End to End Learning for Self-Driving Car

EEML extended abstract - Reproduction

Ali Rahimi Kalahroudi*
Department of Math Science, Sharif University of Technology, Tehran, Iran

I. INTRODUCTION

CafeBazaar is one of the most well-known IT companies in Iran. It has several products; CafeBazaar which is an android app market, Divar which is a free classified ads mobile app, and Balad which is an online map, and navigator.
Every year in an event called *Hackathon*, which is 2-3 days long, employees would form different teams, and work on their new (usually fancy) ideas.
At 7th Hackathon of CafeBazaar, we decided to reach the MVP of a self-driving car, and formed a team to work on this idea.

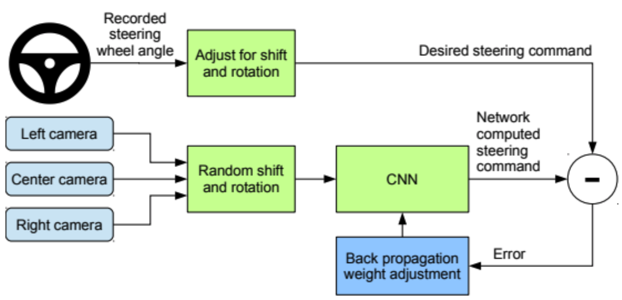

FIG. 1. Training the neural network.

II. OBJECTIVE

In this work, our mission was to reach the MVP of a self-driving car which could sufficiently stay in a line on the roads. After some literature review, *NVIDIA*'s *DAVE-2* [2] was chosen to be implemented. We trained the proposed model on two different data sets, and also *VisualBackProp* method [3], and [1] for checking where the model would focus the most on the road images.
Finally, we put all parts together, and test our model on a real car, and its performance was surprisingly well. The fact that all the work was done in a team of 10 (data scientists, back-end developers, and operational people) in just two days was astonishing.
Our implementation is available on Github [4] .

* Also at: CafeBazaar Research Center, Tehran, Iran; ali.rahimikalahroudi@gmail.com

III. TECHNICAL OUTLINE

A. Overview of the System

Figures 1, and 2 (copied from the original paper [2]) clearly show the work flow, and we take the exact same thing in our implementation.

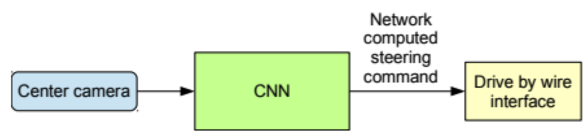

FIG. 2. The trained network is used to generate steering commands from a single front-facing center camera.

B. Data Collection

At first we used an open-source data set available on google drive to boost our implementation, and training part; this data set is also available on our github repo.
On the other side, our operational team was working on a real car, setting up the cameras, and the required devices in order to collect our own data set, and prepare the car for testing time.
One of the challenges for data collecting was to evaluate the steering command, and pairing it with the corresponding 3 images in front of the car. We used image processing techniques, and installed four different colored marks on the steering wheel as shown in figure 3, and we estimated the steering command angle with the pictures of steering wheel.

FIG. 3. Four different colored marks on steering wheel.

C. The Model

FIG. 4. Two different images from the open-source data set. Left is the real front image, and the right is generated from VisualBackProp method.

We used the exact same network architecture explained in the original paper [2], *PilotNet*.
We also used one of pretrained ResNet architectrues, as a different architecture to compare these two models results, but because of the slow convergence of ResNet model, and our limited time, we just continued with PilotNet.
We also implemented the VisualBackProp method explained in [3], a backward architecture to PilotNet made of several deconvolutional layers in order to check where the most focus of PilotNet is on our collected data. Figures 4, and 5 are generated from VisualBackProp method.

One surprising fact we found out for the open-source data was that the model, PilotNet, would focus on the sky part of the pictures as well as road lines which is clear on the second image of figure 4.

FIG. 5. An image from our own collected data set. Left is the real front image, and the right is generated from VisualBackProp method.

D. Implementation

We implemented the model in pure PyTorch, and formed a team of three using mob programming. My contribution to this work was being in this team, and I also implemented the VisualBackProp method.

[1] Bojarski, M., Choromanska, A., Choromanski, K., Firner, B., Jackel, L. D., Muller, U., and Zieba, K. Visualbackprop: visualizing cnns for autonomous driving. *CoRR abs/1611.05418* (2016).

[2] Bojarski, M., Testa, D. D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., and Zieba, K. End to end learning for self-driving cars. *CoRR abs/1604.07316* (2016).

[3] Bojarski, M., Yeres, P., Choromanska, A., Choromanski, K., Firner, B., Jackel, L. D., and Muller, U. Explaining how a deep neural network trained with end-to-end learning steers a car. *CoRR abs/1704.07911* (2017).

[4] https://github.com/MahanFathi/end2end-self-driving-car.