ITK ENGINEERING

Autonomous Lane Detection in a Simulated Environment

TUM Data Innovation Lab

Raju, Sukanya
Tabari, Azadeh

Wednesday, 17. Feb 2018

TUM MI, Garching bei München

Agenda

- Introduction
- Tools Set-up
- Environment Generation
- Convolutional Neural Network
- Interface to Test the Model
- CNN Improvements
- Network Results
- Demo
- Outlook
- References

Agenda

- Introduction
- Tools Set-up
- Environment Generation
- Convolutional Neural Network
- Interface to Test the Model
- CNN Improvements
- Network Results
- Demo
- Outlook
- References

Introduction

- Autonomous driving:

„[s]elf-driving means the autonomous driving of a vehicle to a specific target in real traffic without the intervention of a human driver.“ [Daimler AG]

© www.dailyreckoning.com

- Simulated environment:

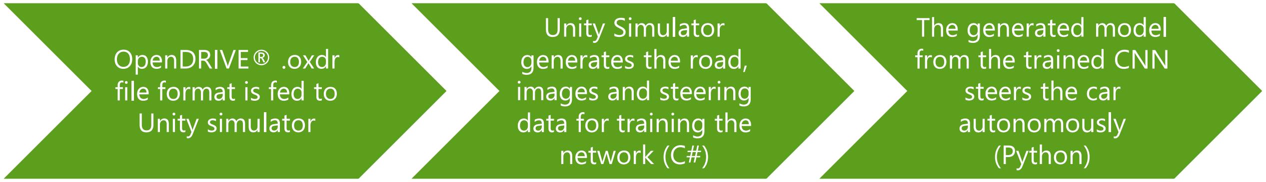
- SCRUM:
  - ✓ Weekly meetings, 3 times per week
  - ✓ 5 Sprints, every 3 weeks

Agenda

- Introduction
- Tools Set-up
- Environment Generation
- Convolutional Neural Network
- Interface to Test the Model
- CNN Improvements
- Network Results
- Demo
- Outlook
- References

Tools Set-up

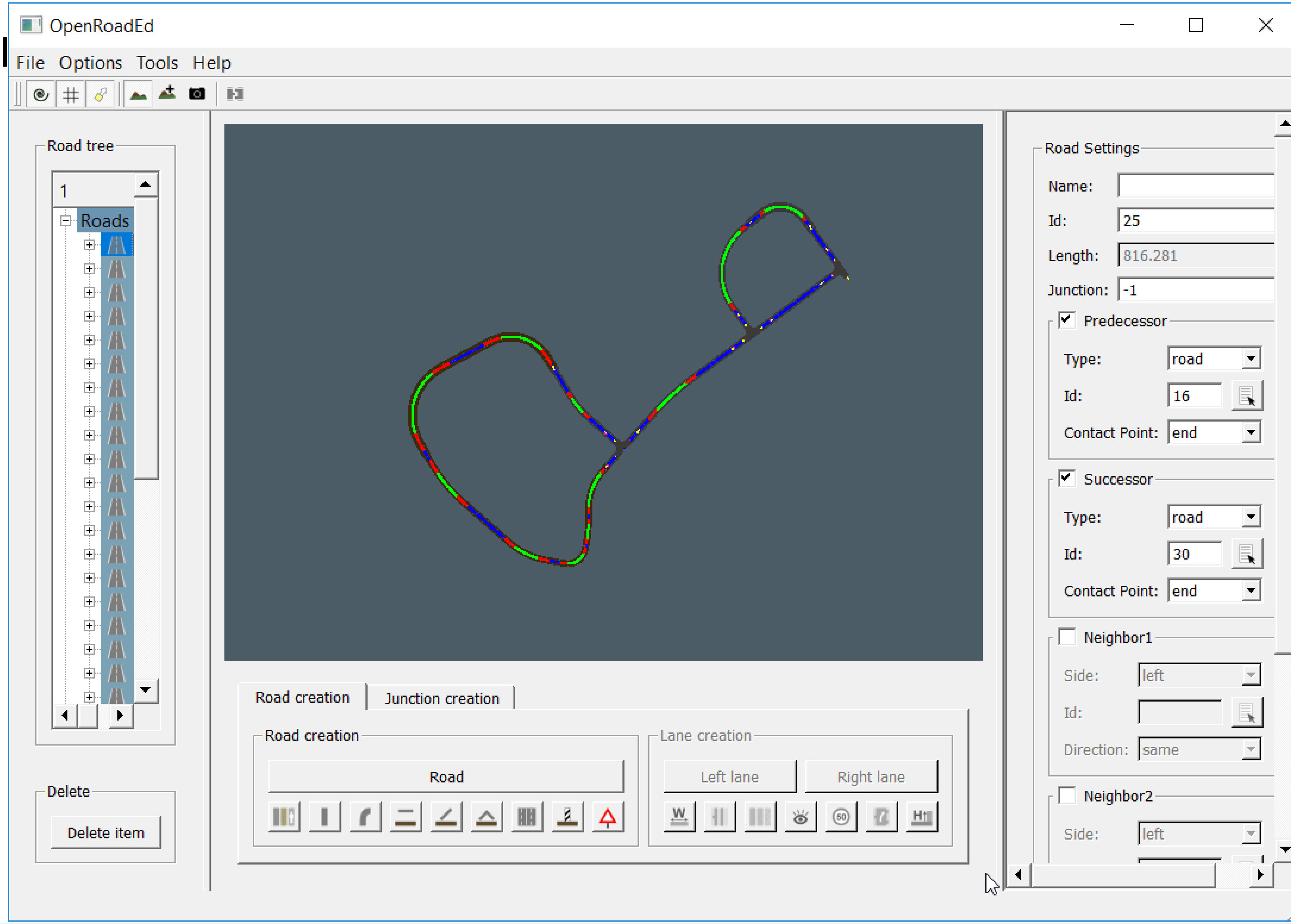
- Simulator:
  - ✓ Udacity's Self-Driving Car Nanodegree program
  - ✓ Unity game engine
- Programming Languages:
  - ✓ Python (Interface and CNN)
  - ✓ C#.NET (Unity)
- Additional:
  - ✓ TensorFlow with Keras
  - ✓ OpenDRIVE®
  - ✓ OpenRoadEd



Agenda

- Introduction
- Tools Set-up
- Environment Generation
- Convolutional Neural Network
- Interface to Test the Model
- CNN Improvements
- Network Results
- Demo
- Outlook
- References

