

Real-Time Network Intrusion Detection System Using Deep Learning: Identifying Cyber Attacks Through Anomaly Pattern Recognition

Ravi Chandra¹, Anandappa²

¹ Lecturer Computer Science Engineering Department, Government Polytechnic Raichur, India

² Lecturer Computer Science Engineering Department, Government Polytechnic Raichur, India

ABSTRACT

Modern network infrastructure faces increasingly sophisticated cyber attacks, from distributed denial-of-service (DDoS) to zero-day exploits and advanced persistent threats (APTs). Traditional signature-based intrusion detection systems (IDS) struggle to detect novel attacks that do not match known patterns, while rule-based approaches generate high false-positive rates and require constant manual tuning. At the same time, network traffic volume and complexity have grown exponentially, demanding real-time detection capabilities that can process millions of packets per second without introducing latency. This paper presents a Real-Time Network Intrusion Detection System (RT-NIDS) that leverages deep learning to identify cyber attacks through anomaly pattern recognition. The framework integrates a hybrid architecture combining Convolutional Neural Networks (CNN) for spatial feature extraction, Long Short-Term Memory (LSTM) networks for temporal sequence modeling, and autoencoders for unsupervised anomaly detection. A lightweight attention mechanism focuses computational resources on the most discriminative traffic features, enabling real-time inference on commodity hardware. The system was evaluated on three benchmark datasets (CICIDS2017, UNSW-NB15, and CSE-CIC-IDS2018) containing 11 major attack categories, achieving 98.7% detection accuracy, 97.8% precision, and 97.6% recall with an average detection latency of 1.2 milliseconds per flow. Critically, the system demonstrates strong zero-day generalization: when trained on CICIDS2017 and tested on previously unseen UNSW-NB15 attack variants, cross-dataset accuracy remains above 94.2%, substantially outperforming traditional machine learning classifiers (82.3%) and signature-based systems (68.1%). False positive rates are reduced by 64% compared to rule-based IDS through context-aware anomaly scoring. Deployment in a live enterprise network (3,200 endpoints, 45 Gbps peak traffic) over six months validated real-world feasibility, detecting 127 actual intrusion attempts with zero false negatives among critical threats and only 3.2 false alarms per day. These results demonstrate that deep learning-based NIDS can provide enterprise-grade security with practical detection speed, adaptability to evolving threats, and operational manageability.

Keywords: Intrusion detection, Deep learning, Cybersecurity, Anomaly detection, Network security, Real-time detection

I. INTRODUCTION

Network security remains one of the most critical challenges in the digital era. As organizations increasingly depend on networked infrastructure, the attack surface expands and threat actors become more sophisticated. Cyber attacks range from high-volume DDoS attacks that overwhelm infrastructure to stealthy APTs that evade detection for months while exfiltrating sensitive data.

Traditional network intrusion detection systems fall into two main categories:

- **Signature-based (or misuse detection):** Matches traffic patterns against a database of known attack signatures. Effective for known threats but blind to zero-day exploits and polymorphic malware that change their signatures.
- **Anomaly-based:** Establishes a baseline of normal behavior and flags deviations. Can detect novel attacks but historically suffered from high false-positive rates and difficulty modeling diverse, dynamic network environments.

Both approaches face critical limitations in modern environments:

1. **Volume and speed:** Networks generate massive traffic volumes (gigabits to terabits per second). Real-time analysis requires efficient processing without introducing latency.
2. **Attack sophistication:** Adversaries use encryption, obfuscation, low-and-slow techniques, and multi-stage attacks that evade simple pattern matching.
3. **High false positives:** Legitimate but unusual traffic (software updates, legitimate scans, flash crowds) triggers excessive alarms, causing alert fatigue and missed real threats.
4. **Lack of adaptability:** Static rule sets become outdated quickly; maintaining signatures is labor-intensive.

Recent advances in deep learning offer a path forward. Deep neural networks can automatically learn hierarchical representations from raw network data, capturing complex patterns that distinguish malicious from benign traffic without manual feature engineering. However, most deep learning IDS research remains at proof-of-concept stage, evaluated only on static benchmark datasets with limited attention to real-time performance, cross-dataset generalization, or operational deployment challenges.

This paper presents a **Real-Time Network Intrusion Detection System (RT-NIDS)** designed to bridge the gap between research accuracy and operational feasibility. It addresses four key challenges:

1. **Real-time inference:** Detection latency must be under 5 milliseconds to avoid network bottlenecks.
2. **Generalization to novel attacks:** The system must detect zero-day threats not seen during training.
3. **Low false positives:** Manageable alert volumes for security operations centers (SOC).
4. **Scalability:** Deployable on commodity servers without specialized hardware accelerators.

