International Journal of Scientific Research & Engineering Trends
Volume 12, Issue 5, Sep-Oct-2026, ISSN (Online): 2395-566X

Intelligent Cybersecurity Threat Detection System: A Hybrid Machine Learning Approach for Known and Unknown Attacks

Yashasvi Sharma[1], Amrish Kumar[2]
Department of Artificial Intelligence and Data Science Poornima Institute of Engineering & Technology Jaipur, Rajasthan, India[1]
Department of computer engineering (AI), Poornima institute of engineering and technology[2]

Abstract — Signature-based security tools detect attacks they already know and are blind to zero-day attacks, while many machine learning intrusion detection systems still suffer from high false-positive rates and have been evaluated mainly on static, pre-recorded datasets. This paper presents an intelligent threat detection system that learns normal network behaviour, flags deviations from it, and combines supervised classifiers with an anomaly detector. A synthetic network-flow generator produces normal traffic, three known attack types (DoS flood, port scan, SSH brute force) and three attack types withheld from training (Slowloris, botnet beacon, data exfiltration). Four detectors are compared over five random runs on 18,000 test flows: signature rules, a Random Forest, an MLP neural network, and a hybrid of Random Forest, MLP and an Isolation Forest fitted on normal traffic. The hybrid detects 100% of known attacks and 68.4% of unseen attacks, against about 90% and 0.1% for signature rules, at a 3.0% false-alarm rate. Its gain is uneven: exfiltration is detected only 5% of the time. The results come from author-generated synthetic data, so they show the behaviour of the design rather than field performance. CICIDS2017, a CNN-LSTM model, SHAP explanations, detection-latency measurement and live traffic remain to be done and are not claimed. The contribution is a design and evaluation framework for detecting known and unknown attacks, not a new algorithm.

Keywords— intrusion detection, threat detection, hybrid machine learning, anomaly detection, zero-day attacks, Random Forest, Isolation Forest, explainable AI, SHAP, cybersecurity

I. INTRODUCTION

Networks are attacked continuously, and the attacks keep changing. Cybersecurity threat detection means identifying malicious activity, intrusions and anomalies in networks and systems. Traditional tools rely on static signatures and fixed rule sets. Modern threats, including zero-day attacks, polymorphic malware and Advanced Persistent Threats (APTs), can evade such defences because nothing in a signature database describes them yet.

An intelligent detection system instead uses artificial intelligence and machine learning (AI/ML) to learn what normal behaviour looks like and to flag deviations, ideally in real time. This promises faster, more accurate and more adaptive detection. It also brings its own problems. Learned detectors need representative training data, cost more to run than simple rules, can raise many false alarms, and are often hard to interpret.

This paper studies one concrete design. A hybrid detector combines a Random Forest, an MLP neural network and an Isolation Forest anomaly detector, and is compared against signature rules on known and unseen attacks. The paper is explicit about what has and has not been done. The implemented experiment uses synthetic flows written by the author. The CNN-LSTM model, the CICIDS2017 dataset, SHAP explainability, latency measurement and live-traffic testing are planned work, and no result is claimed for them.

II. BACKGROUND AND MOTIVATION

1. Evolution of threat detection

Threat detection has developed in stages. In the 1980s and 1990s, signature-based antivirus and firewalls matched traffic against known attack patterns. In the early 2000s, rule-based intrusion detection systems (IDS) such as Snort used fixed rule sets [7]. In the mid-2000s, statistical anomaly detection was introduced to catch previously unseen attacks [6]. In the 2010s, machine learning models such as Decision Trees, Support Vector Machines and Random Forests were applied to classify traffic as normal or malicious. More recently, deep learning (CNN, LSTM, Autoencoders) and behaviour analytics have enabled real-time, adaptive detection.

2. Why unseen attacks and false alarms both matter

A detector that only recognises past attacks fails exactly when a new one arrives. A detector that flags too much normal traffic

© 2026 IJSRET

wastes analyst time and is eventually ignored. The two goals pull against each other: making a detector more sensitive to unfamiliar behaviour usually raises its false-alarm rate. Any fair evaluation therefore has to report unseen-attack detection and false alarms together.