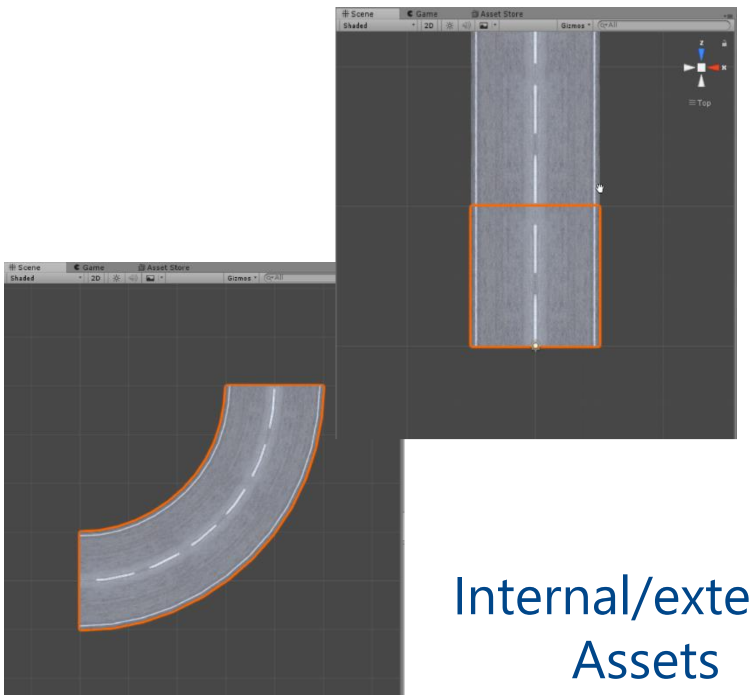
Environment Generation



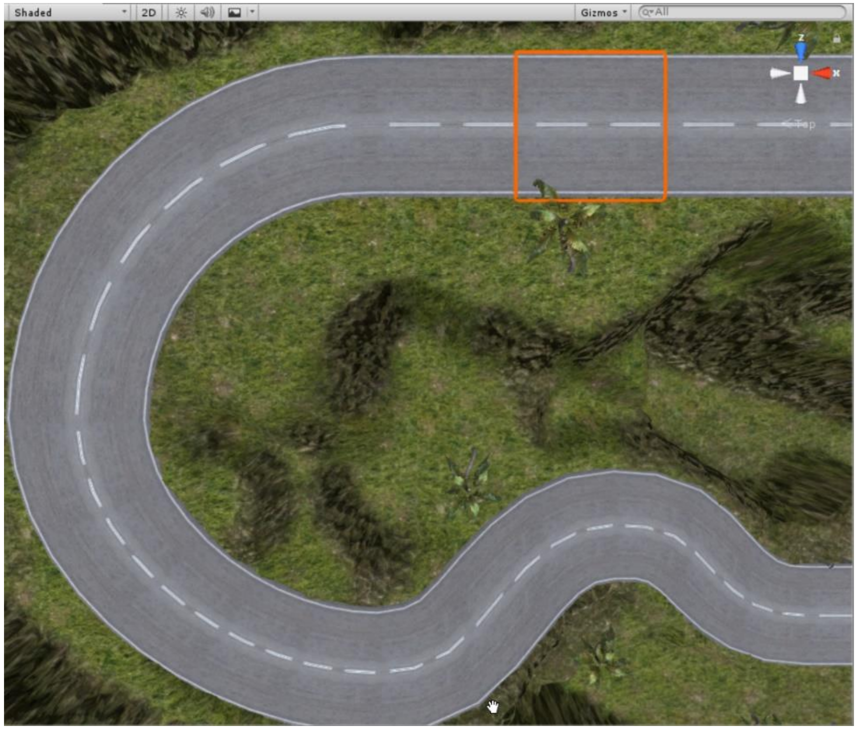
Environment Generation

In XML format with the specifications of openDRIVE®

1. Environment generation in OpenDRIVE® .oxdr format using OpenRoadEd App
2. feeding the generated .xodr file to Unity engine, Generation of the environment from .xodr file in C# code
3. Using the generated environment as a scene to be displayed as the virtual simulation environment



Internal/external Assets



Agenda

- Introduction
- Tools Set-up
- Environment Generation
- Convolutional Neural Network
- Interface to Test the Model
- CNN Improvements
- Network Results
- Demo
- Outlook
- References

Machine Learning

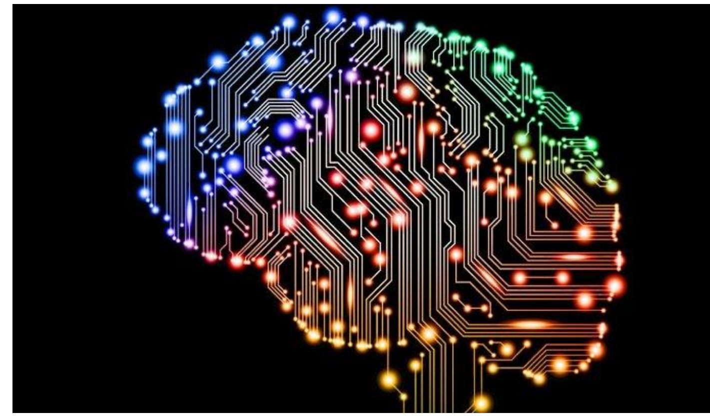
- Learn from experience

Machine Learning

Supervised Learning
-> Labeled Data

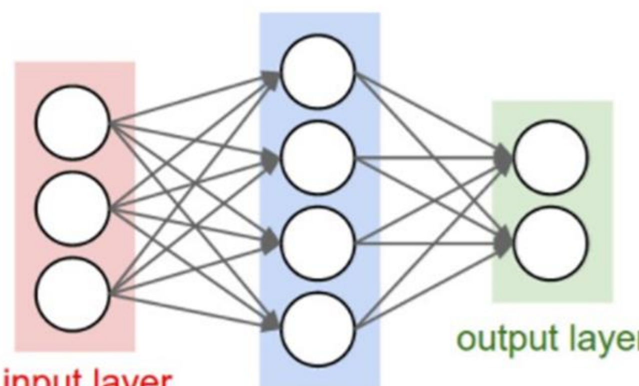
Un-supervised Learning
-> Not Labeled Data

