

Enhancing Cybersecurity through AI-Driven Automated Threat Detection Systems: A Framework for Real-Time Adaptive Defence Mechanisms.

Michael Gyan Darling

michdarlingedu@gmail.com

University of Skills Training and Entrepreneurial Development (USTED) <https://orcid.org/0009-0004-1414-1127>

Joshua Caleb Dagadu

University of Skills Training and Entrepreneurial Development (USTED)

Umar Farouk Ibn Abdulrahman

Kumasi Technical University

Victor Tattrah

University of Skills Training and Entrepreneurial Development (USTED)

Eric Osei Opoku

Kwame Nkrumah University of Science and Technology (KNUST)

Research Article

Keywords: Intrusion detection, Real-Time detection, Binary and Multi-class intrusion, Support Vector Machine, Random Forest, CNN

Posted Date: May 26th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-9799149/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.
[Read Full License](#)

Additional Declarations: The authors declare no competing interests.

Abstract

The prevalence of cyber threats has become increasingly sophisticated and complicated; thus, it requires a smart and automated system to scan, analyse, and detect network intrusions in real-time. This study introduces a novel AI-based Network Intrusion Detection System (Realtime AI-NIDS), that scans networks for live packets, analyses, and predicts binary and multi-class intrusions using Random Forest, Support Vector Machine and Convolutional Neural Network (CNN) algorithms. The proposed Network Intrusion Detection model is developed and trained utilising the NSL-KDD dataset. In an experimental evaluation, the model achieved an impressive accuracy of 98% for normal traffic and 94.3% for U2R attacks, utilising live network packets. The study recorded exceptional precision, recall, and F1-scores for the performance of normal traffic and various threat types, with R2L achieving an impressive 98%, while U2R lagged at 42%. The CNN technique exhibited progressive learning over 100 epochs, reaching a peak accuracy of 99.61%. This indicates a high accuracy across different threat types in the test data, with DoS at 99.8% and normal traffic at 99.7%. The findings underscore the necessity of enhancing threat detection for low-frequency attacks, such as probe and R2L, and recommend the integration of this model into adaptive defense systems.

1. Introduction

The emergence of cloud infrastructures, Internet of Things (IoT) devices, interconnected systems, and critical cyber-physical systems has greatly expanded the attack nature for malevolent actors in today's digital landscape. From low-complexity, opportunistic attacks to highly sophisticated, adaptive, and targeted intrusions that can evade conventional security measures have evolved. The dynamic and complex nature of modern cyberattacks make it challenging for traditional cybersecurity methods, such as firewalls, rule-based anomaly detectors, and signature-based intrusion detection systems (IDS), to keep up (Colombo, 2024). Due to their high false-positive rates, limited scalability, and inability to identify zero-day exploits or previously undiscovered attack vectors, these legacy systems frequently lead to serious vulnerabilities in national and organisational cyber infrastructures (Mohd Syafick Effandi & Darul Ridzuan, 2014).

The emergence of Artificial Intelligence (AI), specifically Machine Learning (ML) and Deep Learning (DL), offers revolutionary techniques to overcome these constraints (Gaba et al., 2024). Large amounts of network traffic can be analysed by AI-driven intrusion detection systems, which can also spot subtle patterns that point to malicious activity (S. Wang et al., 2021) and adjust to new threats without the need for explicit reprogramming (Ozkan-Okay et al., 2024). AI models learn from historical and streaming data, which makes proactive and predictive defence capabilities possible (Rizvi, 2023), in contrast to static rule-based mechanisms. Applying supervised and unsupervised learning techniques, using Random Forests (RF), Support Vector Machines (SVM), Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNNs), to network intrusion detection has shown encouraging results in recent studies (Asad Yaseen, 2023). Achieving high accuracy, low latency, and robust generalisation across various and changing threat landscapes is still fraught with difficulties, though.

The development of unified, adaptive AI-based Network Intrusion Detection Systems (NIDS) that consistently support both binary (normal vs. attack) and multi-class (e.g., DoS, Probe, R2L, U2R) classification in real-time remains a critical research gap (Patel & Kim, 2019). Additionally, without manual intervention or complete model retraining, the majority of suggested frameworks are unable to continuously learn from novel attack patterns or adapt to changing network behaviours. In dynamic enterprise environments where threat actors are continuously improving their strategies, this limitation compromises operational viability (Tatineni, 2023).

This study fills the gap by proposing a novel AI-driven automated threat detection (RealTime AI-NIDS) framework that detects network intrusion in real-time, which will improve cybersecurity and adaptive defence mechanisms. The objective of the study is to develop a new Network Intrusion Detection System (NIDS) model that can effectively and accurately classify both binary and multi-class intrusions in a live network, thereby bridging the gap between theoretical AI models and real-world cybersecurity deployments. This aims to provide a solid, deployable solution that fortifies organisational resilience against the constantly evolving range of cyber threats.

2. Literature Review

Cybersecurity threats have changed over time as digital technology has improved, and are becoming very dangerous for people, businesses, and governments all over the world. These threats encompass a wide array of malicious activities, including malware attacks that compromise systems to exfiltrate data or disrupt operations, such as viruses, worms, trojans, and ransomware (Dieter Gollmann, 2011). WannaCry and NotPetya are two well-known ransomware attacks that have shown that cyber threats are becoming more complicated (Lella et al., 2023). Phishing and social engineering attacks are still common, and cybercriminals use dishonest strategies as spear phishing and business email compromise (BEC), to trick victims into giving out private information (DBIR, 2023). The study of Z. Q. Wang & El Saddik (2023) indicates that internal actors are responsible for approximately 60% of data breaches, underscoring the necessity for robust internal security protocols. Insider threats, whether intentional or inadvertent, pose significant risks. As shown by the SolarWinds hack and things that APT29 is said to have done, advanced persistent threats (APTs) are long-term and focused cyber campaigns that are usually planned by nation-state actors (Al-Shehari et al., 2023). They are big problems because they are unpredictable and can cause a lot of damages, and zero-day exploits target software bugs that have not been found yet; thus, developers cannot fix them (Dieter Gollmann, 2011).

To monitor, identify, and react to unauthorised activity within a network or system, intrusion detection systems (IDS) have become essential components of contemporary cybersecurity frameworks (Mudzingwa, 2012; Khraisat et al., 2019). There are two main types of threat detection systems: those that use signatures and those that look for unusual behaviour. For a long time, signature-based detection has been a key part of keeping computers safe.

2.1 Signature-Based Detection Systems

Signature-based detection is the conventional method of detecting network intrusions. It does this by comparing patterns to known attack signatures. These systems, which are used in well-known tools such as Snort and Suricata, store large amounts of suspicious command structures, exploit code patterns, and malicious packet sequences (Scarfone & Mell, 2007). Figure 1 shows how the system matches network traffic with a stored signature. Commercial solutions have a 99.9% success rate at finding catalogued malware variants, which shows that this method works very well against known threats (Allianz, 2023).

