KSII The 17th Asia Pacific International Conference on Information Science and Technology(APIC-IST) 2022.
Copyright ⓒ 2022 KSII

Dataset construction method for network intelligence

SoYeon Lee[1], Jihoon Park[1], Tae-Jun Yoon[2], MinJi Bae[2], Dae-Young Kim[2*]
[1]Department of Software Convergence, Soonchunhyang University, Asan, South Korea
[2]Department of Computer Software Engineering, Soonchunhyang University, Asan, South Korea
[e-mail: {lsy8647, wlgns12www, 20214004, pigi, dyoung.kim}@sch.ac.kr]
*Corresponding author: Dae-Young Kim

Abstract

Recently, as the accuracy of artificial intelligence (AI) increases, the use of AI technology in many applications is increasing explosively. In network applications, AI is applied to various areas to improve service efficiency. Particularly, Researches on network automation and optimization are actively conducted by analyzing various data collected from the network domains. In general, AI follows a data-driven approach that learns various patterns on its own to accomplish a given task based on data. Therefore, a reliable high-quality dataset for learning efficiency has an important role in AI models. In network AI, network data also affects on learning efficiency. Therefore, in this paper, we describe an effective method to collect network state data to achieve network automation. If the collected data is used as an AI training dataset for network state prediction, it is expected that efficient network automation will be achieved in the future.

***Keywords*:** AI, Data collection, Data analysis, Network Automation, Network Intelligence

1. Introduction

Recently, artificial intelligence has been in the spotlight as a tool for solving complex problems in a wide range of fields with the development of big data. As the use of AI technology is explodingly increasing in many technology industry areas, the use of AI technology for various issues is being explored in the network area as well[1]. Typically, as network complexity increases and service requirements become more diversified, autonomous decision making based on artificial intelligence technology by analyzing various data collected from the network, unlike the method that operation managers managed through manual setting and control in the past Depending on the method, research on network intelligence technology that converts it to a fully automated method is active[2]. Such network intelligence requires the definition of control functions to collect observed state data from the network, extract knowledge used for decision-making, and manage network services required by given network resources[3]. In other words, the key to successful network intelligence is to first secure a high-quality dataset and construct and train a high-quality artificial intelligence/machine learning model[4].

Therefore, this paper describes a method of collecting network status data for future network intelligence. With three terminals serving as Client, Server, DB & AP, network status data was collected by generating traffic through the iperf3 command in the Client and Server. It is expected that efficient network intelligence will be achieved in the future if the collected data is processed in the form of a learning data set for AI and input to an artificial intelligence model.

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. 2021R1C1C1013133), and this work was funded by BK21 FOUR (Fostering Outstanding Universities for Research)(No. :5199990914048)

2. System Design

2.1 Architecture

In this paper, a total of 8 types of significant data among various network state data were collected through the structure shown in **Fig. 1**.

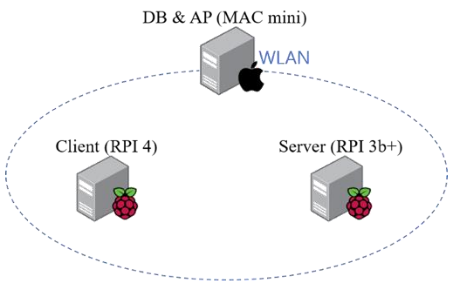


Fig. 1. Network Structure for Network State Data Collection

It consists of two Raspberry Pis to measure the network transmission rate in a continuous streaming environment, a database (DB) that stores the network status data collected from the client, and a MAC mini that simultaneously acts as a wireless AP between the client and the server. These three modules are connected to the same AP. The types of network status data to be collected are SSID, Frequency, BSSID, Signal Level, Datetime RTT, Jitter, and Throughput, and these were collected using Linux utilities such as iwconfig, iperf3[5], and ss(Socket Statistics)[6]. **Table 1** shows information about the data collected using each utility.

Utility	Module	Collected Data
iwconfig	Client	• SSID : Network identifier • BSSID : A network ID that identifies the BSS • Signal Level : AP signal strength (dBm) • Frequency : frequency (Hz)
iperf3	Client, Server	• Throughput : Transfer rate (Mbits/sec)
ss	Server	• RTT : Packet Round trip time (ms) • Jitter : Delay time (ms)

Table. 1. Utilities and other descriptions used to collect network status data

2.2 Data collection Process

It consists of three modules to collect network status data. The overall module structure is shown in **Fig. 2**, which shows the role of each module, the data measured by each utility, and the process of storing the measured data in the database.

