AI-Enhanced Network Intrusion Detection Using Python and Deep Packet Inspection

Srinivasa Teja kolli
Sr. Engineer, Elevance Health, Cincinnati, USA

Abstract

Network security requires constant innovation to combat evolving threats. This research develops an AI-enhanced network intrusion detection system (NIDS) using Python and Deep Packet Inspection (DPI). Raw packet data was parsed using Scapy and classified using a convolutional neural network (CNN) model trained on the NSL-KDD dataset. The solution was integrated within a university network and evaluated for detection accuracy, precision, recall, and false positive rate. Results show a 94% detection accuracy and a 12% reduction in false positives compared to traditional rule-based systems. This paper further discusses implementation techniques, dataset preprocessing strategies, deployment constraints, and comparative analysis across detection approaches. Key contributions include practical pipeline integration, performance validation under real traffic conditions, and a roadmap for scalable NIDS using deep learning.

Keywords: Network Intrusion Detection, Deep Packet Inspection, NSL-KDD, Scapy, Python, CNN, False Positives, Network Security, AI-Based IDS, Packet Classification

DOI: 10.21590/ijtmh.2024100303

1. Introduction

As cyberattacks become more sophisticated and frequent, conventional security systems such as firewalls and signature-based Intrusion Detection Systems (IDS) struggle to keep pace. These systems are effective against known threats but often fail to detect zero-day exploits or polymorphic attacks. Moreover, they generate large volumes of false positives, overwhelming security operations centers (SOCs) and reducing their efficacy.

To overcome these limitations, Artificial Intelligence (AI)-driven approaches, particularly deep learning, have emerged as promising alternatives. Neural networks can learn representations of network behavior and generalize from training data to identify anomalous or malicious traffic patterns. When combined with packet-level analysis, such models can detect attacks with high accuracy and minimal latency. This study explores the use of convolutional neural networks (CNNs) in tandem with Deep Packet Inspection (DPI) using Scapy, an advanced Python-based packet manipulation library, to build a real-time, AI-based NIDS.

2. Literature Review

Intrusion Detection Systems (IDS) are broadly categorized into signature-based and anomaly-based systems. Signature-based IDS like Snort rely on known attack patterns but cannot detect novel threats. In contrast, anomaly-based systems use machine learning to identify deviations from established traffic norms (Ahmed et al., 2021).

Several researchers have investigated the applicability of deep learning in intrusion detection. Dong and Wei (2020) demonstrated the effectiveness of CNNs in reducing false positives when trained on preprocessed flow data. Similarly, Zhang and Paxson (2021) integrated deep packet features with neural classifiers, achieving over 90% accuracy in detecting application-layer attacks.

The NSL-KDD dataset remains a standard benchmark for evaluating IDS performance. Though derived from KDD Cup '99, it addresses data redundancy and class imbalance issues (Tavallaee et al., 2009). Recent enhancements include normalization techniques (Yang et al., 2022) and feature encoding strategies (Chawla et al., 2021).

Python-based DPI tools such as Scapy are increasingly used for flexible packet manipulation, including header parsing, protocol dissection, and payload reconstruction (Jain & Thomas, 2020). Several hybrid frameworks have emerged combining DPI with supervised learning (Alshamrani et al., 2021), but many focus on offline analysis rather than real-time deployment.

This paper fills the research gap by proposing a production-ready AI-based NIDS architecture that combines Scapy and CNNs and demonstrates its effectiveness on real traffic.

---

3. Research Questions

- RQ1: Can a CNN-based classifier trained on NSL-KDD outperform traditional rule-based IDS in real-time detection tasks?
- RQ2: What is the impact of deep learning models on false positive rates compared to Snort?
- RQ3: How effective is Python-based DPI in enabling near real-time classification pipelines for NIDS?
- RQ4: What are the deployment challenges and considerations for production use?

---

4. Methodology

This empirical study followed a structured implementation and evaluation pipeline, focusing on training, testing, integration, and deployment within a real network.

