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

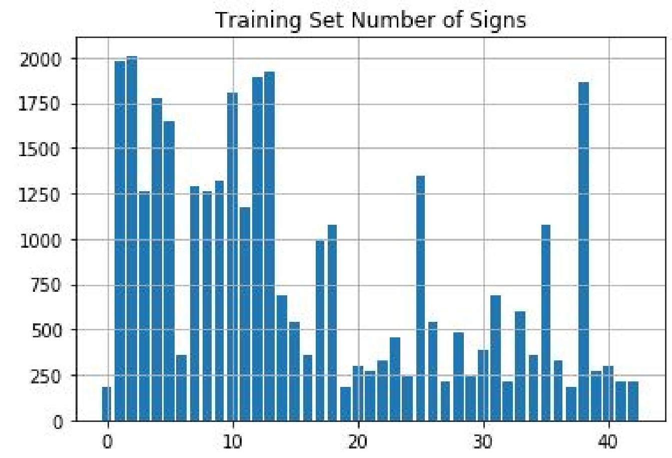
1. Provide a basic summary of the data set. In the code, the analysis should be done using python, numpy and/or pandas methods rather than hardcoding results manually.

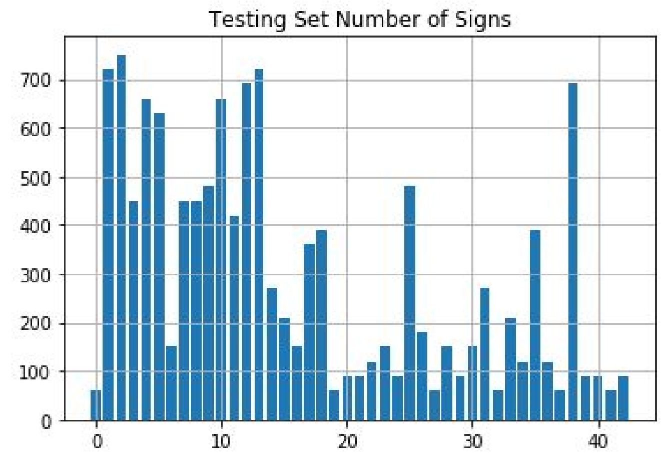
I used the pandas library to calculate summary statistics of the traffic signs data set:

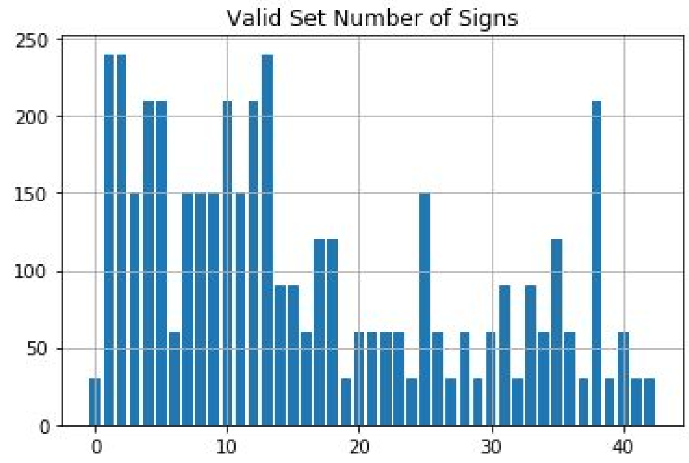
- The size of training set is 34799
- The size of the validation set is 12630
- The size of test set is 4410
- The shape of a traffic sign image is (32,32,3)
- The number of unique classes/labels in the data set is 43

2. Include an exploratory visualization of the dataset.

Here is an exploratory visualization of the data set. It is 3 bar chart showing the number of unique classes/label in each data set, training set, testing set and validation set.







