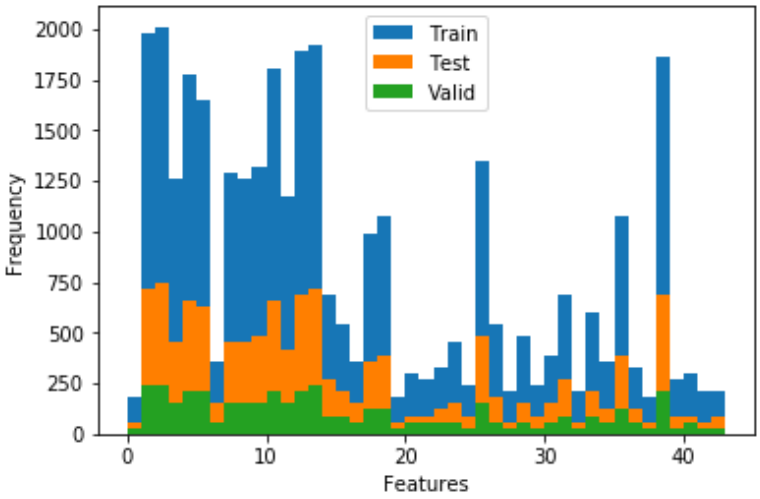
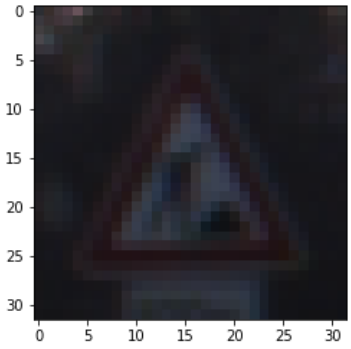
Project 2: Traffic Signal Classifier

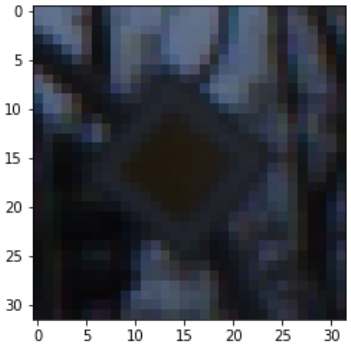
DATASET EXPLORATORY:

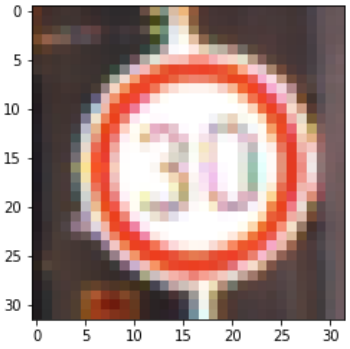
Number of training examples: 34799
Number of testing examples: 12630
Image data shape: 32 x 32 x 3
Number of classes: 43



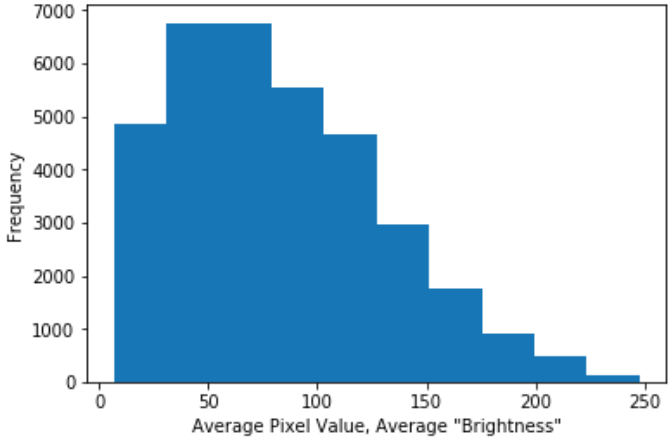
Histogram of classes vs occurrence in dataset.







Images are spread out over a spectrum of brightness values. Traffic signals have a variety of shapes, sizes and coloration.



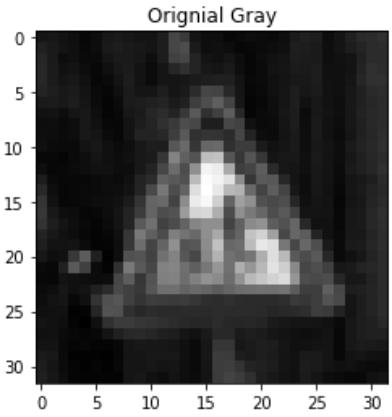
Histogram of average pixel value in an image.

