Project 3: Traffic Sign Classifer

Reflection

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

1. Provide a Writeup / README that includes all the rubric points and how you addressed each one. You can submit your writeup as markdown or pdf. You can use this template as a guide for writing the report. The submission includes the project code.

Rubric

Criteria	Description	Status
Submission	Ipython notebook with code	✓
	HTML output of the code	✓
	A writeup report (either pdf or markdown)	✓
Dataset exploration	Dataset Summary	✓
	Exploratory Visualization	✓
Design and Test a Model Architecture	Preprocessing	✓
	Model Architecture	✓
	Model Training	✓
	Solution Approach	✓
Test a Model on New Images	Acquiring New Images	✓
	Performance on New Images	✓
	Model Certainty - Softmax Probabilities	✓

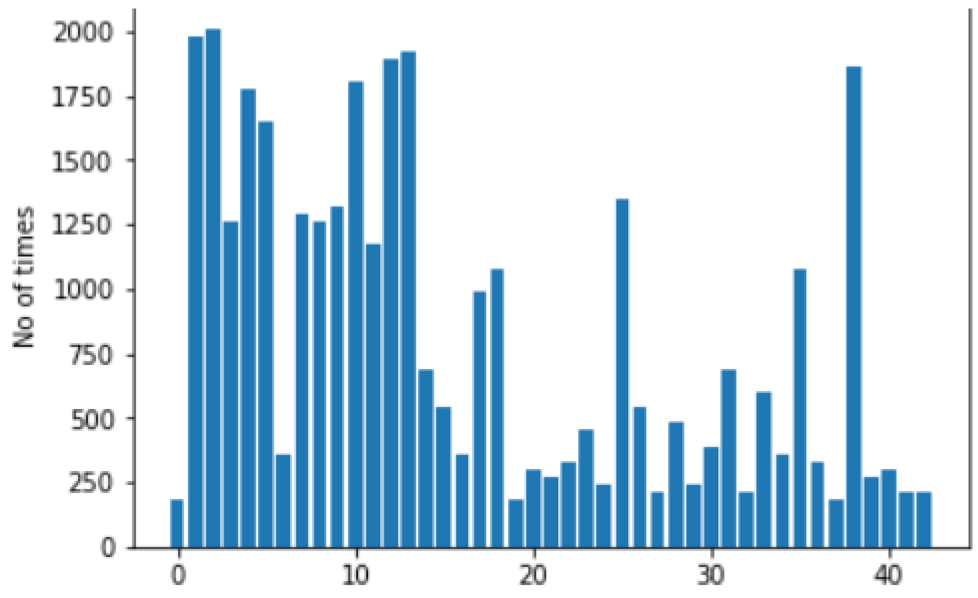
Data Set Summary & Exploration

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.

I used the pandas 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`

Include an exploratory visualization of the dataset.



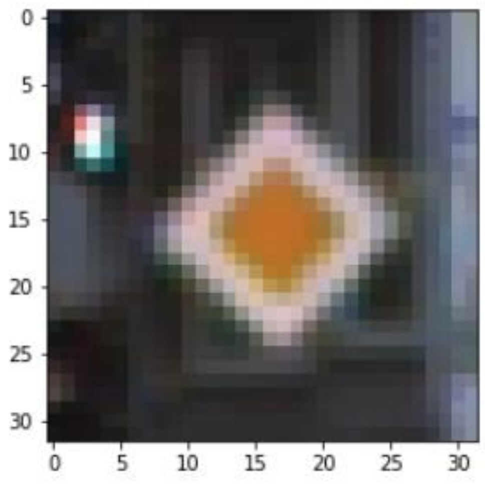
The figure above shows the distribution of the classes of the data set. This visual distribution gives an idea which labels might be classified inaccurate (based on the low count of the data). More of its data and augmentation can be done to increase the accuracy.

Design and Test a Model Architecture

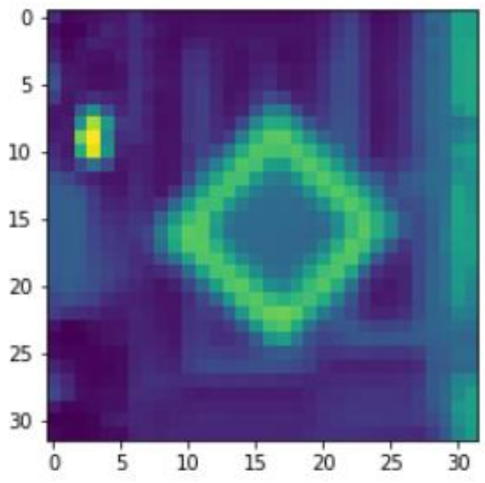
1. Describe how you preprocessed the image data. What techniques were chosen and why did you choose these techniques? Consider including images showing the output of each preprocessing technique. Pre-processing refers to techniques such as converting to grayscale, normalization, etc.

