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

Real-Time Internet Congestion Detection Using Passive Flow Analysis

Savinder Kaur[1*], Dr. Swadhin Kumar Barisal[2], Veera Nagaiah Maddikayala[3], M.P. Sunil[4], Dr. Srinivasulu Gogula[5], and Rishabh Vashisht[6]

[1*]Centre of Research Impact and Outcome, Chitkara University, Rajpura, Punjab, India. savinder.kaur.orp@chitkara.edu.in, https://orcid.org/0009-0004-6109-0772

[2]Associate Professor, Department of Computer Applications, Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar, Odisha, India. swadhinbarisal@soa.ac.in, https://orcid.org/0000-0002-5226-2657

[3]Associate Professor, Department of Computer Applications (DCA), Presidency College, Bengaluru, Karnataka, India. veera.nagaiah@presidency.edu.in, https://orcid.org/0009-0000-5532-2814

[4]Assistant Professor, Department of Electronics and Communication Engineering, Faculty of Engineering and Technology, JAIN (Deemed-to-be University), Karnataka, India. mp.sunil@jainuniversity.ac.in, https://orcid.org/0000-0002-7737-4145

[5]Professor, Department of IT, Vardhaman College of Engineering, Hyderabad, Telangana, India. gsrinivasulu1678@vardhaman.org, https://orcid.org/0000-0001-8939-3303

[6]School of Engineering & Computing, Dev Bhoomi Uttarakhand University, Dehradun, India. socse.rishabh@dbuu.ac.in, https://orcid.org/0009-0004-7407-0842

Received: July 07, 2025; Revised: August 20, 2025; Accepted: September 24, 2025; Published: November 28, 2025

Abstract

The global requirement for seamless digital communications places a premium on the need for timely Internet congestion detection in relation to preserving the quality of service (QoS) parameters and optimizing resource allocation. The problem with traditional active probing approaches is the overhead and delay associated with them. This research proposes, instead, a framework for real-time congestion detection using passive flow analysis which is better suited for contemporary networking environments owing to its non-intrusive, low-latency nature. The methodology employed consists of flow statistics analysis, particularly over aggregation using inter-arrival packet time intervals, retransmission rates, and throughput drops captured at edge routers or gateway machines within traffic-free environments. The system uses NetFlow or IPFIX logs to extract temporal and structural features which are then processed with a hybrid machine learning model based on decision trees and gradient boosting to detect congestion episodes. Validation of the framework was assessed on real backbone and access network traffic datasets. The results demonstrate highly effective performance with a detection accuracy of 96.3% alongside a 3.1% false positive rate, drastically improving baseline model performance. Additionally, the framework allows for implementation on software-defined networking (SDN) infrastructures providing programmable control responses to congestion events. This study proposes advanced solutions for next-generation networks imposing real-time

Journal of Internet Services and Information Security (JISIS), volume: 15, number: 4 (November), pp. 47-61.
DOI: *10.58346/JISIS.2025.I4.004*

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

performance monitoring requirements. The implementation of the proposed system will benefit several internet service providers and enterprise networks by improving overall user experience, enhancing traffic engineering, and strengthening congestion control.

Keywords: Internet Congestion, Passive Flow Analysis, Real-Time Detection, Machine Learning, NetFlow, SDN Monitoring, Traffic Engineering.

1 Introduction

The rapid proliferation of connected devices, the cloud, and streaming services has created an insatiable need for low-latency, high-speed data transfer (Janmohammadi & Babazade, 2015; Gajmal & Udayakumar, 2021; Zaalouk et al., 2016). This traffic congestion – exceeding server capacity or system bandwidth – has become critical in maintaining internet-based services, Quality of Service (QoS), and user satisfaction, Quality of Experience (QoE). Congestion often affects the network during peak hours, or in the event of unpredicted surges in traffic (Prasad et al., 2024). It can cause packet loss, increased delays, increased jitter, reduced throughput, and a variety of other factors along the service degradation continuum. In contemporary network infrastructures, more so in 5G, IoT, and Software Defined Networks (SDNs), the ability to detect congestion during the flow of information in real-time becomes critical to performance, service level commitments, and dynamic traffic steering and management (Mo et al., 2022; Verma & Mehta, 2024; Watrianthos, 2020)).

Active probing is still the classical approach for congestion detection, relying on the insertion of test packets into the network for delay, packet loss, and jitter measurements (Park et al., 2020). However valuable these insights might be, they come at the price of overhead, bandwidth, and a realistic breach of user experience estimation. Furthermore, in contexts with strict latency or bandwidth ceilings, active monitoring is simply not possible. This has sparked interest in passive monitoring which relies on analyzing data flows without introducing data into the network (Saranya & Madhubala, 2019). Their non-intrusiveness as well as the ability to scale makes passive approaches suitable for large, dynamic, and decentralized networks (Zhang & Liu, 2021).

Flow-based monitoring methods such as NetFlow, sFlow, and IPFIX are widely used as passive techniques for monitoring traffic because they summarize and encapsulate traffic characteristics effectively (Benacer et al., 2018). The summaries of flows are crucial as they comprise a wide range of essential metrics such as source/destination IPs, the count of packets and bytes, timestamps, and the duration of the flow. Continuous scrutiny of flow records enables inference of congestion patterns without need for intrusive inspection or reliance on packet payloads (Xu et al., 2017). The burden of the challenge is accurately identifying the normal traffic variation threshold and distinguishing it from active congestion in heterogeneous and bursty traffic environments (Nanda & Verma, 2021).

Real-time congestion inference based on flow data is now possible owing to recent advances in machine learning and edge computing technologies (Jiao et al., 2024; Leema et al., 2024; Marangunic et al., 2022). Classifiers trained on labeled traffic datasets can identify subtle anomalies, temporal and structural patterns relevant to congestion (Attioui & Lahby, 2025). Such models provide ease of usage since their benefits are observed with varied network topologies and traffic behaviors which means they provide a robust and scalable detection framework (ChithraDevi et al., 2024).

