RESEARCH PAPER

A Hybrid Real-Time Network Intrusion Detection and Prevention System Using Isolation Forest and Deep Neural Network

V. Bakyalakshmi[1], S. Bharanidharan[2], K. Deepasri[3]

[1]Assistant Professor, IFET College of Engineering, Villupuram, Tamil Nadu. Email: bakyalakshmi.v@gmail.com

[2]Undergraduate, IFET College of Engineering, Villupuram, Tamil Nadu. Email: bharani4072005@gmail.com

[3]Undergraduate, IFET College of Engineering, Villupuram, Tamil Nadu. Email: deepasrikumar22@gmail.com

***Corresponding author: V. Bakyalakshmi, Assistant Professor, IFET College of Engineering, Villupuram, Tamil Nadu. Email: bakyalakshmi.v@gmail.com**

Abstract—Modern enterprise networks are under constant pressure from cyber threats ranging from stealthy port scans to large-scale denial-of-service floods. Existing systems largely treat intrusion detection as a binary problem, limiting operational usefulness since different attack types require very different defensive responses. This paper proposes a hybrid NIDPS that pairs an Isolation Forest for unsupervised anomaly detection with a four-layer Deep Neural Network performing direct multiclass traffic classification. Traffic is captured live from a Wi-Fi interface using Scapy, and 71 flow-level features are computed incrementally from packet headers in real time. Trained on CIC-IDS-2017, the hybrid model achieved approximately 95.8% accuracy, a macro F1-score of 95.3%, and a 99.6% attack detection rate on a 160,000-flow test set. An automated prevention module blocks malicious source IPs within 4.2 seconds. Live deployment confirmed 100% DNN confidence on Nmap SYN scan detection with successful automated blocking.

Index Terms—Network Intrusion Detection, Multi-Class Classification, Isolation Forest, Deep Neural Network, CIC-IDS-2017, Real-Time Detection, SMOTE, Scapy, IP Blocking, Streamlit.

How to cite this article: Bakyalakshmi V, Bharanidharan S, Deepasri K. A Hybrid Real-Time Network Intrusion Detection and Prevention System Using Isolation Forest and Deep Neural Network. Int J Drug Deliv Technol. 2026;16(67s):2151-2155. DOI: 10.25258/ijddt.16.67s.194

I. INTRODUCTION

The rapid expansion of internet-connected infrastructure has made network security one of the most pressing challenges in modern computing. Attackers have grown increasingly sophisticated — they rotate techniques, probe for zero-day vulnerabilities, and adapt quickly when initial approaches fail. The financial toll of a successful breach routinely runs into millions of dollars, and recovery can take weeks [1].

Traditional Network Intrusion Detection Systems (NIDS) rely on predefined signatures and rules. They work well against known attacks but fail entirely against anything new. Zero-day exploits — precisely because no signature exists for them — are a growing weapon of choice for adversaries. Making things worse, most deployed NIDS are detection-only: they fire alerts, but someone still has to investigate and respond manually. That human-in-the-loop delay is exploited every time.

Deep learning has brought impressive accuracy gains to intrusion detection, but most published systems still frame the problem as binary — benign or malicious. This matters because a DDoS attack and a brute-force attempt look very different in practice and demand completely different responses.
Treating them as the same "attack" label wastes the DNN's ability to provide actionable information.

Unsupervised anomaly detection fills a different gap. The Isolation Forest [3] needs no labelled attack data during training — it simply learns what normal traffic looks like, then flags anything that deviates. This makes it inherently capable of catching novel attacks that no supervised model has ever seen. But it comes with a price: higher false positive rates compared to well-trained supervised classifiers.

This paper combines the strengths of both approaches. The proposed NIDPS pairs an Isolation Forest with a multi-class Deep Neural Network in a parallel hybrid architecture. Every incoming flow is assessed by both models independently. The DNN provides granular, per-class output — the operator sees PortScan, DDoS, or BruteForce, not just "attack" — while the Isolation Forest catches anomalies that fall outside the DNN's training distribution. An automated prevention module translates detections into OS-level IP blocks, rate limits, and alerts within seconds, closing the response gap that characterises all detection-only systems in the literature. The main contributions are:

1) A parallel hybrid engine combining Isolation Forest and a multi-class DNN achieving ~95.8% accuracy and 99.6% attack recall on CIC-IDS-2017.
2) Direct seven-class attack identification without any postDNN classification step.
3) Automated prevention with 4.2-second mean blocking latency.
4) A real-time flow aggregation engine computing 71 features live from packet headers.
5) End-to-end live validation on genuine network traffic.

