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
- Summarize the results with a written report

Rubric Points

Here I will consider the rubric points individually and describe how I addressed each point in my implementation.

Writeup / README

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.

You're reading it! and here is a link to my project code

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.

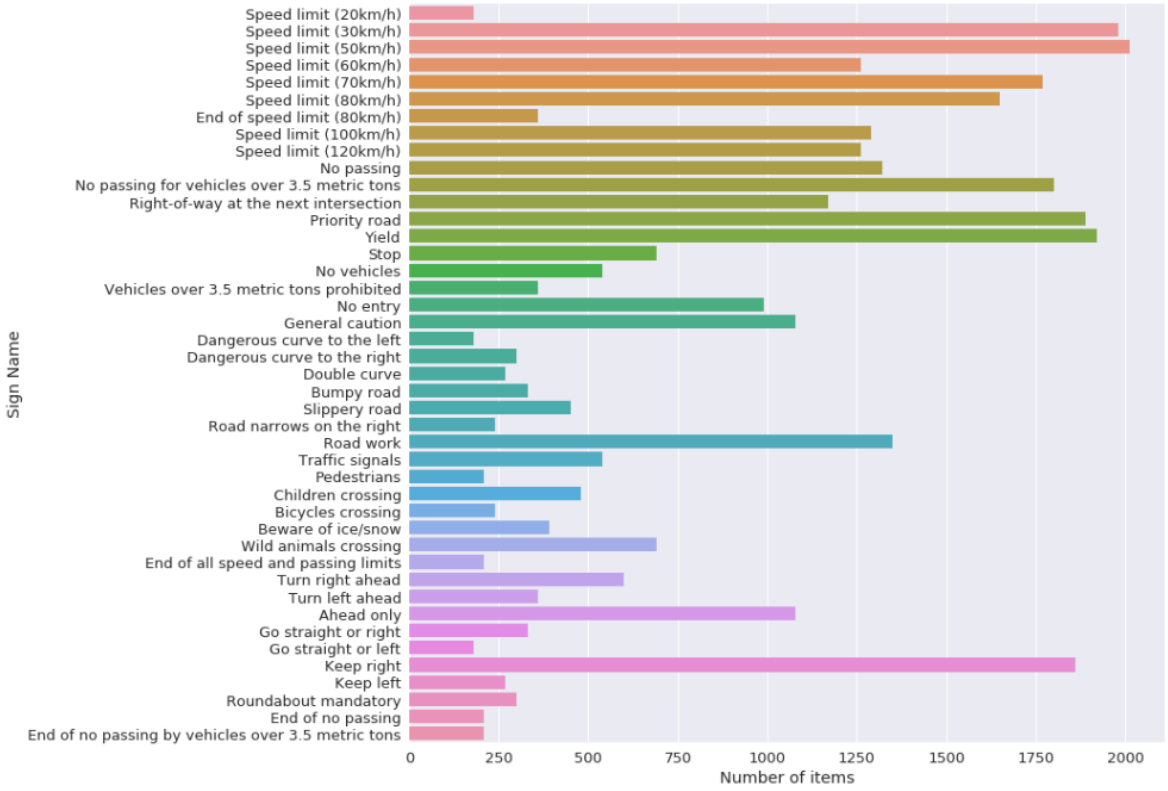
I used the pandas library to calculate summary statistics of the traffic signs data set:

- The size of training set is 34799 samples
- The size of the validation set is 4410 samples
- The size of test set is 12630 samples
- 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. A single image is randomly selected from each sign.

The below is a bar chart showing how many training samples are there for each sign.

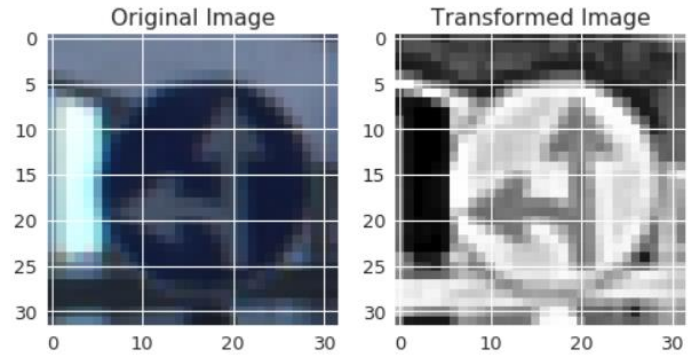


Design and Test a Model Architecture

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 sharpen the image and reduce image noise by using Gaussian blur and then I converted the images to grayscale (selecting the Y component of the YUV colorspace) and adjusted the images by histogram sketching. The reason of converting images to grayscale is that during training data, grayscale images accuracies outperforms color images.

