, weak on minority classes
Naïve Bayes	90%	77%	Efficient, limited by independence assumptions

Table 1: Performance Summary of Models

4.7 Feature Optimization Analysis

The feature selection process significantly influenced performance:

- **LSTM:** Reduced feature dimensions improved convergence time without compromising accuracy.
- **AdaBoost:** While computational efficiency improved, the model remained biased towards majority classes.
- **Naïve Bayes:** Feature reduction streamlined computation but had limited impact on its core assumptions.

5 Conclusion and Future Work

5.1 Conclusion

This study aimed to evaluate the performance of three distinct machine learning algorithms—Long Short-Term Memory (LSTM), AdaBoost, and Naïve Bayes—for multiclass classification of network traffic within the CICIDS2017 dataset with a focus not in the model performance but in the preprocessing task, and how this can help making better models. A robust preprocessing pipeline addressed challenges related to high dimensionality, noisy data, and class imbalance, ensuring that the models were trained on high-quality data.

The results demonstrated that LSTM outperformed the other models in terms of accuracy and recall, particularly for majority classes, due to its ability to capture sequential dependencies in network traffic. AdaBoost achieved competitive results by leveraging ensemble learning, balancing performance and computational efficiency. Naïve Bayes, though computationally lightweight, was less more effective than AdaBoost even when it struggles in handling the complexity and interdependence of features in this high-dimensional dataset.

Feature optimization, including the removal of redundant and low-variance features, played a critical role in enhancing computational efficiency and improving classification accuracy for all models. However, challenges remained in achieving consistent performance across highly imbalanced classes.

This work underscores the importance of selecting algorithmic approaches tailored to the unique characteristics of network intrusion detection tasks. By combining rigorous preprocessing with diverse machine learning techniques, this study provides insights into the trade-offs between accuracy, computational efficiency, and model complexity.

5.2 Future Work

Building on the findings of this study, future work will explore the potential of neural networks to analyze aggregated logs generated by intrusion detection systems (IDS). Aggregated logs, which consolidate and correlate alerts across multiple events, offer a rich source of temporal and contextual information that could reveal patterns of correlated attacks.

The proposed approach involves training a neural network to identify correlations between IDS alerts, aiming to detect whether seemingly separate attacks are part of a coordinated effort. By integrating features such as alert timestamps, source and destination information, and attack types, this system will provide enhanced situational awareness and facilitate the identification of advanced persistent threats (APTs) and other sophisticated attack campaigns.

Additionally, future efforts will focus on:

- **Dimensionality Reduction**: Employing advanced techniques such as Principal Component Analysis (PCA) or t-SNE to further streamline feature sets and reduce computational overhead.
- **Hybrid Models**: Developing hybrid approaches that combine the strengths of LSTM and AdaBoost, leveraging both temporal pattern recognition and ensemble learning capabilities.
- **Real-Time Detection**: Enhancing the models to operate in real-time, ensuring that network administrators can respond promptly to evolving threats.
- **Transfer Learning**: Exploring transfer learning methods to adapt models trained on the CICIDS2017 dataset to other datasets, improving generalizability and applicability across diverse network environments.

The development of systems that integrate aggregated log analysis with advanced machine learning models represents a promising avenue for improving the accuracy, efficiency, and reliability of intrusion detection systems. Such advancements will contribute significantly to proactive cybersecurity measures, enabling organizations to mitigate threats before they cause significant harm.

Code and Data Availability

The Jupyter notebooks containing the implementation and analysis discussed in this paper, along with the associated datasets and preprocessing steps, are available in the following GitHub repository:

`https://github.com/Berna-RV/CIC-IDS2017-Data-Mining`

Readers are encouraged to explore the repository for additional insights and reproducibility of the results.

References

[1] Jason Brownlee. Smote for imbalanced classification with python. `https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/`, 2021. Accessed: 2025-01-21.

[2] GeeksforGeeks. What is lstm – long short term memory? `https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/`, Updated: 2024-06-10. Accessed: 2025-01-21.

[3] GeeksforGeeks. Implementing the adaboost algorithm from scratch. `https://www.geeksforgeeks.org/implementing-the-adaboost-algorithm-from-scratch/`, Updated: 2024-07-15. Accessed: 2025-01-21.

[4] Xueying Han, Susu Cui, Song Liu, Chen Zhang, Bo Jiang, and Zhigang Lu. Network intrusion detection based on n-gram frequency and time-aware transformer. *Computers Security*, 128:103171, 2023.

[5] IBM. What are naïve bayes classifiers? `https://www.ibm.com/think/topics/naive-bayes`, n.a. Accessed: 2025-01-21.

[6] Jinsi Jose and Deepa Jose. Deep learning algorithms for intrusion detection systems in internet of things using cic-ids 2017 dataset. *International Journal of Electrical and Computer Engineering (IJECE)*, 13:1134, 02 2023.

[7] Scikit learn developers. *QuantileTransformer*, n.d. Accessed: 2025-01-21.

[8] Arnaud Rosay, Eloïse Cheval, Florent Carlier, and Pascal Leroux. Network Intrusion Detection: A Comprehensive Analysis of CIC-IDS2017. In *8th International Conference on Information Systems Security and Privacy*, pages 25–36, Online Streaming, France, February 2022. SCITEPRESS - Science and Technology Publications.

[9] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. Intrusion Detection Evaluation Dataset (CIC-IDS2017). Canadian Institute for Cybersecurity, University of New Brunswick, 2018. Accessed: 11/1/2025.

[10] S. Sivamohan, S.S. Sridhar, and S. Krishnaveni. An effective recurrent neural network (rnn) based intrusion detection via bi-directional long short-term memory. In *2021 International Conference on Intelligent Technologies (CONIT)*, pages 1–5, 2021.

[11] Monika Vishwakarma and Nishtha Kesswani. A new two-phase intrusion detection system with naïve bayes machine learning for data classification and elliptic envelop method for anomaly detection. *Decision Analytics Journal*, 7:100233, 2023.

[12] Zhendong Wang, Yong Zeng, Yaodi Liu, and Dahai Li. Deep belief network integrating improved kernel-based extreme learning machine for network intrusion detection. *IEEE Access*, 9:16062–16091, 2021.

[13] Hao Zhang, Yongdan Li, Zhihan Lv, Arun Kumar Sangaiah, and Tao Huang. A real-time and ubiquitous network attack detection based on deep belief network and support vector machine. *IEEE/CAA Journal of Automatica Sinica*, 7(3):790–799, 2020.

[14] Hongpo Zhang, Lulu Huang, Chase Q. Wu, and Zhanbo Li. An effective convolutional neural network based on smote and gaussian mixture model for intrusion detection in imbalanced dataset. *Computer Networks*, 177:107315, 2020.