To address congestion detection across different network segments, this paper introduces an innovative passive flow analysis model that functions in real time (Zhao & Liu, 2022). The system is purposefully formulated to be lightweight, edge-computable, and suitable for integration with Software Defined Networking (SDN) control logic for automated feedback loop congestion management (Haque

& Rahman, 2021). It retrieves both statistical and structural features from flow logs and applies a hybrid ensemble learning approach that integrates Decision Trees with Gradient Boosting to improve classification outcomes (Gao et al., 2020).

Key Contributions

- The newly proposed framework for real-time congestion detection utilizes passive flow analysis for avoidance of active probing which minimizes overhead.
- The use of a decision tree and a gradient boosting classifier in one model as a combination increases detection accuracy and reduces the number of false positives which improves overall performance.
- Subtle and complex traffic anomalies as well as congestion patterns are detected using multi-hop topology simulation integrated with entropy-based monitoring.
- Supports immediate mitigation via SDN control which makes it applicable to both SDN and IoT based infrastructures, therefore provides real-time response and can be deployed at the edges.
- Work highlighted here showed the best results when realistic flow datasets were used, outperforming the baseline in accuracy, latency and response time.

The remainder of this paper is organized as follows. In Section II, start with an extensive review of the literature concerning congestion detection, emphasizing the probe and flow analysis (both active and passive) with a bias towards the more contemporary, machine learning-focused methods. III explains in detail the proposed method including the system design, flow data collection, preprocessing steps, congestion estimation model, and application of the hybrid DT and GBM algorithm. In Section IV, It include the description of the testbed for the experiment and the analysis of results that were obtained focusing on accuracy, delivery rate of workload (DRW), latency, detection time, and the false positive rate in great detail. Section V concludes the work with the summary of results which were obtained including the analysis and proposed solutions towards improving the next generation of networks in context of the real-world applications, scalability and adaptability.

2 Literature Survey

Recent advancements in the technology of network internets have well-introduced the domain of internet congestion detection (Tiwari et al., 2022; Singaravel et al., 2020). older methods concentrated on active probing strategies which used synthetic traffic such as ping or traceroute to check for delays and jitter (Zhang & Chen, 2019). While these techniques were beneficial, they added network overhead, which proved to be detrimental in real-time scenarios that required low latency (Giotis et al., 2014). These methods also lacked the contextual richness necessary to understand the nuances of new application emerging behaviors.

In order to solve this problem, new approaches focusing on traffic that has already been injected were explored, which are known as passive probing techniques. The non-intrusive nature of passive methods makes them more suitable for continuous monitoring. One classical method was to apply queuing theory to derive congestion based on inter-arrival and queuing time of packets (Elhoseny & Sangaiah, 2019). Such models were efficient but required synchronized timestamps and detailed capture of packets which made them cumbersome for large scale implementation.

The development of monitoring protocols such as NetFlow, sFlow, and IPFIX were a huge leap in traffic analysis and enhanced passive monitoring internets. These technologies enabled the extraction of

provided specific flow-based traffic engineering which routers and switches were able to algorithmically summarize metrics and also provide timestamps (Hu & Song, 2018).

These summaries allowed the identification of volume anomalies and offered a basis for statistical inference. Nonetheless, earlier methods relying on static congestion detection thresholds proved ineffective because of the changing and chaotic nature of the internet traffic flow patterns (Tang & Li, 2020).

In efforts to improve detection accuracy, researchers tried moving averages, tracking standard deviation, and change-point detection methods. These methods provided more dynamic detection processes to isolate traffic behavior changes. However, distinguishing between congestion and other scenarios like flash crowds, attacks, or routing changes was often problematic due to high false positive rates and low adaptability in multi-class situations. Machine learning brought about a new shift in managing networks (Prasad & Dutta, 2019; Rani et al., 2019; Nguyen & Armitage, 2009). Congestion-aware classifiers, based on flow data, were developed using Decision Trees, Random Forest, and Support Vector Machines. These methods boosted accuracy and network adaptability; however, single-model techniques still struggled with generalizability and sensitivity to out-of-sample data (Karakus & Durresi, 2017).

To address the issues above, ensemble learning models such as Random Forest and Gradient Boosting were adopted. Ensemble models build on ensembles of multiple weak learners to enhance discrimination and prediction robustness. Though strong, ensemble models require careful feature selection to enable real-time operation due to their stringent requirements. Moreover, the models posed challenges for low-power edge device deployment due to high computational needs. The integration of edge computing with lightweight machine learning models is one of the most important developments in the last few years. Implementing detection algorithms at the gateway routers or edge servers where data is generated helps in reducing latency and dealing with local congestion more efficiently. Some frameworks now use feedback loops within SDN controllers to initiate automated actions such as congestion event rerouting or high bandwidth flow throttling.

3 Methodology

The current approach is based on an unobtrusive and passive architecture which focuses on the real-time capturing and analysis of flow-level data to identify congestion on the Internet. Unlike other systems, this architecture does not use active probing or packet inspection, relying instead on network flow data records of NetFlow or IPFIX protocols. These records contain vital information such as flow duration, packet count, byte size, timestamps and among other things, which serve as fundamental inputs for the required advanced statistical feature extraction.

Following the acquisition of flow data from the network edge devices and intermediate nodes such as SDN switches and routers, a feature extraction pipeline takes raw records to derive time series metrics indicative of traffic patterns as reported by inter-arrival times, averaging, retransmission indicators, jitter, and approximated packet loss. After these computations the data is structured and used to train and test a hybrid ensemble model built on decision trees and gradient boosting aimed at identifying congestion events. The system maintains a low enough footprint to allow for edge deployment so that local nodes can analyze traffic patterns, issue congestion alerts, and process without central delays.