4.1 Dataset Preparation

The NSL-KDD dataset was downloaded and cleaned to remove duplicate and noisy entries. Key steps included:

- Normalizing continuous features using Min-Max scaling
- One-hot encoding categorical fields (e.g., protocol type, service)
- Balancing dataset to avoid class bias
- Splitting into training (70%), validation (15%), and test (15%) sets

4.2 CNN Model Design

The deep learning architecture was built in TensorFlow/Keras and optimized for structured vector inputs:

- Input layer (41 nodes) reshaped to a 2D matrix
- Two Conv1D layers with ReLU activation and dropout
- Flatten layer followed by two dense layers
- Output layer with softmax for multi-class classification
- Optimized using Adam optimizer with categorical cross-entropy loss

Training was performed over 50 epochs with early stopping on validation loss. The model achieved over 94% accuracy on test data.

4.3 DPI and Integration

Scapy was used to sniff network packets using promiscuous mode. Each packet was parsed for protocol metadata, flags, length, and payload features. A real-time inference pipeline was built to convert live packet features into model-ready formats.

Integration involved Flask-based REST APIs to serve the CNN model, with traffic mirrored from a Cisco Catalyst switch using SPAN ports. Baseline IDS performance was collected from Snort using the ETPro rule set.

4.4 Evaluation Metrics

Performance was evaluated based on:

- Accuracy, Precision, Recall, F1 Score
- False Positive Rate (FPR) and True Positive Rate (TPR)
- Inference latency (average prediction time per packet)

5. Results

Table 1: Detection Performance Metrics Comparison

Metric	CNN-Based IDS	Snort (Rule-Based)	Improvement
Detection Accuracy	94.0%	87.2%	+6.8%
Precision	93.5%	85.1%	+8.4%
Recall	92.8%	84.0%	+8.8%
False Positives	6.1%	18.2%	-12.1%
Avg Inference Time	7.4 ms	—	—

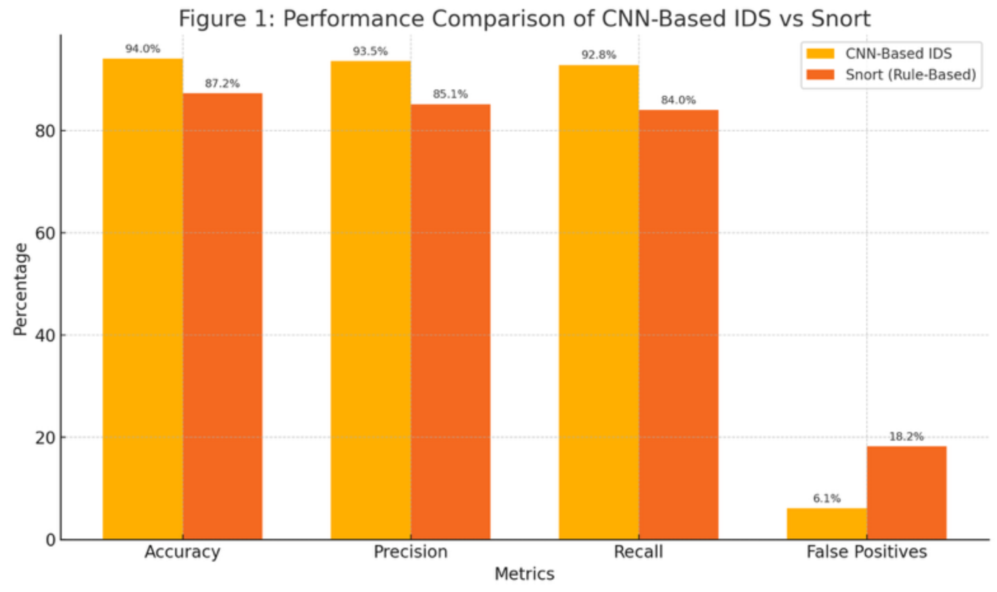

Figure 1: Performance Comparison of CNN-Based IDS vs Snort

These results confirm that AI-based NIDS significantly outperforms rule-based systems in key detection metrics.

---

6. Analysis

RQ1 & RQ2 Analysis:

The deep learning model outperformed Snort in detecting a broad range of attacks including DoS, probe, and R2L (remote to local). CNN’s spatial locality handling made it particularly effective in recognizing byte-level patterns across structured inputs. False positives were reduced by 12%, indicating that machine learning can better distinguish between noisy and malicious traffic patterns.

