Self-Driving Car Engineer Nanodegree Program

Traffic Sign Recognition Project

François Masson
November 2nd, 2018

Pipeline Software

The project consists in being able to recognize traffic signs by using deep learning techniques. In order to fulfill this task, this guideline is applied :

- Loading the data set
- Data Set Summary & Exploration
- Design, train and test a model architecture
- Predictions on new images
- Softmax probabilities of the new images
- Features Map Analysis

Loading the dataset

Data for the project were already provided in a pickle format. They can be download on this site : http://benchmark.ini.rub.de/?section=gtsrb&subsection=dataset

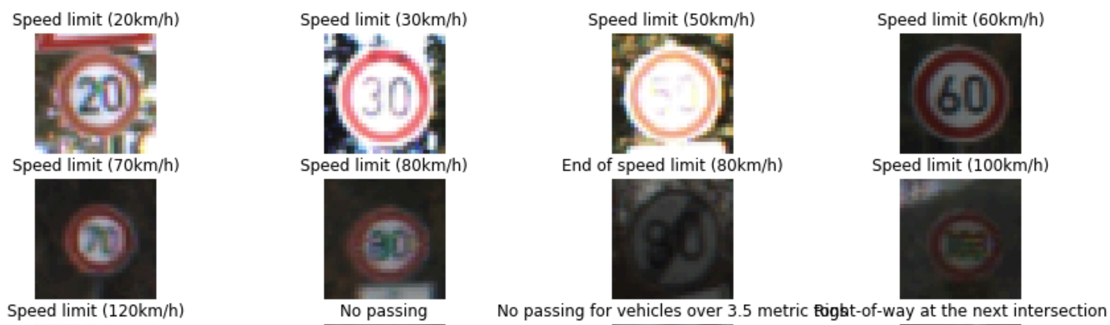
Data Set Summary & Exploration

- 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)
- The number of unique classes/labels in the data set is 43

The labels were provided in a csv format:

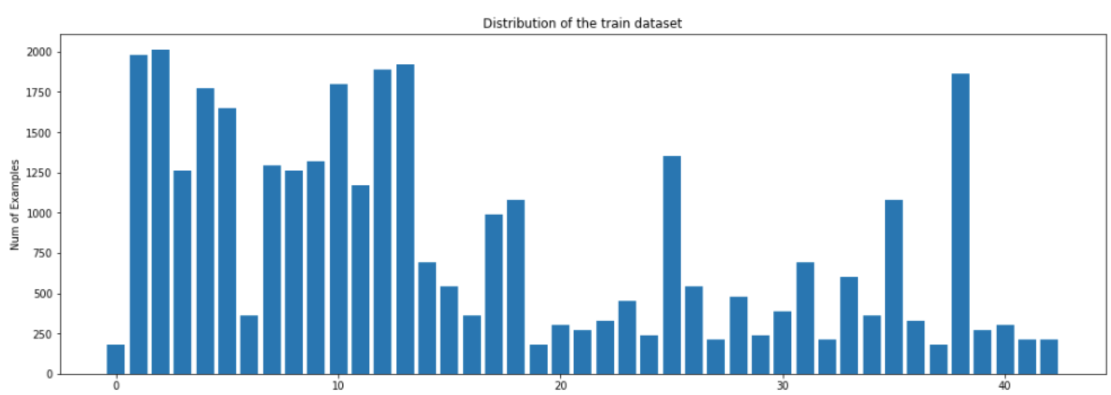
	ClassId	SignName
0	0	Speed limit (20km/h)
1	1	Speed limit (30km/h)
2	2	Speed limit (50km/h)
3	3	Speed limit (60km/h)
4	4	Speed limit (70km/h)
5	5	Speed limit (80km/h)
6	6	End of speed limit (80km/h)
7	7	Speed limit (100km/h)
8	8	Speed limit (120km/h)
9	9	No passing
10	10	No passing for vehicles over 3.5 metric tons
11	11	Right-of-way at the next intersection

As a first observation, we can see that there are 43 classes related to the dataset. A Matplotlib vizualisation of the labeled data looks like that :



An histogram of the number of examples shows that there is a certain disparity between each class. Moreover, the number of examples in certain classes is very low. A data augmentation will therefore be mandatory.

```
Min number of images per class = 180
Max number of images per class = 2010
```