To create a continuous streaming environment, traffic was created using the iperf3 utility between the server and the client. The server sends continuous streaming traffic to the client and the client receives it and at the same time collects various information of the AP through the iwconfig command. The server generates traffic through iperf3 every second and measures RTT and Jitter values through the ss command. The final RTT value delivered to the client is the average value of the measured RTT values, and the jitter value is a value obtained by applying the median absolute deviation (MAD) to the measured RTT values. The client receives the RTT and jitter values transmitted from the server and simultaneously calculates the throughput measured through iperf3. The 8 types of network status data collected through this process are converted into json format, delivered to the AP's DB, and stored in chronological order.

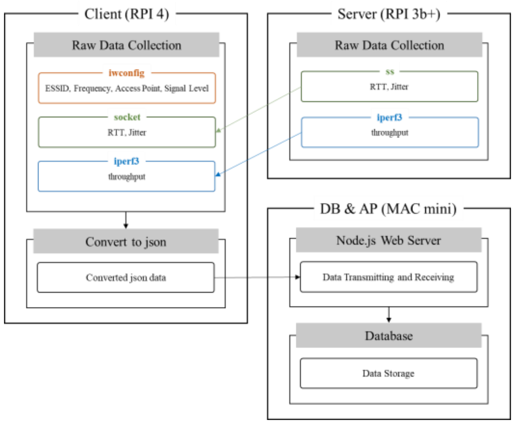
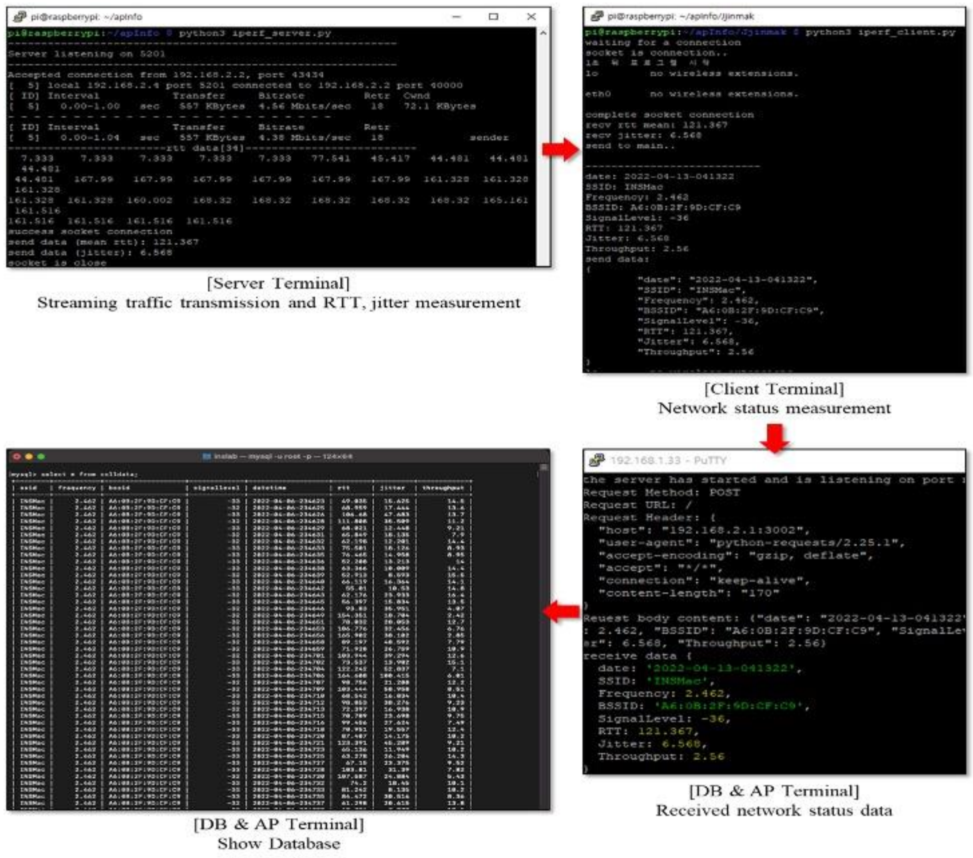