However, signature-based systems have a lot of problems in today's cyberspace because they are reactive, they are not efficient against zero-day exploits and polymorphic malware, which changes its signature with each infection (Sommer & Paxson, 2010). Encrypted traffic is a huge problem to these detection systems because SSL/TLS encryption hides packet contents from engines that match signatures (Anderson, 2021).

2.2 Anomaly-Based Detection Systems

Anomaly-based intrusion detection systems (AIDS) circumvent the problems with signature-based security by simulating normal network behaviour and identifying statistical outliers. These systems used different analytical methods to make behavioural baselines, such as statistical parametric modelling, machine learning classifiers, and protocol analysis (Scarfone & Mell, 2007). For instance, a hospital network might identify common patterns in how medical imaging systems communicate with each other. Changes in the frequency of transfers, the amount of data, or the destination IPs could be signs of an attempt to exfiltrate data. Some of the problems that AIDS faces are "concept drift," where normal behaviour changes over time, computational complexity, and adversarial attacks that subtly change traffic to look like real patterns (Apruzzese et al., 2018).

2.3 Artificial Intelligence in Cybersecurity

Because of the growing number of cyber threats, such as adversarial attacks, polymorphic malware, and zero-day exploits, traditional intrusion detection systems seem out of date (Colombo, 2024; Mills et al., 2024). Asad Yaseen (2023) says that the traditional intrusion detection system (IDS) only works with signature-based methods that compare incoming traffic to known attack patterns or anomaly-based detection that shows changes from a fixed baseline. Signature-based systems are not very good at detecting new attacks, but attacks that are already known (Kothamali et al., 2020). Anomaly-based systems can find unknown threats in theory, but they have a high false-positive rate because they rely too much on strict thresholds (Kenneth Chukwujekwu Nwafor et al., 2024) and can't adapt to changes in the network (Surya et al., 2021). These limitations got even worse with the rise of AI-driven attacks, such as adversarial machine learning, which change input data to avoid detection.

The integration of AI, particularly through machine learning (ML) and deep learning (DL) techniques, represents a transformative alternative in the cybersecurity domain, necessitating scalable and adaptable solutions (Ozkan-Okay et al., 2024). AI-integrated IDSs fix the problems with traditional detection systems by being more accurate, scalable, and adaptable (Nassif et al., 2021). ML and DL

models learn from a lot of data to get better at spotting threats all the time (Ejeofobiri et al., 2024), unlike rule-based methods. Reinforcement learning (RL) agents, for instance, continuously adjust to new attack methods by getting better at finding them based on feedback (Ubeysinghe, 2024) as AI systems work with big datasets (Kelvin Ovabor et al., 2024). For example, cloud-based deep learning models can scan through millions of network packets per second, but traditional intrusion detection systems are unable to do that (Kumar Rajaram, 2024).

According to empirical studies, IDSs that use deep learning are better than those that use traditional anomaly detection and can find network attacks with 97.2% accuracy and IoT botnet attacks with a little bit more accuracy (Deepak Gandham, 2025). Again, Deepak Gandham (2025) showed that recurrent neural networks (RNNs) can handle 123,000 events per second with an accuracy of over 95%, which is an improvement over traditional rule-based system. Additionally, benchmarks utilising the CIC-IDS2017 dataset indicate that hybrid AI models, which integrate both supervised and unsupervised methodologies (Mughal, 2018), can diminish false positives by 40–60%. These characteristics illustrate how AI can transform cybersecurity by facilitating proactive, data-driven defensive strategies (Ubeysinghe, 2024).

In Big Data environments, modern networks create terabytes of data every day, which is more than what traditional systems can handle. Notwithstanding, AI processes this data at speeds that have never been seen before (Kumar Rajaram, 2024).

2.4 Hybrid AI-Driven Detection Architecture

The combination of machine learning and deep learning has changed how well intrusion detection works, especially when it comes to finding complicated multi-stage attacks. Modern hybrid systems use AI models to find new threats and signature matching to find known threats, to make layered defensive architectures (Kelvin Ovabor et al., 2024).

Empirical studies indicate that long short-term memory (LSTM) networks attain a 94% recall rate for advanced persistent threat (APT) detection, underscoring their efficacy in recognising temporal attack patterns within network flow data (Kim et al., 2021). Graph neural networks (GNNs) can also find command-and-control communications and lateral movement that are hard to find by looking at how network entities are related to each other (Kute et al., 2021).

3. The Proposed Realtime AI-based Network Intrusion Detection Model (Realtime AI-NIDS)

This study proposes a novel AI-based network intrusion detection model for detecting intrusions in live networks, utilising Random Forest, Support Vector Machine, and Convolutional Neural Network (CNN) techniques to monitor network packets in real-time. The main goal is to detect intrusion or strange behaviour directly from live packets that are captured, rather than from datasets extracted from a simulated environment and saved over time.

3.1 Description

The proposed model is made up of several parts that work together: data capture, preprocessing and feature extraction, designing and training the CNN algorithm, making predictions in real time, and getting feedback to improve.

Python-based packet sniffing tools and libraries are used to capture the raw network traffic from Ethernet and Wi-Fi interfaces. Through libraries such as Scapy and PyShark, which give access to network interfaces for real-time packet interception, the packet capture and buffering component uses Python commands. Without the need for additional hardware, these tools allow packets to be collected straight from the system's wired (Ethernet) and wireless (Wi-Fi) network adapters.

The processes involved in the proposed real-time network intrusion detection model (Realtime AI-NIDS) are depicted in Fig. 2 (Block diagram of the proposed NIDS); they include:

1. Loading the Dataset (NSL-KDD benchmark dataset).
2. Preprocessing the dataset.
3. Training the algorithms with the NSL-KDD dataset.
4. Saving the state to have a new model.
5. Testing the model (Intrusion classification).
6. Capturing network packets.
7. Preprocess the live network packets.
8. Experiment with the saved model using the new captured packets.
9. Test for intrusion.
10. Classify intrusion

To create the real-time model, the NSL-KDD benchmark dataset is used to train the Random Forest, Support Vector Machine and CNN algorithms after preprocessing, making it suitable for Machine Learning and Deep learning model implementation. The test dataset is used to evaluate the model's accuracy. The state is saved and used as a newly developed model, which can easily be copied as a single file by any researcher who might be interested in it.

3.2 Phase One of the Model

The workflow of the proposed AI-based network intrusion detection model is categorised into two phases, as depicted in Figs. 3 and 4. Phase one (Fig. 3) involves loading the dataset, training, and testing machine learning and deep learning methods to identify network threats. This phase is the most crucial part of the study, as it generates the intelligence that will be utilised in real-time detection systems. The diagram shows a systematic process that starts with gathering datasets and ends with creating a high-performance classification model that is ready to be used in the next stage of the study.

Phase one of the Proposed Model

3.3 Phase Two: Application on Live network

Figure 4 illustrates how data packets are captured from a live network, preprocessed, and then passed through the saved model from Phase One for classification.

Figure 4 shows the rest of the architectural design for the real-time network threat detection system. It also shows the whole process, from capturing live packets to using the already-trained and saved AI model to classify them intelligently.

