

General labelled data generator framework for network machine learning

Kwihoon Kim*, Yong-Geun Hong*, Youn-Hee Han**

* ETRI, 218 Gajeong-ro, Yuseong-gu, Daejeon, 34129, Korea

** KOREATECH, 1600, Chungjeol-ro, Byeongcheon-myeon, Dongnam-gu, Cheonan-si, Chungcheongnam-do, 31253, Korea

kwihoon@etri.re.kr, yghong@etri.re.kr, yghan@koreatech.ac.kr

Corresponding Author: yghan@koreatech.ac.kr, Tel: +82 41 560 1486

Abstract— Artificial Intelligence (AI) technology has made remarkable achievements in various fields. Especially, deep learning technology that is the representative technology of AI, showed high accuracy in speech recognition, image recognition, pattern recognition, natural language processing and translation. In addition, there are many interesting research results such as art, literature and music that cannot be distinguished whether it was made by human or AI. In the field of networks, attempts to solve problems that have not been able to be solved or complex problems using AI have started to become a global trend. However, there is a lack of data sets to apply machine learning to the network and it is difficult to know network problem to solve. So far, there have been a lot of efforts to study network machine learning, but there are few studies to make a necessary dataset. In this paper, we introduce basic network machine learning technology and propose a method to easily generate data for network machine learning. Based on the data generation framework proposed in this paper, the results of automatic generation of labelled data and the results of learning and inferencing from the corresponding dataset are also provided.

Keywords— machine learning, data generator, deep learning, network machine learning, supervised learning

I. INTRODUCTION

Recently, the AI has well known for the general public as the core technology of the fourth industrial revolution. In particular, the go game of alpha go and human Lee-sedol shocked many people. It has been a shocking fact that AI technology that was mainly responsible for simple tasks, can be applied even when complex calculation is required, and even better than humans. On the other hand, various researchers have found that applying deep-learning technology that is a representative technology of AI such as voice recognition, image recognition, pattern recognition, natural language processing and translation is much better than using existing machine learning techniques. Beyond applying AI to a simple task, there are a lot of interesting research results that cannot be distinguished whether human or AI made in areas where creativity is required such as art, literature and music. These global trends are also affecting the network sector. There are many attempts to solve problems that have not been able to be solved using AI. However, there

is a difficulty in studying network machine learning in earnest due to the lack of knowledge of dataset and related algorithms for learning. In this paper, we propose a framework to easily generate data for machine learning.

The training data needs to perform data analysis using network mechanical learning. We suggest a framework for creating training data generally for data analysis purposes using mechanical learning. We used to preprocess log-based data or pcap-based data to generate current network training data. Labelled data is essential in the learning method such as supervised learning in particular. However, there is little research on how data processing methods for various mechanical learning analyses are conducted, and studies of machine learning algorithms are mainly based on the existing datasets.

This paper proposes a framework for creating the labelled data to facilitate various mechanical learning analyses. First, we capture the network interface data and add it to the database by adding the label for the user's learning purposes. If network machine learning algorithm researchers choose the size and labelling of data for desired data analysis purposes, this framework will create the labelled data of supervised learning for desired data analysis purposes. Finally, this paper shows the generated labelled data and network machine learning outcomes for the service classification.

Network machine learning algorithm researchers can easily obtain data for machine learning algorithm by selecting machine learning purposes (such as service type, priority, service class) and data format for machine learning algorithms (such as data size, flow flag).

For machine learning research case, it takes more than 70 % time to create data for learning, and approximately 30 % time to research the algorithm for improving accuracy with by the generated data. This paper propose that the machine learning algorithm researcher easily created the trained data to focus on algorithms research.

II. RELATED WORK

Machine learning has many definitions, Mitchell (1997) says, "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance as tasks in T, as measured by P,

improves with experience E.” In other words, machine learning is a computer program that learns from E (experience) to solve T (problem) maximizing P (performance). It is easier to understand machine running by comparing it with traditional program methods.

In traditional programs, input data and coded program are input to computer and output produces the result data. On the other hand, for machine learning programs, input data and output results are input to a computer and output produces the desired program. As we have seen so far, machine learning is a field of research that computers learn itself without human’s programming. This research makes algorithms to learn and predict from data. With complex mathematics and algorithms, you can discover information and patterns hidden in large data sets through powerful computing.