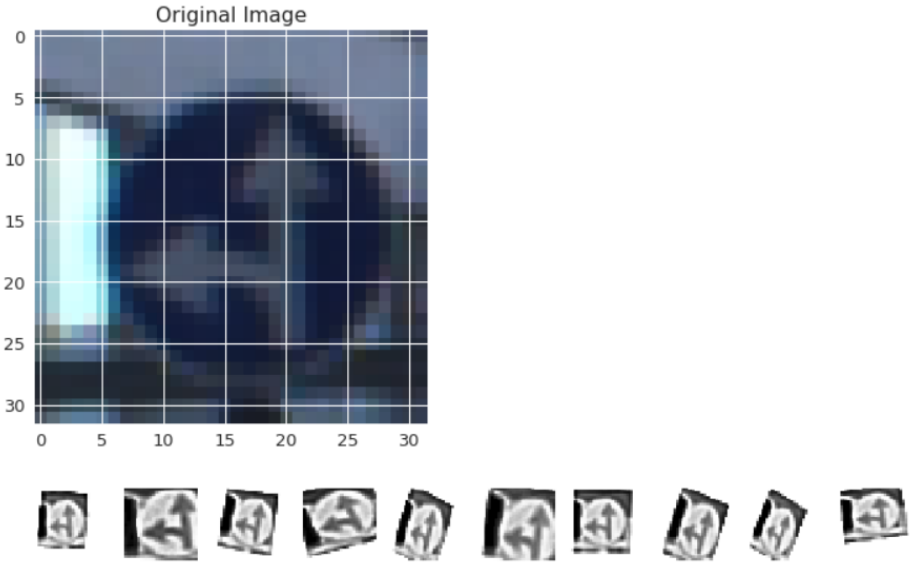
Here is an example of a traffic sign image before and after sharpening, noise reduction and grayscaling.



I decided to generate additional data because it increases the number of training samples and also simulate different camera angles (by rotating the images).

To add more data to the data set, I used affine transformations (including by rotations, translations, shearing). This was inspired by a Medium article Dealing with unbalanced data: Generating additional data by jittering the original image by Vivek Yadav.

Here is an example of an original image and ten augmented images. I generated 10 additional images for each original images using affine transformations techniques, sharpening, noise-reducing and grayscaling.



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 Y component only of YUV colorspace image
Convolution 1	1x1 stride, same padding, outputs 28x28x6
RELU	
Max pooling	2x2 stride, outputs 14x14x6.
Convolution 2	1x1 stride, VALID padding, output = 10x10x16
RELU	
Max pooling	2x2 stride, outputs 5x5x16.
Flatten	Input = 5x5x16. Output = 400
Fully connected	Input = 400. Output = 120
RELU	
Dropout	
Fully connected	Input = 120. Output = 84
RELU	
Dropout	
Full 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.

To train the model, I used an AdamOptimizer, batch size of 128, 100 epochs, learning rate of 0.001 and dropout probability of 0.3.

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:

- validation set accuracy of 93.9%.
- test set accuracy of 89.0%
- accuracy for predictions of images from web is 40%

The classic LeNet-5 Model was used as a starting point of the architecture (LeCun et al., 1998). It is used because it's well known and designed by handwritten and machine-printed character recognition, which share similarities with traffic signs.

And then dropout was introduced just before layer 4 and layer 5 of the LeNet model because dropout reduces model overfitting (Srivastava, Nitish, et al, 2014).

4.1 Model Training and hyperparameters tunings

Firstly, the classic LeNet 5-layer model was used, without any dropouts or images preprocessing. The parameters used are - Learning rate: 0.001, Epochs: 50. It achieved a validation accuracy of 92.8% and test accuracy of 90.9%.

Secondly, I pre-processed the images first before feeding the data the classic LeNet 5-layer model (still without dropouts). It achieved a validation accuracy of 94.3% and test accuracy of 90.3%. The accuracy of predictions for images from the web is 0%, which shows that the model may overfit.

In order to deal with the issue of overfitting, dropouts are introduced. Dropouts are introduced at layer 4 and layer 5 of the LeNet 5-layer model. It achieved a validation accuracy of 93.3% and test accuracy of 89.1%. The accuracy of predictions for images from the web however is 0%.

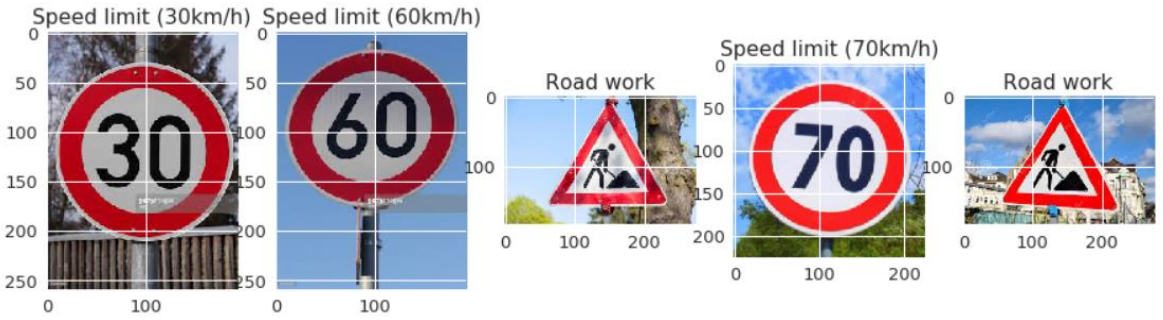
In order to increase prediction accuracies for images from the web, the number of epochs has been increased from 50 to 100. This achieved the final result in heading 4 above.

Other hyperparameters have also been tried, including learning rate of 0.0001 and 0.0005, epochs of 150, dropout probabilities of 0.1, 0.2, 0.4 and 0.5.

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:



The second image might be difficult to classify because the image (before resizing and preprocessing) is skewed towards the top part the image, while for the training data, the sign tends to be in the centre of the images.

The first, third and fifth images may be difficult to classify too as the edges of these images that does not contain the sign are somewhat large than those in training data. In other words, the sign is not centred on four sides in the image.

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).

Here are the results of the prediction:

Image	Prediction
30km speed limit	Road work
60km speed limit	Road work
Road work	Road work
70km speed limit	Road work
Road work limit	Road work

The model was able to correctly guess 2 of the 5 traffic signs, which gives an accuracy of 40%. And somewhat surprisingly the model predicts all signs to be “Road work”.

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. (OPTIONAL: as described in the "Stand Out Suggestions" part of the rubric, visualizations can also be provided such as bar charts)

The top 5 probabilities for each traffic sign are as follows.

Top 5 probabilities for sign Speed limit (30km/h)

Road work : 0.169202

Right-of-way at the next intersection : 0.104825

Beware of ice/snow : 0.0596334

Traffic signals : 0.0583703

Priority road : 0.0515658

Top 5 probabilities for sign Speed limit (60km/h)

Road work : 0.169202

Right-of-way at the next intersection : 0.104825

Beware of ice/snow : 0.0596334

Traffic signals : 0.0583703

Priority road : 0.0515658

Top 5 probabilities for sign Road work

Road work : 0.0918672

Ahead only : 0.0755033

Go straight or right : 0.0712688

Priority road : 0.0632367

Roundabout mandatory : 0.0628946

Top 5 probabilities for sign Speed limit (70km/h)

Road work : 0.169202

Right-of-way at the next intersection : 0.104825

Beware of ice/snow : 0.0596334

Traffic signals : 0.0583703

Priority road : 0.0515658

Top 5 probabilities for sign Road work

Road work : 0.169202

Right-of-way at the next intersection : 0.104825

Beware of ice/snow : 0.0596334

Traffic signals : 0.0583703

Priority road : 0.0515658