Lightweight-Heavyweight Hybrid Approach for Application Traffic Classification

Min-Seong Lee
Computer Information and Science
Korea University
Sejong, Korea
min0764@korea.ac.kr

Jee-Tae Park
Computer Information and Science
Korea University
Sejong, Korea
pjj5846@korea.ac.kr

Ui-Jun Baek
Computer Information and Science
Korea University
Sejong, Korea
pb1069@korea.ac.kr

Chang-Yui Shin
Defense Agency for Technology and Quiality
Daejeon, Korea
realmine@korea.ac.kr

Jeong-Woo Choi
Computer Information and Science
Korea University
Sejong, Korea
choigoya97@korea.ac.kr

Myung-Sup Kim
Computer Information and Science
Korea University
Sejong, Korea
tmskim@korea.ac.kr

***Abstract*— Traffic usage is increasing due to the continuous evolution of networks. As various applications for user convenience appear, various application traffic is also generated. Classifying application traffic is essential to manage high-capacity traffic according to various application traffic and usage. Until now, various studies have been conducted to classify application traffic, but most of them are studies that increase classification accuracy according to the development of deep learning. As environments with large amounts of traffic increase, research on the fast classification speed of application traffic is also needed in terms of efficient network management and network security. In this paper, we propose a system that applies thresholds and improves classification speed using two CNN algorithms. As a result of applying the proposed method, the accuracy increased by 0.06% and the processing speed increased by 0.11 seconds.**

Keywords— traffic classification, traffic identification, convolutional neural network, DL-based classification

I. INTRODUCTION

Due to the continuous advancement of networks, the usage of traffic is increasing. With the emergence of various applications for user convenience, the application traffic generated by these programs is becoming more diverse. In this environment, research is being conducted to achieve efficient network operation and management. To manage high-volume traffic, it is essential to study the classification of application traffic. As the volume of traffic continues to grow, research is being conducted to improve network management, network security, and quality of service (QoS) for users. In the field of application traffic classification, deep learning has been applied to increase classification accuracy, and there have been studies that demonstrate good classification performance. However, there is also a need for research that enables real-time, fast classification of application traffic to manage the large-scale traffic that occurs in practice.[1,2]

Common methods for classifying application traffic include port-based, payload signature-based, and statistical information-based approaches. However, due to the occurrence of encrypted traffic and limitations of existing methods, new techniques for traffic classification have become necessary. Deep learning-based traffic classification methods have been employed to overcome these limitations. CNN-based classification methods are commonly used in deep learning-based traffic classification and have shown high accuracy.[3] In the field of computer vision and image recognition, research is being conducted not only on classification-related studies but also on lightweight models for real-life applications. In the domain of application traffic classification, research related to processing speed for classifying large-scale traffic in real-world environments is also required. In this paper, we propose a method to improve the overall classification processing speed by using two CNN models.[4]

In Chapter 2, related studies are mentioned, and in Chapter 3, the proposed method is described. In Chapter 4, the experimental results are mentioned, and the paper is concluded with the conclusion in Chapter 5.

II. RELATED WORK

A. Deep Learning based Traffic Classification

Deep learning has evolved from machine learning and is being applied in various research fields. Various deep learning models are being studied in the fields of computer vision and image recognition, and successful models have shown good performance in various domains. In the field of application traffic classification, there is research that applies CNN to classify encrypted traffic.[5] Another study involves training traffic data and applying it to a deep learning model, extracting traffic features using SAE (Stacked Autoencoder), and classifying them by applying the extracted data to a CNN model.[6]

B. Study for Traffic Classification Speed

There are research papers related to the speed of application traffic classification that compare speeds through methods such as improving classification speed through parallel processing of traffic classification and classifying real-time traffic. There is a study that applies SVM (Support Vector Machine) in an FPGA (Field Programmable Gate

This work was supported in part by the Technology Innovation Program grant funded By the Ministry of Trade, Industry & Energy (MOTIE, Korea) and the Korea Evaluation Institute of Industrial Technology (KEIT) (No. 20008902, Development of SaaS SW Management Platform based on 5Channel Discovery technology for IT Cost Saving) and "Regional Innovation Strategy (RIS)" through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(MOE)(2021RIS-004).

