Streaming via SDN: Resource forecasting for video streaming in a Software-Defined Network

Syed M. A. H. Bukhari
Department of Computer Engineering, Jeju National University,
Republic of Korea
ammar@office.jejunu.ac.kr

Muhammad Afaq
Department of Computer Engineering, Jeju National University,
Republic of Korea
afaq24@gmail.com

Wang-Cheol Song
Department of Computer Engineering, Jeju National University,
Republic of Korea
philo@jejunu.ac.kr

***Abstract*—With the advancement in network devices and the proliferation of new technologies such as Software-Defined Networking (SDN), managing a network becomes more difficult. In an SDN network, a single physical device acts as a firewall and load balancer at the same time. The management of those devices and the prevention of the resources being exhausted is a challenging task for the network administrator. In this direction, this paper presents an approach to predict resources on a switch in an SDN-based network. For this purpose, a video streaming scenario is deployed in an SDN network and performance metrics are captured. The resources are predicted using four machine learning algorithms. Specifically, the paper proposes a testbed implementation of a video streaming scenario to evaluate the performance of the proposed approach. The proposed approach can help network operators optimize network performance, ensure efficient use of resources, and enhance user experience.**

***Index Terms*—video traffic prediction, time series prediction, LSTM, XGBoost, SDN**

I. Introduction

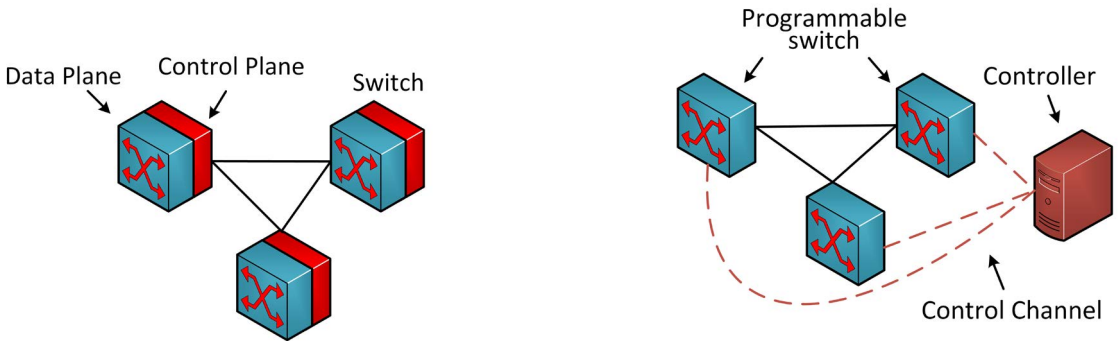
With the growing use of the internet, the volume of internet traffic has been increasing rapidly, which is placing a significant burden on the middleboxes and switches that support the functioning of the internet. The Cisco Annual Internet Report predicts that by the end of 2023, 66% of the world's population will have internet access [1]. This rise in internet traffic is putting a strain on these network devices, making it challenging to handle the volume of traffic. As a result, network operators have been upgrading their infrastructure and investing in more powerful middleboxes and switches to handle the increasing traffic [2], [3]. However, upgrading hardware alone may not always be sufficient to keep up with the demand. Network operators must also optimize their networks and implement more efficient traffic management techniques, such as traffic shaping, to ensure that their networks can handle the increasing traffic load [4].

Technological advancements such as Software-Defined Networking (SDN) have opened up new ways of managing network traffic, enabling network operators to dynamically allocate resources to meet the changing demands of internet traffic [5], [6]. SDN is a network architecture that separates the network control plane from the data plane, allowing network operators to manage their network infrastructure more efficiently. As a result, SDN has emerged as a potential solution for traffic management [7], [8]. With SDN, network operators can centrally manage their network infrastructure through a single controller that can dynamically allocate network resources and adapt to changes in traffic demand, as shown in Figure 1.

SDN can also be an effective solution for managing traffic congestion in networks. With SDN, network operators can create and implement traffic engineering policies [9], [10] to optimize the use of network resources and avoid congestion. These policies define how traffic is routed through the network, which paths to use, and how to prioritize traffic flows to prevent congestion. One key advantage of SDN for traffic congestion management is its ability to provide real-time traffic monitoring and analysis [11], [12]. SDN controllers can continuously monitor network traffic and detect congestion in real-time, allowing them to adjust traffic flows and avoid congestion before it occurs. Additionally, SDN's flexibility and programmability make it easier to implement traffic management policies that can adapt to changing traffic conditions. Network operators can use SDN to automatically adjust network policies and allocate resources based on traffic load or other parameters.