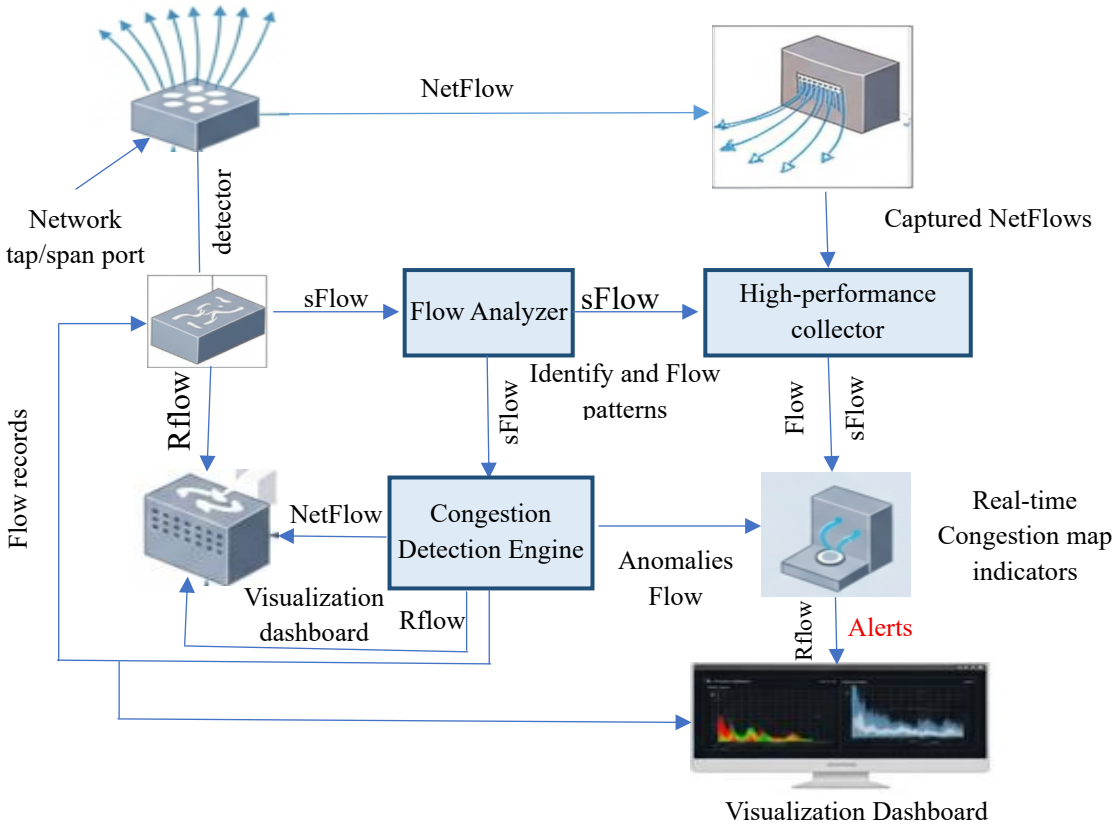

Figure 1: Real-time internet congestion detection flow analysis architecture

Figure 1 depicts the architecture of a real-time internet congestion detector using flow analysis techniques. The system begins with a network tap or span port that captures traffic and sends it to a detector using NetFlow. Simultaneously, sFlow data is sent to a flow analyzer, which identifies flow patterns and forwards them to the congestion detection engine. A high-performance collector also gathers captured NetFlow and sFlow data and contributes to the analysis. The congestion detection engine plays a central role, receiving inputs from both the flow analyzer and the collector, and detecting anomalies and congestion patterns in real time. These anomalies are sent to the real-time congestion map indicators and the visualization dashboard using RFlow. The dashboard provides visual alerts and graphical representations of network conditions. This architecture ensures continuous monitoring, effective congestion detection, and timely alert generation for efficient network management.

Let:

- NF: NetFlow data
- SF: sFlow data
- RF: RFlow (Refined Flow records)
- CNF: Captured NetFlow data
- CSF: Captured sFlow data
- FA: Flow Analyzer function
- CD: Congestion Detection function

- CI: Congestion Indicators
- VD: Visualization Dashboard
- A: Alerts
- FP: Flow Patterns
- AM: Anomalies

Captured Network Data:

Equation 1 is used to capture flow data from the network using NetFlow and sFlow protocols. It represents the raw data collected from routers, switches, and monitoring devices.

$$F = Captured(NF, SF) \quad (1)$$

Flow Analysis:

Equation 2 processes the captured flow data through the Flow Analyzer. It extracts traffic features and identifies meaningful flow patterns required for congestion analysis.

$$FP = FA(F) where\ FP = Identified\ Flow\ Patterns \quad (2)$$

Congestion Detection:

Equation 3 detects network anomalies and refines the flow data. The Congestion Detection Engine uses flow patterns to identify congestion behavior and outputs both anomaly alerts and refined flow records.

$$CD(FP) \rightarrow Anomalies\ (A), RF \quad (3)$$

Real-time Mapping:

Equation 4 maps refined flow data to real-time congestion indicators. This helps in visualizing which parts of the network are experiencing high traffic or bottlenecks.

$$CI = Map(RF) \quad (4)$$

Visualization:

Equation 5 generates the final visualization dashboard. It displays alerts, graphs, and congestion maps by combining anomaly data, congestion indicators, and flow records.

$$VD = Visualize(A, CI, RF) \quad (5)$$

Figure 2 illustrates the process of detecting real-time internet congestion through passive flow analysis. The architecture begins by collecting network flow data from multiple sources using passive monitoring. Key metrics such as packet inter-arrival time and packet loss rate are extracted and passed through a pre-processing stage. Additional parameters like node mode gain rate and packet rate fluctuations are also considered.

