Traffic Sign Recognition

The goals / steps of this project are the following:

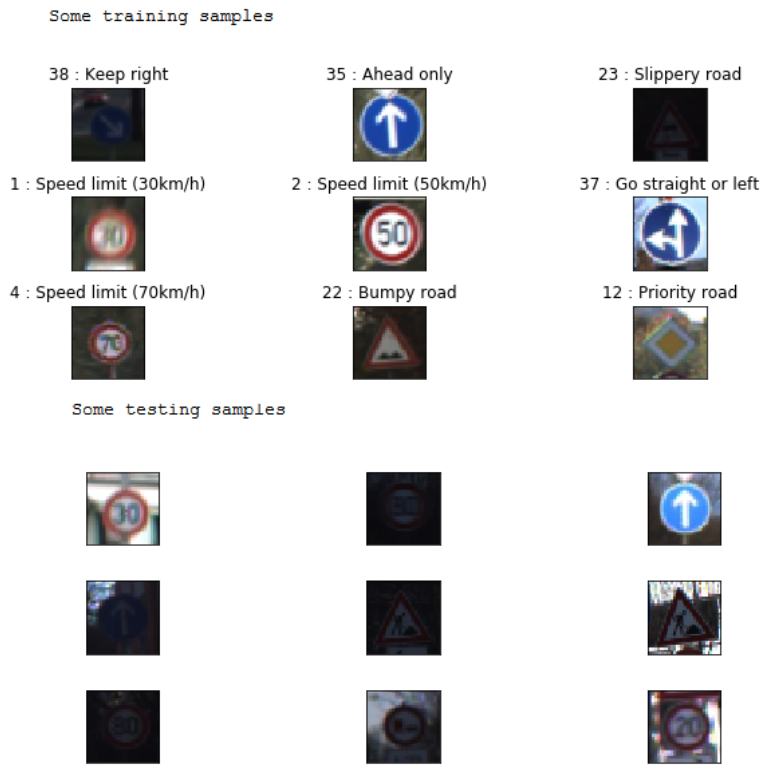
- Load the data set (see below for links to the project data set)
- Explore, summarize and visualize the data set
- Design, train and test a model architecture
- Use the model to make predictions on new images
- Analyze the softmax probabilities of the new images
- Summarize the results with a written report

1. Data Set Summary & Exploration

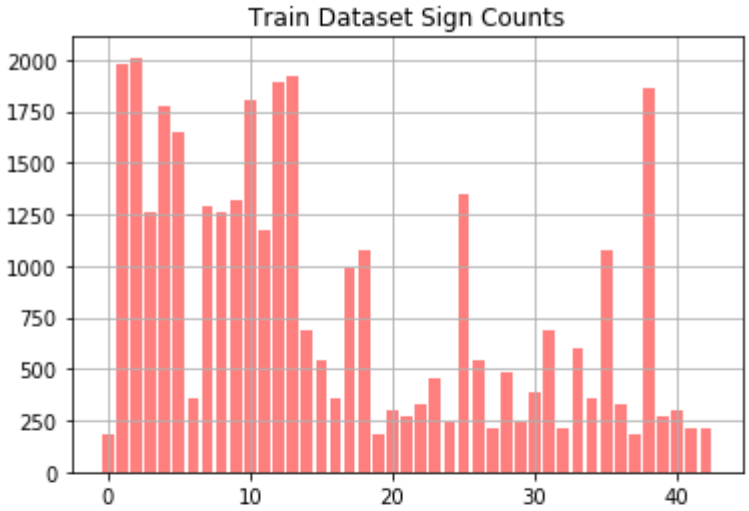
1. A basic summary of the data set is provided bellow.