Fig. 2. Network Status Data Collection Overall Module Structure

2.3 Data Collection Results

Fig. 3 is a diagram showing the format of the data extracted for each process in 2.2. In the Server Terminal, streaming traffic is normally generated through iperf3 and RTT values and Jitter values are measured. In the Client Terminal, it can be seen that the 8 measured data values are converted into json format. In DB&AP Terminal, it can be confirmed that network status data is normally received from the client, and finally, it can be confirmed that all data extracted for each process are stored in the database in chronological order.



```
pi@raspberrypi:~/apInfo $ python3 iperf_server.py
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
Accepted connection from 192.168.2.2, port 43434
[  5] local 192.168.2.4 port 5201 connected to 192.168.2.2 port 40000
[ ID] Interval           Transfer     Bitrate         Retr  Cwnd
[  5]   0.00-1.00   sec   557 KBytes  4.56 Mbits/sec   18   72.1 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate         Retr
[  5]   0.00-1.04   sec   557 KBytes  4.38 Mbits/sec   18             sender
---------------------rtt data[34]-----------------------
  7.333    7.333    7.333    7.333    7.333   77.541   45.417   44.481   44.481
  44.481
 44.481   167.99   167.99   167.99   167.99   167.99   167.99  161.328  161.328
 161.328
161.328  161.328  160.002   168.32   168.32   168.32   168.32   168.32  165.161
 161.516
161.516  161.516  161.516  161.516
success socket connection
send data (mean rtt): 121.367
send data (jitter): 6.568
socket is close
```

[Server Terminal]
Streaming traffic transmission and RTT, jitter measurement

```
pi@raspberrypi:~/apInfo/Jjinmak $ python3 iperf_client.py
waiting for a connection
socket is connection..
1초 뒤 프로그램 시작
lo        no wireless extensions.

eth0      no wireless extensions.

complete socket connection
recv rtt mean: 121.367
recv jitter: 6.568
send to main..

----------------------------
date: 2022-04-13-041322
SSID: INSMac
Frequency: 2.462
BSSID: A6:0B:2F:9D:CF:C9
SignalLevel: -36
RTT: 121.367
Jitter: 6.568
Throughput: 2.56
send data:
{
        "date": "2022-04-13-041322",
        "SSID": "INSMac",
        "Frequency": 2.462,
        "BSSID": "A6:0B:2F:9D:CF:C9",
        "SignalLevel": -36,
        "RTT": 121.367,
        "Jitter": 6.568,
        "Throughput": 2.56
}
```