Reinforcement
-> Reaction to environment



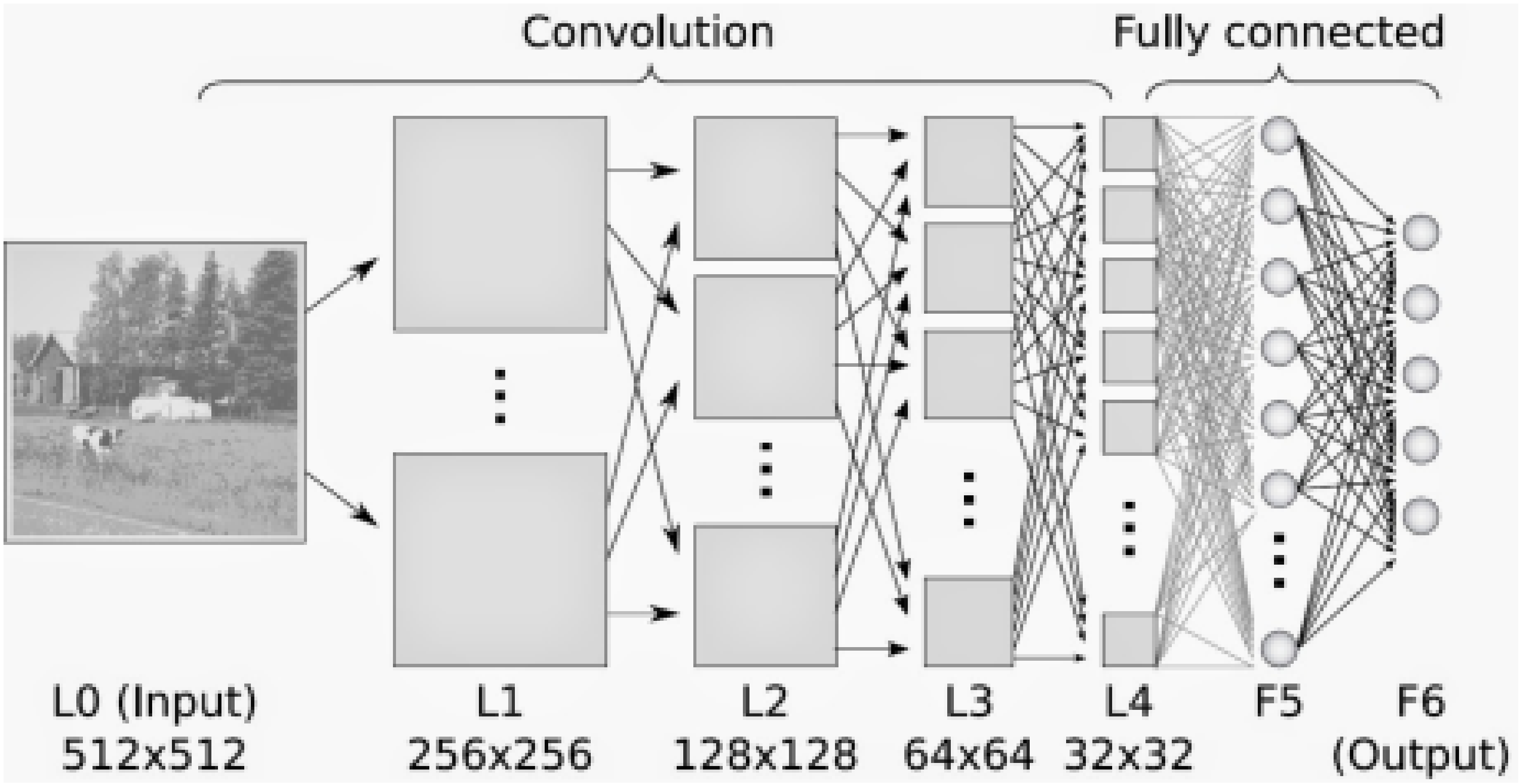
© www.businessfirstfamily.com

Deep Learning



Convolutional Neural Networks

- Using multiple copies of the same neuron in different places



© www.quora.com

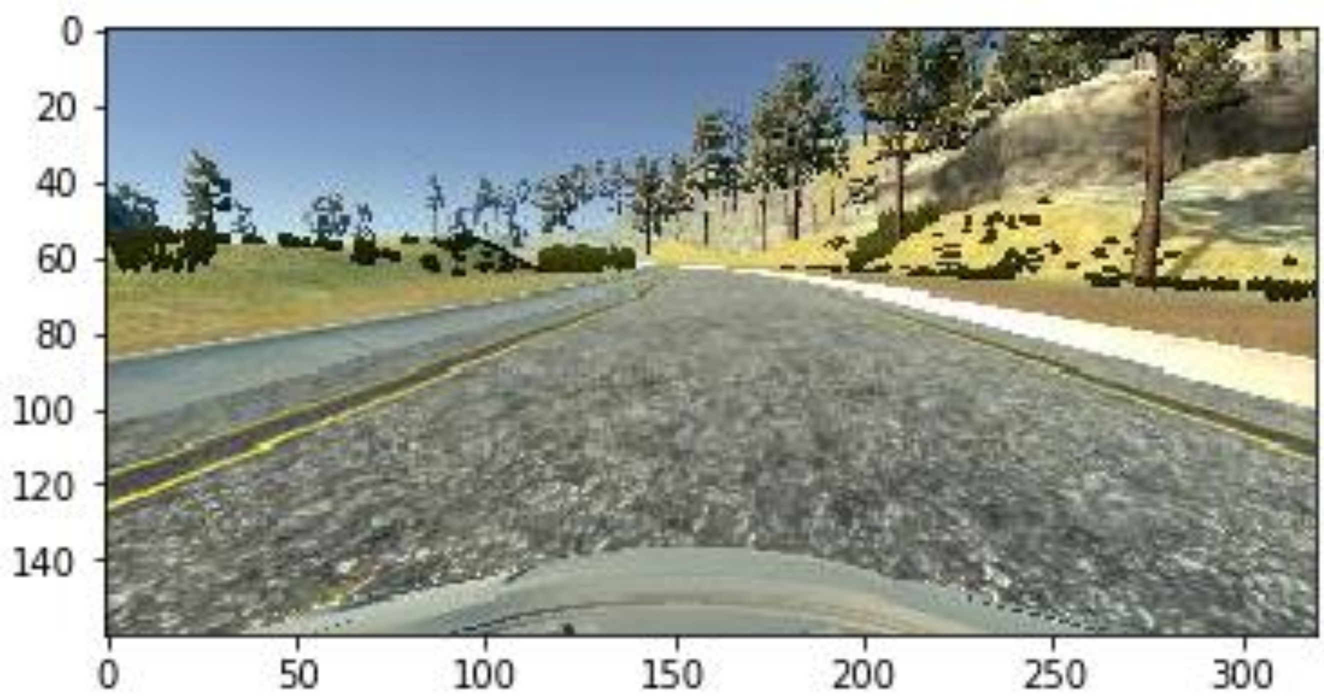
Loss Function

- Measures the compatibility between a prediction and the ground truth label
- Penalizing some measure of complexity of the model

$$Mean\ Squared\ Error = \frac{1}{n}\sum_{i=0}^{n}(\mathrm{y}_{(\mathrm{i})} - \hat{\mathrm{y}}_{(\mathrm{i})})^2$$

Image Received from Simulator

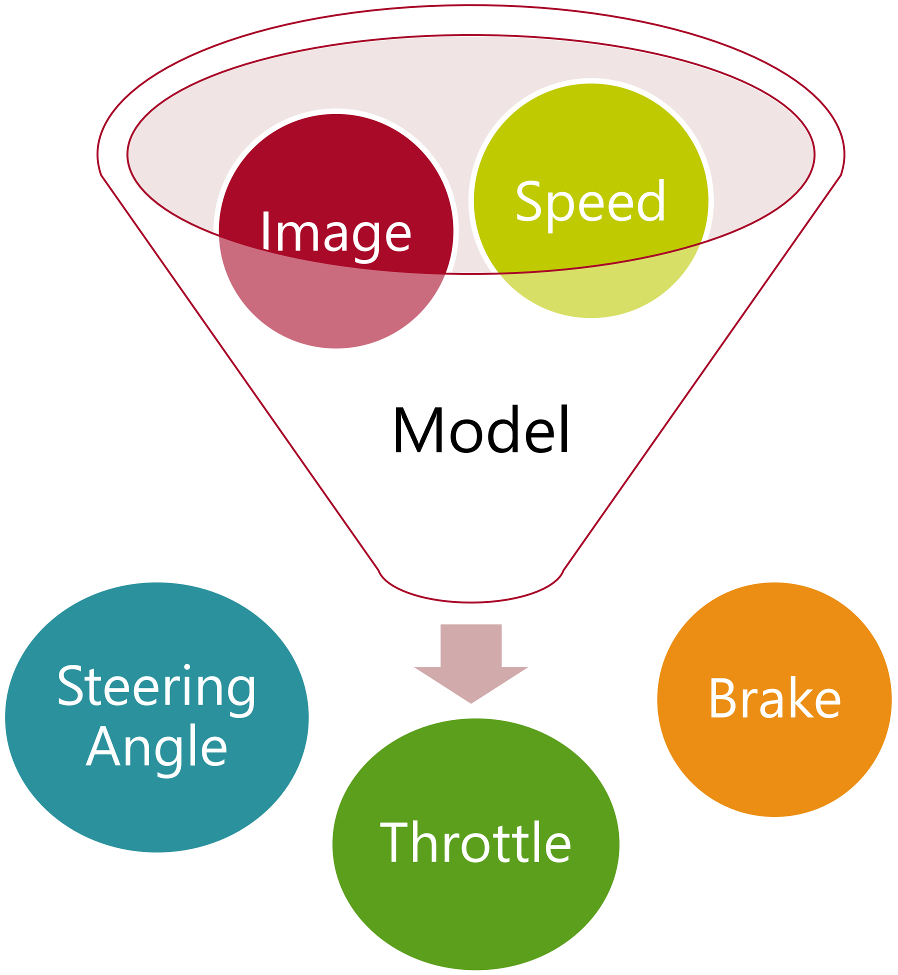
- Images used to train the model
- Size of Image : 160 * 320 * 3



Data Sample (.CSV) Received from Simulator

Steering angle	Throttle	Brake	Speed
0	0	0.4606265	13.05383
0	0	0.3536329	11.81833
0	0	0.1376654	10.90094
-0.1	0	0	9.877307
-0.25	0	0	9.801888
-0.45	0	0	9.677054
-0.3205477	0	0	9.583974
-0.1090438	0	0	9.527431
-0.2079084	0	0	9.447891
-0.3579084	0	0	9.375011
-0.3219513	0	0	9.279725
-0.113026	0	0	9.225466
0	0.05866113	0	9.15232
0	0.2672702	0	9.211304
0	0.4814161	0	9.511002
0	0.5307518	0	9.850943
0	0.2689583	0	10.13124

Model Input vs. Output



Input Pre-processing

- Images are normalized by dividing by 255
- Range of Speed : [6.58111200e-07, 30.5654]
  - ✓ This is normalized to range [0 , 1]

Ground Truth Data Pre-processing

- Steering Angle, Throttle, Brake are the ground truth data

- Range of each parameters :
  - ✓ Steering Angle : [-0.956159400, 0.8500001]

    (negative values for left turns and positive values for right turns)
    - This is normalized to range [0 , 1]
  - ✓ Throttle : [0.00000000 , 1]
  - ✓ Brake : [0.00000000 , 1]

CNN to Train Model

Speed1
Speed2
Speed3
.....
Input

Steering angle1
Steering angle2
Steering angle3
.....
Brake1
Brake2
Brake3
.....
Throttle1
Throttle2
Throttle3
.....
Ground Truth

Feature Extraction & Representation (Convolution)

Mapping (Maxpooling)

