Traffic_Sign_Recognition

Traffic Sign Recognition

Writeup

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

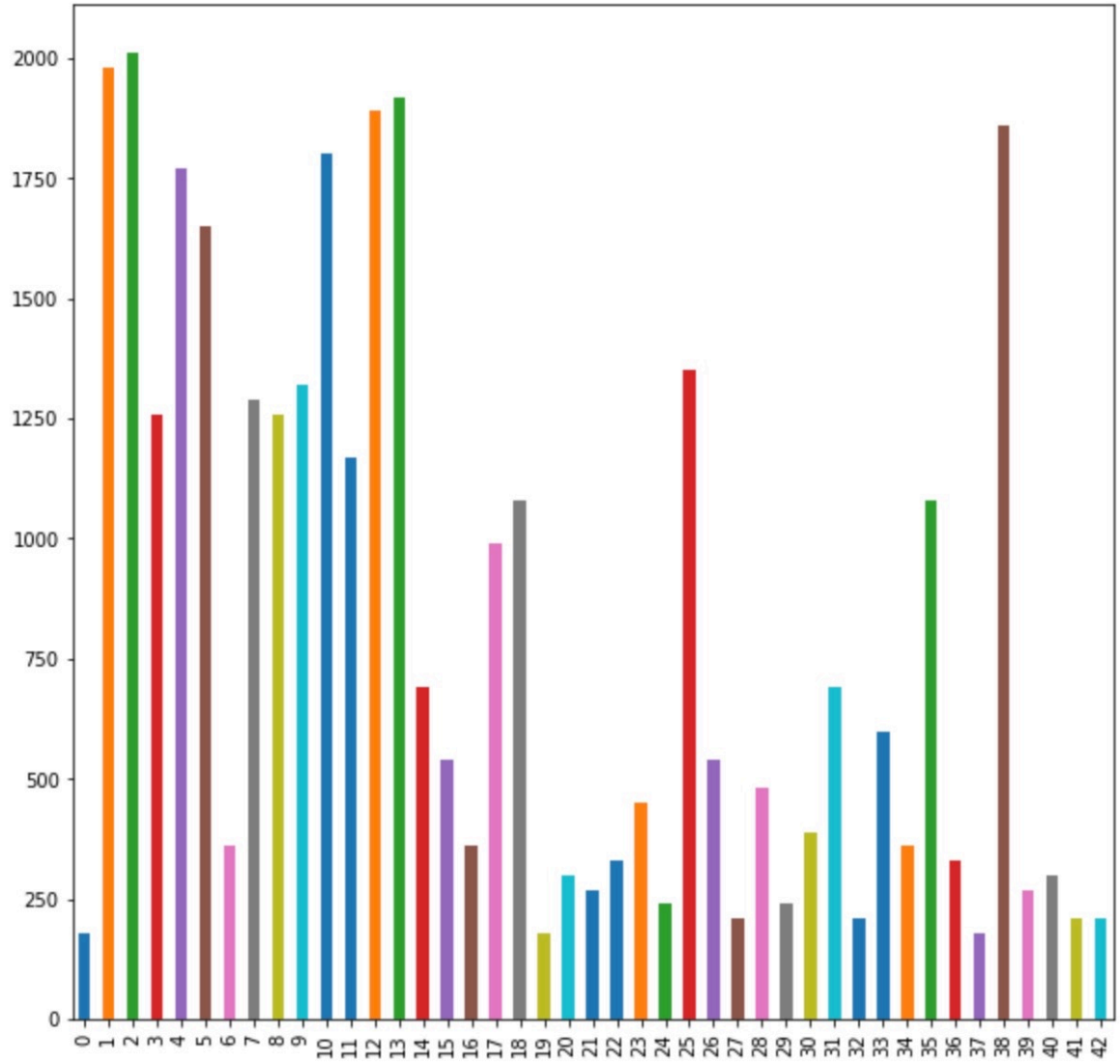
1. Provide a basic summary of the data set. In the code, the analysis should be done using python, numpy and/or pandas methods rather than hardcoding results manually.

I used the numpy 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 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 how the data.



