Traffic Sign Recognition

Build a Traffic Sign Recognition Project

The goals / steps of this project are the following:

- Load the data set (see below for links to the project data set)
- Explore, summarize and visualize the data set
- Design, train and test a model architecture
- Use the model to make predictions on new images
- Analyze the softmax probabilities of the new images
- Summarize the results with a written report

Rubric Points

Here I will consider the rubric points individually and describe how I addressed each point in my implementation.

Data Set Summary & Exploration

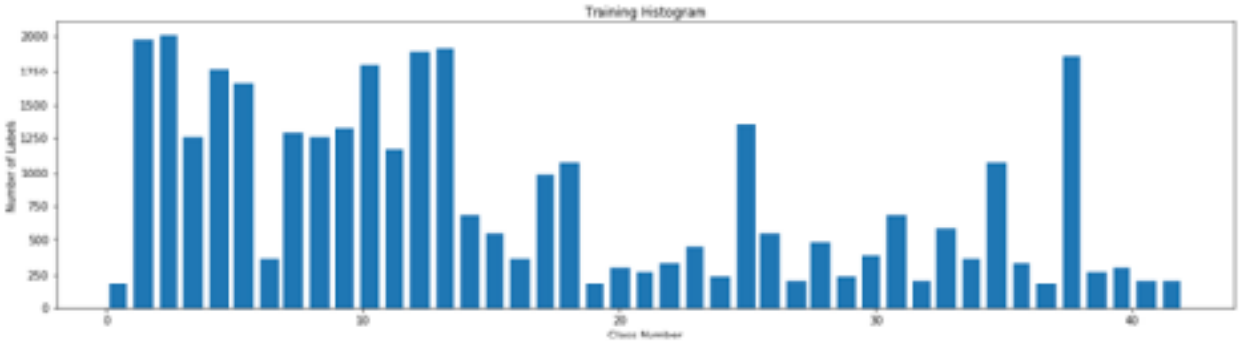
For this I used used the numpy and pandas library to calculate the statistics of the traffic signs dataset:

- The size of training set is: 34.799
- The size of the validation set is: 4,410

- The size of test set is: 12630
- The shape of a traffic sign image is: 32,32,1
- The number of unique classes/labels in the data set is: 43

2. Include an exploratory visualization of the dataset.

Here is a histogram showing the number of labels per class.



Here are five random images from the dataset.

Design and Test a Model Architecture

1. Describe how you preprocessed the image data. What techniques were chosen and why did you choose these techniques? Consider including images showing the output of each preprocessing technique. Pre-processing refers to

techniques such as converting to grayscale, normalization, etc. (OPTIONAL: As described in the "Stand Out Suggestions" part of the rubric, if you generated additional data for training, describe why you decided to generate additional data, how you generated the data, and provide example images of the additional data. Then describe the characteristics of the augmented training set like number of images in the set, number of images for each class, etc.)

I converted the image to grayscale as color is not important and RGB images contain more data and could make the model more complicated.

Next was the normalization process by (pixel - 128) / 128 to allow better consistency for the network to train.

2. Describe what your final model architecture looks like including model type, layers, layer sizes, connectivity, etc.) Consider including a diagram and/or table describing the final model.

My final model consisted of the following layers:

Layer	Description
Input	32x32x1 Grayscale image
Convolution 5x5	1x1 stride, valid padding, outputs 28x28x6
RELU	
Max pooling	2x2 stride, valid padding, outputs 14x14x6
Convolution 5x5	1x1 stride, valid padding, outputs 10x10x16
RELU	
Max pooling	2x2 stride, valid padding, outputs 5x5x16.
Flatten	Input = 5x5x16, Output = 400
Fully connected	Input = 400, Output = 120.
RELU	
Fully connected	Input = 120, Output = 84.
RELU	
Fully connected	Input = 84, Output = n_classes.

3. Describe how you trained your model. The discussion can include the type of optimizer, the batch size, number of epochs and any hyperparameters such as learning rate.

To train the model, I used the same approach we learned in the lessons using TensorFlow's softmax cross entropy and the Adam

optimizer. I used a batch size of 128, epochs of 50, and my learning rate was 0.001.

4. Describe the approach taken for finding a solution and getting the validation set accuracy to be at least 0.93. Include in the discussion the results on the training, validation and test sets and where in the code these were calculated. Your approach may have been an iterative process, in which case, outline the steps you took to get to the final solution and why you chose those steps. Perhaps your solution involved an already well known implementation or architecture. In this case, discuss why you think the architecture is suitable for the current problem.

My final model results were:

- training set accuracy of .999
- validation set accuracy of .942
- test set accuracy of .938

I used the LeNet architecture do to my knowledge of it from the lessons and I have very little experience with other neural networks. I added a dropout to the second Convolutional layer but that is the only change to the the basic architecture. LeNet is a small and simple neural network and is very good with image classification and for the most part the traffic signs are not very complicated. The model's accuracy on the training, validation, test set and low margin for error confirm the model is working well.

Test a Model on New Images

1. **Choose five German traffic signs found on the web and provide them in the report. For each image, discuss what quality or qualities might be difficult to classify.**

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





The images are all better than the images in the dataset with the exception of the last. I tried these first four images to see how well the model would work on bright images and then added the fifth image that is not as clear for comparison.

2. Discuss the model's predictions on these new traffic signs and compare the results to predicting on the test set. At a minimum, discuss what the predictions were, the accuracy on these new predictions, and compare the accuracy to the accuracy on the test set (OPTIONAL: Discuss the results in more detail as described in the "Stand Out Suggestions" part of the rubric).

Here are the results of the prediction:

Image	Prediction
60 km/h	60 km/h
Left turn	Left turn
Straight or right	Straight or right
Roundabout	Roundabout
Yield	Yield

I was able to achieve 100% accuracy predicting all 5 images correctly compared to the test set which was 93.8% but that was also on a much larger number of 12,630 samples.

3. Describe how certain the model is when predicting on each of the five new images by looking at the softmax probabilities for each prediction. Provide the top 5 softmax probabilities for each image along with the sign type of each probability. (OPTIONAL: as described in the "Stand Out Suggestions" part of the rubric, visualizations can also be provided such as bar charts)

The code for making predictions on my final model is located in the 13th cell of the notebook.

Summary of predictions

Probability	Prediction
0.99	60 km/h
0.99	Left turn
0.99	Straight or right
0.99	Roundabout
1.00	Yield

Image 1: 60km/h

Probability	Prediction
0.99	60 km/h

0.20	50 km/h
0.76	30 km/h
0.12	No passing for vehicles over 3.5 metric tons
0.78	80 km/h

Image 2: Left turn

Probability	Prediction
0.99	Left turn
0.11	Right-of-way at the next intersection
0.34	Beware of ice/ snow
0.12	Keep right
0.48	Slippery road

Image 3: Straight or right

Probability	Prediction
0.99	Straight or right
0.20	No passing
0.76	Ahead only

0.12	Turn left
0.78	Road work

Image 4: Roundabout

Probability	Prediction
0.99	Roundabout
0.43	Keep right
0.13	Traffic signals
0.42	No passing for vehicles over 3.5 metric tons
0.26	Dangerous curve to the right

Image 5: Yield

Probability	Prediction
1.00	Yield
0.20	60 km/h
0.73	Keep right
0.36	20 km/h
0.89	No vehicles

(Optional) Visualizing the Neural Network (See Step 4 of the Ipython notebook for more details)

1. Discuss the visual output of your trained network's feature maps. What characteristics did the neural network use to make classifications?