

K-nearest neighbor algorithm analysis for path determination in network simulation using software defined network

Ritzkal¹, Suhadi^{2,3}, Rizky Amalia¹, Yuggo Afrianto¹, Anggra Triawan⁴, Syafrial⁵, Fety Fatimah¹

¹Department of Informatics Engineering, Ibn Khaldun University, Bogor, Indonesia

²Department of Informatics of Engineering, High School of Informatics and Computer Management Bani Saleh, Bekasi, Indonesia

³Ministry of Marine Affairs and Fisheries Republic of Indonesia, Jakarta, Indonesia

⁴Department of Informatics of Engineering, Binaniaga Indonesia University, Bogor, Indonesia

⁵Department of Informatics of System, Binaniaga Indonesia University, Bogor, Indonesia

Article Info

Article history:

Received Sep 24, 2022

Revised Oct 6, 2022

Accepted Nov 27, 2022

Keywords:

Jitters

K-nearest neighbor

Packet loss

Software define network

Throughput

ABSTRACT

Software defined network (SDN) is a new approach concept for planning, developing, and operating computer networks. Routing is the process of selecting paths on a network system to send or forward packets to the destination network. This study aims to obtain the results of the calculation analysis using the k-nearest neighbor (k-NN) method as the implementation of the recommendations. The results of the analysis use a mesh topology design approach with predicted values of throughput, jitter, delay, and packet loss. This value is used as a recommendation to the network manager to determine the best path. The best path selection from the analysis results is i) path-1 (very good) which includes switch 1, switch 2, and switch 4; ii) path-2 (good) to switch 1, switch 3 and switch 4 switch 4; iii) channel-3 (moderate) through switch 1, switch 2, switch 3 and switch 4; and iv) channel-4 (bad) through switch 1, switch 3, switch 2, and switch 4. While the calculation using the confusion matrix is accuracy=72.31%, precision=96.08% and recall=87.84%.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Ritzkal

Department Informatics of Engineering, Universitas Ibn Khaldun

Jalan KH. Sholeh Iskandar KM 2 Kedung Badak, Bogor, Indonesia

Email: ritzkal@ft.uika-bogor.ac.id

1. INTRODUCTION

The need for network equipment technology in various companies is experiencing very rapid development, network performance is adding increasingly complex and large configurations, the network control section will also be more complicated, inflexible and difficult to manage [1]. Software defined networking (SDN) is a network paradigm where the control plane is separate from the forwarding plane, SDN is expected to be able to carry out the methods found in conventional networks such as internet protocol (IP) forwarding and routing [2]. With this technology, it is able to manage networks efficiently and change complex network architectures to be simple and easy to manage with different concepts from traditional networks, it is different from modern concepts [3]. The most important component is the controller, which directly controls the configuration of the network device itself [4]. Basically a controller centralizes network intelligence, while the network maintains the data plane distributed by the OpenFlow switch [5]. Routing is the process of selecting paths for data traffic on one or several networks so that the transfer process of routing routes can only be carried out by network administrators, this will cause greater network downtime if there is a link failure [6]. To be able to overcome this problem, it is necessary to apply dynamic routing technology, one of which is the OpenFlow routing protocol, namely a routing which is a process where a router will

choose a route or path to send or forward a packet to the destination network [7]. Determination of the best path that will be passed by the information sent from the sender to the recipient, routing is one that is used in the process of exchanging data or information.

In previous studies, the increasing demand to move from traditional networks to software-defined networks, this has led to many challenges, so software-defined networks are constantly evolving, which includes the need to address issues such as scalability, and packet loss, transmission delays, and network congestion. This research introduces the concept of multi-controller architecture, although it does not help to balance the load among network systems, so that it will solve the problem of network congestion through load distribution between them. This study proposes a linear multi-controller architecture to explore the impact of increasing the number of linearly connected controllers on network performance. This study is based on simultaneous multi-stream generation of different sizes using the distributed internet traffic generator (D-ITG). From the results of the study, it can be concluded that performance is improved uniformly with the number of controllers while maintaining the same number of open vswitches. As the number of controllers reached four, the quad-controller. Architecture recorded the best results in terms of increasing average delay and average jitter to 62% and 64% as well as increases in throughput, bytes received, average packet speed to 32% and 31.8% [8].

This study uses an OpenFlow controller type that is used to search for a special path on the SDN network. The controller will find the SDN network topology. To find a special path that will be passed by the data, the k-nearest neighbor (k-NN) algorithm is used, a method that calculates the similarity distance between data, data that has the closest similarity distance will be grouped into the same group. Determination of the number of groups (k) can be done by various methods, among others, determined randomly, or determined with certainty [9], [10]. There are four lines that consist of line one for streaming video services with a bandwidth of 2,837 kbps based on predictions of internet user behavior, line two for social media services with a bandwidth of 996 kbps, line three for e-commerce services. with a bandwidth of 875 kbps, and line four for download with a bandwidth of 678 kbps. The implementation of network routing uses software, namely Mininet, where mininet is an emulator that is used to make network topology prototypes. Mininet was created with the aim of supporting research in the field of SDN and OpenFlow, so as to create an easier network, SDN was used [11], [12].

2. METHOD

Three types of models, namely physical, analytical, and simulation models, are used to identify numerical indicators of complex system functions and verify that they comply with the requirements [13], [14]. The physical model presupposes the network component's deployment on actual hardware and actual operation to ascertain performance characteristics. Figure 1 mesh topology design in this study.

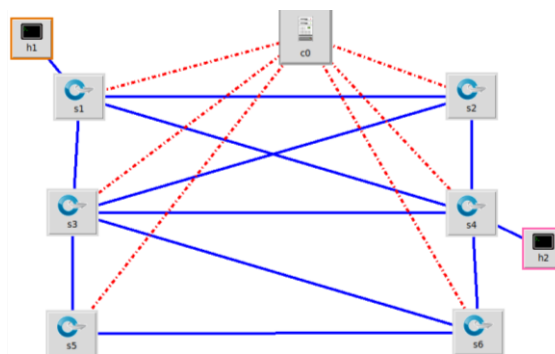


Figure 1. Mesh topology

To obtain the value that is most similar to the genuine value using this method. The topology design is made by applying the k-NN algorithm on the OpenFlow network. The mesh topology in this study uses 4 switches, 2 hosts, and 1 controller. This topology is applied to the mininet emulator [15]. Switch 1 is connected to switch 2 and switch 3, switch 2 is connected to switch 3 and switch 4, switch 3 is connected to switch 4. Switch 1 is connected to host 1, switch 4 is connected to host 2. In the path search process based on the k-NN algorithm there are stages that are carried out, to get a path based on the k-NN algorithm [16]. In Figures 2-5, the tracking performance will be tested based on the scenarios that have been used.