Steering angle1
Steering angle2
Steering angle3
.....
Brake1
Brake2
Brake3
.....
Throttle1
Throttle2
Throttle3
.....
Predictions

Back Propagation (Adam)

Basic Neural Network Details

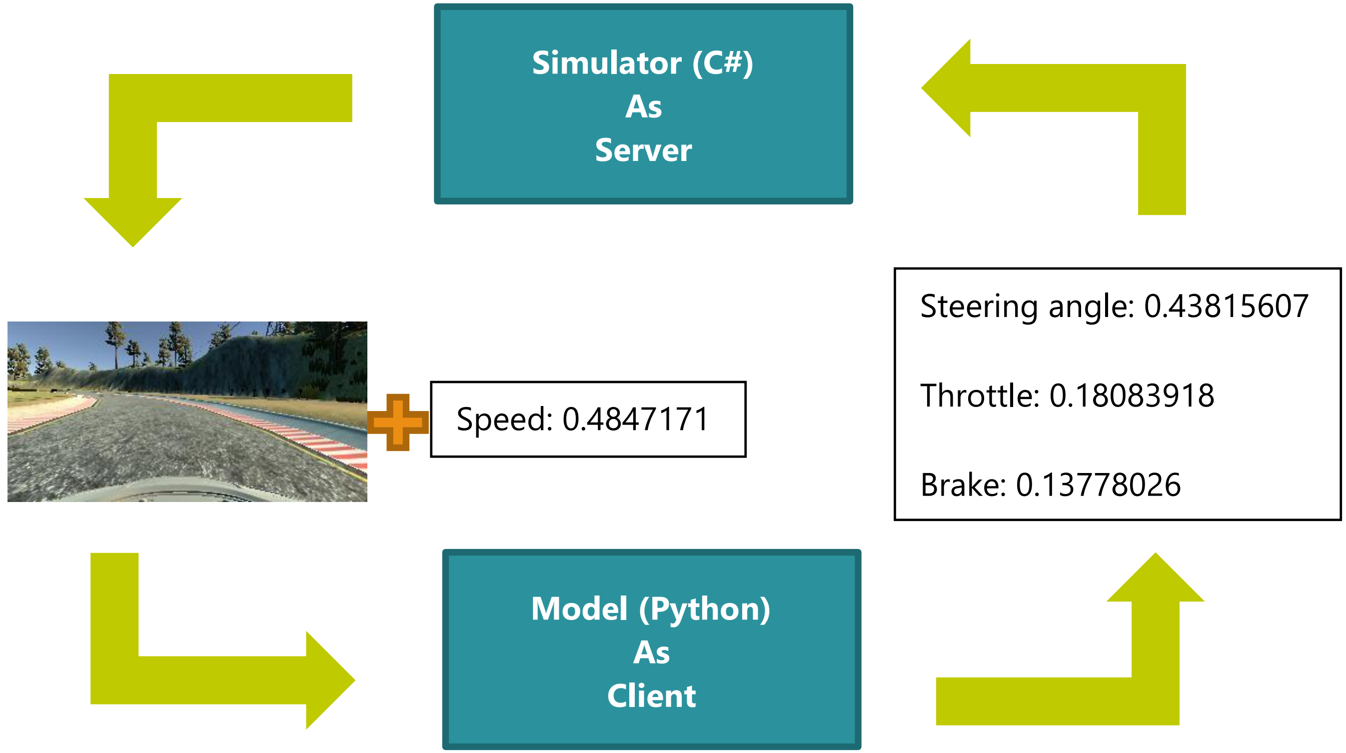
- Image size : 160 * 320 * 3
- Total images : 13842
- Training set : 11212
- Validation set : 1245
- Testing set : 1385
- Loss = mean_squared_error
- metrics= mean absolute error
- Optimizer : Adam
- Batch size : 32
- Epoch : 100

Agenda

- Introduction
- Tools Set-up
- Environment Generation
- Convolutional Neural Network
- Interface to Test the Model
- CNN Improvements
- Network Results
- Demo
- Outlook
- References

Interface to Test the Model

- Socket-IO:




```
Creating image folder at C:/Users/atabari/Pictures/itk
RECORDING THIS RUN ...
(1232) wsgi starting up on http://0.0.0.0:4567
(1232) accepted ('127.0.0.1', 56613)
connect  5f438d2f9861479886369278c5890917
Received data (Steering_angle, throttle, speed):   0.0 0.0 0.438
Image saved under: C:/Users/atabari/Pictures/itk\2018_02_03_22_06_14_161
1/1 [==============================] - 0s
Predicted values (Steering_angle, throttle, brake): -0.5097607970237732 0.09022938460111618 0.0146
```

Interface to Test the Model

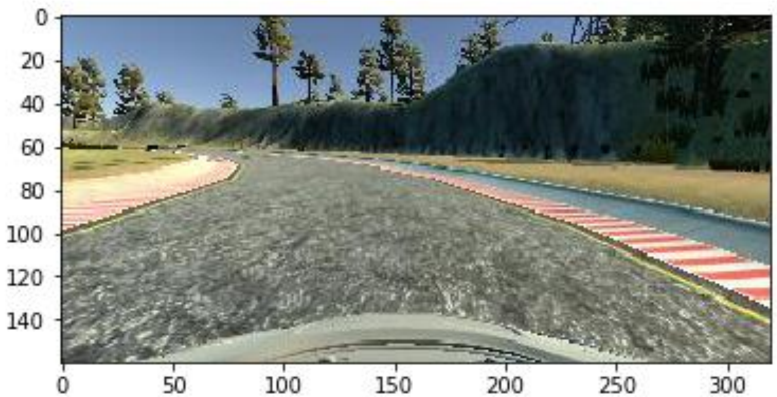
- Normalization: Speed / 30 , Image / 255
- Scaling: [0, 1] → [-1, 1]

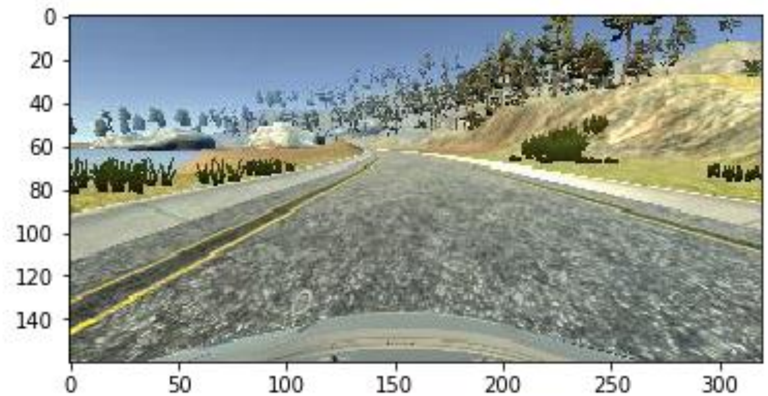
Agenda

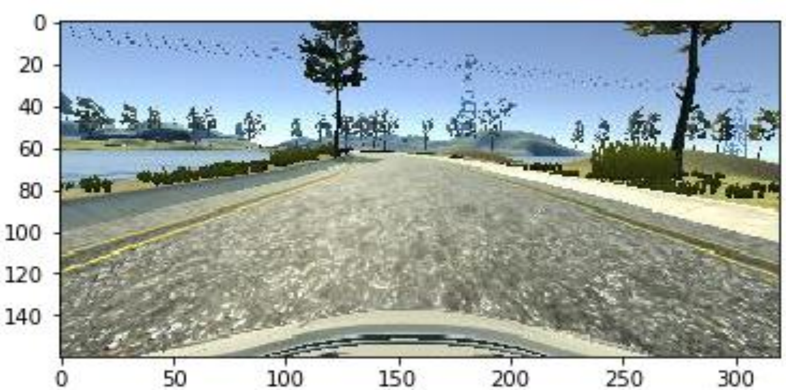
- Introduction
- Tools Set-up
- Environment Generation
- Convolutional Neural Network
- Interface to Test the Model
- CNN Improvements
- Network Results
- Demo
- Outlook
- References

