#**Traffic Sign Recognition**

##Writeup Template

###You can use this file as a template for your writeup if you want to submit it as a markdown file, but feel free to use some other method and submit a pdf if you prefer.

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

[//]: # (Image References)

[image1]: ./examples/visualization.jpg "Visualization"

[image2]: ./examples/grayscale.jpg "Grayscaling"

[image3]: ./examples/random_noise.jpg "Random Noise"

[image4]: ./examples/placeholder.png "Traffic Sign 1"

[image5]: ./examples/placeholder.png "Traffic Sign 2"

[image6]: ./examples/placeholder.png "Traffic Sign 3"

[image7]: ./examples/placeholder.png "Traffic Sign 4"

[image8]: ./examples/placeholder.png "Traffic Sign 5"

Rubric Points

###Here I will consider the [rubric points](https://review.udacity.com/#!/rubrics/481/view) 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](https://github.com/udacity/CarND-Traffic-Sign-Classifier-Project/blob/master/Traffic_Sign_Classifier.ipynb)

###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

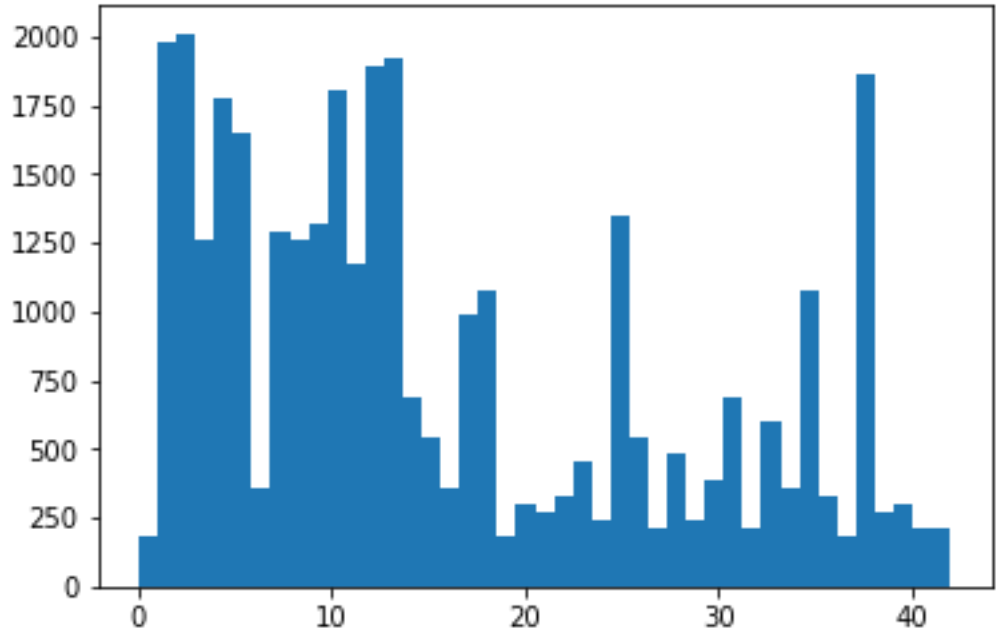
signs data set:

* The size of training set is 34799
* The size of the validation set is 4410
* The size of test set is 12630
* 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 a bar chart showing how the data ...

![Graph of data][./writeup_images/graph.png]



###Design and Test a Model Architecture

####1. I decided to normalize the data with the help and advice of some of the forum mentors. I accessed the variable holding the image data and divided it by 255, subtracting the result from .5. I began to flip the data as well, but while testing my validation rate I saw that the flipped images were not needed.

![alt text][./writeup_images/normalized_data.png]

The difference between the original data set and the augmented data set is the following: normalization

####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	32x32x3 RGB image
Convolution 5x5	1x1 stride, same padding, outputs 10x10x16
Activation(Relu)	
Pooling	ksize=1x2 strides = 1x2
Convolution 5x5	1x1 stride, same padding, outputs 10x10x16

| Activation(RELU) |

| Pooling |ksize=1x2 strides = 1x2
|

| flatten |

| Fully connected | shape = 120,84
|

| Dropout |Input: activation outputs with keep_prob
=tf.placeholder(tf.float32)|

| Fully connected | shape = 120,84
|

| Activation(Relu) | shape = 120,84
|

| Dropout |Input: activation outputs with keep_prob
=tf.placeholder(tf.float32)|

| Max pooling | 2x2 stride, outputs 14x14x6,5x5x16 |

| Convolution 3x3 | etc.
|

| Fully connected | Input = 84. Output = 10.
|

| Softmax | etc.
|

| |
|

| |
|

Printed representation of LeNet model used:

Tensor("add_16:0", shape=(?, 28, 28, 6), dtype=float32)

Tensor("add_17:0", shape=(?, 10, 10, 16), dtype=float32)

Tensor("Flatten_2/Reshape:0", shape=(?, 400), dtype=float32)

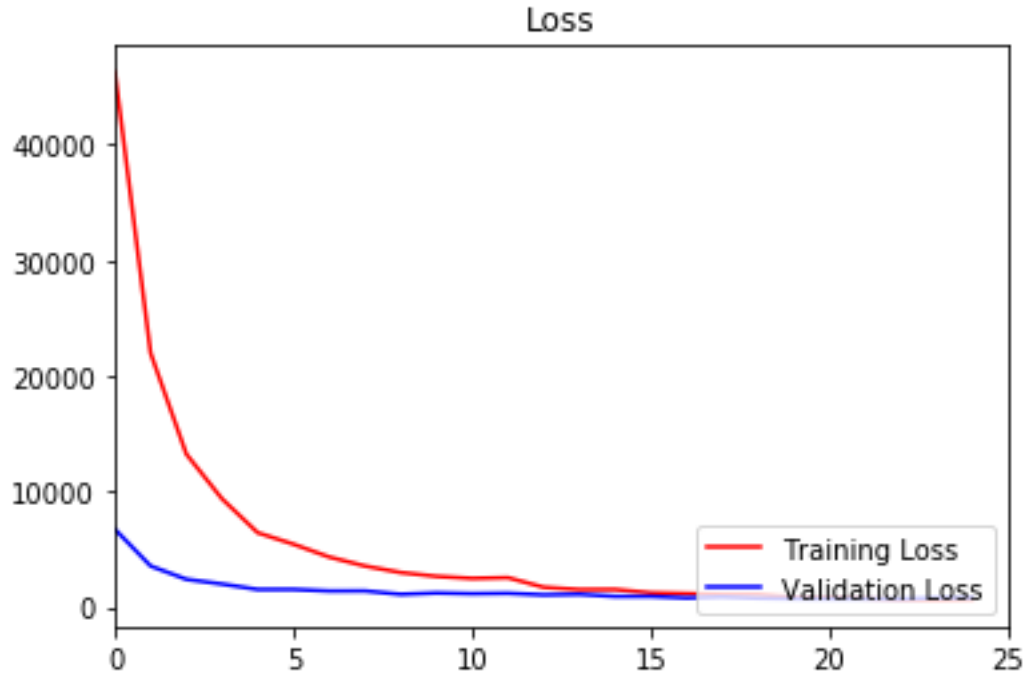
Tensor("add_18:0", shape=(?, 120), dtype=float32)

Tensor("add_19:0", shape=(?, 84), dtype=float32)

####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 LeNet function which consists of the layers described above. I also used 25 EPOCHS and a batch size of 100 per sweep. I also plotted a graph of the validation/training loss to better visualize what was going on with the training.

![alt text][./writeup_images/validation_loss.png]



####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.

When I first started training the model, I could only achieve ~.89. I then enlisted the help of the forums and asked a lot of questions. I found that adding normalized data and dropout would give me the results I was looking for. At the time of training, I can now gain a validation accuracy of ~.95 with a training accuracay of ~.99.

My final model results were:

* training set accuracy of 0.996

* validation set accuracy of 0.951

* test set accuracy of 0.94

If a well known architecture was chosen:

* What architecture was chosen? LeNet

* Why did you believe it would be relevant to the traffic sign application?

* How does the final model's accuracy on the training, validation and test set provide evidence that the model is working well?

###Test a Model on New Images

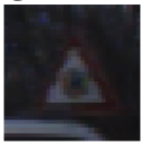
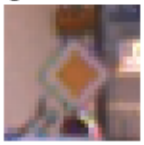
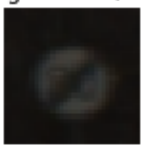
####1. Choose five German traffic signs found on the web and provide them in the report. For each image, discuss what quality or qualities might be difficult to classify.

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

The first image might be difficult to classify because ...

