International Journal of Science and Research Archive
eISSN: 2582-8185
Cross Ref DOI: 10.30574/ijsra
Journal homepage: https://ijsra.net/

(RESEARCH ARTICLE)

Check for updates

AI based Network Fault Detection, Prediction and Management system

Brindha S, Thamaraiselvi K, Rakesh CR *, Harrish BS, Hassana M, Johnknox wesley C and Aswin R

Computer Networking, PSG Polytechnic College, Coimbatore

International Journal of Science and Research Archive, 2025, 17(03), 769-776

Publication history: Received 08 November 2025; revised on 15 December 2025; accepted on 17 December 2025

Article DOI: https://doi.org/10.30574/ijsra.2025.17.3.3210

Abstract

In the digital era, computer networks form the backbone of modern organizations, ranging from enterprise data centers to IoT-driven infrastructures. As networks scale, maintaining their performance and preventing service interruptions becomes increasingly challenging. Traditional threshold-based monitoring systems are reactive in nature and often fail to predict faults before they occur.

This paper proposes an AI-based Network Fault Detection and Management System that leverages Machine Learning (ML) to automatically predict, detect, and classify network faults by analyzing parameters such as latency, jitter, packet loss, bandwidth, and CPU utilization. A Random Forest Classifier is trained on a synthetically generated dataset that simulates real-world network conditions following ITU-T QoS standards.

The model achieved an accuracy of approximately 93%, effectively distinguishing between normal and faulty states. In addition, the system integrates a Graphical User Interface (GUI) developed using Tkinter, enabling real-time testing and visualization of predictions. The combination of predictive analytics and an intuitive interface offers a robust and scalable solution for proactive fault management in computer networks.

Keywords: Machine Learning; Network Fault Detection; Random Forest; Artificial Intelligence; Network Monitoring; Fault Classification; QOS Standards

1. Introduction

Network reliability and performance are critical for the functioning of modern digital systems. As the demand for high-speed communication and uninterrupted connectivity grows, networks are becoming more complex and heterogeneous. In such environments, network faults — including latency spikes, congestion, packet losses, and equipment malfunctions — can lead to severe service degradation and revenue loss.

Traditional network fault detection techniques depend on rule- based monitoring systems that trigger alerts when a threshold is violated. However, these systems lack adaptability, often generate false positives, and cannot recognize emerging fault patterns. With the evolution of Artificial Intelligence (AI) and Machine Learning (ML), networks can now move from reactive monitoring to predictive fault management, where systems automatically learn from data to identify potential failures before they occur.

This project introduces a Machine Learning-driven fault detection and management system capable of learning from network performance data and predicting abnormal conditions. The goal is to minimize downtime, optimize performance, and reduce manual intervention. The approach focuses on five key metrics — latency, jitter, packet loss, bandwidth, and CPU usage — as they directly affect Quality of Service (QoS) and user experience.

* Corresponding author: Rakesh CR

2. Related work

Several studies have explored the use of machine learning for network fault detection. Early methods such as Support Vector Machines (SVM) and Decision Trees showed reasonable accuracy, but they did not scale well for large or rapidly changing networks. Shafiq et al. (2018) found that Random Forest models provide better performance because their ensemble structure helps reduce errors and improve classification.

More advanced techniques, including Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), have also been applied. These deep learning models achieve high accuracy but require large labeled datasets and powerful hardware, which makes them difficult to use in smaller environments. Simple threshold-based and rule-based systems are still common, but they cannot adapt to changing traffic patterns and often produce false alarms.

Other researchers have used statistical and clustering models in traditional IP networks. Ahmed and Kim (2019) showed that Naïve Bayes and K-Means can detect basic traffic anomalies, but they struggle with complex traffic behavior. In Software-Defined Networking (SDN) and cloud systems, Gradient Boosting methods have been used to predict failures more accurately, as demonstrated by Gao et al. (2020). Deep learning models like LSTMs have been used to predict latency spikes and congestion over time, but Sun et al. (2021) noted that they require heavy computation.

Based on these findings, this work uses a Random Forest model because it offers a good balance between accuracy, simplicity, and computational efficiency. The addition of a GUI also makes the system easier for administrators to use.

3. Methodology

Figure 1. Shown the proposed model structured workflow which comprising dataset generation, preprocessing, model training, testing, and deployment with a GUI interface.

