Network Prediction with Traffic Gradient Classification using Convolutional Neural Networks

Taejin Ko, Syed M. Raza, Dang Thien Binh, Moonseong Kim* and Hyunseung Choo
Sungkyunkwan University, Suwon, Korea, *Seoul Theological University, Bucheon, Korea
choo@skku.edu, moonseong@stu.ac.kr

***Abstract*— Current TCP/IP network infrastructures and management systems are facing a tough time in handling the unusual increase in network traffic due to the surge of typical real-time applications. To solve this problem, management system predicts the changes in network traffic and handle them proactively. In this paper, we convert the traffic prediction into a classification problem and use Convolutional Neural Network (CNN) deep-learning technique to classify the fixed time interval traffic into different classes. We implement the CNN model using Python and Keras library. The proposed algorithm shows higher accuracy (92.6%) and F1 score than the existing Random Forest machine learning method.**

Keywords- Network traffic, prediction, deep learning, convolutional neural networks,

I. INTRODUCTION

Expeditious growth in computing power and wireless technologies have triggered the massive adoption of mobile devices, an increase in mobile content and services, and new technologies like Internet of Things (IoT). As a result, global internet traffic per month has increased from 122 Exabytes (EB) in 2017 to 201 EBs in 2019 and is forecasted to increase to 396 EBs in 2022 [1]. The rise in network traffic compounded with elevated user Quality of Service (QoS) requirements is creating new challenges for operators to manage their networks. Traditional traffic management technologies use threshold-based algorithms that are slow to react to rapidly changing traffic. Consequently, traditional approaches fail to maximize network resource utilization and fall short of guaranteeing QoS under new dynamics of network traffic. A proactive approach is one way to overcome the weaknesses of conventional traffic management techniques.

Proactive traffic management requires to predict the forthcoming traffic load to prepare the network before the traffic enters the network. Network traffic prediction has been an active research area for the last many years. where various studies have predicted the workloads using a simple linear regression model [2], Brown's secondary exponential smoothing method using genetic algorithms [3], and ARIMA models [4]. Contrary to these approaches, another way to predict workloads is by understanding the underlying statistical characteristics of network traffic. Machine Learning (ML) algorithms use this approach and recently gained much attention due to their extraordinary results in other research areas.

Supervised ML algorithms learn from the features of already collected data to perform tasks like classification, prediction, and clustering. There are various ML algorithms for classification, and a recent study performs their comparative analysis for dynamic Virtual Network Function (VNF) scaling by classifying traffic load [5]. The objective of this study is to maintain the user QoS in case of increased traffic by dynamically scaling out the VNFs, and minimize the resources when traffic load decreases by scaling in. Conventionally, this is achieved through predicting the traffic load, but authors in [5] transformed it into a classification problem instead. They trained various ML algorithms with traffic load and an appropriate number of VNFs for it as a label at a given time. Their test results show that Random Forest algorithm achieves 96% accuracy in correctly classifying the number of VNFs required for the given traffic. Random forest shows excellent results with small test datasets, but it fails to fully exploit a large amount of data available in real networks, and its results do not improve beyond a certain point. Deep Learning (DL), an emerging ML technique, alleviates this limitation by automatically learning the features of training samples.

The DL algorithm models a high level of abstraction from data using an architecture consisting of multiple nonlinear transformations. One such DL architecture is Convolutional Neural Networks (CNN), consisting of convolutional, pooling, fully connected, and normalization hidden layers. Convolutional and pooling layers differ from the general neural network layer by using convolution and pooling as activation functions, respectively. It has been mainly used in computer vision research, but recently its use in classifying network traffic and intrusion detection has seen a considerable increase. One such study converts raw traffic data to images for malware traffic classification using CNN [6]. It splits the traffic of each flow or session into separate files and makes the size of each file consistent by trimming or padding. Later, each file is converted to an image where each byte of the file represents a pixel. However, this method does not consider the temporal characteristics of network traffic in their classification methodology.

