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

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

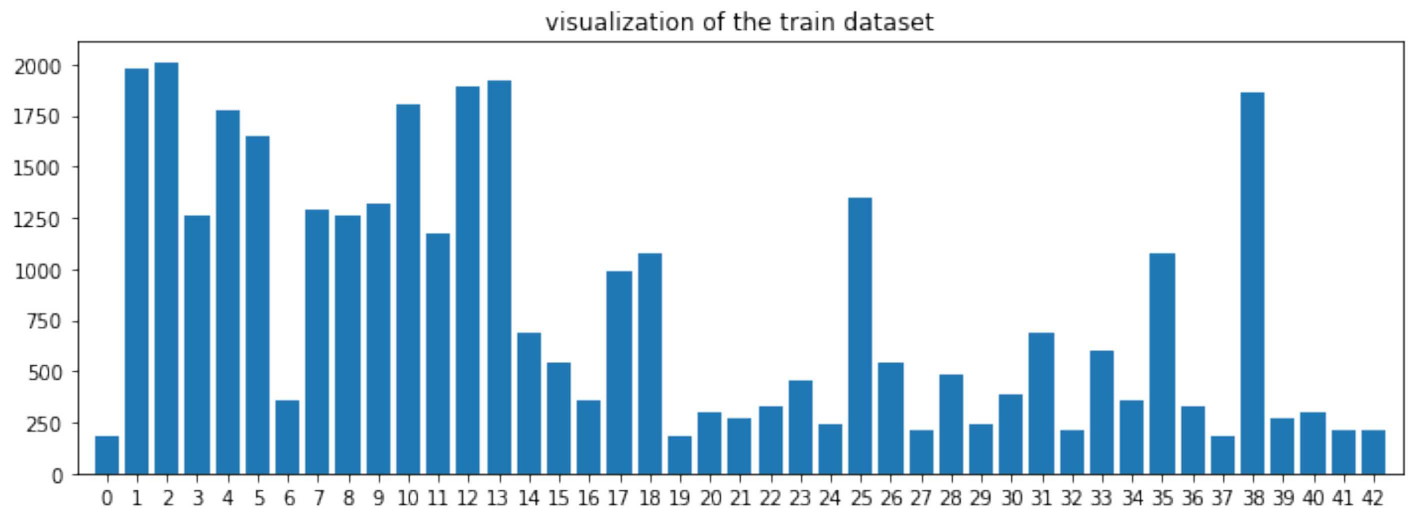
using python, numpy and/or pandas methods rather than hardcoding results manually.

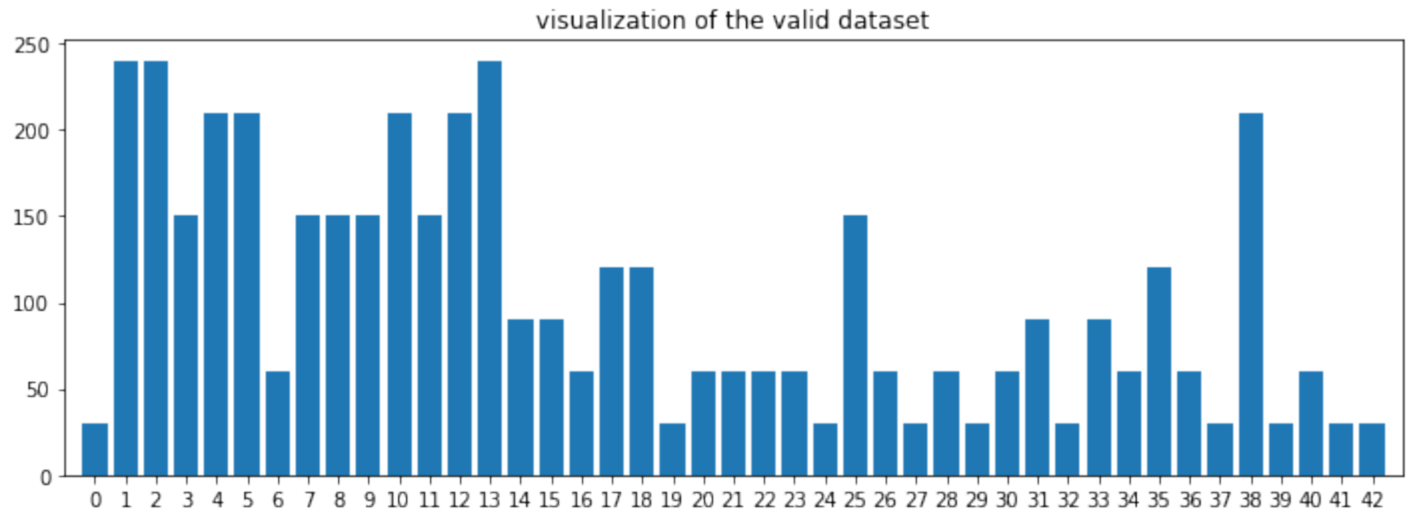
I used the pandas library to calculate summary statistics of the traffic signs data set:

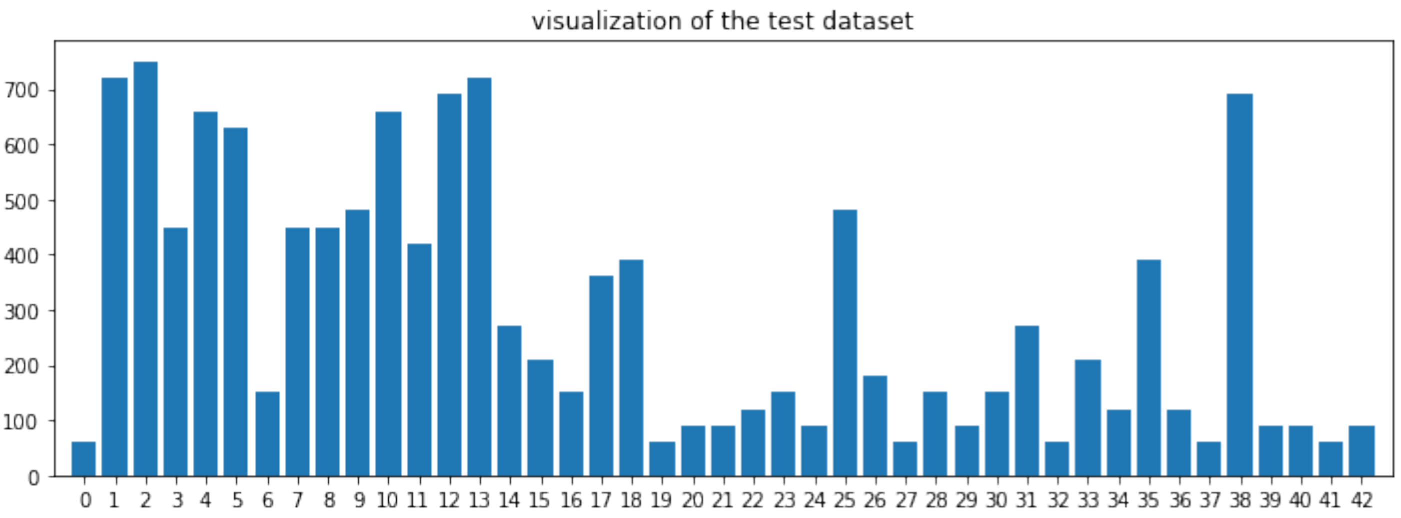
- The size of training set is 34799
- The size of the validation set is 4410
- The size of test set is 12630
- The shape of a traffic sign image is (32, 32, 3)
- The number of unique classes/labels in the data set is 43

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 are distributed in each dataset







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 normalized the image data to: -Standardize the pixel values with formula:(pixel - 128.)/ 128. I did not gray scale the images or other methods.

