Scientific Journal of Informatics
Vol. 9, No. 2, Nov 2022
p-ISSN 2407-7658 http://journal.unnes.ac.id/nju/index.php/sji e-ISSN 2460-0040

Effect of Traditional and Software-Defined Networking on Performance of Computer Network

Isiaka Babatunde Sadiku[1*], Wumi Ajayi[2], Wilson Sakpere[3], Temilola John-Dewole[4], R. A. Badru[5]
[1,2,3,4,5]Department of Computer Science, Faculty of Natural and Applied Sciences,
Lead City University, Ibadan, Nigeria

Abstract.
Purpose: Computer networks and the Internet are changing how we communicate, learn, work, and even play. Conventional computer networks are not smart enough towards processes that contribute to improving online control transactions of services and demand for unlimited communication services. Hence, computer networking has to go smart.
Methods: This paper explores the effect of different computer networking types - traditional computer networking (D0) and Software-Defined Networking (D1). The paper combined traditional computer networking (D0) with Software-Defined Network (D2) running applications (A1, A2, A3, A4, and A5) with the host sending five packets (P1, P2, P3, P4, and P5) across the networks emulated using Mininet network emulation to observe various performance parameters on the network.
Result: It was observed that Application A1 recorded the highest bandwidth, throughput, and latency. The least bandwidth, throughput, and latency were observed in A4. The result showed that, below 80% of the running application's IPv4 packet size (65,507 bytes), the higher the bandwidth, the higher the throughput. Also, the lower the latency, the more statistically similar the jitter experienced. Packet P1 has the highest bandwidth and throughput usage with high latency. The results indicate that the higher the bandwidth and throughput, the higher the latency observed in the packet sent across the network. Traditional computer networking (D1) recorded the highest bandwidth and throughput with the highest jitter. The correlation result showed that the jitter decreases with increasing bandwidth and throughput.
Novelty: This study provides information on traditional computer networking and Software-Defined Networking. The result validates studies that observed significant F-value and stability in the SDN application-awareness experiment.

Keywords: Smart Network, Bytes Size, Bandwidth, Throughput, Latency, Jitter
Received July 2021 / **Revised** March 2022 / **Accepted** October 2022

INTRODUCTION

Computer networks and the Internet are changing how people communicate, learn, work, and play. Computer networks come in different sizes. These technologies are constantly evolving while the demand for unlimited communication services continues to increase for various purposes such as research, education, businesses, online banking, online gaming, shopping, and the new social networking applications [1]–[4]. As a result, the Internet networks architecture would become too complicated and gigantic if the communication network fails to become smart. An Intelligent system or smart software enables control, easy and quick accessibility, and automates operation of different gadgets through which the user can manage and perform the required function at the immediate or remote point [5].

Furthermore, Software-Defined Networking (SDN) enables continuous network development. SDN is a networking paradigm that segregates packet forwarding and accounting (performed on switches) from the routing decisions and advanced protocols (executed on a central controller). In other words, SDN decouples the control plane and data plane from the network core devices for operating purposes. The control plane (i.e., SDN controller) acts as the Network Operating System (NOS). The data plane resides inside the network core devices and is only responsible for forwarding data packets controlled by the central

*Corresponding author.
Email addresses: sibsadiku@gmail.com (Sadiku), wumiajayi1@yahoo.com (Ajayi), sakpere.wilson@lcu.edu.ng (Sakpere), dejob_j@yahoo.com (John-Dewole), olasunkonmi@yahoo.com (Badru)
DOI: 10.15294/sji.v9i2.31315

controller. This segregation increases the networking infrastructure's agility and flexibility and reduces operational expenses [6]. In SDN network, routing and traffic management decisions are made by a centralized controller and communicated to switches via a control channel.

The traditional computer network does not allow a fast evolution towards a process that contributes to improving the online transaction of services. The traditional networks mostly implement devices where the control plane is distributed and mixed with the data plane. Traditional networking consists of computer networking that is primarily implemented from dedicated devices using one or more switches, routers and application delivery controllers. Traditional networking functionality is largely implemented in dedicated hardware, such as application-specific integrated circuits (ASIC). Moreover, SDN is more flexible, allowing users greater control and ease of managing resources virtually throughout the control plane. In contrast, traditional networks use switches, routers, and other physical infrastructure to create connections and run the network. The SDN controller communicates with applications like firewalls or load balancers via its northbound APIs. The controller talks with individual network devices using a southbound interface to configure network devices and choose the optimal network path for application traffic. As a result of this communication, application developers directly program the network, as opposed to using the protocols required by traditional networking. SDN uses open, flexible, and dynamic architecture that is defined through the use of different software programming languages.

In addition, simulation and emulation network platforms play an important role in studying and evaluating different networks' design and performance [7]. Mininet is the most popular SDN platform in the form of a virtual testbed used for testing network tools and protocols [8]–[10]. Also, Quality of Service (QoS) is a set of technologies on a computer network that guarantees its ability to dependably run high-priority applications and traffic under limited network capacity [11]. QoS technologies provide differentiated handling and capacity allocation to specific flows in network traffic to assign handling of packets and afford the bandwidth to that application or traffic flow. The performance parameters commonly measured in QoS are bandwidth (throughput), latency (delay), jitter (variance in latency), and error rate. IPv4 packet (including pings) size ranges from 1 to 65,507 bytes. Ping is the most common network administration utility used to test the reachability of a host on IP networks and to measure the round-trip time for messages sent from the originating host to a destination computer. Ping sends ICMP echo requests/replies to test the connectivity to other hosts. Standard ICMP ping shows that the computer host is responding or otherwise unreachable. In this study, we investigate the effect of traditional and Software-Defined Networking on the performance parameters of computer networks.

