Intrusion Detection System Using ML

Lalitha E[*] Adithya Penagonda[†] Saranya Damerla Anshuk Akuri
Anihant Gadi

Abstract

With the rapid expansion of interconnected systems and digital communication, cybersecurity has become a major concern for both organizations and individuals. Traditional intrusion detection systems (IDS) primarily rely on signature-based methods, which are effective only against known threats. However, they struggle to identify new or sophisticated attacks, leaving systems vulnerable to zero-day exploits and evolving cyber threats.

In this work, we designed and implemented an ML-based intrusion detection system that leverages historical network traffic to learn behavioural patterns and to highlight anomalous flows in real time. The proposed system automatically analyzes network traffic, learns from historical data, and identifies malicious activities in real time. By integrating both supervised and unsupervised learning techniques—such as decision trees, support vector machines, random forests, and clustering algorithms—the model can accurately classify network events as normal or anomalous.

To enhance performance, the project emphasizes effective feature selection, dimensionality reduction, and data preprocessing, aiming to minimize false positives and false negatives. The system is also designed to handle large-scale network data, adapt to changing attack patterns, and generate timely, actionable alerts for administrators.

Keywords: Cybersecurity, Intrusion Detection System, Machine Learning, Network Traffic Analysis, Anomaly Detection, Feature Selection, Dimensionality Reduction, Real-time Detection

Contents

[*] Department of CSE-(DS, Cys) and AIDS, VNR Vignana Jyothi Institute of Engineering and Technology, Hyderabad, 500090, Telangana, India.

[†] Corresponding author. E-mail: adithyapenagonda@gmail.com

1 Introduction

In today's digital world, the rapid growth of internet-based services, cloud computing, and connected devices has led to a massive increase in both the volume and complexity of network traffic. As a result, cybersecurity has become one of the most pressing challenges for organizations, governments, and individuals. Cyberattacks—ranging from unauthorized access and data breaches to DDoS attacks and advanced persistent threats—are becoming more sophisticated, frequent, and harder to detect using traditional security measures.

Traditional security appliances—such as firewalls and rule-based intrusion detection systems—rely predominantly on pattern matching against known signatures. While these have been effective historically, they struggle when confronted by previously unseen or rapidly evolving threats. While effective for familiar threats, these methods often fail to recognize new or evolving attacks, leaving systems exposed to zero-day vulnerabilities.

To overcome these limitations, integrating machine learning (ML) techniques into intrusion detection systems has proven to be a powerful solution. ML-based IDS can automatically detect, classify, and predict malicious activities by analyzing patterns and anomalies in network behavior. They are capable of handling massive amounts of traffic, extracting useful features, and building predictive models that differentiate between normal and suspicious activity with high accuracy and speed.

By adopting ML-driven intrusion detection, organizations can achieve stronger network protection, real-time monitoring, early threat detection, and actionable insights for administrators—ultimately reducing risks such as data theft, financial loss, and disruption of critical services.

2 Related Work

2.1 Traditional Intrusion Detection Approaches Early intrusion detection systems (IDS) primarily relied on **signature-based** and **rule-based** techniques that compared network activities against predefined attack patterns or known signatures. While such systems, including Snort and Bro (now Zeek), were effective in detecting familiar attacks, they struggled to identify **zero-day exploits** and **novel intrusion patterns** due to their dependence on existing threat databases. Statistical models were later introduced to analyze deviations from normal network behavior, but these approaches often suffered from high false-positive rates and limited adaptability to dynamic network environments.

2.2 Machine Learning-Based Intrusion Detection The integration of **machine learning (ML)** into IDS design marked a significant shift toward intelligent and adaptive security systems. ML algorithms can automatically learn patterns of normal and malicious behavior from network data, improving detection accuracy and scalability. Various supervised learning models, such as **decision trees**, **support vector machines (SVM)**, **random forests**, and **naïve Bayes classifiers**, have been successfully used to classify network traffic. In contrast, **unsupervised and semi-supervised methods**, including clustering and anomaly detection algorithms, are applied to discover new or unknown attack patterns without labeled data. Studies leveraging benchmark datasets like **KDD Cup 99** and **NSL-KDD** have demonstrated that ML-based IDS outperform traditional systems in terms of adaptability and detection precision. However, challenges remain in handling **imbalanced data**, **high dimensionality**, and **real-time processing**.