As a first step, I decided to convert the images to grayscale because to reduce the complexity.

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



Before Grayscaling



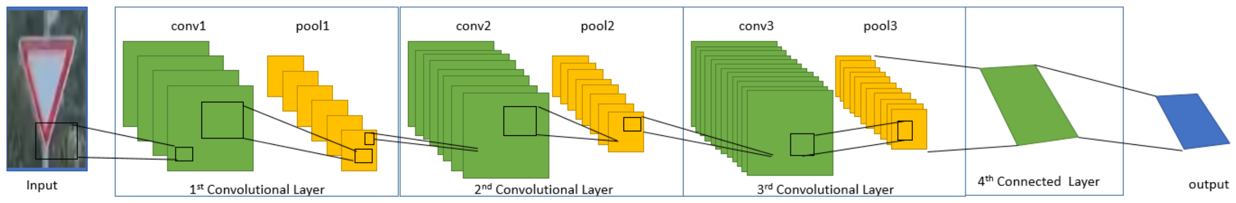
After Grayscaling

As a last step, I normalized the image data because this makes convergence faster while training the network.

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.

Started of with the LeNet 5 model architecture. Added one additional convolutional layer. Removed one fully connected layer

<u>My final model consisted of the following layers:</u>



<u>Final model</u>

1. Layer 1: Convolutional. Input = 32x32x1. Output = 28x28x6.
2. Followed by applying Rectified Linear Unit (relu) activation function.
3. Pooling. Input = 28x28x6. Output = 14x14x6.
4. Layer 2: Convolutional. Output = 10x10x16.
5. Followed by applying Rectified Linear Unit (relu) activation function
6. Pooling. Input = 10x10x16. Output = 5x5x16.
7. Layer 3: Convolutional. Output = 1x1x400.
8. Followed by applying Rectified Linear Unit (relu) activation function
9. Flatten the concatenate the conv2 and cov3 layers
10. Layer 4: Fully Connected. Input = 800. 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.

To train the model, I used an adam optimizer, with the following hyperparameters.

EPOCH = 45

Learning rate = 0.001

Batch size = 155

Mu = 0

Sigma = 0.1

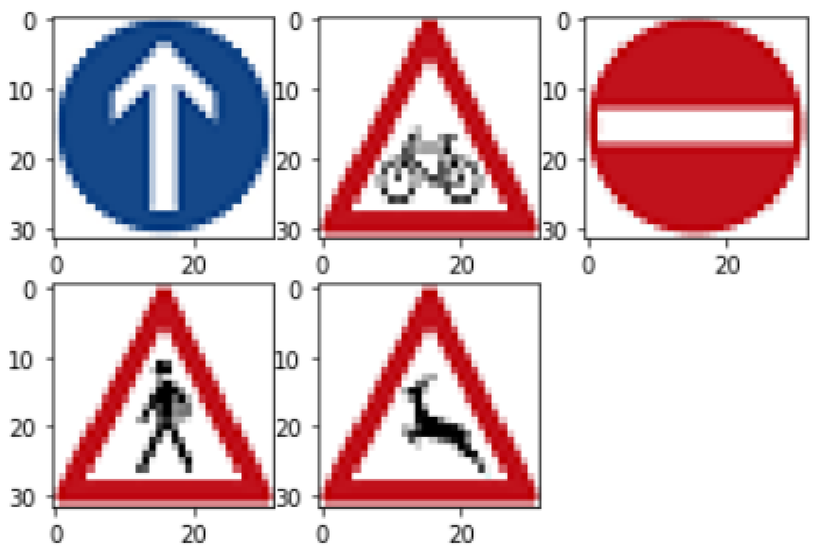
4. Describe the approach taken for finding a solution and getting the validation set accuracy to be at least 0.93. Include in the discussion the results on the training, validation and test sets and where in the code these were calculated. Your approach may have been an iterative process, in which case, 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.

LeNet-5 was chosen first as a default as it was taught in the course. However, the training set accuracy was less than 0.93. Adding convolution layer to make the provide higher accuracy. The batch size and epoch was adjusted based on trial and error method. The tweaking of the hyperparameters was based on how the training accuracy fares. The combination of these changes led to a higher training accuracy.