Network traffic analysis is the process of continuously monitoring and extracting traffic insights to understand network performance, security, and usability [13]–[15]. It is essential for optimizing network performance and preventing intrusions or malware infections. Traffic analysis can also help troubleshoot network issues, such as identifying faulty devices, diagnosing connectivity problems, and resolving performance issues. Most importantly, traffic analysis plays a critical role in capacity planning by predicting future workload on network devices using current network insights. This helps prevent resource exhaustion problems in the future.

Resource prediction is the process of using current traffic patterns and historical data to forecast future traffic on the network. Forecasting network traffic is essential for capacity planning, which involves determining the amount of network capacity required to meet current and future demand. Accurate resource forecasting is important because it enables network operators to plan for future capacity needs, optimize network performance, and ensure that the network can handle expected load. By forecasting resources, network operators can reduce



(a) Traditional network infrastructure

(b) Software-Defined Network infrastructure

Fig. 1: Difference between a traditional network and Software-Defined Network

the risk of network congestion, ensure consistent quality of service for end-users, and meet demand without over-provisioning resources.

Various methods are used for resource forecasting, including statistical analysis [16], time series analysis [17], and machine learning algorithms [18]. Statistical analysis involves examining historical resource data to identify patterns and trends, which can be used to make predictions about future. Time series analysis uses statistical models to analyze and forecast resources based on time-based patterns. Machine learning algorithms, such as neural networks and decision trees, can also be used for resource forecasting. These algorithms analyze large amounts of historical data and use it to make predictions about future load patterns.

This paper presents an approach to predicting CPU utilization on a network switch by using machine learning algorithms on time-series data in an SDN-based network. Specifically, the paper proposes a testbed implementation of a video streaming scenario to evaluate the performance of the proposed approach. The main contribution of this paper is to analyze which machine learning model can perform better in the SDN-based video streaming scenario, which can help network operators optimize network performance, ensure efficient use of network resources, and enhance user experience. The main contributions of this paper are as follows.

- Proposing an SDN-based approach for predicting video traffic load in a network.
- Implement a testbed to evaluate the proposed approach in a real-world scenario with video streaming traffic.
- Applying different machine learning algorithms to predict CPU utilization based on time-series data.
- Analyzing which machine learning algorithm can perform better in video traffic load prediction.

The rest of the paper is organized as follows, Section II presents some background studies in the direction of video traffic prediction, and Section III presents a brief detail about the video streaming architecture. Section IV presents information and details about the implementation of the testbed. Section V presents the result of prediction of video traffic load by leveraging the machine learning algorithms and Section VI concludes the paper.

II. Related Work

Resource prediction aims to anticipate network congestion by analyzing historical and also real-time traffic data. In addition to traffic classification, traffic prediction is important in managing traffic flow to prevent congestion. Accurate prediction of network congestion is essential for providing high-quality network communication. By analyzing traffic data, the SDN controller can direct traffic flows to less congested links, ensuring optimal network performance.

Predicting traffic in advance is critical for providing high-quality communication across the network. By predicting possible congestion, solutions can be offered to prevent drops in Quality of Service (QoS) and Quality of Experience (QoE). Predicting the occurrence of an elephant flow at unusual times, which can be identified as a flow-based intrusion, can provide a more secure network. Additionally, predicting such elephant flows can eliminate the risk of overburdening the SDN controller [19]. Traffic prediction can also help to determine possible congestion on links before they impact QoS and QoE, leading to the routing of traffic to less congested links.

Tang et al. [20] presented a novel deep learning algorithm to predict future traffic load and congestion in the network. The training process involved a combination of a basic Deep Learning (DL) architecture and Deep Convolutional Neural Network (D-CNN). The DL-based prediction algorithm was then integrated with a DL-based channel assignment algorithm to enable intelligent traffic routing.

The study presented in [21] concentrated on utilizing the Bayesian machine learning algorithm to allow SDN switches to identify the controller responsible for generating the flow rules and predict any unmatched packets in the flow table.