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, valid padding, outputs 28x28x6
RELU	
Max pooling	2x2 stride, outputs 14x14x6
Convolution 5x5	1x1 stride, valid padding, outputs 10x10x16
RELU	
Max pooling	2x2 stride, outputs 5x5x16
Flatten	Input = 5x5x16. Output = 400
Fully connected	Input = 400. Output = 120
RELU	
Dropout	Training:0.5
Fully connected	Input = 120. Output = 84
RELU	
Dropout	Training:0.5
Fully connected	Input = 84. Output = 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 an Gradient Descent optimizer.
The batch size is selected as 128.
The number of epochs is selected as 10.
The learning rate is selected as 0.002

The above combination comes from sevral runs of testing.

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 0.987
- validation set accuracy of 0.948
- test set accuracy of 0.929

Following approaches are reported in [report.pdf]()

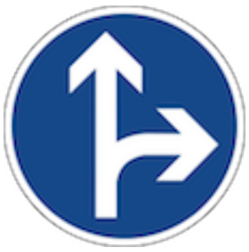
If an iterative approach was chosen: * What was the first architecture that was tried and why was it chosen? * What were some problems with the initial architecture? * How was the architecture adjusted and why was it adjusted? Typical adjustments could include choosing a different model architecture, adding or taking away layers (pooling, dropout, convolution, etc), using an activation function or changing the activation function. 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? * 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?

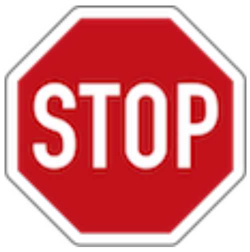
If a well known architecture was chosen: * What architecture was chosen? LeNet-5 with 2 dropout layers * Why did you believe it would be relevant to the traffic sign application? According to the introduciton, it is designed for handwritten and machine-printed character recognition. Dropout layers can help to solve the overfitting. * How does the final model's accuracy on the training, validation and test set provide evidence that the model is working well? The model works well. It can still be improved.

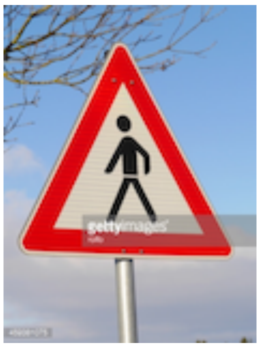
Test a Model on New Images

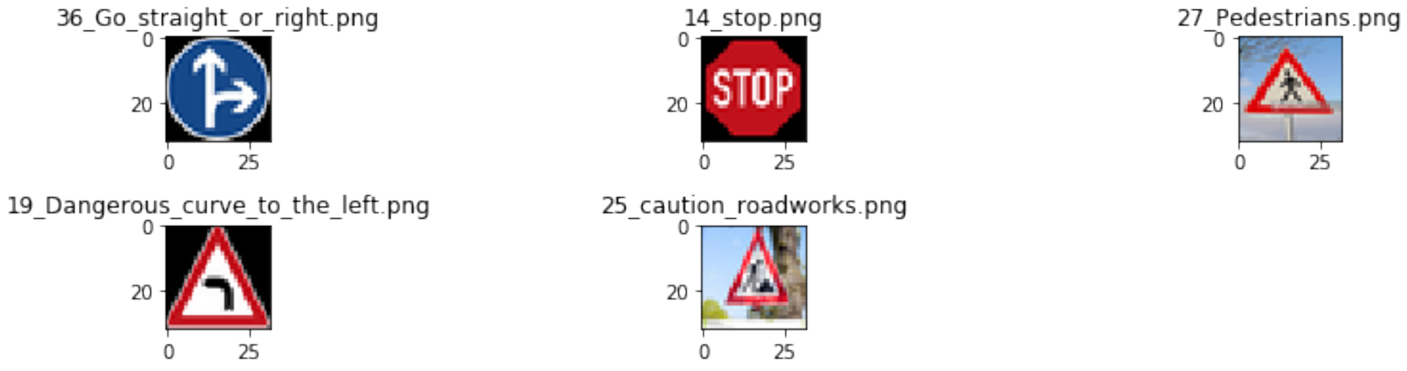
1. Choose five German traffic signs found on the web and provide them in the report. For each image, discuss what quality or qualities might be difficult to classify.