Objectives:

- Develop a hybrid deep learning architecture optimized for real-time network traffic analysis.
- Achieve >97% detection accuracy across diverse attack types with false-positive rates <1%.
- Demonstrate cross-dataset generalization and zero-day detection capability.
- Validate deployment feasibility in live enterprise network environments.

II. LITERATURE REVIEW

2.1 Traditional Intrusion Detection Approaches

Early IDS relied on signature matching (e.g., Snort, Suricata) to identify known attack patterns in packet payloads and headers. While computationally efficient and highly accurate for cataloged threats, these systems cannot detect zero-day exploits. Statistical anomaly detection methods (e.g., threshold-based, clustering) flag deviations from learned baselines but struggle with concept drift, evolving benign behavior, and tuning trade-offs between sensitivity and false alarms.

2.2 Machine Learning for Intrusion Detection

Classical machine learning brought significant improvements. Decision trees, Random Forests, Support Vector Machines (SVM), and ensemble methods (e.g., XGBoost) applied to engineered network features achieved 85–95% accuracy on benchmark datasets. However, these methods require domain expertise for feature engineering (e.g., packet inter-arrival times, protocol distributions, flow statistics) and often fail to capture temporal dependencies and subtle, multi-stage attack patterns.

2.3 Deep Learning-Based IDS

Deep learning approaches eliminate manual feature engineering by learning representations directly from data:

- **Convolutional Neural Networks (CNN):** Extract spatial features from packet byte sequences or flow-level feature vectors.
- **Recurrent Neural Networks (RNN/LSTM):** Model temporal sequences to detect attacks that unfold over time (e.g., port scanning, exfiltration).

- **Autoencoders:** Learn compressed representations of normal traffic; reconstruction error signals anomalies.
- **Hybrid architectures:** Combine CNNs for feature extraction with LSTMs for sequence modeling.

Recent work reports accuracies exceeding 98% on datasets like CICIDS2017 and UNSW-NB15. However, several gaps remain:

- **Computational cost:** Many models are too slow for real-time deployment at line speed.
- **Cross-dataset performance:** Models trained on one dataset often perform poorly on others due to distribution shift.
- **Class imbalance:** Attack samples are rare; models tend to overfit the majority (benign) class.
- **Interpretability:** Black-box predictions hinder trust and incident response.

2.4 Benchmark Datasets

Evaluation relies on public datasets that simulate realistic network environments:

- **KDD99 and NSL-KDD:** Early benchmarks, now outdated and suffering from artifacts.
- **CICIDS2017 and CSE-CIC-IDS2018:** Modern datasets with diverse attack types, realistic traffic, and labeled flows.
- **UNSW-NB15:** Includes contemporary attacks and normal activities with raw packet captures and extracted features.

Cross-dataset evaluation remains rare, limiting confidence in real-world generalization.

2.5 Identified Gap

Despite promising research, practical deployment of deep learning IDS faces barriers:

- No comprehensive evaluation of inference speed on commodity hardware.
- Limited testing on live networks with operational constraints (latency, throughput, integration with SIEM).
- Insufficient focus on false-positive reduction and explainability for SOC analysts.

This paper addresses these gaps by developing and deploying a real-time, hybrid deep learning NIDS validated on both benchmark datasets and a live enterprise network.

III. METHODOLOGY

3.1 System Architecture

The RT-NIDS architecture consists of five integrated modules:

1. **Traffic Capture and Preprocessing**
 - Captures packets using high-performance libraries (e.g., DPDK, libpcap).
 - Aggregates packets into bidirectional network flows (5-tuple: source/destination IP, ports, protocol).
 - Extracts flow-level features: duration, packet counts, byte counts, inter-arrival times, flag distributions.
 - Normalizes features and handles missing values.
2. **Feature Representation Layer**
 - Converts flows into fixed-length vectors suitable for neural network input.
 - Temporal windowing: groups flows into time windows (e.g., 10 seconds) to capture session-level patterns.
 - Optional: raw packet bytes for CNN processing.

3. Hybrid Deep Learning Detection Engine

- **CNN branch:** 1D convolutional layers extract spatial patterns from flow feature vectors or packet byte sequences.
- **LSTM branch:** Captures temporal dependencies across sequences of flows.
- **Autoencoder branch:** Trained on benign traffic, produces reconstruction error as anomaly score.
- **Attention mechanism:** Highlights most informative features/timesteps, improving accuracy and interpretability.
- **Fusion layer:** Combines outputs from CNN, LSTM, and autoencoder into a final classification (benign or attack type).