3. The need for explainability

A flagged flow is only useful if an analyst can decide quickly whether it is a real incident. Model decisions can be hard to interpret without explainability tools. This work plans to add SHAP [9] so that analysts can see which features pushed a flow towards an alert. SHAP is planned work and is not part of the results reported here.

4. Threat landscape

Three kinds of threat explain why static defences fall short. Zero-day attacks exploit weaknesses for which no signature or patch exists yet, so a signature-based tool has nothing to match. Polymorphic malware changes its own form so that each copy looks different from the last, which defeats exact pattern matching. Advanced Persistent Threats (APTs) stay in a network for a long time and imitate normal behaviour, so the individual events they generate may each look harmless. Detection that looks at behaviour over many flows, rather than at fixed byte patterns, is therefore attractive, but it must be tested on attack types it was not trained on, which is the purpose of the experiment in this paper.

5. Measuring detector quality

Accuracy alone can mislead when one class is much rarer than the other: a detector that labels every flow as normal scores high accuracy on traffic that is mostly normal while catching no attacks. For this reason intrusion detection studies report the detection rate (recall) and the false-alarm rate next to accuracy, and often precision, which is the share of raised alerts that are real attacks. The accuracy figures in this paper also depend on how many attack flows the test set contains, so that mix has to be reported with the results. The experiment reports accuracy, false alarms and detection of unseen attacks together for that reason.

III. PROBLEM STATEMENT

The design problem is to build a detector that recognises both known and unknown attacks without an unacceptable rise in false alarms. The system must address the following problems:

- Zero-day blindness: signature rules cannot detect attacks that are not in their rule set.
- False alarms: many ML-based systems still show high false-positive rates, and a detector that cries wolf loses its value.
- Generalisation: a classifier trained only on known attacks may not recognise a new attack type.
- Static evaluation: the base CNN-LSTM IDS was evaluated mainly on static, pre-recorded datasets and not fully validated on live, real-time traffic.
- Interpretability: analysts need to know why a flow was flagged.
- Data validity: results measured on generated data show how the design behaves, not how it will perform on real traffic.

IV. RESEARCH OBJECTIVES AND CONTRIBUTIONS

The objectives are:

- collect and preprocess network traffic and extract features such as packet size, protocol, flow duration and port usage (CICIDS2017 planned);
- train a hybrid ML/DL model to classify traffic as normal or malicious (tested here: Random Forest, MLP and anomaly detector; CNN-LSTM is next);
- test the detector in a simulated real-time setting and compare it with signature-based detection on accuracy, false-alarm rate and detection latency; and
- add SHAP explainability so that analysts can see why a flow was flagged.

The contributions of this paper are: (i) a six-stage detection pipeline from data collection to response and retraining; (ii) a controlled comparison of signature rules, Random Forest, MLP and a hybrid detector, with three attack types withheld from training; (iii) quantitative evidence of the trade-off between unseen-attack detection and false alarms; and (iv) an explicit statement of which components are tested and which are not. The contribution is a design and evaluation framework, not a new algorithm.

V. LITERATURE REVIEW

1. Signature-based and rule-based detection

Signature-based detection matches traffic against known attack patterns. It is fast and precise for known threats but, by construction, blind to new ones. Snort [7] is the standard open-source example of a rule-based network IDS; Suricata is a comparable open-source engine. These tools form the baseline against which learned detectors are usually judged.

© 2026 IJSRET

2. Anomaly-based detection
Anomaly-based detection learns a baseline of normal behaviour and flags deviation from it. Chandola et al. surveyed anomaly detection across application areas and describe the central difficulty: the boundary between normal and anomalous behaviour is hard to define, so false alarms are common [6]. Isolation Forest [4] isolates anomalies by random partitioning and can be fitted on normal data alone, which makes it a natural component for detecting attacks never seen in training.

3. Machine learning classifiers
Decision Trees, Random Forests [3], Support Vector Machines and XGBoost are widely applied to classify flows as normal or malicious. Supervised classifiers reach very high accuracy on attack types they were trained on. Their weakness is generalisation: they learn the attacks in the training set, not attacks in general.

