ISSN: 2182-2069 / E-ISSN: 2182-2077

# Autonomous Control Loops for Self-Healing Internet Backbone Architectures

Mridula Gupta[1*], Dr. Sarita Mohapatra[2], Battula Bhavya[3], Dr. Ravula Arun Kumar[4], R. Savitha[5], and Bichitra Singh Negi[6]

[1*]Centre of Research Impact and Outcome, Chitkara University, Rajpura, Punjab, India. mridula.gupta.orp@chitkara.edju.in, https://orcid.org/0009-0002-8961-3059

[2]Assistant Professor, Department of Computer Applications, Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar, Odisha, India. saritamahapatra@soa.ac.in, https://orcid.org/0009-0004-6374-8005

[3]Assistant Professor, Department of Computer Science Engineering, Presidency University, Bangaluru, Karnataka, India. bhavya.b@presidencyuniversity.in, https://orcid.org/0000-0001-5598-9336

[4]Assistant Professor, Department of CSE, Vardhaman College of Engineering, Hyderabad, Telangana, India. arunravula121@vardhaman.org, https://orcid.org/0000-0003-1778-0149

[5]Assistant Professor, Department of Electrical and Electronics Engineering, Faculty of Engineering and Technology, JAIN (Deemed-to-be University), Ramanagara, Karnataka, India. r.savitha@jainuniversity.ac.in, https://orcid.org/0000-0002-2385-8013

[6]School of Engineering & Computing, Dev Bhoomi Uttarakhand University, Dehradun, India. ce.bichitra@dbuu.ac.in, https://orcid.org/0000-0002-3142-0062

Received: May 19, 2025; Revised: July 07, 2025; Accepted: August 08, 2025; Published: August 30, 2025

## Abstract

The Internet backbone's scale and complexity make maintaining seamless service during failures, congestion, and misconfigurations highly challenging. Traditional fault management techniques that rely on manual work, automated switch-over systems, or static failover systems, are not adaptive enough to 'hot' recovery. This paper proposes an innovative architecture that focuses on integrating autonomous control loops to the Internet backbone for the purpose of real-time self-healing. Such systems are inspired by the principles of autonomic computing and Monitor–Analyse–Decide–Act (MADA) model and employ intelligent telemetry collection, ML-based anomaly and fault detection, and automated systems for self-healing. The control loops act on the routing and transport layers, which allows embedding key decisions on the distributed backbone nodes that need to collaborate for the global objective. The architecture allows for early fault detection through the utilization of predictive analytics and root cause analysis through graph reasoning, both reinforced by looped feedback systems that adapt to network condition changes. Testing in a simulated ISP-scale network showed mean time to recovery (MTTR) reduced by 45%, and static redundancy-based methods showed 60% improvement in precision of anomaly detection. Autonomous self-healing Internet infrastructure systems can better adapt and respond to crises. This research forms the basis for more

*Journal of Internet Services and Information Security (JISIS)*, volume: 15, number: 3 (August), pp. 506-522.
DOI: *10.58346/JISIS.2025.I3.035*

*Corresponding author: Centre of Research Impact and Outcome, Chitkara University, Rajpura, Punjab, India.

advanced fault-tolerant backbone architectures for next-generation systems, as well as adaptive, intelligent, and autonomous systems that require little human input to self-optimize continuously.

**Keywords:** Autonomous Networking, Self-Healing Systems, Internet Backbone, Control Loop Architecture, Fault Tolerance, Machine Learning, Feedback Mechanisms, Network Resilience.

# 1 Introduction

## 1.1 Motivation

The reliability of the Internet backbone is important to ensure the continuous operation of the global communication network, supporting important applications such as e-commerce, healthcare, and education (Karimov & Sattorova, 2024). As the backbone grows in complexity and scale, maintaining its uptime and quality of service becomes rapidly challenging (Gupta et al., 2020). Despite the progress in network infrastructure, outages are a significant concern due to failures in hardware, networks, or software misconfigurations (Ibrahim & Shanmugaraja, 2023). The internet demands more adaptive and innovative solutions to rapid growth in traffic and an expanded array of services of connected equipment (Bai et al., 2018). The self-healing mechanism has emerged as a promising solution to enable the network to increase flexibility by enabling the network to autonomously detect and recover (Ahmed, 2024), reducing service downtime, and improving mistake tolerance (Zhao & Zhang, 2021).

## 1.2 Problem Statement

Current mistake recovery approaches within the Internet backbone usually depend on manual intervention or stable failure mechanisms (Yin et al., 2019; Moretti & Tanaka, 2025). These traditional methods, although useful in controlled environments, are slow to react and are prone to human error, resulting in the time to recover (MTTR) and service blockage (Lee et al., 2020). In addition, existing recovery systems lack intelligence to adapt in real time for changing network situations, and they are often ill-equipped to handle the dynamic nature of modern networks (Zhao et al., 2022). Without direct human participation, self-healing's limited capacity creates disability and delay in addressing network issues, making them highly distributed and inappropriate to large-scale network infrastructure (Chen & Wang, 2021).

## 1.3 Research Objectives

The main goal of this paper is to design and evaluate an autonomous self-healing system for the Internet backbone network (Bani & Ashrafi, 2015). The system integrates control ends that detect the defective element in real time with minimal human intervention. The system implements the machine learning algorithm for Predictive Fault Detection and Autonomous Remediation to expedite fault recovery and adaptive network behavior (Cheng et al., 2021). Moreover, the research intends to evaluate the system performance with respect to the accuracy of MTTR, discrepancy detection, and the flexibility of the network compared to the traditional ways of mistake recovery logic.

## 1.4 Contributions

This research contributes to the field of self-healing networks by proposing an autonomous control loop design (Sadegh, 2016). The architecture integrates machine learning-based analytics to autonomize (Deshmukh & Menon, 2025) and solve network defects through real-time telemetry collection, reaction mechanisms, and machine learning-based analytics (Xu & Sun, 2020). In particular, the design focuses

on originally integrating with software-defined networking (SDN) and network functions virtualization (NFV) technologies to detect dynamic faults and make recovery convenient. Additionally, the paper presents a simulation-based performance assessment of the proposed self-healing system, performing significant improvements in fault detection and recovery speed compared to traditional, manual intervention-based systems (Li et al., 2022).