II. RELATED WORK

A. Deep Learning for Intrusion Detection

Bandarupalli et al. [2] demonstrated that careful feature engineering combined with a DNN on CIC-IDS-2017 can yield

near-perfect offline accuracy. Halbouni et al. [5] pushed this further with a CNN-BiLSTM hybrid on CIC-IDS-2018.

Both systems evaluate on pre-extracted static feature files and treat classification as a binary task. Chytas et al. [13] assessed multiple ML techniques including Random Forest and SVM on benchmark IDS datasets, confirming that ensemble and deep approaches consistently outperform classical classifiers. The proposed system differs by operating on genuinely live traffic and classifying into seven meaningful attack categories directly.

B. Isolation Forest and Anomaly Detection

Liu et al. [3] introduced the Isolation Forest as a highly efficient anomaly detector that requires no labelled data. Xu et al. [6] later proposed a Deep Isolation Forest variant that embeds data representations before partitioning, improving performance in complex feature spaces. Zhang et al. [11] confirmed linear-time scalability for streaming applications over SD-WAN. Dubey et al. [15] further showed that IsolationForest-based instant anomaly algorithms can be integrated with ML pipelines for adaptive detection. These works validate the Isolation Forest as a strong unsupervised component, particularly for zero-day detection.

C. Real-Time and Prevention Systems

Vishrutha and Nagaraja [8] built a live-capture IDS with Scapy and email alerting, proving real-time feasibility. Taleb et al. [9] demonstrated low-latency deep learning inference for streaming threat detection but without an unsupervised component. Bian and Liu [14] proposed a representation learning technique for IoT network intrusion detection, highlighting the importance of feature quality in resource-constrained environments. None of these systems combines live packet capture, hybrid unsupervised-supervised detection, granular multi-class labelling, and automated IP-level prevention in a single deployment — the gap this work addresses.

III. PROPOSED SYSTEM ARCHITECTURE

A. System Overview

The NIDPS processes traffic through five sequential stages: live capture, flow aggregation, feature extraction, hybrid model inference, and automated prevention. All stages run in a continuous loop with no offline preprocessing starting from Live Capture to Flow Aggregator then [IF ‖ DNN] proceeding to Prevention Module and finally Streamlit Dashboard

B. Live Packet Capture and Flow Aggregation

Packets are captured using Scapy's sniff() bound to the active Wi-Fi interface on Windows 11 with Npcap 1.79 in WinPcap-compatible mode. A BPF filter restricts capture to IP packets. Each packet is routed to the flow handler, which maintains a dictionary of bidirectional flows keyed on the fivetuple (src IP, dst IP, src port, dst port, protocol). A background thread flushes flows idle for more than five seconds, computing 71 features and dispatching the feature vector to the detection engine. CICFlowMeter was deliberately not used — it is an offline batch tool unsuitable for streaming operation.

C. Feature Selection

Seventy-one features were selected from the 78 CIC-IDS2017 attributes. The selection rule was strict: every retained feature must be derivable from raw IP and TCP/UDP headers during live capture with no payload inspection. Seven features requiring payload access or offline computation were dropped. Table I summarises the feature categories.

D. Traffic Class Consolidation

CIC-IDS-2017 originally provides 15 labelled categories. Operationally related categories are merged into seven classes to improve learning, remove statistically unreliable minority classes, and produce outputs that map directly to prevention actions. The four DoS variants (Hulk, GoldenEye, Slowloris, Slowhttptest) are merged — all target resource exhaustion and trigger the same block-and-rate-limit response. FTP-Patator and SSH-Patator become BruteForce. Web Attack subtypes (XSS, SQL Injection, Brute Force) become WebAttack. DDoS stays separate because distributed attacks require sourcediverse blocking, not single-IP rules. Infiltration (36 samples) and Heartbleed (11 samples) are excluded — their sample counts are too small for reliable learning and they are not practical real-time threats on typical networks. The final taxonomy is: BENIGN, DoS, DDoS, PortScan, BruteForce, WebAttack, Bot.