Open-Source Hybrid IP/SDN (OSHI) networking has been proposed as a hybrid approach that allows the coexistence of traditional IP routing with SDN-based forwarding within the same provider domain [12]. Wei et al. [13] studied the energy-efficient traffic engineering problem in hybrid SDN/IP networks. They noted that IP routers perform the shortest path routing using the distributed OSPF link weight. The SDNs perform the multi-path routing with traffic flow splitting by the global SDN controller. Thus, the underutilized links can be turned off to save energy [13]. Yang et al. [14] proposed a mechanism for decoupling wireless resource allocation from a mobile terminal by mapping different IP flows to different virtual wireless nodes to support integrated management in wired and wireless networks. The researchers extended the SDN framework to support the virtual network interface so that the traditional controller has a global view of the network and calculates the IP stream policy for the mobile terminal with multiple virtual network interfaces. The result shows that the mechanism achieves efficient IP flow-level management in both wired and wireless networks.

Nacshon et al. [6] proposed 'Floware,' an OpenFlow application that allows discovery and monitoring of active flows at any required aggregation level by monitoring overhead among many switches to reduce its negative effect on network performance. Floware integrates with monitoring systems based on legacy protocols such as NetFlow. Duque et al. [15] observed that due to the centralized control of Software-Defined Networking, forwarding functions with the quality-of-service features are enabled in data networks based on layer-2 devices. The researchers determined the aspects that enable Software-Defined Networking to provide quality of service features in data networks using network simulation over the same base network and under the same working conditions by carrying out measurements of the packet forwarding response time and management of the transported bandwidth. Nugroho et al. [16] reported that the QoS analysis of the Software-Defined Network architecture performed better than conventional traditional network architectures with the value of latency delay on the Software-Defined Network ranging from 0.019ms to

0.084ms, and with 0% packet loss when it addressed network traffics of 10-100Mbps. The authors reported the Quality of Service (QoS) performances between the two networks employing SDN-based architecture and without SDN-based architecture using Mininet as a software emulator. Hu et al. [17] proposed an application-awareness mechanism including three phases, i.e., traffic collection, data pre-processing, and application awareness, which outperforms the three benchmarks on recall ratio, precision ratio, F-value, and stability. The researchers used Convolutional Neural Network (CNN) based on a deep learning mechanism for application awareness. They also noted that the traditional Internet can only obtain the local network view, which belongs to the offline awareness mode and cannot adapt to the dynamical network environment.

METHODS

To investigate the effects of traditional and Software-Defined Networking on the performance parameters of a computer network, fifty hosts from each switch (s1, s2, s3, s4, and s5) were attached to different computer networks. The computer networks are traditional computer networking (D0), Software-Defined Networking (D1), and a combination of traditional and Software-Defined Network (D2) emulated using the default Mininet emulation tool version 2.3.0. Each of the connected links has an optimum bandwidth allocation (10 Mbps) following the procedure of Chaudhari [18]. Figures 1- 3 show the conceptual topology for this experiment. Figure 1 consists of traditional switches (s1, s2, s3, s4, and s5) connected to coordinated switch s6. Figure 2 consists of a controller (c0) connected to the OpenFlow switch as switch s6 coordinating switches s1, s2, s3, s4, and s5. Figure 3 consists of the controller (c0) connected to OpenFlow switches s1, s2, s3, s4, and s5. The experiments were emulated using Mininet running on Ubuntu 20.04 LTS operating system on Lenovo Thinkpad X201t with 8 GB RAM, Intel Core i5 CPU, 2.9 GHz (see Figure 4).

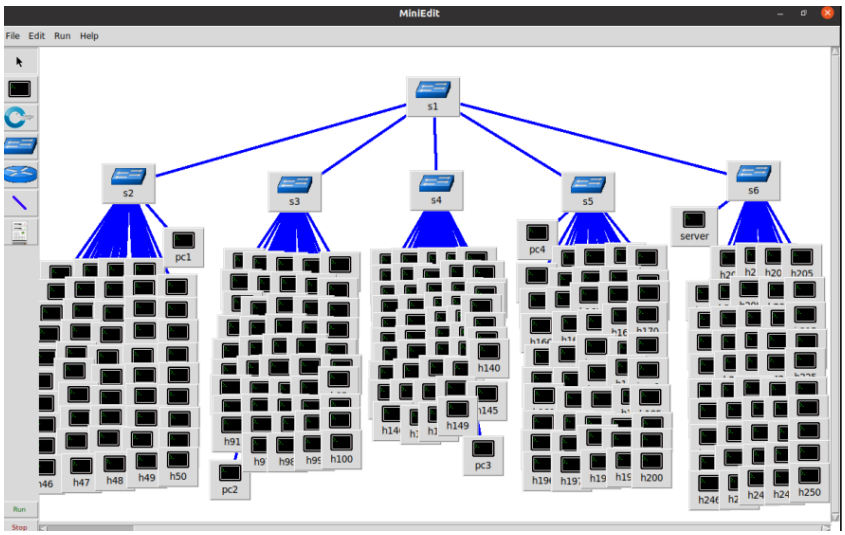

Figure 1. Traditional computer networking (D0)

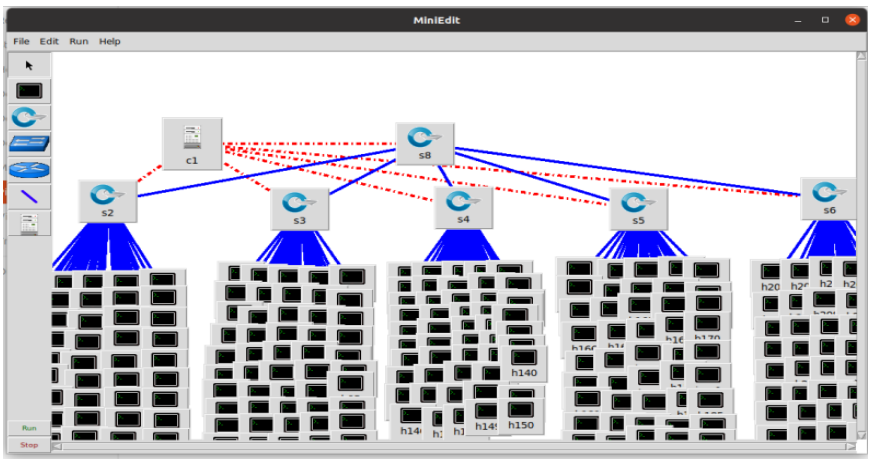

Figure 2. Software-Defined Networking (D1)