Machine learning can be categorized into three types as follows.

- Supervised learning: It learns the mapping relation between input and target output. There are a discrete classification of the target output and a continuous regression of the target output.
- Unsupervised learning: There is only input value and there is no target output. Clustering is used to classify input data into K groups. This includes association analysis that finds the correlation between the data.
- Reinforcement learning: It is a decision process that simulates human judgment process. When an action is taken in a specific state, the following judgment is made according to the compensation system.

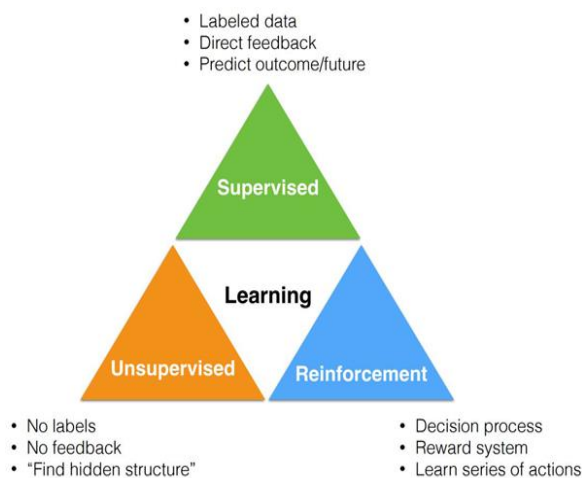


Figure 1. Machine learning category

Machine learning is characterized by the fact that computers are programmed through learning unlike those that people program directly. Currently, it performs much better than conventional people programming in image recognition, speech recognition, and natural language processing [1]-[2]. With the growth of cloud computing and big data in the network sector, more and more complicated problems arise. The network needs to support various devices, various protocols and various applications optimally. However, it is

difficult to solve this problem with the conventional method. Therefore, it is necessary to learn intelligently complex network environment through machine learning to minimize human behavior and to adjust the protocol automatically [3]-[5]. DDoS attacks, network management, self-organization and traffic classification are some of the examples of applying machine learning in the network. Supervised learning includes “link adaptation in wireless networks”, unsupervised learning includes “automatic traffic classification” and reinforcement learning includes “self-organization networks in heterogeneous networks” [6].

Machine learning on the network is still at an early stage. Brocade's Dave Meyer said it is still in the new growth phase of applying machine learning to the network, but is expected to come [7]-[8].

III. PROPOSED GENERAL LABELLED DATA GENERATOR FRAMEWORK

A. Framework concept & system configuration

Figure 2 shows the concept of the General Labelled Data Generator (GLDG) framework. The main requirements for designing the framework are as follows.

- GLDG framework should be scalable so that they can consist of multiple devices.
- GLDG framework should support various operating systems of various devices (mac OS, window, ubuntu, android, etc.)
- Data generated from multiple devices should be stored on the consolidation server.
- Network data capture should be possible in real time through the network interface.
- The researcher should be able to freely select input data and output label data for machine learning.
- The algorithm should be configured to handle various labelled data.
- Input and output data for machine learning should be stored in a database for easy recycling.

To meet the above requirements, the concept of GLDG framework is designed as follows. The GLDG consists of four modules as shown figure 2.

- Network interface processing module: It handles the wired and wireless network interface.
- Labelled data decision controller module: It control to make the labelled data for the selected data type for machine learning researcher.
- Required labelled data input processing module: It allows the machine learning researcher to easily select the required labelled data type.
- Database output processing module: It stores the input data and the labelled output data selected by the machine learning researcher for each flow, packet and size in the DB.

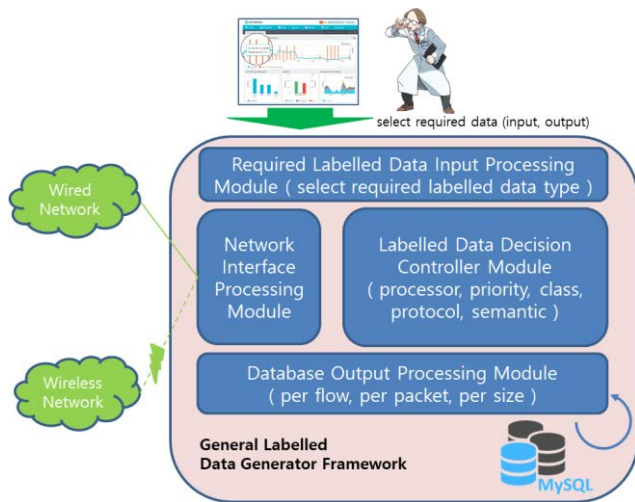


Figure 2. Proposed general labelled data generator framework concept

GLDG can be performed on devices connected to various wired and wireless networks and data acquired from scalable connected devices can be stored on the integration server. The following is the system configuration of the representative GLDG framework.