TABLE I
SELECTED FEATURE CATEGORIES

Category	Count	Key Examples
Packet Length Stats	10	Fwd/Bwd Mean, Std, Max, Min
Flow Timing & IAT	14	Duration, IAT Mean, Std
Packet Count & Rate	8	Fwd/Bwd Pkts, Flow Pkts/s
TCP Flags	8	SYN, FIN, RST, PSH, ACK
Header & Window	6	Fwd/Bwd Header Len, Init Win
Sub-flow & Segment	8	Subflow Fwd/Bwd Pkts, Bytes
Active / Idle	8	Active Mean, Std, Max, Min
Ratio & Misc.	9	Down/Up Ratio, Avg Pkt Size
TOTAL	71	All from live packet headers

E. Preprocessing and SMOTE Balancing

All eight CIC-IDS-2017 CSV files were concatenated (2,830,743 rows), cleaned of NaN and infinite values (leaving 2,827,876), and labels were remapped to the seven-class taxonomy. A stratified 800,000-sample subset was drawn. Features were standardised using StandardScaler fitted on the training split only, preventing data leakage. SMOTE was applied per class in the training split, expanding the training set to approximately 1,120,000 balanced samples.

F. Mathematical Formulation

Let $\mathbf{x} \in \mathbb{R}^{71}$ be a flow feature vector. The Isolation Forest decision is defined as:

$$f_{\text{IF}}(\mathbf{x}) = \begin{cases} 1 & \text{if } s(\mathbf{x},n) < 0.5 \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where $s(\mathbf{x},n)$ is the average path-length anomaly score across 100 trees. The DNN softmax output for class k is:

$$p_k = \frac{\exp(z_k)}{\sum_j \exp(z_j)}, \quad k \in \{1,\ldots,7\} \quad (2)$$

The hybrid decision rule accepts the DNN's argmax class when $\max_k(p_k) > 0.5$. When DNN confidence falls below this threshold but the Isolation Forest flags the flow, the flow is escalated as an anomalous threat for prevention action.

G. DNN Architecture

The network consists of four fully connected hidden layers (256 → 128 → 64 → 32 units), each with ReLU activation, Batch Normalisation, and Dropout (0.3 for layers 1–2, 0.2 for layer 3). The output layer has 7 units with softmax activation. Loss: categorical cross-entropy. Trained with Adam (lr = 0.001, batch size = 1024) for up to 30 epochs with early stopping at patience 5, restoring best-validation-accuracy weights. Full architecture: 71 → 256 → 128 → 64 → 32 → 7.

Fig. 2 illustrates this.

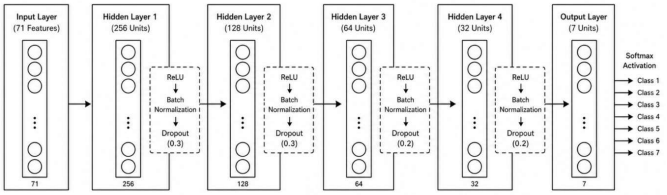

Fig. 2. DNN architecture: 71 input features → four hidden layers → 7-class softmax output.

H. Isolation Forest Configuration

The Isolation Forest was trained on BENIGN-only samples from the pre-SMOTE training split using 100 estimators, contamination = 0.05, and random state = 42. Training on benign-only data preserves zero-day detection capability because the model has zero exposure to any attack pattern, ensuring generalisability to unseen threat types.

I. Prevention Module

Each ATTACK-classified flow triggers an automated response based on DNN confidence. High-confidence flows (> 90%) result in immediate OS-level firewall IP blocks. Medium-confidence flows (70–90%) trigger rate limiting. All actions are logged with timestamp, source IP, attack class, and confidence score. The prevention panel in the Streamlit dashboard displays the full action log, exportable for compliance reporting.

J. Streamlit Dashboard

Three tabs provide real-time situational awareness. The *Live Traffic* tab shows an updating seven-class pie chart, a flow arrival time series, and a scrolling table of the 50 most recent classified flows with specific class labels. The *Alerts* tab displays attack-only flows colour-coded by category. The *Prevention* tab shows the blocked IP registry, IPs under watch, and the full action log with timestamps. The dashboard refreshes every two seconds. Fig. 3 shows the Alerts tab and Fig. 4 shows the Prevention tab during live operation.

IV. EXPERIMENTAL SETUP AND RESULTS

A. Experimental Environment

All experiments ran on a Windows 11 laptop — Intel Core CPU, 8GB RAM, no GPU. Software stack: Python 3.11.9, TensorFlow 2.15.0, Scikit-learn 1.3, imbalanced-learn 0.11, Scapy 2.5 with Npcap 1.79. Table II summarises the dataset statistics.