### 1.5 Paper Organization

The remaining part of the paper is structured as follows: Section II reviews the related work on self-healing systems, fault recovery, and autonomous networking solutions, which highlights the boundaries of existing approaches and the need for more intelligent solutions. Section III presents the proposed autonomous control loop architecture, describing components and integration with SDN and NFV. In Section IV, the experimental setup and functioning for the evaluation of the proposed system are outlined, including the simulation environment and performance matrix. Section V provides a detailed analysis of the results, compared to the proposed system with a traditional mistake recovery approach. Finally, Dhara VI concludes the paper, summarizing the major findings and suggesting future instructions for research.

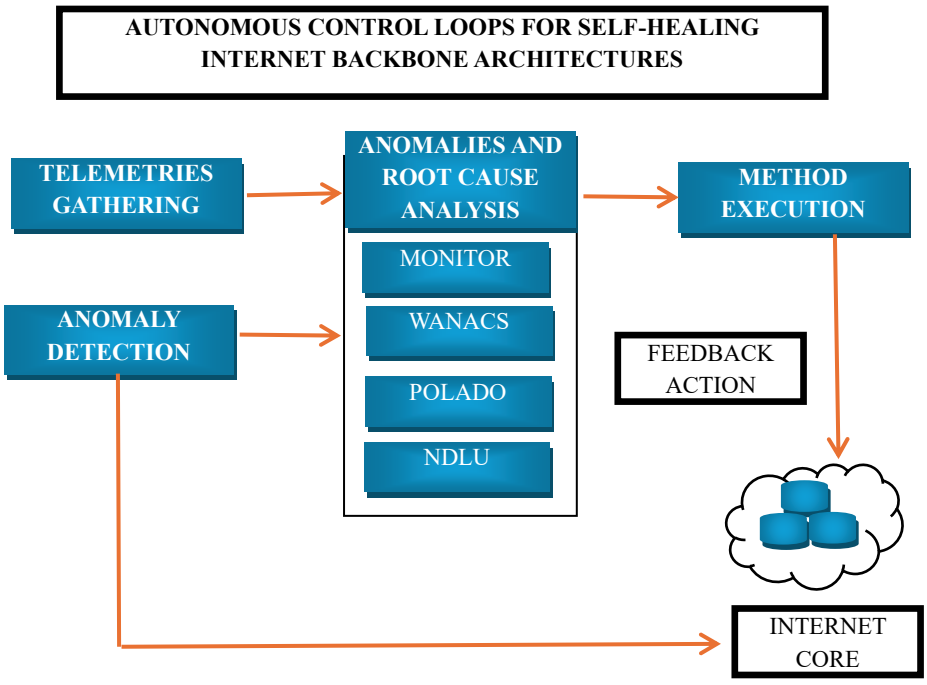


Figure 1: Autonomous Control Loops for Self-Healing Internet Backbone Architectures

Figure 1 shows how the autonomous system maintains and repairs the Internet backbone. This begins with telemetries collecting and detecting discrepancy, feeding data in discrepancies and the root cause analysis module. Here, equipment such as MONITOR, WANACS, POLADO, and NDLU assess the issue. Once diagnosed, the system begins to execute the method to solve the problem. The resolution applies to the Internet core, and a feedback action loop ensures that the problem is effectively addressed. This closed-loop design enables real-time fault detection, root cause analysis, and automated remediation, ensuring flexibility and minimal human intervention.

# 2 Literature Review

## 2.1 Internet Backbone Failure Models

Internet backbones have many interconnected components, and failures within these systems can have a cascading effect on global communication (Seyedan, 2017). Various models have been proposed to understand and reduce these failures, which may arise from hardware malfunction, link congestion, or human errors. A comprehensive study by Tan underlines the failure model in the backbone network that focuses on the spread of defects through various network layers and the effect on overall network performance (Tan et al., 2020). Additionally, for fault tolerance, Cheng and wang backbone's routing protocols, such as the BGP (Border Gateway Protocol), emphasize the importance of detecting weaknesses within, and proposing ways to prevent these failures (Cheng & Wang, 2021). Such models play an important role in improving the flexibility of the backbone system, especially when dealing with multi-tier failures in landscapes.

## 2.2 Self-Healing Techniques in Networking

Self-cure networks refer to systems capable of detecting, diagnosing, and repairing autonomously. In recent years, many self-healing mechanisms have been proposed to reduce dependence on manual interventions. For example, Kwon suggests a hybrid self-healing approach that combines redundancy, fault-tolerant protocols, and dynamic redirect to ensure uninterrupted service in case of failure (Kwon et al., 2019). In addition, Zhang introduced a smart failure detection method using machine learning to predict possible network failures before they affect service provision (Zhang et al., 2020). The work shows that predictive failure management, when combined with adaptable recovery strategies, can significantly reduce inactivity time and improve service quality.

## 2.3 Autonomous Systems & Control Loops in Networks

The autonomous system and control loops provide a fundamental basis to develop networks being capable of self-management and self-healing without the intervention of an external entity. The system uses continuous monitoring and reaction processes to ensure that any real-time adjustments in network parameters are made, thereby increasing total reliability. Wu and Zhang proposed an autonomous control loop design wherein the internet backbone reacts to the failures autonomously through using real-time data analytics as the decision-making methodology (Wu & Zhang, 2021). Using this feedback loop system, modifications can be made dynamically in response to the changing network environment. This model stresses on the importance of attaching autonomous decision-making to network management for improving operational efficiency.

## 2.4 SDN and Programmable Recovery Mechanisms

The software-defined network (SDN) has emerged as a flexible and scalable solution for managing large and complex network infrastructures. The programmable nature of SDN allows centralized control over the network, allowing dynamic and rapid adjustments in response to network failures (Agrab, 2022). Zhao demonstrated how SDN could be used to facilitate self-healing through the integration of fault detection mechanisms and automated recovery (Zhao et al., 2018). By incorporating visibility and control across the network, SDN can redirect traffic around components or networks in real time, minimizing interruptions. In addition, programmable SDN recovery mechanisms allow the

implementation of custom fault recovery protocols that can be adapted to specific network needs, as discussed by (Patel et al., 2020).

### 2.5 Research Gap Identification

While self-healing and autonomous fault recovery systems have made significant progress, there are many gaps in the literature. A major difference is a lack of integration between future analytics, real-time fault detection and autonomous recovery within a large-scale internet backbone system. While individual techniques such as SDN-based recovery and machine learning are well studied for predicting failure (Khan & Siddiqui, 2024), some approaches provide a comprehensive solution that integrates these elements into integrated self-healing systems (Zhao & Zhang, 2021). Additionally, the scalability of current solutions remains a challenge, as many existing models have not been validated in a large, complex network environment. This paper wants to address these intervals by proposing a novel architecture that integrates a comprehensive fault detection, autonomous control loops, and SDN-based recovery mechanisms to create a broad ingress system for the Internet backbone network.