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








36_Go_straight_or_right.png
14_stop.png
27_Pedestrians.png
19_Dangerous_curve_to_the_left.png
25_caution_roadworks.png

The first, second and fourth image should be easy to identify because they are standard signs download from wikipedia . All of them have high quality and no other noises to disturb the classification. Unfortunately, not all of them are identited correctly, one reason could be our model extract too many disturbing information due to the quality of images in train data set.
The thrid image might be difficult to classify because it includes some additon information like characters,tree and cloud. The tree in last image will disturb the clssify, so the last one could be very difficult to classify.

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
Go straight or right	Go straight or right
stop	stop
Pedestrians	Speed limit (30km/h)
Danger curve to left	Slippery road
Road work	Road work

The model was able to correctly guess 3 of the 5 traffic signs, which gives an accuracy of 60%.

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 11th cell of the Ipython notebook.

1)The first image is : sign "Go straight or right".
The model is relatively sure that this is a "Go*straight*or*right sign" , and the image does contain a
Go*straight*or*right sign.
The top five soft max probabilities were:

Probability	Prediction
1.	Go*straight*or_right
0	keep right
0	Ahead only
0	Roundabout mandatory
0	Go straight or left

The model was very certain about the prediction which gave the probability as 1.   This could be caused by the high image quality and model quality for this image.

2)The second image is a "stop" sign.
The model relatively sure that this is a stop sign (probability of 0.67), and the image does contain a stop sign. The top five soft max probabilities were:

Probability	Prediction
0.67	stop
0.14	Speed limit (120km/h)
0.09	Speed limit (30km/h)
0.07	Speed limit (20km/h)
0.03	Yield

It is noticed that, the model was not quite sure about the prediction. The probability is about 0.67. The next 3 smaller probabilities are speed limit signs which give 0.3 prbabilities in total. Speed limit and stop signs have some common charaters. This could be a hence that the model still need to be improved to distinguish stop sign and speed limit sign.

3) The thrid image is "Pedestrians" sign. The model relatively sure that this is a Speed limit (30km/h) sign (probability of 0.69), and the image does not contain a Speed limit (30km/h) sign.
The top five soft max probabilities were:

Probability	**Prediction**
0.69	Speed limit (30km/h)
0.3	Speed limit (20km/h)
0.01	Stop
0	Speed limit (120km/h)
0	Bicycles crossing

Again, speed limit signs are recognized. This tell us the model learned too many informaiton about speed limit. If we go back to look the train dataset distribution, we can found the there are more images about speed limit and stop signs. That could be the reason why the model like speed limit signs. To improve this, we should take care of the dataset collection.

4) The 4th image is "Danger curve to left".

The model is sure that this is a Slippery road sign (probability of 0.98), and the image does not contain a Slippery road sign. The top five soft max probabilities were:

Probability	**Prediction**
0.98	Slippery road
0.01	Dangerous curve to the left
0	No passing
0	Ahead only
0	Double curve

The probability of correct prediction "Dangerous curve to the left" is only 0.01. The model did not get the import information that there must include a vehicle in "Slippery road" sign. The reason could be the dataset distribution or the vehicle is distinguished as background information.

5) The 5th image is relatively sure that this is a Road work sign (probability of 0.89), and the image does contain a Road work sign. The top five soft max probabilities were:

Probability	Prediction
0.89	Road work
0.05	Stop
0.03	Bicycles crossing
0.02	Wild animals crossing
0.01	Beware of ice/snow

There are a lot of road work images in train dataset so the model can work relative well. Once again, this tell us the important of dataset selection.

The distribution of train, test and valid dataset are similar. The model can be trained to get a relative high accuracy. But if we use the model to predict new images, the image could be miss predicted because it is a image the model does not learned good enough.

(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?