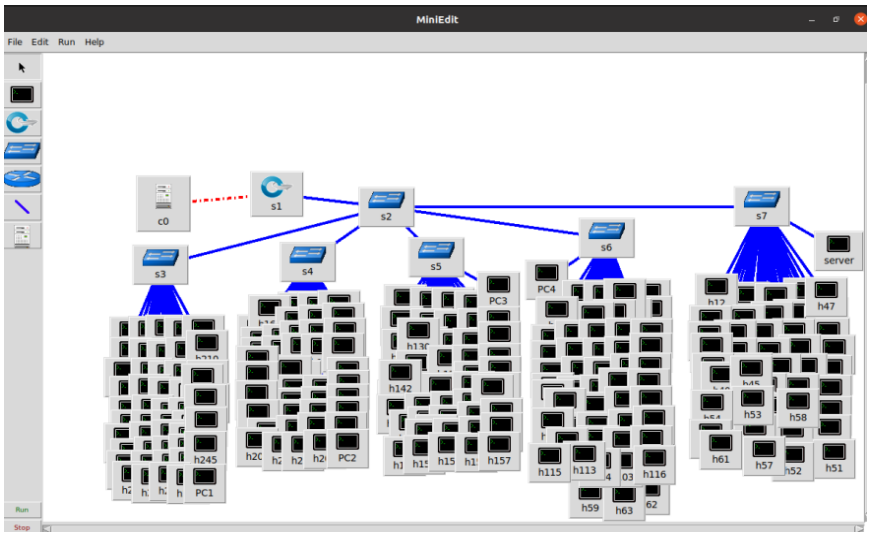

Figure 3. Combined traditional with Software-Defined Network (D2)

The three factors of a factorial experiment were set up in a completely randomized design (D3 × A5 × P5) with ten replicates each on the performance of the computer network based on traditional and Software-Defined Networking. The network type (D0, D1, and D2) is the first factor. The second factor is the Packet Internet groper (Ping) command representing a running application on a host under s1 that sends five (5) different levels (100, 90, 80, 70 % of 65,507 bytes, and 10,000 bytes) engaging the server's services. According to IETF RFC 793 and 768, the theoretical maximum size for Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) is about 64 kB (TCP 65535 bytes and UDP 65507 bytes) [19]. But packets of this size are not commonly used due to the Maximum Transmission Unit (MTU), the maximum data size that can be sent at a time. This means that data with a size larger than the MTU is passed from the transport layer to the network layer, and the data are fragmented into smaller packages that are provided with the IP header with the final destination address. The third factor is the number of packets (32, 64, 128, 256 and 512) sent across the network. To generate traffic, the Iperf command was executed using '**sudo iperf -s -p 5566 -i 1**', to set up the Iperf server for TCP on a server's host command-line interface (CLI) on s5, and '**sudo iperf -s -u -p 5566 -i 1**', for setting Iperf server for UDP traffic on server's host CLI on s5. Iperf client connection from PC3 and PC4 were connected to the TCP and UDP server using '**sudo iperf -c <server address> -p 5566 -i 1**' and '**sudo iperf -c <server address> -u -b 10M –t 15 -p 5566**' respectively. The experiment was carried out using the Mininet emulation tool. Mininet runs the collection of end devices (like Laptops and PC), intermediary devices (like switches and routers), and links on a single Linux kernel. The results from Mininet emulation are the same as an entire computer network [19]. A cleanup process for the experiment is run at the end of every new network topology execution to avoid any previous logs and cache data building up with the current experiment.