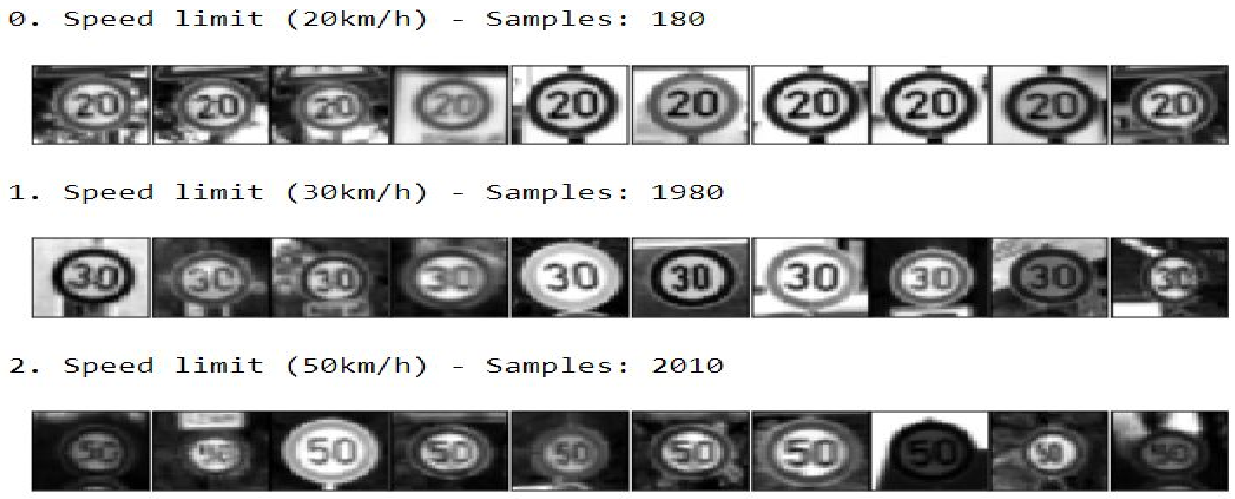
Also included above these bar graphs on the jupyter notebook are the pictures of each unique class and the corresponding label. Below is a short snippet.

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 preprocessed the image data by using both grayscale conversion and normalization. I chose the gray scale technique because color should not be a factor in identifying a traffic sign. It adds unnecessary parameters and makes the learning more difficult for the neural network. Also normalization makes the math much easier and faster for the neural network to calculate and learn.

Below i have given some examples of 3 unique classes once they are grayscaled and normalized. This gives each image a shape of (32,32,1)



```
0. Speed limit (20km/h) - Samples: 180
```

```
1. Speed limit (30km/h) - Samples: 1980
```

```
2. Speed limit (50km/h) - Samples: 2010
```

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.

Using the LeNet lab solution as a template I used convolutional neurel network for the traffic signs which consisted of layers of convolution, max pooling, flattening and fully connecting. I used the default hyper parameters of mu = 0 and sigma = 0.1

My final model consisted of the following layers:

Layer	Description	Input	Output
Convolution 5x5	Stride = 1, Padding = valid, Activation = RELU	32x32x1	28x28x48
Max Pooling	Stride = 2, Window = 2	28x28x48	14x14x48
Convolution 5x5	Stride = 1, Padding = valid, Activation = RELU	14x14x48	10x10x96
Max Pooling	Stride = 2, Window = 2	10x10x96	5x5x96
Convolution 3x3	Stride = 1, Padding = valid, Activation = RELU	5x5x96	3x3x172
Max Pooling	Stride = 1, Window = 2	3x3x172	2x2x172
Flatten	3D to 1D	2x2x172	688
Full Connected	Connect everything from above	688	84
Full Connected	Connect everything from above	84	43

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.

Model Training

I trained the model on my local machine with a CPU because of difficulties accessing credit to Amazon's Web Services to use their GPU. Training the model did not take as long as I expected since I was able to finish 25 EPOCHS in about 30 min. Im sure however if I SSH'd into Amazons GPU, it would be at least 3 times faster.

Training Parameters:

EPOCHS = 25

BATCH_SIZE = 128

SIGMA = 0.1

OPTIMIZER learning rate = .0001

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.