Array) environment for real-time processing of traffic at high speed.[7] Additionally, there is research that applies parallel traffic classification for traffic classification to increase training speed and classification speed.[8]

III. Proposed Method

A. Architecture

This paragraph describes the overall structure of the proposed method. The proposed method utilizes two models sequentially. As a condition for using the two models, the first model used should be lighter and have faster classification speed than the second model, while the second model should have higher accuracy than the first model. Based on the training data, training models for the two models are generated during the learning process, and it is necessary to verify whether they meet the conditions of the proposed system through validation results. Figure 1 illustrates the overall structure of the proposed method.

During the generation of the training model in the first model, validation results are produced. The validation results represent the numerical classification results of each data for the classes. A threshold is applied to the numerical results for each class, and the reliability is measured based on the applied threshold. When a threshold is set, data that does not meet the threshold is not classified by the first model and is passed on to the second model. In the second model, only the remaining data that was not processed by the first model is classified, reducing the overall processing time for traffic classifica

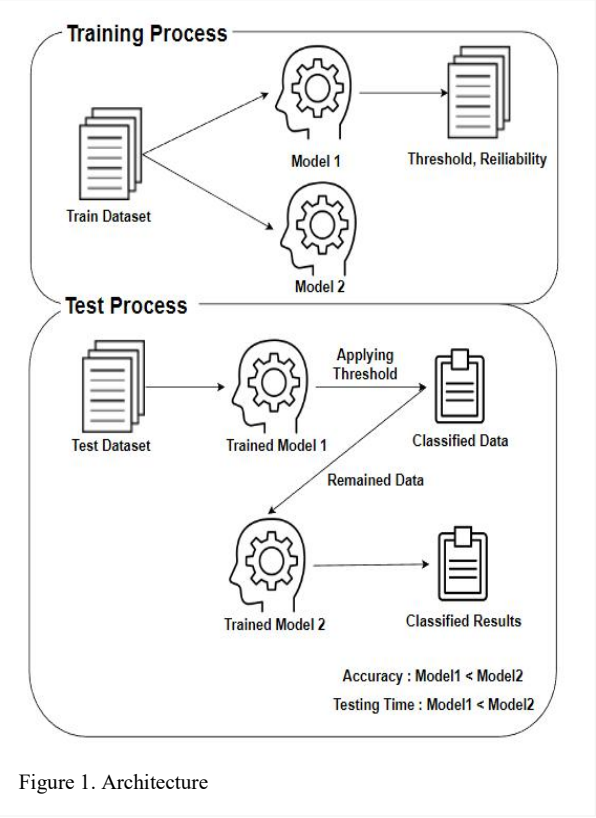

Figure 1. Architecture

B. Thresholds and Reliability

In this paragraph, definitions of the previously mentioned threshold and reliability are provided. In a multi-classification model, the classification results for each feature are represented numerically. These numerical values play a role in determining the class. The threshold is the value applied to the numerical results that represent the classification results for the classes. Data that satisfies the threshold is classified by the first model, while data that does not satisfy the threshold is classified by the second model. Reliability refers to the ratio of correct answers among the data that satisfies the threshold when the threshold is applied. Since the classification results from the first model affect the overall accuracy, it is important to be able to classify a sufficient amount of data even when applying the threshold. A smaller threshold value increases the dependency on the first model, while a higher threshold value increases the dependency on the second model. Therefore, setting an appropriate threshold is crucial.

C. Without Softmax in CNN

In most deep learning models, the output layer of the multi-class classification model uses the Softmax function to normalize the resulting value. When the Softmax function is used in the proposed method, the value for the verification result is not evenly distributed, and tends to converge to a value between 0.9 and 1 to classify the class. Therefore, the Softmax function is removed from the output layer of the CNN model in the first model so that the threshold can be applied.

