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

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

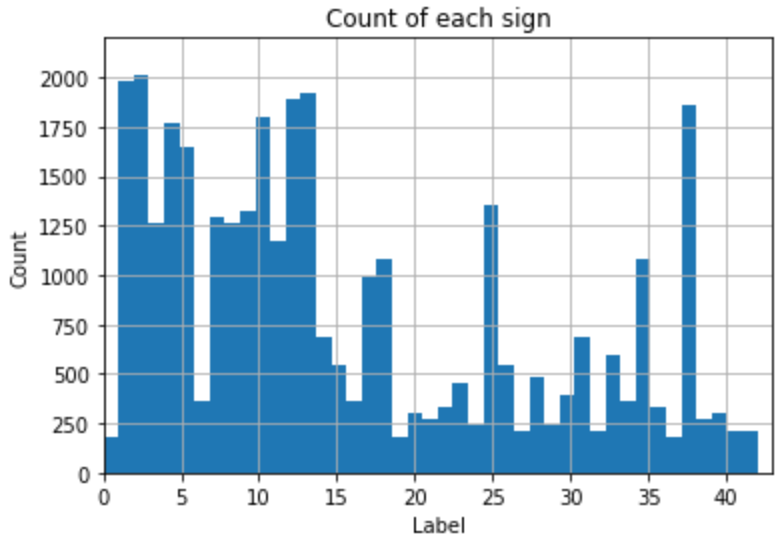
1. Summary of the data set.

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.Visualization of the dataset.