Traffic Signs in Data set. A preliminary observation is that some of the traffic sign images have traffic signs at an angle. It might help if the preprocessed images are augmented with traffic signs

DESIGN AND TEST A MODEL ARCHITECTURE:

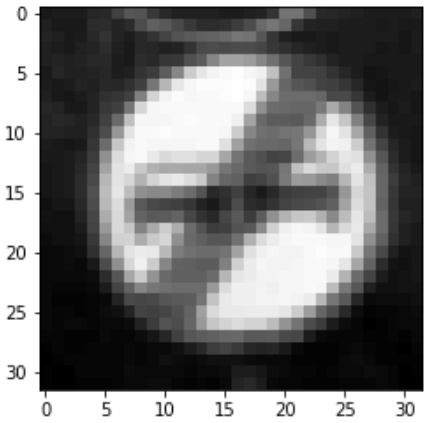
Preprocessing:

GrayScale Conversion: Convert 3 channel image to 1 channel image i.e. RGB to Grayscale.



Normalization and Zero-mean Conversion: Centering pixel values around zero and normalizing with respect to standard deviation.

Max Pixel: 97.0 —> 1.969
Min Pixel: 17.0 —> -1.106
Average Pixel Value Original Image: 45.7
Standard Deviation: 26



Model Architecture:

Layer 1:
- Type: Convolution
- Filter Size: 5 x 5 x 10
- Stride: 1
- Padding: Valid
- Input: 32 x 32 x 1 (Grayscale Image)
- Output: 28 x 28 x10

Layer 2:
- Type: Convolution
- Filter Size: 5 x 5 x 20
- Stride: 1
- Padding: Valid
- Input: 28 x 28 x 10
- Output: 24 x 24 x 20

Layer 3:
- Type: Convolution
- Filter Size: 5 x 5 x 30
- Stride: 1
- Padding: Valid
- Input: 24 x 24 x 20
- Output: 20 x 20 x 30

Layer 3.1:
- Type: Pooling
- Size: 2
- Stride: 2
- Input: 20 x 20 x 30
- Output: 10 x 10 x 30

Layer 4:
- Type: Convolution
- Filter Size: 5 x 5 x 40
- Stride: 1
- Padding: Valid
- Input: 10 x 10 x 30
- Output: 6 x 6 x 40

Layer 4.1:
- Type: Pooling
- Size: 2
- Stride: 2
- Input: 6 x 6 x 40
- Output: 3 x 3 x 40

Layer 5:
- Type: Fully Connected
- Input: 360 (flattened)
- Output: 180

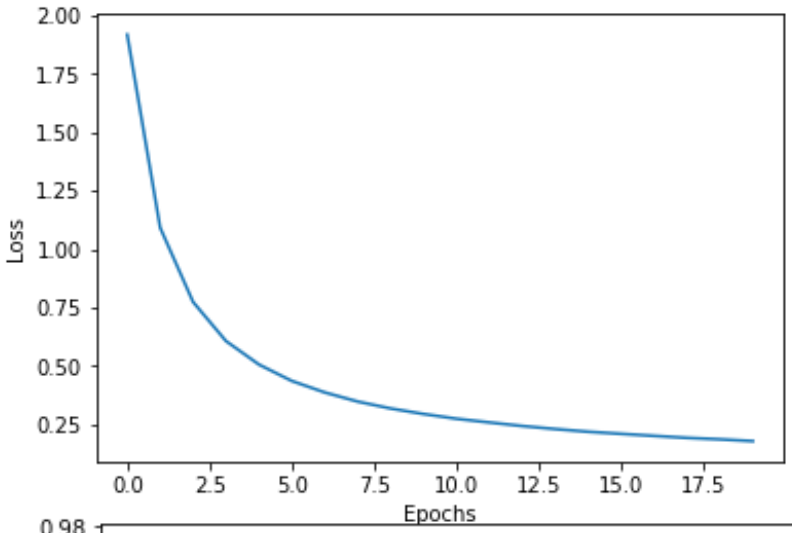
Layer 6:
- Type: Fully Connected
- Input: 180
- Output: 90

