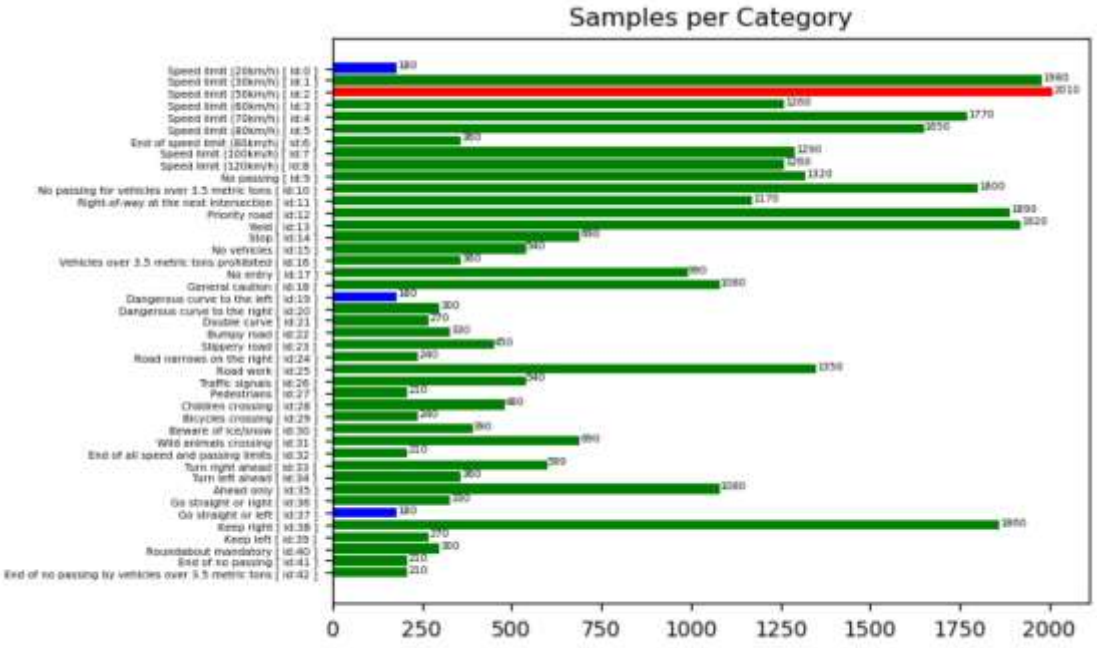
UDACITY Traffic_Sign_Classifier Project

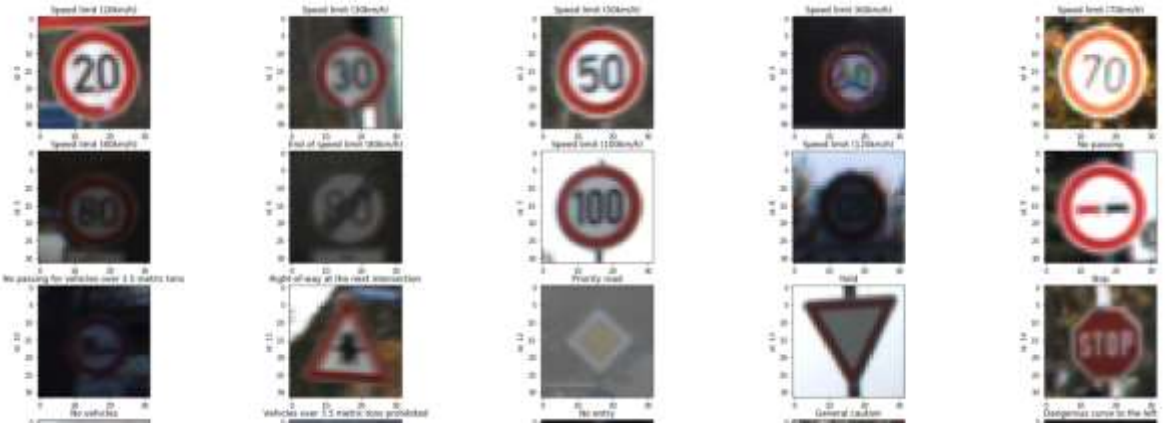
Overview of the process choosen

1. **Load The Data**
2. **DataSet Summary & Exploration**
   2.1 Retrieve Summary from DataSet
   2.2 Visualizing oft the DataSet
   Viewing the Samples per Category distribution



2.3 *Showing Images of the DataSet*

Showing some random images from the provided data set



3. **Design an Test a Model Architecture**

3.1 PreProcess the DataSet

3.1.1 Shuffle the Data

To shuffle the training data is a good practice to avoid biased learning

3.1.2 Prepare the Images

Normalize the images and convert the images from RGB to GRAYSCALE

3.2 *Define the Model*

Implementation of the LeNet function

Adams Optimizer has been choosen, as it is known for minimizing the loss function

Hyperparameters as in the submitted solution have been found by testing some variants, changed learning rate, mu and numbers of learning cycles (EPOCHS).

e.g. using the given learning rate of 0.0009, keep propability of 0.7 and a stdev of 0.01 using EPOCHS > 39 did not improve the accuracy.

3.3 Train, Validate and Test the Model

3.3.1 Features & Labels

3.3.2 Training Pipeline

3.3.3 Evaluation of the Model

Definition of the Evaluation function

3.3.4 Training of the Data Model

3.3.5 Evaluate the Trained Model Using Test Data

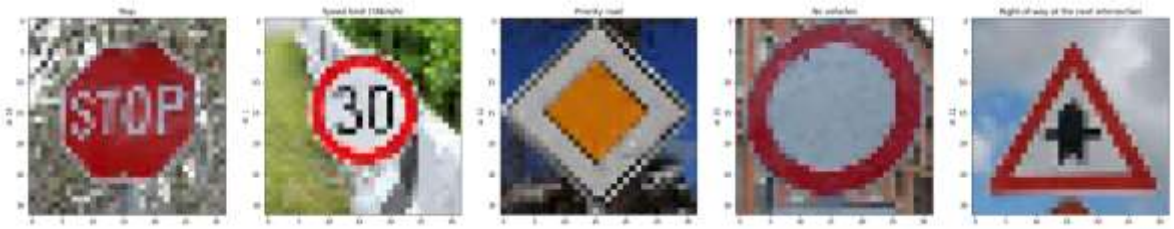
4. Test A Model On New Images

5 Images found on the internet used to verify the accuracy of the solution.

Type	*Image*	*Possible Issues*
14-Stop		*Easy*
15-No-vehicles		*Rather small but still easy*
1-Speed-limit-30-km-h		*Wall in the background might have impact*
11-Right-of-way-at-the-next-intersection		*Easy*
12-Priority-road		*Rather small but still easy*

4.1 Load And Output The Images

Show the 5 internet found images



4.2 Predict The Sign Type For Each Image

4.3 Analyze Performance

4.4 Output Top 5 Softmax Propabilities For Each Image Found On The Web

Summary

Attribution :

- Book : « Neuronale Netze selbst Programmieren » (Tariq Rashid)
- Google and GitHub: various samples reviewed to better unerstand viewing algorithms used
- Udacity: e.g. source for the LeNet algorithm