978-1-7281-5453-4/20/$31.00 ©2020 IEEE

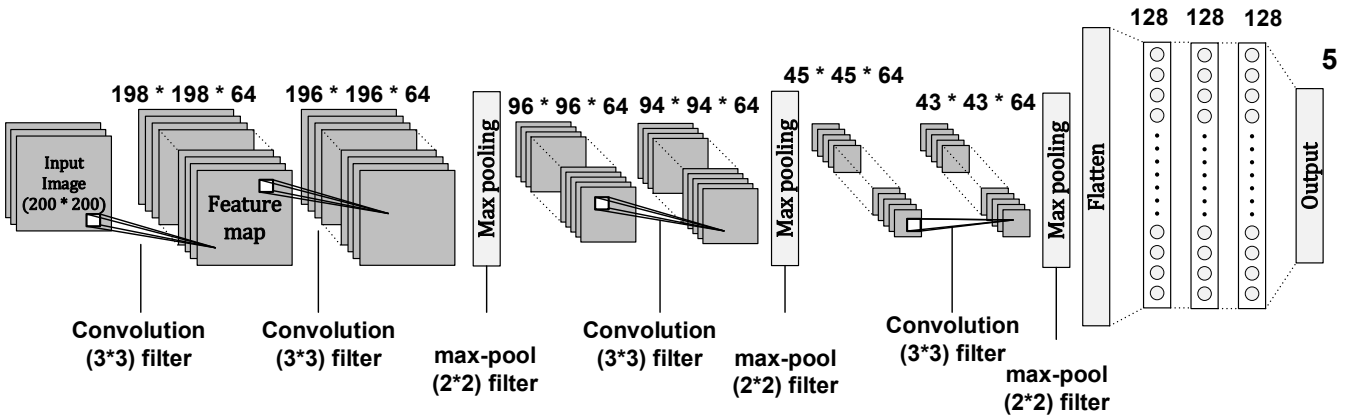


Figure 1. Architecture of CNN model for traffic prediction with feature extraction and classification

This paper presents a CNN based traffic prediction mechanism for proactive traffic management. The developed CNN model is trained using images that are created using raw packet traces data. The images capture the temporal characteristics of network traffic by showing the number of packets in a time interval and are classified into five classes of very low, low, normal, high, and very high. Trained CNN model classifies the test data into one of the five classes, which is taken as a prediction of forthcoming traffic, expecting no abrupt changes. The performance of the developed prediction mechanism is evaluated using images with different time intervals, and it shows 93% accuracy, which is 4% better than Random Forest, which shows 87% accuracy on our dataset.

II. NETWORK TRAFFIC PREDICTION WITH CNN

A. Covolutional Neural Networks (CNN) model

Network traffic prediction is generally performed based on the current traffic state recorded in the given time interval. If the time interval is small, it can be assumed that the current traffic state will persist for some period in the future. Based on this, coarse traffic prediction can be made by the classifying current traffic state. This transforms the prediction problem into a classification problem where the granularity of the prediction depends on the number of classification classes, *i.e.,* higher the number of classes higher the granularity and vice versa. In this paper, we develop a CNN model to classify the current traffic state.

CNN is a supervised DL algorithm that differs from typical deep neural networks by using convolution as an activation function, which is a widely used operation in image processing. Hence, CNN is specialized for computer vision applications such as image recognition, video recognition, and object detections, where it learns by extracting features from images [6]. CNN architecture can be divided into feature extraction part and classification part. The feature extraction part creates feature map by extracting image features from raw data, and it consists of convolution and max-pooling layers. The classification part consists of fully connected neural network with one or more hidden layers where a non-linear function is used in neurons for activation. It takes a feature map as input from feature extraction part and learns by updating the weights based on the loss function.