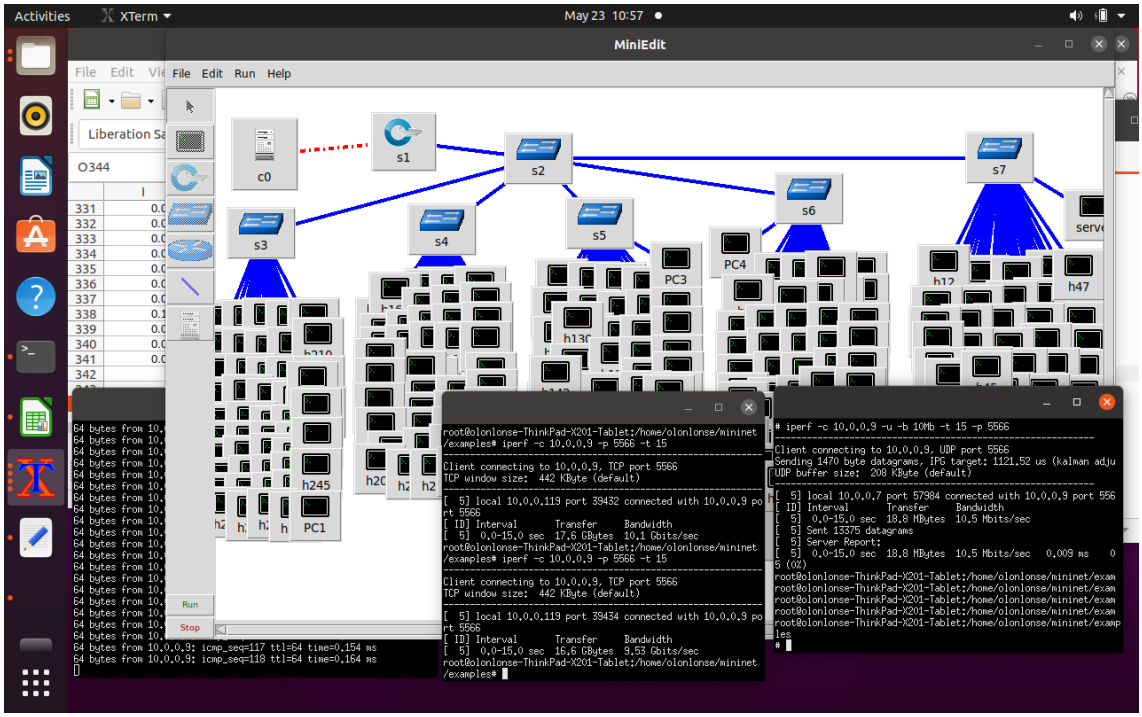

Figure 4. Tests conducted during the experimental study

RESULT AND DISCUSSION

A total of 750 tests were conducted during the experimental study. Table 1 shows that the F-value is high and the Pr > F-value is less than 0.05. The ANOVA results indicate that all the factors used in the tests had a significant effect on latency, bandwidth, throughput, and jitter except the number of packets sent, the interaction of the application running, and the type of network used (see Table 1).

Table 1. Analysis of variance (ANOVA) table

Factor	Latency		Bandwidth		Throughput		Jitter	
	F-value	Pr(>F)	F-value	Pr(>F)	F-value	Pr(>F)	F-value	Pr(>F)
APPLICATION (A)	27.426*	< 2.2e-16	7.107*	1.316e-05	8.651*	8.199e-07	2.342*	0.0536275
DEFINED_ NETWORK (D)	102.907*	< 2.2e-16	335.257*	< 2.2e-16	187.435*	< 2.2e-16	391.088*	< 2.2e-16
PACKET (P)	6.889*	1.945e-05	81.706*	< 2.2e-16	45.340*	< 2.2e-16	1.721ns	0.14ns
A×D	12.638*	< 2.2e-16	9.582*	1.284e-12	12.560*	< 2.2e-16	1.398ns	0.19ns
A×P	4.506*	1.380e-08	5.985*	2.051e-12	5.903*	3.360e-12	3.006*	0.00006978
D×P	6.331*	6.217e-08	44.419*	< 2.2e-16	32.167*	< 2.2e-16	2.957*	0.0029410
A×D×P	4.368*	9.819e-14	5.452*	< 2.2e-16	4.263*	2.944e-13	2.236*	0.0001371

Note: "" means "is significant", "ns" means "is not significant at p≥0.05"*

From Table 2, the influence of the running application on the bandwidth shows that A2 and A3 are statistically similar, as well as A4 and A5. Also, D0 and D1, as well as P3, P4, and P5, are similar. Generally, all the performance parameters are significantly influenced by the types of running applications, defined networks, and packets. The influence of the types of running applications on the bandwidth shows that A1 is the highest, while A4 is the lowest. However, A2 and A3 are statistically similar, as well as A4 and A5. The defined network D0 has the greatest influence on the bandwidth, while D2 has the least influence. Generally, D0 and D1 are statistically similar. The influence of packets on the bandwidth showed that P1, P2, and P3 have the greatest influence on bandwidth. However, P3, P4, and P5 are statistically similar in their influence on bandwidth.

Table 2. Result of duncan's new multiple range tests, provides significance levels for the difference between pair of means

		Bandwidth (Gbps)	Throughput (GB)	Latency (ms)	Jitter
APPLICATION	A1	9.47 ± 0.473^{a}	16.55 ± 0.82^{a}	0.241 ± 0.190^{a}	0.0046 ± 0.0025^{a}
	A2	9.38 ± 0.429^{ab}	16.37 ± 0.73^{ab}	0.202 ± 0.120^{b}	0.0050 ± 0.0028^{a}
	A3	9.37 ± 0.44^{ab}	16.35 ± 0.78^{ab}	0.169 ± 0.054^{c}	0.0049 ± 0.0025^{a}
	A4	9.33 ± 0.54^{b}	16.29 ± 0.94^{b}	0.162 ± 0.033^{c}	0.0048 ± 0.0024^{a}
	A5	9.35 ± 0.34^{b}	16.18 ± 1.12^{b}	0.157 ± 0.049^{c}	0.0046 ± 0.0023^{a}
DEFINED NETWORK	D0	9.54 ± 0.42^{a}	16.57 ± 1.06^{a}	0.152 ± 0.031^{b}	0.0023 ± 0.0014^{c}
	D1	9.56 ± 0.27^{a}	16.69 ± 0.48^{a}	0.161 ± 0.013^{b}	0.0062 ± 0.0020^{a}
	D2	9.04 ± 0.43^{b}	15.78 ± 0.75^{b}	0.247 ± 0.175^{a}	0.0058 ± 0.0020^{b}
PACKET	P1	9.67 ± 0.48^{a}	16.89 ± 0.85^{a}	0.180 ± 0.108^{bc}	0.0046 ± 0.0026^{a}
	P2	9.45 ± 0.33^{b}	16.36 ± 1.14^{b}	0.175 ± 0.103^{c}	0.0048 ± 0.0027^{a}
	P3	9.29 ± 0.40^{c}	16.21 ± 0.69^{bc}	0.202 ± 0.117^{ab}	0.0047 ± 0.0027^{a}
	P4	9.22 ± 0.46^{c}	16.09 ± 0.81^{c}	0.167 ± 0.033^{c}	0.0050 ± 0.0023^{a}
	P5	9.26 ± 0.40^{c}	16.19 ± 0.69^{bc}	0.208 ± 0.154^{a}	0.0047 ± 0.0023^{a}

Note: The mean with the same letter is not significantly different along the column at $p \geq 0.05$

The influence of the type of running application and defined network on throughput follows a similar trend as that of bandwidth. Applications A2 and A3, A4 and A5, have a similar statistical influence on bandwidth, respectively. Also, the defined network D0 and D1 are similar, and packets P3 and P5 have a similar influence on the throughput.

