Exercise 3: Training End-to-end driving networks

Michael Floßmann, Kshitij Sirohi, Hendrik Vloet

June 11, 2018

1 Introduction

1.1 Goals

The main objective of this assignment was to reproduce the work done in the paper by Codevilla et al. [1]. This included:

- Reconstructing the described networks in pytorch ("command input" and "branched")
- Training the reconstructed networks on the original dataset from [1] with image augmentations implemented in a previous assignment (with some ablation studies).
- Optional: Implement a recurrent architechture to model the temporal domain.

1.2 Issues

During the assignment we encountered several which delayed our development progress. Due to these problems we where not able to achieve the goals above. Below we will state a short list with the most problematic issues, attends to solve them and whether this worked or not. After that, a short list of results we can provide within this report will follow.

Corrupted files While reading in the dataset, we encountered OSErrors. This was due to the corruption of one (and sometimes more than one) file. This file was basically empty and we had to write a "CheckCorruption" like routine to scan the dataset beforehand in order to avoid further crashes. Our routine works and removes corrupted files from the training set.

Disk Space Disk space was not sufficient to unpack the whole dataset and as consequence we where force to fetch the dataset from the university network which is slower than using the internal SSD (where access was restricted)

Runtime Training and Validation are quite time consuming, around 10h to 15h are normal for 5 epochs of training. We tried to speed it up by increasing the number of workers in the pytorch loader and accessing the validation sets in a more sequential way. This reduced the time consumption significantly.

Nightly Reboots We wanted to train during the night, but due to the nightly reboots, the training was interrupted. We could do nothing regarding this problem. The only thing we could do was to save the model on a periodic basis and continue after the reboot with a new training run, but the old model.

Listing 1: Nigthly Reboot Interruption

```
25153.85s - Train Epoch: 2 [344800/480640 (72%)]    Loss: 0.006571
25156.82s - Train Epoch: 2 [344900/480640 (72%)]    Loss: 0.006483
25159.51s - Train Epoch: 2 [345000/480640 (72%)]    Loss: 0.026645
25162.00s - Train Epoch: 2 [345100/480640 (72%)]    Loss: 0.019915
```

```
*** FINAL System shutdown message from root@login2 ***
System going down IMMEDIATELY

Nightly routine re-boot

25164.69s - Train Epoch: 2 [345200/480640 (72%)]    Loss: 0.006574
25167.05s - Train Epoch: 2 [345300/480640 (72%)]    Loss: 0.006452
25170.02s - Train Epoch: 2 [345400/480640 (72%)]    Loss: 0.027064
25173.58s - Train Epoch: 2 [345500/480640 (72%)]    Loss: 0.023583
Connection to login.informatik.uni-freiburg.de closed by remote host.
Connection to login.informatik.uni-freiburg.de closed.
```

Retain Graph Error This error occurred two times in total. Both in the way like listed below. After the first time, we did as suggested by the program and set the **retain_graph** attribute to true. But it did not help and the error did pop up again. This happened only on one of our private homestations, so we were basically forced to work on the pool computers.

Listing 2: Nigthly Reboot Interruption

```
2251.09s - Train Epoch: 1 [120000/526080 (23%)]    Loss: 0.041661
2251.59s - Train Epoch: 1 [120050/526080 (23%)]    Loss: 0.039515
2252.20s - Train Epoch: 1 [120100/526080 (23%)]    Loss: 0.037241
2252.68s - Train Epoch: 1 [120150/526080 (23%)]    Loss: 0.035107
Traceback (most recent call last):
  File "./Branched.py", line 401, in <module>
    main()
  File "./Branched.py", line 359, in main
    loss.backward()
  File "/home/hive/pytorch_venv/lib/python3.5
  /site-packages/torch/tensor.py", line 93, in backward
    torch.autograd.backward(self, gradient, retain_graph,
     create_graph)
  File "/home/hive/pytorch_venv/lib/python3.5/site-packages/torch/
  autograd/__init__.py", line 89, in backward
    allow_unreachable=True)  # allow_unreachable flag
RuntimeError: Trying to backward through the graph a second time,
but the buffers have already been freed.
Specify retain_graph=True when calling backward the first time.
```

Deus Ex Machina Somebody turns of a pool machine where we are training on.

