

Traffic Sign Classifier Project

Build a Traffic Sign Recognition Project

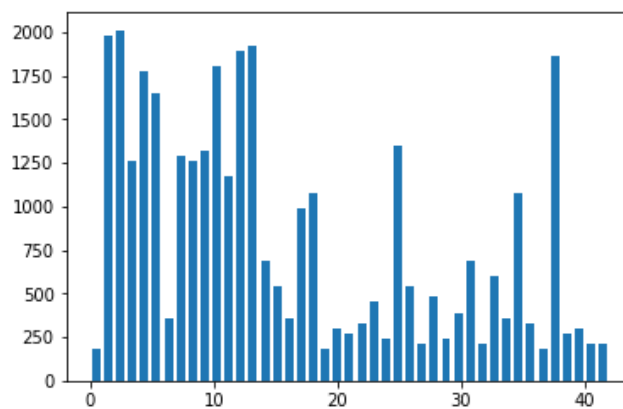
- Dataset Exploration

- Dataset Summary:

- The data contains 34799 samples for the training set, 12630 samples for the test set;
 - There is 43 unique classes/labels in the dataset;
 - Each sample is a 32x32x3 color image

- Exploratory Visualization

- The following is the histogram of the sample occurrence in each 43 sign category

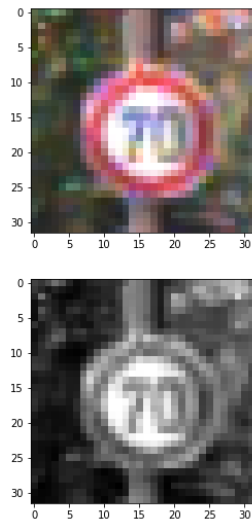


- The following are the image samples for first 40 sign categories (with category its number on top) :



- Design and Test a Model Architecture

- Preprocessing: As the brightness and illumination condition vary quite a lot from sample to sample. It is beneficial to do normalization to make sure the image properties are more consistent across the samples. I first converted the color image to grayscale image, and then normalized the brightness of the grayscale images to between -1 and 1. The following are the color and gray scale sample images.



- Model Architecture:

My chose a model architecture similar to LeNet. The final model consisted of the following layers:

Layer	Description
Input	32x32x1 normalized grayscale image
Convolution 5x5x6	1x1 stride, same padding, outputs 28x28x6
RELU	
Max pooling 2x2	1x1 stride, outputs 14x14x6

Convolution 5x5x16	1x1 stride, same padding, outputs 10x10x16
RELU	
Max pooling 2x2	1x1 stride, outputs 5x5x16
Flatten	Input 5x5x6; output: 400
Fully connected	input: 400; output 120
RELU	
Dropout	Keep 50%
Fully connected	input: 120; output 43
Softmax	

The optimizer is Adam optimizer, the batch size is 100, I tried 10, 30 and 60 epochs, learning rate is set to 0.001. I tried started the training without dropout layer, and shuffle the training data, after 60 epochs of training. The accuracy of on the test set is ~91.6%. Then I shuffled the training data, the test accuracy improved to ~92.9%. I added one dropout layer (with 50% dropout rate) between the two fully-connected layer, the test accuracy improved to ~93.7%.

My final model results were:

- validation set accuracy of 94~95% ?
- test set accuracy of 93~94%

- Test a Model on New Images

- Here are five German traffic signs that I found on the web:



1.jpg



2.jpg



3.jpg



4.jpg



5.jpg

- I converted them into 32x32 gray scale images:



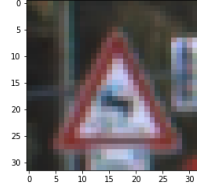
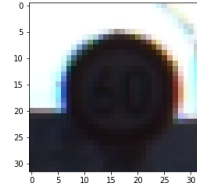
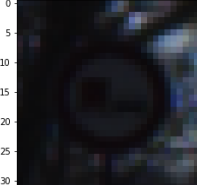
The following are the results of the prediction vs. the label. The model correctly predict 4 out of 5 traffic sign (80% accuracy).

Image label category	Prediction category
3 (60km/h)	3
11 (Arrow)	11
29 (Bike sign)	23
25 (Road work)	25
12 (Diamond sign)	12

The one traffic sign the model predicted wrong is the “bike sign”:



The top five softmax probabilities for the wrongly predicted “bike sign” (category 29) were:

Probability	Predicted category	Sample image
0.998	23	
0.0016	3	
0.000039	10	
0.000011	31	
0.000011	5	

The model has very high confidence to predict the bike sign (#29) to be #23. Both sign (#29 and #23) have triangular shape, which probably was captured by the model. However, the challenge, I think, comes from the “bike” in the “bike” sign has very fine feature, which can not be fully represented in the training data due to the limited 32x32 image resolution.