Effect of application on latency showed that A3, A4, and A5 are similar, as well as D0 and D1 having a similar influence on the defined network, while P2 and P4 are similar for the influence of packet on latency. The results of the influence of the variables are quite different on the jitter. All the levels of the running applications as well as packets, have a similar influence on the jitter. However, there are significant variations in the influence of the defined network on the jitter. Application A1 recorded the highest bandwidth, throughput, and latency under the running application. The least bandwidth, throughput, and latency are observed in A4. The result shows that, from 80% downward of the IPv4 packet size (65,507 bytes), the higher the bandwidth, the higher the throughput, and the lower and statistically similar the latency and jitter experienced.

Packet P1 has the highest bandwidth and throughput usage with the highest latency. The results indicate that the higher the bandwidth and throughput, the higher the latency observed in the application running and the packet sent across the network. Traditional computer networking (N0) recorded the highest bandwidth, while N1 recorded the highest throughput and significant impact on the jitter column. The result corroborated [16] as they observed significant F-value and stability in the SDN application-awareness experiment.

The combined traditional with Software-Defined Network (D2) recorded the least bandwidth usage and least throughput, but the highest latency with statistical significance. Jitter in the defined networking is not statistically similar. This suggests that a combined traditional with Software-Defined Network may add to the latency and jitter experience in computer networking. Nugroho et al. note that the QoS parameters are better observed in SDN-based networking [15].

Latency has an insignificant negative correlation with bandwidth, throughput and an insignificant positive correlation with jitter, as seen in Table 3. Bandwidth has a significant positive correlation with throughput but an insignificant negative correlation with jitter. Throughput had a negative insignificant correlation with jitter. The result implies that the higher the bandwidth, the lower the latency; the higher the throughput, the lower the latency. On the other hand, the higher the bandwidth, the higher the throughput. However, the jitter decreases with increasing bandwidth and throughput. The result supports the findings of [19].

Table 3. Correlation matrix

	Latency	Bandwidth	Throughput	Jitter
Latency	1			
Bandwidth	-0.25	1		
Throughput	-0.2	0.86 **	1	
Jitter	0.12	-0.21	-0.15	1

Figures 5-8 show that regression analysis produced a linear relationship between application, packet, and responses (bandwidth, throughput, latency, and jitter).

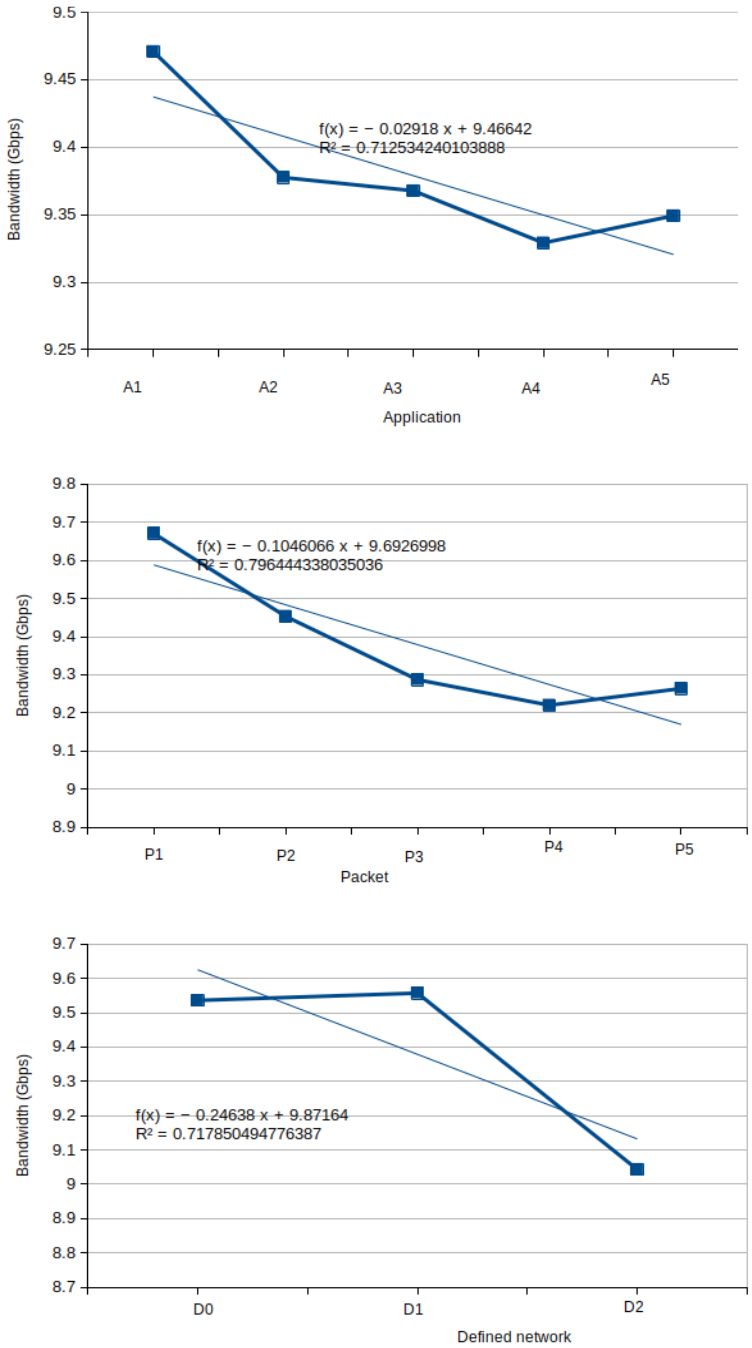

Figure 5. Regression for bandwidth showing relationship between application, packet and defined network