[Client Terminal]
Network status measurement

```
mysql> select * from celldata;
+--------+-----------+-------------------+-------------+--------------------+---------+---------+------------+
| ssid   | frequency | bssid             | signallevel | datetime           | rtt     | jitter  | throughput |
+--------+-----------+-------------------+-------------+--------------------+---------+---------+------------+
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234623  |  49.835 |  16.626 |       14.8 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -32 | 2022-04-06-234625  |  68.959 |  17.444 |       13.6 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234626  |  106.68 |  47.683 |       13.7 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234628  | 111.808 |  36.609 |       11.2 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234629  |  68.021 |  12.448 |       9.21 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -32 | 2022-04-06-234631  |  65.849 |  18.135 |        7.9 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -32 | 2022-04-06-234632  |  62.196 |  12.201 |       14.4 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234633  |  75.581 |  18.126 |       8.93 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234635  |  76.465 |  14.958 |       8.95 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234636  |  62.208 |  13.213 |         14 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234638  |  63.366 |  10.009 |       14.4 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -32 | 2022-04-06-234639  |  62.913 |   8.693 |       15.6 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234640  |  66.119 |  16.364 |       14.1 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234642  |   55.01 |   10.53 |       14.8 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234643  |  62.176 |  25.933 |       16.4 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234645  |  66.397 |  15.834 |       13.6 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -32 | 2022-04-06-234646  |   93.83 |  36.951 |       4.07 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -32 | 2022-04-06-234649  | 154.351 |  10.704 |       2.42 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -32 | 2022-04-06-234651  |  70.832 |  20.053 |       12.7 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -32 | 2022-04-06-234653  | 106.776 |  32.456 |       6.76 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234656  | 165.902 |  30.102 |       2.85 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -32 | 2022-04-06-234658  |  89.197 |  40.592 |       7.79 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -32 | 2022-04-06-234659  |  71.928 |  26.759 |       10.9 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -32 | 2022-04-06-234701  | 103.944 |  39.294 |       12.6 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -30 | 2022-04-06-234702  |  73.537 |  13.982 |       15.1 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234704  | 122.242 |  52.037 |        7.1 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234706  | 164.600 | 100.415 |       6.01 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -32 | 2022-04-06-234707  |  98.756 |  21.200 |       12.2 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -32 | 2022-04-06-234709  | 103.444 |  50.958 |       8.51 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234710  |  68.542 |  16.034 |       10.4 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234712  |  98.853 |  30.276 |       9.22 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -32 | 2022-04-06-234713  |  72.397 |  16.938 |       10.9 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234715  |  70.709 |  23.698 |       9.75 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234716  |  99.456 |  27.624 |       7.49 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234718  |  70.951 |  19.597 |       12.4 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234720  |  87.487 |  14.175 |       10.2 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234721  | 123.391 |  45.289 |       9.21 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234723  |  60.136 |  11.949 |       10.2 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234725  |  63.278 |  20.284 |       14.3 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234727  |   67.15 |  22.375 |       9.52 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234728  |  103.81 |   31.39 |       7.82 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234730  | 107.887 |  24.884 |       5.43 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234732  |    74.2 |   10.46 |       10.1 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -35 | 2022-04-06-234733  |  81.242 |   8.135 |       10.2 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234735  |  86.472 |  30.516 |       8.36 |
| INSMac |     2.462 | A6:0B:2F:9D:CF:C9 |         -33 | 2022-04-06-234737  |  61.298 |  30.615 |       13.8 |
```

[DB & AP Terminal]
Show Database

```
the server has started and is listening on port
Request Method: POST
Request URL: /
Request Header: {
  "host": "192.168.2.1:3002",
  "user-agent": "python-requests/2.25.1",
  "accept-encoding": "gzip, deflate",
  "accept": "*/*",
  "connection": "keep-alive",
  "content-length": "170"
}
Reuest body content: {"date": "2022-04-13-041322"
: 2.462, "BSSID": "A6:0B:2F:9D:CF:C9", "SignalLe
er": 6.568, "Throughput": 2.56}
receive data {
  date: '2022-04-13-041322',
  SSID: 'INSMac',
  Frequency: 2.462,
  BSSID: 'A6:0B:2F:9D:CF:C9',
  SignalLevel: -36,
  RTT: 121.367,
  Jitter: 6.568,
  Throughput: 2.56
}
```

[DB & AP Terminal]
Received network status data

Fig. 3. Network Status Data Storage Result

3. Conclusions

It is essential to collect network state data to implement AI-based network intelligence and optimization, and research to collect and analyze it is actively underway. In this study, a method for collecting network status data using various Linux network utilities was described. Using the data collected as a follow-up study, we plan to conduct a study to predict the future state by learning network patterns over time. If such research continues, it is expected that network intelligence will be achieved in the future.

References

[1] Tae Yeon Kim, Namseok Ko, Sunhee Yang, Sun Me Kim, "Trends in Network and AI Technologies," *Electronics and telecommunications trends*, Vol.35 No.5 pp.1-13, 2020

[2] Shin M.K, Lee S.H, Yi J.H, "Trends of 5G Network Automation and Intelligence Technologies Standardization", *Electronics and telecommunications trends*, Vol.34 No.2, pp.92-100, 2019

[3] National Information Society Agency, "Trends in Reinforcement Learning for Autonomous Networks", 2021

[4] Jooyoung Lee, Seungjae Shin, Seunghyun Yoon, and Taeyeon Kim, "Survey on Artificial Intelligence & Machine Learning Models and Datasets for Network Intelligence, *The Journal of Korean Institute of Communications and Information Sciences*, Vol.47, No.6, pp.625-643, 2022

[5] iPerf3: A TCP UDP and SCTP network bandwidth measurement tool, 11 2019, [online] Available: https://github.com/esnet/iperf

[6] ss, [online] Available : https://man7.org/linux/man-pages/man8/ss.8.html