The process begins at the live network interface, where devices that are connected to a local or wide-area network are always sending and receiving network traffic, which includes data generated by system services, user applications, and potentially malicious entities attempting to obtain unauthorised access or execute cyberattacks. The system utilises a packet-capturing module that uses low-level packet sniffing libraries such as Scapy in Python to actively listen to a designated network interface, both on Ethernet and Wi-Fi, to monitor this traffic.

This part captures all incoming and outgoing packets so that important metadata, such as source and destination IP addresses, port numbers, protocol types (TCP, UDP, and ICMP), packet sizes, and transmission flags, can be kept. Then, the packets are sent to the next stage of the pipeline for processing. After the capturing, the raw packets go through a lot of preprocessing and feature extraction techniques to make sure they work with the trained model. Some of the most important tasks at this stage are filtering out incomplete or unnecessary packets, normalising time-based fields, encoding categorical variables (protocol and service types) into numerical representations, and scaling numerical features using a pre-fitted scaler (StandardScaler) to keep the distribution of the training data consistent.

The system extracts a predetermined set of features, such as duration, source and destination bytes, flag usage, connection frequency, and service type, which are the same features used in the NSL-KDD dataset. By putting these features into a fixed-length vector that looks like the model's expected input format, unstructured network traffic can be turned into a standard format that the model can use. The preprocessed feature vector is then given to the AI-driven trained classification model that was saved and improved during Phase One of the study.

During this inference phase, the model analyses the input vector and figures out how likely it will belong to one of the following classes: Normal, Denial-of-Service (DoS), Probe, Remote-to-Local (R2L), or User-to-Root (U2R). The system chooses the threat category that is most likely based on the highest confidence score. The output is a probability distribution across these classes. The final part of the framework is the classification output, which explains the prediction result.

4. Experimentation and Results

The process begins with loading the NSL-KDD dataset, which is a well-known standard and benchmark dataset in intrusion detection research. The labelled network traffic records in this dataset include both

normal communication patterns and different types of cyberattacks, such as Denial-of-Service (DoS), Probe, Remote-to-Local (R2L), and User-to-Root (U2R) attacks. To load the dataset into the computational environment, the data handling library Pandas was used. This lets you analyse and change the data in a structured way.

The next step sets up a thorough data preprocessing pipeline that changes the raw records into a format that machine learning algorithms can use. This is because the dataset has both numerical and categorical features. During preprocessing, several important tasks are done, which include normalising numerical attributes with tools such as StandardScaler, dealing with missing values (which are few in NSL-KDD), encoding categorical variables (protocol type and service) into numbers with one-hot encoding, and getting rid of features that are not needed or are the same as others. This step makes sure that the data meets the assumptions of the learning algorithms and that all of the features are on the same scale. It is very important to keep the training and testing splits consistent so that the performance evaluation stays accurate.

After preprocessing, the dataset is split into training and validation subsets, usually in a 75:25 ratio. This makes it easier for the model to learn and for objective performance evaluation. The training set is used to fit the chosen machine learning models, which are Support Vector Machine (SVM), Random Forest (RF), and Convolutional Neural Network (CNN), whereas the testing set is reserved until evaluation. Each model is trained to find the basic patterns that set normal traffic apart from different types of attacks by using the fact that the dataset is labelled, which improves classification accuracy.

Hyperparameter tuning is done to get the best performance of traditional machine learning models of SVM and RF. The CNN's deep learning architecture has convolutional, pooling, and fully connected layers, which automatically pull out spatial and hierarchical features from the input data. We use metrics of accuracy, loss, precision, recall, and F1-score to keep track of how well the model is doing over several epochs of training on both the training and validation sets.

Learning curves are looked at to find signs of overfitting or underfitting, and changes are made as needed. After training, each method is put through a full test set evaluation. To find the best classifier, we use confusion matrices, precision, recall, F1-scores, and overall accuracy to look at and compare performance metrics.

4.1 Experimental setup

We use Python libraries such as Scikit-learn (for SVM and RF), TensorFlow/Keras (for CNN), and Pandas/Numpy (for data manipulation) to run all of our experiments. The hardware setup had a computer with an Intel Core i5 processor, 8GB of RAM, and Intel(R) Core (TM) i5-6300U CPU @ 2.40GHz for the experiment and a Router to create a simple network using 5 different virtual machines running Linux Kali, Windows 11, Windows 10 and Windows Server 2016 operating systems.

4.2 Results

The following metrics are used to measure how well the model works for both binary and multiclass classification (Ejeofobiri et al., 2024 ; Reddy Chirra, 2024)

- Precision: $TP / (TP + FP)$ • Accuracy: $(TP + TN) / (TP + TN + FP + FN)$
- Recall (Sensitivity): $TP / (TP + FN)$
- F1-Score = $2 * (Precision * Recall) / (Precision + Recall)$

A confusion matrix is created for each class in detail, especially for multi-class prediction.

4.3.1 Performance Matrix

Performance of the proposed novel network intrusion detection model is summarised in Table 1.

Table 1 Model Overall Testing Accuracies from Techniques.		
No.	Technique	Testing Accuracy
1	Support Vector Machine (SVM)	97.3%
2	Random Forest (RF)	99.9%
3	Convolutional Neural Network (CNN)	99.6%
Live Network Packet Classification		
4	Convolutional Neural Network (CNN)	98.9%

A total of 11,628 packets of live network were captured and pre-processed, which were then sent into the saved CNN-based model for classification and the following results were recorded.

Table 2 Threats found in the live network packets.		
Classes	Mapping Threat	Accuracy
Class 0	Normal	0.989
Class 1	DoS	0.795
Class 2	Probe	-
Class 3	R2L	-
Class 4	U2R	0.943

4.3.2 Receiver Operating Characteristic (ROC)

The Receiver Operating Characteristic graph is a method for organising, visualising, and choosing classifiers based on their performance. In signal detection theory, ROC graphs have long been used to

illustrate the trade-off between classifiers' hit and false alarm rates (Fawcett, 2006).

It represents a plot of a binary classification model's True Positive Rate (TPR) (sensitivity) against its False Positive Rate (FPR) (1-specificity) at different classification thresholds (Hoo et al., 2017). Models perform better when the curve shifts towards the top-left corner of the graph, indicating higher accuracy. This curve graphically depicts the trade-off between correctly identifying positive cases and incorrectly identifying negative cases. The model's overall discriminatory power can be measured numerically using the Area Under the Curve (AUC), where 0.5 denotes random chance, and 1 denotes a perfect model (Gonçalves et al., 2014).

The Receiver Operating Characteristic (ROC) curve for the created classification model is shown in Fig. 5. The suggested model's ROC curve, which shows remarkable classification ability, is shown to be well above the diagonal and approaching the top-left corner. Near-perfect performance is indicated by an AUC value near 1.0, which measures the model's overall capacity to discriminate between classes. These findings imply that the model is capable of accurately differentiating between positive and negative occurrences and indicate the robustness and generalisability of the model (Fawcett, 2006).

