© Volume 2, Issue 10, ***October*** 2026 | JATIR
e-ISSN: 2839-5786, p-ISSN: 3139-8464

Self-Evolving Network Intelligence System *An AI-Driven Framework for Autonomous Network Management, Optimization, Security, And Self-Healing*

[1]Prof. Komal Palikondawar, [2]Puja Deshmukh, [3]Pooja Munde, [4]Manasi Patil
[1,2,3,4]Department of Information Technology, Parvatibai Genba Moze College of Engineering, Wagholi, Pune

***Abstract*—The Self Evolving Network Intelligence System is an autonomous framework designed to manage, optimize, and secure modern computer networks using Artificial Intelligence and Machine Learning. Traditional network management relies heavily on static configurations, predefined security rules, and manual intervention, which may not respond effectively to changing traffic conditions and emerging network threats. The proposed system integrates Software Defined Networking with Deep Learning and Reinforcement Learning to continuously monitor network traffic, predict congestion, detect anomalous behavior, and optimize network operations. Network telemetry is analyzed using parameters such as bandwidth utilization, latency, packet flow, and anomalous behavior. Based on these observations, the system can dynamically adjust routing policies, allocate network resources, isolate suspicious traffic, and reroute traffic around failed or congested nodes an administrator dashboard provides visibility into network status, detected anomalies, AI decisions, and remediation activities. The proposed framework is intended to reduce manual intervention, improve network availability, optimize resource utilization, and provide adaptive network security. The system can be applied to dynamic environments such as cloud networks, Internet of Things networks, and next generation communication networks.**

***Index Terms*—Self Evolving Network, Artificial Intelligence, Machine Learning, Software Defined Networking, Deep Learning, Reinforcement Learning**

I. INTRODUCTION

The rapid expansion of digital infrastructure has transformed modern communication networks into large, dynamic, and highly interconnected systems. Enterprises, telecommunication providers,

cloud platforms, and Internet of Things environments continuously generate large volumes of network traffic. The growth of 5G, edge computing, cloud computing, and connected devices has increased the requirement for efficient traffic management, security, scalability, and availability.

Traditional network infrastructures depend significantly on predefined configurations, static routing policies, threshold-based monitoring, and human administrators. Such approaches may become inefficient when traffic conditions change rapidly or when previously unknown security threats appear. The project synopsis identifies static resource allocation, manual threat mitigation, and statically trained intrusion detection models as major limitations of existing network management approaches.

Software Defined Networking provides a programmable network architecture in which network control can be centralized through an SDN controller. This programmability creates an opportunity for integrating intelligent decision-making with network control.

The proposed Self Evolving Network Intelligence System combines SDN with Machine Learning, Deep Learning, and Reinforcement Learning. The system continuously observes network telemetry, predicts network conditions, detects anomalous behavior, selects appropriate corrective actions, and sends those actions to the network through the SDN controller. The central objective is to move network management from a predominantly reactive approach toward an intelligent and adaptive approach capable of supporting traffic optimization, security monitoring, and self-healing.

II. LITERATURE REVIEW

Software Defined Networking has provided a foundation for programmable and centrally controlled network management. McKeown et al. introduced OpenFlow as an approach for enabling experimental and programmable control over network switches through standardized flow-table operations [1].

Research has subsequently investigated the integration of Machine Learning and Reinforcement Learning with SDN. Reinforcement Learning is suitable for network optimization because an agent can learn policies based on network states, actions, and resulting rewards. RL-Routing applies Reinforcement Learning to SDN traffic engineering and uses network information such as throughput and delay when learning routing policies [5].

More recent work has explored Deep Reinforcement Learning for traffic engineering. Pei et al. proposed a Deep Reinforcement Learning approach for efficient routing and traffic engineering in SDN [3]. ScaleDRL also investigated Deep Reinforcement Learning for scalable traffic engineering in SDN environments [4].

These studies demonstrate the potential of intelligent routing and traffic engineering. However, the SENI project aims at a broader integration. Its proposed architecture combines traffic prediction, anomaly detection, Reinforcement Learning, SDN based execution, feedback, and administrator visibility within one framework.

e-ISSN: 2839-5786, p-ISSN: 3139-8464

Research gap:
Existing approaches commonly concentrate on individual problems such as traffic engineering, routing optimization, intrusion detection, fault detection, resource allocation, or network monitoring. SENI proposes an integrated framework in which network performance and security information are jointly considered by an intelligent decision engine, with a feedback mechanism intended to adapt future decisions.

III. PROBLEM STATEMENT AND OBJECTIVE

A. Problem Statement

Modern enterprise and telecommunication networks suffer from inefficiencies caused by static resource allocation, manual threat mitigation, and limited adaptation to changing traffic conditions. Sudden traffic increases can result in congestion, latency, packet loss, and inefficient bandwidth utilization. Conventional intrusion detection systems may depend on fixed signatures or statically trained models, which can lose effectiveness when network behavior changes. SENI therefore addresses the need for an intelligent system capable of autonomously learning, adapting, and modifying network management and security strategies in real time.

