Traffic Sign Recognition Project

Data Set Summary & Exploration

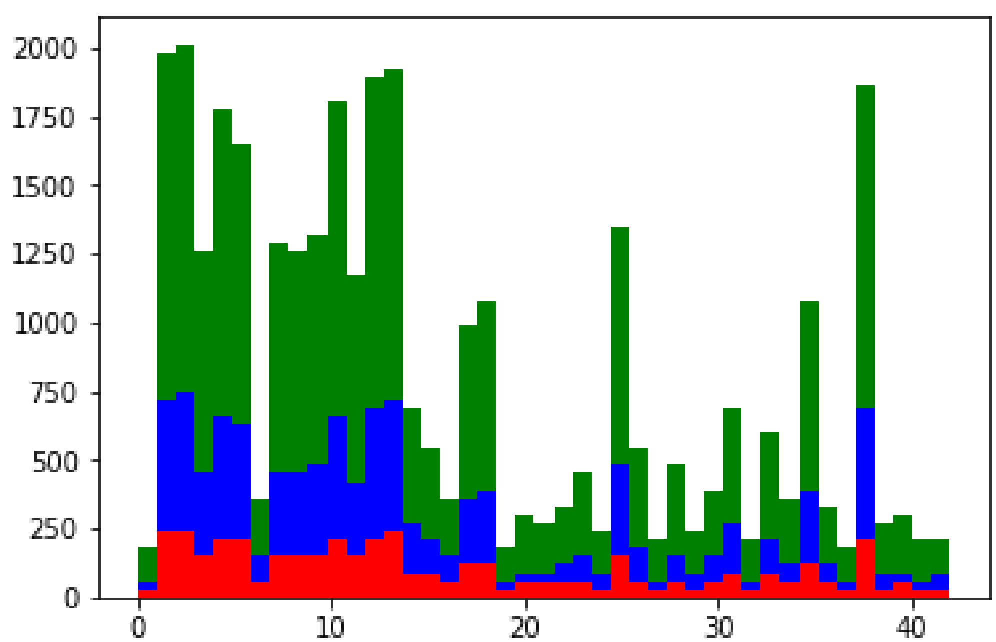
1. Provide a basic summary of the data set.

I calculated summary statistics of the traffic signs data set:

* Number of training examples = 34799
* Number of testing examples = 12630
* Number of validation examples = 4410
* Image data shape = (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 ...



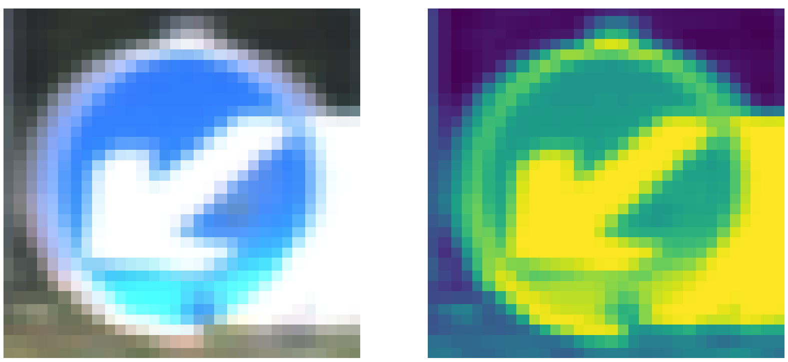
Design and Test a Model Architecture

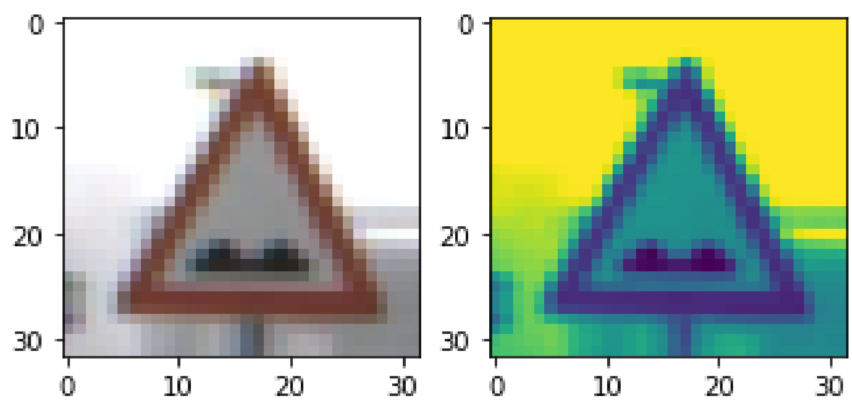
1. Preprocessing

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 gray-scale for some reasons:
-we don't need color component .
-Using RGB images consume more time than gray-scale image (Decreasing data to one third)

Here are 2 examples of a traffic sign images before and after gray-scaling.