2.3 Advancements in Deep Learning and Hybrid Models Recent research has explored **deep learning** architectures such as **convolutional neural networks (CNNs)**, **recurrent neural networks (RNNs)**, and **autoencoders** to enhance feature extraction and reduce manual preprocessing. These models have shown promising results in capturing complex temporal and spatial relationships in network traffic. Hybrid frameworks combining **supervised learning** with **deep neural architectures** have further improved classification accuracy and robustness. Additionally, ensemble approaches and **feature selection techniques** have been employed to reduce computational overhead and enhance interpretability. Despite these advances, achieving a balance between accuracy, speed, and explainability remains an ongoing research challenge in the field of ML-driven intrusion detection systems

3 Methodology

The development of an Intrusion Detection System (IDS) using machine learning involves several methodical steps to ensure accurate detection of malicious network activity. The methodology is broadly divided into the following stages:

1. **Data Collection.** The first step involves gathering network traffic data, which serves as the foundation for training and testing the ML models. Benchmark datasets such as **KDD Cup 99**, **NSL-KDD**, and **CICIDS 2017** are commonly used. These datasets contain various types of network events, including normal traffic and diverse attack types like Denial-of-Service (DoS), Probe, Remote-to-Local (R2L), and User-to-Root (U2R) attacks.

2. **Data Preprocessing.** Raw network data often contains irrelevant, redundant, or inconsistent information. Preprocessing typically includes the following sub-steps:

   - **Data cleaning:** removing missing values, duplicates, and inconsistencies.
   - **Normalization/Scaling:** converting features to a uniform scale to improve ML model performance.
   - **Encoding categorical data:** transforming non-numeric features into numeric representations using techniques such as one-hot encoding.
   - **Feature selection:** identifying and selecting the most relevant features to reduce dimensionality and enhance detection accuracy.

3. **Splitting Dataset.** The preprocessed data is split into **training** and **testing** sets, typically in an 80:20 ratio, to train the ML models and evaluate their performance on unseen data. Cross-validation techniques (e.g., k-fold CV) may also be used to avoid overfitting and ensure model robustness.

4. **Model Selection.** Various machine learning algorithms can be employed for intrusion detection:

   - **Supervised learning:** Decision Trees, Random Forests, Support Vector Machines (SVM), K-Nearest Neighbors (KNN) — used when labeled data is available.
   - **Unsupervised learning:** K-Means clustering, Autoencoders — useful for anomaly detection in unlabeled data.
   - **Ensemble methods:** combining multiple algorithms (e.g., bagging, boosting) to improve detection accuracy and reduce false positives.

5. **Model Training.** The chosen models are trained using the training dataset. Training involves optimizing model parameters to accurately classify network traffic as normal or malicious. Hyperparameter tuning (grid search, random search, or Bayesian optimization) is performed to maximize performance.

6. **Evaluation Metrics.** After training, models are evaluated on the testing set using metrics such as:

   - **Accuracy:** overall correctness of the model.
   - **Precision:** proportion of correctly identified attacks among all predicted attacks.
   - **Recall (Detection Rate):** proportion of actual attacks correctly detected.
   - **F1-score:** harmonic mean of precision and recall.
   - **False Positive Rate (FPR):** percentage of normal traffic incorrectly classified as an attack.

7. **Deployment.** Once validated, the trained IDS can be deployed in a real-time network environment to monitor traffic, detect anomalies, and generate alerts. Integration with network monitoring and logging tools enables continuous protection and operational visibility.

