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

Data Set Summary & Exploration

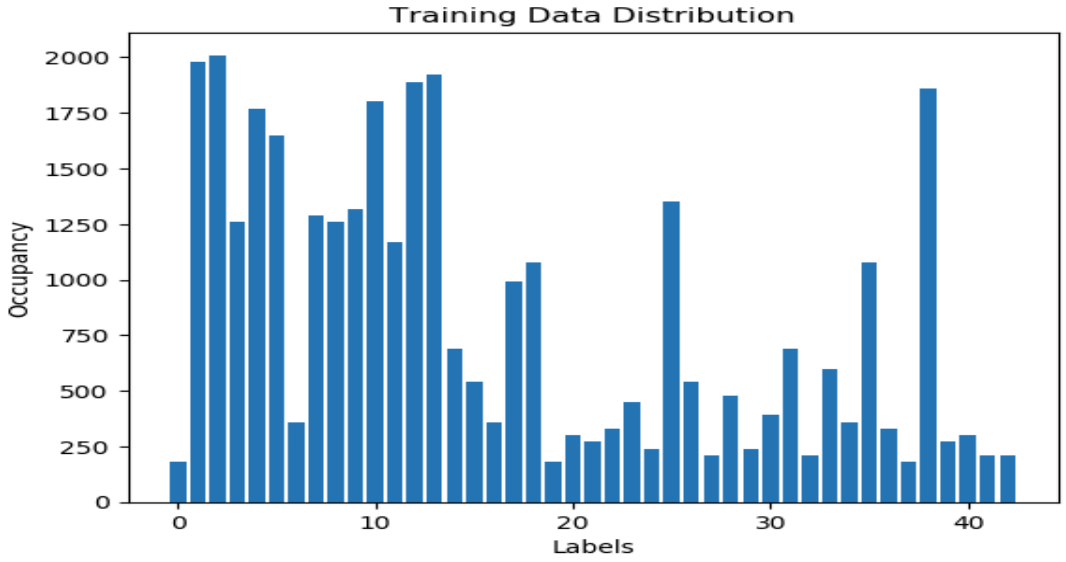
1. **Provide a basic summary of the data set. In the code, the analysis should be done using python, numpy and/or pandas methods rather than hardcoding results manually.**

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

- The size of training set is 34,799.
- The size of the validation set is 4,410.
- The size of test set is 12,630.
- 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.**

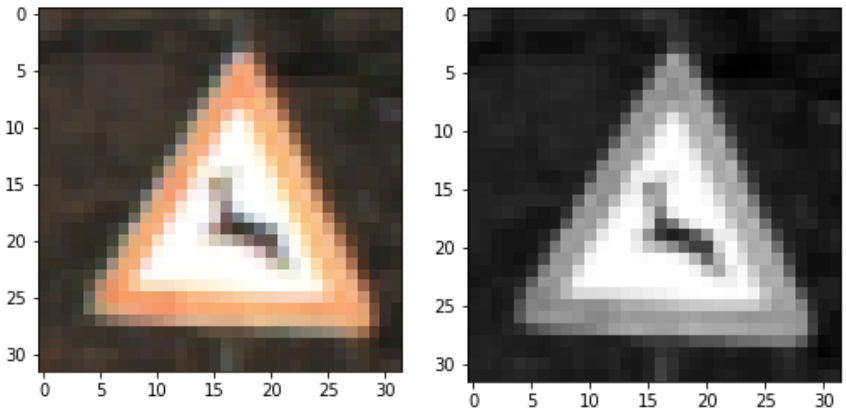
Here is an exploratory visualization of the data set. It is a bar chart showing how the training data is distributed into different labels.



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.)**

   As first step, I convert the image to gray scale, the reason is to eliminate the color impact on the result. Here is an example of before and after gray scaling.



   In addition, I shuffled the sequence of the training set, to avoid the neural network memorize the result based on sequence while exploring training parameters.

   Lastly, I normalized the data with (pixel – 128) / 128, because the neural network will train better if the data set has a mean of zero and a small variance.

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.**

   I choose to use LeNet convolution neural network to train my classifier.

Layer	Input	Output
Input	32x32x1 RGB Image	
Convolution 5x5, valid, stride 1	32x32x1	28x28x16
RELU		
Max Pooling 2x2, valid, stride 2	28x28x16	14x14x16
Convolution 5x5, valid, stride 1	14x14x16	10x10x24

RELU		
Max Pooling 2x2, valid, stride 2	10x10x24	5x5x24
Flatten		
Fully Connected	600	400
RELU, Dropout		
Fully Connected	400	150
RELU, Dropout		
RELU, Dropout	150	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.**

   To train the model, I used a learning rate of 0.002. Adam optimizer performed on reduce mean of cross entropy. The batch size is 900 and the epoch is 10. I set the keep probability of 0.5 for the training. The validation accuracy is still increasing for a learning rate of 0.001 at epoch 10, increase the learning rate to 0.002 accelerate the training and reached a higher accuracy. I tried batch size between 800 and 1200, and 900 has been producing the best accuracy with a 0.5 drop out probability. The reason I add the 0.5 drop out is to generalize the training data.

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 0.993
   - validation set accuracy of 0.952.
   - test set accuracy of 0.938.

   If a well-known architecture was chosen:

   - What architecture was chosen?

     LeNet.