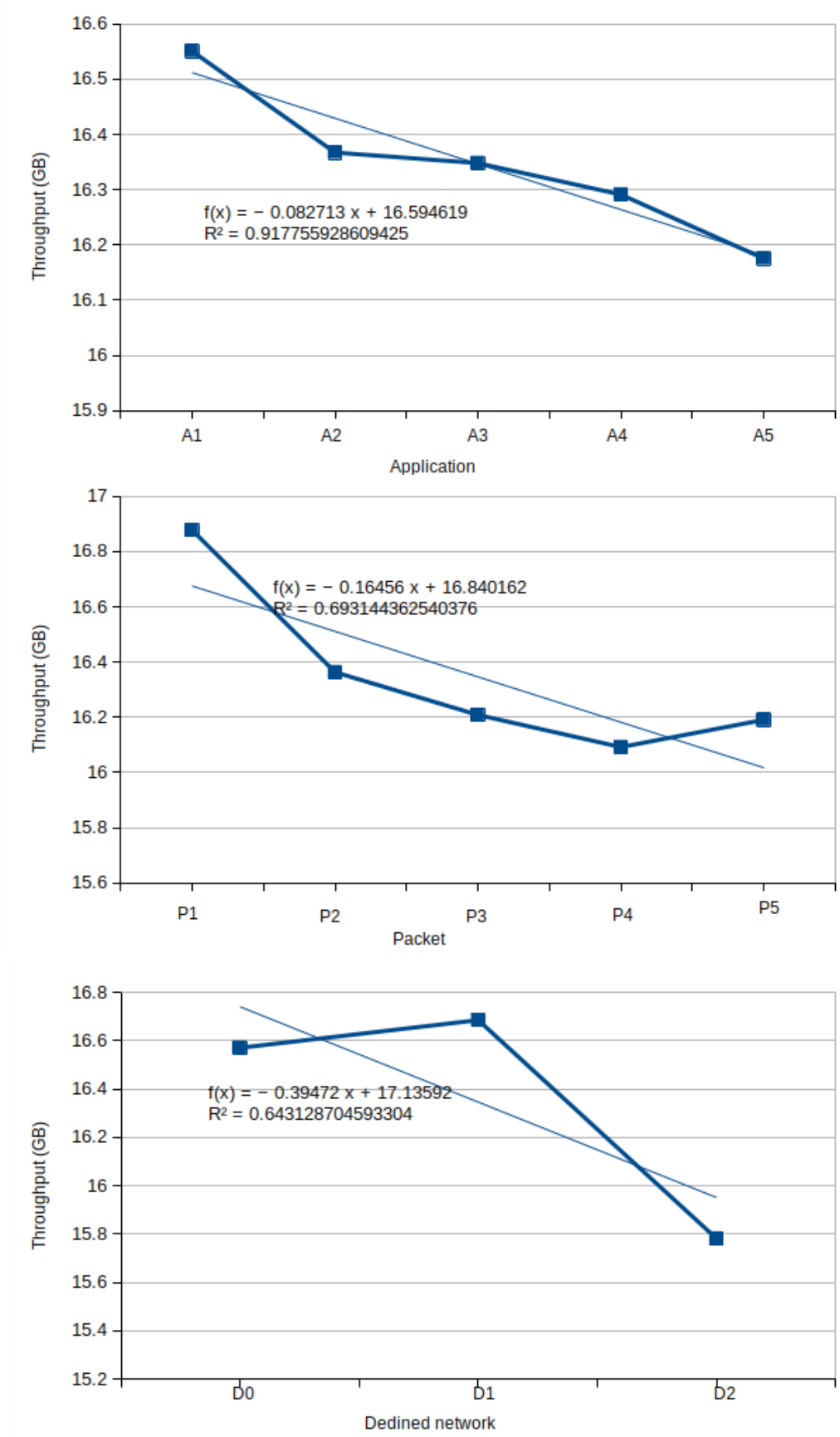

Figure 6. Regression for throughput showing relationship between application, packet and defined network