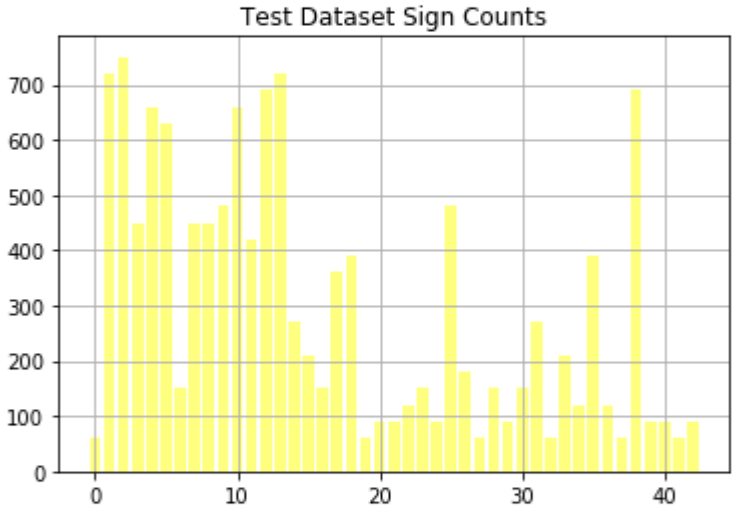
Number of training examples = 34799
Number of testing examples = 12630
Number of validation examples = 4410
Image data shape = (32, 32, 3)
Number of classes = 43

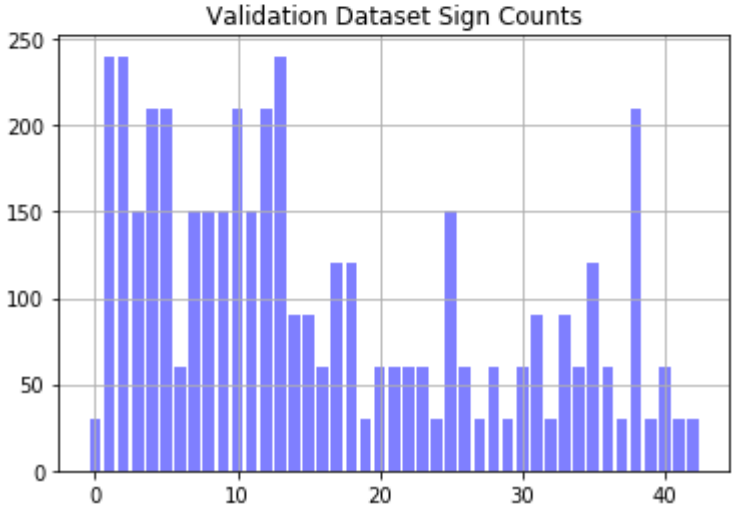
2. A screenshot with some images from the training and testing datasets are displayed below. I have also included the class for the training samples.



Here is an exploratory visualization of the data set. The bar chart is showing the data distribution from the training, testing and validation datasets. On the horizontal x axis we have the number of classes and on the y axis the number of samples per class.



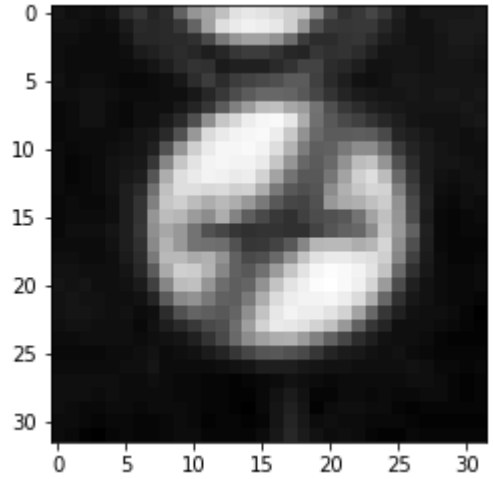




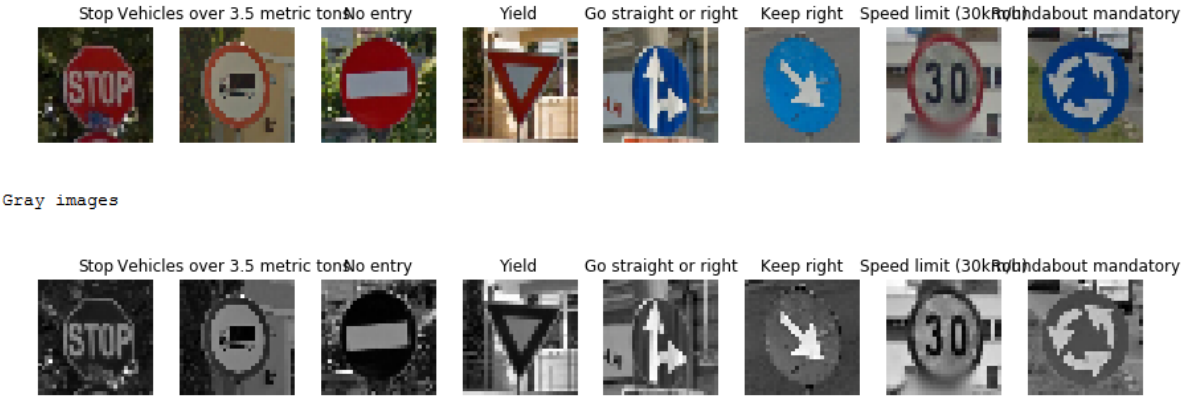
2. Design and Test a Model Architecture

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.