- Master GLDG: It includes database system and cluster file system.
- Slave GLDG: It includes wired and wireless network connection and various OS support.
- Preprocessor: It includes data preprocessing for machine learning.
- Machine learning (learner): It includes learning processor for machine learning.
- Machine learning (inferencer): It includes inferencing processor for machine learning.



Figure 3. The system configuration of the GLDG framework

B. Total data generation procedure

Data generation, data preprocessing and ML learning (or inference) are performed entirely. The following six steps are performed.

- 1) To enter the desired labelled data.
 - Desired output label: processor, priority, class, protocol, semantic
 - Desired input data unit: per flow, per packet, per size
- 2) To split by input size from the start of packet capture.
- 3) To save the segmented packet data to database.
- 4) To create output data labels for input data.
- 5) To perform data preprocessing.
 - Data preprocessing the label by separator in the first preprocessor
 - Data preprocessing for machine learning (CNN, RNN) in the second preprocessor
- 6) To execute ML learning (or inference).

C. Labelled data collection method

After executing the network data collector, it stores information of the collector and collected packet and flow information in DB. In order to define the flow of data collected in real-time, 5-tuple (source IP, destination IP, source port, destination port, protocol) are defined as the same flow when the same packets occur within 3600 seconds after connection establishment. Although the 5-tuple is the same, it is newly defined as another flow if the packet is generated after 3600 seconds. The procedure of the whole GLDG based on the flow definition is shown in the figure 4.

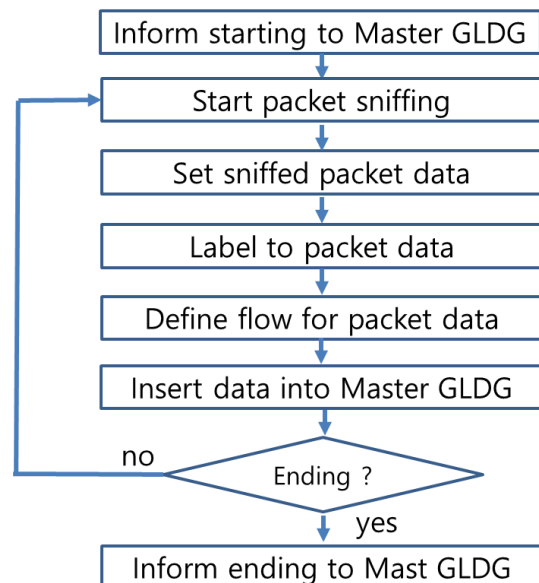


Figure 4. Network data collection procedure

It selects a label to store in the database as follows.

- In case of processor, the process name can be obtained differently for each OS. Therefore representative process name is required.
- Priority, class, etc. can be created by using the representative names of processes.
- Per flow and per packet can use the definition of flows.
- Per protocol can use the protocol number.

- Per semantic can be implemented so that the user can arbitrarily adjust above combination.

D. Database structure for requested data

The database structure to store for each request data is as follows.

- To create the “data_information” table (to store collected data)

```
CREATE TABLE `data_information` (
  `data_id` int(32) unsigned NOT NULL AUTO_INCREMENT,
  `timestamp` int(32) DEFAULT NULL,
  `src_ip` varchar(20) DEFAULT NULL,
  `dst_ip` varchar(20) DEFAULT NULL,
  `src_port` int(16) DEFAULT NULL,
  `dst_port` int(16) DEFAULT NULL,
  `protocol_no` varchar(45) DEFAULT NULL,
  `data_process` varchar(100) DEFAULT NULL,
  `data_payload` blob,
  `data_payload_size` int(16) unsigned DEFAULT NULL,
  `data_stored_time` datetime(6) DEFAULT NULL,
  `data_generator_id` varchar(36) DEFAULT NULL,
  `data_packet_size` int(11) DEFAULT NULL,
  PRIMARY KEY (`data_id`),
  UNIQUE KEY `data_id_UNIQUE` (`data_id`)
);
```

Figure 5. Database structure for request data

IV. SIMULATION RESULTS

In this chapter, we show the result of data generation and the learning result by implementing the prototype system.