2 Results

We managed to let the network ”Command Input” run a training with augmented images and non-augmented images. The training run with the augmented images is non-complete, since it was interrupted by a nightly reboot. Settings were the same for both runs:

- Batch size: 100
- 80% training, 20% validation split
- Learning Rate 0.0002
- Evaluation after every 500 batches

Missing information from the paper The paper doesn't state several important hyperparameters like the weights in equation (6) (page 4), or the magnitudes of the random parameters in the data augmentation. A git issue has been filed for the weight parameter, but the information given was inconclusive. (`https://github.com/carla-simulator/imitation-learning/issues/19`)

2.1 Results for Command Input: Augmented

The loss plots are displayed in the figures 1 and 2. The training error is too noisy to read anything substantial from it and the validation loss doesn't show a clear trend except for right after the very first validation when the model was completely untrained.

This may be an effect of the data augmentation hyperparameters not being tuned correctly, distorting the image too much and making the model underfit. The errors for the test set are listed in table 1.

Table 1: Error values for augmented command values

	Mean squared error	Median squared error
Steer	0.0357	0.0010
Gas	0.0879	0.0098

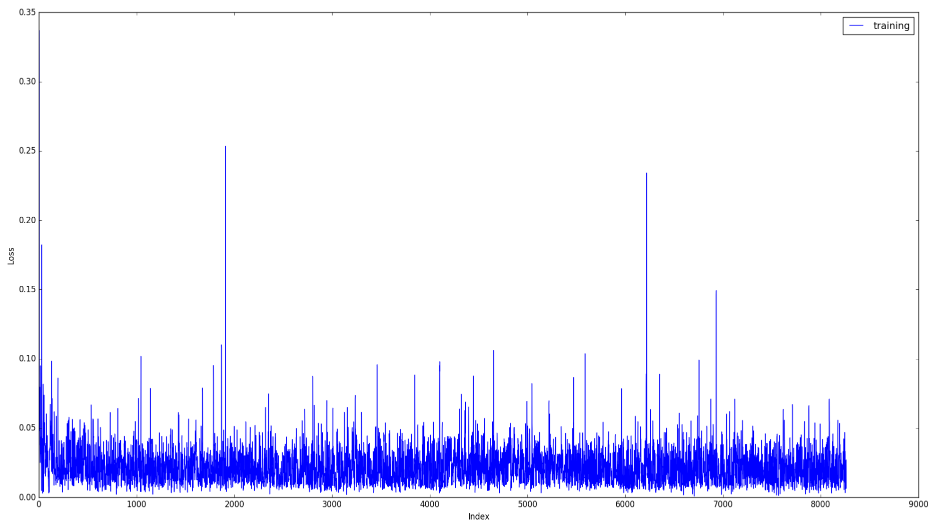

Figure 1: Train losses for augmented command input

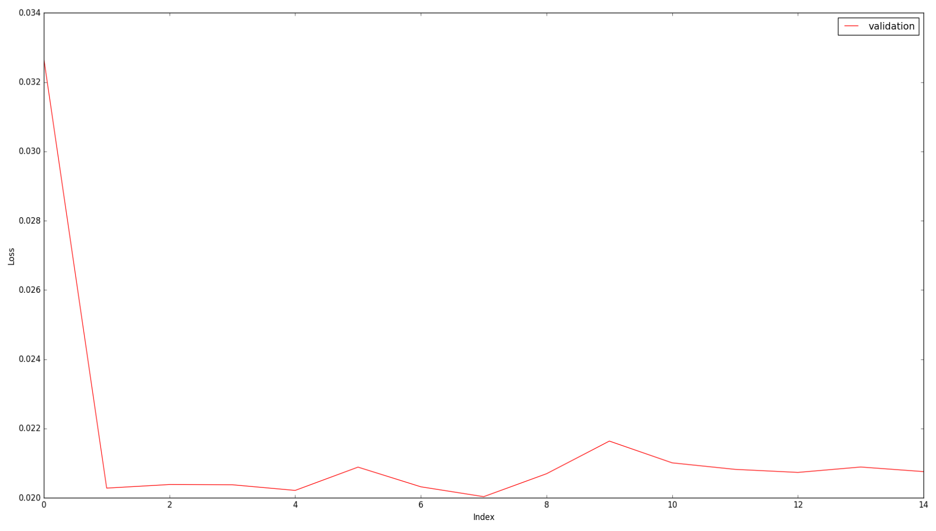

Figure 2: Validation losses for augmented command input

2.2 Command Input: Non-augmented

The loss plots are displayed in the figure 3 and 4. While the training error is very noisy still, the validation error shows a diminishing trend each epoch. This could suggest that more training epochs are needed for proper predictions. Also, the visible trend furtherly suggests that for the augmented training the augmentation hyperparameters must be better tuned in order not to underfit the model.

