**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

**Data Set Summary & Exploration**

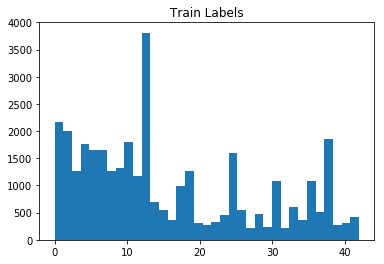
**1. Provide a basic summary of the data set**

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

* The size of training set is 34799
* The size of the validation set is 12630
* The size of test set is 4410
* The shape of a traffic sign image is 32x32x3
* 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 train labels are distributed.



### Design and Test a Model Architecture

#### 1. Image Preprocessing

As a first step, I decided to convert the images to grayscale because usually color information is not needed.

Next, I normalized the image data to have zero mean and equal variance. For image data, `(pixel - 128)/ 128` is a quick way to approximately normalize and has been used.

I tried subtracting mean and dividing by standard deviation and it did not work well.

#### 2. Model description

The model is very similar to LeNet. I have added an additional conv layer. I have also added drop out after the first two FC layers. Increased the no of filters in each conv layer so that network can learn more patterns/features.

#### 3. Model training

To train the model, I used the Adam Optimizer and cross entropy. The model ran 15 epochs. I initially chose a batch size of 128 and later reduced it to 64 as I saw better training with 64.

#### 4. Model selection

Tried different architectures. Initially I tried LeNet and found that it was not giving good accuracy. Then I added a conv layer and saw little improvement in accuracy. Then I stated to experiment with different no of filters in conv layers. I also added drop out after every FC layer so that network does not overfit and generalize well as usually there are more no of parameters in FC layer.