A. Software experimental environment

The software required for the system is as follows.

- Windows 10, (Ubuntu 16.04)
- Mysql 5.7.18
- nmap 0.10 release 10
- Python 3.5.3
- Microsoft Visual C++ 2015 build tools
- Wireshark 2.4.1
- Tensorflow 1.3

B. Data Collection and Labelling Results

Prior to entering the purification process, the largest nine number of labels were collected based on the flow number. The nine process names are “slack.exe”, “ASDSvc.exe”, “NaverAdminAPISvc.exe”, “KakaoTalk.exe”, “iexplore.exe”, “LavasoTtpService.exe”, “ServiceHostApp.exe”, “Skype.exe” and “svchost.exe”. Only those flows labeled with these nine labels are selected and only the application layer payload of the internal packet of the flow is filtered and extracted.

Eight application layer payload data files were created based on the second data header of Table 1 extracted payload data. Then, we consolidated the two kinds of traffic data using the same “ServiceHostApp.exe” and “svchost.exe” into “svchost.exe”. Finally, the dataset labels were determined as (1)“slack”, (2)“ASDSvc”, (3)“KakaoTalk”, (4)“iexplore”, (5)“Skype”. Through the integration process, we created five '(process)_payload_data' files.

TABLE 1. FILE DATA HEADER

File name	Data header
all_payload_data	flow_id#start_time#end_time#local_ip#remote_ip#local_port#remote_port#transport_protocol#operating_system#process_name#labels#-#-#
(process)_payload_data	flow_id#start_time#end_time#Payload#-#-#

The data file has the header information for each flow, and payload is separated by using '#' for each packet. That is, each file is a collection of flows with the same label, only the payload of packets and the payload is separated by packet. Payload is classified by packet and is a character string of 4 bits in size. Table 2 shows statistical information of completed '(process)_payload_data' files.

C. Preprocessor data creation

Using the completed payload data, a data set necessary for deep learning is produced. The data set is refined into data suitable for Convolution Neural Networks (CNN) that are the most famous learning models of deep learning. CNN is a deep learning technology that is used most often for image learning and classification. CNN will convert the network packets into image form and then proceed with learning and classification per packet. CNN's learning model data set has 700,000 'train data' and 'train label', 200,000 'validation data' and 'validation label', and 100,000 'test data' and 'test label'. Train, validation and test data are processed by converting the packet payload of the 5 application payload data files described above into an image. One data format is a two-dimensional array with the size of $2^n \times 2^n$, one pixel has a size of 2ⁿ bits, and the value of the character is replaced with a floating-point value.

Train, validation and test have labels for 5 applications so they are represented as a one-hot vector of length 5. A one-hot vector label is a vector with a value of only one element. A value of 1 can be defined as a label indicating the application name.

D. ML learning and inference results

The result of learning CNN model with dataset of independent packet image was not bad and showed better results when we increased packet image size.

The reason for this is that the payload of the processor is exchanged with the specific protocol used by the application. Due to the nature of the protocol, the payload of the processor is independent of the format, semantically independent and the process of communication is clear and distinguishable. Considering this, it can be interpreted that the protocol header data of payload can be applied to CNN learning more when packet image data size is increased.

V. CONCLUSIONS

In this paper, we have dealt with the definition of basic network machine learning techniques and data generators that can easily generate data so that researchers can concentrate on machine learning algorithms. We found that network machine learning technology is an early stage in terms of industry and standardization and we have found that we are trying to utilize machine learning technology in network area of various fields as research trends. At present, when we see the image recognition contest such as AlphaGo Zero or ImageNet in the media, we promote that the machine learning is better than the human being [9]-[10]. However, until now, machine learning has been superior to humans in limited areas and there are still many problems to be solved. Nevertheless, there is a reason why AI is becoming the core technology of the 4th industrial revolution with the global trend [11]. This is because the impact of applying AI is large. Currently, we are trying to apply machine learning to the early stage in the network field, but in the near future, we will be able to solve the complex problem faster and more accurately through machine learning. We expect to be able to solve various network machine learning problems by using the data generator proposed in this paper.

