Deep Learning Approach for In-Car Caching in Multi-access Edge Computing

Anselme Ndikumana, Choong Seon Hong*
Department of Computer Science and Engineering, Kyung Hee University, Rep. of Korea
{anselme, cshong}@khu.ac.kr

Abstract

Once self-driving car becomes a reality and people are not worrying about it, they will need to find new ways to entertain themselves. Therefore, for traveling people, self-driving car will be a new space for entertainment. However, there is a lack of literature on caching for entertainment content in self-driving car. Therefore, in order to address this challenge, we propose a deep learning approach for in-car caching in Multi-access Edge Computing (MEC), where convolutional neural network (CNN) is used in self-driving car for classifying people on board, in order to provide them high quality of entertainment contents. The classification is based on age, gender, and location. However, CNN results are insufficient to decide on which entertainment contents to cache. Therefore, the self-driving car needs to be assisted by MEC and data center (DC). At DC, long short-term memory (LSTM) framework is used for predicting content's probability for being requested in specific areas by certain categories of people. LSTM results are deployed to MEC servers located at the edge of a network in close proximity to the self-driving cars. Based on profile of people on board, the self-driving car can request MEC server the predicted results from DC, and compare it with its CNN results for deciding on which entertainment contents to cache that are appropriate to the people on board. The numerical results show that our proposal can be easily implemented in a realistic self-driving car environment.

1. Introduction

According to US National Highway Traffic Safety Administration (NHTSA) data of 2015, in Georgia, 94% of car accidents caused by human errors and bad decisions [1]. Therefore, in order to save lives, and prevent human errors and bad decisions, many research projects for self-driving car have been introduced [2].

In order to make self-driving car more intelligent, the self-driving car needs to be equipped with smart sensors and analytics tools that collect and analyze heterogeneous data related to people on board and environment in real-time. Furthermore, we consider self-driving car's resources to be limited, and it needs to be assisted by remote cloud. However, for effective self-driving car's data analytics, there is a strong need for low-latency and reliable communication. To reduce end-to-end delay needed for extensive car-cloud communication, we consider multi-access edge computing (MEC) [3,4] as a suitable technology that can help self-driving car for edge analytics. At MEC, communication, computation, control and caching (4C) is pushed to the MEC servers located to the edge of the network [5]. Here, in our proposal, MEC servers are deployed at the macro base stations (BSs), WIFI access points (WAPs) and roadside units (RSUs) in close proximity to the self-driving cars.

1.1. Challenges for In-Car Caching

For traveling people, self-driving car will be considered as new space for entertainment, where content providers, game developers, etc., can do business transactions within people on board. In other words, people within in the self-driving car will no longer be limited only on radio and TV, they can spend their time watching media, playing games, socializing online, and many more on their devices via the car's 4C system [6]. However, there is a lack of literature on caching of entertainment content in the self-driving car. Furthermore, as related work, self-driving car caching forum was introduced by GEOCACHING in March 2018 [7], but still there is no clear proposal on how caching in self-driving car can be implemented.

1.2. Contributions

In order to address the above highlighted challenges, we propose a CNN profiling approach for classifying people within a self-driving car, where CNN classification is based on age, gender, and location. However, CNN profiling results are insufficient to decide on which entertainment contents need to be requested and cached. Therefore, self-driving car needs to be assisted by MEC and data center (DC). At DC, we propose long short term memory (LSTM) framework for predicting the probability of the content to be requested in specific areas by certain categories of people. In order to minimize delays, the results of DC prediction are deployed in MEC servers attached to BSs, WAPs, RSUs in close proximity to the self-driving cars. Furthermore, based on car's CNN results, the self-driving car can request nearby MEC server the predicted results from DC and compare it with its profiling results. The output of the comparison helps self-driving car in deciding on which entertainment contents to request and cache in its content store that are appropriate to the age, location, and gender of people on board.