In their work, Jain et al. [22] adopted a systematic approach to improve QoS in SDN. The approach involved performing big data analytic to uncover data correlations, identifying the root causes of any detected issues, and ultimately leveraging the insights gained to make predictions and analyze future network trends.

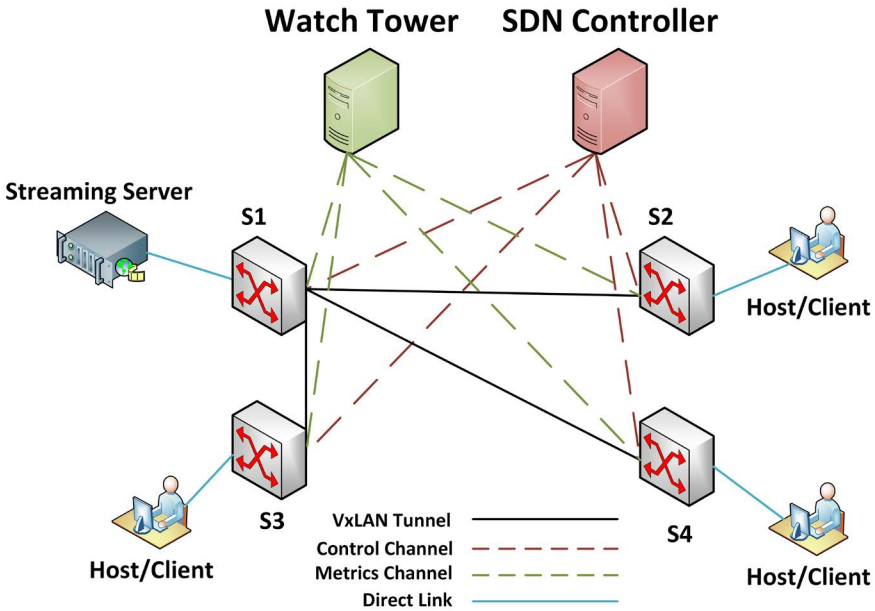


Fig. 2: Streaming architecture in an SDN-based network

Mestres et al. [23] conducted a study using neural networks to model network delays. The authors trained different neural network architectures under various scenarios, including different network typologies, sizes, traffic intensities, and routing. The goal is to provide guidelines for training neural networks to accurately model delays in different network settings.

Selvi et al. [24] address the communication overhead associated with deep learning models, the article utilize fusion learning between the data plane and control plane of the SDN environment. The proposed framework is tested with simulated data on the Abilene network topology with the RYU SDN controller and exhibits an improved accuracy for both local and global models.

All of the aforementioned studies relied solely on simulations, and there is no information on any testbed used. Additionally, most of the studies focused on hop count and control messages, and none of them attempted to forecast the load on network devices. In contrast, this paper aims to create a testbed and implement a video streaming scenario, extracting metrics from the testbed to predict future traffic loads on the network.

III. System Design

This section presents the system architecture and the components involved in the creation of the scenario. Figure 2 depicts the complete architecture of the video streaming system. The system is composed of five components: (a) the video streaming server, (b) clients, (c) Open Virtual Switches (OVSs), (d) the SDN Controller, and (e) the Watch Tower.

The video streaming server streams the video from the library to the clients upon receiving the requests for the video. The streaming server is connected to the OVS *S1* through a direct link. Similarly, the clients, which request the videos from the video streaming server, are directly connected to the other OVSs. In this system, we are more focused on the video traffic prediction of the OVS, we take a simple scenario that only a single client is connected to every OVS. The clients are requesting for a video from a list of videos from the video streaming server.

The three OVSs, S_2, S_3, and S_4, are connected to OVS S_1 through a Virtual Extensible LAN (VXLAN) tunnel, creating a P2P connection. VXLAN is a Layer 2 tunneling protocol that creates a virtual network over a public network. The VXLAN creates a Virtual Local Area Network (VLAN) for each tenant and allows devices on different VLANs to communicate as if they are on the same LAN.

The SDN controller controls the network through a control channel. The SDN controller is a centralized entity that communicates and manages network devices such as switches and routers. Moreover, it also manages the network traffic by implementing the traffic policies and flow rules on the network devices by taking decisions in real time about the network behavior. Network management is performed using Southbound Application Programming Interfaces (APIs), which allow the SDN to manage the network programmatically.

