Real-time Segmentation of Drivable Area Using Supervised Learning

Chanwoo Ahn [1] and Jaeheung Park [1]

***Abstract*— Supervised learning is used to determine the driving range of images in self-driving. However, conventional methods cannot be used for actual vehicle driving due to their low accuracy and high computational time. Therefore, in this paper, we propose methods to address these problems by, i) simplification of class, ii) image resizing and iii) transmitting frequency tuning**

I. INTRODUCTION

One of the most important things to accomplish in self-driving technology is to predict a driving area. We can basically make this prediction by knowing which part of the image data from the camera is a driving area.

In this study, we use methods of supervised learning [1] to predict a driving area. Therefore, methods of a supervised learning with nonlinear characteristics are suitable. There are other studies which use supervised learning to detect a driving area, [2]–[4]. However, these studies typically have two problems; low accuracy and high computational time. Due to these problems, conventional supervised learning techniques are not available for actual vehicle driving.

To address these problems, three methods are used in this study. First, the number of output classes in the network is reduced to two categories: the driving area and the non-driving area to make deep learning network itself a problem of primary classification. Second, when training a deep neural network, unnecessary parts of images from camera sensor are cut out. Final, the number of images that are transmitted from camera sensors to deep learning networks is cut out, thereby reducing the time that a vehicle takes to determine a driving area.

II. METHOD

A. Supervised learning

Supervised learning is a method of learning a function which maps an given input to an output based on example input-output pairs. It infers a function g: X→Y from labeled training data consisting of a set of training examples, $((x_1, y_1), (x_2, y_2), ..., (x_N, y_N))$. x_i is a featured vector of ith example and y_i is a label. When the set of training examples are given, the supervised learning algorithm seeks the function g which minimizes a loss function. The averaged loss function, $R(g)$ is as follows.

$$R(g) = \frac{1}{N} \sum_i L(y_i, g(x_i)), \tag{1}$$

*This work was not supported by any organization

[1]are with the DYROS (Dynamic Robotics Systems) Lab, Graduate School of Convergence Science and Technology, Seoul National University, Seoul, Republic of Korea. Jaeheung Park is the corresponding author. a_chanu0612, park73@snu.ac.kr

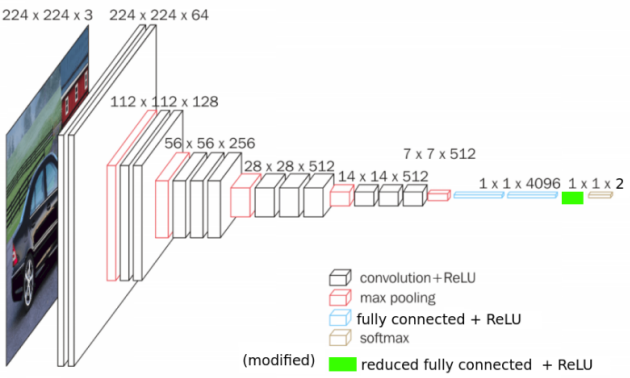

Fig. 1. Architecture of Network

where N is the number of training examples, L is a loss function for $g(x_i)$ which calculates a loss between the predicted value by $g(x_i)$ and a label, y_i. Hence, the supervised learning algorithm can be constructed by applying an optimization algorithm to find the function g which minimizes averaged loss function $R(g)$.

B. Modified Network architecture

VGG16-NET [5] is a base model of this study and we modify the architecture of the network. Fig. 1 represents a modified version of the architecture based on VGG16-NET. The input to the first convolutional layer is assumed to be a fixed size of a 224x224 RGB image. The image is passed through a stack of convolutional (conv.) layers, where filters were used with a receptive field: 3x3. In one of the configurations, it also utilizes 1x1 convolution filters, which can be seen as a linear transformation of the input channels (followed by non-linearity). The convolution stride is fixed to 1 pixel; the spatial padding of conv. layer is 1-pixel for 3x3 conv. layers such that the spatial resolution is preserved. Spatial pooling is carried out by five max-pooling layers, which follow some of the conv. layers (not all the conv. layers are followed by max-pooling). Max-pooling is performed over a 2x2 pixel window, with stride 2.

