Traffic Sign Recognition

Build a Traffic Sign Recognition Project

The goals / steps of this project are the following:

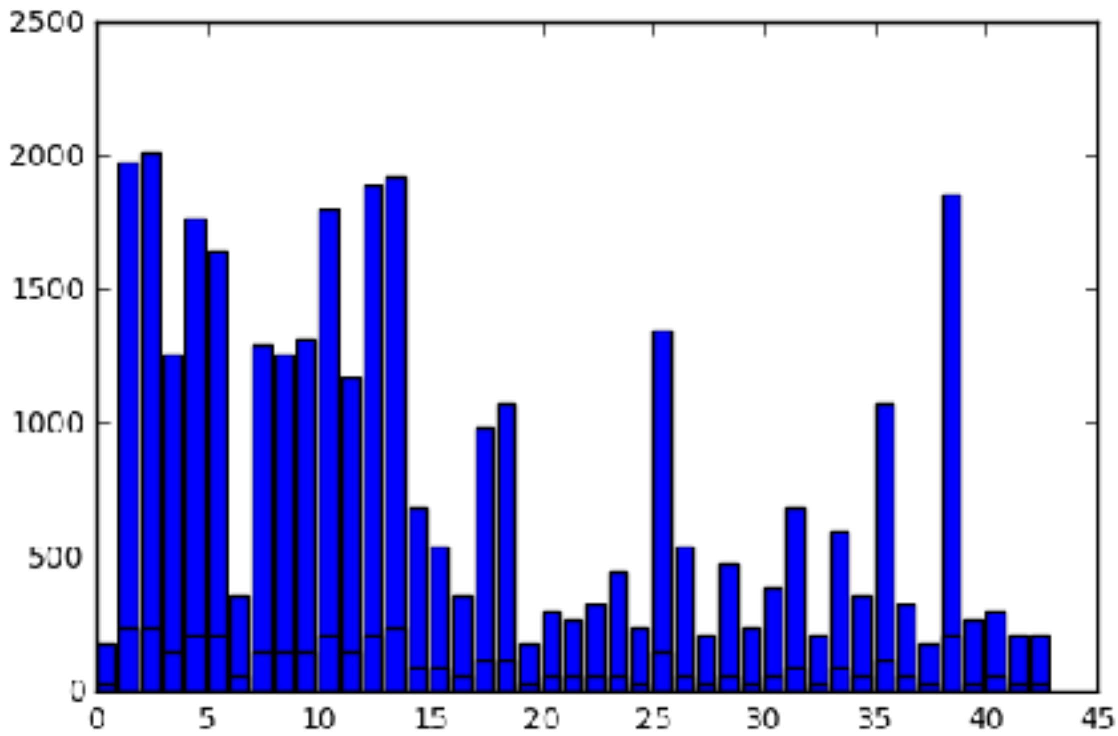
- Load the data set (see below for links to the project data set)
- Explore, summarize and visualize the data set
- Design, train and test a model architecture
- Use the model to make predictions on new images
- Analyze the softmax probabilities of the new images
- Summarize the results with a written report

Data Set Summary & Exploration

1. 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.

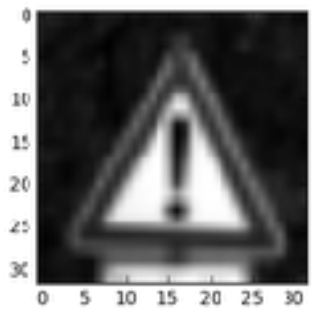


ClassID	SignName
0	Speed limit (20km/h)
1	Speed limit (30km/h)
2	Speed limit (50km/h)
3	Speed limit (60km/h)
4	Speed limit (70km/h)
5	Speed limit (80km/h)
6	End of speed limit (80km/h)
7	Speed limit (100km/h)
8	Speed limit (120km/h)
9	No passing
10	No passing for vehicles over 3.5 metric tons
11	Right-of-way at the next intersection
12	Priority road
13	Yield
14	Stop
15	No vehicles
16	Vehicles over 3.5 metric tons prohibited
17	No entry
18	General caution
19	Dangerous curve to the left
20	Dangerous curve to the right
21	Double curve
22	Bumpy road
23	Slippery road
24	Road narrows on the right
25	Road work
26	Traffic signals
27	Pedestrians
28	Children crossing
29	Bicycles crossing
30	Beware of ice/snow
31	Wild animals crossing
32	End of all speed and passing limits
33	Turn right ahead
34	Turn left ahead
35	Ahead only
36	Go straight or right
37	Go straight or left
38	Keep right
39	Keep left

```
40      Roundabout mandatory
41      End of no passing
42      End of no passing by vehicles over 3.5 metric
tons
```

Design and Test a Model Architecture
1.The image input goes through standard pre-processing steps meant for images
The steps employed are-
1. Grayscale conversion - image color is not a distinguishing feature for traffic signs. IOW there are no two traffic signs with different colors and same symbol.
2. Centering the image values - (x_train-128.0)/128.0 as these values work well with CNNs that have RELU activations.