4.3.3 Comparative Analysis

The Proposed model achieved 97.3% accuracy with incredible performances of precision, recall and F1-score as compared with Lee et al. (2019) which attained an accuracy of 89.7% in an experiment utilising "conventional machine learning" techniques with the NSL-KDD dataset. These show how well the proposed model in this study performed and how it did better than other models, as shown in Tables 3 and 4.

Table 3
Comparison of results of other SVM models.

Author	Technique	Accuracy	Precision	Recall	F1-Score
		%	%	%	%
"Proposed model"	SVM	97.3	96.3	97.9	97.0
Buczak & Guven (2016)	SVM	94	-	-	-
Lee and Kim (2019)	SVM	89.7			

Again, the Random Forest method for binary classification had an accuracy of 99.9%, a running time of 14.2 seconds, and a precision of 99.8%, a recall of 99.9%, and an F1-Score of 98.8%. The proposed model's performance is much better than that of a similar experiment done by (Liu et al., 2021), which had an accuracy of 85.24%.

The RF technique developed by Farzaan et al. (2020) chalked up 90.2% accuracy for binary classification, whereas Lee et al. (2019) obtained 99.8% accuracy, 99.7% precision, 99.8% recall, and 99.7% F1-score, with 99.5% accuracy achieved in their cross-validation experiment. The performance of this study for RF

aligns with Lee et al. (2019) in terms of performance comparison, yet it surpassed other models, as summarised in Table 4.

Table 4
Comparison of results of other RF models for Binary classification.

Author	Technique	Accuracy	Precision	Recall	F1-Score
		%	%	%	%
“Proposed model”	RF	99.9	99.8	99.8	98.8
Lee et al., (2019)	RF	99.8	99.7	99.8	99.7
Liu et al., (2021)	RF	85.2	-	-	-
Farzaan et al., (2020)	RF	90.2	-	-	-
Al-Shehari et al., 2023	RF	99.0			

In the realm of multi-class classification, this study attained remarkable outcomes, achieving an overall accuracy of 99.8% in 17.5 seconds through the application of the Random Forest technique. This indicates enhanced performance relative to other models employing RF on the same NSL-KDD dataset, achieving 95.66% accuracy as documented by (B. X. Wang et al., 2022).

Similarly,Mills et al. (2024) obtained 99.84% accuracy, andBagaa et al. (2020) achieving 98% accuracy for multi-class classification in their experiment on the NSL-KDD dataset. This shows how the proposed model performed even though it had slightly different threat support as indicated in Table 5.

Table 5
Results comparison with other models for multi-class classification.

No.	Authors	DoS	Probe	R2L	U2L
1.	Proposed model	1.00	1.00	1.00	1.00
	Support	11,484	2,947	274	15
2.	(Mills et al., 2024)	1.00	1.00	0.97	1.00
	Support	9,234	2,289	209	11
3.	Bagaa et al., (2020)	1.00	0.99	0.82	0.99
	Support	-	-	-	-

The model's capacity to detect network threats from real-time packet capture data demonstrates significant advantages and notable disadvantages that require thorough consideration (as illustrated in Table 2). The evaluation results show that the model does well in some classes but not so well in others. The observed performance is critically examined, and the implications of the findings are interpreted.

The model is very reliable at telling the difference between bad and good network activity, with an amazing 98.9% accuracy rate for normal traffic. This high performance is to be expected because normal traffic usually has repeating and consistent patterns in things like protocol distribution, packet size, and inter-arrival time. CNNs are good at capturing these patterns with their convolutional filters. To cut down on false positives, which can lead to alert fatigue and loss of trust in the system in operational settings, you need to be able to correctly identify almost all benign connections. This result shows that the model has learnt the network's normal behaviour well, which is a good start for finding intrusions. The model was only 79.5% accurate at finding Denial-of-Service attacks, which is much lower than its accuracy at finding regular traffic.

The model also gets an unexpectedly high accuracy of 94.3% for U2R (User to Root) attacks, which are some of the rarest and hardest to find. U2R attacks happen when a local user exploits a flaw in the system to gain root-level access. These attacks often leave few traces on the network. However, other types of threats were not found in the captured dataset, so no prediction was made.

5. Conclusion

This study anticipated the categorisation of threat types from a live network packet, attaining an accuracy of 99.6%, with 94.3% for U2R attacks and 79.5% for DoS attacks from the captured and pre-processed packets. Sadly, the prediction during the experiment did not show any probing or R2L attacks.

In summary, all three methods used in the study showed that they could accurately classify both binary and multi-class intrusions. With an accuracy of 97.3% for SVM, 99.9% for RF, and 99.6% for CNN, even for live network packets, this model is robust enough to be recommended to be adopted on a larger scale in cybersecurity, as cyber threats are growing, becoming complex and sophisticated by the day.

To enhance the efficacy, adaptability, and utility of real-time intrusion detection systems, forthcoming research should focus on the following principal domains:

1. Development of Adaptive and Online Learning Models:

The evaluated models exhibit limited adaptability to evolving attack patterns within dynamic network environments, primarily due to their training on static datasets and operation in batch mode. To facilitate real-time knowledge updates for models as new data emerges, forthcoming research ought to explore online learning and incremental learning methodologies. This would make the system more responsive and effective in the long run by letting it find new threats, like zero-day attacks, without having to retrain completely.

2. Combining different techniques to detect threats:

Current models do well against known attack types but not so well against rare or hidden attacks, such as R2L and U2R. This is because there is a big difference between the number of classes and the number of samples that represent each class. Combining supervised learning with unsupervised or semi-supervised methods such as autoencoders, clustering, or anomaly-based detection, to make hybrid detection frameworks, is a promising way to find new and rare threats more easily.

3. Deployment Optimisation for Edge and Resource-Constrained Environments:

Even though deep learning models such as CNNs are very accurate, they need a lot of processing power and are often not suitable for use on edge devices or networks with limited processing power. Future research should focus on model compression, lightweight architecture design, and edge-based inference optimisation to allow IoT devices, routers, or local network gateways to detect threats in real time.

