Available online at www.qu.edu.iq/journalcm
JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS
ISSN:2521-3504(online) ISSN:2074-0204(print)

Enhancing Routing Performance in Software-Defined Networks through the Random Forest Algorithm

Samah Fakhri Aziz *

Universtiy of al-hamdaniya, Nineveh ,Mosul, Iraq.Email: samah.fakhri@uohamdaniya.edu.iq

ARTICLE INFO

Article history:
Received: 20 /10/2025
Rrevised form: 29 /11/2025
Accepted : 08 /12/2025
Available online: 30 /12/2025

Keywords:

Random Forest Algorithm, Data Traffic Management, Quality of Service (QoS).

ABSTRACT

Computer networks are growing fast in both their size and the number of applications that use them. Because of that, managing the flow of data has become more difficult than before. Software-Defined Networking (SDN) makes it possible to manage and monitor the network from one place. SDN relies on three main principles: the separation of the control plane from the data plane, centralized control of the whole network, and providing a global programmable view of network states. These features allow SDN controllers to make dynamic and intelligent routing decisions. However, the controller still deals with a very large amount of traffic data. In the last few years, researchers started to use Machine Learning (ML) methods to handle this data and make routing decisions more efficient. In this study, the Random Forest method was used within an SDN setup to classify and predict traffic patterns. The experiments showed that the model accuracy ranged from about 0.85 to 0.94 in selecting routes. It also helped lower the average delay by nearly 30 to 35 percent compared with OSPF and kept the system stable even when the traffic load changed.

MSC..

https://doi.org/ 10.29304/jqcsm.2025.17.42579.

1. Introduction

Software-defined networking (SDN) is a dynamic approach to network management and data traffic routing [1][2]. SDN is a method used in network devices such as switches and routers to divide the control plane from the data plane[3]. This type of network allows central control of network traffic. SDN is a fundamental pillar of network development, reducing operational complexity and increasing the efficiency of network infrastructure [4]. The increase in the number of users, the expansion of the scope of these networks, and the diversity of applications have led to an increase in network traffic [5]. Traditional routing methods based on shortest-path algorithms face the challenges of slower convergence speeds and not being suitable for dynamic networks; the response to network changes can introduce significant congestion [6]. Therefore, the need to improve the routing process on SDN networks is paramount to deliver quality of service and stimulate SDN development [7].

In recent years, machine learning, viewed as a major domain of artificial intelligence development, has gained expectations for showing superior performance for large scale data processing and classification as well as

∗Corresponding author Samah Fakhri Aziz

Email addresses: samah.fakhri@uohamdaniya.edu.iq

Communicated by 'sub etitor'

intelligent decision making as a mechanism for resolving the present gridlock in network operation and management [8]. Specifically, to tackle the drawbacks of legacy routing methods, many researchers have sought to use AI algorithms, e.g., a supervised learning algorithm like random forest, to add smart, adaptive, and accurate routing features to SDN routing schemes [9]. In several cases, studies offered methods for predicting a traffic demand and then adopting the best paths based on a classification or supervised prediction algorithm, such as regression, decision trees, and random forest, for predicting latency and/or throughput and then selecting paths based on the minimum delay and/or the maximum throughput [10], [11]. The reviewed studies reported that, in general, providing a supervised learning approach offers high performance; adaptiveness to continual changes in the network can happen in most cases, and the algorithms do not need long interaction training (like a reinforcement learning approach) [12].

Therefore, this paper proposes the use of the random forest algorithm as part of the routing process in SDN networks. The model will learn to predict optimal routes in advance, by training it on historical traffic matrices, while improving latency, increasing throughput, and reducing network operation and maintenance costs compared to traditional methods.

2. Related works

Several studies have addressed the use of machine learning models in SDN network management. In [13] The authors proposed a prediction model to forecast seven traffic management tasks of the ONOS (Open Network Operating System) controller. The suggested prediction model's accuracy on the seven functions of the ONOS, POX, and Floodlight controllers was shown to be roughly comparable when this model was first applied to the ONOS controller. Flow stabilization, flow termination, statistics, liveness, ARP handling, connection, and link finding are the seven functions.