4. Deep learning detectors
Deep learning models such as CNNs, LSTMs and Autoencoders learn spatial and sequential patterns in traffic. The base paper for this work studied a CNN-LSTM hybrid IDS on the NSL-KDD and CICIDS2017 benchmark datasets [1], [2], in which the CNN extracts spatial features and the LSTM models temporal attack patterns. It reports about 98% detection accuracy with a reduced false-alarm rate. Its limitation is that it was evaluated mainly on static, pre-recorded datasets rather than live traffic. The full citation of the base paper is to be added by the author as [8].

Table 1: Detection Approaches and Their Relation to this Work

Approach	Strength	Limitation relative to this work
Signature rules (e.g. Snort [7])	Fast; precise on known attacks	Blind to unseen attacks
Anomaly detection (e.g. Isolation Forest [4])	Can flag unseen behaviour; fitted on normal data only	Tends to raise false alarms
Supervised ML (RF [3], MLP)	Very high detection of trained attack types	Generalises poorly to unseen attacks
CNN-LSTM IDS (base paper)	About 98% accuracy reported on benchmarks	Mainly static datasets; not validated on live traffic; untested in this work
Hybrid (this work)	Combines classifier accuracy with anomaly coverage	Synthetic-data evidence only; higher false-alarm rate

5. Hybrid systems and datasets
Hybrid systems combine signature-based and ML-based detection, or several learned models, for balanced accuracy and coverage. SIEM and behaviour analytics correlate logs across systems for enterprise-wide visibility. Benchmark datasets matter for comparison. KDD CUP 99 has well-documented flaws [2], and CICIDS2017 was created to give a more realistic intrusion dataset [1]. Table I summarises the relationship between the approaches and this work.

6. Explainability and privacy-preserving learning
Explainability is treated as part of the detection problem rather than an afterthought. SHAP [9] assigns each input feature a contribution to a single prediction, which for an IDS would let an analyst see, for example, that a high connection rate and many SYN packets drove an alert. Federated learning [10] is a related line of work in which models are trained on data that stays at each site, so that sensitive raw traffic need not be shared. Both are future work here and are not evaluated in this paper.

The base paper reports about 98% accuracy on benchmark data, while this paper reports 92.7% accuracy for the hybrid on synthetic data. These two figures must not be compared: the datasets, the attack types, the proportion of attacks in the test set and the train/test protocol all differ.

VI. RESEARCH GAP

Prior work has produced signature systems, anomaly detectors, ML classifiers and deep learning IDS. The gap addressed here is narrower. Many systems still show high false-positive rates,

© 2026 IJSRET

and the base CNN-LSTM IDS was tested mainly on static, pre-recorded data. There is a need for a clear, reproducible comparison of signature-only, supervised and hybrid detection that (a) withholds attack types from training so that unseen-attack detection is actually measured, (b) reports false alarms next to detection, and (c) is honest about the data it was measured on. This paper offers such a comparison as a prototype evaluation. It does not claim that hybrid detection is new.

Proposed System Architecture

The proposed system has six stages (Fig. 1).

- Data collection: network traffic is collected. The experiment uses synthetic flows; CICIDS2017 and real-time packet capture are planned.
- Preprocessing and feature extraction: each flow is described by ten features, including size, ports, rate, SYN count and login attempts.
- ML/DL model: the detector scores each flow. The tested models are signature rules, Random Forest, MLP and the hybrid; CNN-LSTM is planned.
- Threat classification: each flow is labelled normal or attack.
- Alert generation: flagged flows raise alerts, which can be prioritised by threat severity.
- Response and mitigation: a response action is taken, and outcomes are used to retrain the model over time.

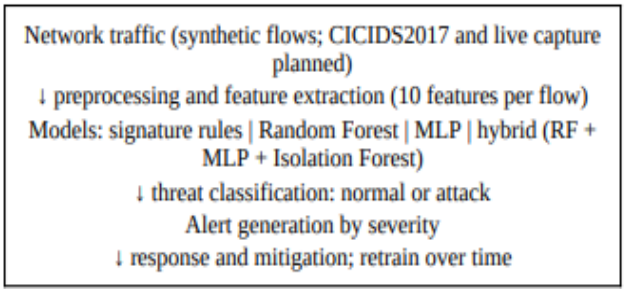


Fig. 1. Logical architecture of the threat detection system. Stages 1 to 4 are exercised in the reported experiment; stages 5 and 6 are design only.