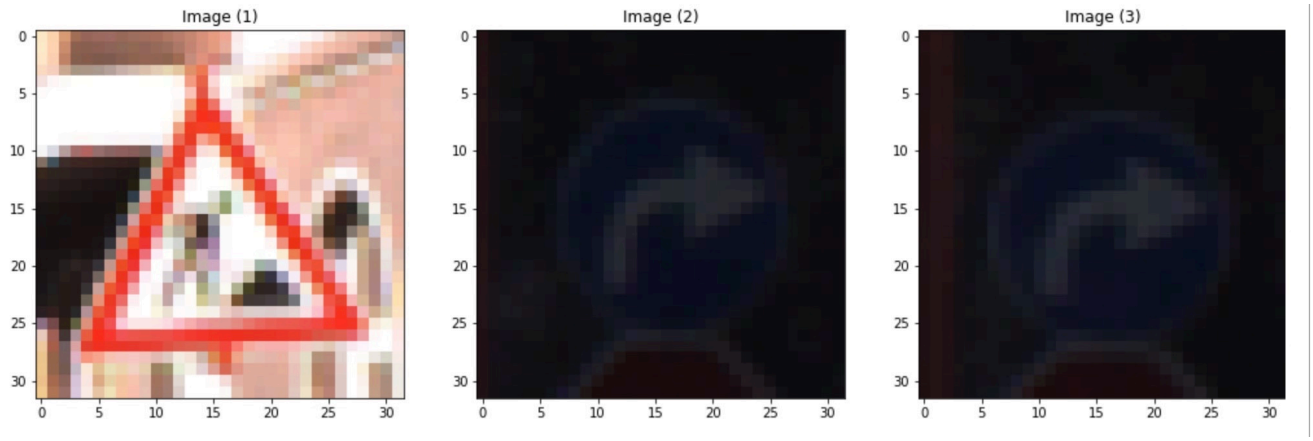
The image shows that for different labels the amount of data is not the same.

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 a first step, I decided to convert the images to normalize the images because it the BP step will be more stable if we normalize the data into a scale between -1 and 1.
The equation I use is $X = \frac{X-128}{128}$, $X \in [0, 255]$. The X is the pixel value, for every X we use that equation to normalize.

Here is an example of three traffic sign images in the training set.



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	5x5 stride, valid padding, outputs 28x28x6
RELU	
Max pooling	2x2 stride, outputs 14x14x6
Convolution 5x5	5x5 stride, valid padding, outputs 10x10x16
RELU	
Max pooling	2x2 stride, outputs 5x5x16
Flatten	400 vector
Fully connected	Outputs 120
Dropout	Dropout rate 0.5
RELU	
Fully connected	Outputs 84
Dropout	Dropout rate 0.5
RELU	
Fully connected	Outputs 43
Softmax	Outputs 43
Cross Entropy	Loss function
Adam	Optimizer, learning rate 0.001

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 Adam Algorithm as the optimizer. And then select the batch size as 1024, and epochs as 200, the initial learning rate is 0.001, when the algorithm is training, the dropout rate is 0.5, otherwise the rate is 1.0.

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 99%
- validation set accuracy of 96%
- test set accuracy of 95%

If a well known architecture was chosen:

- What architecture was chosen?
  The architecture is the basic LeNet for mnist classification, but change the input and output layer slightly to fit the traffic sign data.

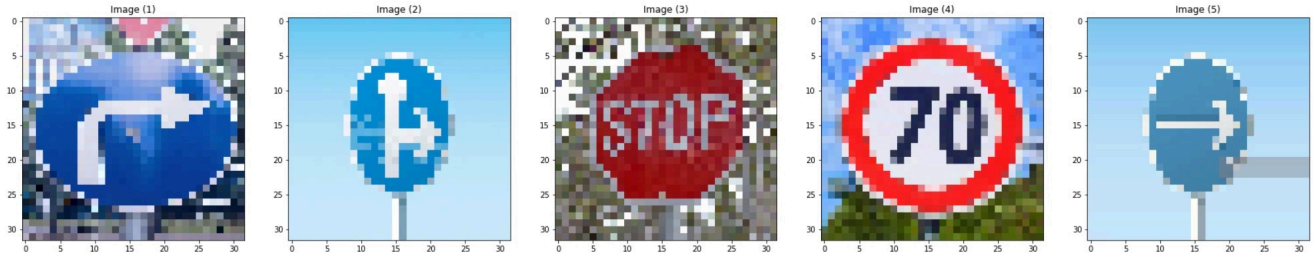
- Why did you believe it would be relevant to the traffic sign application?
  The problem we use are trying to solve is the same kind of classification problem. And I know that LeNet is a successful structure to solve ImageNet dataset, so that I think with only 43 classes, it will also do a great job.