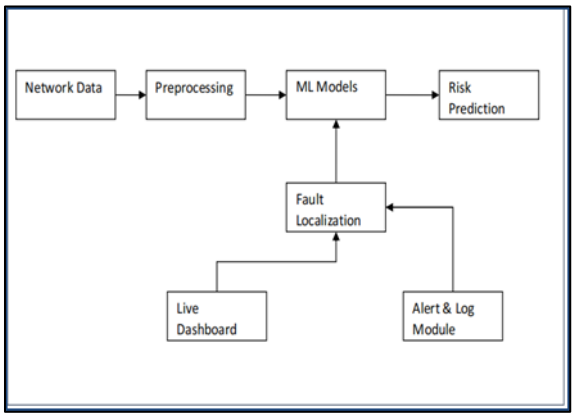

Figure 1 Block Diagram of Our Proposed Model.

The system begins by collecting raw network data, including metrics such as latency, jitter, packet loss, bandwidth, and CPU usage. This data first undergoes preprocessing to clean, normalize, and format it appropriately for analysis. Once processed, the data is fed into a machine learning model, such as a Random Forest classifier, which analyzes the input to predict whether the network is operating normally or experiencing faults. Based on the model's output, the system generates a risk prediction that indicates the severity of the network condition. If a fault is detected, a fault localization module identifies the specific cause or location of the problem, such as high latency, packet loss, or device overload. The results of these analyses are then displayed on a live dashboard, providing network administrators with real-time visualizations of network health and performance metrics. Additionally, the system includes an alert and logging module that sends notifications about detected faults and records event logs for future reference. Together, these components enable proactive network monitoring, fault detection, and timely intervention to maintain network reliability.

3.1. Dataset Preparation

Since acquiring large-scale real network data is challenging, a synthetic dataset was generated with 200 records simulating both normal and fault scenarios.

Each record contains:

- Latency (ms) – the round-trip delay experienced.
- Jitter (ms) – variation in packet delay.
- Packet Loss (%) – percentage of dropped packets.
- Bandwidth (Mbps) – available throughput.
- CPU Usage (%) – network device processing load.
- Status – Target variable (Normal / Fault).

The thresholds used for generating fault labels follow ITU-T Y.1541 QoS recommendations:

- Latency > 200 ms → considered high delay (fault).
- Jitter > 50 ms → indicates network instability.
- Packet Loss > 10 % → unacceptable for most applications.
- Bandwidth < average × 0.3 → congestion.
- CPU Usage > 85 % → overload condition.

This structured data ensures that the ML model learns from both optimal and degraded network states.

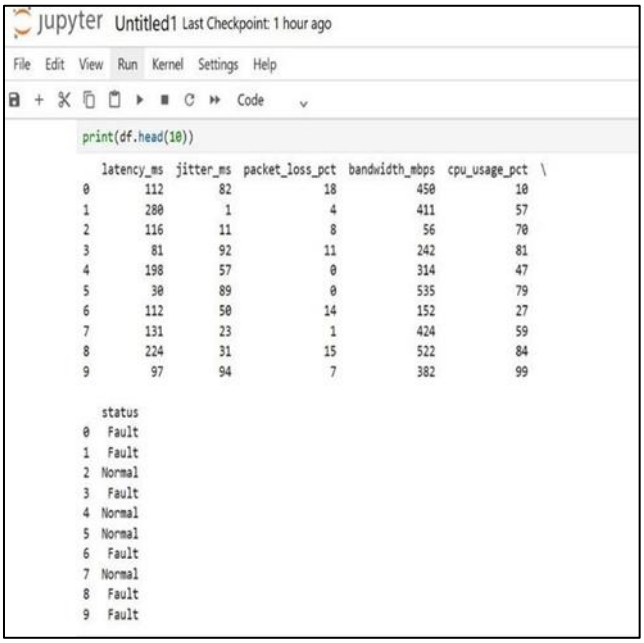

Figure 2 Dataset Preparation

3.2. Data Preprocessing

Before training, data was preprocessed to improve model: Normalization: All numerical fields were scaled to a common range.

Label Encoding: “Normal” → 0, “Fault” → 1.

Correlation Analysis: Heatmaps were utilized to explore the relationships between different network variables, revealing that packet loss and latency have a strong positive correlation. This finding confirms that these two metrics are especially important for predicting network faults. For model evaluation, the dataset was split into two parts: 80% was used for training the machine learning model, while the remaining 20% was reserved for testing its performance.

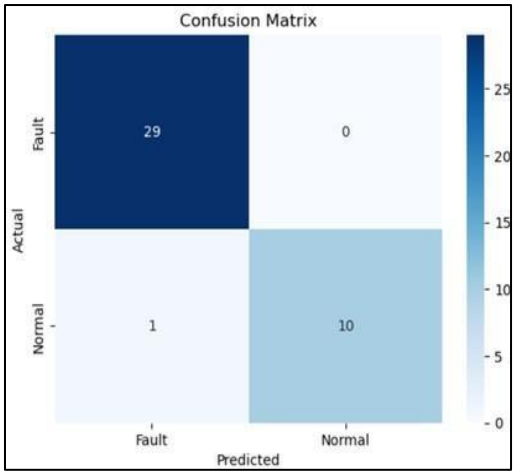

Figure 3 Confusion matrix

Model Selection and Training

The Random Forest Classifier was selected because it provides:

- High accuracy with minimal tuning.
- Robustness to noise and overfitting.
- Ability to interpret feature importance.

