Mrs.D.Mercy Smilin., 2026, 14:4
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

AI-Powered Cyber Threat Detection Using Network Traffic Analysis

Mrs.D.Mercy Smilin[1], A.Sabikka Barvin[2], G. Gopika[3], P. Nathir Sahith[4], V. Rathika[5], M. Saranya[6]

Assistant Professor, Anna University, Chinnai/J.P.College of Engineering, Tenkasi[1]
Student, Anna University, Chennai/J.P.College of Engineering, Tenkasi [2,3,4,5,6]

Abstract- Modern networks generate large volumes of traffic that must be continuously monitored to identify malicious activity. Traditional signature-based intrusion detection systems struggle to identify novel or evolving attacks and often produce a high rate of false alerts. This paper presents a framework for AI-powered cyber threat detection based on the analysis of network traffic. Traffic flows are captured, pre-processed, and represented as feature vectors describing packet, flow, and behavioral characteristics. Machine learning and deep learning models are trained to distinguish normal traffic from malicious activity and to classify detected threats into categories such as denial-of-service, port scanning, brute-force access attempts, and malware communication. The framework emphasizes a repeatable pipeline covering data collection, feature extraction, model training, real-time inference, and alert generation. The proposed approach provides a basis for building adaptive, data-driven intrusion detection systems capable of identifying both known and previously unseen threats.

Keywords— Cyber Threat Detection, Network Traffic Analysis, Intrusion Detection, Machine Learning, Deep Learning, Anomaly Detection, Network Security, Artificial Intelligence.

I. INTRODUCTION

Network security depends on the ability to detect malicious activity within large and continuously changing volumes of traffic. Firewalls, access controls, and signature-based intrusion detection systems form the traditional line of defense, but these approaches rely on predefined rules and known attack signatures. As attackers adopt new techniques and modify existing ones, static rule-based systems become less effective, allowing novel or disguised threats to pass undetected.

Artificial intelligence offers an alternative approach by learning patterns directly from network traffic data rather than relying solely on fixed rules. Machine learning and deep learning models can be trained on historical traffic to distinguish between benign and malicious behavior, and to generalize this knowledge to traffic patterns not explicitly seen during training. Effective detection depends on the quality of captured traffic, the features extracted from it, the choice of learning model, and the ability of the system to operate under real-time conditions.

II. LITERATURE SURVEY

Earlier work on network intrusion detection has explored statistical anomaly detection, rule-based expert systems, and classical machine learning classifiers such as decision trees, support vector machines, and ensemble methods applied to labeled traffic datasets. More recent studies have examined deep learning architectures, including convolutional and recurrent neural networks, for learning traffic patterns directly from raw or minimally processed packet and flow data. Statistical and rule-based approaches were among the earliest methods applied to intrusion detection. These systems compare observed traffic against fixed thresholds or known attack signatures and can achieve high precision for previously documented attacks, but they generally fail to detect variants that fall outside the defined rule set. Anomaly-based statistical methods attempt to address this limitation by modeling normal traffic behavior and flagging deviations, though they are prone to elevated false-positive rates when normal traffic itself is highly variable. Classical machine learning approaches, including decision trees, random forests, support vector machines, and k-nearest-neighbor classifiers,

have been widely applied to flow-based traffic features extracted from benchmark datasets. These models generally offer good interpretability and comparatively low computational cost, and several studies report strong classification performance when trained and tested on the same dataset. However, their performance often degrades when applied to traffic distributions that differ from the training data, reflecting limited generalization across network environments. Deep learning approaches, including convolutional neural networks, recurrent and long short-term memory networks, and autoencoder-based anomaly detectors, have been investigated for their ability to learn traffic representations automatically rather than relying on hand-crafted features. Reported results suggest that these models can capture temporal and sequential patterns in traffic flows, which is useful for detecting multi-stage or slow-developing attacks. Training such models typically requires larger volumes of labeled data and greater computational resources, and their internal decision process is less directly interpretable than that of classical classifiers. Hybrid and ensemble approaches that combine multiple models, or that pair anomaly detection with supervised classification, have also been proposed to balance detection coverage against false-positive rates. Some studies additionally explore semi-supervised and unsupervised techniques to reduce dependence on fully labeled attack data, since labeled malicious traffic is often scarce relative to benign traffic in real deployments. A recurring challenge across this body of work is the availability of representative, well-labeled traffic datasets and the difficulty of evaluating models under conditions that reflect real network environments. Reported detection accuracy can vary considerably depending on the dataset, the feature set, and the evaluation methodology used, making direct comparison between studies difficult. Differences in traffic capture conditions, attack diversity, class balance, and the presence of encrypted or obfuscated traffic further complicate the interpretation of published results, motivating the need for a consistent evaluation framework such as the one proposed in this paper.