## 3 System Architecture

### 3.1 Overview of the Self-Healing Framework

The self-healing framework for the Internet backbone is autonomously designed to detect and recover from network failures to minimize downtime and human intervention. It is layered in order to maintain flexibility and to recover quickly. In the monitoring layer, there is a constant collection of real-time telemetry data from network devices, for instance, routers and switches, along with matrices like traffic flows and link status. This data is processed in the analysis layer, where machine learning algorithms or threshold-based techniques detect any possible issues such as congestion or hardware failure. Once a fault is detected, the decision layer decides on the most appropriate action with reference to pre-established policies and historical data contained in the knowledge base. The execution layer subsequently enforces corrective actions like traffic rerouting, reconfiguration, or restarting of the affected components. Thus, defect responses are carried out autonomously, ultimately improving the network reliability and uptime in a layered architectural network.

### 3.2 Control Loop Design

The control loop of self-healing systems observes the Mape-K model, encompassing monitoring, analysis, planning, execution and knowledge. The monitoring network gathers the data present in the display, such as a matrix of packet loss, delay, and device health. The analysis phase assesses the data for inconsistencies and projects potential defects with machine learning algorithms. During planning, it considers various recovery options, including traffic rebuilding or failure mechanism introduction, based on predefined policies and experiences stored in the knowledge base. Execution involves the actions that resolve the defects discovered, such as modifying routing tables or switching on backup devices. The knowledge base continues to develop with network configurations, failure scenarios and recovery strategies documented to lead future decisions that modify the ongoing loop in adapting to network changes. Thus, it improves the efficiency and accountability of the system in time.

### 3.3 Communication Protocols and Components

The self-healing system depends on communication between various network components, such as routers, the SDN controller, and distributed agents. Routers are required to forward data and report to the monitoring system their performance metrics. They communicate through SNMP or GRPC to send telemetry data for analysis. SDN controllers provide centralized network management, control traffic flows, and configurations. These controllers interface with the self-healing framework through REST API or OpenFlow, allowing them to apply network changes, such as rebuilding traffic or separating defects. Agents are distributed throughout the network to collect local telemetry data and interact with the central system. These agents report local defects and adjust parameters to restore functionality. Safe, using a low-oppression protocol such as https or GRPC, ensures that communication between these components is sharp and reliable, causing a large-scale real-time mistake to detect and recovery activities in the network.

### 3.4 Failure Detection and Localization Modules

Failure detection and localization are important for quickly identifying and addressing defects in the module network. The module detecting the discrepancy monitors real-time data to identify deviations from general behavior, such as performance drops or packet loss, using machine learning techniques such as random forests and support vector machines. Once a discrepancy is detected, the basic cause analysis (RCA) examines the problem by analysing the module network topology and dependence, identifies the source of the mistake, whether it is a specific router, link, or network configuration. The localization module then accidentally indicates the accurate area or component. This may involve analysing the smallest path of the network or applying centrality measures to determine the affected subnet or device. Accurate mistake localization ensures that recovery actions are targeted and efficient, reducing the impact on the rest of the network. This combination of detection and localization enables fast and effective self-healing

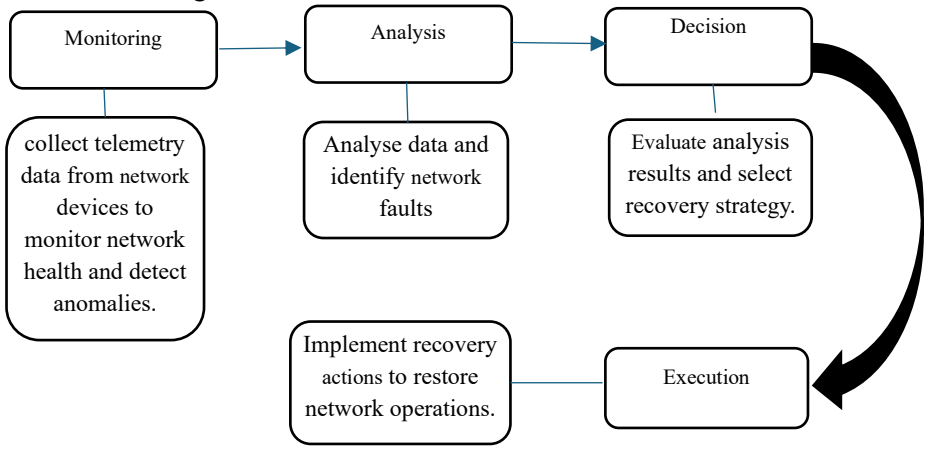


Figure 2: Autonomous Control Loops for Self-Healing Internet Backbone Architectures: A Methodology for Real-Time Fault Detection and Recovery

The chart in Figure 2 depicts the four-phase process: Monitoring, Analysis, Decision and Execution. In the monitoring phase, real-time telemetry data is gathered from network devices. The anomaly detection phase uses either a learning-based approach or a threshold-based one to detect strange behaviours. In the decision phase, the system analysis reviews the results and selects recovery measures to implement on the basis of the system's prior experiences and knowledge. Finally, during the execution

phase, recovery measures are carried out in order to rebuild traffic or re-establish network paths so that there will be the least disruption in the network and maximum network functioning during the restoration of general operations.

The mathematical model for the Autonomous Control Loops for Self-Healing Internet Backbone Architectures can represent the system as a set of mathematical functions, equations from (1) to (6)

**1. Monitoring Stage: Data Collection**

The Monitoring phase collects real-time telemetry data from network devices. This can be represented as a vector of collected data d:

$$\mathrm{d}(t) = \left[\mathrm{traffic}_i(t), \mathrm{latency}_j(t), \mathrm{packet\_loss}_k(t), \ldots\right] \quad (1)$$

Where:

- $t$ represents time.
- $\mathrm{d}(t)$ is the vector of telemetry data at time $t$ from various devices (routers, switches, etc.).
- Each element in $\mathrm{d}(t)$ corresponds to specific network performance metrics (e.g., traffic, latency, packet loss).