Lastly, a watch-tower is also installed to collect network metrics through metrics channel during communication. When communication takes place between the client and the streaming server, the watch tower extracts the traffic metrics from the OVSs such as transferred bytes, memory utilization, Input/Output operations, and CPU utilization. The aforementioned metrics are then used to train a machine learning model to forecast the traffic burden on S_1, for better traffic management.

IV. Testbed Setup

For the resource forecasting scenario in a SDN-based video streaming network, we created a test bed to implement the video streaming scenario. For this purpose, an SDN-based network is created in such a way that it contains three clients requesting for the videos stored in the video streaming server. A controller is installed at the top of the network for network management. Moreover, a metrics collector is also deployed over the network which collects the metrics to get the real-time monitoring information of the network. The components of the network are shown in Figure 2 and implementation details are discussed in the following sub-sections.

A. Dataset

The dataset utilize in this testbed is downloaded from YouTube, contains 100 videos of different genres. These videos are downloaded from famous YouTube channels such as WWE, Cocomelon - Nursery Rhymes, T-Series, Grizzy & the Lemmings and so on. These genres are then categorized into five general categories, which are Game-play, Animation, Songs, Movies, and Sports. To download YouTube videos, a command-line tool YouTube-dl[1] is used which can download the videos in multiple representations. However, in this scenario, only 240p video representation of videos is considered due to resource constraint.

B. Software-Defined Network (SDN) Controller

A Software-Defined Networking (SDN) Controller is a software application that manages and controls an SDN-based network. The SDN controller acts as the central point of control for the network, providing a unified interface for administrators to manage and configure network devices. The SDN controller communicates with the switches in the network using the OpenFlow protocol, which allows it to program and configure the forwarding behavior of the switches. By decoupling the control plane from the data plane, SDN enables more flexible and programmable network management. In this scenario, a python-based controller RYU is used for the network management.

C. Open Virtual Switch

Open vSwitch (OVS) is an open-source virtual switch that can be used in a variety of virtualization environments, including cloud computing and data centers. The OVS is used for the creation of virtual networks that can span multiple hosts or even data centers. In the aforementioned scenario, virtual machines are used to create the OVS. Each virtual machine is equipped with 4 cores, 100 GB of hard disk, 8 GB memory, and Ubuntu 20.04 LTS operating system. After configuring the virtual machines, the Open vSwitch[2] application is installed on each virtual machine.

[1]https://youtube-dl.org/

[2]https://www.openvswitch.org

D. Streaming Server

A streaming server is a type of server that is used to deliver streaming content to end-users over the Internet. This streaming content can be in the form of live streaming, video on-demand, and other multimedia content. The streaming server is also responsible for encoding, transcoding, and delivering the multimedia content to the end-user. In this scenario, the streaming server is configured as a video on-demand, such that the client requests a already stored video from the streaming server. For this purpose, FFMPEG[3] is used for streaming the videos to the client. A server request agent is deployed at the video streaming server, which accepts the request from the client and send the video to the client. In the current situation, video is streamed to the client by using the User Datagram Protocol (UDP). However, in the future we are planning to deploy the streaming server by using the Dynamic Adaptive Streaming over HTTP (DASH) video delivery mechanism. The video streaming server is equipped with 16 cores processor, 16 GB memory, and 500 GB storage space, and Ubuntu 20.04 LTS operating systems.

E. Clients

In computer network, client is an application which requests some services or resources from the server or any other computer. In this scenario client is a program which randomly request the videos from the dataset. The pattern of client request follows a Poisson distribution with value of lambda equals to 8. The popularity of videos in the dataset follows Zipf distribution. Keeping the values of these parameters, workload for video request for each of the three clients agents are generated and save as a CSV file. The client agents read the CSV files and request the videos from the server. Each of three clients are connected to the three OVS with a direct link.

