UDACITY

‹ Return to "Self-Driving Car Engineer" in the classroom

DISCUSS ON STUDENT HUB

Use Deep Learning to Clone Driving Behavior

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

Congratulations, your submission is awesome!
The car drives around extremely well on both tracks!
You are already using a very good set of techniques in your model.
Thus, to improve it is a matter of tunning (e.g., try to use ELU or Leaky ReLU as the activation function) and working on the
input data.
Continue working well and hard, you're doing a great work. 👏👏

Required Files

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

Quality of Code

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

The model was able to successfully operate the car in the simulation. Well done!

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 in model.py is clean and well organized.
Furthermore, a generator was used to feed the input data. Well done!
See more details about python generators in https://www.oreilly.com/ideas/2-great-benets-of-python-generators-andhow-they-changed-me-forever

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.

You have made a good use of convolution, dropout, batch normalization, and fully connected layers. Good job!
There is a good explanation of the importance of nonlinear activations in
https://towardsdatascience.com/exploringactivation-functions-for-neural-networks-73498da59b02
You can also read more about activation functions in Section 5 of this page:
http://lamda.nju.edu.cn/weixs/project/CNNTricks/CNNTricks.html

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

You have used train/validation splits and dropout layers to reduce overtting. Well done!
You can read more about dropout in https://medium.com/@amarbudhiraja/https-medium-com-amarbudhiraja-learningless-to-learn-better-dropout-in-deep-machine-learning-74334da4bfc5

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

Good job! Adam optimizer is a great choice for this project.

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

Good job selecting the data to train your model and using all available cameras.

Architecture and Training Documentation

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

Your README is awesome! Good job describing the approach to solve this 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 architecture was properly explained.
A representation of the model was provided with its layers and parameters.
Good job!

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.

The training method was explained properly.

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).

Awesome! The car is pretty stable

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review