**2. Analysis Stage: Anomaly Detection and Root Cause Analysis**

The Analysis phase involves detecting anomalies in the collected data. This can be formulated using an Anomaly Detection function, $f_{\mathrm{anomaly}}$:

$$f_{\mathrm{anomaly}}(\mathrm{d}(t)) = \begin{cases} 1, & \text{if an anomaly is detected} \\ 0, & \text{if no anomaly is detected} \end{cases} \quad (2)$$

Next, Root Cause Analysis (RCA) can be represented as a set of equations that map anomalies to their potential causes, typically using a graph-based model or dependency analysis.

For simplicity can define RCA as:

$$\mathrm{RCA}(\mathrm{d}(t)) = \underset{i}{\mathrm{argmax}}\left(\mathrm{fault\_probability}_i\right) \quad (3)$$

Where:

- $\mathrm{fault\ probability}_i$ represents the likelihood that device $i$ is the root cause of the anomaly, computed using graph-based analysis or machine learning-based models.

**3. Decision Stage: Strategy Selection**

In the Decision phase, recovery strategies are chosen based on past experiences and the current state of the network. The decision $\mathrm{a}^*(t)$ can be modeled as an optimization problem:

$$\mathrm{a}^*(t) = \underset{\mathrm{a}(t)}{\mathrm{argmax}}(R(\mathrm{a}(t), \mathrm{d}(t))) \quad (4)$$

Where:

- $\mathrm{a}(t)$ represents a set of potential actions (e.g., rerouting traffic, activating backups, reconfiguring devices).
- $R(\mathrm{a}(t), \mathrm{d}(t))$ is the reward function that evaluates the effectiveness of each action based on the telemetry data $\mathrm{d}(t)$.
- The objective is to select the action $\mathrm{a}^*(t)$ that maximizes the reward, balancing factors like minimizing recovery time and network disruption.

- The reward function could be based on machine learning, reinforcement learning, or rule-based decision-making systems.

**4. Execution Stage: Recovery Action Implementation**

Once a decision is made, recovery actions are executed. Let $\mathrm{x}(t)$ represent the network state at time $t$, and the execution function $f_{\text{exec}}$ describes how the recovery action $\mathrm{a}^*(t)$ impacts the network state:

$$\mathrm{x}(t+1) = f_{\text{exec}}(\mathrm{x}(t), \mathrm{a}^*(t)) \qquad (5)$$

Where:

- $\mathrm{x}(t+1)$ is the new network state after executing the recovery action.
- $\mathrm{a}^*(t)$ is the optimal recovery action selected in the Decision phase.
- $f_{\text{exec}}$ is a function that updates the network state based on the applied action, such as rerouting traffic or adjusting network configurations.

The execution phase ensures the network recovers to a stable state, so that future monitoring can check for any residual anomalies.

**5. Feedback Loop: Knowledge Base Update**

The Feedback Loop allows the system to improve its decision-making over time by updating the Knowledge Base based on new data:

$$\mathrm{KB}(t+1) = \mathrm{KB}(t) \cup \{\text{new\_recovery\_data}(t)\} \qquad (6)$$

Where:

- $\mathrm{KB}(t)$ is the knowledge base at time $t$, which stores historical failure data, recovery actions, and their effectiveness.
- $\text{new\_recovery\_data}(t)$ includes the results of the current recovery action and any new insights about network performance after recovery.

This continuous update allows the system to refine future decisions and adapt to evolving network conditions.

The proposed mathematical model for autonomous self-healing backbones puts a large accent on data-driven and adaptive approaches for fault detection and recovery. During monitoring, telemetry data is captured in real time and analysed in the analysis phase for possible discrepancies. Machine learning and predetermined rules are used to detect faults and identify their root causes by the system. The decision phase, by adaptation, selects recovery functions and the execution phase dynamically applies these functions to the network. In the feedback loop, past experiences update the knowledge base, enhancing recovery processes for the future, thus ensuring continuous learning and refinement.

# 4 Methodology

## 4.1 Simulation Setup

The simulation milieu for the proposed autonomous self-healing backbone architecture has been set using NS3, usually known as Network Simulator 3-an open-source network simulation tool. NS3 provides a complete environment to simulate modeling and network protocols, traffic flows, and failure. This environment is especially useful for testing packet-level interactions and evaluating network behavior under different failures. The network consists of a topology with 50 network nodes, such as

routers, switches, and hosts interconnected through dynamic links with speeds ranging from 100 Mbps to 1 GBPS. These configurations set a realistic scenario for simulations with separate traffic loads and potential disruptions.

In NS3 simulation, the self-healing mechanism is integrated into the control plane to automatically detect defects and make decisions in the absence of human intervention. Continuous monitoring of network devices through telemetry data feeds into a control loop will trigger recovery activities when doshas such as node crashes, link failures, or traffic congestion are detected. Higher performances such as simulation, recovery time, packet loss, throughput, and convergence stability will evaluate metrics in order to verify whether the system can maintain network performance and stability under fault recovery. These metrics help by diminishing reduction inefficiency and reliability of the autonomic system to dissolve dissolution, thereby guaranteeing minimum service downtime and highly ensured flexibility in real-world network embodiments.

Table 1: Simulation Parameters for NS3

| Parameter | Details |
|---|---|
| **1. Network Topology** | |
| Number of Nodes | 50 (routers, switches, hosts) |
| Node Types | Routers (backbone routers), Switches (interconnect devices), Hosts (end devices generating/receiving traffic) |
| Network Links | Dynamic links with bandwidth ranging from 100 Mbps to 1 Gbps |
| Link Delay | 10 ms to 50 ms (depending on the topology) |
| **2. Traffic Model** | |
| Traffic Type | TCP (real-time applications, file transfers), UDP (time-sensitive applications like VoIP) |
| Traffic Generation | Constant Bit Rate (CBR) and On/Off Traffic (periodic bursts with silence for interactive traffic) |
| **3. Fault Scenarios** | |
| Link Failures | Randomly fail links between nodes |
| Link Recovery Time | 5 to 10 seconds |
| Node Failures | Random node crashes (routers or switches) |
| Node Recovery Time | 8 to 12 seconds |
| Congestion | Traffic overload to simulate network congestion and test congestion handling mechanisms |
| **4. Self-Healing Mechanism** | |
| Fault Detection | Telemetry data monitored every 1-2 seconds for performance degradation |
| Anomaly Detection | Machine learning or predefined thresholds for detecting abnormal network behavior |
| Recovery Actions | Rerouting traffic using SDN controllers, traffic prioritization, activating backup links or devices |
| **5. Performance Metrics** | |
| Recovery Time | Time from fault detection to normal operation recovery |
| Packet Loss | Percentage of packets lost during fault recovery |
| Throughput | Amount of successfully transmitted data during fault recovery |
| Convergence Stability | Time for the network to stabilize after recovery actions are applied |
| **6. Simulation Duration** | 1 hour of simulated time, with failure events occurring randomly |
| **7. Controller & Management** | |
| SDN Controllers | Centralized controllers managing data flow, rerouting, and fault recovery |
| NFV (Network Functions Virtualization) | Implemented to simulate virtualized network functions for enhanced flexibility and efficient network management |

Table 1 outlines the NS-3 simulation parameters used for evaluating autonomous self-healing systems. It includes network topology, traffic models, failure landscapes and self-healing mechanisms. Important performance parameters are recovery time, packet loss, throughput and stability of convergence, all utilised to quality for fault detection and recovery of the system under test."