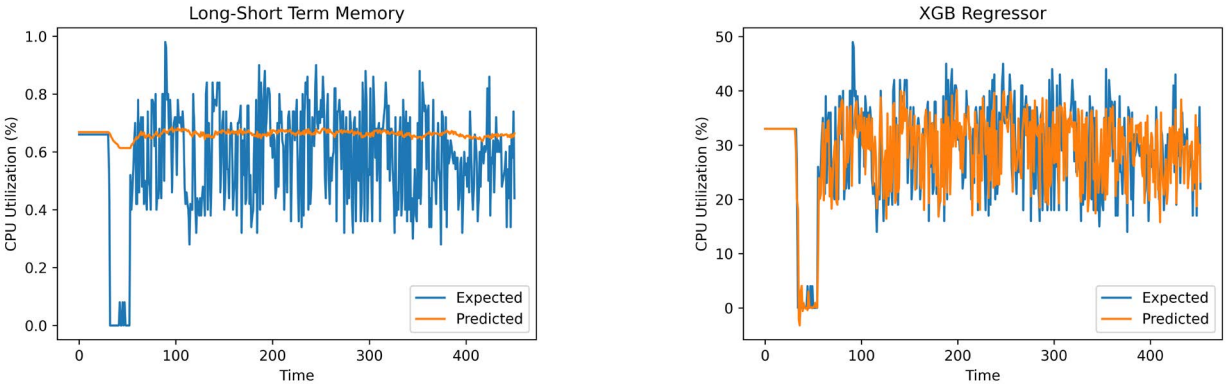
F. Watch Tower

The Watch Tower terminology is used to represent network monitoring tools which oversees the network. A virtual machine is deployed which contains the network monitoring, metrics collectors, and visualization tools. It enables monitoring the network in real-time by capturing, storing, and displaying various metrics of the network. In this scenario, we install a Prometheus exporter called Node Exporters on the OVSs, which exports the node level metrics such as CPU and memory utilization, packets in/out, and traffic bytes coming to and going from the OVS. The aforementioned metrics are exported to the Prometheus. Prometheus is a time-series database which collects and store time-series data, and can be used to monitor and alert on the performance and availability of the network. Grafana is used to visualize these metrics stored in the Prometheus database.

V. Results and Discussion

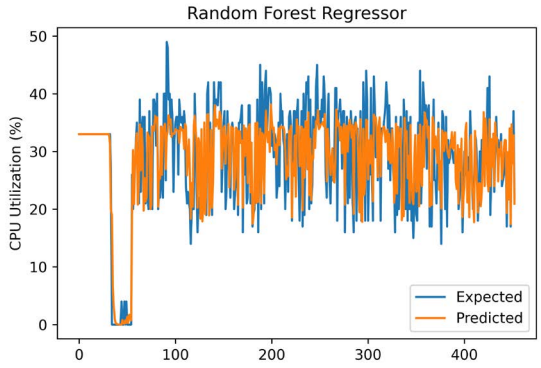
This section presents the results of machine learning models used for predicting the performance metrics gathered from

[3]http://ffmpeg.org/

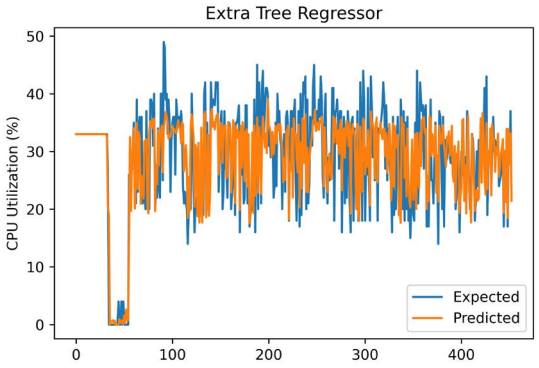


(a) Long-Short Term Memory

(b) XGBoost



(c) Random Forest



(d) Extra Tree

Fig. 3: Difference between actual and predicted CPU utilization by different machine learning models

the monitoring tools mentioned earlier. For this purpose, a request pattern of 24 hours is generated, and each client requested the video from the server, which then streamed the video to the client. During this process, Prometheus collected performance metrics. There are several metrics exported by the node exporter such as Bytes processed, TCP in and out segments, UDP in and out segments, CPU utilization, and memory usage. This paper focuses on the prediction of CPU utilization as output based on the aforementioned performance metrics given as input to different machine learning models. These metrics are important for the network manager to make decisions about implementing policies to reduce the burden on the network.