The errors for the test set are listed in table 2.

Table 2: Error values for non-augmented command values

	Mean squared error	Median squared error
Steer	0.0357	0.0017
Gas	0.0865	0.0109

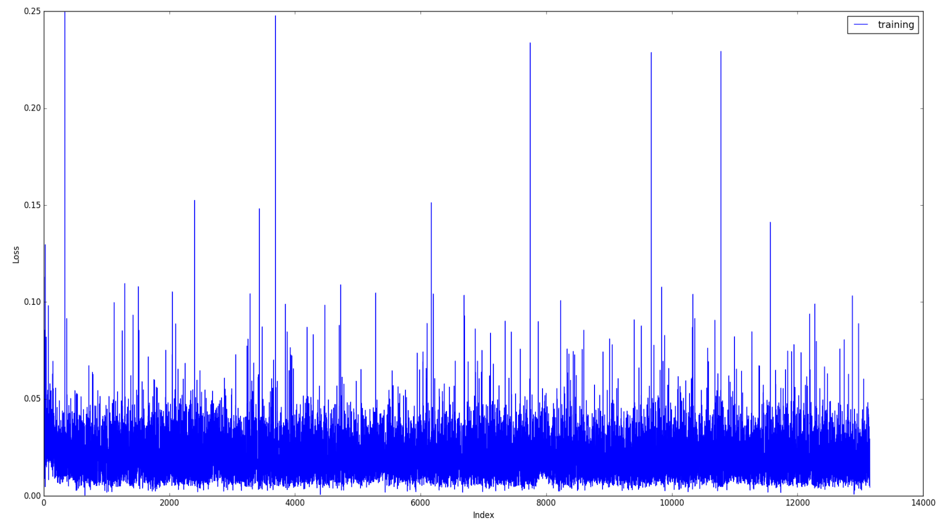

Figure 3: Train losses for not augmented command input

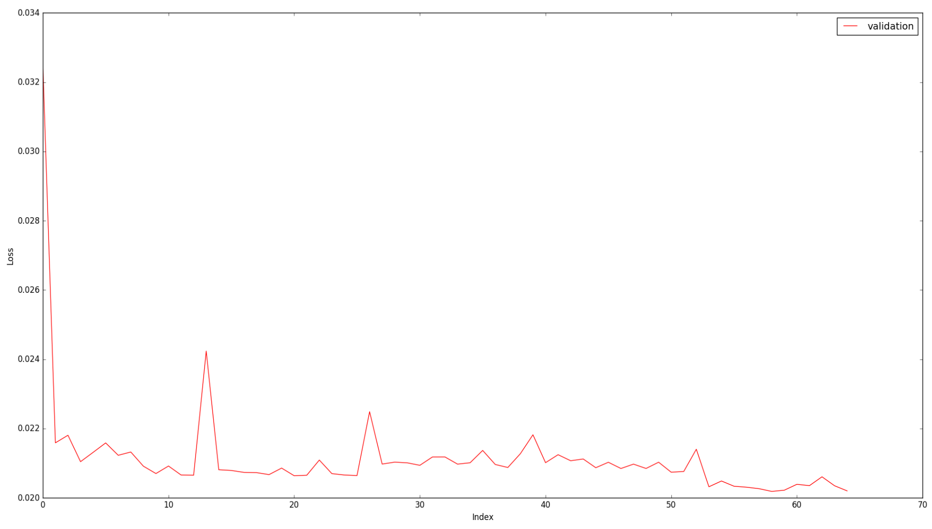

Figure 4: Validation losses for not augmented command input

2.3 Branched

The branched could not be analyzed because several issues got in the way. This will be addressed for the next assignment.

References

[1] Codevilla, Felipe and Müller, Matthias and López, Antonio and Koltun, Vladlen and Dosovitskiy, Alexey. *End-to-end Driving via Conditional Imitation Learning.* ICRA 2018