B. Aim

The aim of SENI is to design and develop a system that autonomously monitors network traffic, predicts and prevents congestion, detects anomalous behavior, and optimizes routing policies using continuous Machine Learning and Software Defined Networking.

C. Objectives

1) Develop real time network traffic monitoring. 2) Capture packet flow and node status. 3) Implement continuous or online learning. 4) Predict network congestion. 5) Dynamically allocate bandwidth. 6) Detect anomalous network behavior. 7) Identify and isolate potential security threats. 8) Use an SDN controller for automated routing changes. 9) Implement self healing mechanisms. 10) Provide a web-based administrator dashboard. 11) Reduce dependence on manual configuration through zero touch orchestration.

IV. PROPOSED SENI SYSTEM

The proposed Self Evolving Network Intelligence System integrates three major technologies: Software Defined Networking, Artificial Intelligence and Machine Learning, and Reinforcement Learning.

The system continuously collects network telemetry from switches and routers. The collected information is preprocessed and relevant network features are extracted. These features are supplied to AI models responsible for traffic prediction and anomaly detection.

The system continuously collects network telemetry from switches and routers. The collected information is preprocessed and relevant network features are extracted. These features are

supplied to AI models responsible for traffic prediction and anomaly detection.
The system continuously collects network telemetry from switches and routers. The collected information is preprocessed and relevant network features are extracted. These features are supplied to AI models responsible for traffic prediction and anomaly detection.

The output of these models is provided to the Reinforcement Learning based decision engine. The decision engine evaluates the current network state and determines an appropriate action. Possible actions include rerouting traffic, changing routing policies, allocating network resources, blocking suspicious traffic, and isolating compromised nodes.
The selected action is passed to the SDN controller, which applies the corresponding configuration to the network. After execution, the system observes the resulting network state and calculates feedback. This feedback is used by the learning mechanism to improve future decisions.

V. SYSTEM ARCHITECTURE

a. Data Plane
The Data Plane contains the network devices, simulated switches, routers, hosts, and links. These devices generate traffic and network telemetry.

b. Control Plane
The Control Plane contains the SDN controller. The controller collects network information and provides programmable control over network devices. OpenFlow or controller APIs can be used to apply routing and flow-management decisions.

c. Application and Intelligence Plane
This layer contains data preprocessing, traffic prediction, anomaly detection, Reinforcement Learning, the decision engine, administrator dashboard, and feedback management.

d. System Flow
Network Devices SDN Controller Data Collection Preprocessing AI Models Decision Engine SDN Controller
Network Devices Feedback. The administrator dashboard receives information about topology, traffic, threats, and AI decisions.

VI. PROPOSED METHODOLOGY

A. Data Ingestion
Network telemetry is continuously collected from switches and routers. The telemetry may include packet flow information, source and destination addresses, bandwidth usage, latency, packet frequency, and node status.

B. Preprocessing and Feature Extraction

Raw network information is normalized and converted into features suitable for machine learning. Possible features include packet size, packet frequency, source address, destination address, bandwidth utilization, latency, packet count, node status, and traffic variation.

C. Traffic Prediction

A Machine Learning or time-series model estimates future network traffic and congestion conditions. The predicted information can assist the system in taking preventive action before severe congestion occurs.

D. Anomaly Detection

An anomaly detection model analyzes network behavior and identifies deviations from expected patterns. Unsupervised learning or clustering can be used where labelled attack data is limited.

E. Reinforcement Learning Decision Engine

The Reinforcement Learning agent receives the current network state and predictions from the AI models. It selects an action based on an objective or reward function.

F. SDN Execution

After the decision is selected, the decision engine communicates with the SDN controller. The controller translates the decision into appropriate network rules or API commands.

G. Feedback Loop

After an action is executed, network conditions are monitored again. The system evaluates whether the action improved the network condition. The resulting feedback is used as a reward for the Reinforcement Learning component. The cycle is: Observe Predict Decide Execute Measure Learn Repeat.

VII. SYSTEM MODULE

A. Network Telemetry Module: Collects network flow and device information and maintains the current network state.
B. Traffic Prediction Module: Analyzes historical and current traffic information and predicts possible congestion or increased demand.
C. Self-Evolving Anomaly Detection Module: Detects abnormal traffic patterns and potential security threats.
D. Automated Remediation Module: Executes corrective actions such as rerouting, traffic blocking, node isolation, or policy modification.
E. Reinforcement Feedback Module: Evaluates the effect of network actions and provides feedback to the learning system.