The performance metrics used to evaluate the performance of the machine learning models are the Mean Absolute Error (MAE), the Mean Squared Error (MSE), and the R2 score. Mean Absolute Error (MAE) is a popular metric used to evaluate the accuracy of a model's predictions. It measures the average of the absolute differences between the predicted values and the actual values over a certain period of time. Similarly, the MSE measures the average of the squared differences between the predicted values and the actual values over a certain period of time. However, the R2 score measures the goodness of fit and can help determine how well the model can generalize to new data.

For resource forecasting, four machine learning algorithms are utilized which are Long Short-Term Memory (LSTM), eXtreme Gradient Boosting (XGBoost), Random Forest (RF), and Extra Tree (ET). LSTM is a type of neural network architecture that is well suited for time series forecasting tasks. LSTM is a type of recurrent neural network (RNN) that have been designed to capture long-term dependencies and patterns in sequential data, making them particularly useful for time series analysis. XGBoost is a popular machine learning algorithm for classification and regression problems, including time series forecasting. XGB's ability to handle complex interactions between variables and nonlinear relationships make it a popular choice for time series forecasting tasks.

RF is also a popular ensemble learning algorithm used in machine learning for both classification and regression tasks. RF works by combining multiple decision trees, each trained on a different subset of the data, to improve the overall accuracy and robustness of the model. ET, or Extremely Randomized Trees, is a variation of the RF algorithm that further randomizes the decision trees by choosing random splits instead of searching for the best split, making it faster to train and less prone to overfitting. ET is also used for both classification and regression tasks.

Figure 3 shows the difference between the actual and the predicted CPU utilization by different machine learning algorithms. From the figure, it can be seen that the LSTM model's performance is not suitable in predicting the traffic load. There is a huge difference between the actual and the predicted values. The reason could be LSTM might not be able to extract the pattern as compared to the other models. In this scenario, RF, ET, and XGBoost shows a similar performance with R2 score equals to 0.68, 0.68, and 0.64 respectively. Table I shows the performance metrics scores of different machine learning models.

TABLE I: Performance metric scores of different machine learning models

Models	MAE	MSE	R2
ET	3.950949	29.12086	0.684705
RF	3.967967	29.49514	0.680652
XBG	4.146592	33.1861	0.64069
LSTM	0.153644	0.043153	-0.16399

VI. Conclusion

The prediction of resources plays an important role in network management. It helps the network administrator to take the precautions proactively. This paper aims to present a machine learning based resource prediction using different machine learning algorithms. For this purpose, four machine learning algorithms are utilized to forecast the CPU utilization on a SDN switch. Moreover, a testbed based on video streaming scenario is implemented and data is extracted with the help of Prometheus. The result shows that ET outperforms its counterpart. However, the R2 score for ET is less than 70. This is due to the fact that the results are based on 24 hours of video streaming scenario synthetic dataset. In the future, the authors aim to increase the dataset to a month and also apply some more complex machine learning and deep learning models for compute and network resources prediction. The load prediction will help the network administrator to take the decision about the network proactively and to prevent the network from overloading.

VII. Acknowledgement

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education (2022R1I1A1A01064556). This research was supported by Brain Pool program funded by the Ministry of Science and ICT through the National Research Foundation of Korea (2019H1D3A1A01102980).

References

[1] U. Cisco, "Cisco annual internet report (2018–2023) white paper," *Cisco: San Jose, CA, USA*, vol. 10, no. 1, pp. 1–35, 2020.

[2] P. Chanclou, A. Cui, F. Geilhardt, H. Nakamura, and D. Nesset, "Network operator requirements for the next generation of optical access networks," *IEEE Network*, vol. 26, no. 2, pp. 8–14, 2012.

[3] D.-E. Meddour, T. Rasheed, and Y. Gourhant, "On the role of infrastructure sharing for mobile network operators in emerging markets," *Computer networks*, vol. 55, no. 7, pp. 1576–1591, 2011.

[4] Y. Yoo, G. Yang, M. Kang, and C. Yoo, "Adaptive control channel traffic shaping for virtualized sdn in clouds," in *2020 IEEE 13th International Conference on Cloud Computing (CLOUD)*. IEEE, 2020, pp. 22–24.

[5] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 27–51, 2014.

[6] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[7] H. Kim and N. Feamster, "Improving network management with software defined networking," *IEEE Communications Magazine*, vol. 51, no. 2, pp. 114–119, 2013.