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:

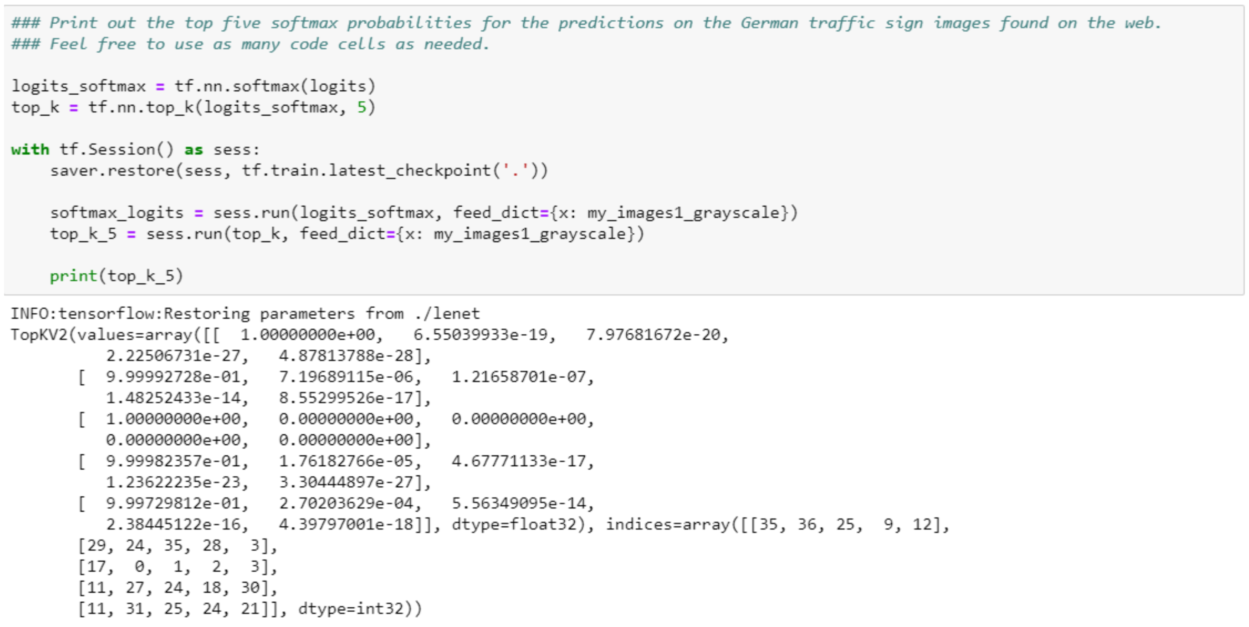


2. Discuss the model's predictions on these new traffic signs and compare the results to predicting on the test set. At a minimum, discuss what the predictions were, the accuracy on these new predictions, and compare the accuracy to the accuracy on the test set

The traffic classifier accuracy obtained was 0.60, 60%. In general, these images are perfect images (not real life images). The traffic sign classifier was trained on real life data, which explains the accuracy of classifying the traffic signs.

11. **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**

my_labels = [35,29,17,22,31]



```
### Print out the top five softmax probabilities for the predictions on the German traffic sign images found on the web.
### Feel free to use as many code cells as needed.

logits_softmax = tf.nn.softmax(logits)
top_k = tf.nn.top_k(logits_softmax, 5)

with tf.Session() as sess:
    saver.restore(sess, tf.train.latest_checkpoint('.'))

    softmax_logits = sess.run(logits_softmax, feed_dict={x: my_images1_grayscale})
    top_k_5 = sess.run(top_k, feed_dict={x: my_images1_grayscale})

    print(top_k_5)
```

```
INFO:tensorflow:Restoring parameters from ./lenet
TopKV2(values=array([[  1.00000000e+00,   6.55039933e-19,   7.97681672e-20,
          2.22506731e-27,   4.87813788e-28],
       [  9.99992728e-01,   7.19689115e-06,   1.21658701e-07,
          1.48252433e-14,   8.55299526e-17],
       [  1.00000000e+00,   0.00000000e+00,   0.00000000e+00,
          0.00000000e+00,   0.00000000e+00],
       [  9.99982357e-01,   1.76182766e-05,   4.67771133e-17,
          1.23622235e-23,   3.30444897e-27],
       [  9.99729812e-01,   2.70203629e-04,   5.56349095e-14,
          2.38445122e-16,   4.39797001e-18]], dtype=float32), indices=array([[35, 36, 25,  9, 12],
       [29, 24, 35, 28,  3],
       [17,  0,  1,  2,  3],
       [11, 27, 24, 18, 30],
       [11, 31, 25, 24, 21]], dtype=int32))
```