Table 2: distinguishes what is implemented from what is planned

Component	Status in this paper
Synthetic flow generator (normal, 3 known, 3 unseen attacks)	Implemented and used
Signature rules	Implemented and evaluated
Random Forest	Implemented and evaluated
MLP neural network	Implemented and evaluated
Hybrid (RF + MLP + Isolation Forest)	Implemented and evaluated
CICIDS2017 dataset	Planned; not tested
CNN-LSTM model	Planned; not tested (the MLP stands in for it)
Detection latency in a simulated real-time setting	Planned; not measured
SHAP explainability	Planned; not implemented
Alert prioritisation, response and retraining loop	Designed; not evaluated
Live traffic	Planned; not tested

System Design and Workflow

The intended workflow is as follows:

- Network flows are collected or generated.
- Ten features are extracted per flow.
- The models are trained on normal traffic and three known attack types only.
- The trained models are tested on normal traffic, the three known attacks and three attack types never seen in training.
- Each detector's accuracy, false alarms and unseen-attack detection are recorded.
- The experiment is repeated over five random runs and the mean is reported.
- In the full design, flagged flows trigger severity-ranked alerts and a response, and outcomes feed back into retraining.

Threat Model and Assumptions

The detector is assumed to observe flow-level features of the traffic that crosses one monitored point in a network. The assumptions are:

- an attacker may use known attack types and attack types that were not present in training;
- the attacker does not control the detector, its host or its training data;
- the attacker does not adapt traffic to the detector's decision rule; and
- training uses normal traffic that is assumed to be free of attacks, as required for fitting the anomaly detector.

Several threats lie outside this model. Inspection of encrypted payload content is not attempted. An insider who acts normally with legitimate access may produce flows that look benign.

© 2026 IJSRET

Compromise of the detector host, poisoning of the training or retraining data, and deliberate evasion by an adaptive attacker are real risks, listed in Table VIII and planned for later testing.

The results hold under these assumptions only.

Data, Features and Attack Scenarios

The experiment uses a synthetic flow generator written by the author rather than captured traffic. A synthetic generator allows full control over which attack types appear in training and in testing, which is what makes the unseen-attack comparison possible. Its cost is that the classes may be easier to separate than real traffic. Table III lists the attack types and Table IV the flow features.

Table 3: Attack Scenarios in the Experiment

Attack type	Role	Typical real-world behaviour
DoS flood	Known (trained)	Very high packet or request rate towards one target
Attack type	**Role**	**Typical real-world behaviour**
Port scan	Known (trained)	Many connection attempts across many ports
SSH brute force	Known (trained)	Repeated login attempts on the SSH service
Slowloris	Unseen (test only)	Many slow, incomplete HTTP connections that hold server resources
Botnet beacon	Unseen (test only)	Small, periodic messages to a command-and-control server
Data exfiltration	Unseen (test only)	Unusually large or sustained outbound data transfer

The last column describes how these attacks usually behave in real networks. The exact parameters the generator uses for each class must be taken from the experiment code.

Table 4: Flow Features Used by the Detectors

Feature	What it captures
Size	Volume of data in a packet or flow
Ports	Which ports a flow touches
Rate	How fast packets or requests arrive
SYN	Connection-initiation (SYN) activity
Logins	Login attempts in the flow
Other five	To be listed from the generator code

Ten features are computed per flow; the five named in Table IV are those reported for the experiment. All are flow-level attributes, not packet payload contents. The planned CICIDS2017 experiments would add features such as flow duration, protocol and connection frequency.

Detection Models

Signature rules

The baseline is a set of fixed rules that matches known attack patterns. It represents a signature-based tool: effective on the attacks it was written for and blind to others.