Original Image Received from Simulator

- Images used to train the model
- Size of Image : 160 * 320 * 3







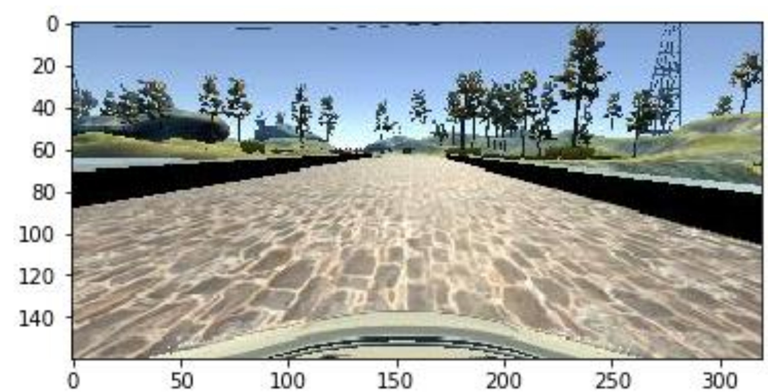
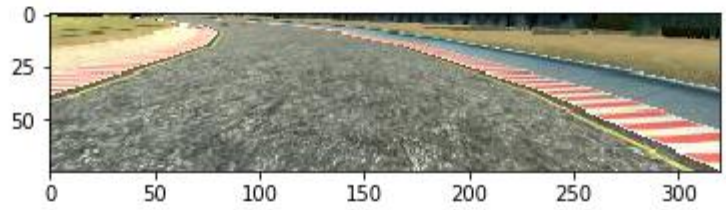
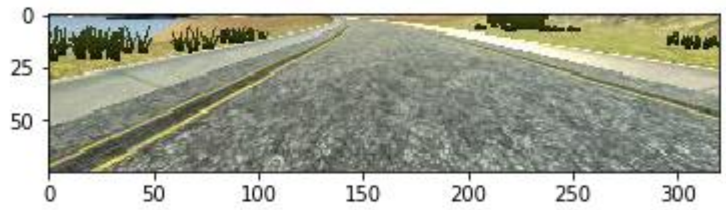
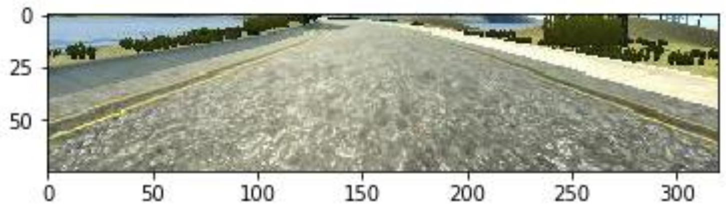
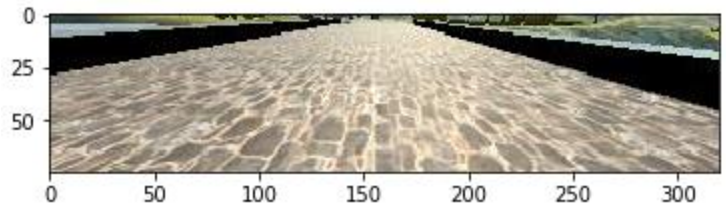


Image Pre-processing

- Images are cropped to show only the road in front
- Size of Image : 75 * 320 * 3

Training Scenario

- Data samples to train the model:
  - ✓ Car driven on road data

Model with Only Straight Road Data

Training Scenario

- Data samples to train the model:
  - ✓ Car coming back to road

Model Trained to Come Back on Road

Model Improvements

- Increase the depth of the CNN
- Sample weight added :
  - ✓ Steering angle : 2
  - ✓ Throttle : 5
  - ✓ Brake : 1

$$Loss\ function = \frac{1}{n}\sum_{i=0}^{n}(\mathrm{y}_{(\mathrm{i})} - \hat{y}_{(\mathrm{i})})^2\,(Weight)$$

Final Neural Network Details

- Image size : 75 * 320 * 3
- Total images : 17595
- Training set : 14252
- Validation set : 1583
- Testing set : 1759
- Loss = mean_squared_error
- metrics= mean absolute error
- Optimizer : Adam
- Batch size : 32
- Epoch : 100

Convolution Layers Details

Layer (type)	***Output Shape***	***Param #***	***Connected to***
main_input (InputLayer)	(None, 75, 320, 3)	0	
conv2d_185 (Conv2D)	(None, 75, 320, 12)	336	main_input[0][0]
max_pooling2d_185 (MaxPooling2D)	(None, 37, 160, 12)	0	conv2d_185[0][0]
conv2d_186 (Conv2D)	(None, 35, 158, 24)	2616	max_pooling2d_185[0][0]
max_pooling2d_186 (MaxPooling2D)	(None, 17, 79, 24)	0	conv2d_186[0][0]
conv2d_187 (Conv2D)	(None, 75, 320, 12)	6944	max_pooling2d_186
max_pooling2d_187 (MaxPooling2D)	(None, 8, 39, 32)	0	conv2d_187[0][0]
conv2d_188 (Conv2D)	(None, 4, 35, 64)	512645	max_pooling2d_187[0][0]
max_pooling2d_188 (MaxPooling2D)	(None, 2, 17, 64)	0	conv2d_188[0][0]
flatten_64 (Flatten)	(None, 2176)	0	max_pooling2d_188[0][0]
speed_input (InputLayer)	(None, 1)	0	
concatenate_44 (Concatenate)	(None, 2177)	0	speed_input[0][0], flatten_64[0][0]