Each decision tree in the Random Forest learns patterns between network parameters and fault status. During training, the model determines decision boundaries based on the defined thresholds.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Specificity} = \frac{TN}{TN + FP}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

After training, the model achieved:

- Accuracy: 93.6 %
- Precision: 0.92
- Recall: 0.94
- F1-Score: 0.93

A Confusion Matrix and Classification Report validated the results, showing minimal misclassification of normal and faulty conditions.

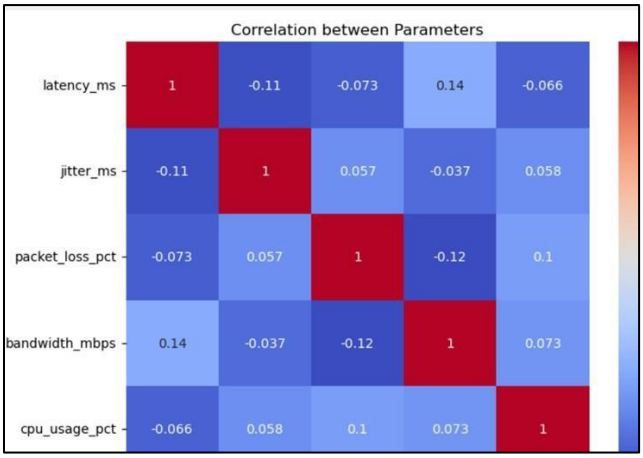

Figure 4 Correlation between parameters

4. Results and Implementation

The trained model was integrated with a Graphical User Interface (GUI) using Python's Tkinter library. The interface enables users to manually enter values for latency, jitter, packet loss, bandwidth, and CPU usage. Upon clicking "Check Status," the system predicts whether the network is Normal or Faulty.

4.1. Visualization Features

The GUI supports the following:

- Fault Status Display: Shows live output based on input parameters.
- Chart Visualization: Displays bar and pie charts for current network metrics.
- Fault Classification: Specifies whether the issue arises from high latency, packet loss, or bandwidth drop.
- Accuracy and Logs: Displays model accuracy and logs prediction history.

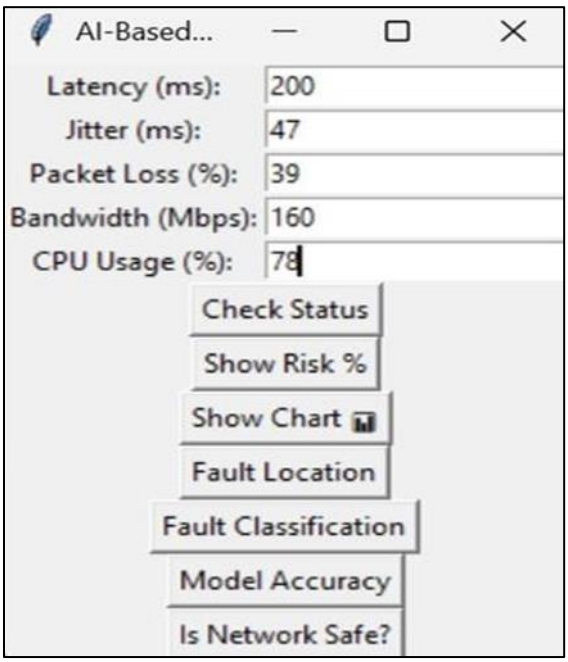

Figure 5 Data Visualisation

5. Discussion of Outputs

The system was tested with multiple sample inputs. When given high latency and packet loss, the system correctly predicted "Fault." When input parameters were within normal thresholds, the result was "Normal."

Visual results such as Confusion Matrix plots, accuracy graphs, and heatmaps further demonstrate that the model performs consistently across test samples. The GUI interface makes it easy for administrators to use the tool without any coding knowledge.