IV. Experimental Results

A. Datasets

The dataset used in the experiment is ISCXVPN2016. This dataset consists of encrypted application traffic and is commonly used as a public dataset in the field of application traffic classification. Table 1 provides information about the six classes and their content in the ISCXVPN2016 dataset.

In the experiment, classification is performed for six categories: CHAT, Email, File_Transfer, P2P, Streaming, and VOIP. Flows are extracted from the dataset, and the payload of five packets within each flow is used. The length of each feature is standardized to 784 bytes. If a feature is longer than 784 bytes, it is truncated, and if it is shorter than 784 bytes, it is padded with zeros for the experiment. The ratio of data used for training and testing follows an 8:2 ratio from the entire dataset. Table 2 shows the number of training and testing data for each of the six classes.

Table 1. Information of ISCXVPN2016 Datasets

Traffic	Content
CHAT	ICQ, AIM, Skype, Facebook and Hangouts
Email	SMPTS, POP3S, and IMAPS
File_Transfer	Skype, FTPS and SFTP using Filezilla and an external service
P2P	uTorrent and Transmission (Bittorrent)
STREAMNG	Vimeo and Youtube
VOIP	Facebook, Skype and Hangouts voice calls(1h duration)

Table 2. Training and Test Datasets

	Train	Test	Total
CHAT	1,931	483	2,414
Email	1,426	356	1,782
File_Transfer	5,962	1,490	7,452
P2P	290	73	363
Streaming	635	159	794
VoIP	13,304	3,327	16,631
Total	23,548	5,888	29,436

B. Train Process in Model1, Model2

Before the experiment, training is conducted for Model 1 and Model 2. Both models utilize CNN and have different configurations of convolution layers and filter sizes. Table 3 provides information about the two models used in the experiment, including accuracy and classification time when using each model individually. Model 1 employs two convolution layers with a filter size of 64, resulting in a total of 3,091,014 parameters. Model 2, on the other hand, uses four convolution layers with a filter size of 128, resulting in a total of 12,391,814 parameters. Both models are trained for 500 epochs. Model 1 achieves an accuracy of 92.74% with a classification time of 0.52 seconds, while Model 2 achieves an accuracy of 94.09% with a classification time of 0.91 seconds.

Table 3. Information of Models

	Model 1(CNN)	Model 2(CNN)
Conv Layer	2	4
Filter Size	64	128
Parameters	3,091,014	12,391,814
Accuracy	92.74%	94.09%
Testing Time	0.52sec	0.91sec
Epoch	500	

C. Softmax in CNN

To apply a threshold to the output of the CNN model, the Softmax function is removed. When the Softmax function is applied, the class classification results are represented as values between 0 and 1. However, most of the class classification results tend to converge above 0.9, making it difficult to apply the threshold proposed in this method. Therefore, in order to apply the threshold, the Softmax function is removed, and the experiment is conducted accordingly. When Softmax is removed, the class classification results range from a maximum of 1.42 to a minimum of -0.45 in probability for class classification. This

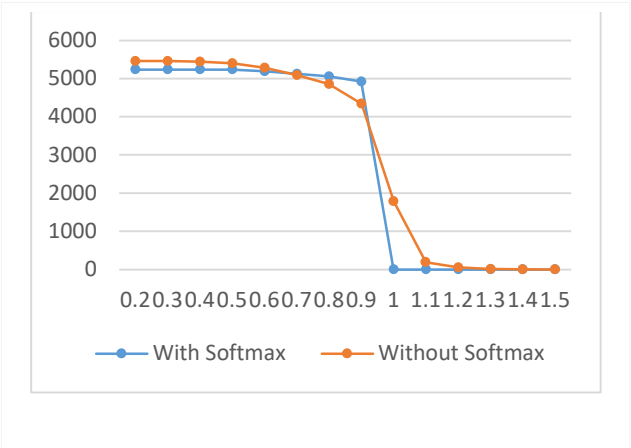

Figure 1. CNN Results with Softmax and without Softmax

distribution can be utilized to apply the threshold. Figure 2 compares the number of data samples satisfying the threshold when Softmax is used versus when it is not used.