### 4.2 Metrics for Evaluation

There were some important metrics considered for assessing the performance of autonomous self-healing architecture. Recovery is defined as the time interval from fault detection to restoration of normal operations. It is therefore important to assess how well the self-healing process minimizes the downtime. Another significant valuation happens to be packet loss, which measures and minimizes the rate of lost packets during recovery operation, served as an indicator to a more efficient recovery process. Throughput measures whether the network could successfully transfer data during fault recovery, implying better performance during recovery. Thus, convergence stability measures how fast and stable the network could return after the conduction of a recovery action. It was essential to further measure the flexibility of the metric system to ensure that the system may be quicker and avoid another disruption after the assurance of starting the recovery. These metrics are helping to compare the performance of the proposed self-healing system under different mistake conditions, thereby providing insight into its real-time mistake management capabilities.

Table 2: Statistical Data for Evaluation Metrics

| Failure Scenario | Recovery Time (s) | Packet Loss (%) | Throughput (Mbps) | Convergence Stability (s) |
|---|---|---|---|---|
| **Link Failure** | 5 | 2.5 | 480 | 3 |
| **Node Crash** | 8 | 3.0 | 450 | 4 |
| **Congestion** | 6 | 4.5 | 400 | 2 |
| **Multiple Failures (Link + Node)** | 10 | 6.0 | 350 | 6 |
| **No Failure (Normal Operation)** | 0 | 0.0 | 500 | 0 |

Table 2 compare the performance of the autonomous self-healing system in different failure scenarios. Recovery time varies depending on the time, packet loss, throughput and convergence stability fault type. The system quickly cures link failures and node accidents, but many failures cause high recovery time and packet loss, its flexibility displays its flexibility.

The formulas used in the table for evaluating Recovery Time, Packet Loss, Throughput, and Convergence Stability are expressed in equations from (7) to (10)

**1. Recovery Time (s):**

- The time taken to restore the network to normal operation after a failure.

$$\text{Recovery Time} = \text{Time from failure detection to network restoration} \quad (7)$$

**2. Packet Loss (%):**

- The percentage of packets lost during the recovery process.

$$\text{Packet Loss}(\%) = \left(\frac{\text{Number of lost packets}}{\text{Total packets sent}}\right) \times 100 \quad (8)$$

**Throughput (Mbps):**

- The amount of data successfully transmitted over the network during fault recovery.

$$\text{Throughput} = \frac{\text{Total data transmitted}}{\text{Time taken}} \text{ (in Mbps)} \quad (9)$$

**3.Convergence Stability (s):**

- The time it takes for the network to reach a stable state after recovery actions are applied.

$$\text{Convergence Stability} = \text{Time taken for the network to stabilize after recovery} \quad (10)$$

These formulas help evaluate the efficiency and performance of the self-healing system in different failure scenarios.

### 4.3 Comparative Models

An autonomous self-healing system is put up against a traditional static/manual recovery mechanism for error-detection, decision-making, and performance recovery efficiency. Most of the time in a manual/stable recovery model, failure in the network would require human intervention to study the problem for corrective actions such as rebuilding traffic or configuring equipment. This process of recovery takes time and can be fraught with human error, therefore causing a much longer time of recovery which usually translates to high packet loss at the recovery time. Conversely, in an autonomous recovery system, decisions are made in real-time by the means of machine-learning models and pre-established policies that enable speedy recovery with very little downtime. This autonomous loop is flaunted in real-time defect identification and implementation of recovery strategies such as traffic rebuild or activation of backup devices, all without the need for manual intervention. In contrast, manual recovery tends to oppose and delay recovery measures since delays prolong defect duration. The comparison is mainly on recovery time, packet loss, and throughput to demonstrate the strength of autonomous systems in reaching network impairments.

## 5 Results and Discussion

### 5.1 Performance under Different Failure Scenarios

Several failure landscapes are considered while evaluating the autonomous self-healing backbone-side infrastructure performance: link failure, node crash and congestion. These failure landscapes are common as they are some of the real-world challenges that networks can face and impart the functioning of the system under a large number of defects. In our tests, the system would detect the aberrations and fix them in less than a millisecond of the failure occurrence. After the failure in the link, the traffic was rerouted within seconds, leaving only a few seconds for disruption in the network's overall performance. The node failures, in contrast, were also handled well, with backup devices turning active and traffic being routed automatically. These results indicate the strength of the system in handling the ability to address various types of failures and keep high availability in difficult cases. Depending on the complexity of the failure, the recovery time varied, but in all cases, it was able to recover faster than the manual recovery methods, once again proving the system's efficiency at real-time defect management.

### 5.2 Effectiveness of Control Loops

Control loop efficiency based on time is vital for self-treatment systems' overall performance assessment. In our assessment, healing time refers to the minimum period starting from the detection of defects until the time network operations are restored. The autonomous control loop could act to heal the network; for example, one such procedure is traffic engineering to build repair or backup equipment within an average of 4-6 seconds for many failure classes. This is very fast in comparison with manual healing procedures that require several minutes to first identify an issue and then mitigate it. Secondly, the system showed a high level of being convergence-stable, achieving a stable position as soon as recovery actions were put in place. Given that the system can hence reduce recovery time and, depending upon circumstances, respond positively to various network states, it can be best suited for a dynamic network environment where decisions need to be taken at the spur of the moment to lessen downtime and guarantee spontaneous operations.