Random Forest and MLP

The Random Forest [3] is an ensemble of decision trees trained on labelled flows. The MLP is a feed-forward neural network trained on the same data. Both are supervised and were implemented with scikit-learn [5]. In this work the MLP stands in for the planned CNN-LSTM, which remains untested.

Hybrid detector

The hybrid averages the Random Forest and MLP outputs and adds an Isolation Forest [4] fitted on normal traffic. Writing p_RF(x) and p_MLP(x) for the attack probabilities of a flow x, the classifier score is

$$s(x) = \tfrac{1}{2} \cdot (p_{RF}(x) + p_{MLP}(x)) \qquad (1)$$

and the Isolation Forest supplies an anomaly decision a(x), which is positive when x looks unlike normal traffic. A flow is reported as an attack when the classifier score or the anomaly decision indicates one. The exact combination rule and thresholds must be confirmed against the experiment code. The design intent is that the classifiers cover attacks resembling the training set, while the anomaly detector, which has never seen any attack, covers behaviour that is merely unusual.

Training and inference procedure

The conceptual procedure for one experimental run is:

- generate labelled flows for normal traffic and the three known attack types;
- train the Random Forest and the MLP on these labelled flows;
- fit the Isolation Forest on the normal flows only, so that it never sees an attack;
- generate the test flows: normal traffic, the three known attacks and the three unseen attacks;
- for each test flow, compute the classifier score s(x) and the anomaly decision a(x), and combine them into a final label;
- compare the labels with the ground truth and compute the metrics defined in the experimental setup; and
- repeat with a new random seed, five runs in total, and average the results.

© 2026 IJSRET

This is a description of the intended design. The code that implements it is the authority on details such as thresholds and parameters.

Design Rationale for the Hybrid

Each component of the hybrid has a reason, and each has a cost. Averaging two classifiers. The Random Forest and the MLP learn in different ways, so their errors need not coincide. Averaging their attack probabilities is a simple way to combine them. In the experiment the MLP caught almost no unseen attacks while the Random Forest caught 40.6%, so the average is dominated by what the Random Forest knows; the benefit of the MLP is therefore not shown by these results and would need an ablation (hybrid with and without the MLP) to establish.

Adding an anomaly detector. The Isolation Forest is fitted on normal traffic only. It needs no attack labels and so can react to behaviour that no classifier was trained to recognise. Its cost is the false-alarm increase from 0.0% to 3.0% observed here. Its decision threshold is the control that trades unseen-attack coverage against false alarms; sweeping this threshold and reporting the resulting curve is a straightforward next experiment.

Alternatives. Other anomaly detectors, such as one-class models or Autoencoders, could replace the Isolation Forest. They were not tested, and nothing here shows that the Isolation Forest is the best choice.

VII. RESEARCH METHODOLOGY

The study is a controlled comparative experiment on synthetic data.

Data. A synthetic generator written by the author produces flows for normal traffic and six attack types. Each flow has ten features: size, ports, rate, SYN count, login attempts and related flow attributes.

Train and test split by attack type. Training uses normal traffic and three known attacks: DoS flood, port scan and SSH brute force. Testing adds three attack types never seen in training: Slowloris, botnet beacon and data exfiltration. This design makes unseen-attack detection measurable.

Repetition. Each experiment is run five times with different random seeds, and the mean is reported. The test set contains 18,000 synthetic flows.

Detectors. Signature rules, Random Forest, MLP, and the hybrid of Random Forest, MLP and Isolation Forest.

Why attack types are withheld. If every attack type appears in training, a high detection rate shows only that the model memorised familiar patterns. Withholding three attack types turns the test into a question about generalisation, which is the question a zero-day attack poses in practice. The same reasoning is why the Isolation Forest is fitted on normal traffic alone.

Reproducibility. A reader should be able to repeat the experiment. This requires the generator code, the random seeds, the model parameters and the software versions to be published with the results. Where this paper does not state them, they are to be supplied from the experiment code.