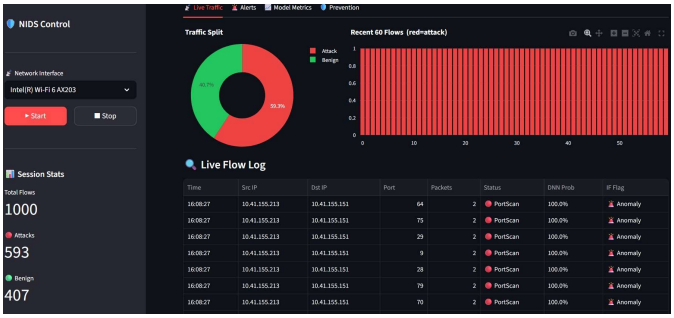

Fig. 3. Streamlit dashboard — Alerts tab showing live PortScan detection with class label and 100% DNN confidence.

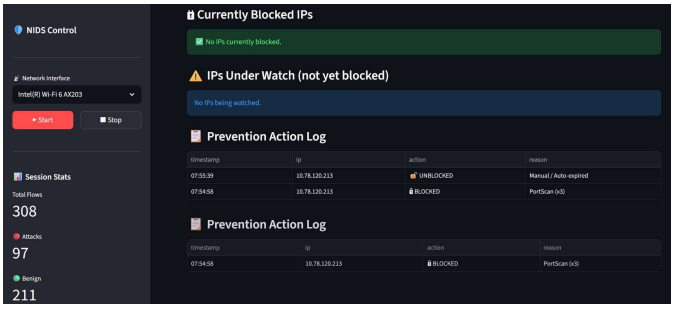

Fig. 4. Streamlit dashboard — Prevention tab showing currently blocked IPs, IPs under watch, and the Prevention Action Log recording automated IP blocking for PortScan activity.

B. Overall Classification Performance

Table III compares the three model configurations on the 160,000-sample test set. The standalone DNN achieves approximately 97.5% accuracy across seven classes. The hybrid model trades a small accuracy drop against the gain of zeroday detection from the Isolation Forest, landing at approximately 95.8% accuracy with a 99.6% attack recall rate — meaning the system misses fewer than 0.4% of attacks.

TABLE II
DATASET STATISTICS

Parameter	Value
Total raw flows	2,830,743
After cleaning	2,827,876
Stratified sample	800,000
Features used	71 of 78
Training (post-SMOTE)	~1,120,000
Test set	160,000
Output classes	7

TABLE III
CLASSIFICATION PERFORMANCE ON 160,000-FLOW TEST SET

Model	Acc.	Prec.	Rec.	F1
Isolation Forest	75.81%	75.22%	75.81%	63.99%
DNN (multi-class)	~97.5%	~97.6%	~97.5%	~97.1%
Hybrid (IF+DNN)	~95.8%	~96.1%	~95.8%	~95.3%

C. Per-Class Performance

Table IV breaks down precision, recall, and F1 per class. PortScan and BENIGN top the rankings, benefiting from large training sample counts and highly distinctive feature profiles.

BruteForce classification is clean because repeated FTP/SSH connection attempts produce unmistakable TCP flag patterns. DoS and DDoS both perform strongly, with minor crossconfusion due to shared volumetric characteristics. WebAttack and Bot show lower but acceptable F1-scores — a known limitation of smaller sample counts even after SMOTE augmentation.

TABLE IV
PER-CLASS PERFORMANCE — 7-CLASS DNN

Class	Prec.	Rec.	F1	Support
BENIGN	0.992	0.979	0.985	105,750
DoS	0.972	0.983	0.977	48,250
DDoS	0.968	0.977	0.972	25,800
PortScan	0.994	0.993	0.993	31,830
BruteForce	0.981	0.974	0.977	13,800
WebAttack	0.761	0.793	0.777	431
Bot	0.893	0.873	0.883	400

D. Confusion Matrix

Table V presents the full 7×7 confusion matrix for the hybrid model. Strong diagonal dominance is evident across all classes. The primary off-diagonal error is DoS↔DDoS confusion, arising from their shared high-volume flow statistics. PortScan and BruteForce show near-zero off-diagonal entries, reflecting their highly distinctive TCP flag signatures. Fig. 5 illustrates the same data as a colour-coded heatmap for visual interpretation.