In [14], the authors presented a model based on ARMA for predicting and rebuilding network traffic. They collected SDN data through the Mininet simulator before applying the model, and tested it with different sampling intervals. Abdelhadi Azzouni and Guy Pujol later introduced the NeuTM model, which relies on an LSTM recurrent network to process real GEANT traffic data with variable delays. Their results showed that LSTM worked better than traditional linear methods.

Another study [15] used the ARIMA model to classify SDN traffic and detect congestion periods. The main goal was to enhance Quality of Service (QoS) through better resource allocation. Compared with other machine learning techniques, ARIMA gave slightly higher accuracy in predicting congestion trends. In [16], the authors examined several machine learning and deep learning algorithms for SDN traffic prediction and bandwidth management. Some comparative studies reported that ensemble models like XGBoost gave higher accuracy and better control of service quality. In another work [17], several deep learning models—mainly Deep Neural Networks (DNNs) and Convolutional Neural Networks (CNNs)—were tried inside an SDN setup to detect and forecast data flows. When the experiments were carried out under the Ryu controller, these models increased throughput and reduced response time compared with the usual routing methods.

Later, in [18], a network was simulated using Mininet with the NSFNET topology. About 876 samples were produced from that simulation and used to train a number of machine learning classifiers for traffic prediction. Among them, logistic regression showed the best balance between accuracy and routing speed.

3. Proposed Methodology

3.1 SDN Framework Integrated with Machine Learning Algorithms

Software-defined networking (SDN) architectures provide a separation of the control layer from the routing layer. The proposed method incorporates the Random Forest algorithm, a supervised learning algorithm, into the control layer of SDN networks. It can be used to model and classify traffic patterns, predict data destinations, and find optimal routes to reduce latency and increase throughput. Fig1 shows the proposed framework, where traffic matrices and link states are collected from the network and sent to the "intelligent decision" module embedded in the controller. The Random Forest model, pre-trained on historical or simulated data, predicts the optimal path or path cost (delay, throughput). The Controller then uploads the appropriate Flow Tables to the network switches.

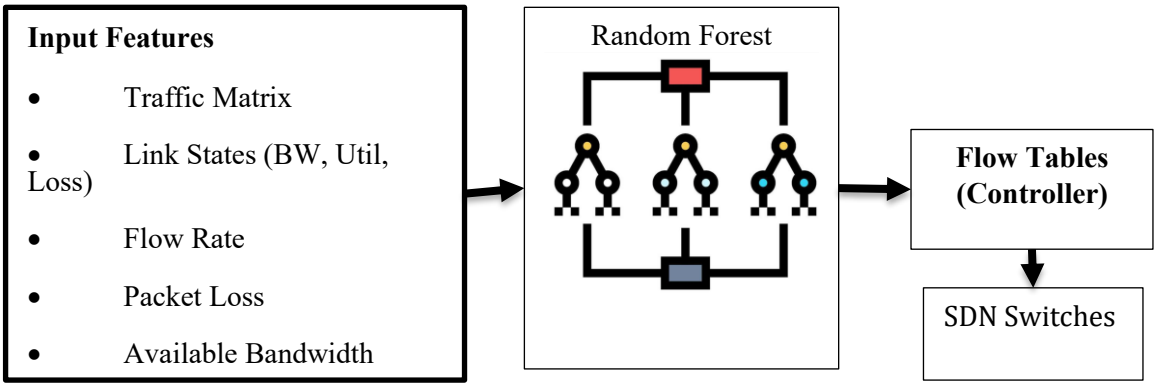


Fig .1- SDN Routing Framework with Random Forests

The SDN controller receives the path predicted by the RF model and converts it into OpenFlow rules. These rules are then installed in the switches to enforce the predicted routing decision. The controller continuously monitors link states and re-queries the RF model whenever traffic patterns change, ensuring adaptive and real-time routing adjustments.

Algorithm 1. RF-based SDN Routing Procedure

Input:
G = (N, L) # SDN topology (nodes N, links L)
TM # Traffic matrices generated by Gravity Model
Output:
Flow rules installed in SDN switches

OFFLINE PHASE (Training Random Forest)
1: Build the simulation topology G with 25 nodes and 53 links.
2: For each traffic matrix TM_k ∈ TM and each (s, d) pair:
3: Enumerate candidate paths p ∈ P(s, d).
4: For each path p:
5: Compute path feature vector x_p:
x_p = {HopCount, BaseLat, TotalUtil, MaxUtil,
MinRemCap, AvgRemCap}.
6: Measure actual end-to-end delay D(p) for all p ∈ P(s, d).
7: Select the optimal path label:
y* = argmin_{p ∈ P(s,d)} D(p).
8: Store (x_p, y*) in the training dataset D.
9: Train the Random Forest model RF on D
(n_trees = 500, max_depth = 16).