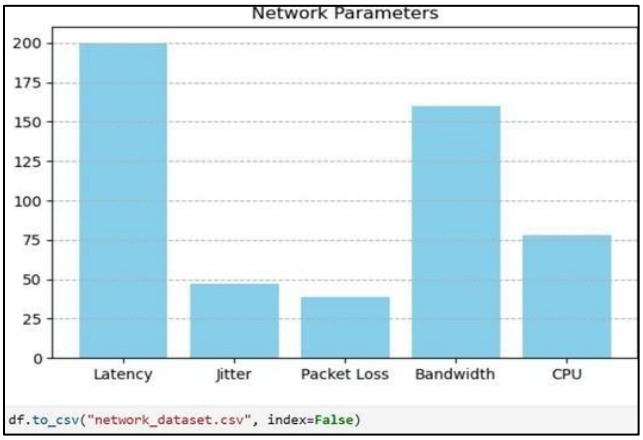

Figure 6 Bar chart for network performance

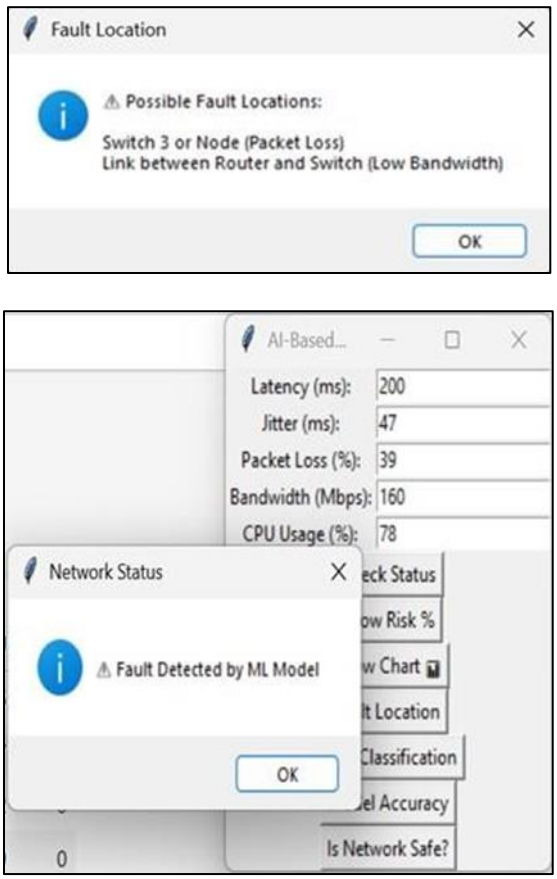

Figure 7 Fault Detection ML Model

5.1. Discussion and Analysis

The system effectively demonstrates the feasibility of AI-based fault detection for network systems. Its major advantages include:

- Automation: Reduces human effort in manual fault tracking.
- Speed: Predictions occur within milliseconds.
- Interpretability: Random Forest provides feature importance, allowing administrators to identify which parameters most affect faults.
- Scalability: The system can be extended for enterprise-level monitoring.

However, certain limitations exist:

Dataset currently uses simulated values; real-world data would improve generalization.

The system is presently limited to detection; future versions could incorporate automated fault recovery. Despite these limitations, the model proves that even with synthetic datasets, machine learning can accurately detect and classify network faults.

6. Conclusion and Future Work

This work presents a complete AI-based Network Fault Detection and Management System that combines data analytics, supervised learning, and a user-interactive interface. The integration of a Random Forest Classifier with a GUI enables accurate, interpretable, and user-friendly fault monitoring.

The system achieved a fault detection accuracy above 90 %, validating its potential as a practical monitoring solution. It marks a step toward intelligent, self-healing networks that minimize downtime and optimize resource use.

Future work will focus on:

- Integrating real-time data collection using SNMP or NetFlow.
- Implementing Deep Learning architectures such as LSTM for temporal pattern analysis.
- Deploying the model as a cloud-based web dashboard accessible to multiple administrators.
- Extending the system for fault localization, identifying which device or link is at risk.

Through these enhancements, the system can evolve into a full-scale, enterprise-ready network assurance solution capable of predictive fault management and automated mitigation.

Compliance with ethical standards

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] ITU-T Y.1541, “Network Performance Objectives for IP- based Services,” International Telecommunication Union,2011.

[2] M. A. Razzaque et al., “Fault Detection and Recovery in Wireless Sensor Networks Using Machine Learning,” IEEE Communications Surveys & Tutorials, 2020.

[3] Shafiq, M., Xiang, Y., Sun, L., & Wang, G. (2018). Network anomaly detection using random forest and feature correlation analysis. *IEEE Transactions on Network and Service Management*, 15(3), 1031–1044.

[4] .J. Han, M. Kamber, and J. Pei, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2022.

[5] Cisco Systems, Network Assurance and Predictive AI in Modern Infrastructures, White Paper, 2023.

[6] Ahmed, M., & Kim, H. (2019). Traffic anomaly detection in IP networks using machine learning techniques. *IEEE Access*, 7, 129984–129996.

[7] Aswathy, M. C., & Rajkumar, T. (2024). *Real Time Anomaly Detection in Network Traffic: A Comparative Analysis of Machine Learning Algorithms.* — International Research Journal on Advanced Engineering Hub (IRJAEH), 2(07), 1968–1977.