Three Fully-Connected (FC) layers follow the convolutional layers (which has a different depth in different architectures): the first two have 4096 channels each, the third performs 2-way classification and thus contains 2 channels (one for each class). The final layer is the soft-max layer [7]. The configuration of the fully connected layers is the same in all networks.

(a)

(b)

Fig. 2. Original, Resized image and magnified output image from the network. (a) Original, Resized Image, (b) Output Image (magnified)

C. Image Resizing and Transmitting Frequency Tuning

Images from a camera sensor are used to generate a set of training examples. The images are labeled as a driving area (r: 255, g: 0, b: 0) and a non-driving area (r: 255, g: 0, b: 255). And then, unnecessary parts of images are cut out to lower the computational time of the network. Fig. 2(a) shows the original image and the resized image.

A parameter which determines transmitting frequency of image from sensor to neural network is manipulated for lowering the computational time. By reducing the transmitting frequency, the number of transmitted images from sensor to neural network is reduced, therefore faster computation is available.

III. Evaluation

A. Experiment Setup

The proposed method is tested in actual driving situation. ROS (Robot Operating System) [6] is used for receiving images while actual driving, and data which contains images of an unknown environment are prepared to test the neural network.

B. Result

In Fig. 2(b), a red area is the driving area and a blue area is the non driving area. Test accuracy of the trained network is over 95% in unknown driving situation. This accuracy is estimated by comparing predicted image with labeled image from unknown environment pixel-wisely.

Additionally, computational time of proposed method is 9 frames/sec and that of original method is 3 frames/sec.

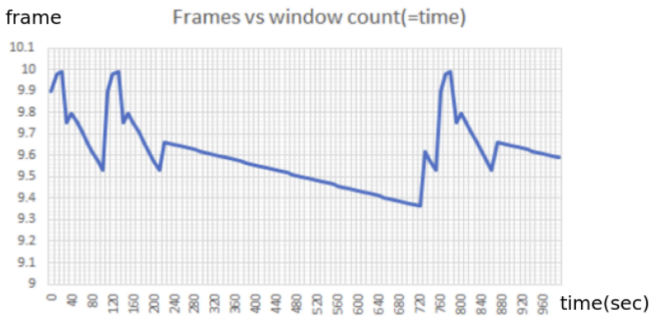

Fig. 3. Plotting of frames versus time when using the proposed method.

These result is shown in Fig. 3. Fig. 3 shows that frame versus time when using the proposed method is above 9 frames/sec. It means that prediction of driving area is available in real time while actual driving. Online video shows the whole result of the proposed method (`https://www.youtube.com/watch?v=6HkKbFcjONA`).

IV. Conclusions

In this study, we have proposed improved supervised learning method to obtain the driving area with high accuracy and low computational time during actual vehicle driving. We define only two classes in the network for more accurate prediction. Furthermore, the raw camera image is resized and transmitting frequency of image from camera sensor is reduced for faster computation. After driving in an unlearned environment, the proposed method could identify the road on which the vehicle is able to travel in real time for most of the cases. Even if the result shows high accuracy for the most of the cases, a small portion of the cases still shows low accuracy. Additionally, if a vehicle gets faster, a driving area cannot be predicted in real-time. In future work, some works include optimization of hyperparameters of neural network and trying other networks which have properties of more accurate prediction and lower computational time.

References

[1] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics New York, 2001, vol. 1, no. 10.

[2] M. Teichmann, M. Weber, M. Zoellner, R. Cipolla, and R. Urtasun, "Multinet: Real-time joint semantic reasoning for autonomous driving," *arXiv preprint arXiv:1612.07695*, 2016.

[3] M. Siam, S. Elkerdawy, M. Jagersand, and S. Yogamani, "Deep semantic segmentation for automated driving: Taxonomy, roadmap and challenges," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, Oct 2017, pp. 1–8.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, pp. 84–90, 2012.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2015.

[6] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[7] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.