Image after Preprocessing:



2.My final model consisted of the following layers:

Layer	**Description**
Input	| grayscale image | 32x32x1
Convolution 1	`5x5x1x6	1x1x1 stride, VALID padding	outputs 28x28x6`
Relu 1	
Max pooling	`2x2x1	2x2x1 stride, VALID padding	outputs 14x14x6`
Convolution 2	`5x5x6x16	1x1x1 stride, VALID padding	outputs 10x10x16`
Relu 2	

Max pooling	`2x2x1	2x2x1 stride, VALID padding	outputs 5x5x16`
Fully Connected	`400*120	outputs 120`
Relu 3	
Dropout 1	`0.7`
Fully Connected	`120*84	outputs 84`
Relu 4	
Dropout 2	`0.7`
Fully Connected	`84*43	outputs 43`
Softmax	

To train the model, I used Stochastic Gradient Descent optimized by Adam Optimizer at a learning rate of 0.001. Each batch was a randomized sample of 128 training samples. The loss converged for the validation set at around 21 epochs training on GPU.
The approach to classify the traffic symbols was to implement a standard LeNet CNN and iteratively tune it to improve performance for this specific dataset. The LeNet model comprises of a stack of two convolution layers and three fully connected layers with RELU activations interleaved between them. The convolutions layers outputs are also fed through MaxPooling layers after RELU. One of the changes that improved performance for this dataset is the inclusion of dropout layers connected to fully-connected layers. This was added when I noticed the model was overfitting to the training data set. Learning rate, batch size and the probability for the dropout layers were the most important hyper-parameters that I had to tune. My initial learning rate of 0.1 with the GradientDescent optimizer was failing to train, possibly getting stuck at a local optima. Reducing learning rate by an order was sufficient to get the model to train. I also switched the optimizer to Adam Optimizer as it converged significantly faster than GradientDescent.

The final model results, with just 21 epochs of training on GPU were -

- Training set accuracy of 0.955
- Validation set accuracy of 0.950567
- Test set accuracy 0.92692

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:

One of the interesting things I noticed was the model fails to classify a "known" traffic sign if the sign is not centered or does not cover a significant part of the image. Cropping the image to mostly include just the sign gives 100% accuracy. This shows that the dataset is insufficient and makes a good case for augmenting the data set with transformed images.

Another observation is that model appears to have low precision in some cases. Testing with an unseen input - "No stopping" results in the model classifying it with 70% accuracy as a "Roundabout mandatory". I believe, this probability would have been lesser if the color components were included in images used for training. The second and last image are not in the training dataset.

Here are the results of the prediction:

```
        Prediction                                        Actual
Sign 1: Road work                                         Road work
Sign 2: Roundabout mandatory                              No Stopping -
Not in the dataset
```

```
Sign 3: Right-of-way at the next intersection   Right-of-way at
the next intersection
Sign 4: Speed limit (60km/h)                    Speed limit
(60km/h)
Sign 5: Children crossing                       No Parking - Not
in the dataset
```

The model classifies 3 of the 5 traffic signs correctly but all 3 signs known to the model are classified with 100% accuracy.
The top five soft max probabilities for the 5 test data are below. The model classifies the first, third and fourth signs with almost 100% certainty. The rest two, second and fifth, are negative test cases where the model is expected to be not certain as these traffic signs are not in the training data.

```
        Sign                     Softmax Probability
Sign 1: Road work          '1.00', '0.00', '0.00', '0.00', '0.00'

Sign 2: Roundabout mandatory  '0.72', '0.28', '0.00', '0.00',
'0.00'

Sign 3: Right-of-way at the next intersection '1.00', '0.00',
'0.00', '0.00', '0.00'

Sign 4: Speed limit (60km/h) '0.99', '0.00', '0.00', '0.00',
'0.00'

Sign 5: Children crossing  '0.64', '0.36', '0.00', '0.00',
'0.00'
```

Visualising the Neural Network

Visualising the parameters of the first convolution layer for a 60 km/hr traffic sign looks like this: