UDACITY

Back to Self-Driving Car Engineer

Use Deep Learning to Clone Driving Behavior

REVIEW

CODE REVIEW 3

HISTORY

Meets Specifications

Congrats on passing this challenging project!

You've done a great job of using deep learning to clone driving behavior and are now well on your way to completing Term 1 of the ND program.

Looking forward to more good work from you in the near future!

Required Files

The submission includes a model.py file, drive.py, model.h5 a writeup report and video.mp4.

Quality of Code

The model provided can be used to successfully operate the simulation.

The code in `model.py` uses a Python generator, if needed, to generate data for training rather than storing the training data in memory. The `model.py` code is clearly organized and comments are included where needed.

Well done on the quality of code!

Your model successfully operates the simulation and your `model.py` code is easy to read and uses a Python generator appropriately to generate data for training.

Model Architecture and Training Strategy

The neural network uses convolution layers with appropriate filter sizes. Layers exist to introduce nonlinearity into the model. The data is normalized in the model.

Good job implementing a CNN for this task, using ideas from the Nvidia model!

You've appropriately used ReLU activation to introduce non-linearity and normalized the data using the Keras lambda layer.

Besides ReLU, here are some other activation methods you may want to experiment with:

- Advanced Activations Layers
- Fast and Accurate Deep Network Learning by Exponential Linear Units

To speed up neural network training, you may also like to read up on Batch Normalization:

- Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift - 2015 paper

Train/validation/test splits have been used, and the model uses dropout layers or other methods to reduce overfitting.

Nice work using train/validation split and dropout layer in the network to reduce overfitting!

You may also like to read the Dropout paper on how it prevents overfitting in neural networks:

- Dropout: A Simple Way to Prevent Neural Networks from Overfitting - 2014 paper

Learning rate parameters are chosen with explanation, or an Adam optimizer is used.

Adam optimizer is used to control learning rate.

Training data has been chosen to induce the desired behavior in the simulation (i.e. keeping the car on the track).

Well done choosing appropriate training data to keep the car on the track!

- "I create two aditional data sets, one in the default drive direction and other in opposite way, in each data set approximately 12 laps were taken including laps with center lane driving, left and right driving and zig zag driving..."
- "I found that the use of Udacity and non inverted PS3 data sets in a transfer learning approach shows adequate behavior in both tracks working with a speed value=3 in a drive.py file. After some tests the objective was achieved by first training with the data set of udacity for a total of 5 epochs with a strong adjustment in the left and right camera angles followed by a fine adjustment with PS3 data for 2 epochs with slight adjustment of left and right camera angles..."

Architecture and Training Documentation

The README thoroughly discusses the approach taken for deriving and designing a model architecture fit for solving the given problem.

The README provides sufficient details of the characteristics and qualities of the architecture, such as the type of model used, the number of layers, the size of each layer. Visualizations emphasizing particular qualities of the architecture are encouraged.

The README describes how the model was trained and what the characteristics of the dataset are. Information such as how the dataset was generated and examples of images from the dataset must be included.

Simulation

No tire may leave the drivable portion of the track surface. The car may not pop up onto ledges or roll over any surfaces that would otherwise be considered unsafe (if humans were in the vehicle).

Nice job here!

As can be seen the video, the car is able to drive successfully throughout the track.

If you'd like to improve the driving performance further, some modifications/enhancements to the training data you could consider are:

- instead of recording zigzag driving behavior, record only recovery driving behavior (ie. avoid recording drifting out towards the track borders)
- augmenting the training data with image processing techniques (shifts, brightness adjustments, etc.) to create more diverse examples for the model to learn from

DOWNLOAD PROJECT

3 CODE REVIEW COMMENTS ›

RETURN TO PATH

Rate this review