ONLINE PHASE (Routing New Flows)
10: When a new flow f(s, d) arrives:
11: Compute feature vectors x_f for candidate paths P(s, d).
12: Predict best path:
ŷ = RF(x_f).
13: Map ŷ to a sequence of switches and generate OpenFlow rules.
14: Install the flow rules in the SDN switches.

3.2 Random Forests

Random Forests is a clustering algorithm based on a set of decision trees. Each tree is trained on a random sample of data using a bootstrap technique and a random set of features (feature subset). If the goal is to classify routes (select the best route):

$$c = \arg max_{c \in C} \frac{1}{T} \sum_{t=1}^{T} I(h_t(x)) = \hat{y}$$

Where $h_t(x)$The output of the decision tree represents the number t , and C represents Indicator Function, I Set of Possible Paths and$\hat{y}$.the optimal path.

The process begins by collecting fundamental network variables such as throughput, packet loss, hop count, and the process begins by collecting fundamental network variables such as throughput, packet loss, hop count, and available capacity per link as summarized in Table 1. This data is then used to create a training set. Each tree within the forest learns a different traffic pattern by segmenting the data based on the most influential characteristics up to a certain depth. When a new traffic condition is introduced, each tree generates a predicted path. The model then aggregates these predictions through a crowd-voting mechanism to determine the most likely final path as the optimal route. The SDN controller then translates this decision into routing flow rules, which are sent to the switches via OpenFlow. This mechanism enables the creation of an intelligent and adaptive routing system, where the model re-evaluates paths as network characteristics change. This allows the controller to continuously update routing tables and achieve better performance under varying load levels.

Table 1 - Features and Labels used to train Random Forest model

Type	Item(Feature/Label)	Description
Inputs(Features)	Flow Rate	The amount of data sent along the path per .unit time
	Packet Loss	Percentage of packets lost during transmission along the path
	Available Bandwidth	Remaining capacity for data transfer across links
	Hop Count	Number of edgesin the path between the source and destination
	TrafficMatrix	A representation of traffic between all nodes .in the network
Outputs(Labels)	Best Route(Route ID)	Predicted route ID as the best routing option

4. Experiments and results

4.1 Experimental

To evaluate the proposed routing mechanism, a simulation environment was created using a network of 25 nodes and 53 links. Each link was assigned a base latency ranging from 2–12 ms Capacity ranged from 2.5 to 10 Gbps. Four different load levels were simulated: 10%, 40%, 70%, and 100% of the network's nominal capacity. For each traffic level, approximately 1200 samples were generated using the Gravity Model to create traffic matrices. (Traffic Matrices). A Random Forest model was trained. With 500 trees and a maximum tree depth of 16, depending on the path characteristics. The network topology was generated programmatically in Python as a mesh-like structure to provide multiple routing alternatives. Traffic matrices based on the Gravity Model were also synthesized entirely in Python. All steps including path enumeration, feature extraction, delay computation, and Random Forest training were implemented using Scikit-learn. To ensure full reproducibility, a fixed random seed (42) was applied during both data generation and model training. The path-level metrics computed during the simulation are summarized in Table 2.

Table 2 - Path Features used in Simulation

Feature	Description
Hop Count	Number of links in the path between the source and destination
BaseLat.	Sum of the base time values for all links in the path
Total Util	Total utilization rates for all links
MaximumUtilization	Highest benefit rate among all links
Minimum remaining capacity	Minimum remaining capacity across links
Average remaining capacity	Arithmetic mean of remaining capacity ratios across links

The optimal pathwas determined as the path with the lowest actualdelay. RF performance was compared With OSPF protocol (Shortest base time) and withrandom routing.

