UDACITY

PROJECT

Use Deep Learning to Clone Driving Behavior

A part of the Self Driving Car Engineer Nanodegree Program

PROJECT REVIEW

CODE REVIEW 4

NOTES

SHARE YOUR ACCOMPLISHMENT!

Meets Specifications

Congratulations, your project meets all specifications!

I've enjoyed very much reviewing your project, you can be proud of the hard work you've done!
Few articles:

NVIDIA's neural network model and image augmentation
Behavioral Cloning - Make a car drive like yourself
Batch size discussion
Using Augmentation to Mimic Human Driving
Cloning a Car to mimic Human Driving

Quality of Code

✓

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

Very good job providing a perfectly functional model, it works very well in the simulator! :)

✓

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.

- The code is presented in a very good quality, well organized, commented, and formatted, nice job!
- It is excellent that you are making use of the python generator, that allows you to use large dataset for training without worry about the memory limitations.

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.

- A very good job using both convolution and nonlinearity layers!
- Data is properly normalized prior training the model, great job here as well! :)

✓

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

- Great job splitting the training dataset into training and validation subsets!
- Dropout layers were properly used to reduce overfitting, nice job here as well!
  http://machinelearningmastery.com/dropout-regularization-deep-learning-models-keras/
  http://www.cs.toronto.edu/~rsalakhu/papers/srivastava14a.pdf

✓

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

✓

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

Training data was properly chosen to induce the desired driving behavior in the simulation.

Architecture and Training Documentation

✓

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

Good job explaining your approach for solving the given problem, a deep discussion/explanation is provided on how you approached achieving your final and functional model. :)

✓

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 architecture of the implemented model is properly explained in the README.

✓

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 should be included.

The training process is properly described in the README, great job!

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).

Excellent job here, model's simulation was good, the car drives itself across the entire circuit and always maintaining the required trajectory, on the drivable surface. :)

DOWNLOAD PROJECT

4 CODE REVIEW COMMENTS ❯

RETURN TO PATH