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 the project code is present in workspace.

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

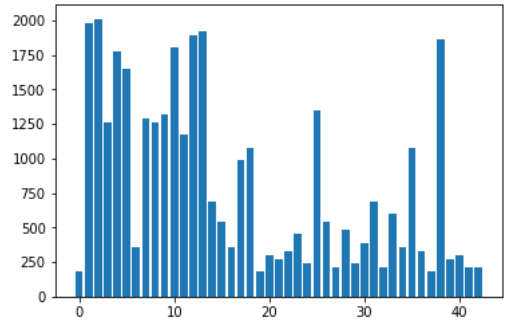
hardcoding results manually.

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 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.

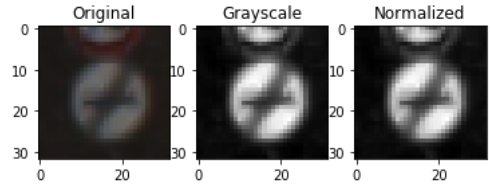
Here is an exploratory visualization of the data set. It is a bar chart showing the frequency of each class of image.



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 used the formula from Wikipedia to convert RGB image to Grayscale.
- I divided the grayscale image by 255 to normalize it.



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
Fully connected	Input = 400, Output = 120
RELU	
Fully connected	Input = 120, Output = 84
RELU	
Softmax	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.

- I tried changing multiple hyperparameters such as learning rate, batch size, layers etc.
- I got more than 93% validation accuracy with
  - learning rate = 0.005
  - epochs = 200
  - batch size = 256

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.

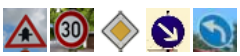
My final model results were:

- training validation set accuracy of 95% (but it keeps varying between 93% to 95%).
- test set accuracy of 92.5%.
- What architecture was chosen?
  - I chose LeNet architecture for training the traffic signs.
- Why did you believe it would be relevant to the traffic sign application?
  - The architecture was used to classify MNIST dataset and it worked perfectly.
  - MNIST data set consisted of 32x32 images having only one digits. Similarly German traffic signs consist of images having only one sign.
  - I believe it would work for other classification tasks such as flower classification provided the image contains only one flower.
- How does the final model's accuracy on the training, validation and test set provide evidence that the model is working well?
  - The validation accuracy often reaches 95%
  - The test accuracy is 92.5%
  - I beleive it can be increased.

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:



I don't think there seems to be any difficulty in order to classify the images as it seems quite clear to me. The intensity seems to be more but since the images are grayscaled and normalize I think this is also not an issue.

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:

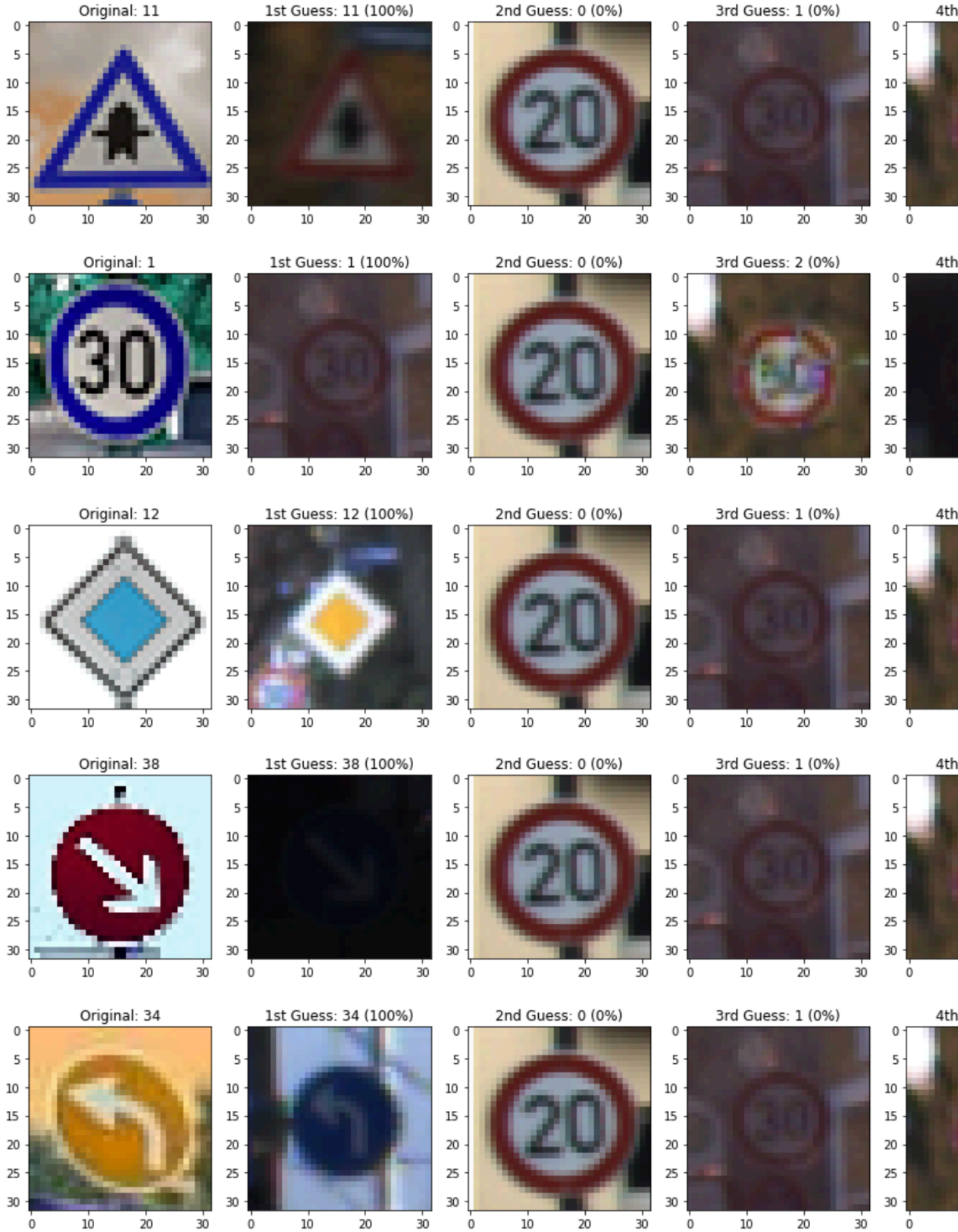
Image	Prediction
Right-of-way at the next intersection	Right-of-way at the next intersection
Speed limit (30km/h)	Speed limit (30km/h)
Priority road	Priority road
Keep right	Keep right
Turn left ahead	Turn left ahead

The model was able to correctly guess 5 of the 5 traffic signs, which gives an accuracy of 100%.

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 24th cell of the Ipython notebook.

The 5 probabilties for each images are as displayed below in the image:


Original: 11
1st Guess: 11 (100%)
2nd Guess: 0 (0%)
3rd Guess: 1 (0%)
4th
Original: 1
1st Guess: 1 (100%)
2nd Guess: 0 (0%)
3rd Guess: 2 (0%)
4th
Original: 12
1st Guess: 12 (100%)
2nd Guess: 0 (0%)
3rd Guess: 1 (0%)
4th
Original: 38
1st Guess: 38 (100%)
2nd Guess: 0 (0%)
3rd Guess: 1 (0%)
4th
Original: 34
1st Guess: 34 (100%)
2nd Guess: 0 (0%)
3rd Guess: 1 (0%)
4th

(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?