As a last step, I normalized the image data from 0-255 range to 0.1-0.9 range for accuracy issues as doing calculations in wide range of numbers is less accurate than doing them in a small range

2.Model Architecture

My final model is the same as LeNet : (5 layers):{2 Convolutional and 3 Fully Connected layers}
Here is a description of it.

Layer/Process	Input Shape	Output Shape	Description
Convolutional-1	32x32x1	28x28x6	2D convolution : conv2d(x, w, strides=[1, 1, 1, 1], padding='VALID') + b
Activation	28x28x6	28x28x6	relu
Pooling	28x28x6	14x14x6	ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], valid padding
Convolutional-2	14x14x6	10x10x16	2D convolution : conv2d(x, w, strides=[1, 1, 1, 1], padding='VALID') + b
Activation	10x10x16	10x10x16	relu
Pooling	10x10x16	5x5x16	ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], valid padding
Flatten	5x5x16	400	Covert from 3D to 1D
Fully Connected-1	400	300	matrix multiplication(matmul(wx) + b)
Activation	300	300	relu
Dropout	300	300	keep_prob=0.7
Fully Connected-2	300	150	matrix multiplication (matmul(wx) + b)
Activation	150	150	relu
Dropout	150	150	keep_prob=0.7
Fully Connected-3	150	43	matrix multiplication (matmul(wx) + b)

3. Model Training

To train the model, I used an Adam optimizer with batch size =100 , no. of EPOCHS =50 , learning rate =0.003 , mu=0 , sigma=0.1 , keep_prob=0.7

I tried multiple set of parameters of (no. of EPOCHS , learning rate , batch_size).
And I found that :
– decreasing batch_size has a good impact till 100 then it get worse.
- keep_prob more than 0.7 is so bad , It doesn't ensure randomness and non-linearity of data.
- no. of EPOCHS =50 , learning rate =0.003 is a good pair of parameters.

4. Solution Approach

I used LeNet architecture.
Increasing no.of Epochs over 50 make over-fitting . The validation accuracy still the same or decreases.
Highly Increasing or decreasing batch_size isn't good as well .
Leaning rate is also suitable with the last values of no.of Epochs , batch_size

My final model results were:
* training set accuracy of 0.999
* validation set accuracy of 0.953
* test set accuracy of 0.937 ~ 0.945

Test a Model on New Images

1. The test pictures

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




2.Model's predictions

Sign name	real class	predicted class	First five predicted classes
`Go straight or left`	37	18	`[18  5  2 31  3]`
`General caution`	18	18	`[18  5  2 31 25]`
`Speed limit (60km/h)`	3	2	`[ 2 10  9  8  5]`
`Turn left ahead`	34	38	`[38 34 32 14  3]`
`Yield`	13	13	`[13 38 15 28 12]`

As shown in the previous table:

The network isn't so good although it has good testing and validation accuracy.

It has 2 of 5 right prediction from the first time and 1 of 5 right predictions from second time.

So it has an accuracy of 40 % to get the sign right from the first time . And 60 % to get it after 2 predictions according to this set.

3. Model certainty for prediction

```
Image 0 probabilities: [ 0.48180729  0.26821437  0.16585499  0.03625952
0.01893119]
 and predicted classes: [18  5  2 31  3]
Real sign name Go straight or left , Real sign class  37
------------------------------------
Image 1 probabilities: [ 0.48180729  0.26821437  0.16585499  0.03625952
0.01893119]
 and predicted classes: [18  5  2 31 25]
Real sign name General caution , Real sign class  18
------------------------------------
Image 2 probabilities: [ 0.90080762  0.03703117  0.0314642   0.01247476
0.00860436]
 and predicted classes: [ 2 10  9  8  5]
Real sign name Speed limit (60km/h) , Real sign class  3
------------------------------------
Image 3 probabilities: [  9.99952435e-01   4.54503606e-05   1.48997003e-06
7.36304628e-07
   1.17560562e-11]
 and predicted classes: [38 34 32 14  3]
Real sign name Turn left ahead , Real sign class  34
------------------------------------
Image 4 probabilities: [  1.00000000e+00   3.21964980e-24   2.24095039e-30
1.81391074e-31
   1.26817009e-31]
 and predicted classes: [13 38 15 28 12]
Real sign name Yield , Real sign class  13
```

Sign name	real class	predicted class	First five probabilities if right
Go straight or left	37	18	Not right
General caution	18	18	[**0.48180729** 0.26821437 0.16585499 0.03625952 0.01893119]
Speed limit (60km/h)	3	2	Not right
Turn left ahead	34	38	[0.999952435 **0.454503606** 1.48997003e-06 7.36304628e-07 1.17560562e-11]
Yield	13	13	[**1.00000000** 3.21964980e-24 2.24095039e-30 1.81391074e-31 1.26817009e-31]