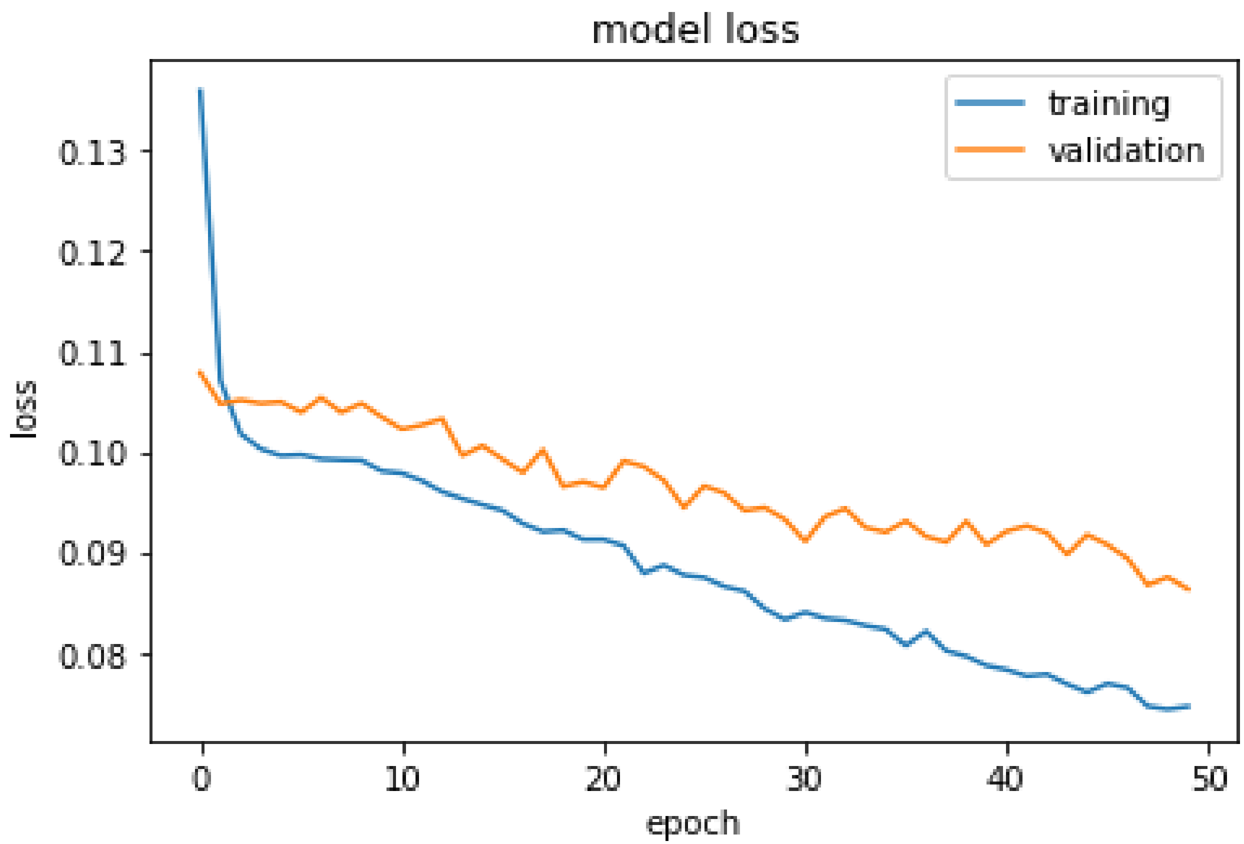
Convolution Layers Details

Layer (type)	***Output Shape***	***Param #***	***Connected to***
dropout_127 (Dropout)	(None, 2177)	0	concatenate_44[0][0]
dense_150 (Dense)	(None, 64)	139392	dropout_128[0][0]
dropout_128 (Dropout)	(None, 64)	0	dense_150[0][0]
dense_151 (Dense)	(None, 1)	65	dropout_128[0][0]
dense_152 (Dense)	(None, 1)	65	dropout_128[0][0]
dense_153 (Dense)	(None, 1)	65	dropout_128[0][0]
Total params: 200,747			
Trainable params: 200,747			
Non-trainable params: 0			

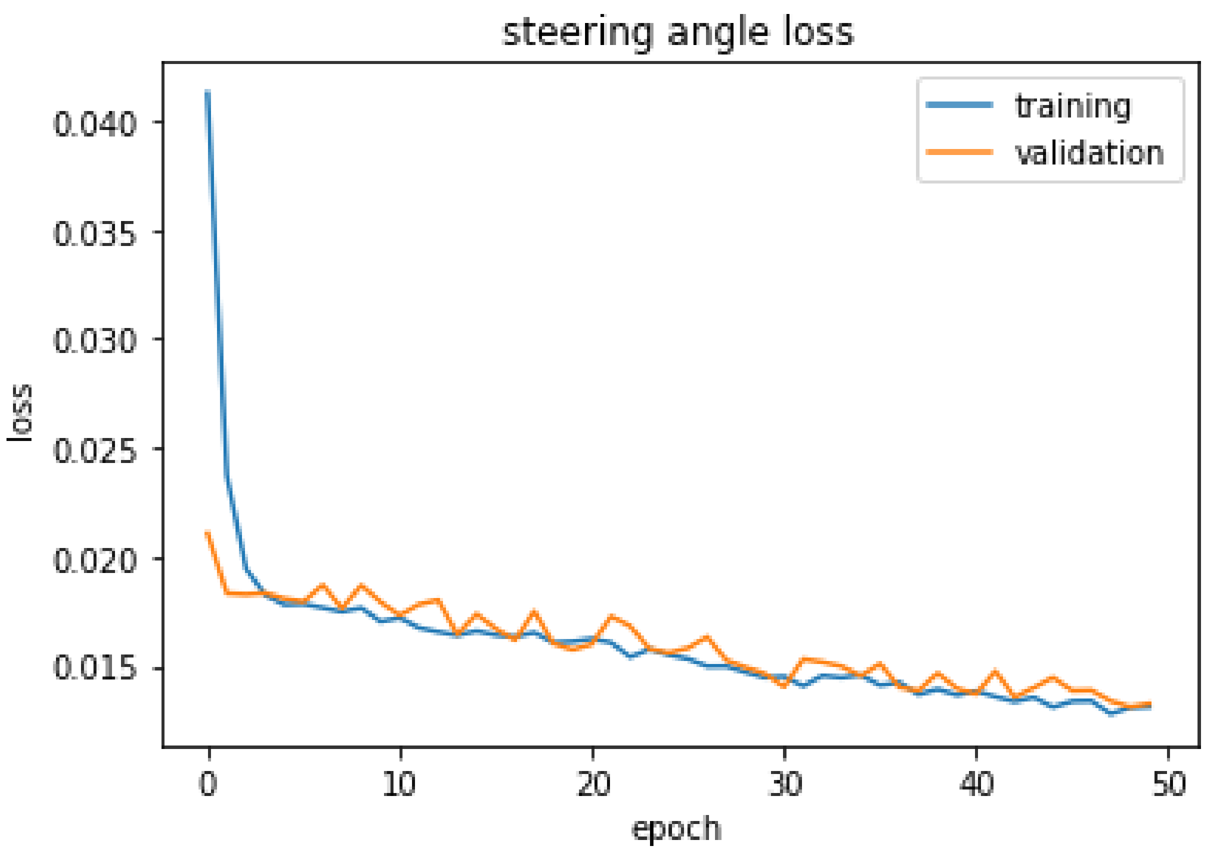
Agenda

- Introduction
- Tools Set-up
- Environment Generation
- Convolutional Neural Network
- Interface to Test the Model
- CNN Improvements
- Network Results
- Demo
- Outlook
- References