ACKNOWLEDGMENT

This work was supported by the National Research Council of Science and Technology (NST) grant by the Korea government (MSIP) (No.CRC-15-05-ETRI).

REFERENCES

- [1] M. Alsheikh, S. Lin, D. Niyato and H. Tan, "Machine Learning in Wireless Sensor Networks: Algorithms, Strategies, and Applications," *IEEE Commun. Sur. & Tut.*, vol. 16, No. 4, pp. 1996-2018, 2014.
- [2] M. A. Wijaya, K. Fukawa and H. Suzuki, "Intercell-Interference Cancellation and Neural Network Transmit Power Optimization for MIMO Channels," *IEEE Vehicular Technology Conference Fall*, Sep. 2015.
- [3] A. Mestres, etc, "Knowledge-Defined Networking," *ACM SIGCOMM Computer Communication Review.*, 2016.
- [4] G. Stampa, etc, "A Deep-Reinforcement Learning Approach for Software-Defined Networking Routing Optimization," *Arxiv.*, 2017.
- [5] N. Kato, etc, "The Deep Learning Vision for Heterogeneous Network Traffic Control: Proposal, Challenges, and Future Perspective," *IEEE Wireless Communication*, 2017.

- [6] L. Liu, etc, "Deep Learning Based Optimization in Wireless Network," *IEEE ICC 2017*, 2017.
- [7] L. Mekinda and L. Muscariello, "Supervised Machine Learning-Based Routing for Named Data Networking," *2016 IEEE GLOBELOCOM*, vol.20, pp.1-6, Dec. 2016.
- [8] J. Joung, "Machine Learning-based Antenna Selection in Wireless Communications," *IEEE Commun. Letters*, vol. 20, pp. 2241 - 2244, Nov. 2016.
- [9] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770-778, 2016.
- [10] J. Deng, W. Dong, R. Socher, L. Li, K. Li and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.
- [11] E. Park, W. Liu, O. Russakovsky, J. Deng, L. Fei-Fei and A. Berg, "ILSVRC-2017," URL <http://www.image-net.org/challenges/LSVRC/2017/>, 2017.



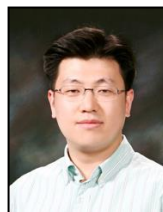
intelligent service.

Kwihoon Kim studied in KAIST, M.S. degree and Ph.D. degree in 2000 and 2013, respectively. He worked in LG DACOM 2000~2005 and is a research engineer in ETRI since 2005. He is a principal research engineer of intelligent IoT networking research team, ETRI now. He is an editor and rapporteur of ITU-T SG11 since 2006. His interested fields are Fog/edge computing, Internet of Things, 5G/IMT2020, deep learning, machine learning, reinforcement learning, GAN and knowledge-converged



network intelligence.

Yong-Geun Hong is a director of the Electronics and Telecommunications Research Institute (ETRI). He is a project leader of IoT Network and Intelligent Network R&D projects in ETRI. He received his B.S., M.S. and Ph.D in computer engineering from the Kyoungpook National University, Daegu, Korea. He is also working for the IoT related standardization at IETF (Internet Engineering Task Force) and ITU-T. His research interests include IPv6, Internet Mobility, M2M/IoT, and



Youn-Hee Han received his B.S. degree in Mathematics from Korea University, Seoul, Korea, in 1996. He received his M.S. and Ph.D. degrees in Computer Science and Engineering from Korea University in 1998 and 2002, respectively. From March 2002 to February 2006, he was a senior researcher in the Next Generation Network Group of Samsung Advanced Institute of Technology. Since March 2006, he has been a Professor in the School of Computer Science and Engineering at Korea University of Technology and Education, Cheonan, Korea. His primary research interests include theory and application of mobile computing, including protocol design and performance analysis. Since 2002, his activities have focused on Internet host mobility, sensor mobility, media independent handover, and cross-layer optimization for efficient mobility support on IEEE 802/LTE wireless networks. Recently, his research focus has been moved to social network analysis and deep learning application to computer networks. He has also made several contributions in IETF and IEEE standardization about IPv6 and mobile IP technology.