Build a Traffic Sign Classifier Project

The goals / steps of this project are the following:

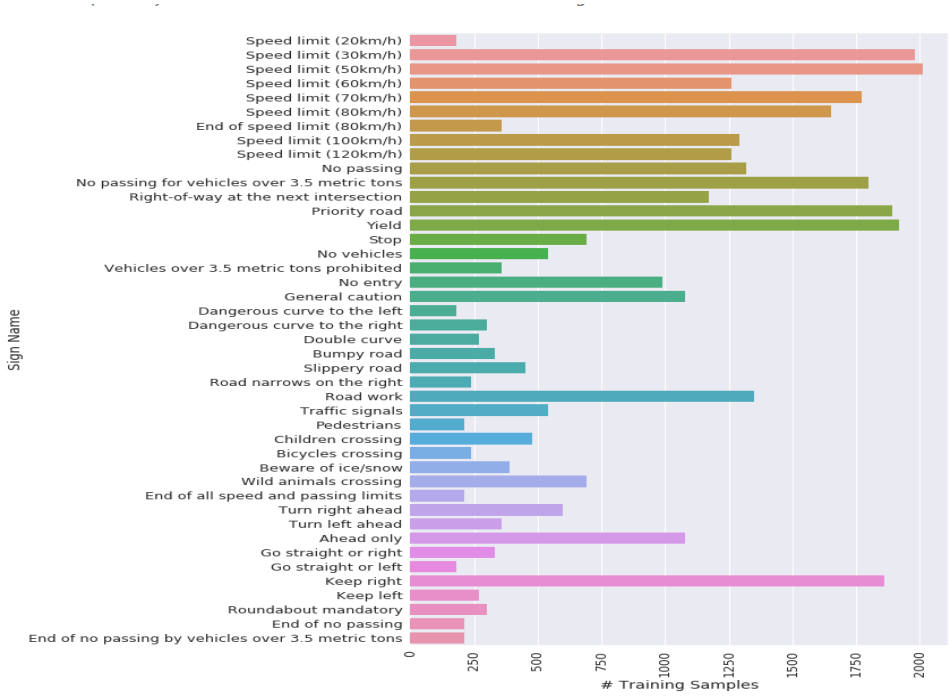
- Load the data set
- Explore, summarize and visualize the data set
- Preprocess images
- Design, train and test a model architecture
- Use the model to make predictions on new images
- Analyze the softmax probabilities of the new images
- Summarize the results with a written report

Data Set Summary & Exploration:

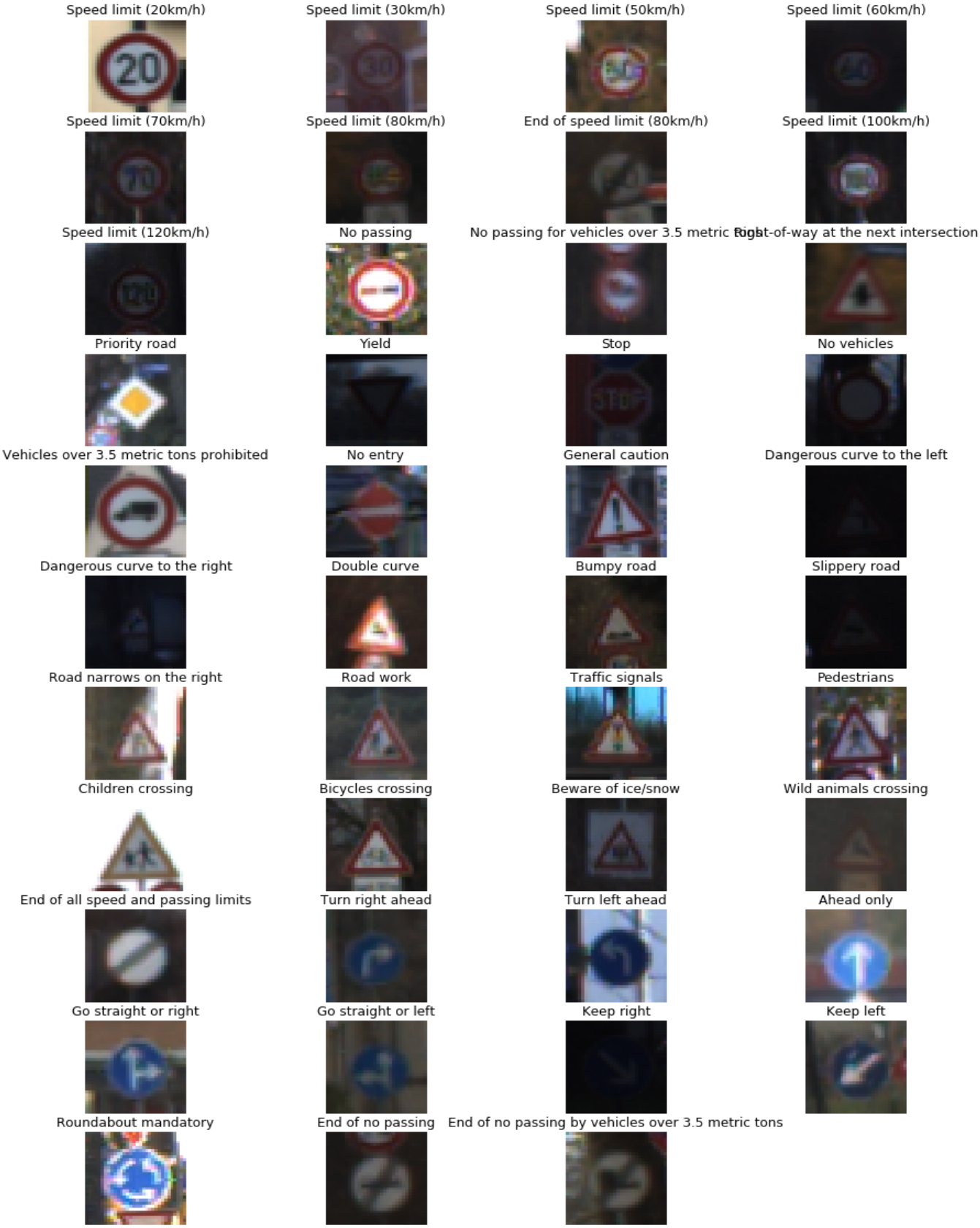
I used the pandas and numpy libraries to calculate summary statistics of the traffic signs data set:

- 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

1- Data distribution across the different labels:



2- Exploratory visualization of the data set:

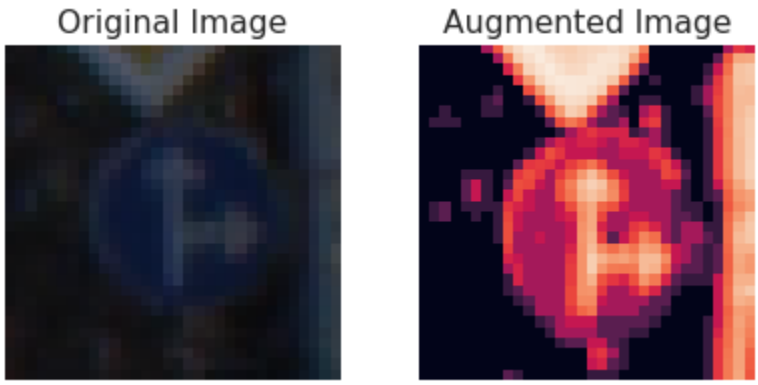


Design and Test a Model Architecture:

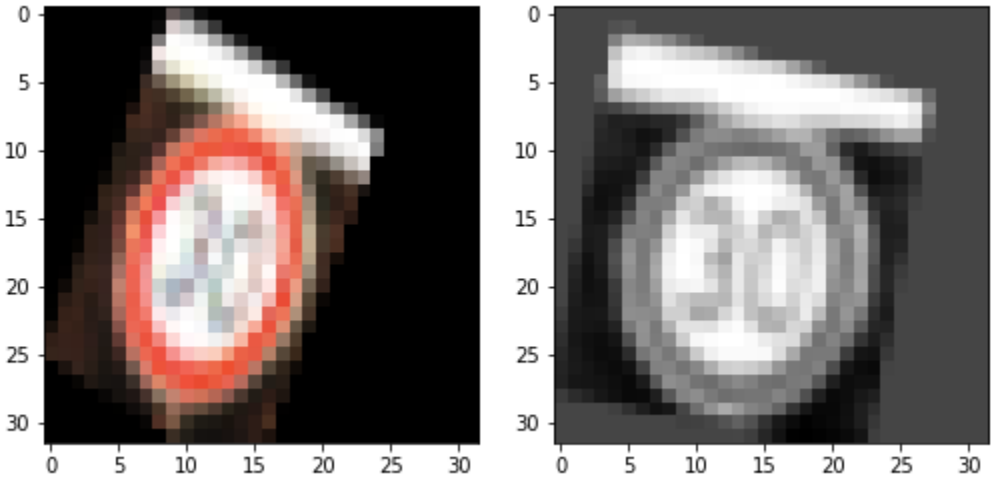
1 – Preprocessing pipeline

- **Normalizing the image data** to have image data with zero mean and equal variance if its not case then it will be difficult for optimizer to find solution and because very large or very small numbers can lead to numerical instability .