- How does the final model's accuracy on the training, validation and test set provide evidence that the model is working well?
  The training accuracy is quite high, it arrived 99% in 200 eopches. And the validation accuracy is also going high after 200 epoches, it reached 96%.
  After fine tuinning the model to get 96% in validation set. I use the test set to test the algorithm, it got an accuracy at 95%, which is as good as the validation 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:



The images are downloaded from network, and scale to 32x32 pixels, it can be recognized quite well by human eyes even in such small resolution.

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
Turn right ahead	Turn right ahead
Go straight or right	Go straight or right
Stop	Stop
Speed Limit(70 Km/h)	Speed Limit(70km/h)
Keep Right	Turn left ahead

The model was able to correctly guess 4 of the 5 traffic signs, which gives an accuracy of 80%. This compares favorably to the accuracy on the test set of 90%.

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 79th cell of the Jupyter notebook.

1. For the first image, the model is relatively sure that this is a No passing (probability of 0.99), and the image doesn't contain a No passing. The top five soft max probabilities were

Probability	Prediction
.99	Turn right ahead
.00	Ahead only

.00	Go straingt or left
.00	Turn left ahead
.00	keep left

The image is a clear sign of turn right, so that it is not so difficult to classify the image. Even when we preprocessing the image into a 32x32 pixels, the sign is clear and big enough to be seen.

2. For the second image, the model is relatively sure that this is a Go straight or right (probability of 0.68), and the image does contain a Go straight or right. The top five soft max probabilities were

Probability	**Prediction**
.68	Go straight or right
.28	Turn left ahead
.00	Keep right
.00	Roundabout mandatory
.00	Go straight or left

For the second image, the algorithm only get the confidence of 68%, I think the main reason is that, the sign only occpies a small region of the image. Thus the sign is a little small compared to the images in the training data. So the algorithm cannot see it clearly. But it still make a correct prediction.

3. For the third image, the model is relatively sure that this is a Stop sign (probability of 0.74), and the image does contain such sign. The top five soft max probabilities were

Probability	**Prediction**
.74	Stop
.26	No entry
.00	Bumpy road
.00	No vehicles
.00	Bicycles crossing

The stop sign is correctly predicted but in a 74% confidence, as the No entry is the most confusing option for the algorithm. For the two signs are similar in their round shapes and red color. Actually, because of the blurness of the image, it contains a lot of noise in its background, that may be the reason for the confidency, it should be better to scale the image using a more 'clear' algorithm.

4. For the fourth image, the model is relatively sure that this is a Speed limit (70km/h) sign (probability of 1), and the image does contain such sign. The top five soft max probabilities were

Probability	**Prediction**
1.0	Speed limit (70km/h)

.00	Speed limit (30km/h)
.00	Speed limit (20km/h)
.00	No entry
.00	keep right

The algorithm has enough confidence to predict such speed limit sign, because this kind of signs has a lot data in the training set. It has been trained a lot so that it can recognize the sign correctly.

5. For the fifth image, the model is relatively sure that this is a Turn right ahead sign (probability of 0.99), and the image does contain such sign. The top five soft max probabilities were

Probability	**Prediction**
.99	Turn left ahead
.00	keep right
.00	Turn right ahead
.00	Ahead only
.00	Yield

This is the wrong prediction of the five images, the sign should be turn right ahead but it predicted as turn left ahead. What is more wired that it has quite a high confidence for its prediction. But as you can see in the top five predictions, the right prediction is in the third place. As we look at the dataset description, it only contains a small amount of 'Turn left ahead' signs. And the picture is not blurred, it is clear enough to be recognized. We may use data argument in this skewed dataset to fix bad prediction.