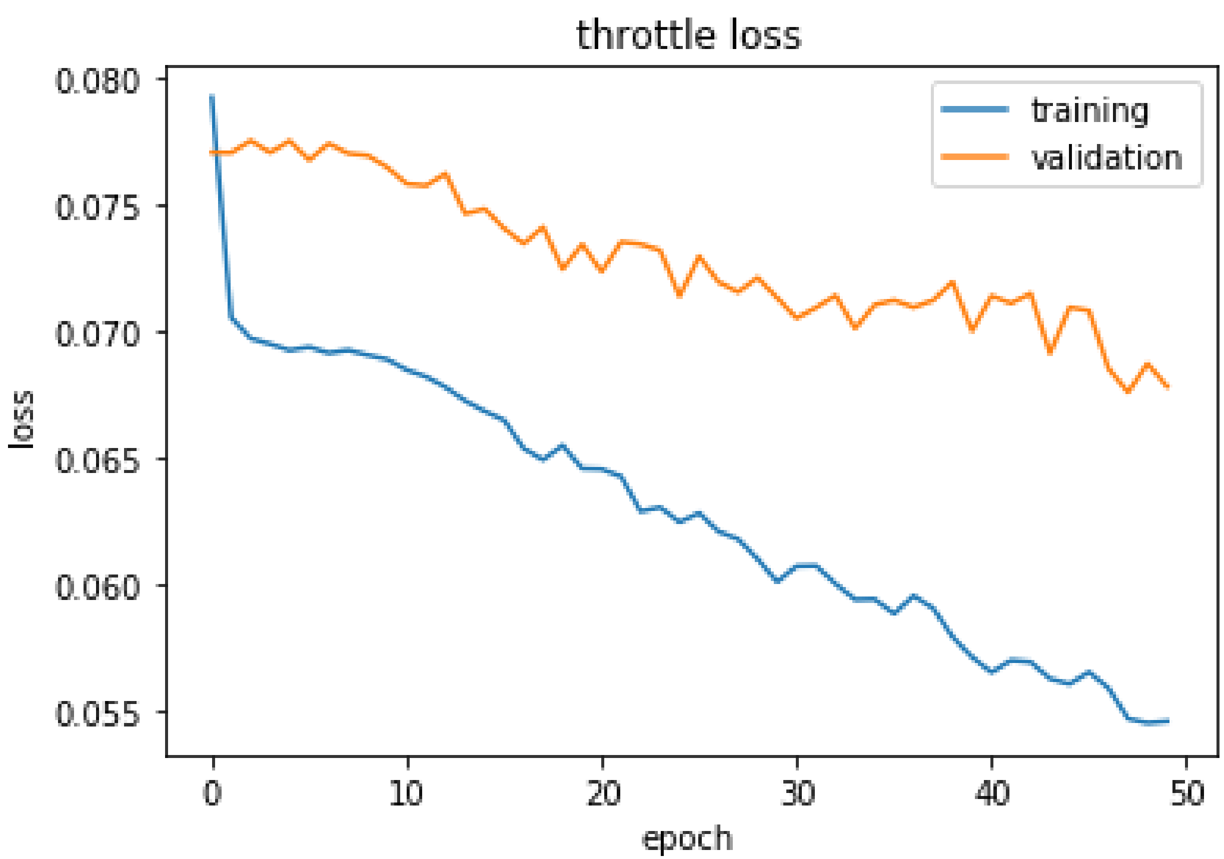
Network Results



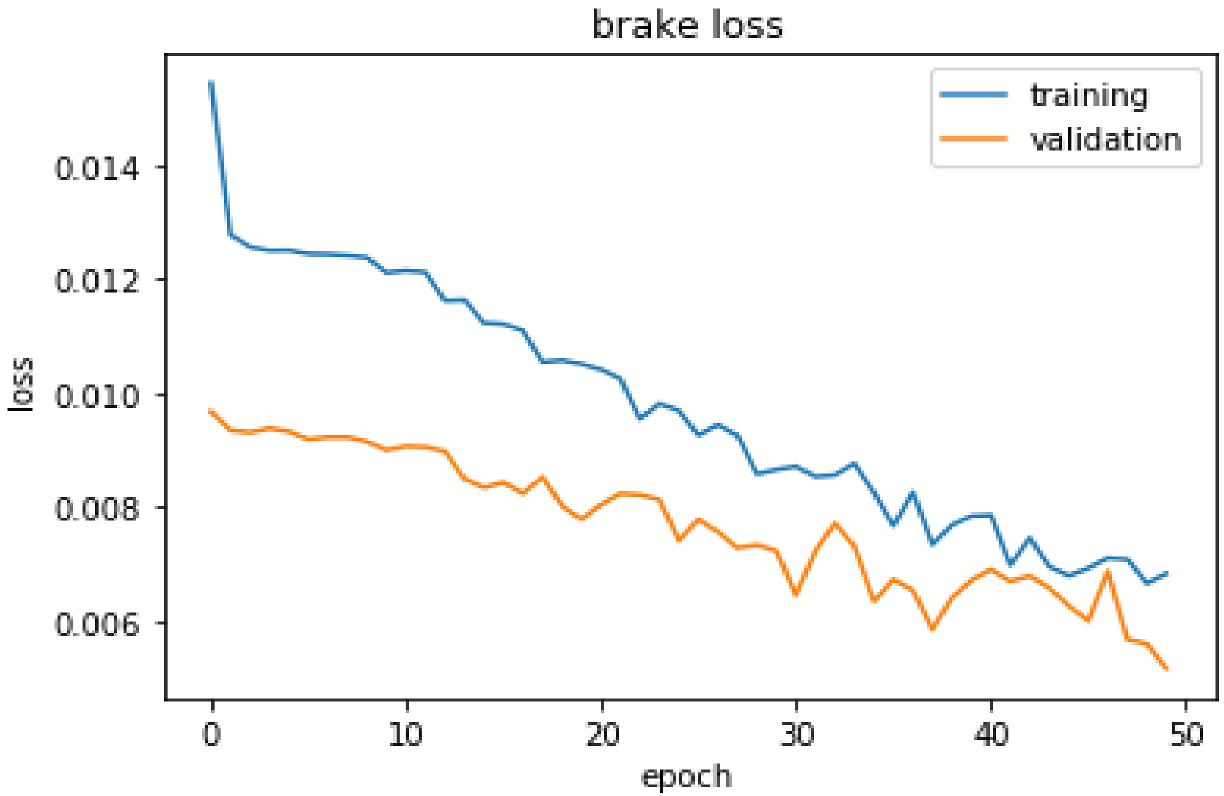
Network Results



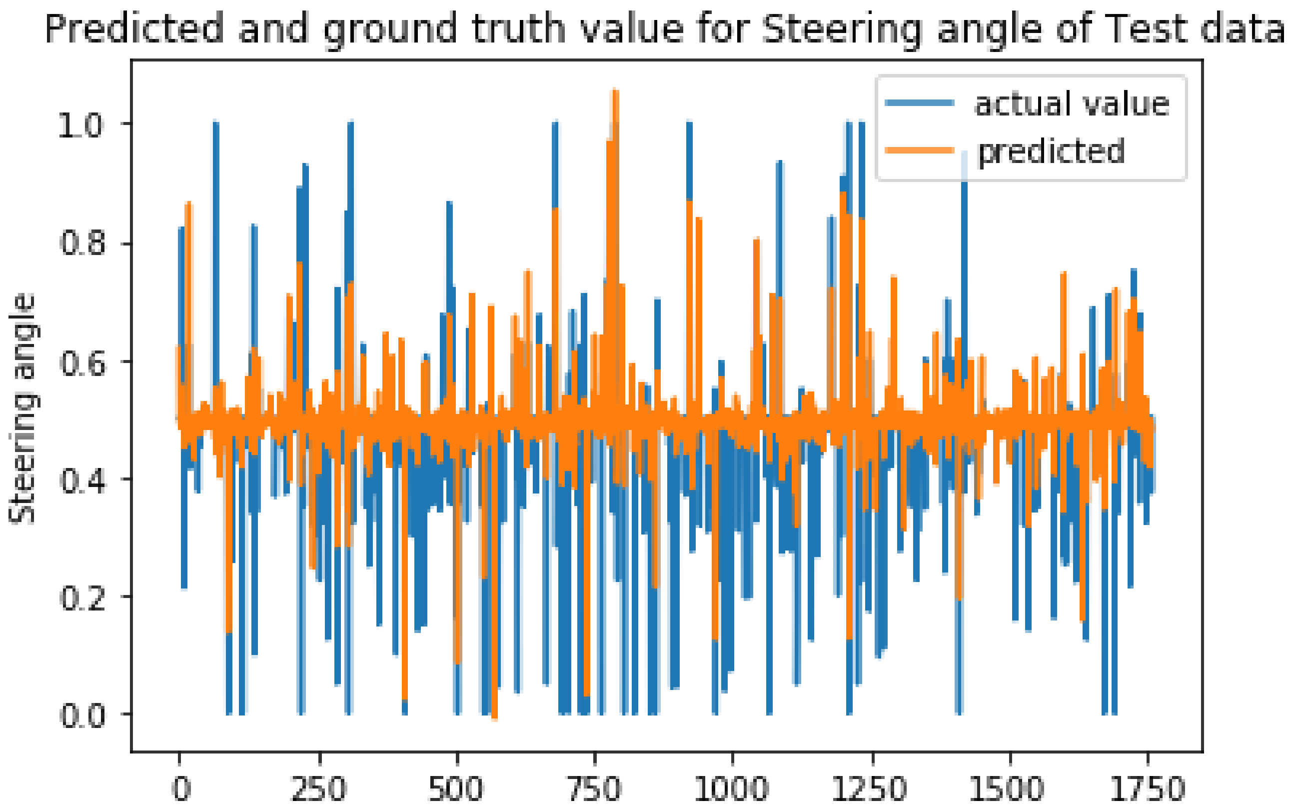
Network Results



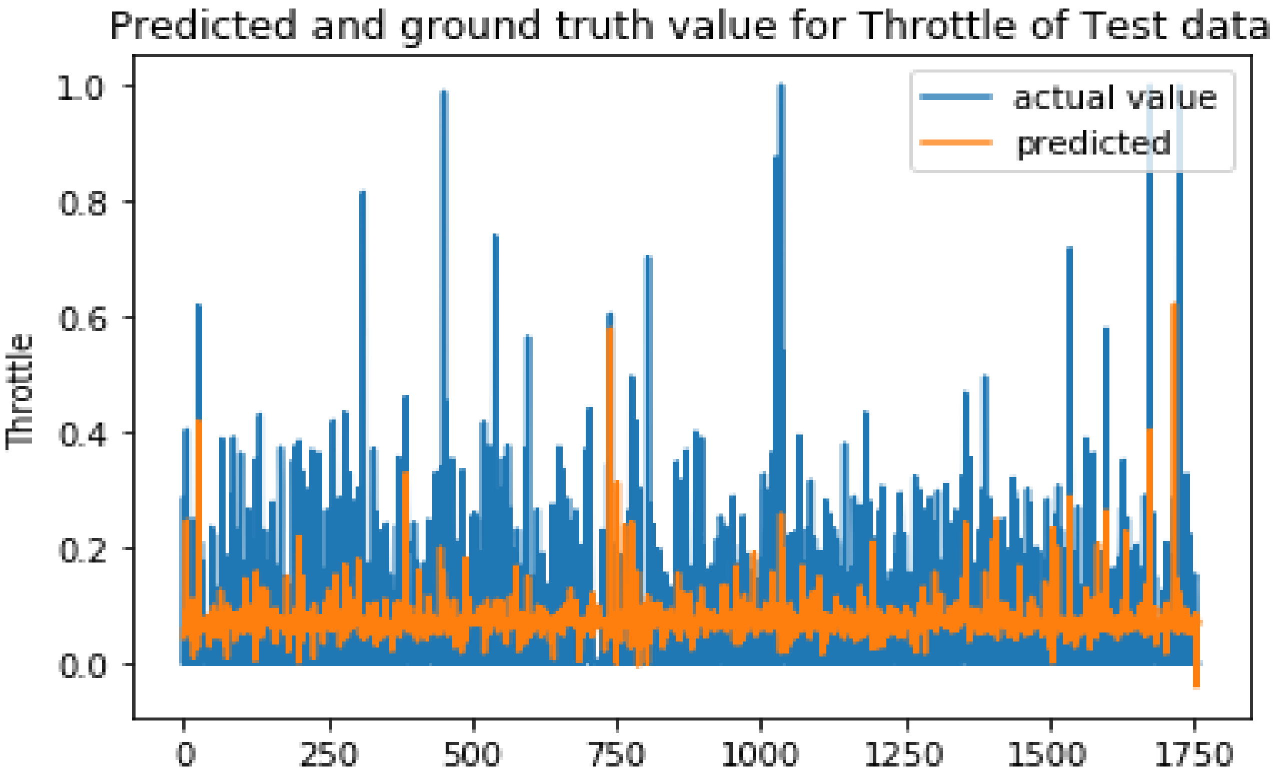
Network Results



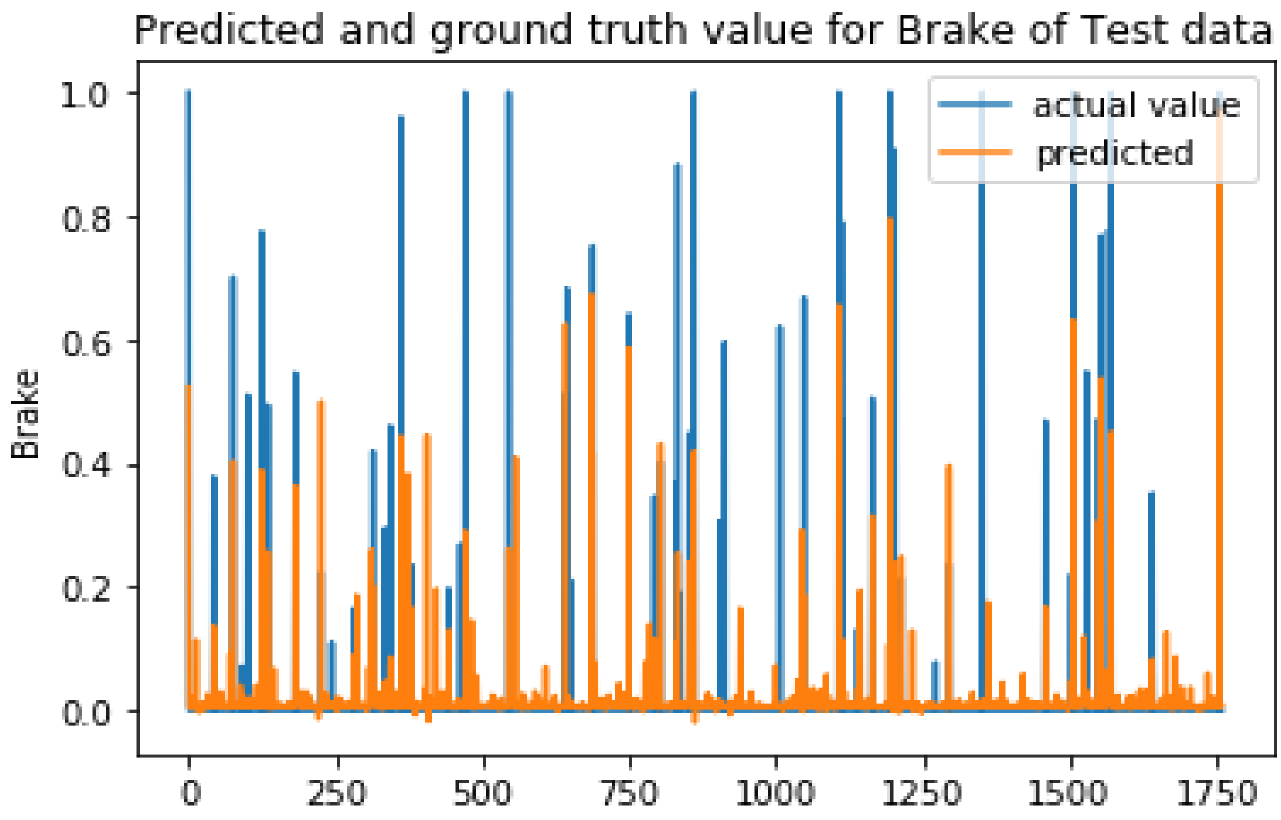
Network Results



Network Results



Network Results



Agenda

- Introduction
- Tools Set-up
- Environment Generation
- Convolutional Neural Network
- Interface to Test the Model
- CNN Improvements
- Network Results
- Demo
- Outlook
- References

Demo

Agenda

- Introduction
- Tools Set-up
- Environment Generation
- Convolutional Neural Network
- Interface to Test the Model
- CNN Improvements
- Network Results
- Demo
- Outlook
- References

Outlook

- To train the car with the new generated environment
- To create an environment without any steps on the corner of the road
- To use the images from left and right cameras and other additional sensors or elements
- To find the right amount of weight to be added to the throttle
- To train the car with more data to get the best-trained model
- Implementing a recurrent neural network

Thank You!

Prof. Dr.
Massimo Fornasier
TUM Data
Innovation Lab

Dr. Ricardo
Acevedo Cabra
TUM Data
Innovation Lab

Dr. Stefan Held
ITK Engineering
GmbH

Felix Wempe,
M.Sc.
ITK Engineering
GmbH

Johannes Klotz,
M.Sc.
ITK Engineering
GmbH

References

[1] daimler.com. (2018). Daimler - Home. [online] Available at: https://www.daimler.com/innovation/autonomous-driving/special/definition.html [Accessed 14 Feb. 2018].

[2] Python 3.5 Documentation (2018). Python documentation and libraries. [online] Available at: https://docs.python.org/3/library/ [Accessed 14 Feb. 2018].