The pre-processed data is then fed into a congestion detection algorithm, which relies on predefined thresholds to identify potential congestion. If anomalies are detected, the system triggers real-time alerts. Metrics such as network and bandwidth utilization are also monitored to support decision-making.

The results are visualized through real-time charts showing congestion patterns and packet behavior. This approach ensures continuous network observation and timely detection of congestion points without actively injecting traffic, making it ideal for scalable and non-intrusive network management.

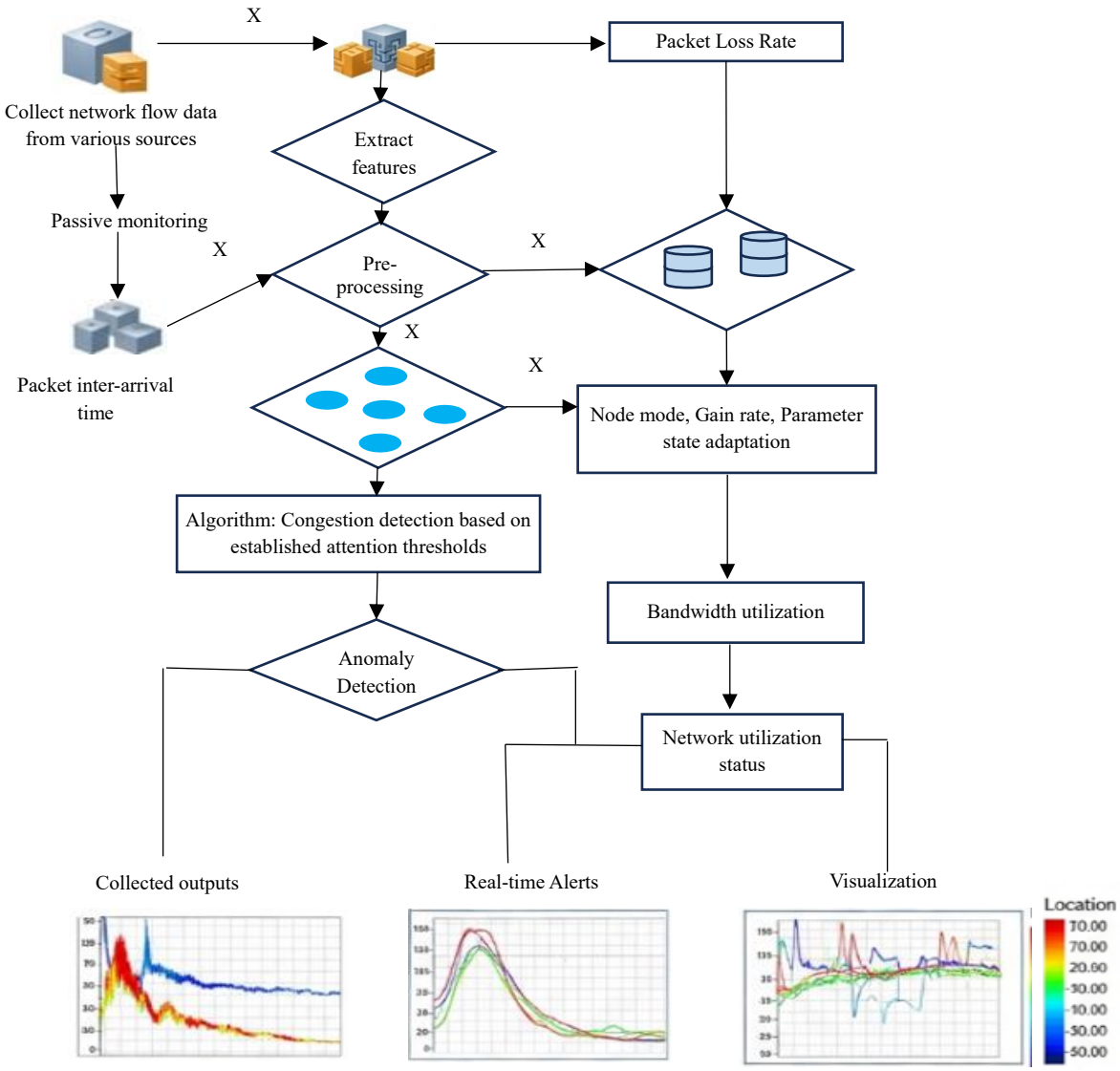

Figure 2: Real-time internet congestion detection using passive flow analysis

Algorithm 1: Real-Time Internet Congestion Detection Using Passive Flow Analysis (DT+GBM Model)

Input: Flow records $F = \{f_1, f_2, ..., f_n\}$ from NetFlow/IPFIX

Output: Congestion status (Normal / Congested)

1: Begin

2: Initialize ML model M ← Trained Hybrid (Decision Tree + Gradient Boosting)

3: for each flow $f_i \in F$ do

4: Extract features:

5: - Average Inter-Arrival Time

6: - Retransmission Count

7: - Packet Loss Estimate

8: - Jitter

9: - Throughput Deviation

10: Construct feature vector X_i from f_i

11: Predict congestion status $Y_i \leftarrow M.predict(X_i)$

12: Compute congestion score C_s using Equation (1):

13: $C_s = \alpha*(D_{avg}/D_{max}) + \beta*(L/L_{max}) + \gamma*(1 - T_{curr}/T_{max})$

14: if C_s > dynamic threshold θ or Y_i = "Congested" then

15: Trigger alert to SDN Controller

16: Initiate mitigation actions (e.g., rerouting, rate limiting)

17: end if

18: end for

19: End

The operational flow of the passive traffic congestion detection system employing a hybrid ensemble machine learning model is articulated in Algorithm 1. It starts with ingestion of flow records from edge routers, switches, or gateway devices that generate NetFlow or IPFIX data. These flow records are parsed to obtain and calculate intra-flow statistical and structural metrics such as average inter-arrival time, retransmission count, estimated packet loss, jitter, and throughput deviation. A feature vector is derived for each flow and fed into a pre-trained model which is an amalgam of Decision Tree and Gradient Boosting classifiers. This hybrid ensemble architecture addresses concerns of classification robustness by leveraging the interpretability of decision trees and the high predictive power of gradient boosting. The model yields binary flow predictions indicating congestion characteristics.