Layer 7:
    Type: Fully Connected
    Input: 90
    Output: 43

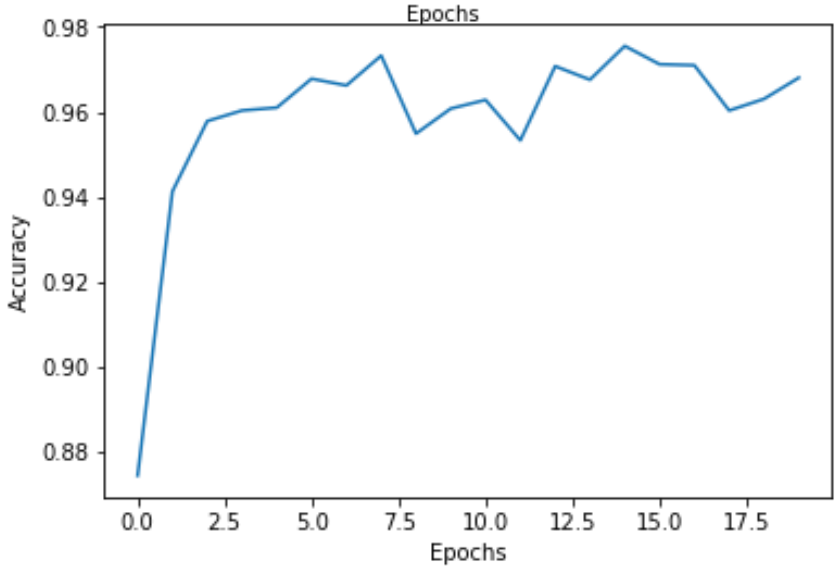
Model Training:

Optimizer: Adam Optimizer
Hyperparameters:
    Learning rate: .002 -.000003*epoch_i*epoch_i
Batch Size: 128
Epochs: 20

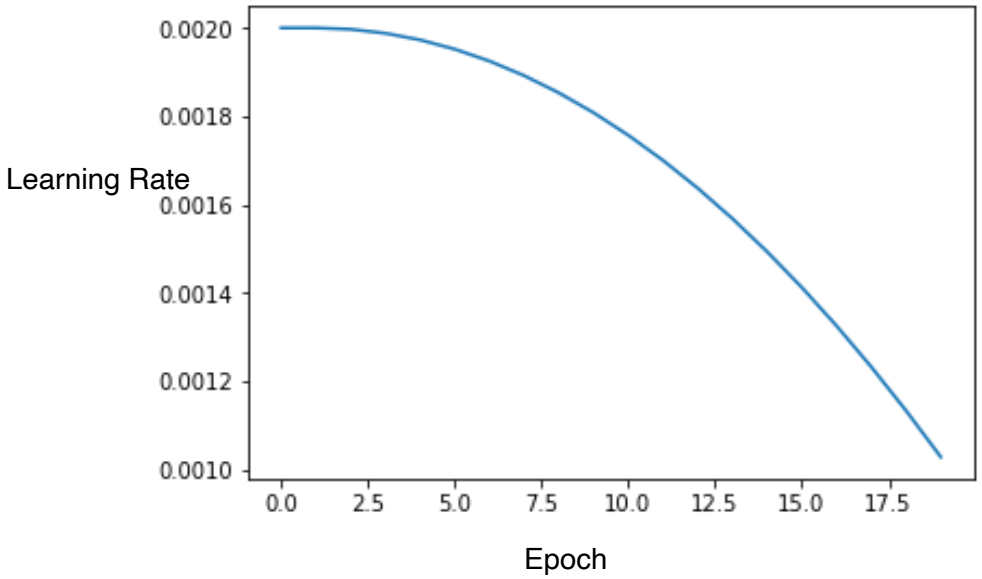
Loss VS Epoch



Accuracy Vs Epoch



Learning Rate Vs Epoch



Approach to Finding the solution:

Step 1. Identified preprocessing techniques to improve overall accuracy of training.
- Grayscale Images
- Zero mean Images
- Normalized Images
- Balanced Data set: For balancing data set, augmentation techniques were implemented

Step 2. Starting with LeNet network trained the network to understand the effect of learning rate, batch and epochs on the accuracy and training loss.

