

[登出](#)

## 项目

## Traffic Sign Classification

此部分属于 Self Driving Car Engineer Nanodegree Program

项目审阅

代码审阅

注释

## Requires Changes

SHARE YOUR ACCOMPLISHMENT

还需满足 1 个要求 变化



Lately on slack few students asked for a good Deep Learning book. So here's a book which is also recommended by Elon Musk:

- Deep Learning (Adaptive Computation and Machine Learning series) [Github](#) and on [Amazon](#)
- [Fast.ai](#)
- [A Guide to Deep Learning](#)

Really nice job with this first submission. Your next will certainly be the ONE. Keep up the good work !!!

## Files Submitted

The project submission includes all required files.

## Dataset Exploration

The submission includes a basic summary of the data set.

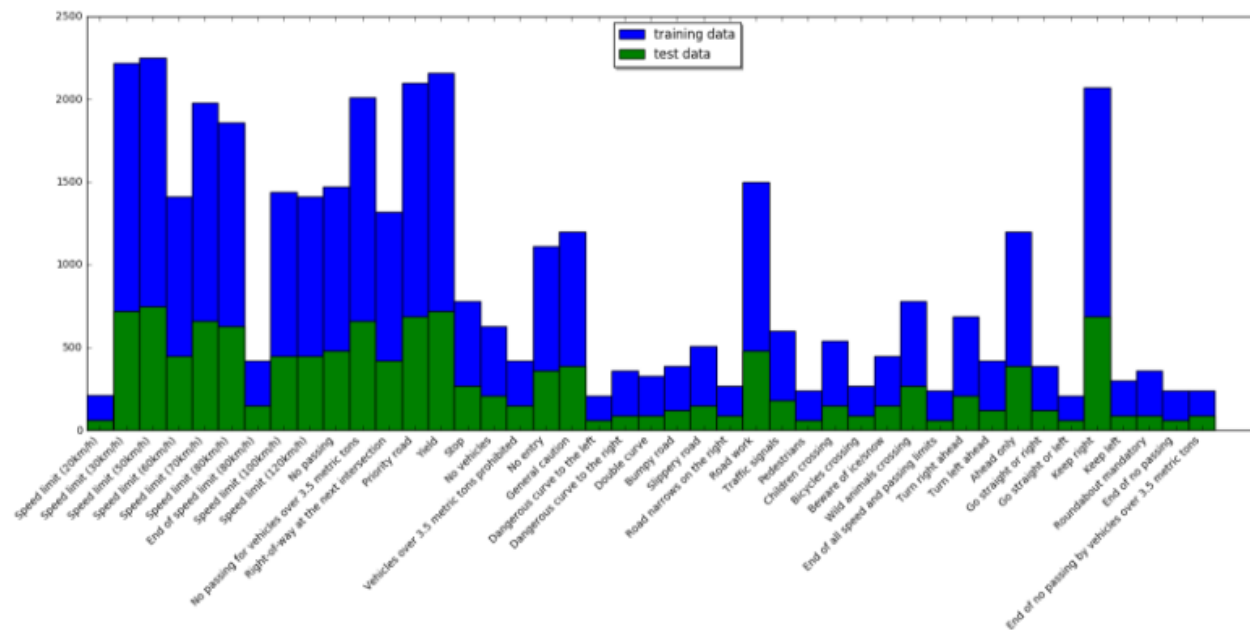
- A basic summary statistics of the dataset has been printed out
- Nice job !!!

The submission includes an exploratory visualization on the dataset.

- Very nice job with the visualizations here

## Suggestion

- Here's a histogram showing the distribution inside both the training and the testing set



## Design and Test a Model Architecture

The submission describes the preprocessing techniques used and why these techniques were chosen.

- Preprocessing steps taken have been discussed and justified

### Suggestions & Comments

Here are some preprocessing techniques that have proven to work on this dataset (some of which you did think of so good job on that )

- Performing translations, scalings and rotations of the training set considerably improves generalization
- Values for translation, rotation and scaling can be drawn from a uniform distribution in a specified range, i.e.  $\pm T$  % of the image size for translation,  $1 \pm S/100$  for scaling and  $\pm R^\circ$  for rotation.
- A normalization method that yields very impressive results is the Contrast-limited Adaptive Histogram Equalization (CLAHE)
- Normalization can help in case of High contrast variation among the images
- More preprocessing steps could be included
- These resources ([1](#) & [2](#) & [3](#)) might provide some more intuition on the subject here.

The submission provides details of the characteristics and qualities of the architecture, including the type of model used, the number of layers, and the size of each layer. Visualizations emphasizing particular qualities of the architecture are encouraged.