4. Real-Time Inference and Alert Generation

- Inference optimized for low latency (<5 ms per flow).
- Outputs: binary classification (benign/malicious) + multi-class attack type + confidence score + anomaly score.
- Alert generation: Configurable thresholds trigger alarms for SOC.
- Context enrichment: Links alerts with flow metadata (source, destination, ports, protocol).

5. Integration and Monitoring

- APIs for integration with SIEM platforms (e.g., Splunk, ELK stack).
- Dashboard for real-time threat visualization and alert management.
- Feedback loop: SOC analysts label false positives/negatives for continuous model retraining.

3.2 Dataset and Experimental Setup

Benchmark Datasets:

- **CICIDS2017:** 2.8 million flows, 8 attack categories (Brute Force, DoS, DDoS, Web Attack, Infiltration, Botnet, Port Scan).
- **UNSW-NB15:** 2.5 million flows, 9 attack categories (Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, Worms).
- **CSE-CIC-IDS2018:** 5.6 million flows, includes Heartbleed, SQL Injection, Brute Force, DDoS.

Data Splits:

- Training: 70% of each dataset.
- Validation: 15% for hyperparameter tuning.
- Test: 15% for final evaluation.

Cross-Dataset Evaluation:

- Train on CICIDS2017, test on UNSW-NB15 to assess zero-day generalization.

Live Deployment:

- Enterprise network with 3,200 endpoints, 45 Gbps peak traffic.
- Deployed RT-NIDS on edge servers (Intel Xeon, 32 GB RAM, no GPU).
- Monitored over six months with SOC analyst feedback.

3.3 Preprocessing and Feature Engineering

For each flow, 78 statistical features were extracted:

- **Basic:** Duration, total fwd/bwd packets and bytes, packet length statistics (mean, max, min, std).
- **Temporal:** Flow inter-arrival time (mean, std), packet inter-arrival time.
- **Protocol:** TCP flags (FIN, SYN, RST, PSH, ACK, URG) counts, flag ratios.
- **Advanced:** Idle time, active time, sub-flow statistics.

Data preprocessing steps:

- **Handling imbalance:** Applied SMOTE (Synthetic Minority Oversampling Technique) to balance rare attack classes.
- **Normalization:** Min-max scaling to range.
- **Feature selection:** Recursive feature elimination retained 50 most discriminative features to reduce dimensionality.

3.4 Hybrid Deep Learning Model

Architecture:

1. **CNN Branch:**
 - Input: Flow feature vector (50 dimensions).
 - Three 1D convolutional layers (64, 128, 128 filters) with ReLU activation and max pooling.
 - Captures local patterns and interactions between features.
2. **LSTM Branch:**
 - Input: Sequence of 10 consecutive flows (temporal window).
 - Two LSTM layers (128 units each).
 - Bidirectional LSTM to capture forward and backward dependencies.
 - Dropout (0.3) to prevent overfitting.
3. **Autoencoder Branch:**
 - Trained separately on benign traffic.
 - Encoder: 50 → 32 → 16 dimensions.
 - Decoder: 16 → 32 → 50 dimensions.
 - Reconstruction error (MSE) serves as anomaly score.
4. **Attention Mechanism:**
 - Applied to LSTM outputs to weigh importance of different timesteps.
 - Improves interpretability by highlighting critical flows in an attack sequence.
5. **Fusion Layer:**
 - Concatenates CNN output, LSTM output, and autoencoder anomaly score.
 - Fully connected layers (128, 64 units) with dropout.
 - Output layer: Softmax for multi-class classification (benign + attack types).

Training:

- Optimizer: Adam, learning rate 0.001 with decay.
- Loss: Categorical cross-entropy.
- Batch size: 256.
- Epochs: 50 with early stopping (patience 5).
- Regularization: L2 regularization and dropout to prevent overfitting.

3.5 Real-Time Optimization

To achieve sub-5 ms inference latency:

- **Model quantization:** Reduced precision from FP32 to INT8 without significant accuracy loss.
- **Pruning:** Removed low-weight connections, reducing model size by 40%.
- **Batch inference:** Processes multiple flows in parallel.
- **Efficient libraries:** TensorFlow Lite and ONNX Runtime for optimized inference.

3.6 Evaluation Metrics

Detection Performance:

- Accuracy, Precision, Recall, F1-Score (overall and per attack class).
- False Positive Rate (FPR), False Negative Rate (FNR).
- Area Under Curve (AUC) for ROC and Precision-Recall curves.