PATH 1

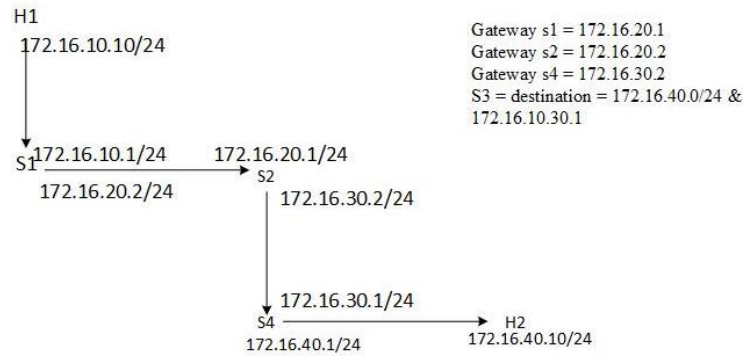


Figure 2. Path-1 scenario testing

PATH 2

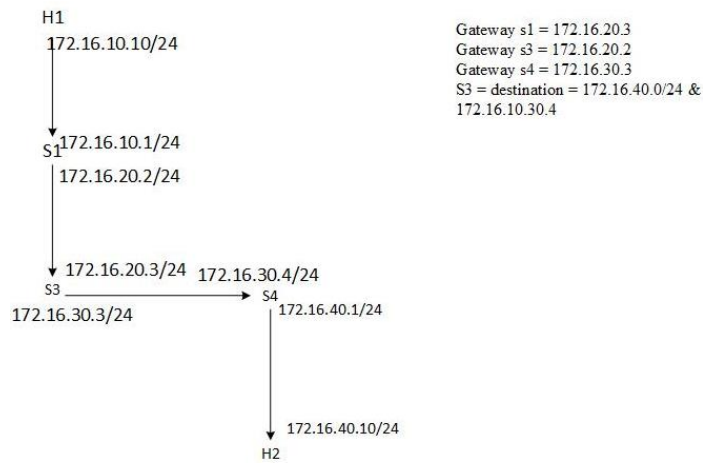


Figure 3. Path-2 scenario testing

PATH 3

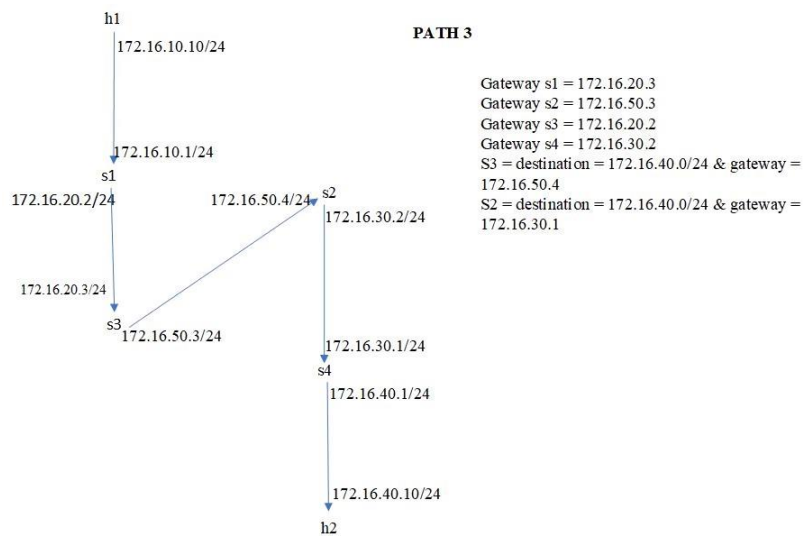


Figure 4. Path-3 scenario testing

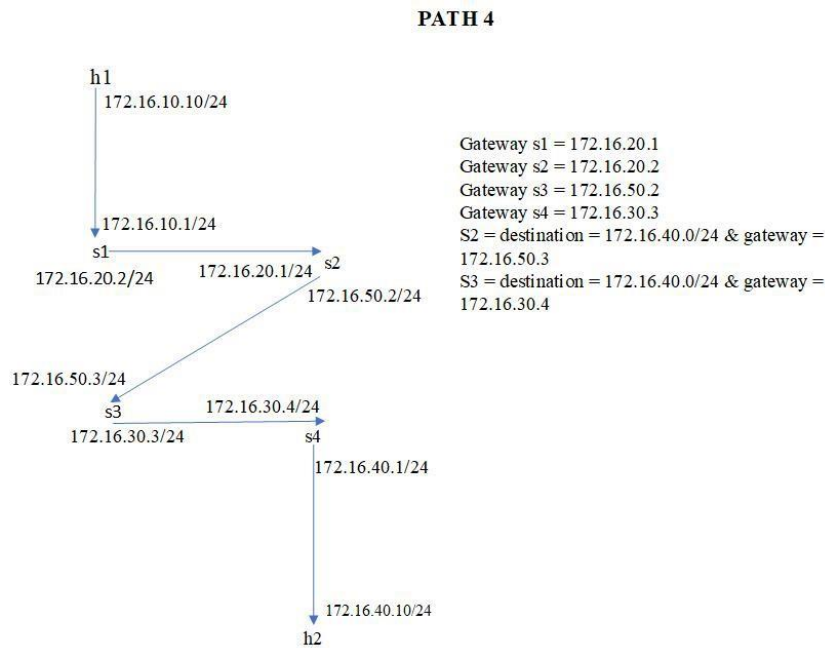


Figure 5. Path-4 scenario testing

2.1. Open networking foundation

To study the performance of SDN according to the open networking foundation (ONF) it is defined as a network architecture that separates control and forwarding functions to enable network control to be programmed directly and the underlying infrastructure for network applications and services [17]. The SDN architecture divides the network into 3 layers, namely the application layer, control layer, and infrastructure/data layer. The application layer is an interface to an admin or researcher in managing or developing a SDN network. The control plane contains a controller that is centralized and based on software. Subordinate hardware is fully controlled by the controlling plane or controller in making forwarding decisions, all subordinates are connected to the controller [18], [19]. The purpose of this SDN network method is to make it easier to build a computer network, make it easier to maintain. The working principle of the switch in the SDN network method, among others, is to connect a switch with a switch, a switch with a controller, and a switch with a host [20], [21]. The analysis was carried out to see the use of the k-NN algorithm method that affects the performance of the custom topology network. SDN is a new application provided in ONF and is effectively used to monitor networks with very good performance, each test is carried out when given background traffic of 5 mbps, 10 mbps, and 15 mbps from each lane, the final results and analysis refer to the telecommunications and internet protocol harmonization over network (TIPHON) version of the quality of service (QoS) standard [22], [23]. OpenFlow is a standard open communication protocol between the data plane and control plane on SDN. In contrast to conventional networks, where the data plane and control plane functions are on a switch/router network, the OpenFlow installed network allows the data plane and control plane functions to be on separate devices, making it easier to perform extensions or add functions to a network. A switch in OpenFlow is just a data plane device that only performs a packet forwarding function and a set of actions or actions that are applied to the packet [24].