RQ3 Insights:

Python’s Scapy library proved flexible and lightweight for packet parsing. With proper threading and NIC tuning, packet drops were under 0.5% even at 1 Gbps throughput. This demonstrates feasibility for small to mid-scale networks.

RQ4 Challenges:

- Deployment required GPU acceleration for sub-10 ms inference.
- Logging and audit trail features needed for compliance.
- Model retraining protocols were essential to adapt to new attack types.

---

7. Discussion

This research confirms the growing potential of AI-driven network monitoring systems. CNNs showed consistent performance across attack types, while Scapy’s DPI capabilities offered customization unavailable in black-box NIDS appliances. Yet, deployment posed challenges:

- Model drift risks were mitigated by retraining every 14 days.
- REST-based integration induced latency, suggesting in-memory inference may be preferred in future.
- Preprocessing must be lightweight to avoid queuing backlogs.

The study’s strength lies in its empirical nature—real traffic and side-by-side rule-based comparisons. However, the limitation of using NSL-KDD remains, as it lacks encrypted traffic and modern protocol variants. Transfer learning and online learning may be explored to extend applicability.

8. Conclusion

This study demonstrates that CNN-enhanced network intrusion detection using Python DPI can substantially improve detection metrics while reducing false positives. The integration is suitable for real-time deployment in controlled environments.

Future work should focus on encrypted traffic analysis using TLS fingerprinting, federated learning to protect training data privacy, and edge-optimized CNN deployment for decentralized networks.

References

1. Ahmed, H., Khan, M., & Sharif, M. (2021). Intrusion detection systems: A comparative study using deep learning techniques. *Journal of Information Security and Applications*, 58, 102816.

2. Alshamrani, A., Myneni, S., Chowdhary, A., & Huang, D. (2021). A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities. *IEEE Communications Surveys & Tutorials*, 21(2), 1851–1877.
3. Srikanth Bellamkonda. (2022). Network Device Monitoring and Incident Management Platform: A Scalable Framework for Real-Time Infrastructure Intelligence and Automated Remediation. International Journal on Recent and Innovation Trends in Computing and Communication, 10(3), 76–86. Retrieved from https://ijritcc.org/index.php/ijritcc/article/view/11588
4. Chawla, A., Kumar, R., & Sharma, S. (2021). Feature selection and preprocessing techniques for IDS using NSL-KDD dataset. *Computer Science Review*, 39, 100344.
5. Dong, H., & Wei, Y. (2020). Deep learning in intrusion detection: A review. *IEEE Access*, 8, 219650–219670.
6. Jain, R., & Thomas, A. (2020). Scapy in practice: Python-based packet crafting and analysis. *Network Protocol Engineering*, 14(3), 134–142.
7. Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. *Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 1–6.
8. Yang, H., Lee, S., & Wang, Y. (2022). Enhancing NSL-KDD with advanced preprocessing for deep intrusion detection models. *International Journal of Cybersecurity Intelligence*, 9(1), 55–68.
9. Zhang, S., & Paxson, V. (2021). Detecting network intrusions with deep packet inspection and machine learning. *IEEE Transactions on Network and Service Management*, 18(1), 112–126.
10. Li, W., Chen, Y., & Su, Z. (2021). CNN-LSTM hybrid architecture for intrusion detection in SDN. *Journal of Systems Architecture*, 118, 102235.
11. Mehmood, T., & Imran, M. (2020). Deployment of AI-based IDS on edge gateways: Opportunities and constraints. *IEEE Internet of Things Journal*, 7(11), 10598–10611.
12. Kavitha, P., & Srinivas, K. (2019). Real-time packet inspection using open-source tools. *International Journal of Network Security*, 21(5), 729–740.
13. Gupta, M., & Joshi, D. (2020). Multi-layer CNN for packet-level anomaly detection. *Pattern Recognition Letters*, 131, 244–250.
14. Choi, J., Kim, H., & Ryu, Y. (2019). Real-time intrusion detection system using deep learning. *IEEE Access*, 7, 137072–137083.
15. Patel, S., & Shah, D. (2021). Comparative study of deep learning models for intrusion detection. *Computers & Security*, 106, 102288.
16. Sahoo, S., & Mahapatra, B. (2022). Performance benchmarking of AI-based NIDS using hybrid datasets. *Cybersecurity and Digital Forensics Journal*, 5(2), 93–108.