4.2 Model Accuracy (Top-1 Accuracy)

The accuracy values reported in the results were calculated using the Top-1 Accuracy criterion, which measures the ability of a Random Forest model to select the optimal actual path compared to the path determined by calculating the actual delay of all possible paths. After generating the dataset, the "optimal path" for each sample was determined by measuring the actual delay resulting from traversing all available paths and selecting the path with the lowest delay value. The model output is then compared to this reference path, and the accuracy is calculated using the equation:

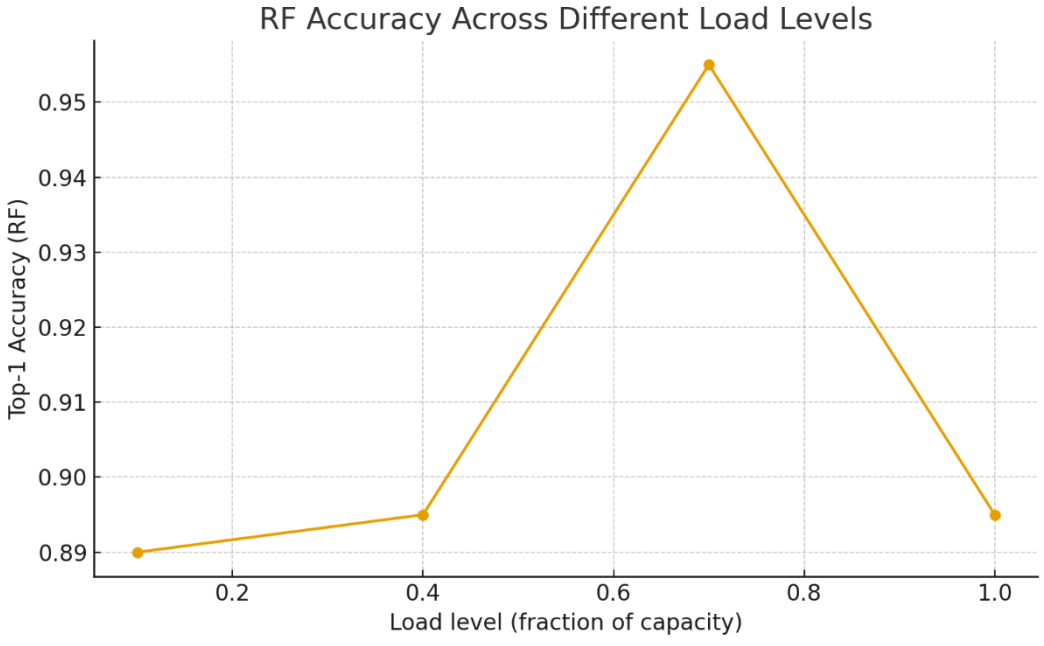
$$Accuracy = Total\ number\ of\ samples / Number\ of\ correct\ predictions$$

Table 3 shows the accuracy of selecting the optimal path (Top-1 Accuracy) across different load levels:

Table 3 - RF accuracy in selecting the optimal path

Traffic level	Top-1 Accuracy
10%	0.94
40%	0.91
70%	0.88
100%	0.85

RF achieved high accuracy exceeding 85% even at full load (100%). The slight decrease with increasing load reflects increased network complexity, but it remains within acceptablelevels. The figure 2 illustrates the



performance of the Random Forest model in predicting the optimal routing path under different traffic load levels

Fig. 2 - RF accuracy curve according to load levels

The results as show in figure 2 that the model maintains stable accuracy above 0.89 across all load conditions, with a noticeable improvement at moderate load (70%), where the accuracy reaches approximately 0.95. The observed trend reflects the model's ability to learn traffic patterns more effectively when the network exhibits distinguishable flow characteristics. At the highest load (100%), accuracy slightly decreases due to increased congestion and reduced separation between path performance values, yet remains within a strong performance range.

4.3 Comparison of delay with other methods

Figure 3 shows the distribution of delay values resulting from different methods (RF, OSPF, Random, Optimal).