Experimental Setup and Evaluation Metrics

For a given detector, let TP, FP, TN and FN denote true positives, false positives, true negatives and false negatives. The reported metrics are accuracy, the false-alarm rate (the share of normal flows wrongly flagged), and the detection rate for an attack type:

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN) \quad (2)$$

$$\text{False-alarm rate} = FP / (FP + TN) \quad (3)$$

$$\text{Detection rate} = TP / (TP + FN) \quad (4)$$

The detection rate is reported separately for the three known attack types (known caught) and for the three unseen attack types (unknown caught). The software used was scikit-learn

(RandomForestClassifier, MLPClassifier and IsolationForest). Detection latency, which the planned real-time test would measure, is not part of the reported results.

Configuration to be recorded. For each run the author should record the random seed; the number of flows per class in training and testing; the Random Forest parameters (number of trees, depth); the MLP architecture and training settings; the Isolation Forest parameters and decision threshold; the rule set used for the signature baseline; and the software versions and hardware. The reported figures are means over five runs, and the per-attack chart in the poster shows the standard deviation across those runs as error bars.

VIII. RESULTS AND DISCUSSION

© 2026 IJSRET

All figures below are means over five random runs on 18,000 synthetic test flows (Table V).

Table 5: Overall Comparison of the Four Detectors

Detector	Accuracy	False alarms	Unknown caught
Signature rules	80.1%	2.3%	0.1%
Random Forest	90.1%	0.0%	40.6%
MLP neural net	83.3%	0.1%	0.1%
Hybrid (RF + MLP + anomaly)	**92.7%**	**3.0%**	**68.4%**

Known attacks. Signature rules caught 89.6% of known attacks, and all three ML-based detectors caught between 99.9% and 100%. The hybrid caught 100%.

Unseen attacks. Signature rules and the MLP caught almost none (about 0.1%). Random Forest caught 40.6%, and the hybrid caught 68.4%. For the hybrid, Slowloris and botnet beacon were detected 100% of the time, but data exfiltration only 5% of the time.

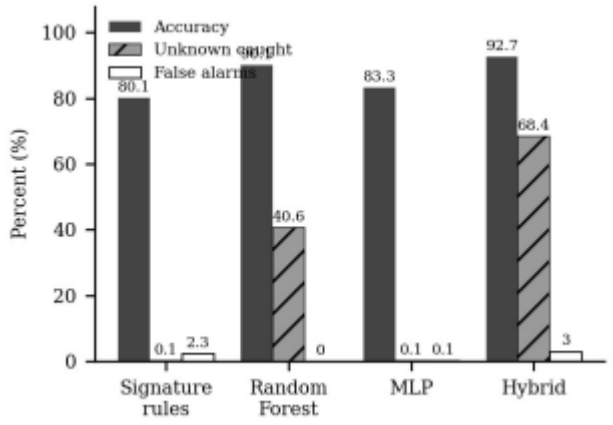


Fig. 2. Accuracy, unknown-attack detection and false alarms for the four detectors (values from Table V; mean of five runs on 18,000 synthetic test flows).

Discussion. First, learning from known attacks alone does not generalise. The MLP matched the Random Forest on known attacks yet caught almost no unseen ones, so strong performance on trained attack types says little about new ones. Second, the anomaly detector helps, at a price. Adding the Isolation Forest raised unseen-attack detection from 40.6% (Random Forest) to 68.4%, and raised false alarms from 0.0% to 3.0%. The hybrid has the highest accuracy (92.7%) but not the lowest false-alarm rate. Whether 3.0% is acceptable depends on traffic volume and analyst capacity, because a small false-alarm rate on a large volume of normal traffic still produces many alerts. Third, the gain is uneven. Exfiltration, whose flows may look closer to normal traffic, was almost missed, and a single unseen-attack figure hides this. Fourth, near-perfect known-attack scores reflect the generator. The data was written by the author, so very high scores on known attacks show that the generator produces separable classes, not that real attacks are this easy to detect.

Reading the per-attack results. The hybrid detected Slowloris and botnet beacon in full but almost missed exfiltration. A plausible explanation is that the first two differ strongly from normal flows on features such as rate or timing, while exfiltration looks close to a normal long transfer. This is a hypothesis, not a tested finding. Testing it would need per-feature distributions of each attack class and, ideally, SHAP attributions for the missed flows.