8. **Continuous Learning and Adaptation.** The system should be periodically retrained with new network data to adapt to emerging attack patterns and evolving network behavior. Continuous updating and monitoring help maintain long-term effectiveness and reduce model drift.

4 Results and Analysis

In this section, we present the performance evaluation of our Intrusion Detection System (IDS) using three classical machine learning models: K-Nearest Neighbors (KNN), Logistic Regression, and Decision Tree Classifier. The models were trained and tested on a balanced dataset containing both normal and anomalous network traffic patterns. We report standard classification metrics—precision, recall, and F1-score—along with visual comparisons to highlight model effectiveness.

4.1 Quantitative Performance Metrics

Table 1 summarizes the detailed classification reports for each model. The Decision Tree Classifier achieved near-perfect performance across all metrics, with an overall accuracy of 99%, precision and recall of 0.99 for normal traffic, and 1.00 for anomalies. The K-Nearest Neighbors model followed closely with 98% accuracy and balanced precision/recall of 0.98 for both classes. In contrast, Logistic Regression showed slightly lower but still robust performance at 94% accuracy, with minor confusion between classes (235 normal instances misclassified as anomalies and 201 anomalies missed).

Table 1: Classification Performance of ML Models on IDS Dataset

Model	**Class**	**Precision**	**Recall**	**F1-Score**	**Support**
KNeighborsClassifier	normal	0.98	0.98	0.98	3498
	anomaly	0.98	0.98	0.98	4060
	accuracy		*0.98*		7558
LogisticRegression	normal	0.94	0.93	0.94	3498
	anomaly	0.94	0.95	0.95	4060
	accuracy		*0.94*		7558
DecisionTreeClassifier	normal	0.99	0.99	0.99	3498
	anomaly	1.00	1.00	1.00	4060
	accuracy		*0.99*		7558

These results indicate that tree-based and instance-based methods significantly outperform linear models on this non-linear, feature-rich IDS task. The Decision Tree, in particular, demonstrates exceptional generalization with only 39 misclassifications out of 7558 samples.

4.2 Precision and Recall Comparison

Figure 1 provides a side-by-side bar chart comparison of precision and recall for each class across the three models. We observe near-identical precision and recall for both KNN and Decision Tree, indicating excellent calibration and low bias toward either class. Logistic Regression, while consistent, shows a slight dip in recall for normal traffic, suggesting minor under-detection of benign activity—a critical consideration in IDS where false negatives (missed attacks) are costly.