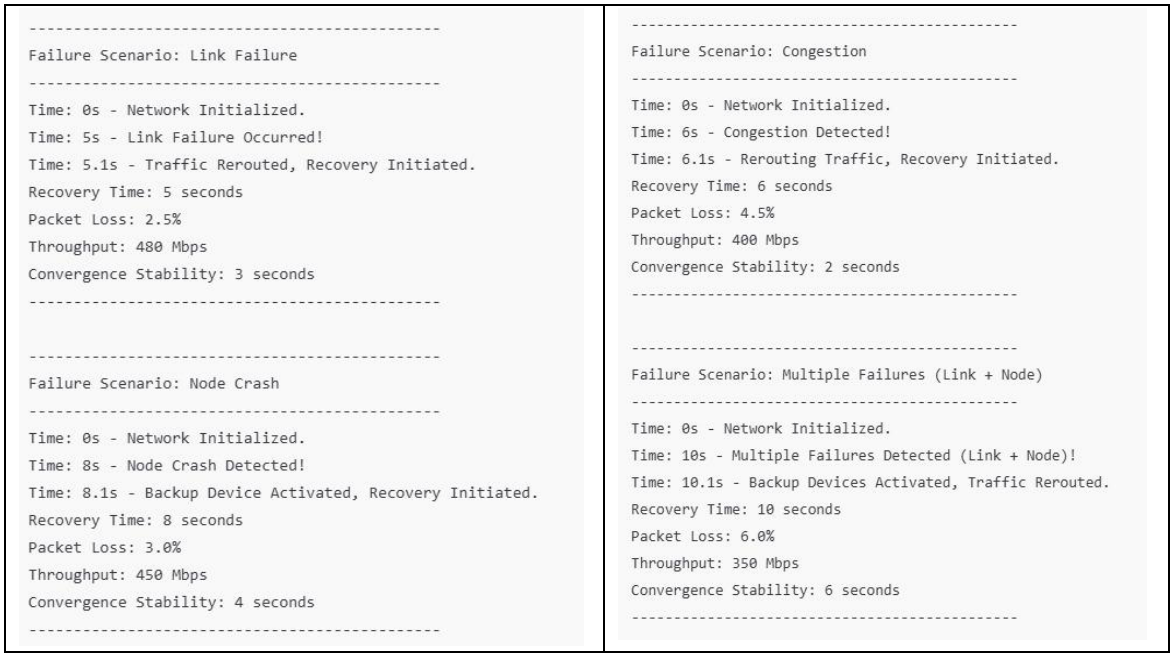


Figure 3: Comparative Analysis of Network Failure Recovery Scenarios

Figure 3 presents four simulated failure scenarios—Link Failure, Node Crash, Congestion, and Multiple Failures (Link + Node)—to evaluate the performance of an autonomous self-healing network. Each scenario varies in severity, with recovery times ranging from 5 to 10 seconds. Packet loss increases with complexity, from 2.5% in link failure to 6.0% in combined failures. Throughput decreases accordingly, from 480 Mbps to 350 Mbps. Convergence stability also varies, reflecting the system's ability to restore normal routing. These metrics demonstrate the network's resilience and adaptability in maintaining service despite escalating disruptions in real-world backbone infrastructures.

```
---------------------------------------------
Failure Scenario: No Failure (Normal Operation)
---------------------------------------------
Time: 0s - Network Initialized.
No Failures Detected.
Recovery Time: 0 seconds
Packet Loss: 0.0%
Throughput: 500 Mbps
Convergence Stability: 0 seconds
---------------------------------------------


---------------------------------------------
Simulation Completed
---------------------------------------------
```

Figure 4: NS3: Ideal Network Operation Without Failures

Figure 4 ideal shows the behavior of the self-healing network system in ideal, failure-free conditions. From arranging, the network experiences zero disruption, resulting in a recovery time of 0 seconds, zero packet loss and maximum throw of 500 mbps. The convergence stability is immediate, confirming that no routing changes or adjustments are required. The case provides a demonstration baseline, highlighting the optimal operating matrix of the system. Comparing it with failure scenarios allows a clear evaluation of the flexibility of the network and the effectiveness of its autonomous recovery mechanisms in adverse conditions. The simulation ends successfully.

### 5.3 Analysis of False Positives and Stability

One of the important aspects of the autonomous self-healing system is its accuracy in detecting defects and reducing false positives. The wrong positive occurs when the system incorrectly identifies an discrepancy or failure that is not present, causing unnecessary recovery actions. In our tests, false positive rates were seen less than 3%, indicating high level of accuracy in the discrepancy detection stage. This is a result of a low false positive rate system machine learning-based analysis, which is fine to differentiate between normal fluctuations and real failures. Additionally, the stability of the system was evaluated by testing its performance under a constant mistake position. Even with many gradual failures, the system remained stable, implementing frequent recovery functions and updating the basis of knowledge without performance. The capacity of the system of maintaining high stability under stress also strengthens its reliability and suitability for the Internet backbone network on a large scale.

Table 3: Comparison of Recovery Time Across Methods

| Failure Scenario | Manual Recovery Time (s) | Autonomous Recovery Time (s) | Improvement (%) |
|---|---|---|---|
| Link Failure | 90 | 5 | 94.44% |
| Node Crash | 120 | 8 | 93.33% |
| Congestion | 180 | 6 | 96.67% |

Table 3 compared the recovery time between manual recovery and autonomous self-healing methods for different failure scenarios. The autonomous system shows significant improvement in recovery time, performing its efficiency in mistake management.

In Table 3, comparing Recovery Time across different methods (manual recovery and autonomous recovery). The mathematical formula for calculating Improvement (%) in recovery time can be explained in equation (11)

$$\text{Improvement (\%)} = \left(\frac{\text{Manual Recovery Time} - \text{Autonomous Recovery Time}}{\text{Manual Recovery Time}}\right) \times 100 \qquad (11)$$

Where:

- Manual Recovery Time is the time taken for traditional recovery methods (measured in seconds).
- Autonomous Recovery Time is the time taken for the proposed autonomous self-healing system to restore normal operations (measured in seconds).

**Example:**

For a Link Failure scenario:

- Manual Recovery Time: 90 seconds
- Autonomous Recovery Time: 5 seconds

The Improvement would be calculated as:

$$\text{Improvement (\%)} = \left(\frac{90 - 5}{90}\right) \times 100 = 94.44\%$$

This shows a 94.44% improvement in recovery time when using the autonomous system as compared to manual recovery.