D. Model1 Test with Threshold

Here are the test results for Model 1. The classification accuracy is 92.74%, and the total processing time is 0.52 seconds. When the threshold is set low, the classification relies heavily on Model 1 for all the data. Setting the threshold too high would result in all the data being passed to Model 2, increasing the dependency on Model 2. Therefore, it is necessary to apply an appropriate threshold. Table 4 represents the number of data samples satisfying the threshold, their reliability, and the data classification ratio when using Model 1.

Table 4. Results of Model 1 Applying Thresholds

Threshold	P	TP	R	CDR
0.2	5888	5461	92.74%	92.74%
0.3	5884	5460	92.79%	92.73%
0.4	5858	5444	92.93%	92.45%
0.5	5786	5401	93.34%	91.72%
0.6	5585	5285	94.62%	89.75%
0.7	5278	5090	96.43%	86.44%
0.8	4976	4850	97.46%	82.37%
0.9	4408	4345	98.57%	73.79%
1.0	1795	1781	99.22%	30.24%
1.1	190	187	98.42%	3.17%
1.2	54	51	94.44%	0.86%
1.3	13	12	92.30%	0.20%
1.4	1	1	100%	0.01%
1.5	0	0	0%	0.00%

[a.] P: Number of Predict Data Applying Threshold

[b.] TP: Number of True Positive out of P

[c.] R : Reiliability

[d.] CDR : Classified Data Rate

Table 5. Experimental Results of Proposed Method

Threshold	Model1 CDR	Model2 CDR	Model2 Test Time	Total Test Time	Total Accuracy
0.2(Model 1)	92.74%	0.00%	0.00sec	0.52sec	92.74%
0.3	92.73%	0.03%	0.06sec	0.58sec	92.76%
0.4	92.45%	0.32%	0.06sec	0.58sec	92.78%
0.5	91.72%	1.00%	0.07sec	0.59sec	92.73%
0.6	89.75%	3.37%	0.10sec	0.62sec	93.13%
0.7	86.44%	7.32%	0.15sec	0.67sec	93.76%
0.8	82.37%	11.54%	0.19sec	0.71sec	93.91%
0.9	**73.79%**	**20.36%**	**0.28sec**	**0.80sec**	**94.15%**
1.0	30.24%	63.85%	0.63sec	1.15sec	94.10%
1.1	3.17%	90.93%	0.90sec	1.42sec	94.10%
1.2	0.86%	93.22%	0.91sec	1.43sec	94.09%
1.3	0.20%	93.22%	0.91sec	1.43sec	94.09%
1.4	0.01%	94.07%	0.92sec	1.44sec	94.09%
1.5(Model 2)	0.00%	94.09%	0.92sec	0.92sec	94.09%

E. Model1 + Model2

This paragraph presents the experimental results when applying the overall threshold and using the two models sequentially. When the threshold is set to the lowest value, it yields the same result as using Model 1 as a standalone model. Conversely, when the threshold is set to the highest value, it becomes equivalent to using Model 2 as a standalone model. When the threshold is set to 0.9, the overall accuracy is 94.15%, and the classification processing time is 0.8 seconds. This achieves a higher accuracy of 0.06% compared to using Model 2 as a standalone model, with a reduction in processing time by 0.11 seconds.