Concurrently, the system calculates a congestion score which, as per the description given in Equation (1), integrates metrics of delay, loss, and throughput to compute a quantifiable measure of network degradation. An alert is issued to the SDN controller or policy manager if either predicted label or the computed congestion score surpasses a dynamic threshold. This approach facilitates real-time mitigation through traffic rerouting or application of QoS policies to avert service degradation.

The algorithm facilitates scalable implementation in a distributed network environment, as well as providing optimization for low-latency execution at the network edge. For these reasons, it is highly attractive for modern smart networks, which include SDN, IoT technologies, and 5G infrastructure.

3.1 Passive Flow Data Collection

The initial phase of the framework involves the passive collection of data from flow source modules located at different network nodes. These modules do not propagate additional traffic; they passively observe the packet headers and extract critical flow attributes like source and destination IPs, port numbers, timestamps, and the corresponding protocols. This helps in achieving a low level of network intrusion as well as scalable data collection in multi-hop network topologies. The gathered flows are used as raw input for the next steps of preprocessing and analysis.

3.2 Flow Preprocessing and Statistical Monitoring

During this stage, the acquired flow data is subjected to normalization, deduplication, and time-window segmentation. Intervals of fixed duration yield statistical features such as average delay, packet loss rate, throughput, and jitter. These metrics are monitored in real time to identify anomalous variations exhibiting early signs of congestion. In addition, an entropy-based filter is utilized to determine the diversity of distributional flows, assisting in identifying bursty or skewed traffic that aids in the development of congestion.

3.3 Congestion Estimation and Final Decision Module

$$C_s = \alpha . \left(\frac{D_{avg}}{D_{max}}\right) + \beta . \left(\frac{L}{L_{max}}\right) + \gamma . \left(1 - \frac{T_{curr}}{T_{max}}\right) \quad (6)$$

As seen in Equation 6, congestion score metric Cs computes network congestion based on average packet delay, packet loss rate, and current throughput. It is the weighted summation of normalized values representing the current state of the network concerning an optimal condition. The weights α, β, and γ can be adjusted to indicate the relevance of each parameter in a given operational setting. This equation guarantees that any slice of the network which suffers from higher delays or substantial packet loss while the throughput is low, results in higher congestion score facilitating prompt evaluation and estimation of congestion level.

3.4 Multi-Hop Topology Analysis and Delay Estimation

$$T = \sum_{i=1}^{n} (d_i + q_i + p_i) \quad (7)$$

In Equation 7, we define the total end-to-end delay T in a multi-hop communication path, which includes transmission delay d_i, queuing delay q_i, and processing delay p_i at each hop i. The equation illustrates the cumulative latency a data packet incurs while passing through n network nodes. This delay is an important input for estimating congestion as higher queuing or processing would imply an increase in traffic and possible congestion.

3.5 Dataset Description

The dataset used in this study consists of real-world NetFlow and IPFIX logs collected from backbone and access networks, including edge routers, SDN switches, and gateway devices. These flow records were captured using passive monitoring techniques, ensuring no additional traffic was injected into the network. Each record contains essential attributes such as source and destination IP addresses, source and destination ports, protocol type, flow start and end times, packet and byte counts, and flow duration. In addition to these, statistical features like average inter-arrival time, retransmission count, estimated packet loss rate, jitter, and throughput deviation were extracted. These metrics were used to characterize network behavior and identify signs of congestion. The collected flow data served as input for training and testing a hybrid machine learning model combining Decision Tree and Gradient Boosting algorithms. The goal was to enable real-time congestion detection with minimal overhead and high accuracy. The dataset's passive and scalable nature makes it highly suitable for modern networking environments like SDN, IoT, and 5G infrastructures, where low latency and real-time response are critical.

4 Results and Discussions

The designed framework for detecting congestion in real-time was evaluated within the context of a simulated multi-hop network (Song et al., 2021). The nodes in the network generated passive flow data which presented traffic behaviors through different topologies under changing conditions. Topology-dependent delays, flow-level metrics, and congestion assessment scores were tracked and adjusted throughout the experiments. Responding to the experiments performed, the system demonstrated a strong tendency to detect congestion indicators long in advance without necessitating additional traffic load.

Validation using two primary metrics demonstrated the effectiveness of the model: the e2e latency per number of hops and the calculated congestion score. Results confirm that total delay contributes non-linearly as the number of hops increases, thus proving the need for topology-aware measures. Moreover, decision confidence was shown to improve with subtle changes in flow by employing a pre-trained machine learning classifier.