References

1. Allianz (2023) *Cyber security trends 2023*
2. Al-Shehari T, Al-Razgan M, Alfakih T, Alsowail RA, Pandiaraj S (2023) Insider Threat Detection Model Using Anomaly-Based Isolation Forest Algorithm. *IEEE Access* 11:118170–118185. <https://doi.org/10.1109/ACCESS.2023.3326750>
3. Anderson R (2021) *Security Engineering: A Guide to Building Dependable Distributed Systems*. John Wiley & Sons Inc
4. Apruzzese G, Colajanni M, Ferretti L, Guido A, Marchetti M (2018) On the effectiveness of machine and deep learning for cyber security. *International Conference on Cyber Conflict, CYCON, 2018-May*, 371–389. <https://doi.org/10.23919/CYCON.2018.8405026>
5. Asad Yaseen (2023) *AI-DRIVEN THREAT DETECTION AND RESPONSE: A PARADIGM SHIFT IN CYBERSECURITY* Asad Yaseen. <https://doi.org/https://orcid.org/0009-0002-8950-0767>
6. Bagaa M, Taleb T, Bernabe JB, Skarmeta A (2020) A Machine Learning Security Framework for IoT Systems. *IEEE Access* 8:114066–114077. <https://doi.org/10.1109/ACCESS.2020.2996214>
7. Buczak AL, Guven E (2016) A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE Commun Surv Tutor* 18(2):1153–1176. <https://doi.org/10.1109/COMST.2015.2494502>
8. Colombo M (2024) *Automated Attack Response Through Reinforcement Learning*
9. DBIR (2023) *DBIR 2023 Data Breach Investigations Report Public Sector Snapshot*
10. Deepak Gandham (2025) Understanding AI-Driven Threat Detection and Response Systems: A Technical Deep Dive. *Int J Sci Res Comput Sci Eng Inform Technol* 11(1):3074–3079. <https://doi.org/10.32628/CSEIT251112324>
11. Dieter Gollmann (2011) *Computer Security THIRD EDITION*

12. Ejeofobiri CK, Victor-Igun OO, Okoye C (2024) AI-Driven Secure Intrusion Detection for Internet of Things (IOT) Networks. *Asian J Math Comput Res* 31(4):40–55.
<https://doi.org/10.56557/ajomcor/2024/v31i48971>
13. Farzaan MAM, Ghanem MC, El-Hajjar A, Ratnayake DN (2020) *AI-Enabled System for Efficient and Effective Cyber Incident Detection and Response in Cloud Environments*
14. Fawcett T (2006) An introduction to ROC analysis. *Pattern Recognit Lett* 27(8):861–874.
<https://doi.org/10.1016/J.PATREC.2005.10.010>
15. Gaba S, Budhiraja I, Kumar V, Martha S, Khurmi J, Singh A, Singh KK, Askar SS, Abouhawwash M (2024) A Systematic Analysis of Enhancing Cyber Security Using Deep Learning for Cyber Physical Systems. *IEEE Access* 12:6017–6035. <https://doi.org/10.1109/ACCESS.2023.3349022>
16. Gonçalves L, Subtil A, Oliveira MR, De P, Bermudez Z (2014) ROC CURVE ESTIMATION: AN OVERVIEW. In *REVSTAT-Statistical Journal* (Vol. 12, Number 1)
17. Hoo ZH, Candlish J, Teare D (2017) What is an ROC curve? *Emerg Med J* 34(6):357–359.
<https://doi.org/10.1136/emered-2017-206735>
18. Kelvin Ovabor IO, Sule-Odu T, Atkison Adetutu Temitope Fabusoro, & Joseph Oluwaseun Benedict. (2024). AI-driven threat intelligence for real-time cybersecurity: Frameworks, tools, and future directions. *Open Access Res J Sci Technol*, 12(2), 040–048.
<https://doi.org/10.53022/oarjst.2024.12.2.0135>
19. Nwafor KC, Ikudabo AO, Onyeje CC, Daniel OT, Ihenacho (2024) Mitigating cybersecurity risks in financial institutions: The role of AI and data analytics. *Int J Sci Res Archive* 13(1):2895–2910.
<https://doi.org/10.30574/ijrsra.2024.13.1.2014>
20. Khraisat A, Gondal I, Vamplew P, Kamruzzaman J (2019) Survey of intrusion detection systems: techniques, datasets and challenges. *Cybersecurity* 2(1). <https://doi.org/10.1186/s42400-019-0038-7>
21. Kim H, Lee Y, Lee E, Lee T (2021) Cost-Effective Valuable Data Detection Based on the Reliability of Artificial Intelligence. *IEEE Access* 9:108959–108974.
<https://doi.org/10.1109/ACCESS.2021.3101257>
22. Kothamali PR, Banik S, Nadimpalli SV (2020) Introduction to Threat Detection in Cybersecurity. In *Int J Adv Eng Technol Innovations* (01)
23. Kumar Rajaram S (2024) Educational Administration: Theory Pract 285–296.
<https://doi.org/10.53555/kuey.v28i4.7529>. AI-Driven Threat Detection: Leveraging Big Data For Advanced Cybersecurity Compliance
24. Kute DV, Pradhan B, Shukla N, Alamri A (2021) Deep Learning and Explainable Artificial Intelligence Techniques Applied for Detecting Money Laundering-A Critical Review. *IEEE Access*, vol 9. Institute of Electrical and Electronics Engineers Inc, pp 82300–82317.
<https://doi.org/10.1109/ACCESS.2021.3086230>
25. Lee J, Kim J, Kim I, Han K (2019a) Cyber Threat Detection Based on Artificial Neural Networks Using Event Profiles. *IEEE Access* 7:165607–165626. <https://doi.org/10.1109/ACCESS.2019.2953095>

26. Lee J, Kim J, Kim I, Han K (2019b) Cyber Threat Detection Based on Artificial Neural Networks Using Event Profiles. *IEEE Access* 7:165607–165626. <https://doi.org/10.1109/ACCESS.2019.2953095>
27. Lella I, Tsekmezoglou E, Theocharidou M, Magonara E, Malatras A, Naydenov S, Rossen., Ciobanu (2023) ENISA threat landscape 2023: July 2022 to June 2023. ENISA, Cosmin
28. Liu C, Gu Z, Wang J (2021) A Hybrid Intrusion Detection System Based on Scalable K-Means+ Random Forest and Deep Learning. *IEEE Access* 9:75729–75740. <https://doi.org/10.1109/ACCESS.2021.3082147>
29. Mills GA, Acquah DK, Sowah RA (2024) Network Intrusion Detection and Prevention System Using Hybrid Machine Learning with Supervised Ensemble Stacking Model. *Journal of Computer Networks and Communications*, 2024. <https://doi.org/10.1155/2024/5775671>
30. Effandi MS, B. B., Ridzuan D (2014) P. *NETWORK INTRUSION DETECTION SYSTEM*
31. Mughal AA (2018) *JAMM Journal of Artificial Intelligence and Machine Learning in Management Artificial Intelligence in Information Security: Exploring the Advantages, Challenges, and Future Directions*
32. Nassif AB, Talib MA, Nasir Q, Albadani H, Dakalbab FM (2021) Machine Learning for Cloud Security: A Systematic Review. *IEEE Access*, vol 9. Institute of Electrical and Electronics Engineers Inc, pp 20717–20735. <https://doi.org/10.1109/ACCESS.2021.3054129>
33. Ozkan-Okay M, Akin E, Aslan O, Kosunalp S, Iliev T, Stoyanov I, Beloev I (2024) A Comprehensive Survey: Evaluating the Efficiency of Artificial Intelligence and Machine Learning Techniques on Cyber Security Solutions. *IEEE Access* 12:12229–12256. <https://doi.org/10.1109/ACCESS.2024.3355547>
34. Patel N, Kim E (2019) AI-Driven Threat Detection: Enhancing Cloud Security with Cutting-Edge Technologies the Creative Commons Attribution License (CC BY 4.0) (<http://creativecommons.org/licenses/by/4.0>). In *International Journal of Trend in Scientific Research and Development*. www.ijtsrd.com/papers/ijtsrd29520
35. Reddy Chirra B (2024) *AI-Driven Threat Detection Systems: A Paradigm Shift in Cybersecurity*
36. Rizvi M (2023) Enhancing cybersecurity: The power of artificial intelligence in threat detection and prevention. *Int J Adv Eng Res Sci* 10(5):055–060. <https://doi.org/10.22161/ijaers.105.8>
37. Scarfone KA, Mell PM (2007) *Guide to Intrusion Detection and Prevention Systems (IDPS)*. <https://doi.org/10.6028/NIST.SP.800-94>
38. Sommer R, Paxson V (2010) Outside the closed world: On using machine learning for network intrusion detection. *Proceedings - IEEE Symposium on Security and Privacy*, 305–316. <https://doi.org/10.1109/SP.2010.25>
39. Surya S, Dandyala M, Banik S (2021) Traditional Methods of Threat Detection. In *Int J Adv Eng Technol Innovations* (01)
40. Tatineni S (2023) AI-Infused Threat Detection and Incident Response in Cloud Security. *Int J Sci Res (IJSR)* 12(11):998–1004. <https://doi.org/10.21275/sr231113063646>