I have pre-processed the data by converting each image to grayscale. I have tried out other techniques such as normalization and conversion to another color space, however after training my neural network the results were worse, so I did not include them in the end. Below is an example of an image from the training set converted to gray.



In the image below we show the gray conversion of some images found on the web, which will be classified later.



Gray images

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.

The final neural network model used has the following layers:

Layer	Description
Input	32x32x1 Gray Image
Convolution 5x5	1x1 stride, valid padding, outputs 28x28x6
Softmax	
RELU	
Max Pooling	2x2 stride, outputs 14x14x6
Convolution 5x5	1x1 stride, valid padding
RELU	
Max Pooling	2x2 stride, outputs 5x5x16
Convolution 1x1	2x2 stride valid padding outputs 1x1x412
RELU	
Fully Connected	412 input, 122 output
RELU	
Fully Connected	122 input, 84 output
RELU	
Fully Connected	Input 84, 43 output

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 have used the example provided in the LeNet Lab solution. I have used the AdamOptimizer, which is similar to the stochastic gradient descent optimizer. I have trained my network for 100 epochs with a batch size of 100. I have experimented with numerous epochs and batch sizes, however the best results were obtained for the above mentioned values. The learning rate used is fixed and set to 0.0009. I have experienced with different parameters and obtained over 93% test accuracy even for the original LeNet architecture. For the proposed model the accuracy can span from 93,5% to 95%.

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.

My final model results were:

Train Accuracy = 1.000
Validation Accuracy = 0.960
Test Accuracy = 0.946

For my implementation an iterative approach was chosen. I have found a satisfying model after 22 iterations.

If an iterative approach was chosen:

- **What was the first architecture that was tried and why was it chosen?**

  The first architecture that I tried was the LeNet architecture. I have tried it because of its popularity, its simplicity and also due to the fact that it was presented in the course.

- **What were some problems with the initial architecture?**

  The original architecture with its original parameters, did not achieve an accuracy over 93%. Some of the problems regarding the initial architecture had to do with the original values of the learning rate and batch size, neuron number in layers, but also with the images fed to the network. At first, several things were tried on the initial architecture before modifying it. For example I tried to see how much the network results improved after grayscale, normalizing my data and increasing the number of neurons. Even though without major improvements the network offered a result of 93.5% in accuracy, the results were not very stable, meaning that only some times when I trained the network I obtained this result; usually the results were between 92% and 93.5%. I was not very satisfied and I tried to push the results even further.

- **How was the architecture adjusted and why was it adjusted?**

  The architecture was adjusted by adding more convolution layers. I have also increased the number of neurons from some layers. I have experimented with several modifications of the initial architecture, however I did not obtain a satisfying accuracy until I added more layers and included more neurons on some of the layers. The reason why my model performed poor at first was due to underfitting.

- **Which parameters were tuned? How were they adjusted and why?**

The parameters that were tuned were: epoch, learning rate, batch size and number of neurons for the layers. The values that are currently in my solution are:

Epoch = 100; learning_rate = 0.0009, batch_size = 100;

I have trained my network for a higher number of epochs so that I would get a higher accuracy. I have seen that if I increased the number of epochs the accuracy results would also increase. The batch size is the number of training samples your training will use in order to make one update to the model parameters. I have experienced with multiple batch sizes and the one that offered the best result was the batch size of 100 (I have tried 64, 128, 156, 256 etc). The learning rate was decreased from 0.001 to 0.0009. Even though the difference between the original and the proposed learning rate is small, the final accuracy using the proposed learning rate is bigger. Other modifications were made on the number of neurons from the network layers. By adding a few neurons on some of the network layers, the network was able to better learn the given data.

- **What are some of the important design choices and why were they chosen?**

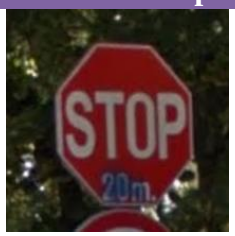
I think one of the most important things that increased the accuracy of my model was the introduction of more convolutions, which are able to capture more features. Even so I have tried a lot of changes including dropout and more convolutions before leaving this final version of architecture. From my point of view these design choices are more like an art, not necessary engineering.