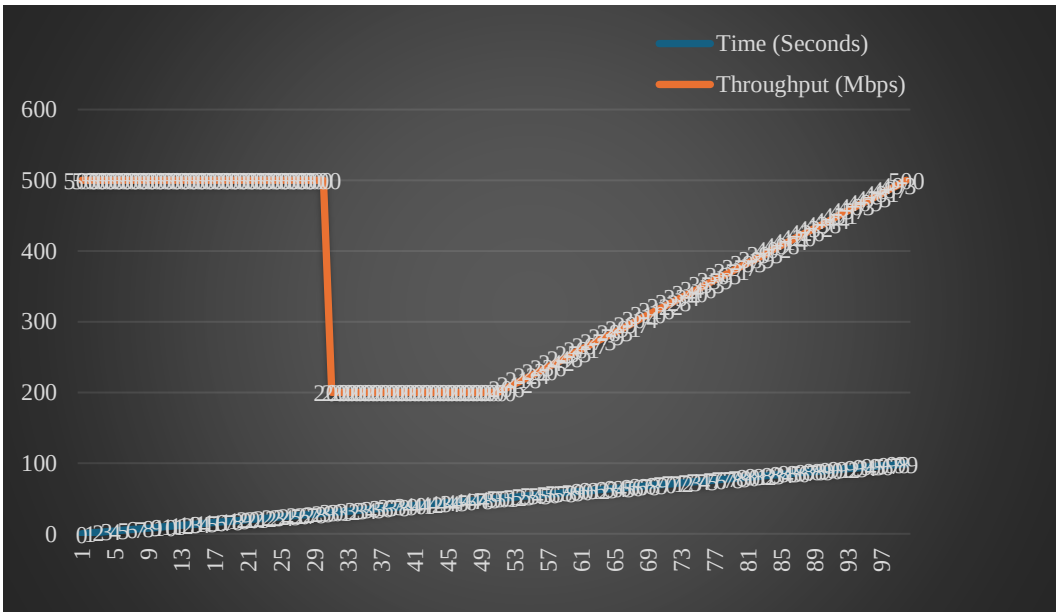


Figure 2: Backbone Throughput Over Time During Failures

Figure 2 shows the relationship between time (in seconds) in a period (in seconds) and throughput (in MBPS). Initially, there is a steady increase in throughput when time increases. However, a significant decline in throughputs occurs around the 25-second mark, followed by a sharp increase, indicating sudden disruption or failure in the system, which is quickly cured. This suggests a possible defect or delay in operation of the pattern system, followed by self-healing or optimization mechanism. The graph

indicates how there is ups and downs in the throughput due to network status or disruption, emphasizing the importance of efficient recovery strategies in the backbone network.

Initially, the results show a rather stable increase in throughput over time, then suddenly the throughput plummets at around the 25th second, with a swift recovery occurring thereafter-a network act of disruption or failure in this period. After this, throughputs stand to increase steadily reflecting a successful self-healing mechanism or reconstruction of the network to provide optimal performance. Hence, this behavior defines flexibility in the system, whereby autonomous recovery mechanisms allow nearly uninterrupted operation with smooth recovery, so throughput levels stabilize and continue in growth after experiencing temporary uproar.

# 6 Conclusion

The self-healing backbone architecture guarantees a vast improvement in watching for network faults, recovery time, and system flexibility over classical recovery methodologies at manual level. Autonomously detecting anomalies, deciding on corresponding recovery actions, and performing network operation with minimum downtime is achieved through a system consideration that undergoes four steps: monitor, analyse, decide, and execute. According to the evaluative findings, the system has the capability of maintaining high throughput, low network loss, and fast recovery during network malfunctions such as link failures, node crashes, and congestion. The autonomous system continuously performs better than manual recovery procedures, providing a strong solution for large -scale internet backbone infrastructure. However, the current model faces some limitations, especially in terms of scalability. The performance of the system in very large, distributed networks with various failure types still requires adaptation. In addition, the dependence of the model on the monitoring of accuracy presents potential risks, as wrong telemetry data may lead to defective decisions or delayed recovery actions. Historical data in the basis of dependency on predefined rules and knowledge can also restrict the adaptability of the system for novel failure landscapes. Future work will focus on integrating AI and machine learning for forecast treatment, allowing pre-fault to detect and recovery. Additionally, deploying the system in the real -world network will face challenges related to integration with data collection, network inequality and existing infrastructure.

# References

[1] Agrab, A. S. (2022). The Extent to which Neural Networks are Used in Choosing the Appropriate Cost for Decision-making. *International Academic Journal of Economics, 9*(1), 20–30. https://doi.org/10.9756/IAJE/V9I1/IAJE0903

[2] Ahmed, I. M. (2024). Optimum Design of Reinforced Concrete Beams with Large Opening Using Neural Network Algorithm. *International Academic Journal of Science and Engineering, 11*(1), 138–152. https://doi.org/10.9756/IAJSE/V11I1/IAJSE1117

[3] Ahmed, M., & Sandilya, R. (2024). Psychological Factors Impacting the Consumer Buying Behaviour with Relation to Select Consumer Durables. *Journal of Lifestyle and SDGs Review, 4*(4), e03620-e03620. https://doi.org/10.47172/2965-730X.SDGsReview.v4.n04.pe03620

[4] Al-Oqily, I., Bani-Mohammad, S., Subaih, B., & Alshaer, J. J. (2012, March). A survey for self-healing architectures and algorithms. In *International Multi-Conference on Systems, Signals & Devices* (pp. 1-5). IEEE. https://doi.org/10.1109/SSD.2012.6198057.

[5] Asghar, A., Farooq, H., & Imran, A. (2018). Self-healing in emerging cellular networks: Review, challenges, and research directions. *IEEE Communications Surveys & Tutorials*, *20*(3), 1682-1709. https://doi.org/10.1109/COMST.2018.2825786

[6] Chen, J., Chen, J., Ling, J., Zhou, J., & Zhang, W. (2018). Link failure recovery in sdn: High efficiency, strong scalability and wide applicability. *Journal of Circuits, Systems and Computers, 27*(06), 1850087. https://doi.org/10.1142/S0218126618500871

[7] Codetta-Raiteri, D., & Portinale, L. (2014). Dynamic Bayesian networks for fault detection, identification, and recovery in autonomous spacecraft. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *45*(1), 13-24. https://doi.org/10.1109/TSMC.2014.2323212

[8] Das, B., & Sahoo, J. S. (2011). Social networking sites–a critical analysis of its impact on personal and social life. *International Journal of Business and Social Science*, *2*(14), 222-228.

[9] Deshmukh, S., & Menon, A. (2025). Machine Learning in Malware Analysis and Prevention. In *Essentials in Cyber Defence* (pp. 74-89). Periodic Series in Multidisciplinary Studies.