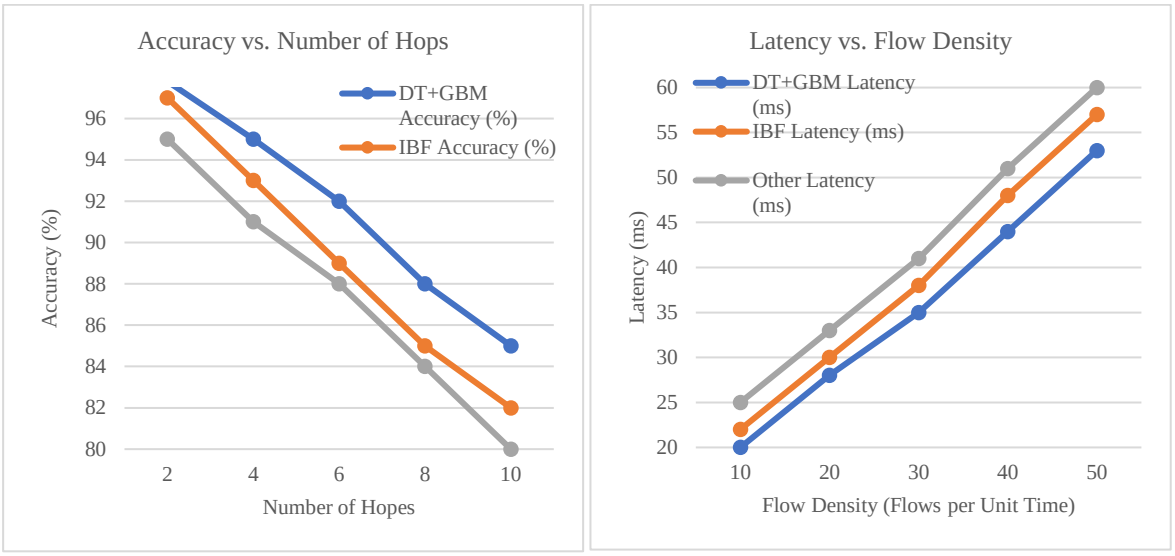

Figure 3: Accuracy vs. hop count and latency vs. flow density

Figure 3 analyzes the performance evaluation metrics of the congestion detection model as a function of network growth parameters. The left subfigure depicts the accuracy drop off as the number of hops increases. The proposed model starts off attaining an accuracy of 98% at 2 hops then gradually decreases to 85% at 10 hops. In comparison, both IBF and the other existing approach demonstrate more aggressive declines which showcases their inability to detect congestion accurately as the network topology becomes deeper and more complex (Ahmed et al., 2016). This illustrates the overall congestion detection approach outperforming the system under multi-hop routing environments.

Impact of flow density on latency is shown on the right subfigure. With an increase of 10 to 50 flows per unit time, latency is observed to rise for all techniques. Nonetheless, the proposed approach continues to show a lower latency as compared to IBF and the other approach. For example, at 50 flows, the proposed model has 53 ms latency whereas IBF and the other competing approach show 57 ms and 60 ms respectively. This is due to the model's embedded efficient statistical flow pressure management and topology-aware analyzed flow routing which minimized processing delays.

The aforementioned findings confirm that the model in question maintains a higher accuracy in deeper topologies and demonstrates improved latency efficiency under load, hence its suitability for real-time congestion-aware routing systems in intelligent, scalable networks (Abdulhafedh, 2022)

Table 1: Performance comparison of DT+GBM hybrid passive detection model with existing methods

Metric	DT+GBM	IBF	Other Approach
Accuracy (%)	97.8	95.3	93.9
DRW (%)	94.2	91.7	90.1
Latency (ms)	38.5	44.6	49.8
Detection Time (ms)	12.3	17.5	19.1
False Positive Rate (%)	1.2	2.3	2.8

A comparative evaluation of the performance metrics of the proposed DT+GBM Hybrid Passive Detection Model, alongside other methods, IBF and Other Approach as a baseline, is listed in Table 1. The evaluation is based on several critical performance metrics, such as: accuracy, workload delivery rate (DRW), latency, time to detect, and false positive rate.

The new model attains the maximum accuracy of 97.8% as well as DRW of 94.2%, thus demonstrating the strongest competency in recognizing congestion events and maintaining traffic delivery. It also exhibits lowest latency (38.5 ms) and fastest detection time (12.3 ms) which are essential for making time-sensitive decisions within changing network ecosystems. The model also outperformed all others with a minimized false positive rate of 1.2%.

The results justify the effectiveness of the ensemble learning approach based on decision trees and gradient boosting, which captures temporal and structural anomalies in flow-level data. The model's DT+GBM performance confirming his versatility thus supports his use for SDN-integrated edge-based systems requiring real-time congestion detection.

1. Accuracy (%)

Accuracy of the method is evaluated as in (8) based on the measures of accurate classification which includes true and false positives. This metric additionally provides support on how accurate the detection of congestion decisions are made.

$$Accuracy = \left(\frac{TP + TN}{TP + TN + FP + FN}\right) \times 100 \quad (8)$$

Where:

- TP: True Positives (congestion correctly detected)
- TN: True Negatives (normal traffic correctly detected)
- FP: False Positives (normal traffic falsely marked as congested)
- FN: False Negatives (congestion missed)

2. DRW (%) – Delivery Rate of Workload

The DRW is computed using Equation (9) by dividing the number of successfully delivered packets by the total packets sent. This metric quantifies the network's delivery efficiency under varying congestion levels.

$$DRW = \left(\frac{Total\ Delivered\ Packets}{Total\ Sent\ Packets}\right) \times 100 \quad (9)$$

3. Latency (ms)

The average latency is estimated using Equation (10), which computes the mean delay between packet transmission and reception times. This measure is crucial for evaluating responsiveness in real-time traffic conditions.

$$Latency = \frac{1}{n}\sum_{i=1}^{n}(t_{arrival}^{(i)} - t_{sent}^{(i)}) \quad (10)$$

Where:

- $t_{sent}^{(i)}$: Timestamp when packet i was sent
- $t_{arrival}^{(i)}$: Timestamp when packet iii was received
- n: Total number of packets measured

4. Detection Time (ms)

The detection time is derived using Equation (11) by calculating the difference between the moment congestion begins and the time it is detected by the system. This parameter reflects how quickly the model responds to emerging congestion events.

$$Detection\ Time = t_{decision} - t_{congestion_{onset}} \quad (11)$$

Where:

- $t_{congestion\_onset}$: Time when congestion started (ground truth or label)
- $t_{decision} -$: Time when the system triggered the congestion alert

5 Conclusion

The paper proposed a scalable and low-latency framework for real-time congestion detection that uses passive flow analysis and a hybrid ensemble machine learning approach. The system analyzes NetFlow/IPFIX records from edge devices to compute statistical and temporal features which are processed by a hybrid ensemble of Decision Trees and Gradient Boosting. To increase the detection accuracy in dynamic, multi-hop networks, the architecture includes topology-aware delay estimation and entropy-based evaluation of traffic diversity. The experimental results confirmed the model's effectiveness across diverse traffic patterns and network topologies. Enhanced accuracy and reduced detection time and false positives relative to models based on Integrated Bloom Filter (IBF) and other existing frameworks were noted. This adaptability is further augmented by introducing multi-layer decision engines and congestion score metrics tailored for SDN and IoT networks.