41. Ubeyasinghe R (2024) *AI-Powered Threat Detection in Cybersecurity: A Comprehensive Review*. <https://www.researchgate.net/publication/387271355>
42. Wang BX, Chen JL, Yu CL (2022) An AI-Powered Network Threat Detection System. *IEEE Access* 10:54029–54037. <https://doi.org/10.1109/ACCESS.2022.3175886>
43. Wang S, Balarezo JF, Kandeepan S, Al-Hourani A, Chavez KG, Rubinstein B (2021) Machine learning in network anomaly detection: A survey. *IEEE Access* 9:152379–152396. <https://doi.org/10.1109/ACCESS.2021.3126834>
44. Wang ZQ, El Saddik A (2023) DTITD: An Intelligent Insider Threat Detection Framework Based on Digital Twin and Self-Attention Based Deep Learning Models. *IEEE Access* 11:114013–114030. <https://doi.org/10.1109/ACCESS.2023.3324371>

Figures

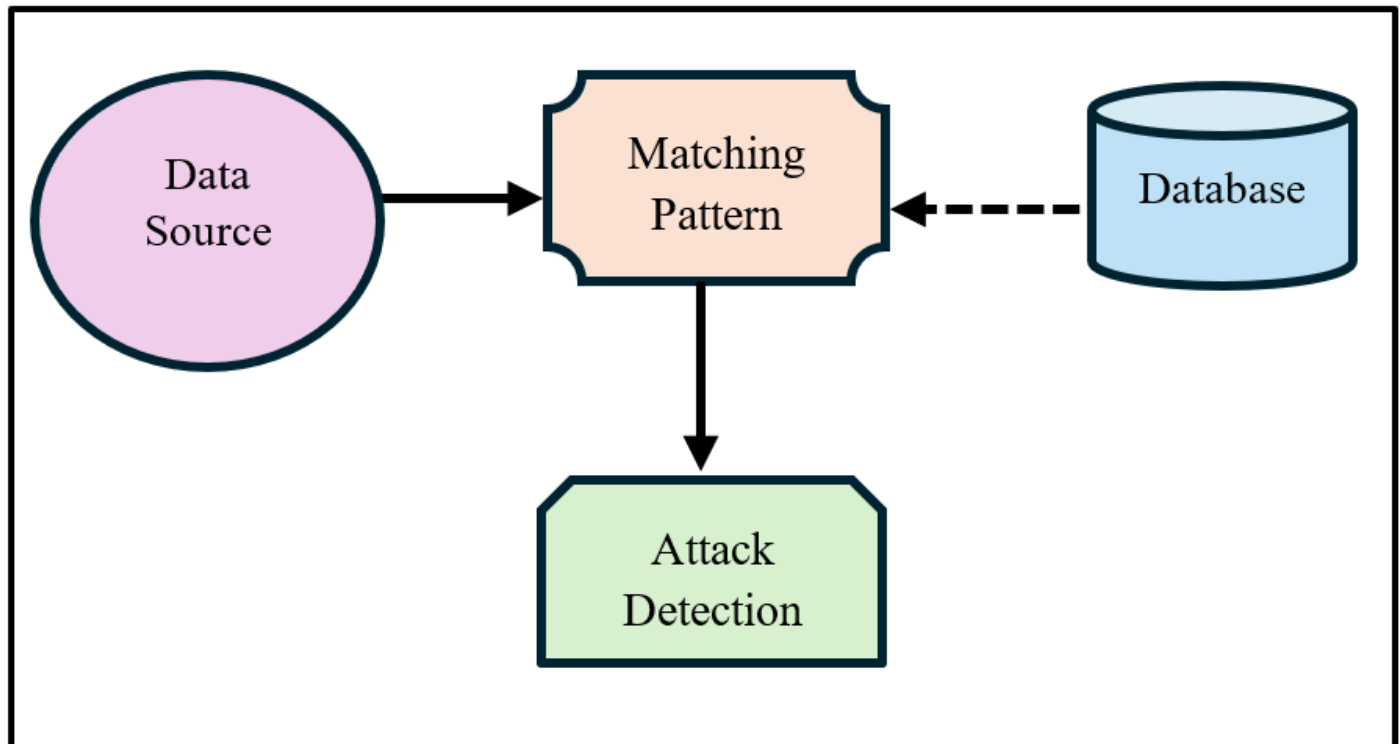


Figure 1

Signature-based detection.

