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
* Summarize the results with a written report

Data Set Summary & Exploration

**1. Provide a basic summary of the data set. In the code, the analysis should be done using python, numpy and/or pandas methods rather than hardcoding results manually.**

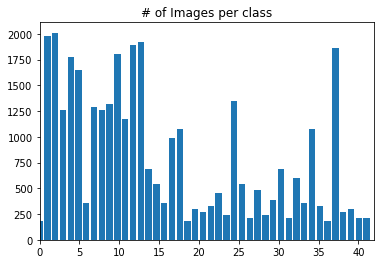
The size of training set is 34799

The size of the validation set is ?

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

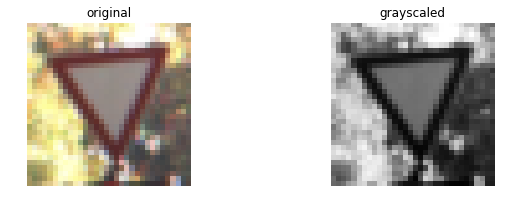
**2. Include an exploratory visualization of the dataset.** Here is an exploratory visualization of the data set. 

**Design and Test a Model Architecture**

**1. 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 grayscale because color attribute is not needed in this classification problem.

Here is an example of a traffic sign image before and after grayscaling.

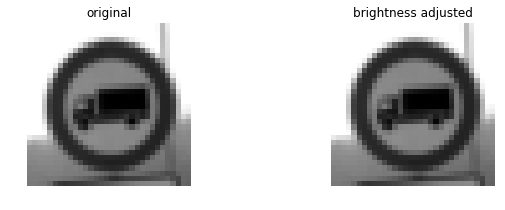


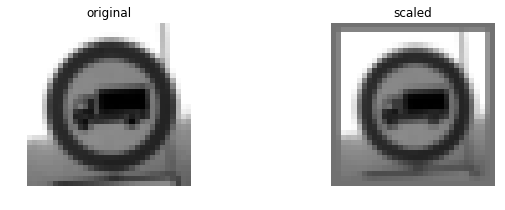
I also normalized the image data to help make solution more well-conditioned. This will help finding the solution quickly.

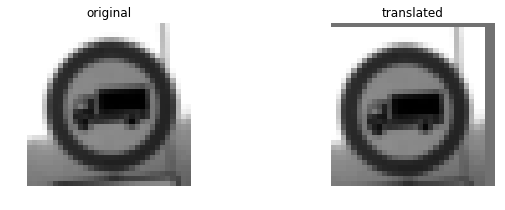
I first decided to use the data set without augmentation along with as is LENET but could not cross training accuracy of 85%. I added dropout to LENET and even then, the accuracy was hovering around 88 to 90%.

In augmentation I adjusted brightens, translated and scaled the images in classes with less than 800 images.

Here is an example of an original image and an augmented image:







**2. 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.**

My final model consisted of the following layers:

Layer Description

|  |
| --- |
| Input 32x32x1 RGB image |
| Convolution 5x5 1x1 stride, Valid padding, outputs 28X28X6 |
| RELU |
| Max pooling 2x2 stride, outputs 14X14X6 |
| Convolution 5x5 1x1 stride, Valid padding, outputs 10X10X16 |
| RELU |
| Max pooling 2x2 stride, outputs 5X5X6 |
| Flatten |
| Fully connected Input 400, output 300 |
| RELU |
| DROPOUT |
| Fully connected Input 300, output 120 |
| RELU |
| DROPOUT |
| Fully connected Input 120, output 84 |
| RELU |
| DROPOUT |
| Fully connected Input 84, output 43 |

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

Here are my parameters

* epochs: 100
* batch size: 128
* learning rate: 0.0009
* mu: 0
* sigma: 0.1
* dropout keep probability: 0.5

**4. Describe the approach taken for finding a solution and getting the validation set accuracy to be at least 0.93. Include in the discussion the results on the training, validation and test sets and where in the code these were calculated. Your approach may have been an iterative process, in which case, outline the steps you took to get to the final solution and why you chose those steps. Perhaps your solution involved an already well known implementation or architecture. In this case, discuss why you think the architecture is suitable for the current problem.**

My final model results were:

Train Accuracy = 0.999

Valid Accuracy = 0.959

Test Accuracy = 0.925

Approach:

* I started off by creating my own model(based on LENET), started with smaller strides sizes and added 2 extra fully connected layer but could not get validation accuracy greater than 80%. I played around with epochs, learning rate and batch size but I couldn’t tell if the model was improving. The results would vary +/- 6% accuracy. I then added dropout it helped but could quite get to 90% validation accuracy
* I augmented the data by using methods from previous Udacity’s submitted project [Reference](https://github.com/jeremy-shannon/CarND-Traffic-Sign-Classifier-Project). I added about 12K more imaged to original data set.
* With my original model I could get up to 90% validation accuracy.
* I switched to LENET with dropout. Using augumented dataset, I was able to get validation accuracy of 95.9%

**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.**

I downloaded 8 German traffic sign images from internet. I resized them to 32X32 to meet LENET architecture requirements. Some of the pictures had copyright text in the background, I intentionally picked those to see the sensitivity of my model. Quality wise pictures were bright, good contrast and good aspect ratio.

**2. 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 (OPTIONAL: Discuss the results in more detail as described in the "Stand Out Suggestions" part of the rubric).**

I got 100% accuracy in my testing.

My images were



**3. 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.**

Example: Softmax probablilty for the first image is below

[[0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00

0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00

0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00

0.00000000e+00 3.45556454e-31 0.00000000e+00 1.50381581e-28

0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00

0.00000000e+00 0.00000000e+00 **1.00000000e+00[Bumpy road, 22]** 0.00000000e+00

0.00000000e+00 0.00000000e+00 4.33371462e-26 0.00000000e+00

0.00000000e+00 8.67911151e-30 0.00000000e+00 0.00000000e+00

0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00

0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00

0.00000000e+00 0.00000000e+00 0.00000000e+00]