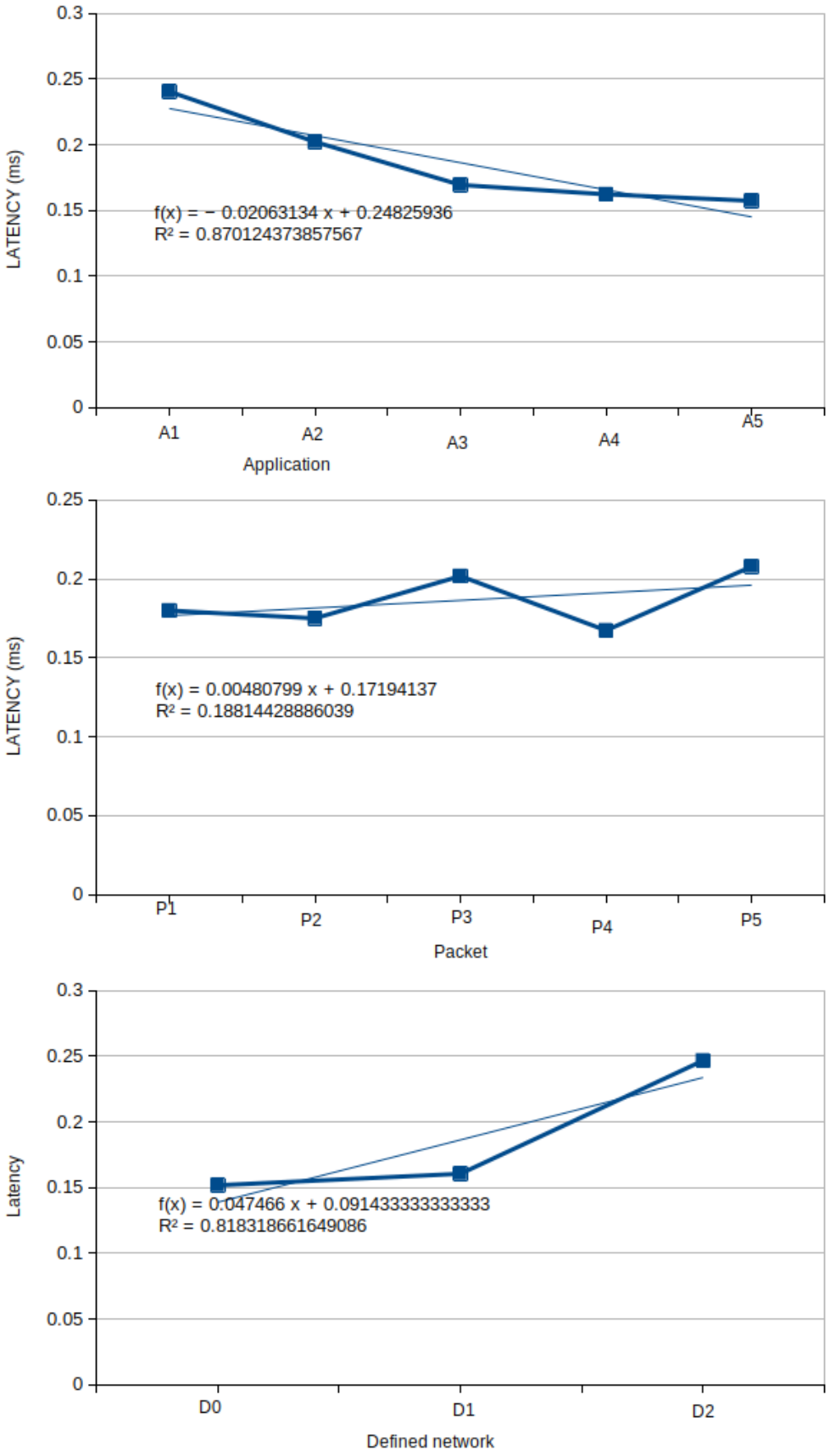

Figure 7. Regression for latency showing relationship between application, packet and defined network

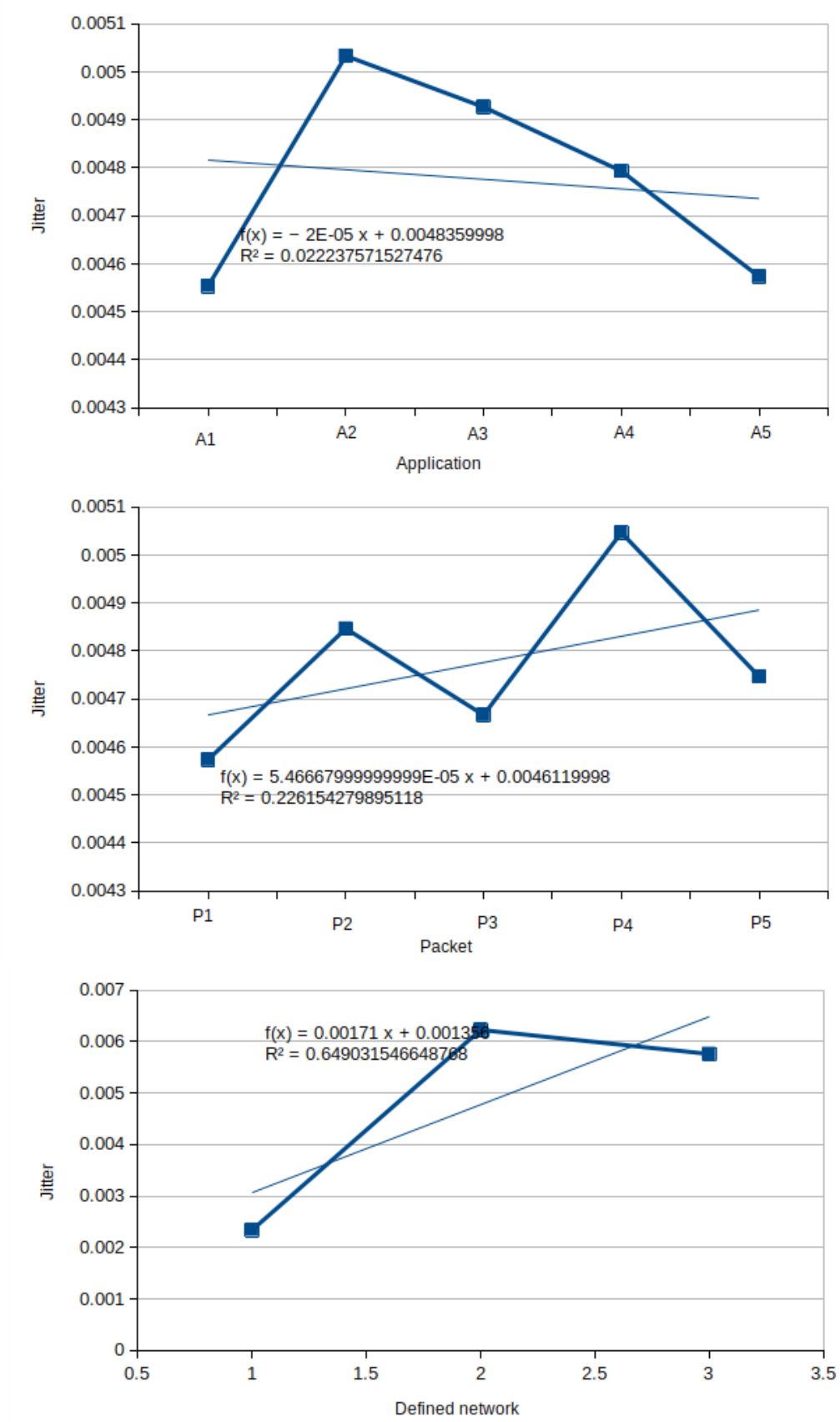

Figure 8. Regression for jitter showing relationship between application, packet and defined network