The architecture of CNN model developed in this paper for the classification of network traffic is shown in Figure 1. The feature extraction part consists of multiple convolution and max-pooling layers and classification part is composed of three fully connected hidden layers. At the beginning of feature extraction part, two convolution layers are stacked to effectively extract the features of the input image with 64 filters of size 3x3. In addition, pooling is performed to remove minor features from the output feature map and extract only the main values to make them smaller outputs. Max-pooling method is used in the pooling layer with size 2x2, and its objective is to select a large value for the filter. The combination convolution and pooling layer are repeated three times in the feature extraction part of architecture with an additional convolution layer at the beginning. The final features extracted from this part are converted to a single dimension through the flattening layer and fed into the first fully connected layer for the classification. All three fully connected hidden layers use Relu as activation function, and the output layer uses softmax activation function.

B. Data preprocessing

This paper uses EDU1 dataset [7, 8], which consists of network traffic traces from university campus datacenter. The datacenter consists of 22 network devices and 500 servers, and traces in the form of pcap files span 12 hours over multiple days. We have converted the network traffic in pcap files into images to exploit the image processing capabilities of CNN. The information about the amount of network traffic (*i.e.,* number of packets) in every second is extracted from the pcap files and stored separately. Using this information graphs for amount of network traffic in different time intervals are created, where each graph is an individual image file. Figure 2 shows an initial network traffic graph of 12 seconds, which is then cut into smaller images of 3, 5, and 7 seconds intervals. Resultantly, 3,040, 1,820, and 1,300 images are created for 3, 5, and 7 seconds interval,

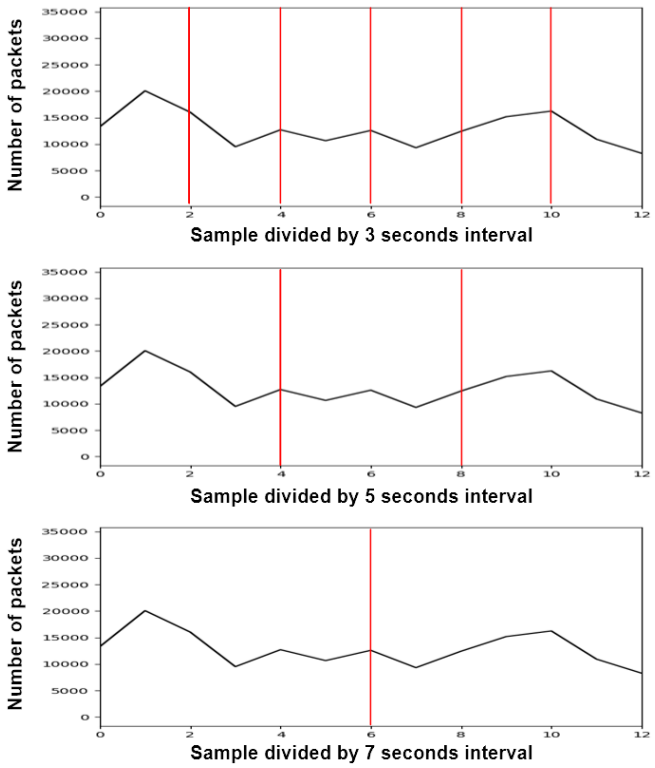


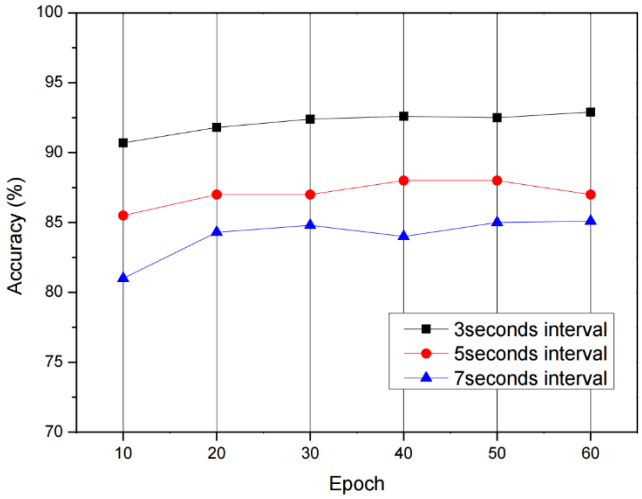
Figure 2. Example images with 3, 5, and 7 seconds intervals over same network traffic data

respectively. For a single experiment, CNN model is trained with images of the same interval size, and the performance comparison is drawn between different interval sizes.