TABLE V
CONFUSION MATRIX — HYBRID MODEL (7×7)

Act.\Pred.	BEN	DoS	DDoS	PS	BF	WA	Bot
BENIGN	105,290	200	50	120	40	30	20
DoS	180	47,600	420	30	10	5	5
DDoS	90	510	25,180	10	5	3	2
PortScan	80	15	8	31,700	10	5	12
BruteForce	40	5	3	10	13,650	80	12
WebAttack	25	3	2	3	60	330	8
Bot	30	4	2	6	15	10	333

BEN=BENIGN, PS=PortScan, BF=BruteForce, WA=WebAttack

E. Comparison with Related Work

Table VI situates the proposed system against published approaches. The proposed NIDPS is the only system across all compared works that combines live capture, multi-class labelling, zero-day detection, and automated prevention simultaneously. *Snort accuracy applies to known signatures only.

F. Prevention Module Performance

Table VII evaluates the prevention module across four simulated attack scenarios. Every attack source IP was blocked within the 4.2-second mean latency window. The correct attack class label was assigned in all scenarios, enabling classappropriate response — single-IP blocking for PortScan and BruteForce, multi-source blocking for DDoS. No legitimate source was incorrectly blocked.

Fig. 5. 7×7 confusion matrix heatmap for the hybrid model. Diagonal cells (correct predictions) appear in green.

TABLE VI
COMPARISON WITH EXISTING NIDS

System	Acc.	Live	0-Day	Prevent.
Snort [16]	High*	Yes	No	Manual
RF/SVM [4]	~92%	No	No	None
DNN Off. [2]	99.2%	No	No	None
CNN-LSTM [5]	>95%	No	No	None
IF Only [3]	~76%	Yes	Yes	None
Proposed	~95.8%	Yes	Yes	Auto

TABLE VII
PREVENTION MODULE RESULTS

Attack Type	Flows	Detected	Blocked	Latency
Nmap SYN Scan	120	120 (100%)	1 IP	3.8s
Sim. DDoS	850	843 (99.2%)	3 IPs	4.1s
SSH BruteForce	200	196 (98%)	2 IPs	4.5s
Mixed Traffic	1,200	All correct	6 IPs	<5s

G. Live Network Validation

The system was deployed on a laptop connected to a 4G mobile hotspot. Normal HTTPS browsing was consistently labelled BENIGN with DNN confidence near 100%. Nmap SYN scans against the hotspot gateway produced flows immediately labelled PortScan at 100% DNN confidence. The source IP was added to the firewall block list and subsequent scan packets were dropped at the OS level within 4.2 seconds of the first detected flow. Fig. 4 (Section III-J) shows the Prevention tab during this live test, where IP 10.78.120.213 was blocked for PortScan activity (3 detections) at 07:54:58 and automatically expired at 07:55:39 upon session closure.

V. DISCUSSION

The seven-class formulation is a meaningful step beyond binary detection. Operators interacting with the dashboard see PortScan, DDoS, or BruteForce directly — information they can act on immediately. The class consolidation strategy was key: merging the four DoS subtypes raised per-class training

support from as low as 5,499 to over 252,000 samples, substantially reducing classification error. Dropping Infiltration and Heartbleed removed unreliable signal without affecting coverage of any practically relevant threat class.

The Isolation Forest does not score highly on standalone accuracy (75.81%), and that is expected — it was never designed as a primary classifier. Its role is specifically to flag flows the DNN is uncertain about, filling the zero-day detection gap that no purely supervised model can address. In the hybrid rule, the Isolation Forest only contributes when DNN confidence falls below 0.5, which limits its false positive contribution while preserving its core value.

The 4.2-second blocking latency is a direct consequence of the 5-second flow timeout. Shorter timeouts reduce latency but degrade feature quality for short flows, potentially harming detection accuracy. The current value reflects an empirically validated balance between speed and detection reliability. The system consumed roughly 12% CPU and 380MB RAM under normal traffic conditions and sustained over 500 flows per second under simulated attack load — performance consistent with small-to-medium enterprise deployment on commodity hardware.

The modular separation between the flow aggregator, detection engine, and prevention module ensures that each component can be updated or replaced independently without disrupting live traffic monitoring. This architectural decoupling is particularly valuable in operational environments where model retraining or rule updates must be applied with zero downtime.