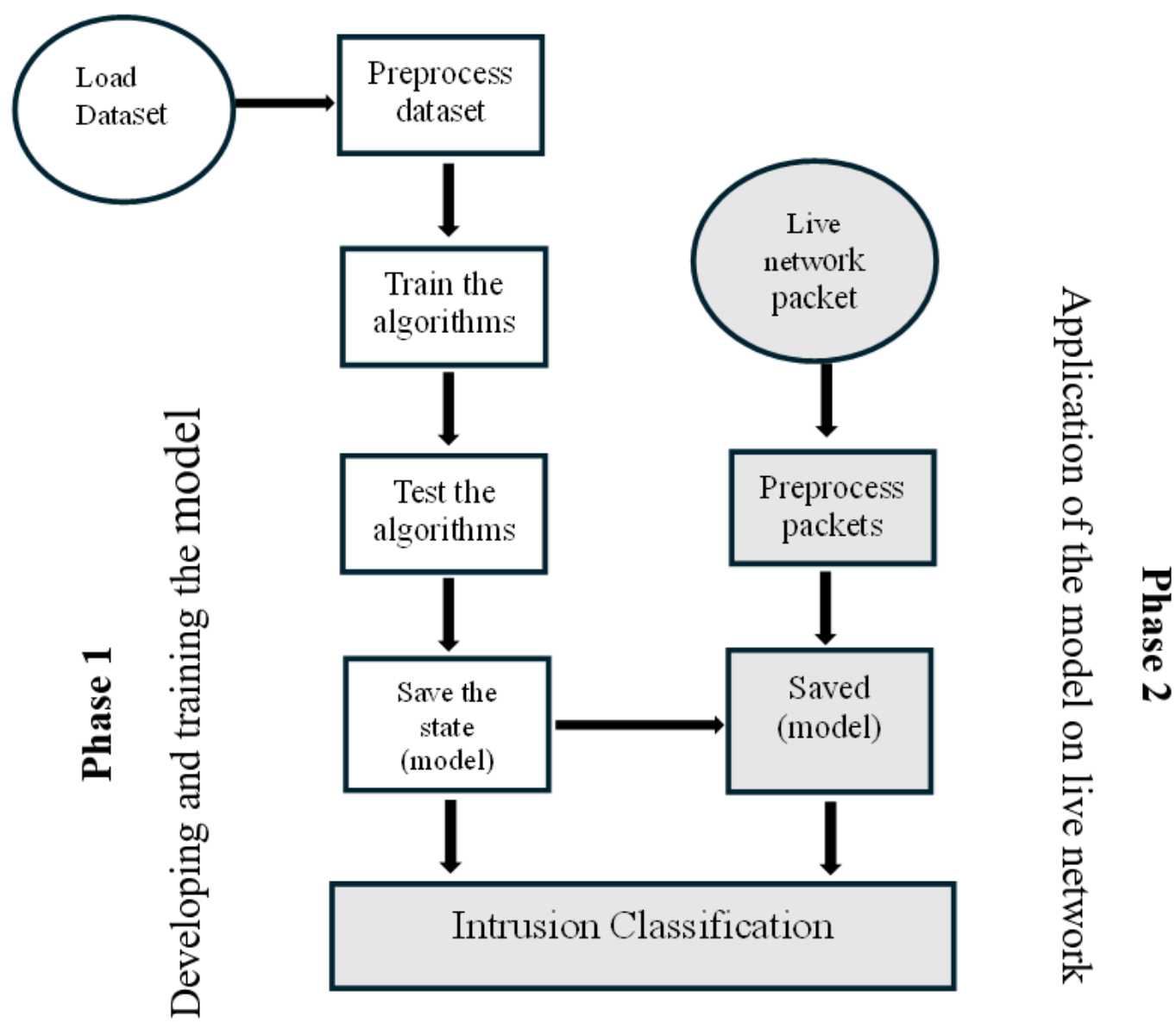


Figure 2

Block diagram of the novel NIDS.

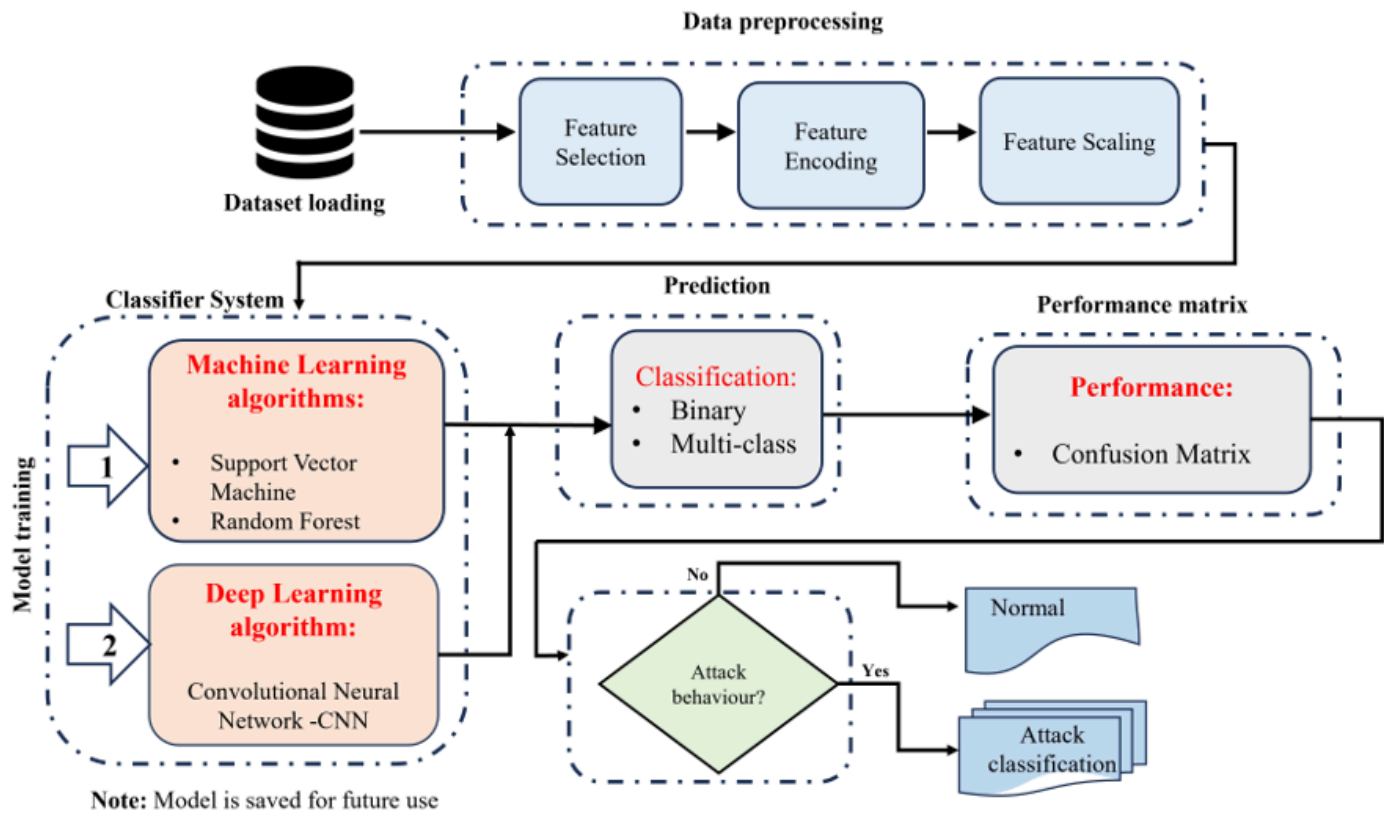


Figure 3

Developing the model.