CNN requires a label for each image to be provided explicitly. The number of distinct labels depends on the number of classification classes. For simplicity, this paper uses only five classes: very low, low, normal, high, and very high. Based on this class, each image is assigned a corresponding label. The label for an image is decided based on the gradient between start and end point of the image. If the gradient value is higher than threshold set for very high traffic than the image is labeled as very high, and similarly if gradient value is lower than a threshold set for very low traffic than image is labeled as very low. Same process is applied for images with high, low, and normal gradients. The thresholds for image labeling are set after analyzing the gradient values of all the images.

III. Results and Analysis

We implement the CNN model using Python and Keras library. Distribution of data for training and testing is 80% and 20%, respectively, with 30% of the training data is assigned for model validation. Training in each experiment is done with batch size 10, and step per epoch is set to $\frac{data\ count}{batch\ size}$. Input images are of size 200x200, which is the largest possible due to memory limitations. The hardware environment for experiments consists of Intel i7 CPU, 32GB of RAM, and GeForce RTX 2080 super GPU. We use CUDA



library from Nvidia to exploit the computation capabilities of GPU for performance gains. To establish statistical relevance all experiments are conducted 100 times.

The overall accuracy of our CNN model for different interval size images is evaluated in the first experiment with increasing epochs. The results in Figure 3 show that 3 seconds interval images achieve the highest classification accuracy of 93%, and 7 seconds interval images provide the lowest accuracy. This is counter-intuitive as conceptually historical data over a longer period should deliver better performance. We believe that the worst performance by 7 seconds interval images is due to a lack of data samples of training. Similarly, the highest performance by 3 seconds interval images is mainly due to enough data samples for training. Accuracy results for all three image types show a roughly 5% increase in accuracy with increasing epochs, and additionally, results of 7 seconds interval images show indications of overfitting after 30 epochs. This is again understandable due to a lack of training samples. Lastly, the result shows that the gap between minimum and maximum for each data type decreases as the number of epochs increases. Due to the superior performance of 3 seconds interval images, we compare their performance to random forest algorithm, which achieved the best performance among other ML algorithms [5].

Classification accuracy is the most intuitive evaluation indicator of the model performance. However, if there is a bias in the data being learned, accuracy cannot correctly express the model performance. In such a case, other evaluation indicators like F1 score can provide better insight into model performance. F1 score is a function of precision and recall, and it is used to measure model performance when labels are unevenly distributed in the dataset. Its formula is given as:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

, where

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

True positive is the number of cases when a positive value is correctly classified as positive, and False Positive is the number of cases when a negative value is wrongly classified as positive. Similarly, False negative is when a positive value is wrongly classified as negative.

