Oliver Cameron Follow
I lead the self-driving car team at @udacity. Previously founder of a @ycombinator startup.
5 days ago · 6 min read

Challenge #2: Using Deep Learning to Predict Steering Angles

Udacity Self-Driving Car Challenge #2

The Udacity Self-Driving Car

As detailed in this post, a critical part of our process in launching the Self-Driving Car Nanodegree program is to build our own self-driving vehicle. To achieve this, we formed a core Self-Driving Car Team with Google Self-Driving Car founder and Udacity President Sebastian Thrun. One of the first decisions we made together? **Open source code, written by hundreds of students from across the globe!**

We are breaking down the problem of making the car autonomous into Udacity Challenges. The first challenge is complete, and you can read about it here. We're now ready to introduce Challenge #2!

. . .

Challenge #2—Now Open

You may have seen this incredible video from Nvidia, one of our Nanodegree partners, which highlights their efforts of teaching a car how to drive using only cameras and deep learning. Their DAVE-2 deep learning system is capable of driving in many different weather conditions, avoiding obstacles, and even going off-road! You may have noticed their setup looks pretty similar to our 2016 Lincoln MKZ, and that's for good reason. One of the first ways that we want to get this car on the road is to implement a similar end-to-end solution, and release that to the world for free.

The second challenge for the Udacity Self-Driving Car initiative is to replicate these results using a convolutional neural network that you design and build!

End-to-end solutions like this, where a single network takes raw input (camera imagery) and produces a direct steering command, are considered the holy-grail of current autonomous vehicle technology, and are speculated to populate the first wave of self-driving cars on roads and highways. By letting the car figure out how to interpret images on its own, we can skip a lot of the complexity that exists in manually selecting features to detect, and drastically reduce the cost required to get an autonomous vehicle on the road by avoiding LiDAR-based solutions.

We want students to show us what they can produce, and compete for an incredible array of prizes! By contributing to Challenge #2, you and your team will not only get to run your code on a real self-driving car, but you'll be in the running for cash, and even a trip to Mountain View, California to brainstorm with Sebastian Thrun, father of the self-driving car and founder of Udacity!

Important!

To join a team and get involved in the community growing around the Udacity Self-Driving Car, please join the Slack team here.

Looking for some more specific rules and information? Read on!

. . .

Challenge Overview

The purpose of the challenge is to take image frames from a camera mounted to the windshield of our car, and predict the appropriate steering angle using convolutional neural networks and deep learning. Training can take as long as you'd like, but the final network itself has to run in real-time.

While the dataset may include auxilary information, you may only use camera imagery and steering wheel angle information to train. The topic containing steering wheel information will not be present in the testing dataset. The extra data is only there to give you a full picture of the current state of the car, and to become familiar with the data format. To be clear, your network may only use camera imagery as input to your neural network.

The metric used to determine the winner will be measured and evaluated by the network's performance in simulation. Teams will be provided with two ROSbag-based datasets; one will be accompanied by steering angles, but the other won't. Training should be completed on the first dataset, and testing on the second. Teams will be required to generate a CSV file indicating their networks steering decisions, and this file will be uploaded and processed to determine team rankings. At the end of the competition period, the winning team's code will be adapted and integrated into our self-driving car platform, and taken out for a spin on the real deal.

Prizes

First Place: One-time sum of $10,000
Second Place: All-expenses-paid trip for the team leader and 2 other teammates to Udacity HQ in Mountain View, California to meet and brainstorm with Sebastian Thrun
Third Place: To be announced!

Timeline

Start Date: 09/29/2016
Submission Deadline: 10/28/2016
End Date: 11/04/2016

Essential Rules

- Toolchain used must be Python on top of TensorFlow.
- One team per participant, one submission per team, no maximum team size.
- Teams must have a designated lead who will submit their team's entry.
- A submission will be considered ineligible if it was developed using code containing or depending on software that is not approved by the Open Source Initiative, or a license that prohibits commercial use.

- No restrictions on training time, but must process a frame faster than 1/20th of a second.
- Winners must submit runnable code (with documentation and description of resources/dependencies required to run the solution) with reproducible results within (1) week of being selected as the Challenge winner.
- All code submitted will be open-sourced, and there should be no expectation of maintaining exclusive IP over submitted code.
- **No hand-labelling of test dataset allowed.**

For full contest rules, please read this.

. . .

Scoring

Udacity will provide the teams with two datasets, training and testing. The training set will be accompanied by steering wheel angle values for each frame, but the testing/evaluation set will not. The teams will then build a model on the training data, use it to predict on the testing data, and create a file with predicted steering angles for the test set (again for each frame). Teams will then upload this file with predictions to our servers, and we will calculate the score against the actual steering angles from the test set. Teams will test their code and evaluate locally before their submission by splitting the training set into their own training and validation set.

Evaluation Metric

Root Mean Square Error. From Wikipedia:

> *The root-mean-square deviation (RMSD) or root-mean-square error (RMSE) is a frequently used measure of the differences between values (sample and population values) predicted by a model or an estimator and the values actually observed. The RMSD represents the sample standard deviation of the differences between predicted values and observed values.*

. . .

Frequently Asked Questions

Submission Format & Details

Teams will be able to submit their final results only once on the testing set in CSV format via email to self-driving-car@udacity.com. The submission email must be accompanied by a list of teammates, team name, and code/documentation. More information on this format will be released in the coming weeks.

How do I get started?

1. Join the Slack team and join the **#challenges** channel
2. Download the example dataset (40GB) here
3. Install Ubuntu 14.04 in a virtual machine or directly onto your system.
4. Install ROS to playback data and convert into different formats
5. Begin building and testing your convolutional neural network using Python, TensorFlow, or Theano.
6. **Submit your results to self-driving-car@udacity.com with code (preferably in a Git repo) and team information.**

Data Format

All sensor data including imagery and steering wheel angles is provided in the ROSbag format. To playback this data, you will need to install ROS on a Ubuntu Linux platform and test from there. Additionally, you can convert the data into any format you like.

Definition of Real-Time Performance

Video processing latency has not been measured yet on target hardware with GigE camera. Anticipate a GTX 1070, i7–4770TE CPU, and 16GB+ RAM. There is some wiggle room on “real time performance.” Essentially, your network has to process 15+ frames a second. We expect difficulty here with replication until we have an AWS/Azure instance specification for later challenges.

Open Source Tools Clarification

In pre and post processing of your neural networks, you may use proprietary code and tools, as long as your final code/network/solution operates independently of any closed source code, as defined in the above rules.

. . .

Here We Go!

Udacity is dedicated to democratizing education, and we couldn't be more excited to bring this philosophy to such a revolutionary platform—the self-driving car! We anticipate this project to have an incredible impact on the industry, giving anyone access to the tools required to get an autonomous vehicle on the road. Help us achieve this dream by joining a team and competing in our challenges. Plus, if you're looking to gain the skills necessary to launch a career building cars that drive themselves, we encourage you to check out our Self-Driving Car Nanodegree program.