Sufficient details about the model's architecture is provided

### Suggestions & Comments

- In order to further improve this work, it would be better to visualize the architecture that is adopted. And [TensorBoard](#) is a very good visualization tool.

The submission describes how the model was trained by discussing what optimizer was used, batch size, number of epochs and values for hyperparameters.

## Suggestions & Comments

- Using StratifiedShuffleSplit over the traditional train–test–split, enables the use of the entire training data while keeping data for cross–validation: the distribution of the labels in both would be similar. Around 1% of the data can be used for the cross–validation set.
- [Here's a nice discussion](#) on how to choose the batch\_size of Stochastic Gradient Decent
- And a [Discussion on Adam Optimizer](#)

f

The approach taken is nicely discussed. I left some comments below.

## Suggestions & Comments

Here are few questions to think about while designing the architecture:

- How would I choose the optimizer? What is its Pros & Cons and how would I evaluate it?
- How would I decide the number and type of layers?
- How would I tune the hyperparameter? How many values should I test and how to decide the values?
- How would I preprocess my data? Why do I need to apply a certain technique?
- How would I train the model?
- How would I evaluate the model? What is the metric? How do I set the benchmark?
- In case you're interested, here's an interesting Inception's [example](#).

## Test a Model on New Images

The submission includes five new German Traffic signs found on the web, and the images are visualized. Discussion is made as to particular qualities of the images or traffic signs in the images that are of interest, such as whether they would be difficult for the model to classify.

## Required

- Nice job printing out the images here but you also are required to discuss before doing the actual prediction, what qualities your new images have (e.g: brightness, contrast, etc. ) that might cause your model to misclassify them.

**The submission documents the performance of the model when tested on the captured images. The performance on the new images is compared to the accuracy results of the test set.**

No, the model performs worse on my test data than the given test data set

- A fair comparison has been made between the prediction accuracy of the model on the captured images and those on the testing set.
- To increase the performance of your model on unseen data, I suggest you include more preprocessing into your pipeline. I have provided you above with a list of preprocessing techniques that work quite well with this dataset

**The top five softmax probabilities of the predictions on the captured images are outputted. The submission discusses how certain or uncertain the model is of its predictions.**

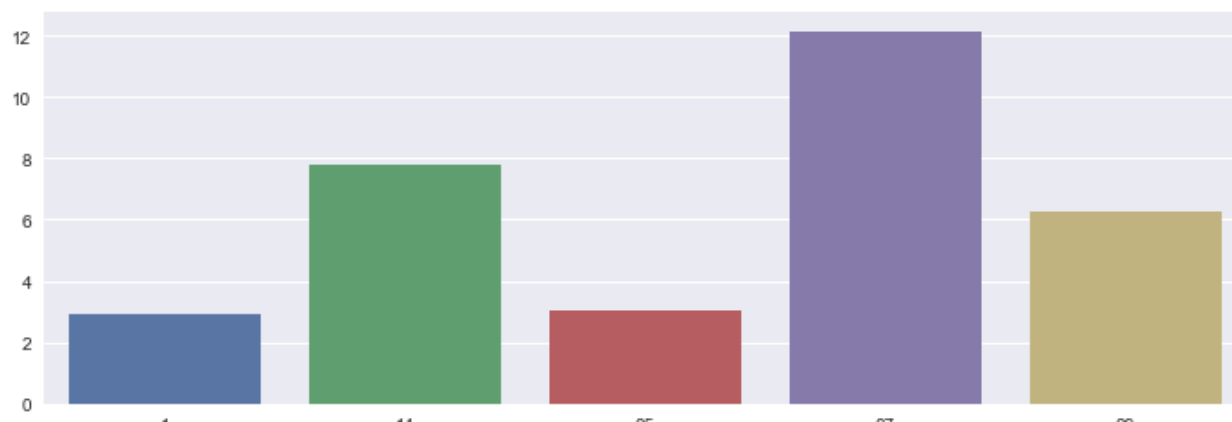
- Nice job visualizing the softmax probabilities here
- And discussing how certain/uncertain your model is of its predictions

## Suggestions & Comments

- Here is an example code from a student for visualizing the softmax probabilities:

```
### Print out the top five softmax probabilities for the predictions on the German traffic sign images
### Feel free to use as many code cells as needed.
for i in range(6):
    print(new_images[i])
    print("Predicted Class: {} | True Class: {}".format(preds[i], y_new[i]))
    fig = plt.figure(figsize=(12,4))
    sns.barplot(x=top_k[1][i], y=top_k[0][i])
    plt.show()
```

- And this is an example output from the code above:



🔄 重新提交项目

📄 下载项目

了解 [修改和重新提交项目的最佳做法](#)。

[返回](#) [PATH](#)

---

[学员 FAQ](#)