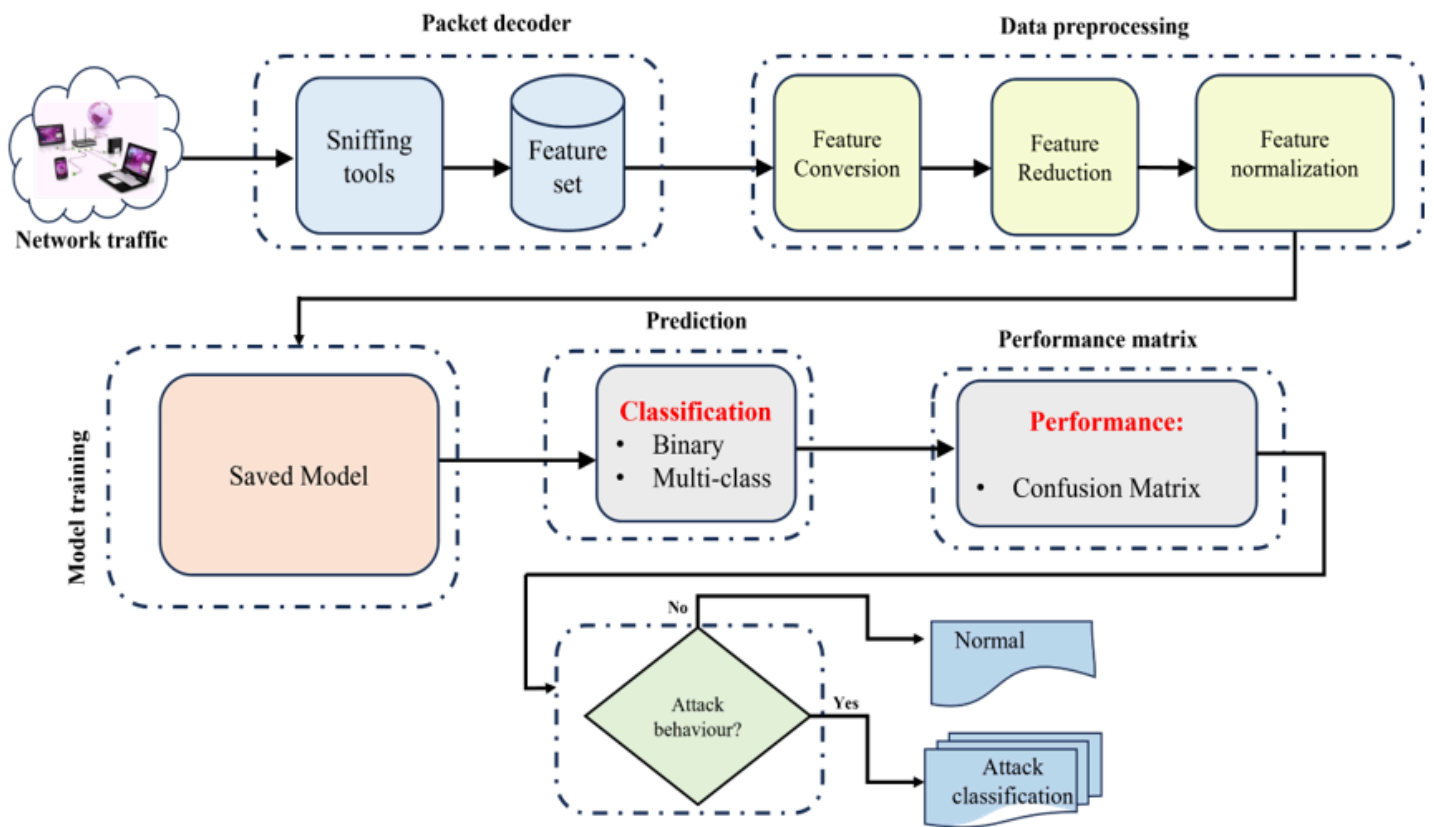


Figure 4

Application of model on live Network.

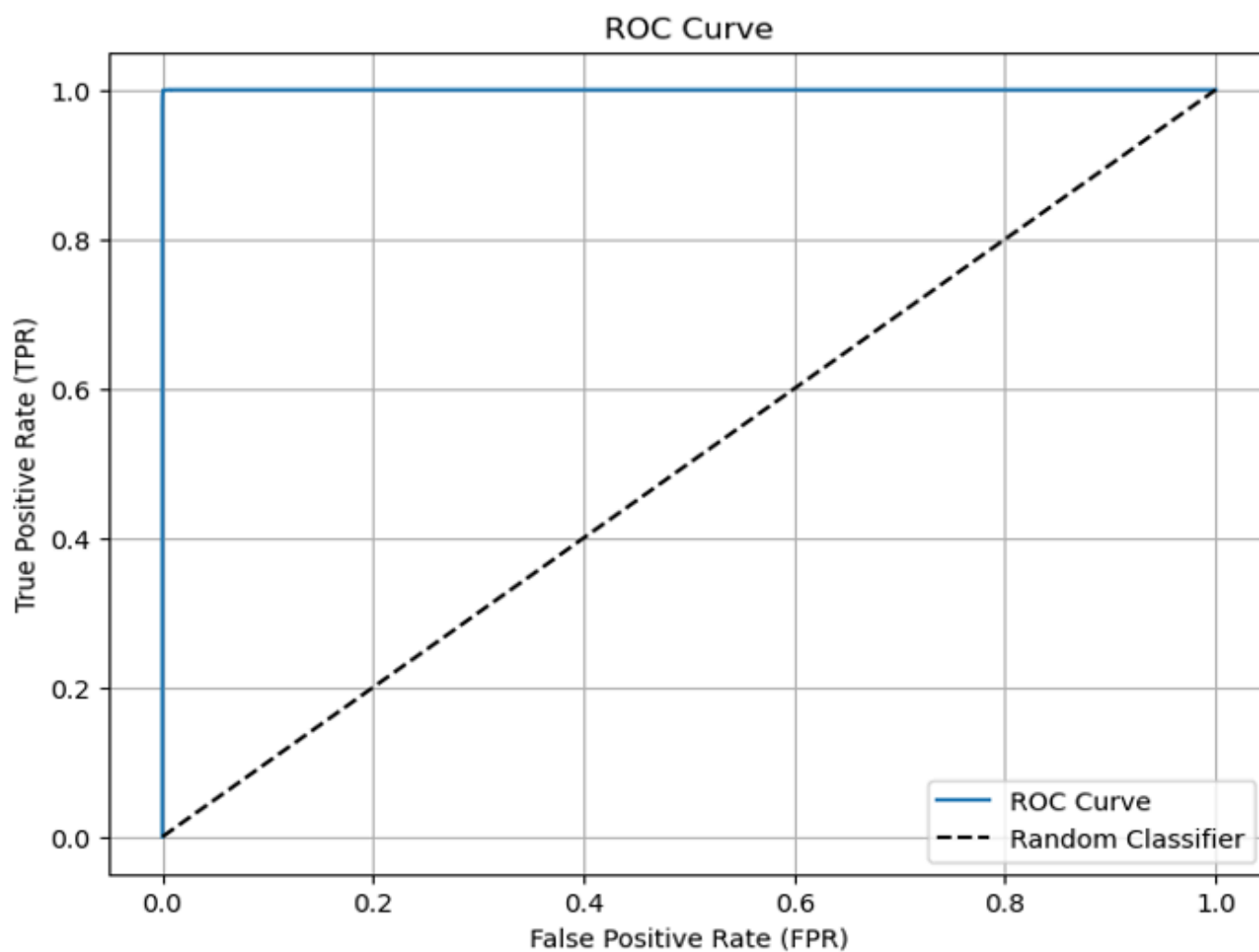


Figure 5

Model's ROC Curve.