Final Results:

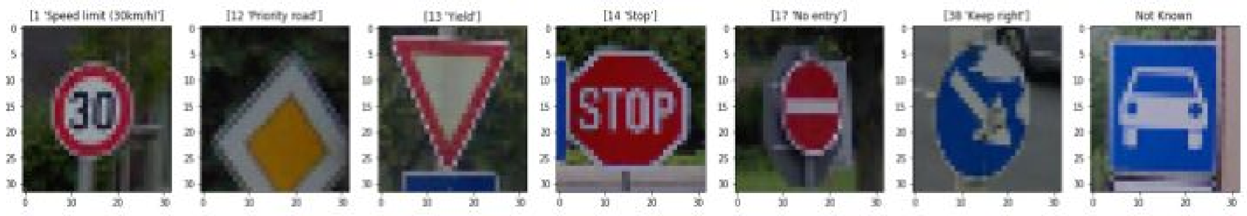
Validation Accuracy = 94.1%

Test Accuracy = 93.3 %

Discussion: Using the LeNet-5 architecture, I just changed around some of the parameters as discussed in the project lectures to fit colored images. The accuracies for each set were close but not at 93%. I decided to grayscale and normalize the images as was told in the Udacity lectures and I started to get better results. After modifying some of the conventional, flattening, fully connecting and max pooling layers in the LeNet architecture, I finally got over 93% accuracy. However it seemed to cap there and would not change even if I increased the epochs.

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.



I found a set of images online that seemed to correspond to my training data set and so I used these. Some of these signs are pretty straight forward and quite similar to some of the testing images for the neural network and so I would expect my model to have no trouble classifying those. Other signs like number 2 and number 6 are both augmented just a little bit. Number 2 is a little bit off the screen and so the entire image is not present when trying to classify. Number 6 may have some difficulty since the sign is worn out and color is fading away which may give the gray scaling some difficulty in identifying certain shapes. The last photo is a an image of unknown sign just to test the validity of the network.

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:

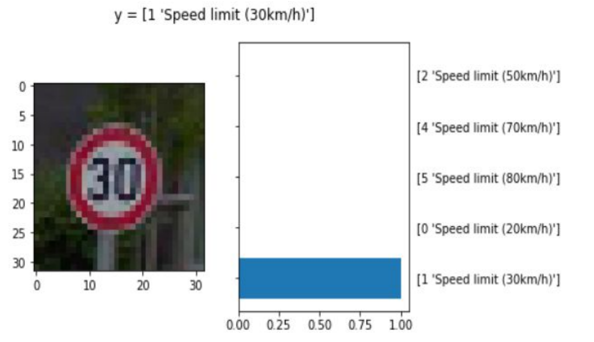
Image	Prediction
Speed limit (30km/h)	Correct
Priority Road	Correct
Yield	Correct

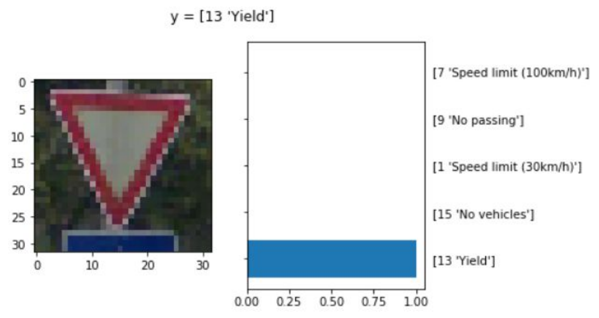
Stop	Correct
No Entry	Correct
Keep Right	Correct
Unknown Image	Incorrect labeled “Stop”

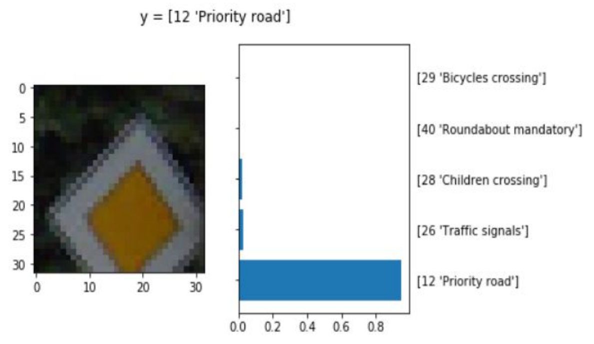
Looking at the images that were in the training set, the network was able to correctly classify all of the. However if you look at the set with the unknown image, this gives us a percentage of 6/7 = 86% accuracy with new images from the web.