III. PROBLEM STATEMENT

Conventional intrusion detection systems primarily rely on predefined signatures and static rules, which limits their ability to identify new or evolving attack patterns and often results in a high number of false positive alerts. Existing machine-learning-based approaches vary in effectiveness because they differ in the datasets used, the features extracted from traffic, the learning models applied, and the conditions under which they are evaluated. There is therefore a need for a structured framework that captures network traffic, extracts meaningful features, trains and evaluates AI-based detection models under consistent conditions, and supports real-time classification of network threats.

IV. DATASET DESCRIPTION

Evaluation of the proposed framework relies on labeled network traffic that reflects both normal usage and a representative range of attack behavior. Suitable sources include publicly available benchmark intrusion-detection datasets as well as traffic captured from a controlled testbed in which attack scenarios are generated in a supervised setting alongside normal background traffic. Using more than one source helps assess whether a trained model generalizes beyond the traffic distribution it was trained on.Each traffic sample is represented as a labeled record describing either an individual packet or an aggregated flow between a source and destination endpoint. Labels distinguish benign traffic from malicious traffic and, where available, further identify the specific attack category, such as denial-of-service, distributed denial-of-service, port scanning, brute-force login attempts, web-application attacks, and malware command-and-control communication.

Table 1. Traffic Categories Used for Model Training and Evaluation

Traffic Category	Description	Representative Features
Benign	Normal user and application traffic	Typical packet size, stable flow duration
DoS / DDoS	Volumetric flooding of a target host or service	High packet rate, short inter-arrival time
Scanning	Systematic probing of ports or hosts	Many short flows, repeated destination ports
Brute-Force	Repeated authentication attempts	Repetitive connection attempts, small payloads
Malware C2	Command-and-control communication	Periodic beaconing, unusual destination patterns

Class imbalance is expected, since benign traffic typically outnumbers malicious traffic by a wide margin in real network environments. This is addressed through a combination of stratified sampling, class weighting during model training, and, where appropriate, controlled oversampling of minority attack classes so that rare but significant attack types are not overwhelmed by the volume of normal traffic.

V. PROPOSED METHODOLOGY

The proposed framework begins with the collection of network traffic from a monitored environment, using packet capture or flow-export mechanisms positioned at a suitable point in the network, such as a gateway, switch mirror port, or virtual network tap. Captured traffic is organized into flows or packet sequences and labeled as benign or malicious using a reference dataset or controlled attack simulation.

Network Traffic Capture Layer
↓
Pre-processing and Feature Extraction
↓
AI-Based Classification Model
↓
Alert Generation and Reporting

Fig. 1. Proposed architecture for AI-powered network traffic threat detection.

Extracted features include packet-level attributes such as protocol type, packet size, and header flags, as well as flow-level attributes such as duration, byte and packet counts, and inter-arrival timing statistics. These features are normalized and passed to a classification model. Selected models are trained on a labeled portion of the traffic and validated on data not used during training.

A controlled experimental design is important because network traffic composition can vary naturally across time and environments. The dataset should therefore be partitioned into training, validation, and test sets, and class imbalance between benign and malicious samples should be addressed using appropriate sampling or weighting techniques.

Table 2. Candidate Detection Models

Model	Description	Primary Use
Baseline	Rule/signature-based reference detector	Comparison benchmark
M1	Classical machine learning classifier (e.g., random forest)	Flow-based classification
M2	Deep neural network / sequence model	Pattern and behavior learning
M3	Hybrid or ensemble model	Combined detection and classification

Model performance is recorded using the same evaluation protocol for every configuration. Accuracy, precision, recall, F1-score, and false-

positive rate can be calculated for each model, followed by an appropriate statistical comparison. If certain models show consistent improvement over the baseline, results can be examined for a possible generalizable detection pattern.

VI. SYSTEM ARCHITECTURE

The system architecture connects the traffic capture layer, feature processing stage, AI-based detection engine, and alerting layer. The capture layer collects raw traffic, the processing stage converts it into structured feature vectors, the detection engine classifies the traffic as benign or malicious, and the alerting layer records and reports identified threats to a security dashboard or incident-response workflow.