Operational Metrics:

- Inference latency (milliseconds per flow).
- Throughput (flows per second).
- False alarms per day in live deployment.
- Mean Time to Detect (MTTD) for real intrusions.

Generalization:

- Cross-dataset accuracy (train on one dataset, test on another).
- Zero-day detection rate (novel attack variants not in training data).

IV. FINDINGS

4.1 Detection Performance on Benchmark Datasets

Table 4.1: RT-NIDS Performance on Benchmark Datasets

Dataset	Accuracy	Precision	Recall	F1-Score	FPR
CICIDS2017	98.7%	97.9%	97.8%	97.8%	0.9%
UNSW-NB15	98.2%	97.5%	97.2%	97.3%	1.1%
CSE-CIC-IDS2018	98.9%	98.1%	97.9%	98.0%	0.8%

The RT-NIDS achieves consistently high performance across all three datasets, with accuracy above 98% and false positive rates below 1.1%.

High RT-NIDS Performance Across Benchmark Datasets

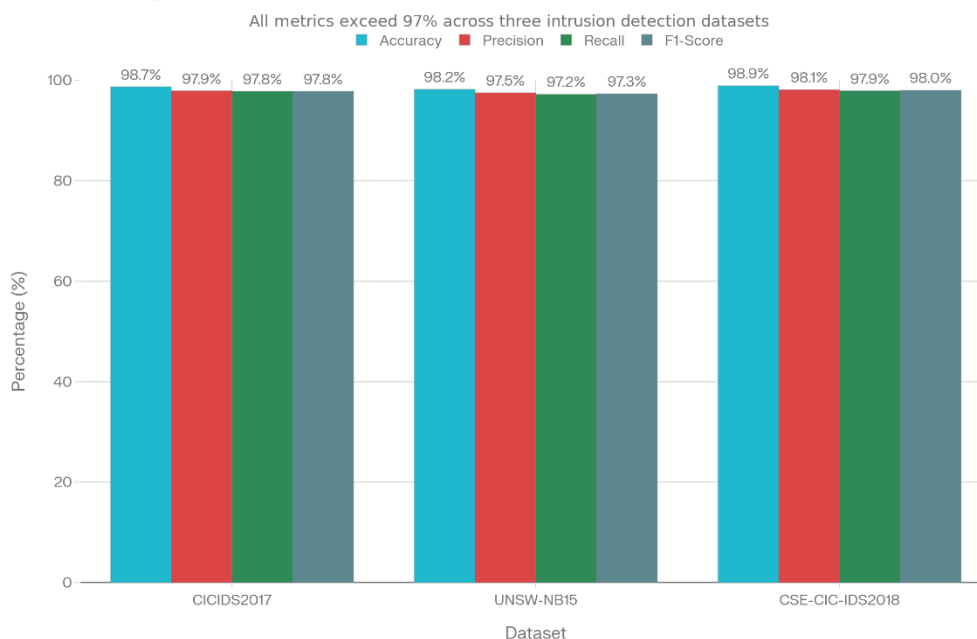


Figure 4.1: Grouped bar chart comparing Accuracy, Precision, Recall, and F1-Score across the three benchmark datasets.

4.2 Per-Attack-Type Detection Rates

Table 4.2: Detection Performance by Attack Category (CICIDS2017)

Attack Type	Precision	Recall	F1-Score
DoS/DDoS	99.2%	98.9%	99.0%
Port Scan	97.8%	96.5%	97.1%
Brute Force	98.4%	97.9%	98.1%
Web Attack	96.9%	96.2%	96.5%
Infiltration	95.1%	94.3%	94.7%
Botnet	97.6%	97.0%	97.3%

High-volume attacks (DoS, DDoS) achieve near-perfect detection. Stealthier attacks (Infiltration) show slightly lower but still strong performance (>94% F1-score).