2. Deep Learning Approach for In-Car Caching

2.1. LSTM at Data Center

As shown in Fig. 1, we consider DC as a facility that host dataset from data market for prediction purpose. Therefore,

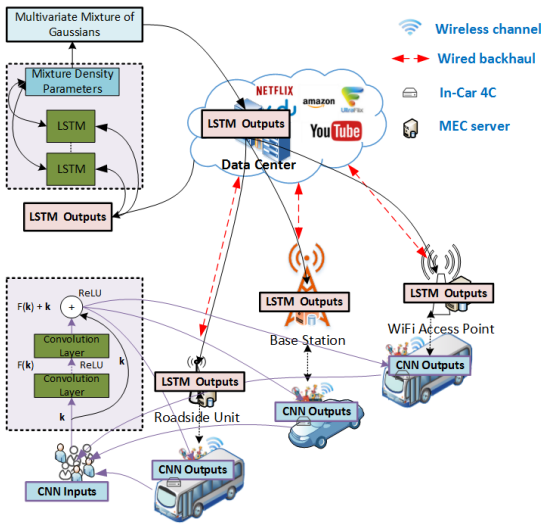

Figure 1: Illustration of our system model.

LSTM, which is described in [8], is deployed at DC for predicting user demands for entertainment contents, where the outputs of prediction are labeled based on age, gender, location. In LSTM, the dataset is partitioned into areas based on location of viewers with day-long time series data. At each time step t, we use i_t to denote content i, its viewer age and gender and location. We use $\mathrm{d_t}$ to denote day, hour, and minute, in which content i was viewed. The inputs of LSTM, at each time slot t is denoted $x_t = \{i_t, d_t\}$. From the inputs, LSTM tries to predict $y'_t = x_{t+1}$ as the output, where the outputs are based on current input, input from previous LSTM layer, and internal state. Therefore, in order to minimize error, the consolidated outputs of all LSTM layers will be compared with the ground truth data y_t.

Furthermore, in LSTM, the consolidated outputs of all layers are the mixture density of $\mathrm{M} \times (\mathrm{N}+2)$ parameters. M is used to denote the number of Gaussian kernels, and N is used to denote the number of areas. In addition, for each content i, we need to predict its probability for being requested in specific areas by certain categories of people (based on their age, gender and location). Each area corresponds to one neuron, in which we have $\mu_m(x)$ as mean and $\sigma_m(x)$ as variance. We use w_m as probabilistic weight, where $\sum_{m=1}^{M} w_m = 1$. Furthermore, each neuron is processed via softmax function, where the softmax function helps in transforming a vector of demands x into a set of probability values. Therefore, the softmax function is expressed as:

$$softmax(x)_j = \frac{e^{x_j}}{\sum_{m=1}^{M} e^{x_m}}. \quad (1)$$

We model the probability density function $p(y_t|x)$ of the next output y_t, through the use of the weighted sum of Gaussian kernels. Therefore, $p(y_t|x)$ is given by:

$$p(y_t|x) = \sum_{m=1}^{M} w_m p_m(y_t|x), \quad (2)$$

where $p_m(y_t|x)$ is the $\mathrm{m^{th}}$ multivariate Gaussian kernel. Therefore, $p_m(y_t|x)$ can be expressed as:

$$p_m(y_t|x) = \frac{1}{(2\pi)^{\frac{N}{2}}\sigma_m(x)} exp\left(\frac{\rho_k ||y_t - \mu_m(x)||^2}{2\sigma_m(x)^2}\right) \quad (3)$$

Furthermore, we define a negative logarithm likelihood as follows:

$$E_t = -\ln\left(\sum_{m=1}^{M} w_m p_m(y_t|x)\right). \quad (4)$$

Moreover, after the model is trained, based on inputs at time step t, the demands for content i at $t+1$ can be predicted. The results of the prediction, which are multivariate Gaussian mixed distribution, are deployed at RSUs, BSs and WAPs based on their locations.

2.2. CNN Profiling for Self-Driving Car

At the edge, we consider that self-driving car has 4C system that can provide in-car entertainment for people on board through caching entertainment contents. Therefore, for deciding on which entertainment contents to request and cache, we propose CNN profiling approach for people within a car, where self-driving car is equipped with a camera system for recording incoming people.