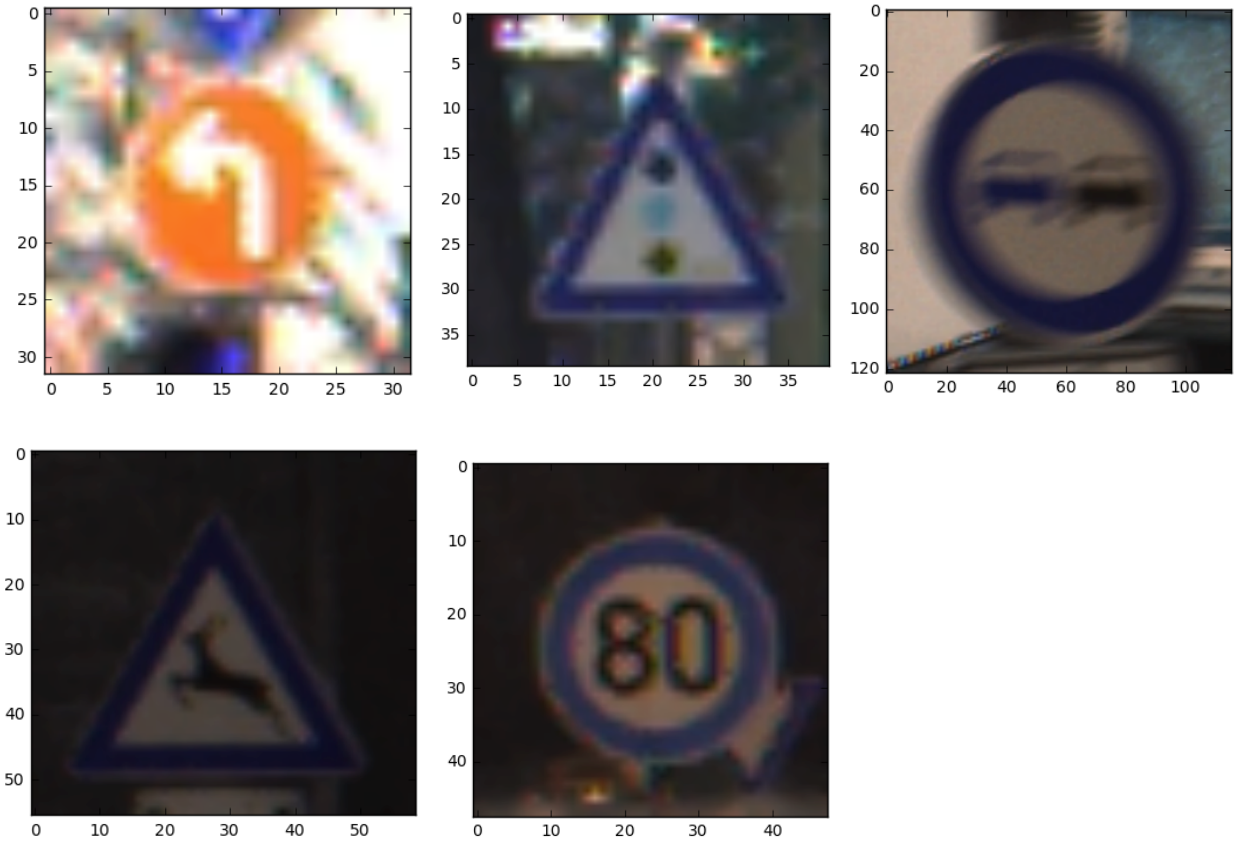
   - Why did you believe it would be relevant to the traffic sign application?

     LeNet is mainly built with convolutional neural network layers and fully connected layers. The CNN layers are commonly applied to analyzing visual imagery, it is good at picking up the important data and leaving out the less important information. Connected after CNN, the fully connected layers analyze that important information to predict a good result.

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:



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).**

   As in the code line 60 and 63, all the 5 images are guessed correctly:
   image 1 is Turn left ahead
   image 2 is Traffic signals
   image 3 is No passing
   image 4 is Wild animals crossing
   image 5 is Speed limit (80km/h)

   The 100% accuracy is better than the test set accuracy which is 93.8%.

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 in the code line 105, the SoftMax probabilities for each images are:

Image 1 's top 5 certainties are:
Turn left ahead :0.993
Ahead only :0.007
Yield :0.000
Road work :0.000
Keep right :0.000
The ahead only and turn left sign both contains the upper arrow.

Image 2 's top 5 certainties are:
Traffic signals :0.996
General caution :0.003
Bumpy road :0.000
Road narrows on the right :0.000
Dangerous curve to the right :0.000
The general caution sign contains round edge at the top and bottom which is similar as traffic signals.

Image 3 's top 5 certainties are:
No passing :1.000
No passing for vehicles over 3.5 metric tons :0.000
Slippery road :0.000
Dangerous curve to the right :0.000
Vehicles over 3.5 metric tons prohibited :0.000

Image 4 's top 5 certainties are:
Wild animals crossing :0.999
Double curve :0.000
Dangerous curve to the left :0.000
Slippery road :0.000
Bicycles crossing :0.000

Image 5 's top 5 certainties are:
Speed limit (80km/h) :0.998
Speed limit (60km/h) :0.002
Speed limit (100km/h) :0.000
Speed limit (50km/h) :0.000
No passing for vehicles over 3.5 metric tons :0.000
Well, sometimes it’s even hard for human to distinguish 8 and 6.