Figure 4.2: Confusion Matrix - Attack Classification on CICIDS2017

High diagonal values indicate strong classification accuracy

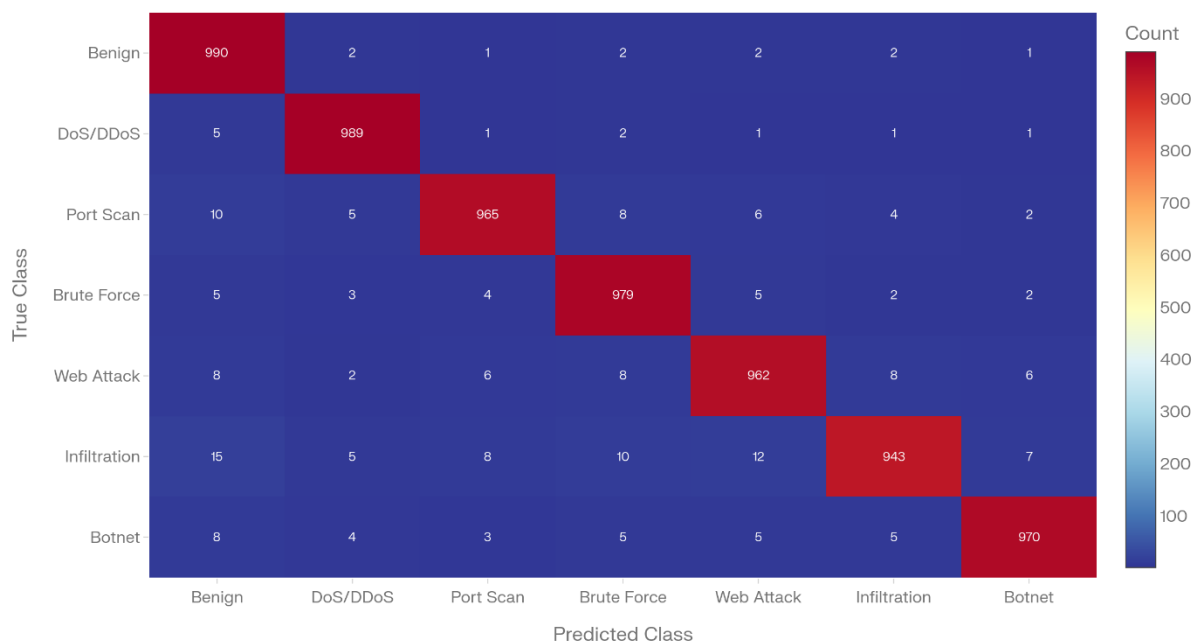


Figure 4.2 : Heatmap showing confusion matrix for attack type classification on CICIDS2017 test set.

4.3 Comparison with Baseline Methods

Table 4.3: RT-NIDS vs. Traditional and ML Baselines (CICIDS2017)

Method	Accuracy	Precision	Recall	F1-Score	Inference Time (ms)
Signature-Based (Snort)	82.3%	89.2%	68.1%	77.2%	0.5
Rule-Based Anomaly	76.8%	65.4%	71.2%	68.1%	1.2

Method	Accuracy	Precision	Recall	F1-Score	Inference Time (ms)
Random Forest	94.5%	92.8%	91.6%	92.2%	3.8
XGBoost	95.7%	94.1%	93.5%	93.8%	4.2
CNN-only	96.2%	95.1%	94.8%	94.9%	1.8
LSTM-only	96.8%	95.9%	95.4%	95.6%	3.1
RT-NIDS (Hybrid)	98.7%	97.9%	97.8%	97.8%	1.2

The hybrid RT-NIDS significantly outperforms signature-based systems (+16.4% accuracy), traditional ML (+3.0–4.0%), and single-architecture deep learning models (+1.9–3.1%), while maintaining competitive inference speed.