Table. 3. System architecture connecting traffic capture, AI-based detection, and alerting.

Traffic Capture	Feature & Model Engine	Alerting Layer
Packet/flow capture Protocol parsing Traffic aggregation	Feature extraction Normalization AI classification model	Threat classification Severity scoring Security dashboard

Data flows from left to right through the architecture: raw traffic is captured, converted into features, classified by the trained model, and reported through the alerting layer.

Network Traffic Source
↓
Feature Extraction Module
↓
Trained AI Detection Model
↓
Classification: Benign / Malicious
↓
Alert and Log Generation

Fig. 3. Detection flowchart from traffic ingestion to alert generation.

VII. ALGORITHM

Algorithm 1: AI-Based Network Traffic Threat Detection

- Step 1: Capture network traffic from the monitored segment using packet or flow collection.
- Step 2: Parse captured traffic and organize it into individual flows or sessions.
- Step 3: Extract packet-level and flow-level features for each traffic sample.
- Step 4: Normalize and encode extracted features for model input.
- Step 5: Split labeled data into training, validation, and test sets.
- Step 6: Train candidate machine learning and deep learning models on the training set.
- Step 7: Validate each model and tune parameters using the validation set.
- Step 8: Evaluate final models on the held-out test set.
- Step 9: Deploy the selected model for real-time or batch traffic classification.
- Step 10: Classify incoming traffic as benign or malicious, with threat-category labeling where applicable.
- Step 11: Generate alerts and logs for traffic classified as malicious.
- Step 12: Periodically retrain the model as new labeled traffic becomes available.
- Input: Captured network traffic, labeled training data, defined feature set, and selected AI model configuration.
- Output: Traffic classification results, threat alerts, performance metrics, and detection logs.

Evaluation Metrics and Features

The primary evaluation metrics are accuracy, precision, recall, F1-score, false-positive rate, and detection latency. These metrics provide direct information about detection effectiveness and operational suitability. Feature categories may include packet-level attributes, flow statistics, timing behavior, and payload-derived indicators, depending on the capture method and privacy constraints of the monitored environment.

Detection accuracy (%) = ((TP + TN) / (TP + TN + FP + FN)) × 100.

Precision = TP / (TP + FP). Recall (Detection Rate) = TP / (TP + FN).

F1-score = 2 × (Precision × Recall) / (Precision + Recall).

False-Positive Rate = FP / (FP + TN).

Here, TP (true positives) denotes malicious traffic correctly identified as malicious, TN (true negatives) denotes benign traffic correctly identified as benign, FP (false positives) denotes benign traffic incorrectly flagged as malicious, and FN (false negatives) denotes malicious traffic that is not detected. Detection latency, measured as the time between traffic capture and alert generation, is recorded separately to assess suitability for real-time deployment. Statistical validation, such as k-fold cross-validation or repeated test runs, should be used before concluding that a model provides a meaningful improvement in detection performance.

VIII. EXPECTED RESULTS

The proposed framework is expected to show that AI-based models can identify malicious traffic patterns with higher accuracy and lower false-positive rates than a static signature-based baseline, particularly for attack types not explicitly represented in predefined signatures. Some models may perform better on specific attack categories, such as denial-of-service or scanning activity, while others may be more effective at identifying subtle or low-volume malicious behavior. Detection performance may also depend on the diversity and balance of the training data.It is anticipated that classical machine learning models will provide competitive accuracy with lower computational overhead on well-structured flow features, while deep learning models will show an advantage on attack categories that depend on sequential or temporal patterns, such as slow scanning or multi-stage intrusion attempts. Ensemble or hybrid configurations are expected to offer the most balanced trade-off between detection rate and false-positive rate, at the cost of increased model complexity and training time. Based on trends reported in comparable studies, the signature-based baseline is expected to achieve high precision on known attack signatures but a noticeably lower recall on novel or modified attacks, since traffic that does not match a stored signature passes undetected. AI-based models trained on flow and behavioral features are expected to close this recall gap substantially, though typically with a modest increase in false-positive rate relative to the baseline. The projected performance ranges for each candidate model, expressed as expected bands rather than fixed values, are summarized below.