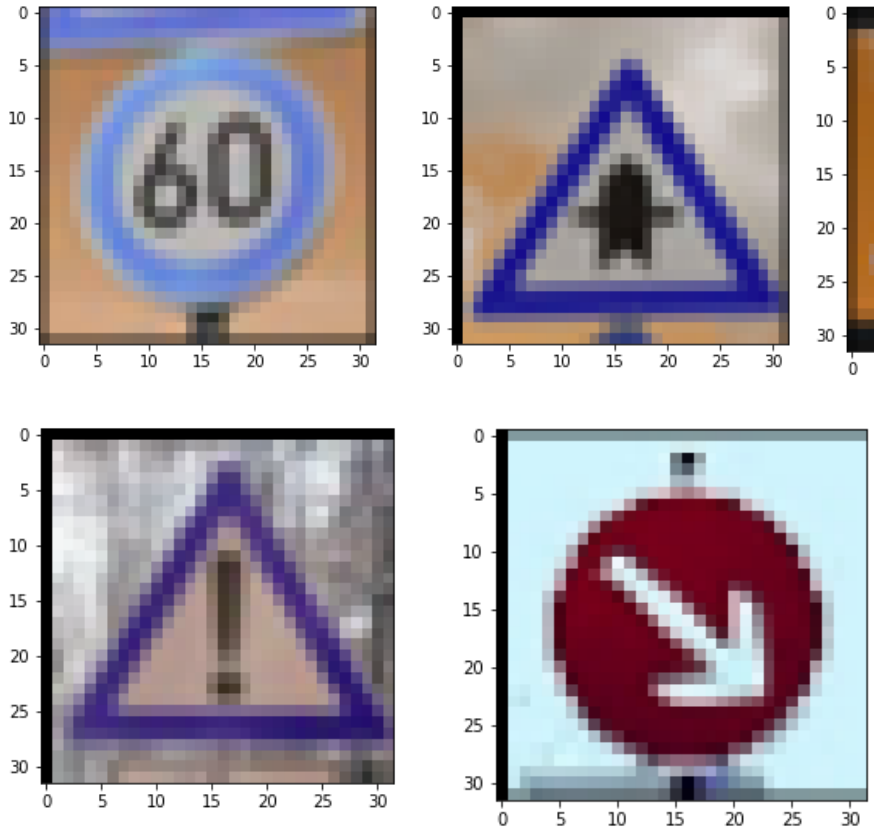
Step 3. Added CNN layers and reduced pooling in LeNet to extract more features

Step 4. Tweaked learning rate and found a learning rate which works well enough to train the model

FINAL Results:

Epochs : 20
Validation Accuracy: 0.968
Test Accuracy: 0.94
Loss: 0.17

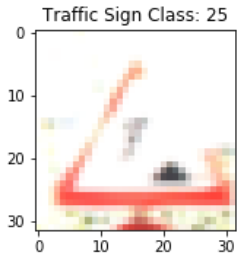
TEST Images:



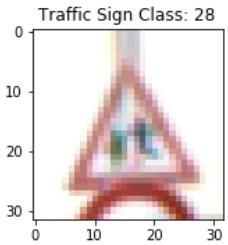
The 5 test images were pulled off the internet. The images were bigger than the required 32x32 resolution for the model and were resized. The images should not be difficult to classify even though during the RGB coloration got mixed up. Due to resizing the images did get pixelated. This should not be an issue as the training images were also pixelated.

Performance:

The model predicts 4 out of 5 images correctly i.e. 80% accuracy. The accuracy with the test data available with the assignment was 94%. There seems to be a disconnect in the accuracy. Image 3 was incorrectly identified at class 28 . The image is class 25. Class 28 and class 25 have slight similarity as can be seen below. It might be possible to fix this error in classification by augmenting the data for these particular classes and retraining the model.



Class 25 (Left)



Class 28 (Right)

MODEL CERTAINITY:

```
[[ 1.00000000e+00,   3.03101615e-08,   1.84917229e-12,
         1.52913652e-12,   3.82816419e-13],
       [ 1.00000000e+00,   3.84411965e-18,   3.21992823e-19,
         1.37902542e-22,   1.12543327e-22],
       [ 9.92984772e-01,   3.66540626e-03,   2.03930447e-03,
         9.42560611e-04,   1.33355265e-04],
       [ 1.00000000e+00,   2.48177090e-10,   2.06301533e-19,
         4.31393495e-20,   3.46104582e-20],
       [ 1.00000000e+00,   4.92328905e-19,   2.62176270e-25,
         3.58431586e-26,   1.37215375e-26]], dtype=float32),
 indices=array([[ 3,  5,  2, 36, 35],
                [11, 20, 30, 25, 27],
                [28, 41, 22, 32, 29],
                [18, 26, 39, 20, 11],
                [38,  2, 14, 12, 34]], dtype=int32))
```

The model predicted the classes with a certainty of ~1 for the correctly predicted classes with residual uncertainty being zero for all intensive purposes. The incorrectly predicted class was predicted with a certainty of .992 and with minuscule technical certainty.