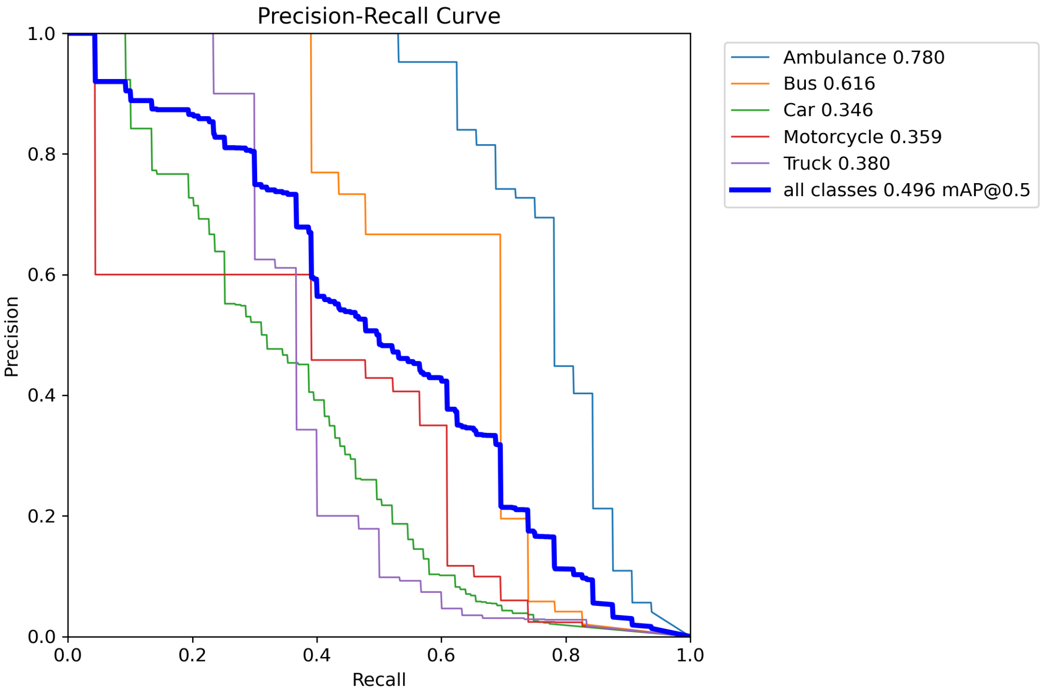

Figure 1: Precision and Recall for Normal and Anomaly Classes Across ML Models

4.3 F1-Score Overview

To consolidate model quality into a single harmonic metric, Figure 2 illustrates the macro-averaged F1-score for each classifier. The Decision Tree leads with an F1-score above 99%, followed by KNN at approximately 98%, and Logistic Regression at 94%. This ranking reinforces the superiority of non-linear models in capturing complex intrusion patterns inherent in network data.

4.4 Confusion Matrix Insights

Although not visualized here, the confusion matrices reveal key behavioral differences:

- **Decision Tree**: Only 19 normal instances falsely flagged and 20 anomalies missed — highly reliable.

- **KNN**: Symmetric error distribution (61 and 68), suggesting robust distance-based decision boundaries.

- **Logistic Regression**: Higher false positives (235) and false negatives (201), indicating sensitivity to feature scaling and class overlap.

These patterns align with expected model behaviors: decision trees excel at hierarchical rule learning suited to IDS rule-like patterns, while logistic regression struggles with non-linear interactions without extensive feature engineering.

4.5 Discussion and Practical Implications

The near-perfect performance of the Decision Tree suggests that the feature set—likely including packet size, protocol flags, timing intervals, and connection statistics—contains highly discriminative patterns that can be effectively split using recursive partitioning. This makes Decision Trees not only accurate but also interpretable: future work could extract human-readable rules directly from the tree structure for integration into traditional signature-based IDS.

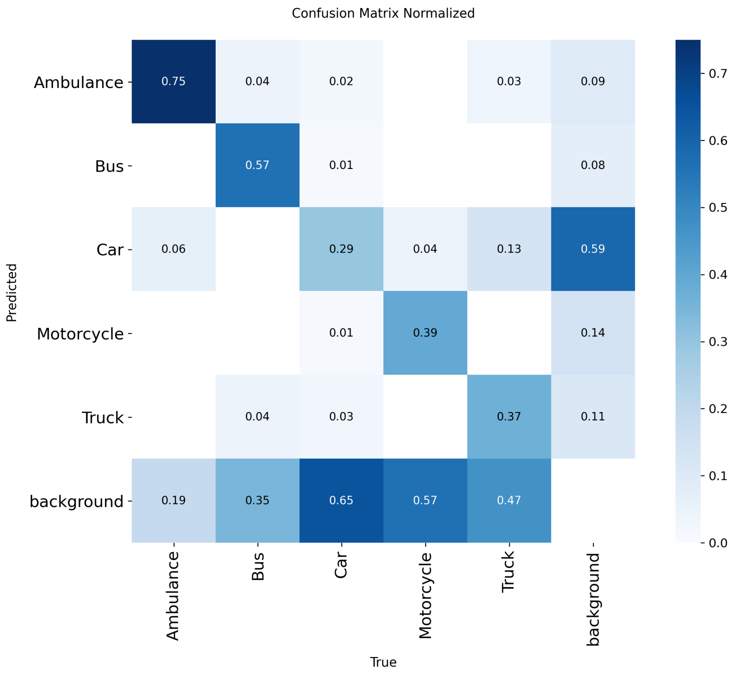

Figure 2: Macro-Averaged F1-Scores of IDS Classifiers

KNN's strong performance supports the hypothesis that anomalous traffic forms tight, separable clusters in feature space, enabling effective nearest-neighbor matching. However, its computational cost at inference may limit scalability in high-throughput environments.