Table 3. Projected Performance Ranges for Candidate Models (Illustrative Estimates)

Model	Expected Accuracy	Expected Recall	Expected F1-score	Expected FPR
Baseline	85% – 90%	60% – 70%	0.68 – 0.75	2% – 4%
M1 (Classical ML)	90% – 95%	85% – 90%	0.86 – 0.91	3% – 6%
M2 (Deep Learning)	92% – 97%	88% – 94%	0.89 – 0.94	3% – 7%
M3 (Hybrid/Ensemble)	94% – 98%	90% – 96%	0.92 – 0.96	2% – 5%

These ranges are illustrative projections drawn from patterns observed in comparable published studies and are intended to guide experimental expectations; actual results must be obtained through the evaluation procedure described in Sections VI and IX before any performance claim is finalized. At the level of individual attack categories, denial-of-service and distributed denial-of-service traffic are expected to be the easiest to detect, since volumetric flooding produces strong, distinctive statistical signatures in packet rate and flow duration that most models should learn readily. Port and network scanning is also expected to be detected with high reliability, owing to the repetitive connection patterns characteristic of probing

behavior. Brute-force authentication attempts are expected to be detected with moderate to high accuracy, though performance may be more sensitive to the specific service being targeted and the similarity between attack traffic and legitimate repeated-login behavior. Malware command-and-control communication is expected to be the most challenging category, since beaconing traffic can closely resemble low-volume legitimate application traffic and may be deliberately designed to evade detection; models incorporating timing and periodicity features are expected to outperform those relying solely on packet-size statistics for this category. Detection latency is expected to remain within operationally acceptable bounds for classical machine learning models, supporting near-real-time classification. Deep learning models, while generally offering higher detection quality, are expected to introduce additional inference latency, which may require optimization, batching, or hardware acceleration for deployment in high-throughput network environments. Overall, results are expected to confirm that no single model dominates across all attack categories, reinforcing the value of the comparative, multi-model evaluation approach adopted in this framework.

X. COMPARATIVE ANALYSIS

To assess relative effectiveness, candidate models are compared against the signature-based baseline and against each other using the metrics defined in Section IX. Comparison is performed on the same held-out test set for every model to ensure that differences in reported performance reflect model behavior rather than differences in evaluation data.

Table IV. Comparative Evaluation Template for Candidate Detection Models

Model	Precision	Recall	F1-score	False-Positive Rate
Baseline	Reference	Reference	Reference	Reference
M1	To be measured	To be measured	To be measured	To be measured
M2	To be measured	To be measured	To be measured	To be measured
M3	To be measured	To be measured	To be measured	To be measured

This comparison template standardizes reporting across models so that improvements can be attributed specifically to model architecture and feature representation rather than to inconsistencies in evaluation procedure. Where computational resources permit, training time and inference latency per flow should also be recorded, since operational suitability depends on both detection quality and processing speed.

Future Scope

Future research can examine a broader range of attack types and compare additional model architectures, including transformer-based and graph-based approaches for traffic analysis. Experiments can be extended to encrypted-traffic scenarios and multi-network deployments to determine whether detection performance generalizes across environments. Automated pipelines can continuously retrain models as new traffic and threat data become available. Advanced studies may investigate explainability of AI-based detection decisions, adversarial robustness against traffic designed to evade detection, and integration with automated incident-response systems. Federated learning approaches could also be explored to allow models to be trained collaboratively across multiple organizations without sharing raw traffic data, addressing privacy concerns while improving generalization. Controlled testbed trials followed by production-scale validation would be required before broad operational deployment could be recommended.

XI. CONCLUSION

This paper presents a framework for AI-powered cyber threat detection based on the analysis of network traffic. By comparing machine learning and deep learning models against a signature-based baseline under consistent data collection and evaluation conditions, the framework can help

determine whether measurable improvements in detection accuracy and false-positive reduction can be achieved. Combining traffic-derived features with structured model evaluation provides a more complete assessment than relying on a single static rule set.The proposed approach can support further research into adaptive, data-driven intrusion detection. Its main contribution is a structured methodology that emphasizes consistent feature extraction, repeatable evaluation, measurable outcomes, and careful interpretation of AI-based detection results.

REFERENCES

1. V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," ACM Computing Surveys, 2009.
2. N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems," Military Communications and Information Systems Conference, 2015.
3. M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009.
4. I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," International Conference on Information Systems Security and Privacy, 2018.
5. R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," IEEE Symposium on Security and Privacy, 2010.
6. N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," IEEE Transactions on Emerging Topics in Computational Intelligence, 2018.
7. Y. Xin et al., "Machine learning and deep learning methods for cybersecurity," IEEE Access, 2018.
8. M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," Journal of Information Security and Applications, 2020.