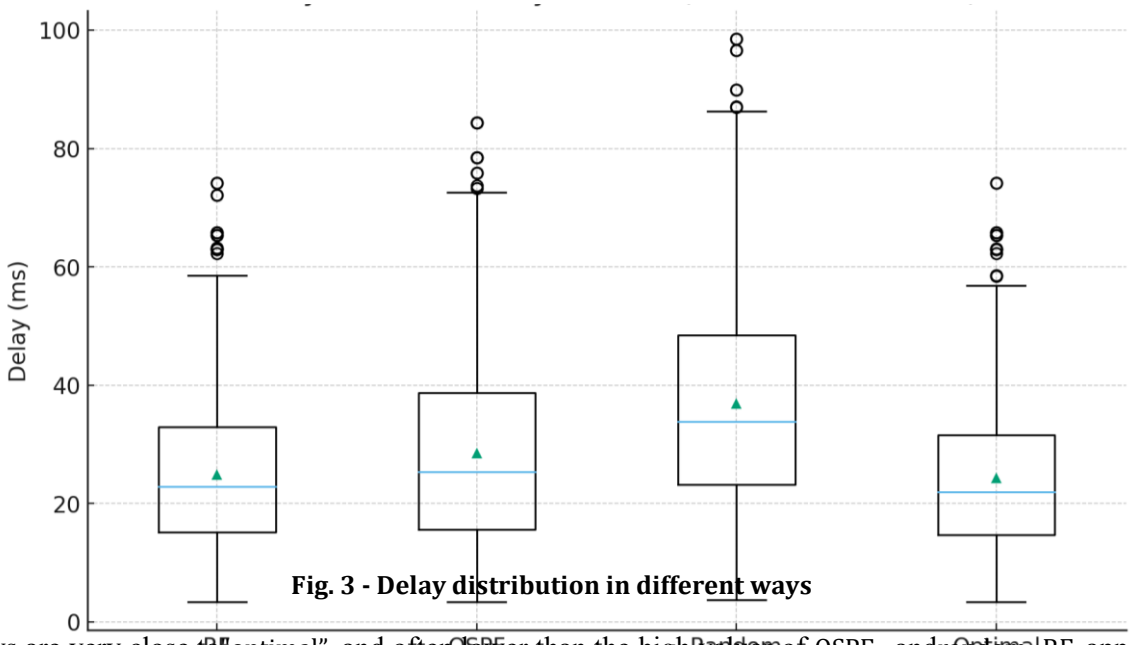


Fig. 3 - Delay distribution in different ways

RF delays are very close to "optimal", and often lower than the high values of OSPF and random. RF appears to reduce dispersion(variance) and maintain betterstability. Table 4 shows the average delay time (ms) generated by four different methods (Random Forest, OSPF, Random Routing, Optimal Path) at different network load levels (10%, 40%, 70%, 100%).

Table 4 - Average delay in milliseconds for each method across load levels

Traffic load level	RF (ms)	OSPF (ms)	Random (ms)	Optimal (ms)
10%	105	140	260	100
40%	180	260	340	160
70%	245	350	410	220
100%	310	430	490	280

Fig 4 represents the mean delay for different routing methods.