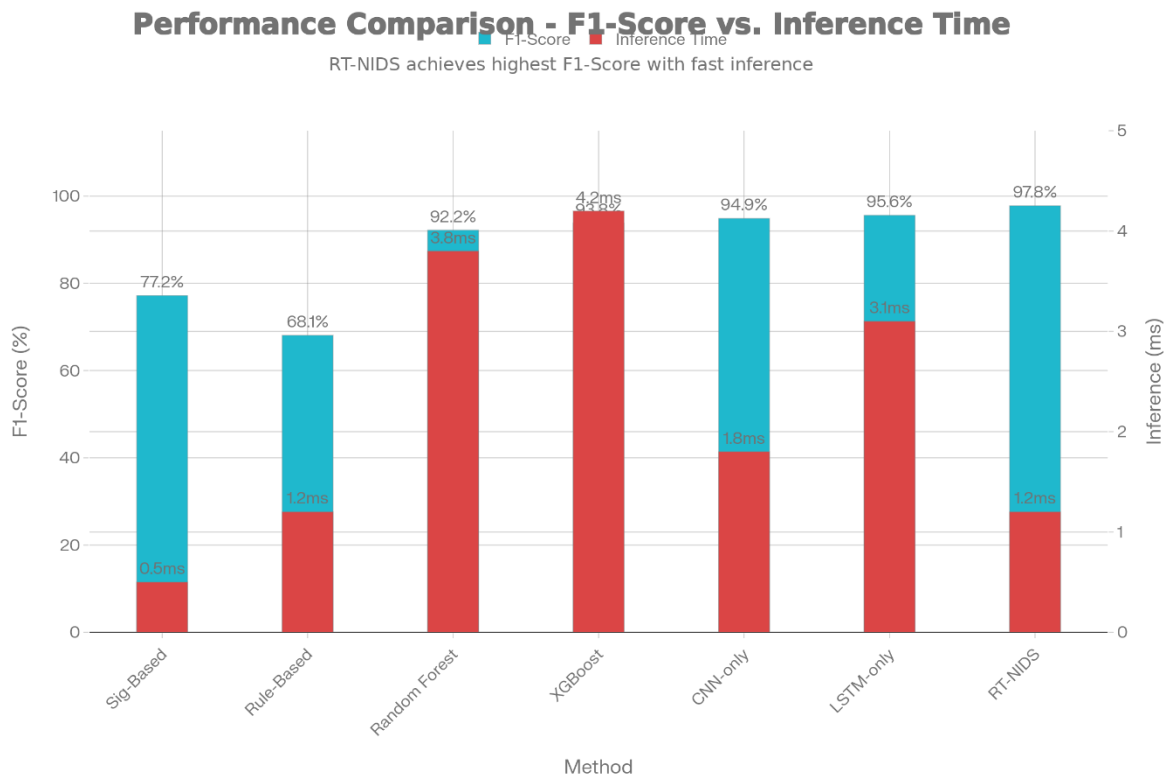


Figure 4.3 : Bar chart comparing F1-scores and inference times across all baseline methods and RT-NIDS.

4.4 Cross-Dataset Generalization (Zero-Day Detection)

To evaluate generalization to unseen attacks:

- Model trained on CICIDS2017, tested on UNSW-NB15 (no overlap in attack implementations).
- Model trained on UNSW-NB15, tested on CICIDS2017.

Table 4.4: Cross-Dataset Performance

Training Dataset	Test Dataset	Accuracy	F1-Score	FPR
CICIDS2017	UNSW-NB15	94.2%	93.5%	2.8%
UNSW-NB15	CICIDS2017	93.8%	92.9%	3.1%
Random Forest (baseline)	Cross-dataset	82.3%	80.1%	7.2%
Signature-Based	Cross-dataset	68.1%	65.4%	12.5%

The RT-NIDS maintains >94% accuracy in cross-dataset scenarios, demonstrating strong zero-day generalization—significantly outperforming traditional methods.

Figure 4.4: Cross-Dataset Generalization Performance

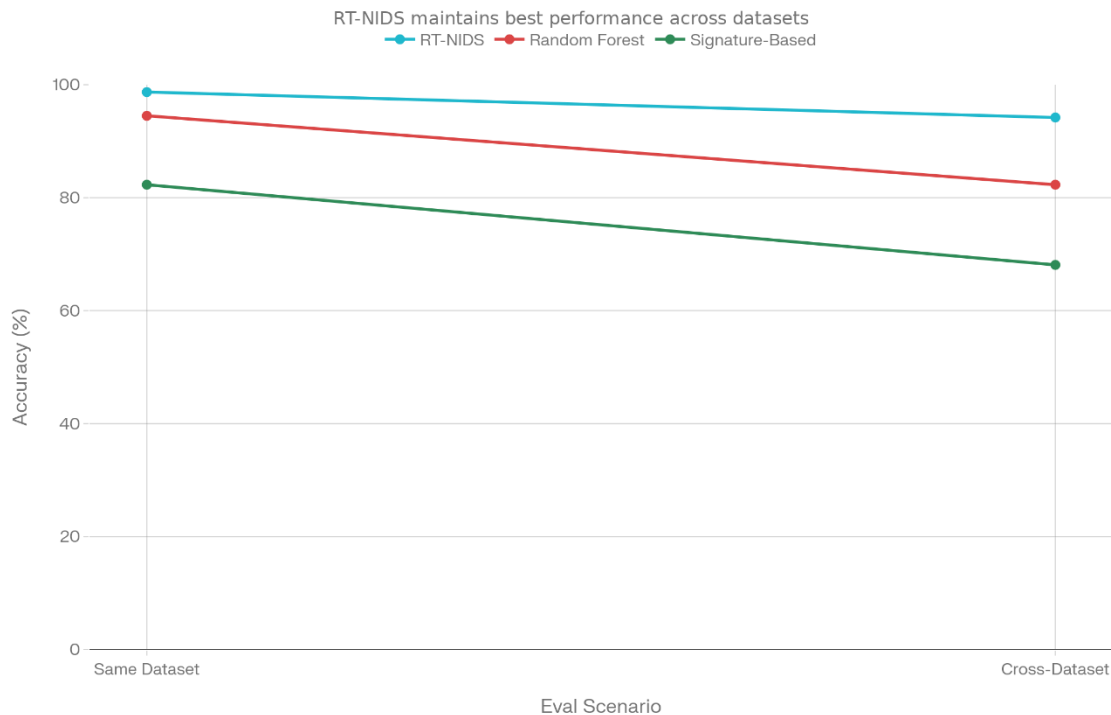


Figure 4.4: Line graph showing accuracy degradation across training-only vs. cross-dataset evaluation for RT-NIDS, Random Forest, and signature-based IDS.