2.2. Network communication data

Throughput is the effective data transfer rate, measured in bits per second (bps) which is the total number of observed packet arrivals at the destination during a certain time interval divided by the duration of that time interval [25]. The general formula for throughput efficiency can be calculated in more than one way, but the general formula is $I=R \cdot T$. Inventory is rate multiplied by time, where rate is throughput. But if to calculate R, it will get $R=I/T$, or rate=inventory divided by time in data transmission, network throughput is the amount of data moved successfully from one place to another in a given time period, and typically measured in bps, as in megabits per second (mbps) or gigabits per second (gbps) [26]. Packet loss describes lost packets of data not reaching their destination after being transmitted across a network. Packet loss occurs when network congestion, hardware issues, software bugs, and a number of other factors cause dropped

packets during data transmission [27]. The probability that the transmission channel which precedes the other wavelength of wave length between two nodes is not occupied.

Delay (latency) is the time it takes data to travel the distance from origin to destination. Delay can be affected by distance, physical media, congestion or also long processing times [28]. Network delay is a design and performance characteristic of a telecommunications network. It specifies the latency for a bit of data to travel across the network from one communication endpoint to another. It is typically measured in multiples or fractions of a second [29]. If there are multiple active sessions, the delay will become significant. Increasing bandwidth decreases transmission delay. The medium access control (MAC) protocol largely influences the delay if the link is shared among multiple devices. Sending and receiving a packet involves a context switch in the operating system, which takes a finite time [30], where distance it takes more time to reach the destination if the distance of the medium is longer and velocity if the velocity (speed) of the signal is higher, the packet will be received faster. Jitter is a variation of the arrival time of data packets. If translated, the data sent from the sender will be in the form of packets that will be sent at the same time. However, the arrival of the package may not be together. This time lag is called jitter [31]. Jitter in IP networks is the variation in the latency on a packet flow between two systems when some packets take longer to travel from one system to the other. Jitter results from network congestion, timing drift and route changes.

2.3. K-nearest neighbor algorithm

The k-NN is the simplest classification algorithm in classifying an image into a label, this method is easy to compare with other methods because it classifies based on the closest distance to the object (neighbor). Used to classify an object, based on the k training data that are closest to the object. The condition for the value of k is that it cannot be greater than the number of training data, and the value of k must be odd and more than one. The proximity or distance of the closest training data to the object to be classified can be calculated using the cosine similarity method [32]. Cosine similarity is a method or method that can be used to see how far it is found to test the size used as a distance approach based where $d(i, j)$ =distance from i-data to j-data, x_{i1} =nth data in i-data and x_{j1} =nth data in j-data as (1):

$$SD = \sqrt{(|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + \dots + |x_{in} - x_{jn}|^2)} \quad (1)$$

for the formula of the k-NN algorithm where x_1 is sample data, x_2 is testing data, i is data variable.

2.4. Confusion matrix

Confusion matrix is a table consisting of the number of rows of test data that are predicted to be true and false by the classification model, this table is needed to determine the performance of a classification model [33]. In the calculation of accuracy, there are four combinations of predicted values and actual values. The four terms are true positive (TP) value, true negative (TN) value, false positive (FP) value, and false negative (FN) value. The TP value is positive data that is predicted to be true. The TN value is the number of correctly detected negative data. The value of FP or also called type-1 error is negative data but is detected as positive data. The value of FN or also called type-2 error is the opposite of true positive, where positive data is detected as negative data [34], as (2):

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (2)$$

Where accuracy is the comparison value of the correct prediction with the entire data. After the results of the model evaluation are obtained, an analysis of the results is carried out on the accuracy value of each combination of methods.

3. RESULTS AND DISCUSSION

Link-state data is taken into account when routing decisions are made using SDN. The k-NN algorithm makes full use of the current traffic matrix to determine and install, in advance, the ideal path in the tracing device. The findings demonstrate that topology design, routing algorithm design, and testing data, namely the k-NN method, are solutions for routing as SDN controllers. In order to generate communication data lines calculated in accordance with the Tiphon standard [35].

3.1. Training data

In Table 1, the training data is used to train the k-NN algorithm in finding a suitable model. The training data used in this study were 85 (eighty five) training data. The data tested included communication data, namely throughput (kbps), data packet loss (%), data delay (ms), and data jitter (ms).

Table 1. Training data

No	Communication data			
	Throughput (kbps)	Paketloss (%)	Delay (ms)	Jitter (ms)
1	0.27	0.32	90	12.93
2	0.00	0.1	30	0.03
3	0.00	-	40	4.85
4	724.5	22	252	70
5	0.00	-	29.79	2.98
6	0.00	0.01	30	0.05
7	0.00	-	30	7.27
8	0.97	0.03	40	0.00
9	0.00	-	1.29	0.00
10	0.00	-	790	0.00
11	0.00	-	30	1.41
12	0.00	-	53.28	4.06
13	0.00	127	16.7	445
14	0.00	16	350	128
15	0.00	28.3	300	130.01
16	0.00	3.73	350	126.67
17	1,734	-	20	0.1
18	1,915	0.3	35	0.001
19	1,843	0.51	40	10
20	1,781	0.01	70	7.21
21	1,265	17	90	75.3
22	763	-	354	76
23	725	16	300	75
24	543	4.20	220	55
25	2,735	26	75	0.2
...
84	0.00	3.80	35	126.66
85	1,728	0.5	26	0.3

3.2. Data testing

In Table 2, data testing is used to determine the performance of the k-NN algorithm that has been trained when finding new data that has never been seen before. Data testing will be used to test and determine the performance of the model obtained at the testing stage for each communication data. The test data used in this study consisted of 36 (thirty six) test data. The data tested includes communication data, namely throughput (kbps), data packet loss (%), data delay (ms), and data jitter (ms).