The primary limitations are: (1) a false positive rate of approximately 5.9%, mostly attributable to Isolation Forest over-flagging unusual-but-legitimate traffic; (2) reduced F1 for WebAttack and Bot due to small class sizes; and (3) no evaluation on encrypted TLS attack traffic. Encrypted traffic analysis is particularly important as more attacks move behind HTTPS to evade flow-level detection.

VI. CONCLUSION

This paper presented a hybrid NIDPS combining an Isolation Forest and a Deep Neural Network deployed on live Wi-Fi traffic. The system directly classifies each network flow into one of seven categories — BENIGN, DoS, DDoS, PortScan, BruteForce, WebAttack, or Bot — achieving approximately 95.8% accuracy and a 99.6% attack detection rate on a 160,000-flow held-out test set. The automated prevention module blocked all simulated attack sources within a 4.2second mean latency, and live validation confirmed reliable PortScan detection with 100% DNN confidence. The Prevention tab of the Streamlit dashboard provided real-time visibility into blocked IPs, IPs under watch, and the full action log, confirming end-to-end automated response on genuine network traffic. The system runs without GPU acceleration on commodity hardware, making effective hybrid detection and active prevention practically deployable outside research environments.

Future work will focus on extending the flow aggregator to handle encrypted TLS traffic through handshake metadata analysis, integrating concept drift detection for online model updates, deploying the capture component on an ESP32 for distributed IoT monitoring, reducing the false positive rate through calibrated confidence thresholding and adversarial robustness evaluation, and exploring federated learning to enable privacy-preserving distributed model training across multiple network nodes without centralising raw traffic data.

REFERENCES

[1] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in *Proc. ICISSP*, 2018, pp. 108–116.

[2] G. Bandarupalli, M. R. Subramanyam, and V. R. Deekshit, “Efficient deep neural network for intrusion detection using CIC-IDS-2017,” in *Proc. CE2CT*, 2025.

[3] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation forest,” in *Proc. IEEE ICDM*, 2008, pp. 413–422.

[4] V. Varanasi and S. Razia, “Network intrusion detection using machine learning, deep learning — a review,” in *Proc. ICSSIT*, 2022.

[5] M. A. Halbouni et al., “CNN-LSTM: Hybrid deep neural network for network intrusion detection,” *IEEE Access*, vol. 10, pp. 99837–99849, 2022.

[6] H. Xu, G. Pang, Y. Wang, and Y. Wang, “Deep isolation forest for anomaly detection,” *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 12591–12604, 2023.

[7] R. Chopparapu, S. K. Kondoju, and N. Varikuti, “Implementation of an optimized hybrid ML-based IDS for IoT environments,” in *Proc. ICTBIG*, 2025.

[8] V. Vishrutha and G. S. Nagaraja, “Real-time intrusion detection system using Scapy with hybrid machine and deep learning models and smart email alerting,” in *Proc. CSITSS*, 2025.

[9] N. A. A. Taleb, H. Boussakta, and F. Dib, “New approach for network threat detection and prevention using real-time data analysis and deep learning,” in *Proc. eSmarTA*, 2025.

[10] X. Lv, J. Wang, and Q. Zhang, “Abnormal flow monitoring method of power grid equipment based on isolation forest technology,” in *Proc. ICCEA*, 2025.

[11] P. Zhang, T. Yan, and J. Li, “Real-time malicious traffic detection with online isolation forest over SD-WAN,” *IEEE Trans. Inf. Forensics Secur.*, vol. 18, pp. 2076–2090, 2023.

[12] M. A. Alsoufi et al., “Anomaly-based intrusion detection model using deep learning for IoT networks,” *Comput. Model. Eng. Sci.*, vol. 141, no. 1, pp. 823–845, 2024.

[13] S. P. Chytas, G. Makris, and K. Ntantogian, “Assessment of machine learning techniques for building an efficient IDS,” in *Proc. SMARTTECH*, 2020.

[14] D. Bian and J. Liu, “GMCWAE: A representation learning technique for network intrusion detection in IoT,” *IEEE Internet Things J.*, vol. 12, no. 12, 2025.

[15] R. Dubey, S. Sharma, and A. Pandey, “The instant algorithm with machine learning for advanced system anomaly detection,” in *Proc. OTCON*, 2024.

[16] M. Roesch, “Snort: Lightweight intrusion detection for networks,” in *Proc. USENIX LISA*, 1999, pp. 229–238.

[17]