- **Data augmentation** to achieve this I have used transform image function which operates on image to give Image Rotation, Translation and Share. That helps to reduce overfitting.



- **Grayscale conversion** because color isn't an important feature in classifying traffic signs, there is more important features like: shape and geometry.



2 - *Model architecture looks like*

Layer	Description
Input	32x32x1 Grayscale image
Convolution_1 5x5	1x1 stride, VALID padding, outputs 28X28X6
RELU	
Dropout	Keep probability = 0.7
Max pooling	2x2 stride, outputs 14x14x6
Convolution_2 5x5	1x1 stride, VALID padding, outputs 10x10x16
RELU	
Max pooling	2x2 stride, outputs 5x5x16
Fully connected_0	Output = 400.
Dropout	Keep probability – 0.6
Fully connected_1	Output = 120.
RELU	
Fully connected_2	Output = 84.
RELU	
Dropout	Keep probability = 0.6
Fully connected_3	Output = 43.

3- Training model

I have used the same Lenet Model as given in class tutorial, and to improve validation accuracy of the model, I have tried different approaches to increase accuracy like adding dropout at fully connected layer, also changing keep probability value, addition of dropout at convolution 1 layer with different keep probability also find out that ADAM optimizer gives better accuracy than SGD optimizer. Also I have tried to change the number of filters of the convolutional layer but it didn't increase accuracy too much , so I keep it like the tutorials .Actually the most important thing in this problem is prepressing cause it prepare the data and gives the network the chance to learn the important features.

4 - approach taken for finding a solution

- **My final model results were:**

  Accuracy Model on Training Images: 0.96
  Accuracy Model on Validation Images: 0.95
  Accuracy Model on Test Images: 0.93

- **Hyperparameters of the network :**

  Learning Rate = 0.0005

  I have tried a range of learning rates and a range from (.001 - .0001) gives great results .

  Epoch = 60

  I have tried different number of epochs (60-80-100-120) ,and I found that error almost stay steady after the epoch 60.

  batch Size = 128 (try and error)

  keep_prop = .7 (try and error)

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:

All these images maybe challenging to classify because:

- The have different sizes.
- One of them have a background different from what the model trained on .

Since these images are not in the right shape accepted by the classifier they were resized and smoothed before applying the trasnform_img function.

Here are the results of the prediction:

```
Top 5 Labels for image 'Double curve':
 - 'Beware of ice/snow' with prob = 0.24
 - 'Dangerous curve to the right' with prob = 0.21
 - 'Road work' with prob = 0.18
 - 'Road narrows on the right' with prob = 0.11
 - 'Double curve' with prob = 0.08
Top 5 Labels for image 'Children crossing':
 - 'Road work' with prob = 0.26
 - 'Priority road' with prob = 0.16
 - 'Children crossing' with prob = 0.13
 - 'Keep left' with prob = 0.11
 - 'Dangerous curve to the right' with prob = 0.05
Top 5 Labels for image 'Speed limit (50km/h)':
 - 'Speed limit (50km/h)' with prob = 0.88
 - 'Speed limit (30km/h)' with prob = 0.12
 - 'Speed limit (80km/h)' with prob = 0.01
 - 'Speed limit (60km/h)' with prob = 0.00
 - 'Speed limit (100km/h)' with prob = 0.00
Top 5 Labels for image 'Stop':
 - 'Road work' with prob = 0.67
 - 'Turn left ahead' with prob = 0.25
 - 'Go straight or left' with prob = 0.03
 - 'Yield' with prob = 0.03
 - 'Keep left' with prob = 0.01
Top 5 Labels for image 'Go straight or left':
 - 'Turn left ahead' with prob = 0.99
 - 'Go straight or left' with prob = 0.01
 - 'Keep right' with prob = 0.00
 - 'Ahead only' with prob = 0.00
 - 'Beware of ice/snow' with prob = 0.00
Top 5 Labels for image 'Speed limit (80km/h)':
 - 'Keep right' with prob = 0.98
 - 'Yield' with prob = 0.01
 - 'Priority road' with prob = 0.00
 - 'General caution' with prob = 0.00
 - 'Speed limit (50km/h)' with prob = 0.00
```