3. 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.

For this task I have chosen 8 images and tested the implemented model.

Proposed Traffic signs

1. 

2. 

3. 

4. 

5. 

6. 

7. 

8. 

Some of the proposed images are easier to classify and some are more difficult.

The difficulty for classifying the first image comes from the fact that the stop sign is not well illuminated and the 20m on it might make the classifier confuse with another sign.

The second sign is well illuminated and seen a bit from the side. This sign is seen quite well however I wanted to see if the classifier can get it right.

The third sign has different illuminations on it which may make it harder for the classifier to identify it.

The fourth sign is very dirty and it also has some blur on it. This may confuse the network

The fifth sign contains shadows on the sign and the background is illuminated. The sign is also seen a bit from the side.

The sixth sign from my point of view does not contain any difficulty to classify. I included it to see if the network can classify it correct.

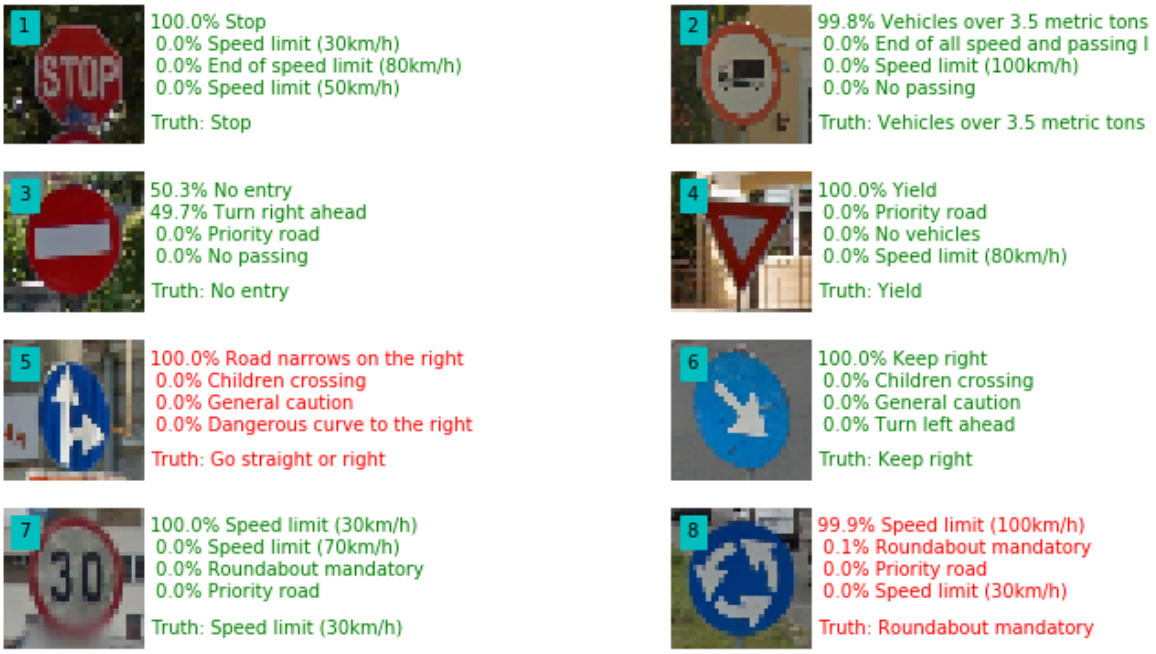
The seventh sign is scratched and dirty. This may cause some difficulties to the neural net.

The eight sign is can be seen very well, however the cars from the background and the grass might trick the neural network.

One difficulty the network would have is with images that are not included in the training set. However due to the fact that I wanted to evaluate the result on the types of images I have trained the classifier with I included only images that were in the training set.

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

A screenshot from the jupyter notebook is displayed bellow. This image clearly shows the results of the classification for each image.



The model was able to correctly classify 6 of the 8 images. The accuracy of the classification is 75%. Considering the overall accuracy and the number of proposed images the results are

quite satisfactory. Of course the result could be improved, by adding more data or by adding noise or other effects on the training images and adding them to the original training set.

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.

The result can be seen in the image above (from point 2).