[3] Udacity self-driving-car-simulator project at GitHub (2018). Udacity/self-driving-car-sim. [online] Available at: https://github.com/udacity/self-driving-car-sim [Accessed 14 Feb. 2018].

[4] Unity game engine (2018). Unity3D - Home. [online] Available at: https://unity3d.com/ [Accessed 14 Feb. 2018].

[5] TensorFlow.com (2018). TensorFlow - Home. [online] Available at: https://www.tensorflow.org/ [Accessed 14 Feb. 2018].

[6] OpenDRIVE 1.4 Specifications Document (2018). openDRIVE® - Home. [online] Available at: http://www.opendrive.org/docs/OpenDRIVEFormatSpecRev1.4H.pdf [Accessed 14 Feb. 2018].

[7] OpenRoadEd.net (2018). OpenRoadEd - Home. [online] Available at: https://sourceforge.net/projects/openroaded/ [Accessed 14 Feb. 2018].

[8] keras.io (2018). Keras - Home. [online] Available at: https://keras.io/ [Accessed 14 Feb. 2018].

[9] CS231n: Convolutional Neural Networks for Visual Recognition (Spring 2017). Courses at Stanford University. [online] Available at: http://cs231n.stanford.edu/index.html [Accessed 14 Feb. 2018].

[10] Deeplearning4J.org (2018). Deep Learning Tutorials. [online] Available at: https://deeplearning4j.org/logistic-regression [Accessed 14 Feb. 2018].

[11] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

[12] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural tuning machines. arXiv preprint arXiv:1410.5401, 2014.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012.

[14] Rising, L., & Janoff, N. S. (2000). The Scrum software development process for small teams. IEEE Software, 17(4), 26-32.

Convolution and Maxpooling layers