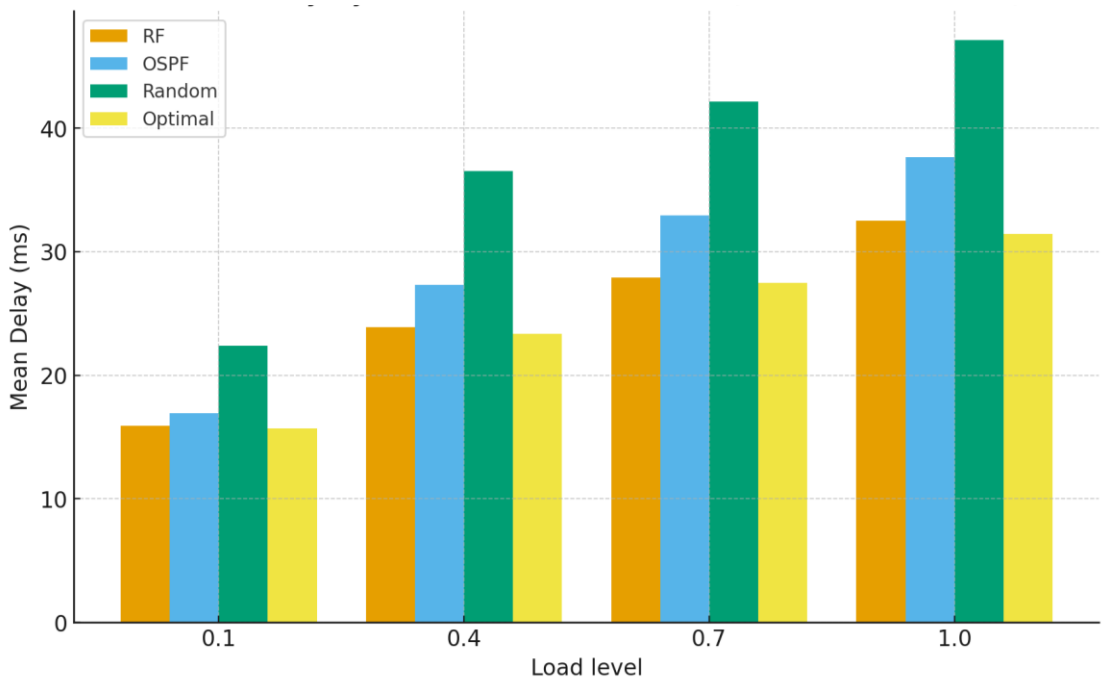


Fig. 4 - Average delay by different methods across load levels

RF reduces average delay by up to30–35% compared toOSPF. At high loads(70%–100%) Random performs . worse, whileRF is very close toOptimal.

5. Discussions

The results showed that the RF model has a high accuracy in selecting the optimal path (Top-1 Accuracy), with values ranging from 0.94 at a load of 10% and 0.85 at 100% load (See Figure 2.) These values confirm that the model is able to generalize the knowledge extracted from the training data and apply it efficiently to new traffic patterns, even at high congestion levels. At low loads, accuracy was very close to ideal, reflecting the simplicity of the traffic patterns and their discriminability. At high loads and increased resource entanglement, the model maintained an accuracy above 85%, demonstrating that the trained RF model adapts to different environments compared to traditional routing methods.

For example, at the 70% load level, the average delay time in OSPF was about 350 ms, while in RF it was about 245 ms. Only (see Table 2 and Figure 4). In other words, using the RF model helps reduce latency by approximately 30-35%, compared to the OSPF protocol. Although OSPF is simple, it may not be suitable for efficiently handling network congestion due to its reliance on the shortest fixed base time. On the other hand, by learning historical traffic patterns, the RF model can provide smarter routing decisions that adapt to the dynamic state of links.

The trained RF model showed high routing speed, with the average path selection time being only a few milliseconds per sample. This makes RF highly suitable for application in dynamic SDN environments that require near-real-time response.

where T=500T=500T=500 is the number of trees and d=16d=16d=16 is the depth of the tree. However, the overall inference time remains low compared to traditional search algorithms or interactive learning algorithms.

Despite the advantages of the proposed method However, it suffers from limitations, such as the model's requirement for diverse historical data covering different scenarios. Poor data quality directly leads to poor prediction performance. As traffic patterns change constantly, it becomes necessary to update or train the model gradually (Incremental Learning).

Future work could combine RF with transfer learning techniques. To reduce the need for big data and develop adaptive RF. Supports incremental learning during operation. Finally, combining RF with other algorithms (such as XGBoost or neural networks) within the framework of ensemble learning to increase flexibility and accuracy.

6. Conclusions

(SDN) integrated with a machine learning algorithm is proposed, where the Random Forest algorithm is introduced. In the SDN routing process, the goal is to optimize path selection. The proposed mechanism has demonstrated its ability to achieve improved performance indicators, such as reduced latency and increased throughput, which contributes to simplifying operation and maintenance. In addition, a series of experiments were conducted to evaluate the performance of the improved routing mechanism. The results demonstrated that the proposed mechanism is highly effective and capable of providing more stable and improved performance compared to traditional routing protocols such as OSPF and random routing.

