Let's Drive: Learning Autonomous Driving through Behavior Cloning

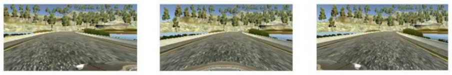
Mario Srouji, Jenna Lee
{msrouji, yunjlee } @ stanford.edu

Introduction

Context: Autonomous Vehicles are the next biggest challenges in AI and ML
Challenge: Learn to output steering angle to have the vehicle stay within the lane via supervised learning on driving examples.
Approach:
- Obtained driving data through Udacity's Self Driving Vehicle Simulator
- Several data augmentation techniques to overcome bias in the dataset
- Implemented, trained and tuned Convolutional Neural Network
- Acceptable error and driving performance given the relatively small number of data

Data Structure

- 17,350 images of the road (320px x160px) from the left, center and the right side of the vehicle

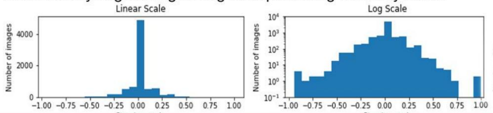


- .csv file of image locations, steering angle, applied brakes and throttle, and vehicle speed was input to the neural net

Left	Center	Right
IMG/point1_left.jpeg	IMG/point1_center.jpeg	IMG/point1_right.jpeg

Steering Angle	Brake	Throttle	Vehicle Speed
0.0617599	0.9855	0	2.124567

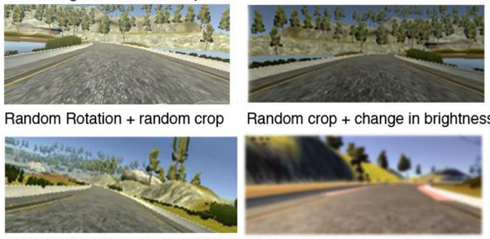
- Biased data; most of our driving examples are along the straight lane. Overcame by augmenting driving examples along the curvy roads.



Data Augmentation

-- Cropping of top and bottom border to input only the road image

Other Augmentation techniques:



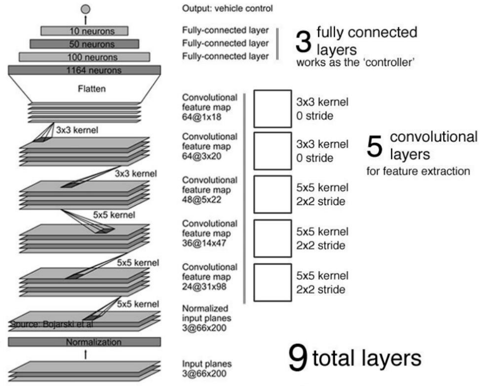
Random Rotation + random crop
Random crop + change in brightness
Random rotation + hue change
Gaussian noise (added as a layer)

-Images were split into YUV channels as inputs to the neural net

Model

Model is trained based on MSE error of steering angle prediction, i,e,.

$$\text{Error} = \frac{1}{N}\sum_{i=1}^{N}(y_{true} - y_{output})^2$$



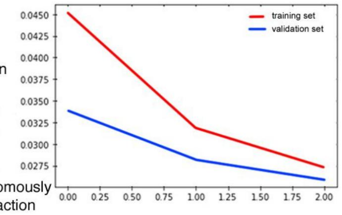
Results

0.025 training error

0.022 validation error

Comparable result with huber loss

Agent can run autonomously without human interaction 86% of time



- Agent struggles more when tested in completely different environment ('challenge' tracks in the simulator)

Discussion

- Relatively small number of images (17k datapoints with random augmentation at each epoch) was enough to fully train the CNN with the acceptable MSE error
- Relatively simple architecture and certainly less complicated design than conventional component-by-component design approach
- Simulation setup greatly simplifies the problem and limits validation
    - Lower vehicle speed
    - No other vehicles on the road
    - Relatively homogeneous environment
- Lower performance on 'challenge' tracks in the simulator indicate that the model potentially has issues with generalization
- MSE error is not representative of how safe the output steering angle is

Future Work

- Reduce the number of CNN parameters
- Use of more expressive loss functions (total variation) to reduce jerkiness in steering output
- Input the series of camera frames as the trainitng example instead of a single YUV channels of camera images using RNN

Reference

[1] Y. LeCun et al., "Backpropagation Applied to Handwritten Zip Code Recognition", Neural Computation, vol. 1, no. 4, pp. 541-551, 1989. Available: 10.1162/neco.1989.1.4.541.
[2] A. Krizhevsky, I. Sutskever and G. Hinton, "ImageNet classification with deep convolutional neural networks", Communications of the ACM, vol. 60, no. 6, pp. 84-90, 2017. Available: 10.1145/3065386.
[3]W. Zhang, J. Zhang, X. Du, Y. Zhang and S. Li, "An end-to-end joint learning framework of artery-specific coronary calcium scoring in non-contrast cardiac CT", Computing, 2018. Available: 10.1007/s00607-018-0678-6.
[4]"udacity/CarND-Behavioral-Cloning-P3", GitHub, 2019. [Online]. Available: https://github.com/udacity/CarND-Behavioral-Cloning-P3. [Accessed: 19- Mar- 2019].