[8] Z. Shu, J. Wan, J. Lin, S. Wang, D. Li, S. Rho, and C. Yang, "Traffic engineering in software-defined networking: Measurement and management," *IEEE access*, vol. 4, pp. 3246–3256, 2016.

[9] Z. A. Qazi, C.-C. Tu, L. Chiang, R. Miao, V. Sekar, and M. Yu, "Simplefying middlebox policy enforcement using sdn," in *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM*, 2013, pp. 27–38.

[10] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, and W. Chou, "A roadmap for traffic engineering in sdn-openflow networks," *Computer Networks*, vol. 71, pp. 1–30, 2014.

[11] N. L. Van Adrichem, C. Doerr, and F. A. Kuipers, "Opennetmon: Network monitoring in openflow software-defined networks," in *2014 IEEE Network Operations and Management Symposium (NOMS)*. IEEE, 2014, pp. 1–8.

[12] P.-W. Tsai, C.-W. Tsai, C.-W. Hsu, and C.-S. Yang, "Network monitoring in software-defined networking: A review," *IEEE Systems Journal*, vol. 12, no. 4, pp. 3958–3969, 2018.

[13] P. William, S. Choubey, A. Choubey, and A. Verma, "Darknet traffic analysis and network management for malicious intent detection by neural network frameworks," in *Using Computational Intelligence for the Dark Web and Illicit Behavior Detection*. IGI global, 2022, pp. 1–19.

[14] F. Zola, L. Segurola-Gil, J. L. Bruse, M. Galar, and R. Orduna-Urrutia, "Network traffic analysis through node behaviour classification: a graph-based approach with temporal dissection and data-level preprocessing," *Computers & Security*, vol. 115, p. 102632, 2022.

[15] A. Shahraki, M. Abbasi, A. Taherkordi, and A. D. Jurcut, "A comparative study on online machine learning techniques for network traffic streams analysis," *Computer Networks*, vol. 207, p. 108836, 2022.

[16] T. Andrysiak, Ł. Saganowski, M. Choraś, and R. Kozik, "Network traffic prediction and anomaly detection based on arfima model," in *International Joint Conference SOCO'14-CISIS'14-ICEUTE'14: Bilbao, Spain, June 25th-27th, 2014, Proceedings*. Springer, 2014, pp. 545–554.

[17] H. Yin, C. Lin, B. Sebastien, B. Li, and G. Min, "Network traffic prediction based on a new time series model," *International Journal of Communication Systems*, vol. 18, no. 8, pp. 711–729, 2005.

[18] B. Yang, S. Sun, J. Li, X. Lin, and Y. Tian, "Traffic flow prediction using lstm with feature enhancement," *Neurocomputing*, vol. 332, pp. 320–327, 2019.

[19] M. Afaq, S. U. Rehman, and W.-C. Song, "A framework for classification and visualization of elephant flows in sdn-based networks," *Procedia Computer Science*, vol. 65, pp. 672–681, 2015.

[20] F. Tang, Z. M. Fadlullah, B. Mao, and N. Kato, "An intelligent traffic load prediction-based adaptive channel assignment algorithm in sdn-iot: A deep learning approach," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 5141–5154, 2018.

[21] A. Baz, "Bayesian machine learning algorithm for flow prediction in sdn switches," in *2018 1st International Conference on Computer Applications & Information Security (ICCAIS)*. IEEE, 2018, pp. 1–7.

[22] S. Jain, M. Khandelwal, A. Katkar, and J. Nygate, "Applying big data technologies to manage qos in an sdn," in *2016 12th International Conference on Network and Service Management (CNSM)*. IEEE, 2016, pp. 302–306.

[23] A. Mestres, E. Alarcón, Y. Ji, and A. Cabellos-Aparicio, "Understanding the modeling of computer network delays using neural networks," in *Proceedings of the 2018 Workshop on Big Data Analytics and Machine Learning for Data Communication Networks*, 2018, pp. 46–52.

[24] K. T. Selvi and R. Thamilselvan, "An intelligent traffic prediction framework for 5g network using sdn and fusion learning," *Peer-to-Peer Networking and Applications*, vol. 15, no. 1, pp. 751–767, 2022.