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.

Writeup / README

1. Provide a Writeup / README that includes all the rubric points and how you addressed each one. You can submit your writeup as markdown or pdf. You can use this template as a guide for writing the report. The submission includes the project code.

You're reading it! and here is a link to my project code

Data Set Summary & Exploration

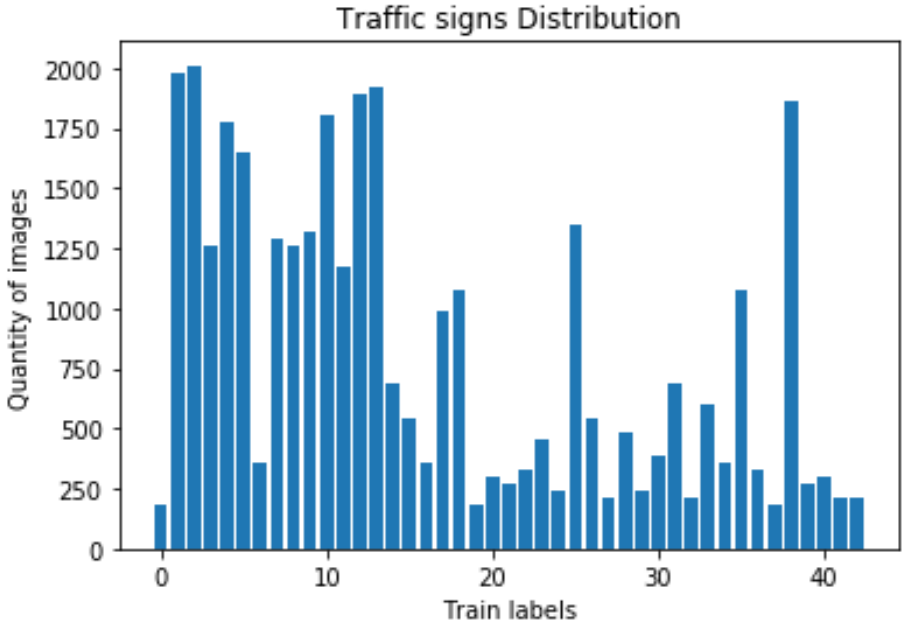
1. Provide a basic summary of the data set. In the code, the analysis should be done using python, numpy and/or pandas methods rather than hardcoding results manually.

I used the pandas library to calculate summary statistics of the traffic signs data set:

- The size of training set is 34799
- The size of the validation set is 12630
- The size of test set is 4410
- The shape of a traffic sign image is 32x32x3
- The number of unique classes/labels in the data set is 43

2. Include an exploratory visualization of the dataset.

Here is an exploratory visualization of the data set. It is a bar chart showing the label counts. The dataset is unbalanced, which might biais the prediction.



The following figure shows the distribution of the pixels for the initial train set :

Design and Test a Model Architecture

1. Image Preprocessing

As a first step, I have built a 1 dimensional Convolution in applying the grayscale images, but the validation accuracy is not that much related to 3dimensional convolution which is 80-90% of validation accuracy. I tried to generate additional images using rotations and noise blurring, in order to rebalance the train set.

As a last step, I normalized the image data because the learning speed increases if the variance of each coordinate of the input vector is equal to 1.

2. Model description

My final model consisted of the following layers:

Layer	Description

Layer	Description
Input	36x36x3 RGB image
Convolution 3x3	1x1 stride, VALID padding, outputs 32x32x16
RELU	
Convolution 3x3	1x1 stride, VALID padding, outputs 28x28x32
RELU	
Max pooling	3x3 kernel,1*1 stride, VALID padding, outputs 26x26x
Convolution 3x3	1x1 stride, VALID padding, outputs 26x26x16
RELU	
Max pooling	2x2 stride, outputs 11x11x16
Flatten	outputs 1936x1
Fully connected	548 nodes, outputs 548x1
Fully connected	120 nodes, outputs 120x1
Fully connected	84 nodes, outputs 84x1
Fully connected	43 nodes, outputs 43x1
Softmax	

It is very similar to the LeNet neural network. The differences are :

- An additional 2 convolutional layers have been added with minute change of pooling layers in between the layers.
- Larger fully connected layers in order to capture more features, because the number of possibles classes is higher (43 instead of 10).