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

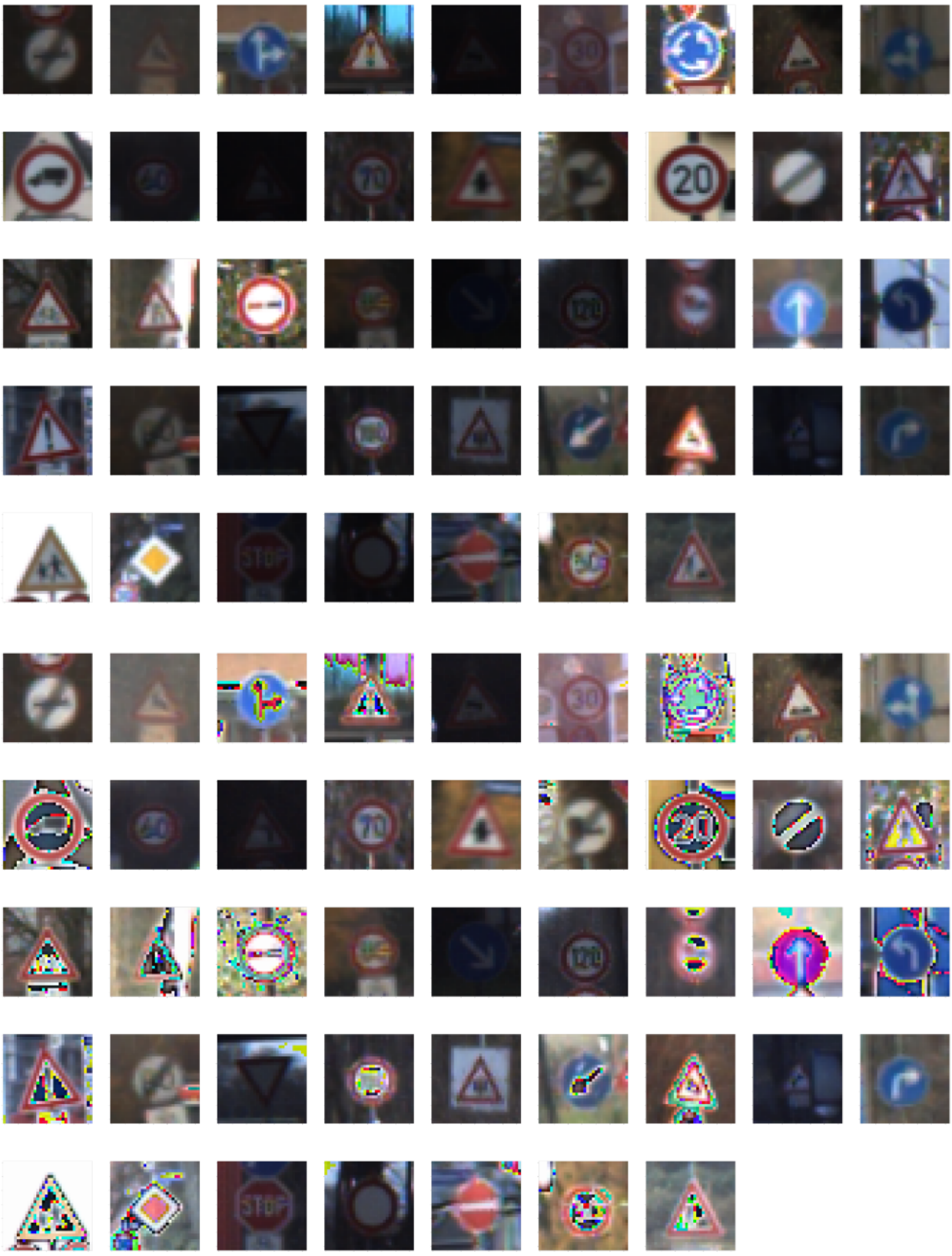


Design and Test a Model Architecture

1. Data Preprocessing

I decided to normalize the image, because it will make the algorithm more robustic.

Here is an example of a traffic sign image before and after normalization.



2. Model architecture

My final model is a LeNet model, consisted of the following layers:

Layer	Description
Input	32x32x3 RGB image
Convolution 3x3	1x1 stride, valid padding, outputs 28x28x12.
RELU	
Max pooling	2x2 stride, outputs 14x14x12
Convolution 3x3	1x1 stride, valid padding, outputs 10x10x32.
RELU	
Max pooling	2x2 stride, outputs 5x5x32
Flatten	Output 800
Fully connected	Output 240
RELU	
Fully connected	Output 129
RELU	
Fully connected	43

Softmax	

3. Model Training

To train the model, I set up an optimizer to minimize softmax cross entropy. The batch size is 128, epochs is 20 and learning rate is 0.001

4. Final Result

My final model results were:

- validation set accuracy of 0.936
- test set accuracy of 0.932

The code located in the 10th cell.

- LeNet architecture was chosen in this project.
- I iterated the optimizer to achieve a higher accuracy, since each time the start point is mostly better than the previous one, it will work well in image recognition.
- The final accuracy of validation set and test set are both above 0.93.

Test a Model on New Images

1.

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

The first image might be difficult to classify because is has watermark on it.

2. The predicitions are 31, 3, 1, 14, 19 which corresponding to Wild animals crossing, Speed limit 60km/h, Speed limit 30km/h, Stop and Dangerous curve to the left. The accuracy is 100%, higher than test results.

Here are the results of the prediction:

Image	Prediction
Wild animals crossing	Wild animals crossing
Speed limit (60km/h)	Speed limit (60km/h)

Speed limit (30km/h)	Speed limit (30km/h)
Stop	Stop
Dangerous curve to the left	Dangerous curve to the left

3.

The code for making predictions on my final model is located in the 20th cell of the Ipython notebook.

All the images have a very high probability to the right one.

Probability	**Prediction**
1.0	Wild animals crossing
1.0	Speed limit (60km/h)
0.95	Speed limit (30km/h)
0.99	Stop
1.0	Dangerous curve to the left

Belowings are details:

```
TopKV2(values=array([[1.00000000e+00, 2.69834932e-10, 1.04328579e-12, 1.43719096e-14,
        7.20789727e-16],
       [1.00000000e+00, 4.26080593e-09, 1.21420208e-13, 1.75116733e-17,
        2.37329471e-18],
       [9.46606040e-01, 5.33932634e-02, 3.80348581e-07, 1.35684999e-07,
        7.36242498e-08],
       [9.99228239e-01, 7.67847465e-04, 2.39677411e-06, 1.25565396e-06,
        1.17991696e-07],
       [1.00000000e+00, 2.95582083e-19, 2.08631811e-23, 4.38923825e-24,
        2.19245508e-24]], dtype=float32), indices=array([[31,  5, 21, 29, 23],
       [ 3,  2,  5, 28, 31],
       [ 1,  2, 21,  5, 31],
       [14,  3, 25,  9,  1],
       [19, 21, 25, 24, 37]], dtype=int32))
```