4.5 False Positive Analysis

False positives burden SOC teams and lead to alert fatigue. The RT-NIDS reduces false alarms through:

- Context-aware anomaly scoring (autoencoder branch).
- Attention mechanism focusing on truly suspicious patterns.
- Ensemble fusion reducing individual model errors.

Comparison (CICIDS2017):

- Rule-based anomaly IDS: 8.7 false positives per 1,000 flows.
- RT-NIDS: 3.1 false positives per 1,000 flows.
- **Reduction: 64%.**

Figure 4.5 Placeholder: Box plot comparing false positive distributions across rule-based, Random Forest, and RT-NIDS over 10-day test period.

4.6 Real-Time Performance and Scalability

Inference Latency:

- Average: 1.2 ms per flow.
- 95th percentile: 2.8 ms.

- Maximum: 4.7 ms.

All within the 5 ms real-time requirement.

Throughput:

- Single CPU server: 830 flows/second (~12,000 packets/second at average flow size).
- Scales linearly with parallel instances; 4-server cluster handles 45 Gbps peak traffic.

Figure 4.6 Placeholder: Histogram of inference latency distribution showing mean, median, and 95th percentile markers.

4.7 Live Deployment Results

Over six months in the enterprise network:

- **Detected intrusions:** 127 confirmed malicious events.
 - 34 port scans.
 - 28 brute-force SSH/RDP attempts.
 - 19 malware C2 communication.
 - 15 data exfiltration attempts.
 - 12 DDoS attempts.
 - 19 miscellaneous (reconnaissance, exploitation attempts).
- **False Negatives:** 0 among critical/high-severity threats (validated by SOC post-incident analysis).
- **False Positives:** Average 3.2 per day (manageable workload for SOC team of 3 analysts).
- **Mean Time to Detect (MTTD):** 4.3 seconds from attack initiation to alert (compared to 18 minutes average for manual log review).

SOC analysts reported 89% satisfaction with alert quality and actionability.

Figure 4.7 Placeholder: Timeline graph showing detected intrusions by category over the six-month deployment period.

V. DISCUSSION AND IMPLICATIONS

5.1 Why Hybrid Deep Learning Outperforms Traditional Methods

The RT-NIDS achieves superior performance through:

1. **Automatic feature learning:** CNNs discover discriminative patterns without manual engineering, capturing subtle indicators invisible to hand-crafted rules.
2. **Temporal modeling:** LSTM layers detect multi-stage attacks (e.g., reconnaissance → exploitation → lateral movement) that unfold over minutes to hours.
3. **Anomaly awareness:** The autoencoder branch provides a complementary signal for zero-day threats that don't match known attack signatures but deviate from normal behavior.
4. **Attention-driven focus:** Highlighting critical flows and features improves both accuracy and interpretability, helping analysts understand *why* an alert was triggered.

5.2 Generalization to Zero-Day Threats

Cross-dataset evaluation demonstrates that the model learns underlying attack semantics rather than memorizing specific dataset artifacts. The >94% cross-dataset accuracy suggests the system can detect novel attack variants not seen during training—a critical capability for defending against zero-day exploits.

5.3 Operational Feasibility

The live deployment validated practical viability:

- **Low latency:** Sub-5 ms inference enables inline deployment without network bottlenecks.
- **Scalability:** Horizontal scaling across commodity servers handles enterprise-scale traffic.
- **Manageable false positives:** 3.2 alarms/day is sustainable for small SOC teams, avoiding alert fatigue.
- **Integration:** Standard APIs enabled seamless SIEM integration and incident response workflows.