3. Model training

To train the model, I used the Adam Optimizer. The model ran 20 epochs.

I have tried to run the model for 10 epochs only, but the model is still learning(even after 9^{th} epoch validation accuracy is still increasing). So I tried to increase the epoch count, but after some stage model is generalizing to overfit and reducing accuracy, so I end up with 20 epochs in training.

Batch size I tried to keep as low as possible, as GPU doing good computational effort in learning the model.

Learning rate 0.0001 and batch_size=32 is making the model to be over fit on the training data of 100% accuracy on training, but the test loss observed is very less like 84 % . So I have decided to use Learning rate of 0.001 and batch size of 128.

Layers depth also, first I have tried of using same LeNet architecture, but the learnable feature of model are high, which can't be learnt by 7 layers, so I have tried increasing it to 9 layers depth of Neural Network.

4. Model selection

- What was the first architecture that was tried and why was it chosen?

  The LeNet convolutional Architecture is the first one, I have tried on Grayscale images,

- What were some problems with the initial architecture?

  Gray scale images could'nt carry much information in remembering the patterns of traffic sign images.

- How was the architecture adjusted and why was it adjusted? Typical adjustments could include choosing a different model architecture, adding or taking away layers (pooling, dropout, convolution, etc), using an activation function or changing the activation function.

I adjusted the network by adding extra two Convolutional layers in remembering the patterns and lowering the Neurons count layer by layer till the last layer.

One common justification for adjusting an architecture would be due to overfitting or underfitting. A high accuracy on the training set but low accuracy on the validation set indicates over fitting; a low accuracy on both sets indicates under fitting.

- Which parameters were tuned? How were they adjusted and why?

  Neurons count and layers were adjusted, Neurons count reduced gradually from Fully connected layers till it reaches the End of predictions layer.

- What are some of the important design choices and why were they chosen? For example, why might a convolution layer work well with this problem? How might a dropout layer help with creating a successful model?

  Convolutional Networks in modeling are important designs in building traffic signal classifier, because the Convolutional Layers are well defined in remembering and sharing the weights information through out the network.

  Drop out helps effectively in reducing the over fitting of network, it will try to stop the learnability of some neurons in network randomly epoch by epoch.

My final model results were:

- training set accuracy of 1.00
- validation set accuracy of 0.9780
- test set accuracy of 0.9615

  The test accuracy is 96%, since the data is unseen, it proves that the model works well. However, the train accuracy of 1 suggest that the model is overfitting to the train set.

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:

5 Random traffic signs downloaded from internet and tried to test with the model,

Model could not perform on all variations of data as it was not trained well on the clear images. For some labels it is performing very well, but in case of speed limit related images, model performing worse.

There is need of unbiased data set of large speed limit variation images. As I can't download well defined data, I tried to build the model best in learning as maximum as possible.

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
Stop	Stop
Up right	Roundabout mandatory
Wild Animal crossing	Wild Animal crossing

Image	Prediction
No entry	No entry
Yield	Yield

The model was able to correctly guess 4 of the 5 traffic signs, which corresponds to an accuracy of 80%. On the other hand,

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)

For the first image, the model is sure that this is a No entry sign (probability of 1), and the image does contain a No entry sign. The top five soft max probabilities were

Probabilities for specific Image -- [1. 0. 0. 0. 0.]
Sign names of probabilities -- Stop, No entry, Yield, Priority road, Keep right

Probabilities for specific Image -- [0.95999998 0.038 0.001 0. 0.]
Sign names of probabilities -- Roundabout mandatory, Go straight or right, Traffic signals, End of no passing, Priority road

Probabilities for specific Image -- [1. 0. 0. 0. 0.]
Sign names of probabilities -- Wild animals crossing, General caution, Dangerous curve to the left, Road narrows on the right, Slippery road

Probabilities for specific Image -- [1. 0. 0. 0. 0.]
Sign names of probabilities -- No entry, Speed limit (20km/h), Stop, Yield, Dangerous c
urve to the right

Probabilities for specific Image -- [1. 0. 0. 0. 0.]
Sign names of probabilities -- Yield, Speed limit (80km/h), Keep right, Speed limit (30km/h), Speed li
mit (50km/h)