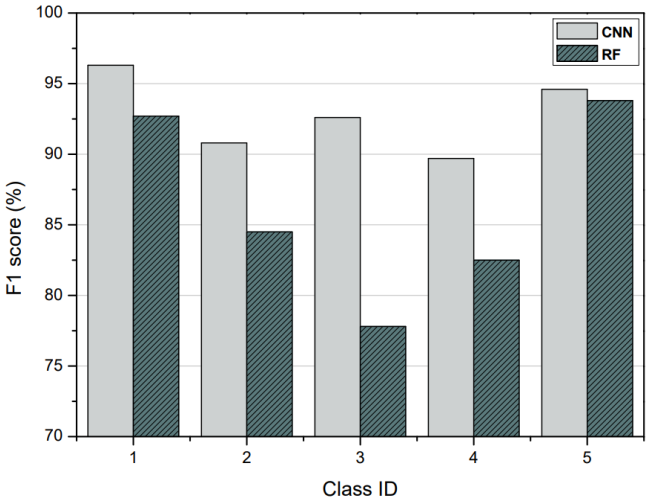


Figure 4. F1 score comparison between implemented CNN model and Random Forest

We compare the F1 score of our CNN model with Random Forest. For this purpose, we implement the Random Forest using scikit-learn library and train it by using our 3 seconds interval dataset. Figure 4 shows the F1 score of each classification class for CNN and Random forest, where classes are mapped to 1 to 5 in order of very low, low, normal, high, and very high. Results show that F1 score of CNN outperforms Random Forest for every class. Moreover, CNN shows a more consistent F1 score across all classes than Random Forest. This clearly indicates that CNN is a better choice for network traffic classification when not only accuracy but other performance indicators like precision, recall, and F1 score are also important.

IV. Conclusion and Future Work

This paper discusses the transformation of the network traffic prediction problem into a classification problem under the assumption that traffic patterns do not change drastically. CNN model is then presented to solve the network traffic classification problem, and to exploit its image processing capabilities, a mechanism for converting network traffic into graph images is presented. Experiment results confirm that our CNN model performs better than the highest performing ML algorithm. More interestingly, the results show that dataset with 3 seconds interval images achieves the highest accuracy of 93% rather than 7 seconds interval images dataset. We believe that this is because of a lack of image samples available for training. In our future, we look to continue our network traffic prediction research and enhance our dataset by generating synthetic graph images by using Deep Convolutional Generative Adversarial Network (DCGAN) [9]. The research will be conducted on the practical effectiveness of application of realistic applications such as management through traffic prediction and resolution, such as VNF auto-scaling. And we will explore a variety of problems with how to manage it.

Acknowledgment

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ICT Consilience Creative program (2019-2015-0-00742), High-Potential Individuals Global Training Program (2019-0-01579), and AI Graduate School Support Program (2019-0-00421), supervised by the IITP (Institute for Information and communications Technology Planning and evaluation). Professors H. Choo and M. Kim are the co-corresponding authors.

References

[1] Cisco, “Cisco Visual Networking Index: Forecast and Trends, 2017–2022,” White paper, 2019.

[2] A. Chandra, W. Gong, and P. Shenoy, “Dynamic resource allocation for shared data centers using online measurements,” Proc., 11th Intl. Conf. on Quality of Service, 2003.

[3] H. Mi, H. Wang, G. Yin, Y. Zhou, D. Shi, and L. Yuan, “Online selfreconfiguration with performance guarantee for energy-efficient largescale cloud computing data centers,” Proc., IEEE Intl. Conf. on Services Computing, 2010.

[4] W. Fang, Z. Lu, J. Wu, and Z. Cao, “RPPS: a novel resource prediction and provisioning scheme in cloud data center,” IEEE 9th Intl. Conf. on Services Computing, 2012.

[5] S. Rahman, T. Ahmed, M. Huynh, M. Tornatore and B. Mukherjee, "Auto-Scaling VNFs Using Machine Learning to Improve QoS and Reduce Cost," *2018 IEEE International Conference on Communications (ICC), 2018.*

[6] Wei Wang, Ming Zhu, Xuewen Zeng, Xiaozhou Ye and Yiqiang Sheng, "Malware traffic classification using convolutional neural network for representation learning," *2017 International Conference on Information Networking (ICOIN), 2017.*

[7] T. Benson, A. Akella, D. A. Maltz , "Network traffic characteristics of data centers in the wild," in Proceedings of IMC '10 Proceedings of the 10th ACM SIGCOMM conference on Internet measurement, 2010.

[8] Datasets for IMC 2010 datacenter measurements: http://pages.cs .wise.edu/~tbenson/IMC10_Data.html, retrieved on 12, 2019

[9] A. Radford, L. Metz, and S. Chintala, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks,” arXiv:1511.06434v2, Jan 2016.