Table 2. Data testing

No	Communication data			
	Throughput (kbps)	Paketloss (%)	Delay (ms)	Jitter (ms)
1	1.34	0.02	170	70
2	0.02	15	450	250
3	700	15.91	300	80
4	3.59	15	90	100
5	26	127.4	370	170
6	1,334	0.25	150	75
7	0.57	20	250	127
8	750	25	370	130
9	780	2.5	140	170
...
35	0.02	20	454	252
36	780	24	167	80

3.3. K-nearest neighbor algorithm calculation process

The calculation process in the k-NN algorithm is an euclidean distance calculation, which is a search method between two variable points, the closer and similar the smaller the distance between the two points [36]. Euclidean distance is said to be good if the new data has a minimum distance and has a high similarity. The distance calculation is carried out using the square distance (SD) formula shown in (1). The results of the analysis show that the prediction process is carried out after calculations using the k-NN algorithm with a total of 58 training data as many as 36 test data as follows:

- Test data-1: throughput=1.34; packetloss=0.02; delay=170; jitter=70

$$SD - 1 = \sqrt{(0.265 - 1.347)^2 + (0.32 - 0.02)^2 + (90 - 170)^2 + (12.93 - 70)^2} = 98.27$$

$$SD - 2 = \sqrt{(0.001 - 1.347)^2 + (0.10 - 0.2)^2 + (30 - 170)^2 + (0.03 - 70)^2} = 156.51$$

$$SD - 3 = \sqrt{(0.001 - 1.347)^2 + (0 - 0.02)^2 + (40 - 170)^2 + (4.85 - 70)^2} = 145.41$$

$$SD - 35 = \sqrt{(0.617 - 0.297)^2 + (0.21 - 20)^2 + (40 - 454)^2 + (0 - 252)^2} = 147.647$$

$$SD - 36 = \sqrt{(0.001 - 780)^2 + (0 - 24)^2 + (20 - 167)^2 + (0 - 80)^2} = 165.534$$

- Test data-2: throughput=0.02; packetloss=15; delay=450; jitter=250

$$SD-1 = \sqrt{(0.265 - 0.029)^2 + (0.32 - 15.913)^2 + (90 - 450)^2 + (12.93 - 70)^2} = 431.29$$

$$SD-2 = \sqrt{(0.001 - 0.029)^2 + (0.1 - 15.913)^2 + (30 - 450)^2 + (0.03 - 250)^2} = 488.98$$

$$SD-3 = \sqrt{(0.015 - 0.029)^2 + (0 - 15.913)^2 + (40 - 450)^2 + (4.85 - 250)^2} = 477.93$$

$$SD-35 = \sqrt{(0.617 - 0.297)^2 + (0.21 - 15)^2 + (40 - 450)^2 + (0 - 250)^2} = 480.433$$

$$SD-36 = \sqrt{(0.001 - 780)^2 + (0 - 24)^2 + (20 - 167)^2 + (0 - 80)^2} = 497.619$$

- Test data-3: throughput=700; packetloss=15.91; delay=300; jitter=80

$$SD-1 = \sqrt{(0.265 - 700)^2 + (0.32 - 15.913)^2 + (90 - 300)^2 + (12.93 - 80)^2} = 733.8$$

$$SD-2 = \sqrt{(0.001 - 700)^2 + (0.1 - 15.913)^2 + (30 - 300)^2 + (0.03 - 80)^2} = 754.68$$

$$SD-3 = \sqrt{(724.5 - 700)^2 + (22 - 15.913)^2 + (252 - 300)^2 + (70 - 80)^2} = 3.041.3$$

$$SD-35 = \sqrt{(0.617 - 700)^2 + (0.21 - 15.913)^2 + (940 - 300)^2 + (0 - 80)^2} = 750.587$$

$$SD-36 = \sqrt{(0.018 - 700)^2 + (0 - 15.913)^2 + (20 - 300)^2 + (0.001 - 80)^2} = 758.587$$

- Test data-35: throughput=0.029; packetloss=20; delay=454; jitter=252

$$SD-1 = \sqrt{(0.265 - 0.359)^2 + (0.32 - 16.9)^2 + (90 - 200)^2 + (12.93 - 70)^2} = 125.02$$

$$SD-2 = \sqrt{(0.001 - 0.359)^2 + (0.1 - 16.9)^2 + (30 - 200)^2 + (0.03 - 70)^2} = 184.6$$

$$SD-3 = \sqrt{(0.001 - 0.359)^2 + (0 - 16.9)^2 + (40 - 200)^2 + (4.85 - 70)^2} = 173.58$$

$$SD-35 = \sqrt{(0.617 - 0.029)^2 + (0.21 - 20)^2 + (40 - 454)^2 + (0 - 252)^2} = 485.25$$

$$SD-36 = \sqrt{(0.001 - 0.029)^2 + (0 - 20)^2 + (20 - 454)^2 + (0.001 - 252)^2} = 502.25$$

- Test data-36: throughput=780; packetloss=24; delay=167; jitter=80

$$SD-1 = \sqrt{(0.265 - 1.405)^2 + (0.32 - 0.78)^2 + (90 - 90)^2 + (12.93 - 0)^2} = 12.98$$

$$SD-2 = \sqrt{(0.001 - 1.405)^2 + (0.1 - 0.07)^2 + (30 - 90)^2 + (0.03 - 0)^2} = 60.01$$

$$SD-3 = \sqrt{(0.001 - 1.405)^2 + (0 - 0.07)^2 + (40 - 90)^2 + (4.85 - 0)^2} = 50.25$$

$$SD-35 = \sqrt{(0.671 - 780)^2 + (0.21 - 24)^2 + (40 - 167)^2 + (0.007 - 80)^2} = 794.05$$

$$SD-36 = \sqrt{(0.001 - 780)^2 + (0 - 24)^2 + (20 - 167)^2 + (0.001 - 80)^2} = 798.11$$

The recapitulation of the calculation results using the k-NN algorithm with SD with a total of 36 training data and 85 complete test data as shown in Tables 3 and 4.

3.4. K-nearest neighbor algorithm calculation results

In Table 5, the recapitulation value of the calculation results of the k-NN algorithm uses the square distance (SD) formula, which is a method for finding the closeness of the distance values of two variables. The test data used in this study consisted of 36 (thirty six) test data. The resulting values are then analyzed to compare the proximity of the SD with the existing query distance. Based on the test data results, namely SD to quire distance, smallest distance, status of the k-NN algorithm and category k-NN algorithm.

3.5. Test data result

The results of the recapitulation of test data using the k-NN algorithm with communication data (throughput, packetloss, delay, and jitter) based on the actual category, prediction category and the path used based on the calculation SD results, as shown in Table 6. In Figure 6 a graph of fluctuations in data communication can be seen from the calculation recapitulation using the k-NN algorithm with SD which includes throughput, packet loss, delay and jitter tests. From the results of the prediction category, it can be said that the data communication is very good from path-1, good path-2, moderate path-3, and bad path-4.