We consider k_0 as the input image of people on board with three dimension spaces, k_j is the feature map after layer j, while $y = H(k)$ is a nonlinear transformation of feature map. Here, we consider CNN block, where a block is composed with sub-layers. Therefore, the desired feature $H(k)$ is defined as follows:

$$H(k) = F(k) + k, \quad (5)$$

where $F(k)$ is a learning function. Furthermore, we consider $conv_j(k_j)$ as convolution layer j applied to the input image k_j, where a residual block is given by:

$$k_{j+1} = ReLU(conv_j(k_j) + k_j). \quad (6)$$

The output of the convolution layer j $(conv_j(k_j))$ is passed through rectified non-linearity (ReLU) function. Furthermore, k_{j+1} is the feature map after convolution layer j. In addition, we use ReLUs to save the output of the final layer, where the number of outputs corresponds to the number of class labels.

2.3. Join LSTM and CNN Results for In-Car Caching

We assume that the MEC server store the LSTM results, which is related to the specific location. In addition, the self-driving car does not have a large number of people on board. From this, in self-driving car, we can assume that comparing the predicted results from DC with its CNN profiling results does not require more computation resource. Therefore, for comparing both results from LSTM and CNN, and recommend entertainment contents to request and cache, we use an existing join algorithm, namely indexed map-reduce join (IMRJ) algorithm [9].

3. Performance Evaluation

For the performance evaluation of our proposal, we use numerical analysis, where Python is used as a programming language. We use processed YouTube demographic dataset from Next Analytics [10], where we consider demographic data as LSTM results from DC that need to be deployed in MEC servers. We present the summarized demographic data in Fig. 2. Furthermore, for self-driving car, we consider one self-driving bus with 32 people on board. The summarized data for self-driving bus is presented in Fig. 3, where we consider self-driving bus's data as the results of CNN prediction.

For deciding on which entertainment contents to cache that are appropriate to people on board, we use both CCN and

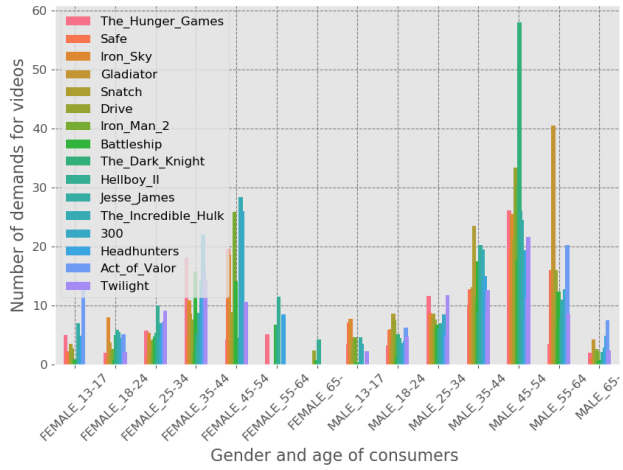

Figure 2: Used data as predicted demands for video at MEC server (LSTM).

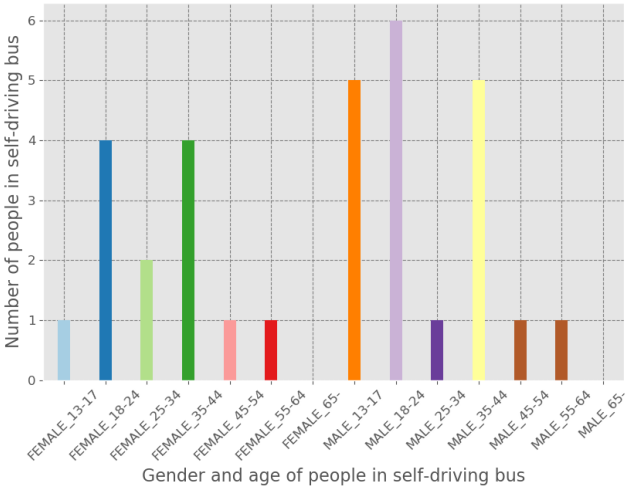

Figure 3: Used data as predicted people in self-driving bus (CNN).

