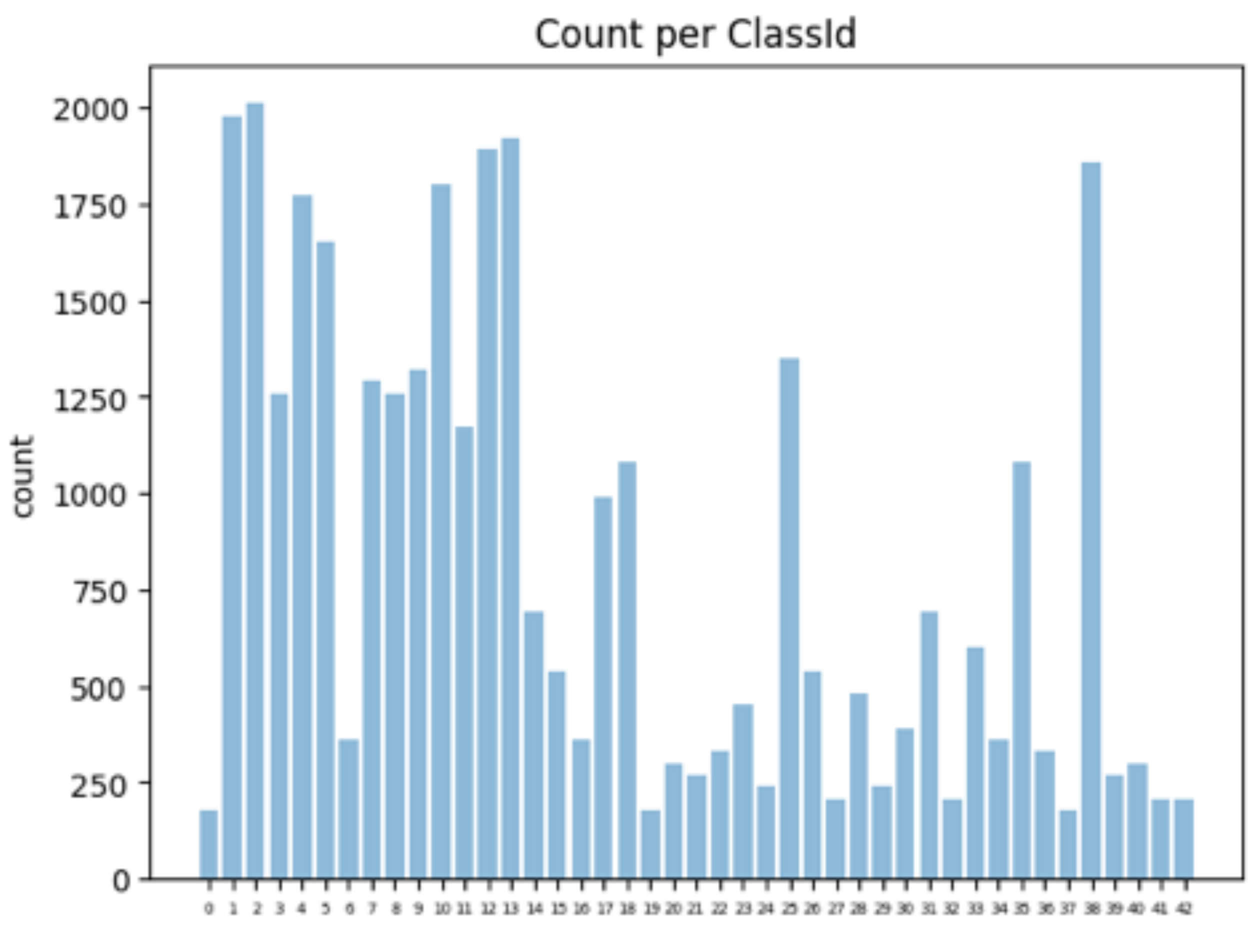
- ## Data Set Summary & Exploration

  I used the numpy 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



  There is unbalanced distribution of dataset amongst 43 classes as seen below. To avoid classifier biased towards highly represented class, there should be balanced distribution of images in 43 class labels.

- ## Design and Test a Model Architecture

  Describe how you preprocessed the image data.

What techniques were chosen and why did you choose these techniques?
As a first step, I used rgb images with zero mean normalisation (R-128/128,G-128/128,B-128/128) for image classification that resulted in 96pc cross validation but 90pc on validation set provided in valid.p. This was with original LeNet model.