Figure 7 shows the parts of the data structure of the transmission network, which consists of a standard switch and a host (computer) with independent processing elements to complete control tasks and data transmission tasks. From the analysis process, it can be obtained the implementation of special routing on the SDN network based on the results of the recommendations and prediction categories used for routing

recommendations, path-1 scenario (very good) which goes through switch 1, switch 2, and switch 4. With a value of throughput=1,334 (kbps), packetloss=0.25%, delay=150 ms, and jitter=75ms.

Table 3. Calculation data uses the k-NN algorithm with SD-1 to SD-6

No	SD					
	SD-1	SD-2	SD-3	SD-4	SD-5	SD-6
1	98.28	431.3	733.81	8.36	346.24	1,336
2	156.52	488.99	754.68	117.6	401.71	1,341
3	145.42	477.94	3,041	108.59	391.26	382.392
4	530,157	596,509	764.65	546,905	522,935	382.368
5	155.41	487.67	754.45	115.22	400.68	1,341
6	156.51	488.98	754.68	117.59	401.73	1,341
7	153.42	485.33	753.95	111.52	369.38	1,341
8	147.65	480.44	750.26	112.83	393.26	1,339
9	182.66	513.87	765.43	134.56	426.32	1,344
10	623.94	422.28	858.34	707.27	471.39	1,481
11	155.91	488.28	754.54	116.44	401.16	1,341
12	134.07	467.01	746.25	103.88	380.47	1,339
13	424.56	488.18	846.06	370.07	448.47	1,396
14	189.79	157.75	703.42	261.53	123.49	1,350
15	145.95	192.55	701.89	212.58	130.37	1,343
16	188.75	159.18	703.44	261.63	135.08	1,349
17	1,740	1,803	1,074	1,734	1,756	427.21
18	1,919	1,975	1,246	1,914	930.18	97
19	1,847	1,903	1,174	1,842	1,857	524.79
20	7,359	7,194	7,215	7,328	7,262	7,149
21	1,266	1,326	602.78	1,261	1,278	2.96
22	783.59	788.57	84.58	804.48	753.98	606.35
23	540,854	578,707	650.01	585,159	514,935	393.629
24	544.18	621.17	178.68	556.77	564.09	794.35
25	2,736	7,620,558	2,048	2,733	2,732	1,405
...
84	193.52	156.08	703.81	266.59	134.37	1,353
85	1,734.	1796	1,066	1,728	1,749	414.11

Table 4. Calculation data uses the k-NN algorithm with SD-7 to SD-36

No	SD				
	SD-7	SD-8	SD-9	SD-10	SD-36
1	197.48	809.21	796.97	704.88	12.98
2	254.79	834.03	805.85	711.75	60.02
3	243.76	829.27	803.54	709.96	50.25
4	527,325	18,183	26,004	18,428	549,404
5	53.52	833.66	805.26	711.41	60.30
6	254.79	834.03	805.84	711.75	60.02
7	251.27	832.94	804.35	710.87	60.46
8	246.23	829.14	803.61	709.61	50
9	279.97	846.15	810.27	715.95	88.72
10	555.09	869.73	1,029	973.34	700
11	254.11	833.82	805.56	711.58	60.03
12	232.84	824.19	802.16	708.68	36.97
13	408.66	892.72	845.42	798.84	468.52
14	100.09	750.32	808.98	737.8	90.24
15	50.78	753.26	797.66	723.92	248.6
16	101.32	750.58	808.94	737.91	289.24
17	1,753	1,052	976.42	1,042	1,734
18	930.74	1,219	1,152	1,221	1,914
19	1,858	1,148	1,079	1,148	1,842
20	7,305	7,161	7,111	7,201	7,426
21	275.56	588.8	496.89	565.99	1,265
22	771.43	62.98	234.37	243.56	809.63
23	530,013	8,631	37,832	733.4	573,537
24	548.23	267.22	275.31	190.06	559.7
25	2,742	2,010	1,963	2,037	2,733
...
84	106.24	750.45	810.24	739.48	803.94
85	1,746	1,045	969.79	1,036	962.02

Table 5. Recapitulation of calculations using k-NN with square distance

Test data	SD to quire distance	Smallest distance	Status k-NN algorithm	Category k-NN algorithm
1	89.45	1	Yes	Good
	106	3	Yes	Good
	98.27	2	Yes	Good
2	192.54	3	Yes	Bad
	159.17	2	Yes	Bad
	157.75	1	Yes	Bad
3	3,041	1	Yes	Currently
	84.58	2	Yes	Good
	178.68	3	Yes	Good
4	88.36	3	Yes	Good
	31.67	1	Yes	Good
	71.67	2	Yes	Good
5	130.36	2	Yes	Bad
	135.08	3	Yes	Bad
	123.49	1	Yes	Bad
6	382.392	2	Yes	Currently
	92.96	1	Yes	Good
	393.629	3	Yes	Currently
7	101.31	3	Yes	Bad
	100.08	2	Yes	Bad
	50.77	1	Yes	Bad
8	18,183	2	Yes	Currently
	62.97	3	Yes	Good
	8,631	1	Yes	Currently
9	234.36	3	Yes	Good
	37,832	2	Yes	Currently
	26,004	1	Yes	Currently
...
35	265	2	Yes	Currently
	365.65	1	Yes	Bad
	278	1	Yes	Bad
36	67.87	3	Yes	Good
	33.54	1	Yes	Bad
	12.34	3	Yes	Good

Table 6. Recapitulation of calculations using k-NN with square distance

No	Throughput (kbps)	Packet loss (%)	Delay (ms)	Jitter (ms)	Actual category	Prediction category	Path
1	1.34	0.02	170	70	Good	Good	Path 2
2	0.03	15	450	250	Bad	Bad	Path 4
3	700	15.91	300	80	Currently	Good	Path 2
4	3.59	15	90	100	Currently	Good	Path 2
5	26	127.4	370	170	Bad	Bad	Path 4
6	1,334	0.25	150	75	Very Good	Very Good	Path 1
7	0.57	20	250	127	Currently	Bad	Path 4
8	750	25	370	130	Currently	Currently	Path 3
9	780	2.5	140	170	Good	Currently	Path 3
...
35	0.029	20	454	252	Bad	Bad	Path 2
36	780	24	167	80	Good	Good	Path 2