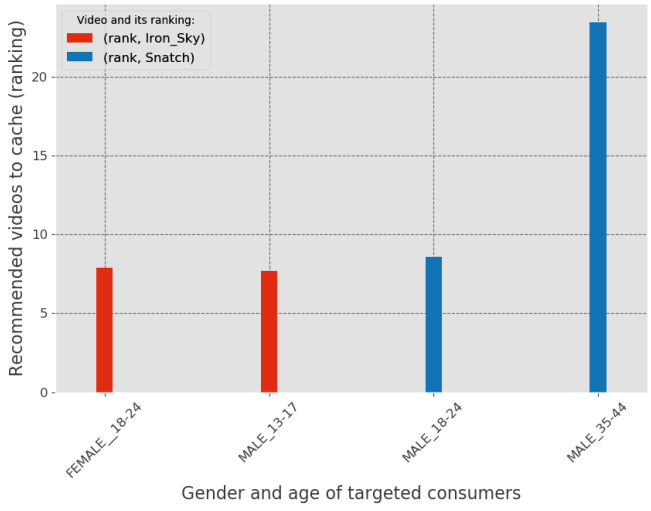

Figure 4: Recommended videos to cache in self-driving bus.

LSTM results. For comparing both results and ranking movies based on LSTM prediction and people in self-driving bus, we use python-pandas merge functionality, which is similar to IMRJ [9]. The numerical analysis results in Fig. 4 show that the largest number of people in self-driving bus are in the range from 18 to 44 years old. However, based on gender and age, people have different choices for movies. The results in this figure show the highest ranked movies that self-driving bus has to request and cache in its cache storage. More details on incentive Mechanism for content caching is provided in [11].

4. Conclusion

In this paper, we propose deep learning approach for in-car caching in Multi-access Edge Computing. CCN and LSTM are used for deciding/recommending entertainment contents to cache in self-driving car that are appropriate to the people on board in self-driving car. The numerical analysis presented in the paper shows that our proposal can be easily implemented in a realistic self-driving car environment.

5. Acknowledgement

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2015-0-00557, Resilient/Fault-Tolerant Autonomic Networking Based on Physicality, Relationship and Service Semantic of IoT Devices).
*Dr. CS Hong is the corresponding author.

6. References

[1]. Icebike, “Real time traffic accident statistics,” https://www.icebike.org/real-time-traffic-accident-statistics/, [Online; accessed Apr. 27, 2018].

[2]. M. Daily, S. Medasani, R. Behringer, and M. Trivedi, “Self-driving cars,” Computer, vol. 50, no. 12, pp. 18–23, 2017.

[3]. Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, “Mobile edge computing—a key technology towards 5G,” White Paper, vol. 11, no. 11, pp. 1–16, 5 Sep. 2015

[4]. A. Ndikumana, S. Ullah, T. LeAnh, N. H. Tran, and C. S. Hong, “Collaborative cache allocation and computation offloading in mobile edge computing,” in Proceedings of 19th IEEE Asia-Pacific Network Operations and Management Symposium (APNOMS), 27-29 Sep. 2017, pp. 366-369.

[5]. Ndikumana, Anselme, et al. "Joint Communication, Computation, Caching, and Control in Big Data Multi-access Edge Computing." arXiv preprint arXiv:1803.11512 (2018).

[6]. H. Lipson and M. Kurman, Driverless: intelligent cars and the road ahead. MIT Press, 23 Sep. 2016.

[7]. GEOCACHING, “Self Driving Car Caching,” https://forums.geocaching.com/GC/index.php?/topic/347757-self-driving-car-caching/, [Online; accessed Apr. 27, 2018].

[8]. Xu, Jun, et al. "Real-Time Prediction of Taxi Demand Using Recurrent Neural Networks." IEEE Transactions on Intelligent Transportation Systems (2017).

[9]. Khafagy, Mohamed Helmy. "Indexed map-reduce join algorithm." International Journal of Scientific & Engineering Research 6.5 (2015): 705-711.

[10]. Next Analytics, “Excel YouTube Analytic Insights and Data Mining,” https://www.nextanalytics.com/excel-youtube-analytic-insights-and-data-mining/page/4/, [Online; accessed Mar. 11, 2018]

[11]. A. Ndikumana, K. Thar, T. M. Ho, N. H. Tran, P. L. Vo, D. Niyato, and C. S. Hong, “In-network caching for paid contents in content centric networking,” in Proceedings of IEEE Global Communications Conference (GLOBECOM), 4-8 Dec. 2017 (Singapore), pp. 1-6