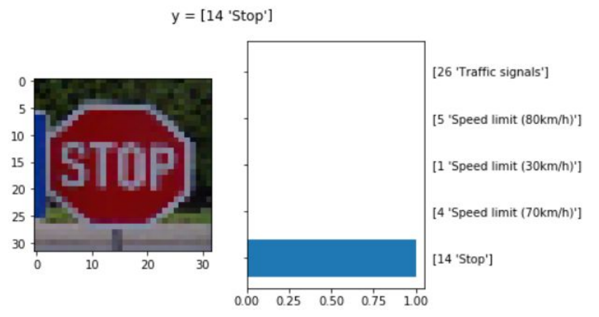
3. Describe how certain the model is when predicting on each of the five new images by looking at the softmax probabilities for each prediction. Provide the top 5 softmax probabilities for each image along with the sign type of each probability. (OPTIONAL: as described in the "Stand Out Suggestions"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

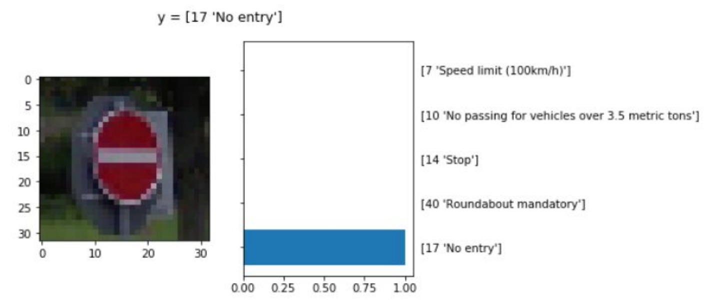
As you can see below, the model for the first 6 images is very confident in its predictions and that is a good thing because it classifies it correctly. However if you look at the last picture containing the unknown image, it has a high confidence in classifying it as a stop sign. This could lead to very dangerous problems as the network does not know how to deal with an unknown image in the real world.

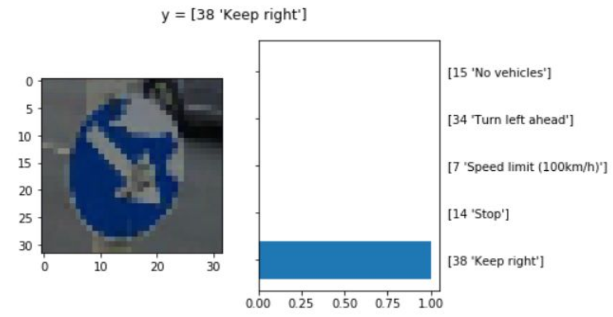


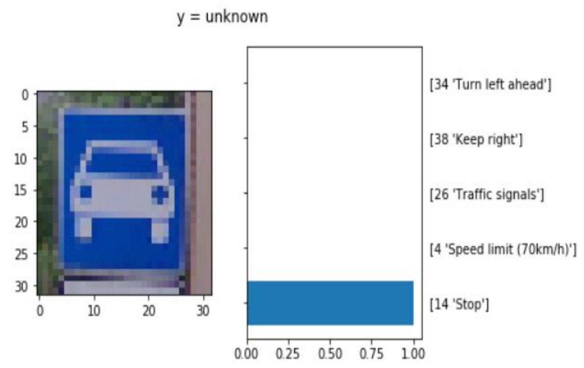







y = [17 'No entry']
[7 'Speed limit (100km/h)']
[10 'No passing for vehicles over 3.5 metric tons']
[14 'Stop']
[40 'Roundabout mandatory']
[17 'No entry']
0.00 0.25 0.50 0.75 1.00


y = [38 'Keep right']
[15 'No vehicles']
[34 'Turn left ahead']
[7 'Speed limit (100km/h)']
[14 'Stop']
[38 'Keep right']
0.00 0.25 0.50 0.75 1.00


y = unknown
[34 'Turn left ahead']
[38 'Keep right']
[26 'Traffic signals']
[4 'Speed limit (70km/h)']
[14 'Stop']
0.00 0.25 0.50 0.75 1.00