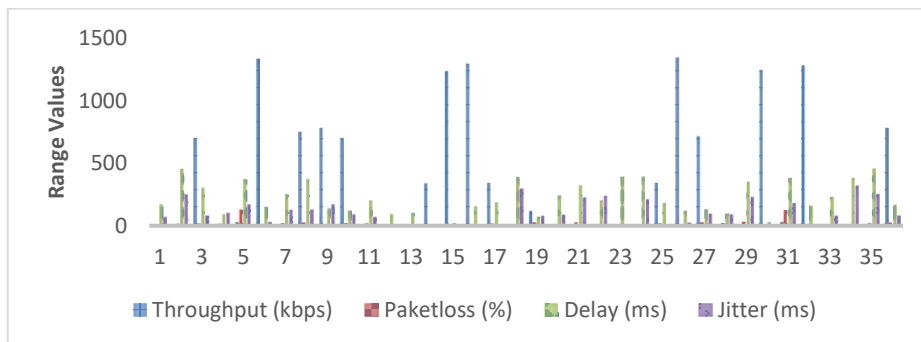


Figure 6. Fluctuating data communication

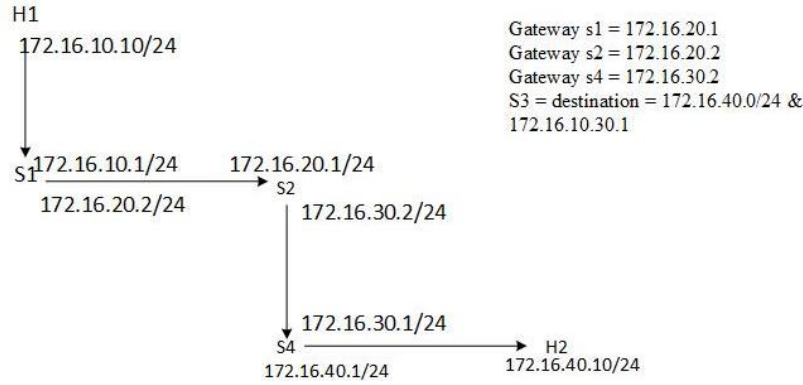


Figure 7. The test results are very good in the path-1 scenario

3.5.1. Confusion metrics

The confusion matrix is also known as the error matrix. Basically the confusion matrix provides information on the comparison of the classification results performed by the system (model) with the actual classification results [37]. The confusion matrix is in the form of a matrix table that describes the performance of the classification model on a series of test data whose actual values are known. The picture below is a confusion matrix with 4 different combinations of predicted values and actual values. Based on the confusion matrix, the calculation using k-NN results prediction values of accuracy, precision and recall as in Table 7.

Table 7. Calculations predict confusion matrix dataset

		Predict				
		Very good	Good	Currently	Bad	
Actual	Very good	15	0	0	0	15
	Good	0	44	0	0	44
	Currently	0	0	11	0	11
	Bad	0	0	0	15	15
		15	44	11	15	85

After getting the predicate value in the confusion matrix, it can be calculated using (2) with results as:

$$Accuracy = \frac{(TP + TN)}{Test\ Data} = \frac{(0.71 + 15) + (0.71 + 44) + (0.71 + 11) + (0.71 + 15)}{85} = \frac{(15.71) + (44.71) + (11.71) + (15.71)}{85} = \frac{87.84}{85} = 72.31\%$$

$$Precision = \frac{TP}{Total\ Predict\ Category} = \frac{(15.71) + (44.71) + (11.71) + (15.71)}{(0.01 + 0.02 + 0.03 + 0.08)} = \frac{87.84}{2.77} = 96.08\%$$

$$Recall = \frac{TP}{(Curently\ Prediction) + (Bad\ Prediction)} = \frac{(15 + 44) + (0.50 + 0.67)}{(0.03 + 0.02)} = \frac{8.00}{0.5} = 87.84\%$$

From the calculations in (2) it can be concluded that the respective predictive values for predictive accuracy are 72.31%, precision are 96.08%, and recall are 87.84%. So, and can be described in Figure 5 the results of calculations using a confusion matrix based on predictive presentations and in Figure 6 the graph of the results of the confusion matrix values based on the criteria. As for the results of the prediction calculation, it can be said that based on the criteria, it is very good as much as 4, good as much as 14, currently as much as 12 and bad as much as 6 as shown in Figure 8.

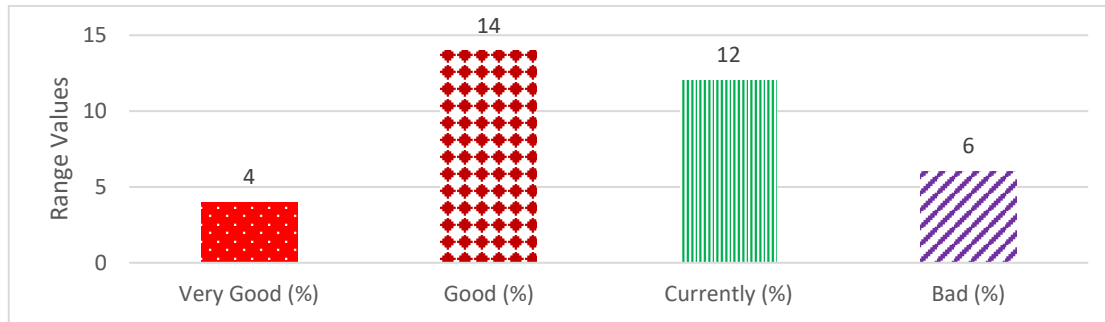


Figure 8. Graph of confusion matrix values based on criteria

4. CONCLUSION

From the results of the analysis of calculations using the k-NN algorithm on communication data, namely the throughput value, jitter value, delay value, and packet loss by finding the predictive value can be used as a recommendation to the admin to determine the best path. From the results of the calculation of the confusion matrix, the reference value is obtained, namely accuracy of 72.31%, precision of 96.08% and recall of 87.84%. So that the selection of the best path from the results of special routing analysis on the SDN network is recommended as follows: i) path-1 (very good) which includes switch 1, switch 2 and switch 4; ii) path-2 (good) which passes through switch 1, switch 3 and switch 4; iii) path-3 (moderate) through switch 1, switch 2, switch 3 and switch 4; and iv) line-4 (bad) trough switches 1-4.

ACKNOWLEDGEMENT

First, we would like to thank the University of Ibn Khaldun Bogor for the funding and support of this research and secondly, a big thank you to Informatics of Engineering, High School of Informatics and Computer Management Bani Saleh, Bekasi for all the knowledge we gained during the research this until the end of the study.