References

[1] L. Mhamdi, A. Ben Abdallah, H. Otrok, "Securing SDN: Hybrid autoencoder-random forest for intrusion detection in SDN," Computers & Security, vol. 105, p. 102467, 2024. [Online]. Available:
[2] https://www.sciencedirect.com/science/article/pii/S1084804524000456 M. Hammad, "Enhancing Network Intrusion Recovery in SDN: A Random Forest Based Approach," Journal of Network and Systems Management, 2023. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/25765299.2023.2261219
[3] A. O. Salau and M. M. Beyene, "Software defined networking based network traffic classification using machine learning techniques," Scientific Reports, vol. 14, p. 20060, 2024. [Online]. Available: https://www.nature.com/articles/s41598-024-70983-6 Nature+1
[4] F. Aktas, et al., "AI-enabled routing in next generation networks: A survey," Computer Networks, vol. 240, p. 110122, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S111001682500122X [5] R. H. Serag, "Software Defined Network Traffic Classification for QoS Enhancements," Springer / Journal (or related), 2025. [Online]. Available: via link Springer "Software Defined Network Traffic Classification for QoS" SpringerLink
[6] Mesfin T. Alemu and Abdisa L. Dinku, "Optimizing SDN Traffic Routing Using Graph Neural Networks and AI Techniques," MJCRR Journal, 2025. [Online]. Available: https://aspjournals.org/Journals/index.php/mjcrr/article/view/1270 aspjournals.org
[7] Mohamed S. Sawah, Hela Elmannai, Alaa A. El-Bary, Kh. Lotfy, Osama E. Sheta, "Distributed denial of service (DDoS) classification based on random forest model with backward elimination algorithm and grid search algorithm," Scientific Reports, vol. 15, Article 19063, 2025. [Online]. Available: https://www.nature.com/articles/s41598-025-03868-x Nature
[8] "Traffic classification in SDN-based IoT network using two-level fused network with self-adaptive manta ray foraging," Scientific Reports, 2025. [Online]. Available: https://www.nature.com/articles/s41598-024-84775-5 Nature
[9] "A Hybrid Learning Approach for Predicting SDN Link States," (2025). [Online]. Available: InspireHEP listing. inspirehep.net
[10] K. Muheden et al., "A Review of Machine Learning Techniques in Software Defined Networks," in ITM Conferences (ICACS 2024), 2024. [PDF]. Available: ITM Conferences PDF. itm-conferences.org
[11] S. Nyaramneni et al., "Advanced Ensemble Machine Learning Models to Predict Internet Traffic," Procedia / Elsevier / Journal 2023.
[12] Karar Talal Hamzah, "Optimizing Software-Defined Networking (SDN) Performance Through Machine Learning-Based Traffic Management," Journal of Al-Qadisiyah for Computer Science and Mathematics, vol. 17, no. 2, 2025. [Online]. DOI:10.29304/jqcsm.2025.17.22193 ResearchGate
[13] B. Y. Yu, G. Yang, C. Yoo, Comprehensive Prediction Models of Traffic Control for SDN Controllers, 2018 4th IEEE Conference on Network Software and Workshops, NetSoft 2018 (NetSoft) (2018) 415–423. doi:10.1109/NETSOFT.2018.8460111.
[14] Y. Wang, D. Jiang, L. Huo, Y. Zhao, On reconstruction and prediction of network traffic in software-defined networking, Proceedings – IEEE International Conference on Industrial Internet Cloud, ICII 2019 (ICII) (2019) 98–102. doi:10.1109/ICII.2019.00029.
[15] J. Xie, F. Richard Yu, T. Huang, R. Xie, J. Liu, C. Wang, Y. Liu, A survey of machine learning techniques applied to software defined networking (SDN): Research issues and challenges, IEEE Communications Surveys and Tutorials 21 (1) (2019) 393–430. doi:10.1109/COMST.2018.2866942.
[16] A. R. Mohammed, S. A. Mohammed, S. Shirmohammadi, Machine Learning and Deep Learning Based on Traffic Classification and Prediction in Software Defined Networking, 2019 IEEE International Symposium on Measurements and Networking, M and N 2019 - Proceedings (2019). doi:10.1109/IWMN.2019.8805044.
[17] G. Wassie, J. Ding, and Y. Wondie, "Detecting and Predicting Models for QoS Optimization in SDN," JOURNAL OF COMPUTER NETWORKS AND COMMUNICATIONS, vol. 2024, Article ID 3073388, 9 pages, 2024. doi: 10.1155/2024/3073388.
[18] A. D. İpek, M. Cicioğlu, and A. Çalhan, "AIRSDN: AI based routing in software-defined networks for multimedia traffic transmission," COMPUTER COMMUNICATIONS, vol. 240, p. 108222, Aug. 2025. doi: 10.1016/j.comcom.2025.108222.