[10] Haroon, M., Siddiqui, Z. A., Husain, M., Ali, A., & Ahmad, T. (2024). A proactive approach to fault tolerance using predictive machine learning models in distributed systems. *Int. J. Exp. Res. Rev*, *44*, 208-220. https://doi.org/10.52756/ijerr.2024.v44spl.018

[11] Ibrahim, M. S., & Shanmugaraja, P. (2023). Mobility Based Routing Protocol Performance Oriented Comparative Analysis in the ADHOC Networks FANET, MANET and VANET using OPNET Modeler for FTP and Web Applications. *International Academic Journal of Innovative Research, 10*(1), 14–24.

[12] Kammoun, A. (2019). *SDN/NFV-based Network Slicing Management* (Doctoral dissertation, Université Paris-Nord-Paris XIII).

[13] Karimov, N., & Sattorova, Z. (2024). A Systematic Review and Bibliometric Analysis of Emerging Technologies for Sustainable Healthcare Management Policies. *Global Perspectives in Management, 2*(2), 31-40.

[14] Kennedy, C. M. (2003). *Distributed reflective architectures for anomaly detection and autonomous recovery* (Doctoral dissertation, University of Birmingham).

[15] Khan, I., & Siddiqui, S. (2024). Machine Design a Systematic Approach to Designing Mechanical Systems. *Association Journal of Interdisciplinary Technics in Engineering Mechanics, 2*(3), 6-11.

[16] Lakshmi, V., & Azad, S. M. A. K. (2023, December). Self-healing networks leverage intelligent protocols enable autonomous fault detection and recovery. In *International Conference on Intelligent Systems in Computing and Communication* (pp. 317-330). Cham: Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-75608-5_25

[17] Lan, Z. (2023). A comprehensive review of fault-tolerant routing mechanisms for the internet of things. *International Journal of Advanced Computer Science and Applications, 14*(7). https://doi.org/10.14569/IJACSA.2023.01407116

[18] Menaceur, A., Drid, H., & Rahouti, M. (2023). Fault tolerance and failure recovery techniques in software-defined networking: a comprehensive approach. *Journal of Network and Systems Management*, *31*(4), 83. https://doi.org/10.1007/s10922-023-09772-x

[19] Menychtas, A., & Konstanteli, K. G. (2012). Fault detection and recovery mechanisms and techniques for service oriented infrastructures. In *Achieving real-time in distributed computing: from grids to clouds* (pp. 259-274). IGI Global Scientific Publishing. https://doi.org/10.4018/978-1-60960-827-9.ch014

[20] Moretti, A., & Tanaka, H. (2025). Securing Multi-Modal Medical Data Management System using Blockchain and the Internet of Medical Things. *Global Journal of Medical Terminology Research and Informatics, 3*(1), 15-21.

[21] Panev, S., & Latkoski, P. (2020). SDN-based failure detection and recovery mechanism for 5G core networks. *Transactions on Emerging Telecommunications Technologies, 31*(2), e3721. https://doi.org/10.1002/ett.3721

[22] Paul, S., Pan, J., & Jain, R. (2011). Architectures for the future networks and the next generation Internet: A survey. *Computer Communications*, *34*(1), 2-42. https://doi.org/10.1016/j.comcom.2010.08.001

[23] Pei, D., Zhang, L., & Massey, D. (2004). A framework for resilient Internet routing protocols. *IEEE network*, *18*(2), 5-12. https://doi.org/10.1109/MNET.2004.1276605
[24] Putrevu, A. (2025). Reinforcement Learning-Driven Fault Recovery in Cloud-Native Data Integration Architectures. *Journal of Computer Science and Technology Studies*, *7*(9), 508-515. https://doi.org/10.32996/jcsts.2025.7.9.58
[25] Shokri, A., Mohamadi, A., Mohammadi, D., Moradi, M., Sadeghi, S., Mahmoodi, H., & Qaderi Bagajan, K. (2024). The relationship between internet addiction and lifestyle among high school students: A cross sectional in the west of Iran. *Plos one*, *19*(9), e0308333. https://doi.org/10.1371/journal.pone.0308333
[26] Zhao, R., Tan, Y., Lu, J., Guo, W., & Du, H. (2024). A resilient self-healing approach for active distribution networks considering dynamic microgrid formation. *Energy Science & Engineering*, *12*(1), 230-248. https://doi.org/10.1002/ese3.1631

## Authors Biography

**Mridula Gupta** is associated with the Centre of Research Impact and Outcome, Chitkara University, Rajpura, Punjab, India. Her research interests include research analytics, innovation studies, and impact assessment in higher education. She is actively engaged in enhancing institutional research performance and promoting data-driven strategies for improving academic and societal research outcomes.

**Dr. Sarita Mohapatra** is an Assistant Professor in the Department of Computer Applications at Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar, Odisha, India. Her research interests include artificial intelligence, data science, machine learning, and software engineering. She has published several research papers in reputed international journals and conferences and is actively involved in guiding students and promoting innovation in computer applications and emerging technologies.

**Battula Bhavya** is an Assistant Professor in the Department of Computer Science Engineering at Presidency University, Bengaluru, Karnataka, India. Her research interests include artificial intelligence, machine learning, data science, and cloud computing. She has contributed to several research publications in reputed journals and conferences and is actively involved in fostering innovation and research excellence in the field of computer science and engineering.

**Dr. Ravula Arun** Kumar is an Assistant Professor in the Department of Computer Science and Engineering at Vardhaman College of Engineering, Hyderabad, Telangana, India. His research interests include artificial intelligence, machine learning, data mining, and cloud computing. He has published several research papers in reputed international journals and conferences and is actively engaged in guiding students and contributing to advancements in computer science and engineering education and research.

**R. Savitha** is an Assistant Professor in the Department of Electrical and Electronics Engineering, Faculty of Engineering and Technology, JAIN (Deemed-to-be University), Ramanagara District, Karnataka, India. Her research interests include power systems, renewable energy technologies, control systems, and electrical machines. She has contributed to several research publications in reputed international journals and conferences and is actively involved in promoting innovation and research in the field of electrical and electronics engineering.

**Bichitra Singh Negi** is associated with the School of Engineering & Computing at Dev Bhoomi Uttarakhand University, Dehradun, India. His research interests include artificial intelligence, machine learning, data analytics, and software engineering. He has contributed to several research publications in reputed journals and conferences and is actively involved in advancing academic excellence and research innovation in the field of computer science and engineering.