REFERENCES

- [1] Ö. Tonkal, H. Polat, E. Başaran, Z. Cömert, and R. Kocaoğlu, "Machine learning approach equipped with neighbourhood component analysis for DDoS attack detection in software-defined networking," *Electronics*, vol. 10, no. 11, pp. 1–16, May 2021, doi: 10.3390/electronics10111227.
- [2] P. Fondo-Ferreiro, F. Gil-Castineira, F. J. Gonzalez-Castano, and D. Candal-Ventureira, "A software-defined networking solution for interconnecting network functions in service-based architectures," *IEEE Access*, vol. 10, pp. 19905–19916, 2022, doi: 10.1109/ACCESS.2022.3152197.
- [3] A. Malik, R. de Fréin, and B. Aziz, "Rapid restoration techniques for software-defined networks," *Applied Sciences*, vol. 10, no. 10, pp. 1–24, May 2020, doi: 10.3390/app10103411.
- [4] A. Shirvar and B. Goswami, "Performance comparison of software-defined network controllers," in *2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*, Feb. 2021, pp. 1–13, doi: 10.1109/ICAECT49130.2021.9392559.
- [5] H. Babbar and S. Rani, "Performance evaluation of QoS metrics in software defined networking using Ryu controller," in *IOP Conference Series: Materials Science and Engineering*, Jan. 2021, pp. 1–11, doi: 10.1088/1757-899X/1022/1/012024.
- [6] S. Syaifuddin, M. F. Azis, and F. D. S. Sumadi, "Comparison analysis of multipath routing implementation in software defined network," *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control*, vol. 6, no. 2, pp. 141–148, May 2021, doi: 10.22219/kinetik.v6i2.1228.
- [7] R. Chaudhary and N. Kumar, "EnFlow: An energy-efficient fast flow forwarding scheme for software-defined networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 8, pp. 5293–5309, Aug. 2021, doi: 10.1109/TITS.2020.2999134.
- [8] H. M. Noman and M. N. Jasim, "A proposed linear multi-controller architecture to improve the performance of software defined networks," in *Journal of Physics: Conference Series*, Feb. 2021, pp. 1–9, doi: 10.1088/1742-6596/1773/1/012008.
- [9] R. Kunneke, C. Ménard, and J. Groenewege, *Network infrastructures: Technology meets institutions*. Cambridge: Cambridge University Press, 2021, doi: 10.1017/9781108962292.004.
- [10] A. D. Mukhamejanova, E. A. Grabs, K. K. Tumanbayeva, and E. M. Lechshinskaya, "Traffic simulation in the LoRaWAN network," *Bulletin of Electrical Engineering and Informatics*, vol. 11, no. 2, pp. 1117–1125, Apr. 2022, doi: 10.11591/eei.v11i2.3484.
- [11] L. C. Costa *et al.*, "OpenFlow data planes performance evaluation," *Performance Evaluation*, vol. 147, pp. 1–23, May 2021, doi: 10.1016/j.peva.2021.102194.
- [12] O. Romanov, I. Saychenko, A. Marinov, and S. Skolets, "Research of SDN network performance parameters using mininet network emulator," *Information and Telecommunication Sciences*, vol. 12, no. 1, pp. 24–32, Jun. 2021, doi: 10.20535/2411-2976.12021.24-32.
- [13] O. I. Romanov and M. M. Nesterenko, "Telecommunication network technologies evolution," in *Advances in the telecommunications 2019*, 2019, pp. 88–107.
- [14] O. Romanov, M. Nesterenko, N. Fesokha, and V. Mankivskiy, "Evaluation of productivity virtualization technologies of switching equipment telecommunications networks," *Information and Telecommunication Sciences*, vol. 11, no. 1, pp. 53–58,