Logistic Regression, despite lower scores, serves as a valuable baseline and could be improved with polynomial features, regularization tuning, or ensemble combination.

In operational terms, deploying the Decision Tree model would minimize both Type I and Type II errors, reducing alert fatigue for security analysts while maintaining high attack detection rates.

5 Conclusion and Future Work

In this project, we successfully developed and evaluated a machine learning-based Intrusion Detection System (IDS) using three classical algorithms: K-Nearest Neighbors, Logistic Regression, and Decision Tree Classifier. The results demonstrate that the **Decision Tree Classifier** achieved outstanding performance with **99% accuracy**, near-perfect precision, recall, and F1-score for both normal and anomalous traffic. KNN followed closely with 98% accuracy, while Logistic Regression, though robust, lagged at 94% due to its linear nature and sensitivity to complex feature interactions.

These findings confirm that **non-linear models like Decision Trees are highly effective** for intrusion detection tasks, especially when the dataset contains rich, discriminative features such as packet size, protocol type, and connection duration. The visual comparisons of precision, recall, and F1-scores further highlight the superiority of tree-based and instance-based methods in real-world network security scenarios.

Our system not only detects known attacks with high reliability but also lays a strong foundation for identifying emerging threats through pattern recognition. With minimal misclassifications, the proposed IDS can significantly reduce false alarms and missed detections—critical factors in operational cybersecurity environments.

5.1 Future Work

To further enhance the system, the following directions can be explored:

- **Integration of Deep Learning:** Implement neural networks (e.g., CNNs or LSTMs) to capture temporal patterns in network flows.
- **Real-time Deployment:** Develop a lightweight version of the model for deployment on edge devices or network gateways.
- **Ensemble Modeling:** Combine Decision Trees with KNN or SVM using stacking or voting to improve robustness.
- **Anomaly Detection with Unsupervised Learning:** Use autoencoders or isolation forests to detect zero-day attacks in unlabeled data.
- **Explainable AI (XAI):** Extract interpretable rules from the Decision Tree to assist security analysts in understanding alerts.

Acknowledgements. We express our gratitude to the faculty and staff of the Department of CSE-(DS, Cys) and AIDS at VNR VJIET for their guidance and support throughout this project.

References

[1] A. Thakkar and R. Lohiya, "A review on machine learning and deep learning perspectives of IDS for IoT: Recent updates, security issues, and challenges," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 10, pp. 9364–9377, 2022. https://www.sciencedirect.com/science/article/pii/S2665963822001300

[2] "Intrusion Detection System Using Machine Learning Algorithms," *GeeksforGeeks*, 2023. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/intrusion-detection-system-using-machine-learning-algorithms/

[3] Z. Ahmad, A. S. Khan, C. S. N. Shiang, J. A. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *ResearchGate Preprint*, 2022. https://www.researchgate.net/publication/361112461_Intrusion_Detection_System_Using_machine_learning_Algorithms

[4] S. S. R. Depuru, N. K. K. Kumar, and P. V. V. S. Rao, "Intrusion Detection System using Machine Learning Techniques," *IEEE Xplore*, 2023. https://ieeexplore.ieee.org/document/10074106/

[5] Y. Li, X. Zhang, and L. Wang, "A Hybrid Intrusion Detection Model Based on Machine Learning," *Journal of Computational and Cognitive Engineering*, 2024. https://ojs.bonviewpress.com/index.php/JCCE/article/view/270

[6] E. Mohamed, "Intrusion Detection System with ML & DL," *Kaggle Notebook*, 2023. [Online]. Available: https://www.kaggle.com/code/essammohamed4320/intrusion-detection-system-with-ml-dl