####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:

input	top guess: 35 (96%)	2nd guess: 34 (2%)	3rd guess: 10 (0%)
input	top guess: 11 (100%)	2nd guess: 30 (0%)	3rd guess: 12 (0%)
			
input	top guess: 6 (90%)	2nd guess: 1 (5%)	3rd guess: 2 (4%)
			
input	top guess: 40 (100%)	2nd guess: 34 (0%)	3rd guess: 41 (0%)
input	top guess: 38 (100%)	2nd guess: 13 (0%)	3rd guess: 23 (0%)
input	top guess: 13 (88%)	2nd guess: 14 (10%)	3rd guess: 17 (1%)
input	top guess: 18 (100%)	2nd guess: 26 (0%)	3rd guess: 27 (0%)
input	top guess: 39 (68%)	2nd guess: 33 (32%)	3rd guess: 37 (0%)

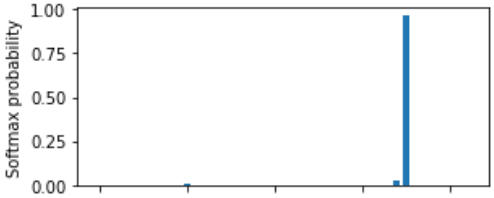
The model was able to correctly guess 3 of the 8 traffic signs, which gives an accuracy of 37.5%.

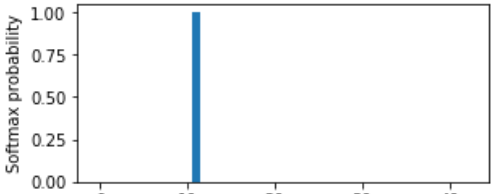
####3. Describe how certain the model is when predicting on each of the five new images by looking at the softmax probabilities for each prediction. Provide the top 5 softmax probabilities for each image along with the sign type of each probability. (OPTIONAL: as described in the "Stand Out Suggestions"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

For the first image, the model is relatively sure that this is a speed limit sign (probability of 0.6), and the image does contain a stop sign. The model still can train a lot better but the starting point here shows that we have some outside data that can be classified.

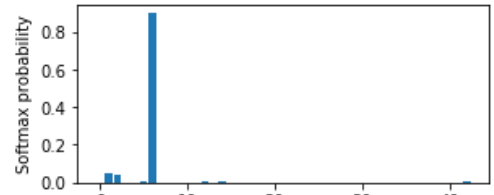
Probability	Prediction
.60	Speed Limit (60mph)
.70	Yield
.90	Speed Limit(30mph)
.90	One side only
.70	Left turn only
.70	Danger
.80	Roadwork

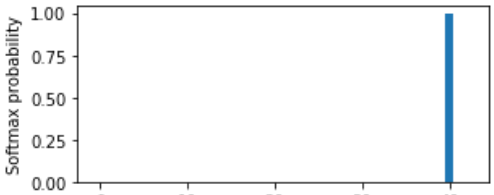
60

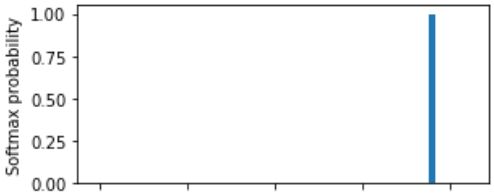

Softmax probability
1.00
0.75
0.50
0.25
0.00
0
10
20
30
40

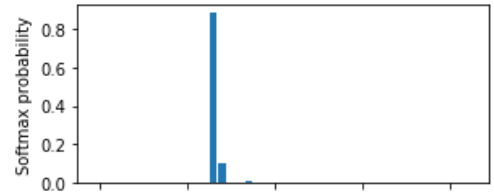

Softmax probability
1.00
0.75
0.50
0.25
0.00
0
10
20
30
40

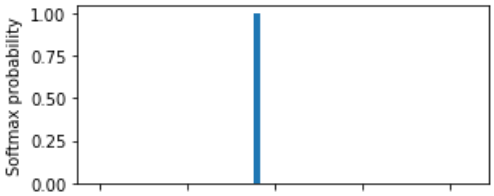
30

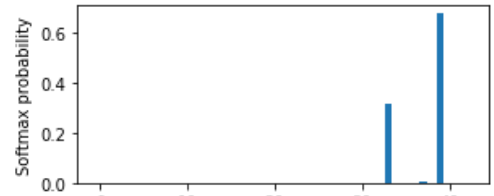

Softmax probability
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40


Softmax probability
1.00
0.75
0.50
0.25
0.00
0
10
20
30
40


Softmax probability
1.00
0.75
0.50
0.25
0.00
0
10
20
30
40


Softmax probability
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40


Softmax probability
1.00
0.75
0.50
0.25
0.00
0
10
20
30
40


Softmax probability
0.6
0.4
0.2
0.0
0
10
20
30
40

For the second image ...

(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?