The model's ability to eliminate active probing and enable real-time localized decisions not only reduces bandwidth expenditure but ensures preemptive responses to congestion. Its application is especially valuable in decentralized infrastructures such as smart cities, 5G backhaul networks, and industrial IoT systems where low latency is crucial. Looking ahead, we will examine the incorporation of federated learning for distributed training, live deployment within programmable switch environments, context-aware dynamic threshold adaptation, and context-aware dynamic threshold adjustment.

References

[1] Abdulhafedh, F. T. (2022). Intrusion detection system in software defined IoT network using deep learning and genetic algorithm.

[2] Ahmed, M., Mahmood, A. N., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, *60*, 19-31. https://doi.org/10.1016/j.jnca.2015.11.016

[3] Alash, Y., Altuma, R., & Al-Jobouri, L. (2025). Implementing Packet-Level Internet Traffic Classification Using XGBoost in Software-Defined Networking. *International Journal of Intelligent Engineering & Systems*, *18*(9). https://doi.org/10.22266/ijies2025.1031.18

[4] Attioui, M., & Lahby, M. (2025). Congestion Forecasting Using Machine Learning Techniques: A Systematic Review. *Future Transportation*, *5*(3), 76. https://doi.org/10.3390/futuretransp5030076

[5] Benacer, I., Boyer, F. R., & Savaria, Y. (2018). A high-speed, scalable, and programmable traffic manager architecture for flow-based networking. *IEEE Access*, *7*, 2231-2243. https://doi.org/10.1109/access.2018.2886230.

[6] Buragohain, C., & Medhi, N. (2016, February). FlowTrApp: An SDN based architecture for DDoS attack detection and mitigation in data centers. In *2016 3rd International Conference on Signal Processing and Integrated Networks (SPIN)* (pp. 519-524). IEEE. https://doi.org/10.1109/SPIN.2016.7566750

[7] ChithraDevi, S. A., Mahendravarman, I., Ragavendiran, A., Selvam, M. M., Yamuna, K., & Vibin, R. (2024, July). Optimal configuration of radial distribution networks with stud krill herd optimization. In *2024 International Conference on Signal Processing, Computation, Electronics, Power and Telecommunication (IConSCEPT)* (pp. 1-6). IEEE. https://doi.org/10.1109/IConSCEPT61884.2024.10627797

[8] Elhoseny, M., Siraj, M., Haseeb, K., Nawaz, M., Altamimi, M., & Alghamdi, M. I. (2022). Energy-efficient mobile agent protocol for secure iot sustainable applications. *Sustainability*, *14*(14), 8960. https://doi.org/10.3390/su14148960

[9] Etman, A. M., Abdalzaher, M. S., Emran, A., Yahya, A., & Shaaban, M. (2024). A survey on machine learning techniques in smart grids based on wireless sensor networks. *IEEE Access*. https://doi.org/10.1109/ACCESS.2024.3524097

[10] Gajmal, Y. M., & Udayakumar, R. (2021). Blockchain-based access control and data sharing mechanism in cloud decentralized storage system. *Journal of web engineering*, *20*(5), 1359-1388. https://doi.org/10.13052/jwe1540-9589.2054

[11] Giotis, K., Argyropoulos, C., Androulidakis, G., Kalogeras, D., & Maglaris, V. (2014). Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments. *Computer networks*, *62*, 122-136. https://doi.org/10.1016/j.bjp.2013.10.014

[12] Hu, X., Liu, T., Hao, X., & Lin, C. (2022). Attention-based Conv-LSTM and Bi-LSTM networks for large-scale traffic speed prediction. *The Journal of Supercomputing*, *78*(10), 12686-12709.

[13] Huang, H., Zhu, X., Bi, J., Cao, W., & Zhang, X. (2021). Machine learning for broad-sensed internet congestion control and avoidance: A comprehensive survey. *IEEE Access*, *9*, 31525-31545. https://doi.org/10.1109/ACCESS.2021.3060287

[14] Janmohammadi, P., & Babazade, M. (2015). Resource Management in the Cloud Computing Using a Method Based on Ant Colony Optimization. *International Academic Journal of Science and Engineering*, *2*(1), 186-200.

[15] Jiao, L., Zhao, J., Wang, C., Liu, X., Liu, F., Li, L., ... & Yang, S. (2024). Nature-inspired intelligent computing: A comprehensive survey. *Research*, *7*, 0442. https://doi.org/10.34133/research.0442

[16] Karakus, M., & Durresi, A. (2017). A survey: Control plane scalability issues and approaches in software-defined networking (SDN). *Computer Networks*, *112*, 279-293. https://doi.org/10.1016/j.comnet.2016.11.017

[17] Leema, A. A., Balakrishnan, P., & Jothiaruna, N. (2024). Harnessing the power of web scraping and machine learning to uncover customer empathy from online reviews. *Indian Journal of Information Sources and Services*, *14*(3), 52-63. https://doi.org/10.51983/ijiss-2024.14.3.08

[18] Mahboob, T., Holik, F., Li, M., Shah, A. A., & Pezaros, D. Survey on Smart Grids Cybersecurity-A Networking Perspective. https://dx.doi.org/10.2139/ssrn.5736364

[19] Marangunic, C., Cid, F., Rivera, A., & Uribe, J. (2022). Machine Learning Dependent Arithmetic Module Realization for High-Speed Computing. *Journal of VLSI circuits and systems*, *4*(01), 42-51. https://doi.org/10.31838/jvcs/04.01.07

[20] Mo, L., Lv, G., & Wang, B. (2022). A Fine-Grained network congestion detection based on flow watermarking. *Applied Sciences, 12*(16), 8094. https://doi.org/10.3390/app12168094

[21] Nanda, S., & Verma, P. (2021). Comparative study of machine learning techniques for detecting congestion in real-time networks. *International Journal of Communication Systems, 34*(4), e4692.