F. Administrator Dashboard: Provides visualization of network topology, traffic conditions, active threats, AI decisions, routing policies, node health, and system alerts.

VIII. IMPLEMENTATION REQUIREMENT

The proposed Self Evolving Network Intelligence System requires a combination of hardware and software components to implement network monitoring, artificial intelligence-based analysis, automated decision making, and SDN based network control. The implementation environment should support network simulation, machine learning model development, data processing, and visualization.

A. Hardware Requirements

The system can be implemented on a computer or server capable of running the network simulation, SDN controller, AI models, and administrator dashboard. A minimum configuration may include a multi-core processor, at least 8 GB RAM, sufficient storage for network telemetry and model data, and a reliable network interface.

B. Software Requirements

The major software components required for the proposed system are:

Component	Proposed Technology
Operating System	Ubuntu Linux
Programming Language	Python
Network Simulation	Mininet
SDN Controller	Ryu or ONOS
Machine Learning	Scikit-learn
Deep Learning	TensorFlow or PyTorch
Data Processing	Pandas and NumPy
Web Framework	Flask or FastAPI
Dashboard	React or Streamlit
Database	MySQL or MongoDB
Network Communication	OpenFlow and SDN APIs

The network telemetry module collects information such as traffic flow, bandwidth utilization, latency, packet statistics, and network conditions. The collected data is processed using Python based data processing and machine learning libraries.

Deep Learning models can be used for traffic prediction and anomaly detection, while Reinforcement Learning can support adaptive routing and resource allocation decisions. The SDN controller executes the decisions generated by the intelligence layer through appropriate network control interfaces.

IX. DATABASE AND UML DESIGN

The database is designed to store network information, telemetry data, detected anomalies, AI decisions, network policies, and system activities. The stored information supports continuous monitoring, analysis, decision making, and evaluation of the proposed system.

A. Database Design

The major database tables proposed for the system are:

Table	Important Fields	Purpose
Network Devices	Device ID, Device Name, IP Address, Device Type, Status	Stores information about network devices
Network Telemetry	Telemetry ID, Device ID, Timestamp, Bandwidth, Latency, Packet Count, Utilization	Stores network monitoring data
Traffic Analysis	Analysis ID, Timestamp, TrafficRate, Congestion Level, Prediction	Stores traffic analysis and prediction results
Anomaly Detection	Anomaly ID, Device ID, Timestamp, Anomaly Type, Severity, Status	Stores detected abnormal network behavior
Routing Decisions	Decision ID, Source, Destination, Selected Path, Timestamp, Reason	Stores routing decisions generated by the intelligence system
Network Policies	Policy ID, Policy Type, Rule, Priority, Status	Stores network control and security policies
Reinforcement	Feedback ID, Action, Reward, Network State,	Stores feedback used for
Feedback	Timestamp	continuous learning
Administrator Activity	Activity ID, Administrator, Action, Timestamp, Description	Records administrator activities and manual interventions

The database enables the system to maintain historical network information and provides data that can be used for analysis, monitoring, feedback, and future model improvement.

B. UML Design

The proposed system contains four major actors: Administrator, AI Engine, SDN Controller, and Network Devices. The Administrator can view network topology, monitor traffic, examine detected anomalies, update policies, and override AI generated decisions when required. The AI Engine analyzes network telemetry, predicts traffic conditions, detects anomalies, and generates appropriate network actions. The SDN Controller receives these decisions and applies the required routing, resource allocation, or security actions to the network devices. Network Devices continuously provide telemetry information to the system.

The major use cases of the system include:

1. View Network Topology: The administrator can view the current network structure and device status.

2. Analyze Network Traffic: The system processes telemetry data to understand network traffic conditions.
3. Predict Congestion: The AI engine predicts possible congestion using network traffic information.
4. Detect Anomaly: The system identifies abnormal or suspicious network behavior.
5. Optimize Routing: The AI engine selects suitable routing actions based on network conditions.
6. Perform Self-Healing: The system reroutes traffic when a link or network path fails or becomes congested.
7. Mitigate Security Threats: Suspicious traffic or compromised network segments can be isolated.
8. Update Network Policy: Network policies can be modified based on system requirements.
9. Override AI Decision: The administrator can manually override an automatically generated decision.
10. Provide Feedback: Network results and rewards are returned to the learning system for continuous improvement.

The overall UML workflow follows the sequence:
Network Devices → Telemetry Collection → SDN Controller → Data Processing → AI Models→ Decision Engine → SDN Controller → Network Devices
The administrator dashboard provides visibility into this workflow and allows the administrator to monitor network conditions and system actions.