F. Proposed Method

In this section, the experimental results are shown when the two models are sequentially used by applying the overall threshold. Table 5 shows the experimental results when using the proposed method. When the threshold is the lowest, the result of using Model 1 as a single model is the same, and when the threshold is the highest, the result is the same as the result of using Model 2 as a single model. When a threshold of 0.9 was used, the overall accuracy was 94.15%, and the classification processing time was 0.8 seconds. This is 0.06% higher in accuracy than when Model 2 is used as a single model, and the processing time is reduced by 0.11 seconds. Figure 3 shows the accuracy and time of the proposed method for each threshold. The higher the threshold, the higher the overall accuracy. After the critical value of 0.9, it can be seen that the accuracy is uniformly distributed. Among them, the best performance is shown at the critical value of 0.9 where the total time is the shortest.

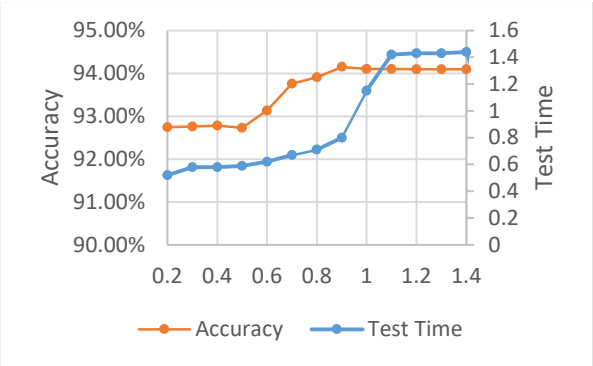

Figure 2. Accuracy and Test Time of Proposed Method

V. Conclusion

In this paper, we proposed a system that improves classification speed in classifying application traffic by sequentially using two CNN models. In order to sequentially use the two CNN models, a threshold was applied to the class classification result in the first model, and the Softmax function was removed to apply the threshold. As a result of the experiment using the models sequentially by applying the threshold, when the two models were used sequentially rather than using the CNN model with higher accuracy, the classification accuracy increased and the time to process the entire classification decreased.

References

[1] M.-S. Kim, Y. J. Won, and J. W.-K. Hong, "Application-level traffic monitoring and an analysis on IP networks," ETRI J., vol. 27, pp. 22-42, 2005.

[2] B. Park, Y. Won, J. Chung, M. S. Kim, and J. W. K. Hong, "Fine-grained traffic classification based on functional separation," Int. J. Network Management, vol. 23, pp. 350-381, Sept. 2013.

[3] Z. Zou, J. Ge, H. Zheng, Y. Wu, C. Han and Z. Yao, "Encrypted Traffic Classification with a Convolutional Long Short-Term Memory Neural Network," 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), 2018, pp. 329-334, doi: 10.1109/HPCC/SmartCity/DSS.2018.00074.

[4] Afeez Ajani Afuwape, Ying Xu, Joseph Henry Anajemba, Gautam Srivastava, Performance evaluation of secured network traffic classification using a machine learning approach, Computer Standards & Interfaces, Volume 78, 2021, 103545, ISSN 0920-5489,https://doi.org/10.1016/j.csi.2021.103545.

[5] Z. Han et al., "An Effective Encrypted Traffic Classification Method Based on Pruning Convolutional Neural Networks for Cloud Platform," 2021 2nd International Conference on Electronics, Communications and Information Technology (CECIT), 2021, pp. 206-211, doi: 10.1109/CECIT53797.2021.00043.

[6] Lotfollahi, M., Jafari Siavoshani, M., Shirali Hossein Zade, R. et al. Deep packet: a novel approach for encrypted traffic classification using deep learning. Soft Comput 24, 1999 - 2012 (2020). https://doi.org/10.1007/s00500-019-04030-2

[7] Guanglu Sun, Xuhang Li, Xiangyu Hou, and Fei Lang. GPU-Accelerated Support Vector Machines for Traffic Classification [J]. Int J Performability Eng, 2018, 14(5): 1088-1098.

[8] T. Groléat, S. Vaton, and M. Arzel, "High-speed flow-based classificationon FPGA," Int. J. Netw. Manag., vol. 24, no. 4, pp. 253-271, Jul. 2014.