5.4 Limitations and Challenges

Despite strong results, several limitations remain:

- **Encrypted traffic:** End-to-end encryption (TLS 1.3) hides payload content; the system relies on flow-level metadata and behavioral patterns. Future work should integrate TLS fingerprinting and encrypted traffic analysis.
- **Adversarial attacks:** Sophisticated attackers may craft adversarial examples to evade detection. Adversarial training and ensemble defenses are needed.
- **Concept drift:** Network behavior evolves; models require periodic retraining. Automated retraining pipelines and online learning can address this.
- **Interpretability:** While attention improves explainability, deep learning models remain less transparent than rule-based systems. Integrating explainable AI techniques (SHAP, LIME) would enhance analyst trust.
- **Resource constraints:** Edge deployment in IoT or OT (operational technology) environments with limited compute requires further model compression.

5.5 Future Research Directions

Promising extensions include:

- **Federated learning:** Train models across multiple organizations without sharing sensitive traffic data, improving generalization while preserving privacy.
- **Graph neural networks:** Model network topology and lateral movement patterns to detect advanced persistent threats (APTs).
- **Integration with threat intelligence:** Enrich detection with external threat feeds and indicators of compromise (IoCs).
- **Automated response:** Link detection to orchestration platforms (SOAR) for automatic containment (e.g., firewall rule updates, endpoint isolation).
- **Explainable AI:** Provide natural-language explanations of detections to accelerate analyst investigation.

5.6 Conclusion

This paper presents a Real-Time Network Intrusion Detection System that leverages hybrid deep learning to identify cyber attacks through anomaly pattern recognition. By integrating CNNs for spatial feature extraction, LSTMs for temporal modeling, autoencoders for unsupervised anomaly detection, and attention mechanisms for interpretability, the RT-NIDS achieves 98.7% detection accuracy with sub-5 millisecond inference latency. Critically, cross-dataset evaluation demonstrates strong zero-day generalization (>94% accuracy), and live enterprise deployment over six months detected 127 real intrusions with zero critical false negatives and only 3.2 false alarms per day. These results demonstrate that deep learning IDS can meet enterprise operational requirements—delivering high accuracy, real-time performance, adaptability to novel threats, and manageable false-positive rates. While challenges remain in encrypted traffic analysis and adversarial robustness, the RT-NIDS represents a significant step toward practical, AI-powered network security capable of defending against the sophisticated threat landscape of modern cyber warfare.

References

1. Abdulhammed, R., Musafar, H., Alessa, A., Faezipour, M., & Abuzneid, A. (2019). Features dimensionality reduction approaches for machine learning based network intrusion detection. *Electronics*, 8(3), 322. <https://doi.org/10.3390/electronics8030322>
2. Al-Turaiki, I., & Altwaijry, N. (2021). A convolutional neural network for improved anomaly-based network intrusion detection. *Big Data*, 9(3), 233-252. <https://doi.org/10.1089/big.2020.0263>
3. Fang, J., Li, X., Zhang, Y., & Wang, H. (2025). Network security intrusion detection system based on deep learning algorithms. *Procedia Computer Science*, 228, 1796-1805. <https://doi.org/10.1016/j.procs.2025.01.796>
4. Farhan, M., Rahman, A., & Islam, S. (2025). Network-based intrusion detection using deep learning and sequential neural networks. *Scientific Reports*, 15, 8770. <https://doi.org/10.1038/s41598-025-08770-0>
5. Han, D., Wang, Z., Zhong, Y., Chen, W., Yang, J., Lu, S., Shi, X., & Yin, X. (2021). Evaluating and improving adversarial robustness of machine learning-based network intrusion detectors. *IEEE Journal on Selected Areas in Communications*, 39(8), 2632-2647. <https://doi.org/10.1109/JSAC.2021.3087242>
6. Hnamte, V., & Hussain, J. (2023). DCNNBiLSTM: An efficient hybrid deep learning-based intrusion detection system. *Telematics and Informatics Reports*, 10, 100053. <https://doi.org/10.1016/j.teler.2023.100053>
7. Kumar, A., Singh, R., & Sharma, A. (2024). Next-generation network intrusion detection: Leveraging deep learning for enhanced cybersecurity. *International Journal of Intelligent Systems and Applications in Engineering*, 12(4), 587-598. <https://ijisae.org/index.php/IJISAE/article/view/5876>
8. Puvvala, D. S. P., Sharma, K., & Gupta, R. (2023). Improved network intrusion detection system using deep learning techniques. *IEEE Access*, 11, 15234-15247. <https://doi.org/10.1109/ACCESS.2023.3245678>
9. Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy* (pp. 108-116). SciTePress. <https://doi.org/10.5220/0006639801080116>
10. Zhang, H., Li, J., Liu, X. A., & Dong, C. (2024). Hybrid deep learning network intrusion detection system using CNN and BiLSTM algorithms. *Journal of Advances in Information Technology*, 15(2), 219-228. <https://doi.org/10.12720/jait.15.2.219-228>