X. PROPOSED EVALUATION FRAMEWORK

Since the supplied synopsis does not contain completed experimental measurements, numerical results should not be presented as actual results. The proposed system can instead be evaluated using latency, throughput, packet loss, bandwidth utilization, congestion rate, detection accuracy, false positive rate, recovery time, Reinforcement Learning reward, and network downtime.
SENI can be compared against a baseline configuration using static routing, conventional threshold-based management, or conventional ML based detection. A suitable experimental procedure would involve generating normal and abnormal traffic in a Mininet based topology, recording network conditions, allowing SENI to make decisions, and comparing the resulting metrics with the baseline.

XI. EXPECTED RESULTS AND DISCUSSION

The expected outcomes are derived from the project objectives and should be validated through actual implementation and experimentation.
The system is expected to provide autonomous detection and mitigation of network threats, improved routing during congestion, and automated recovery from selected network failures. The

Reinforcement Learning feedback mechanism is intended to improve decision making as the system observes additional network states.

The project synopsis identifies autonomous threat detection and mitigation, reduced network downtime through self healing, improved bandwidth utilization, continuous adaptation, and administrator visibility as expected outcomes. Statements such as near-zero packet drop, significant reduction, or elimination of concept drift must be treated as target outcomes rather than proven results until experimental measurements are available.

XII. NOVELTY OF THE PROPOSED SYSTEM

a. Continuous Online Learning: The system is designed to learn from continuously changing network conditions rather than depending exclusively on a fixed training dataset.
b. Unified Network and Security Intelligence: SENI combines network performance optimization and cybersecurity monitoring in one intelligent framework.
c. Zero Touch Remediation: Instead of only generating alerts for administrators, the system is designed to automatically execute appropriate corrective actions through the SDN controller.

XIII. APPLICATION

a. 5G and 6G Networks: Management of highly dynamic traffic conditions and large numbers of connected devices.
b. Enterprise Cloud Data Centers: Traffic engineering, congestion management, and security monitoring between servers.
c. IoT and Smart Cities: Monitoring large numbers of connected devices and identifying abnormal traffic.
d. Financial Institutions: Support for environments requiring high availability, low latency, and strong network security.

XIV. ADVANTAGES

Reduced dependency on manual network configuration; automated traffic management; proactive congestion handling; automated anomaly detection; self healing after selected network failures; adaptive routing; centralized network intelligence; administrator visibility; scalable architecture; and improved utilization of network resources.

XV. LIMITATIONS

Computational overhead from continuous AI processing; training requirements for Reinforcement Learning; safety concerns associated with autonomous actions; false positives and false negatives in anomaly detection; simulation to real network gaps; SDN controller dependency; dependence on data quality; and the absence of completed experimental validation in the supplied synopsis.

XVI. FUTURE SCOPE

Future extensions may include Federated Learning across network domains, Explainable AI for administrator trust, Multi Agent Reinforcement Learning, 5G and 6G network slicing, natural language administration, evaluation on physical network testbeds, and advanced quantum resistant security mechanisms.

XVII. CONCLUSION

The Self Evolving Network Intelligence System presents an intelligent approach to modern network management by combining Software Defined Networking, Machine Learning, Deep Learning, and Reinforcement Learning. The proposed architecture continuously collects network telemetry, analyzes traffic and security conditions, predicts possible congestion, detects anomalous behavior, and selects corrective actions through an intelligent decision engine.

The integration of the decision engine with an SDN controller enables automated network changes such as traffic rerouting, resource allocation, and threat isolation. The feedback mechanism provides the foundation for continuous learning and adaptation.

The proposed system addresses important limitations of static network management, including high human dependency, reactive congestion management, isolated security monitoring, and the need for frequent manual model updates. At the current stage, the project synopsis describes the proposed architecture and expected outcomes rather than completed experimental results. Actual performance claims must therefore be established through implementation and quantitative evaluation.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to Prof. Komal Palikondawar, Department of Information Technology, Parvatibai Genba Moze College of Engineering, Wagholi, Pune, for guidance and support during the development of the Self Evolving Network Intelligence System. The authors also acknowledge the support of the Department of Information Technology and the institution for providing the academic environment and resources required for the project.

REFERENCES

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “OpenFlow: Enabling innovation in campus networks,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Apr. 2008, doi: 10.1145/1355734.1355746.

[2] H. Farhady, H. Lee, and A. Nakao, “Software-defined networking: A survey,” *Comput. Netw.*, vol. 81, pp. 79–95, Apr. 2015.

[3] X. Pei, P. Sun, Y. Hu, D. Li, B. Chen, and L. Tian, “Enabling efficient routing for traffic engineering in SDN with deep reinforcement learning,” *Comput. Netw.*, vol. 241, Art. no.

110220, 2024.
[4] "RL-Routing: An SDN routing algorithm based on deep reinforcement learning," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 3185–3199, 2020.
[5] Journal of Academic Trends and Innovative Research, "Author guidelines," JATIR.