CONCLUSION

The fact that computer networking has to go smart and the growing popularity of SDN within computing environments has led to an interest in observing the effect of different computer networking types. The traditional computer networking (D0), Software-Defined Networking (D1), and combined traditional with Software-Defined Network (D2) were emulated using Mininet network emulation to observe various performance parameters on the network. It was revealed that Application A1 recorded the highest bandwidth, throughput, and latency under the running application. The least bandwidth, throughput, and latency were observed in A4. The result showed that at less than IPv4 packet size of 65,507 bytes, the higher the bytes size of the application running at high bandwidth and throughput, the higher the latency. Packet P1 has the highest bandwidth and throughput usage with the highest latency. The results indicate that the higher the bandwidth and throughput, the higher the latency observed in the application running and the packet sent across the network. Traditional computer networking (N0) recorded the highest bandwidth, and N1 recorded the highest throughput and significant impact on the jitter column. The result supports Hu et al. (2020), as they observed significant F-value and stability in the SDN application-awareness experiment.

ACKNOWLEDGEMENT

The authors are grateful to the Mininet 2.3.0 core team led by Bob Lantz, the Mininet Contributors, as well as other organizations that have supported the Mininet effort financially or otherwise. We thank the anonymous reviewers for their comments on this paper.

REFERENCES

[1] S. Makridakis, "The forthcoming Artificial Intelligence (AI) revolution: Its impact on society and firms," *Futures*, vol. 90, pp. 46–60, 2017.

[2] M. Meeker, *Internet trends*. Kleiner Perkins, 2018.

[3] J. Firth *et al.*, "The 'online brain': how the Internet may be changing our cognition," *World Psychiatry*, vol. 18, no. 2, pp. 119–129, 2019.

[4] W. Sakpere, M. Adeyeye-Oshin, and N. B. W. Mlitwa, "A state-of-the-art survey of indoor positioning and navigation systems and technologies," *S. Afr. Comput, J.*, vol. 29, no. 3, pp. 145–197, 2017.

[5] W. S. Ajayi and O. Awodele, "Evolution of Digital Edifices: from Shanks and Adobe to Smart and Intelligent Edifice; a Trail to the Future," *Int. J. Multidiscip. Sci. Eng.*, vol. 8, no. 2, pp. 11–17, 2017.

[6] L. Nacshon, R. Puzis, and P. Zilberman, "Floware: Balanced flow monitoring in software defined networks," *arXiv:1608.03307*, pp. 1–13, 2016.

[7] D. V. Khoa and T. N. N. Khanh, "Emulation of software-defined network using mininet," *Dalat Univ. J. Sci.*, vol. 11, no. 1, pp. 80–92, 2021.

[8] E. Sturzinger and S. Cilenti, "A Hybrid Software Defined Network Platform for Undergraduate Research and Education," in *West Point Res. Pap. Proc. Natl. Conf. Undergrad. Res. (NCUR) 2019*, 2019, pp. 11–13.

[9] M. Jadin, O. Tilmans, M. Mawait, and O. Bonaventure, "Educational Virtual Routing Labs with IP Mininet," in *ACM SIGCOMM Educ. Workshop 2020*, 2020, pp. 1–5.

[10] X. Yuan, Z. Liu, Y. Park, H. Hu, and H. Li, "Teaching SDN Security Using Hands-on Labs in CloudLab," *J. Colloq. Inf. Syst. Secur. Educ.*, vol. 7, no. 1, pp. 1–6, 2020.

[11] V. Koryachko, D. Perepelkin, M. Ivanchikova, V. Byshov, and I. Tsyganov, "Analysis of QoS metrics in software defined networks," in *2017 6th Mediterr. Conf. Embed. Comput. (MECO)*, 2017, pp. 1–5.

[12] S. Salsano *et al.*, "Hybrid IP/SDN Networking: Open Implementation and Experiment Management Tools," *IEEE Trans. Netw. Serv. Manag.*, vol. 13, no. 1, pp. 138–153, 2016.

[13] Y. Wei, X. Zhang, L. Xie, and S. Leng, "Energy-aware traffic engineering in hybrid SDN/IP backbone networks," *J. Commun. Networks*, vol. 18, no. 4, pp. 559–566, 2016.

[14] H. Yang, H. Li, and Q. Wu, "IP-Stream Oriented Management Mechanism in 802.11 Wireless Network by Extending SDN," in *2017 IEEE Wirel. Commun. Netw. Conf. (WCNC)*, 2017, pp. 1–6.

[15] J. H. P. Duque, D. O. D. Beltrán, and G. A. P. Leguizamón, "On the features of Software Defined Networking for the QoS provision in data networks," *INGE CUC*, vol. 14, no. 2, pp. 106–115, 2018.

[16] H. P. Nugroho, M. Irfan, and A. Faruq, "Software Defined Networks: A Comparative Study and Quality of Services Evaluation," *Sci. J. Informatics*, vol. 6, no. 2, pp. 181–192, 2019.

[17] N. Hu, F. Luan, X. Tian, and C. Wu, "A Novel SDN-Based Application-Awareness Mechanism by Using Deep Learning," *IEEE Access*, vol. 8, pp. 160921–160930, 2020.

[18] J. P. Chaudhari, D. K. Kirange, K. S. Bhagat, and S. D. Patil, "Evaluation of Bandwidth Utilization in SDN," *J. Xi'an Shiyou Univ.*, vol. 15, no. 1, pp. 21–30, 2019.
[19] L. L. Zulu, K. A. Ogudo, and P. O. Umenne, "Emulating Software Defined Network Using Mininet and OpenDaylight Controller Hosted on Amazon Web Services Cloud Platform to Demonstrate a Realistic Programmable Network," in *2018 Int. Conf. Intell. Innov. Comput. Appl. (ICONIC)*, 2018, pp. 1–7.