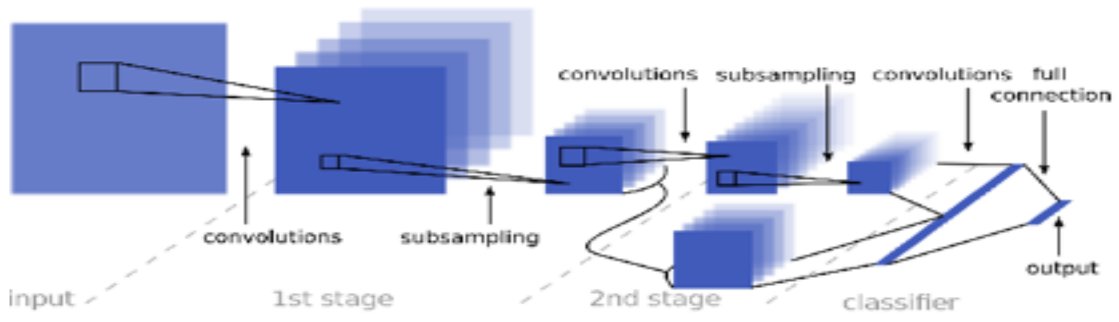
I normalised the image data using zero mean technique because gradient descent is an optimisation technique used in LeNet that updates weights after every epoch. Image features being on different scales, certain weights may update faster than others since the feature values play a role in the weight updates. This may lead to false positive condition and machine learning accuracy may drop. So a well balanced features set was created using normalisation method mentioned in the project.

Then I decided to convert the images to grayscale. With grayscale + normalisation on LeNet model, I could see increase in validation set accuracy to 92.9 pc and cross validation crossed 97pc. Classifier is still overfitted as training accuracy exceeds validation accuracy.

Here is an example of a traffic sign training image(X_train[6658]) before and after grayscaling. After grayscale, there is remarkable improvement seen in features variation for below image.

Describe what your final model architecture looks like including model type, layers, layer sizes, connectivity, etc.) Consider including a diagram and/or table describing the final model.

I further used modified LeNet for image classification as seen in the below table. This helped me achiever 99 pc on cross validation, 95 on validation set and 94 on test dataset.



Modified LeNet Architecture

Layers	Input	Output
1st Convolutional	32*32*1	28*28*1
RELU		
MaxPool	28*28*1	14*14*1
2nd Convolutional	14*14*1	10*10*16
RELU		
MaxPool	10*10*16	5*5*16
3rd Convolutional	5*5*16	1*1*400
RELU		
Fully Connected L1: Concatenate flatten neural of 2nd Layer output and 3rd layer output	1*1*400 + 5*5*16	800
Dropout	0.5(Training)	
Fully Connected L2	800	43

Why did you believe it would be relevant to the traffic sign application?
Modified LeNet model architecture uses convolutional neural network to train the image classifier. CNN recognises features of various images through forward propagation and updating weights through back-propagation/gradient decent. CNN learns image features like curves, lines and color variations on its own through different convolutional layers and thus the best fit for traffic sign image classification.

Further there are dropout and max-pooling techniques used in the model to avoid overfitting of dataset.

How does the final model's accuracy on the training, validation and test set provide evidence that the model is working well?
There is increase in validation accuracy as weights got updated after every epoch. I got cross validation accuracy as 99.3. This proves that model works well for Traffic sign classification.

Describe how you trained your model. The discussion can include the type of optimizer, the batch size, number of epochs and any hyper parameters such as learning rate.

To train the model, I used an Adam Optimiser with a batch size of 128 and 50 epochs. Learning rate used is 0.001
My final model results were:

- training set accuracy of 99.3
- validation set accuracy of 95.5
- test set accuracy of 94

Test a Model on New Images

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.

I chose 1(speed limit 30), 13(Yield), 14(Stop), 17(no entry) and 38 (right only) class images.
Class label 38 has faded features with some portion of traffic sign appear white that could make classifier difficult to detect it. Also class label 17 is tilted that could make classifier difficult to detect the features of this test image.

Discuss the model's predictions on these new traffic signs and compare the results to predicting on the test set. At a minimum, discuss what the predictions were, the accuracy on these new predictions, and compare the accuracy to the accuracy on the test set
Here are the results of the prediction:

The model was able to correctly guess 5 of the 5 traffic signs, which gives an accuracy of 100%. This compares favourably to the accuracy on the test set that has 94 percent accuracy.

Describe how certain the model is when predicting on each of the five new images by looking at the softmax probabilities for each prediction. Provide the top 5 softmax probabilities for each image along with the sign type of each probability.

Top five probabilities for each image corresponds to the correct class label.

INFO:tensorflow:Restoring parameters from ./lenet

```
TopKV2(values=array([[  9.70542908e-01,   2.94326674e-02,
2.36142805e-05,
          5.50055745e-07,   2.33017118e-07],
       [  1.00000000e+00,   2.59534838e-10,
2.97691385e-11,
          2.82503552e-11,   1.04604398e-11],
       [  1.00000000e+00,   6.82174925e-16,
4.02100503e-17,
          1.11501366e-17,   7.12255292e-19],
       [  9.92293477e-01,   7.70496251e-03,
1.51832808e-06,
          1.05853325e-12,   1.74840719e-13],
       [  1.00000000e+00,   4.41269556e-11,
9.58542273e-13,
          1.95975811e-15,   6.32974592e-16]],
dtype=float32),

indices=array([[ 1,  4,  0,  5,  8],
       [13, 35, 12, 25, 15],
       [14, 12, 13, 33, 17],
       [17, 34,  9, 38, 10],
       [38, 13, 34, 25,  9]], dtype=int32))
```