What the false-alarm rate means in operation. As an illustration only, suppose a network produces 100,000 normal flows in a day. A 3.0% false-alarm rate would raise about 3,000 false alerts, a 2.3% rate (the signature rules) about 2,300, and a 0.0% rate (the Random Forest) none. The hybrid therefore buys its extra unseen-attack coverage with a larger triage burden, and that burden must be weighed against the cost of missing a new attack. Severity-based alert prioritisation is the design response, but it has not been evaluated.

The results do not justify claims about real networks, about detection latency, or about CNN-LSTM performance. Table VI records the measurements still missing.

Table 6: Measurements Not Yet Available in this Paper

Measure	Result
Performance on CICIDS2017	Not yet measured
CNN-LSTM accuracy and false alarms	Not yet measured
Detection latency (simulated real time)	Not yet measured
Performance on live traffic	Not yet measured
SHAP feature attributions	Not yet computed
Effect of retraining over time	Not yet measured

Evaluation Plan for Remaining Work

The unfinished components can each be tied to a test, a metric and the evidence needed to report it (Table VII). Until these are done, the corresponding claims must stay out of the paper.

Table 7: Reproducible Evaluation Plan

© 2026 IJSRET

Area	Test	Metric	Evidence needed
Benchmark data	Train and test on CICIDS2017 with attack classes held out	Accuracy, false alarms, per-attack detection	Dataset version, split, seeds
CNN-LSTM	Replace the MLP with a CNN-LSTM under the same protocol	Same metrics	Architecture, training logs
Latency	Score flows in a simulated real-time stream	Median and 95th-percentile latency	Hardware, load, timestamps
Explainability	Compute SHAP for flagged and missed flows	Feature attributions; analyst review	Explainer settings
Live traffic	Run on captured live traffic	Alert volume; precision of triaged alerts	Capture setup, labelling
Retraining	Retrain with analyst feedback over time	Drift in detection and false alarms	Schedule, data used
Evasion	Craft traffic to evade the hybrid	Drop in detection rate	Attack method
Statistics	Repeat runs and report spread	Mean, standard deviation, interval	Run logs

Security and Validity Considerations

A learned detector is itself a target and a source of risk. Table VIII lists the principal risks and possible controls.

Table 8: Principal Risks and Possible Controls

Risk	Possible control	Residual limitation
High false-positive rate	Threshold tuning; alert prioritisation by severity	More sensitive detection tends to raise false alarms
Unrepresentative training data	Train on larger, realistic datasets such as CICIDS2017	Benchmarks may not match a given network
Evasion by an adaptive attacker	Combine signature, supervised and anomaly detection; retrain over time	A determined attacker may still craft traffic that looks normal
Training-data poisoning	Control and audit retraining data	Retraining from detector outcomes can itself be misled
Concept drift	Periodic retraining on recent normal traffic	Retraining cost and timing remain operational choices
Opaque decisions	SHAP explanations for flagged flows	Planned; not yet available

A second point concerns what a score means. A high accuracy on generated data is a property of the generator and the model together. Claims about field performance require real or benchmark traffic, an independent test set and, ideally, live deployment.

Deployment and Ethical Considerations

Integration. In a real deployment the detector would feed alerts into existing tools such as a SIEM, which correlates logs across systems, rather than replace them. Signature rules remain useful for known threats because they are fast and precise, so the learned detector is best seen as an additional layer.

Human oversight and automated response. The design includes an automated response and mitigation stage. An automated block based on a false alarm can cut off legitimate users, so response actions should be staged, for example alert first, then rate-limit, then block, with human review for the most disruptive steps.

Privacy. The features reported here are flow-level attributes, not payload contents, which limits the sensitivity of the data used. Real traffic can still reveal who communicates with whom, so collection, retention and access need policy controls. Federated learning [10] is a possible route to training without centralising raw traffic.

Dual use. Descriptions of how detectors fail can also help attackers who wish to evade them. This paper therefore reports detection behaviour and does not provide attack tooling.