- Jun. 2020, doi: 10.20535/2411-2976.12020.53-58.
- [15] O. M. A. Alssaheli, Z. Z. Abidin, N. A. Zakaria, and Z. A. Abas, "Implementation of network traffic monitoring using software defined networking Ryu controller," *WSEAS Transactions on Systems and Control*, vol. 16, pp. 270–277, May 2021, doi: 10.37394/23203.2021.16.23.
- [16] D. Jiménez-Grande, S. F. Atashzar, E. Martínez-Valdes, and D. Falla, "Muscle network topology analysis for the classification of chronic neck pain based on EMG biomarkers extracted during walking," *PLOS ONE*, vol. 16, no. 6, pp. 1–17, 2021, doi: 10.1371/journal.pone.0252657.
- [17] L. H. Collantes and A. P. Wibawa, "SDN: A different approach for the design and implementation of converged networks," in *2021 3rd East Indonesia Conference on Computer and Information Technology*, 2021, pp. 450–455, doi: 10.1109/EIConCIT50028.2021.9431937.
- [18] B. Almadani, A. Beg, and A. Mahmoud, "DSF: A distributed SDN control plane framework for the east/west interface," *IEEE Access*, vol. 9, pp. 26735–26754, 2021, doi: 10.1109/ACCESS.2021.3057690.
- [19] S. Wang, K. Gomez, K. Sithampanathan, M. R. Asghar, G. Russello, and P. Zanna, "Mitigating DDoS attacks in SDN-based IoT networks leveraging secure control and data plane algorithm," *Applied Sciences*, vol. 11, no. 3, pp. 1–27, 2021, doi: 10.3390/app11030929.
- [20] S. Yeo, Y. Naing, T. Kim, and S. Oh, "Achieving balanced load distribution with reinforcement learning-based switch migration in distributed SDN controllers," *Electronics*, vol. 10, no. 2, pp. 1–16, Jan. 2021, doi: 10.3390/electronics10020162.
- [21] P. Monika, R. M. Negara, and D. D. Sanjoyo, "Performance analysis of software defined network using intent monitor and reroute method on ONOS controller," *Bulletin of Electrical Engineering and Informatics*, vol. 9, no. 5, pp. 2065–2073, Oct. 2020, doi: 10.11591/eei.v9i5.2413.
- [22] R. Ratnasih, D. Perdana, and Y. G. Bisono, "Performance analysis and automatic prototype aquaponic of system design based on internet of things (IoT) using MQTT protocol," *Jurnal Infotel*, vol. 10, no. 3, pp. 130–137, Aug. 2018, doi: 10.20895/infotel.v10i3.388.
- [23] H. F. Mahdi, M. S. Abood, and M. M. Hamdi, "Performance evaluation for vehicular ad-hoc networks based routing protocols," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 2, pp. 1080–1091, Apr. 2021, doi: 10.11591/eei.v10i2.2943.
- [24] H. M. Noman and M. N. Jasim, "A proposed adaptive least load ratio algorithm to improve resources management in software defined network OpenFlow environment," *Karbala International Journal of Modern Science*, vol. 7, no. 1, pp. 39–47, Mar. 2021, doi: 10.33640/2405-609X.2255.
- [25] J. Ali and B. Roh, "Quality of service improvement with optimal software-defined networking controller and control plane clustering," *Computers, Materials & Continua*, vol. 67, no. 1, pp. 849–875, 2021, doi: 10.32604/cmc.2021.014576.
- [26] D. Cabarkapa and D. Rancic, "Performance analysis of Ryu-POX controller in different tree-based SDN topologies," *Advances in Electrical and Computer Engineering*, vol. 21, no. 3, pp. 31–38, 2021, doi: 10.4316/AECE.2021.03004.
- [27] N. N. Josbert, W. Ping, M. Wei, M. S. A. Muthanna, and A. Rafiq, "A framework for managing dynamic routing in industrial networks driven by software-defined networking technology," *IEEE Access*, vol. 9, pp. 74343–74359, 2021, doi: 10.1109/ACCESS.2021.3079896.
- [28] A. Al-Harbi, A. Bahnasse, F. E. Louhab, and M. Talea, "Towards an efficient resource allocation based on software-defined networking approach," *Computers & Electrical Engineering*, vol. 92, pp. 1–15, Jun. 2021, doi: 10.1016/j.compeleceng.2021.107066.
- [29] S. D. A. Shah, M. A. Gregory, S. Li, R. D. R. Fontes, and L. Hou, "SDN-based service mobility management in MEC-enabled 5G and beyond vehicular networks," *IEEE Internet of Things Journal*, vol. 9, no. 15, pp. 13425–13442, Aug. 2022, doi: 10.1109/JIOT.2022.3142157.
- [30] A. Khan, A. A. Siddiqui, F. Ullah, M. Bilal, M. J. Piran, and H. Song, "VP-CAST: Velocity and position-based broadcast suppression for VANETs," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 18512–18525, Oct. 2022, doi: 10.1109/TITS.2022.3153122.
- [31] A. W. Muhammad, C. F. M. Foozy, and K. M. bin Mohammed, "Multischeme feedforward artificial neural network architecture for DDoS attack detection," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 1, pp. 458–465, Feb. 2021, doi: 10.11591/eei.v10i1.2383.
- [32] A. R. Lubis, M. Lubis, and A.- Khowarizmi, "Optimization of distance formula in k-nearest neighbor method," *Bulletin of Electrical Engineering and Informatics*, vol. 9, no. 1, pp. 326–338, Feb. 2020, doi: 10.11591/eei.v9i1.1464.
- [33] I. Markoulidakis, I. Rallis, I. Georgoulas, G. Kopsiaftis, A. Doulamis, and N. Doulamis, "Multiclass confusion matrix reduction method and its application on net promoter score classification problem," *Technologies*, vol. 9, no. 4, pp. 1–22, Nov. 2021, doi: 10.3390/technologies9040081.
- [34] D. Chicco, N. Tötsch, and G. Jurman, "The Matthews correlation coefficient (MCC) is more reliable than balanced accuracy, bookmaker informedness, and markedness in two-class confusion matrix evaluation," *BioData Mining*, vol. 14, no. 1, pp. 1–22, Dec. 2021, doi: 10.1186/s13040-021-00244-z.
- [35] Casas-Velasco D, Rendon O, Da Fonseca N, "Intelligent Routing Based on Reinforcement Learning for Software-Defined Networking," *IEEE Transactions on Network and Service Management*, vol. 6, no. 1, pp. 870-881, 2021, doi: 10.1109/TNSM.2020.3036911.
- [36] D. Sinaga, F. Agustina, N. A. Setiyanto, S. Suprayogi, and C. Jatmoko, "Classification of bird based on face types using gray level co-occurrence matrix (GLCM) feature extraction based on the k-nearest neighbor (K-NN) algorithm," *Journal of Applied Intelligent System*, vol. 6, no. 2, pp. 111–119, Dec. 2021, doi: 10.33633/jais.v6i2.4627.
- [37] S. Ruuska, W. Hämäläinen, S. Kajava, M. Mughal, P. Matilainen, and J. Mononen, "Evaluation of the confusion matrix method in the validation of an automated system for measuring feeding behaviour of cattle," *Behavioural Processes*, vol. 148, pp. 56–62, Mar. 2018, doi: 10.1016/j.beproc.2018.01.004.

BIOGRAPHIES OF AUTHORS






Ritzkal    graduated from Master of Informatics Engineering (M.Kom), Informatics Engineering Study Program, STMIK Eresha-Jakarta in 2011. He is a permanent lecturer at Ibnu Khaldun University, Bogor-West Java, Informatics Engineering Study Program. Research in the field of network-based computing, information systems, data mining. He can be contacted at email: ritzkal@ft.uika-bogor.ac.id.






Suhadi    graduated from Master of Informatics Engineering (M.Kom), Informatics Engineering Study Program, STMIK Eresha-Jakarta in 2011. Functional Computer Institutions of the Directorate of Licence and Fisheries, Directorate General of Capture Fisheries, Ministry of Marine Affairs and Fisheries of the Republic of Indonesia. In 2016 until now he is active as a Lecturer at the College of Information Management and Computers (STMIK) Bani Saleh Bekasi-West Java, Informatics Engineering Study Program. Research in the fields of software engineering, data mining, intelligent systems and digital forensics. He can be contacted at email: hadims71ndl@gmail.com.






Rizky Amalia    graduated from Bachelor of Engineering (S.T), Informatics Engineering Study Program, Ibn Khaldun University in 2022. Research in the fields of network. She can be contacted at email: risskyamalia2@gmail.com.






Yuggo Afrianto    graduated from Master of Informatics Engineering (M.Kom), Informatics Engineering Study Program, Bogor Agricultural Institute in 2017. He is a permanent lecturer at Ibn Khaldun University, Bogor-West Java, Informatics Engineering Study Program. Research in the fields of net centric computing. He can be contacted at email: yuggo@uika-bogor.ac.id.






Anggra Triawan    graduated from Master of Informatics Engineering (M.Kom), Informatics Engineering Study Program, STMIK Eresha-Jakarta in 2015. He is a permanent lecturer at Binaniaga Indonesia University, Bogor-West Java, Informatics Engineering Study Program. Research in the fields of system information and informatics. He can be contacted at email: anggra@unbin.ac.id.



Syafril    graduated from Master of Management (M.M), Management Study Program, Jenderal Soerdiman University in 2009. Informatics Engineering Study Program. He is a permanent lecturer at Binaniaga Indonesia University, Bogor-West Java, Informatics System Study Program. Research in the fields of system information and informatics. He can be contacted at email: syafril@unbin.ac.id.



Fety Fatimah    graduated from Master of Computer (M.Kom), E-Bussiness Study Program, Nusa Mandiri University in 2011. Informatics Engineering Study Program. He is a permanent lecturer at Ibn Khaldun University, Bogor-West Java, Informatics System Study Program. Research in the fields of system information and informatics. She can be contacted at email: fety.fatimah@ft.uika-bogor.ac.id.