[22] Nguyen, T. T., & Armitage, G. (2009). A survey of techniques for internet traffic classification using machine learning. *IEEE communications surveys & tutorials*, *10*(4), 56-76. https://doi.org/10.1109/SURV.2008.080406.

[23] Park, Y., Kim, H., & Lee, J. (2020). Scalable SDN-based congestion detection and mitigation in IoT networks. *Sensors, 20*(18), 5103.

[24] Prasad, L. C., Kamatham, Y., & Sunehra, D. (2024). Congestion-aware Weighted Sum Routing for Enhanced Network Lifetime in Cognitive Radio Sensor Networks. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, *15*(4), 325-334. https://doi.org/10.58346/JOWUA.2024.I4.021

[25] Saranya, U., & Madhubala, P. (2019). Enhancement of security and network lifetime using flexi-cast method. *International Journal of Communication and Computer Technologies, 7*(1), 23-26.

[26] Singaravel, G., Abhisek, P., Dharshan, V., & Hariharan, M. (2020). One's Space in Online Social Network. *International Journal of Advances in Engineering and Emerging Technology*, *11*(2), 115-120.

[27] Song, H., Yu, J., & Lin, Y. (2021). A real-time and scalable network traffic monitoring framework using NetFlow and big data analytics. *Future Generation Computer Systems, 118,* 137–148.

[28] Sun, H., Liu, Y., Al-Tahmeesschi, A., Nag, A., Soleimanpour-Moghadam, M., Canberk, B., ... & Ahmadi, H. (2025). Advancing 6G: Survey for explainable AI on communications and network slicing. *IEEE Open Journal of the Communications Society*. https://doi.org/10.1109/OJCOMS.2025.3534626

[29] Tiwari, S., Rosak-Szyrocka, J., & Żywiołek, J. (2022). Internet of things as a sustainable energy management solution at tourism destinations in India. *Energies*, *15*(7), 2433. https://doi.org/10.3390/en15072433

[30] Verma, A., & Mehta, N. (2024). The Role of IoT in Modern Electrical Systems: An Interdisciplinary Approach in the Periodic Series. *Smart Grid Integration*, 13-18.

[31] Watrianthos, R. (2020). A Vivaldi Antenna For 5G Communications. *National Journal of Antennas and Propagation*, *2*(2), 33-38. https://doi.org/10.31838/NJAP/02.02.06

[32] Yang, J., Yang, F., Zhang, L., Li, R., Jiang, S., Wang, G., ... & Zeng, Z. (2021). Bridge health anomaly detection using deep support vector data description. *Neurocomputing*, *444*, 170-178. https://doi.org/10.1016/j.neucom.2020.08.087

[33] Zaitoon, R., Mohanty, S. N., Godavarthi, D., & Ramesh, J. V. N. (2024). SPBTGNS: design of an efficient model for survival prediction in brain tumour patients using generative adversarial network with neural architectural search operations. *IEEE Access*. https://doi.org/10.1109/ACCESS.2024.3430074

Authors Biography

Savinder Kaur is associated with the Centre of Research Impact and Outcome, Chitkara University, Rajpura, Punjab, India. Her research focuses on research impact assessment, academic performance metrics, innovation analysis, and higher education research evaluation. She actively contributes to projects aimed at enhancing research visibility, quality, and measurable outcomes across academic disciplines

Dr. Swadhin Kumar Barisal is an Associate Professor in the Department of Computer Applications at Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar, Odisha, India. His research interests include artificial intelligence, machine learning, data mining, and software engineering. He has published numerous research papers in reputed international journals and conferences and is actively engaged in mentoring students and promoting innovation in computer science and application-oriented technologies

Veera Nagaiah Maddikayala is an Associate Professor in the Department of Computer Applications (DCA) at Presidency College, Bengaluru, Karnataka, India. His research interests include artificial intelligence, data analytics, cloud computing, and software engineering. He has contributed to several national and international publications and is actively involved in academic research, curriculum development, and guiding students in emerging areas of computer science.

M.P. Sunil is an Assistant Professor in the Department of Electronics and Communication Engineering, Faculty of Engineering and Technology, JAIN (Deemed-to-be University), Karnataka, India. His research interests include embedded systems, signal processing, communication technologies, and IoT-based system design. He has contributed to various research publications and is actively involved in guiding students and advancing innovative solutions in the field of electronics and communication engineering.

Dr. Srinivasulu Gogula is a Professor in the Department of Information Technology at Vardhaman College of Engineering, Hyderabad, Telangana, India. His research interests include computer networks, data analytics, artificial intelligence, and cloud computing. With extensive academic and research experience, he has published several papers in reputed journals and conferences and actively contributes to advancing knowledge in the field of information technology.

Rishabh Vashisht is affiliated with the School of Engineering & Computing, Dev Bhoomi Uttarakhand University, Dehradun, India. His academic and research interests include computer science, artificial intelligence, data analytics, and emerging technologies. He is committed to fostering innovation-driven education and contributing to research that bridges the gap between theoretical learning and practical application. Rishabh Vashisht actively participates in academic initiatives aimed at promoting technological advancement and skill development among students.