Assessment Against the Objectives

Table IX compares the four objectives of Section IV with what has been achieved. None is fully met; the table is intended to prevent the paper from claiming more than the experiment supports.

Table 9: Status of Each Research Objective

Objective	Status	Evidence or gap
1. Collect and preprocess traffic; extract features	Partly met	Synthetic flows with ten features; CICIDS2017 and real packet capture not used
2. Train a hybrid ML/DL classifier	Partly met	RF + MLP + Isolation Forest trained and tested; CNN-LSTM not tested
3. Simulated real-time test and comparison with signature detection	Partly met	Comparison on accuracy and false alarms done; detection latency and real-time operation not measured
4. SHAP explainability	Not met	Planned; no explanations produced

© 2026 IJSRET

The comparison with signature rules, the central question of the study, is answered for the synthetic data: the hybrid detects far more unseen attacks and slightly more known ones, at a higher false-alarm rate. Whether that holds on real traffic is open.

Limitations

First, the data is synthetic and was written by the author, so near-perfect scores on known attacks reflect the generator. Second, the MLP stands in for the CNN-LSTM, which has not been tested. Third, CICIDS2017 has not been used. Fourth, detection latency and live-traffic behaviour were not measured. Fifth, SHAP explanations have not been implemented. Sixth, the hybrid raised the false-alarm rate to 3.0% and detected exfiltration poorly. Seventh, the results are means over five runs of a single design, with no claim of statistical significance. Eighth, advantages such as adaptivity and reduced manual monitoring are properties of the intended design, not measured results. Costs such as the need for large, well-labelled datasets and higher computational cost than rule-based systems apply to this approach.

Future Scope

The next steps are to evaluate on CICIDS2017, implement and test the CNN-LSTM model, add SHAP explanations of flagged flows, and measure detection latency in a simulated real-time environment and then on live traffic. Further work includes extending the system to cloud and IoT environments, adding real-time streaming data pipelines for faster response, using federated learning so that models can be trained without sharing sensitive raw data, and building an automated response and mitigation module alongside detection. Cost-aware alerting, comparison of anomaly detectors, and tests against adaptive adversaries would also strengthen the study.

IX. CONCLUSION

Traditional signature-based systems cannot keep up with rapidly evolving cyber threats, and detectors trained only on known attacks do not generalise to new ones. In a controlled experiment on synthetic network flows, a hybrid of Random Forest, MLP and Isolation Forest detected 100% of known attacks and 68.4% of unseen attacks, against about 90% and 0.1% for signature rules, with the highest accuracy (92.7%). The price was a false-alarm rate of 3.0% and weak detection of data exfiltration (5%). These findings support combining supervised and anomaly detection, and they show the need to report unseen-attack detection and false alarms together. They remain preliminary. The data is synthetic, the CNN-LSTM, CICIDS2017, SHAP, latency and live-traffic components are untested, and no claim is made about performance on real networks until those steps are completed.

REFERENCES

1. I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in Proc. Int. Conf. Information Systems Security and Privacy (ICISSP), 2018.
2. M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A detailed analysis of the KDD CUP 99 data set,” in Proc. IEEE Symp. Computational Intelligence for Security and Defense Applications (CISDA), 2009.
3. L. Breiman, “Random forests,” Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.
4. F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation forest,” in Proc. IEEE Int. Conf. Data Mining (ICDM), 2008, pp. 413–422.
5. F. Pedregosa et al., “Scikit-learn: Machine learning in Python,” J. Machine Learning Research, vol. 12, pp. 2825–2830, 2011.
6. V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” ACM Computing Surveys, vol. 41, no. 3, art. 15, 2009.
7. M. Roesch, “Snort: Lightweight intrusion detection for networks,” in Proc. USENIX Large Installation System Administration Conf. (LISA), 1999, pp. 229–238.
8. [Base paper on a CNN-LSTM based intrusion detection system: author to insert the full citation.]
9. S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” in Advances in Neural Information Processing Systems (NeurIPS), vol. 30, 2017.
10. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in Proc. Int. Conf. Artificial Intelligence and Statistics (AISTATS), 2017.

© 2026 IJSRET