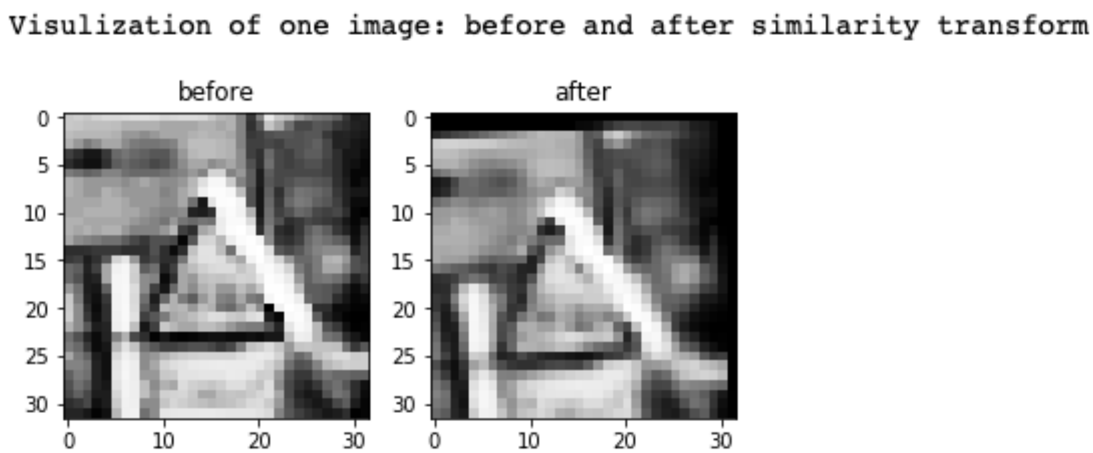


Design, train and test a model architecture

In a first step, the data were pre-processing by using three different techniques:

- Grayscale transformation
- Data augmentation
- Data normalization

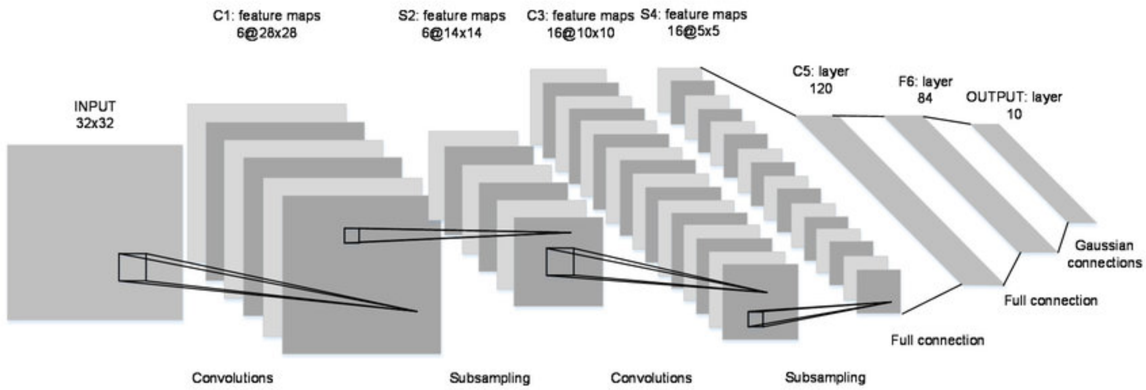
The grayscale transformation reduced the number of parameters to be trained. The data augmentation used some random translation around x and y such as a certain rotation of the images. The normalization makes training less sensitive to the scale of features.



As a comparison, the number of data significantly increased:

	Before data augmentation	After data augmentation
Training set size	34799	167035
Validation set size	4410	46169
Test size	12630	12630

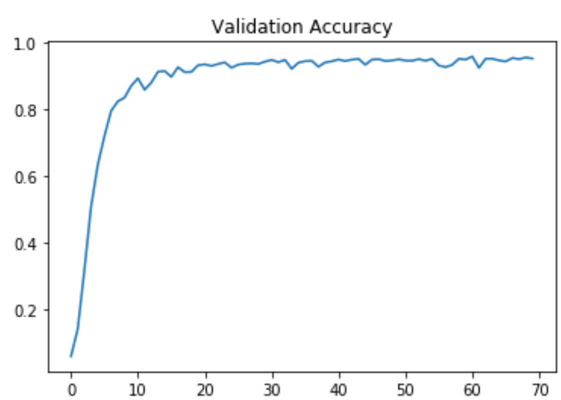
The model used to train the Convolutionnal Neural Network is similar to the original Lenet architecture:



The only modification used in this project results in the presence of some dropouts parameters during the training process. Definitely, increasing the number of dataset gave better accuracy results but lead to a higher processing time.

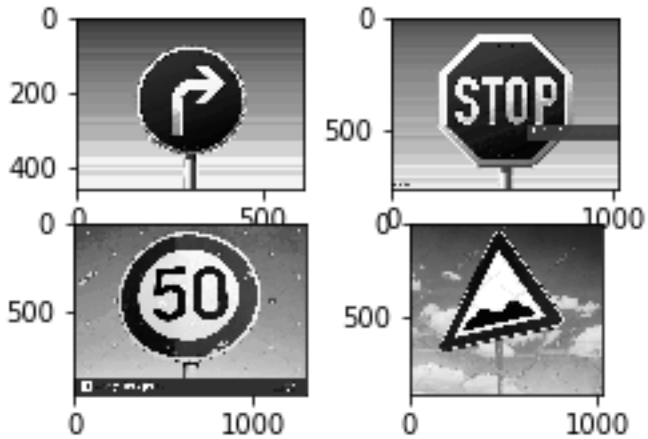
The accuracy obtained are:

- Validation accuracy: 0.952
- Test accuracy: 0.924



Predictions on new images

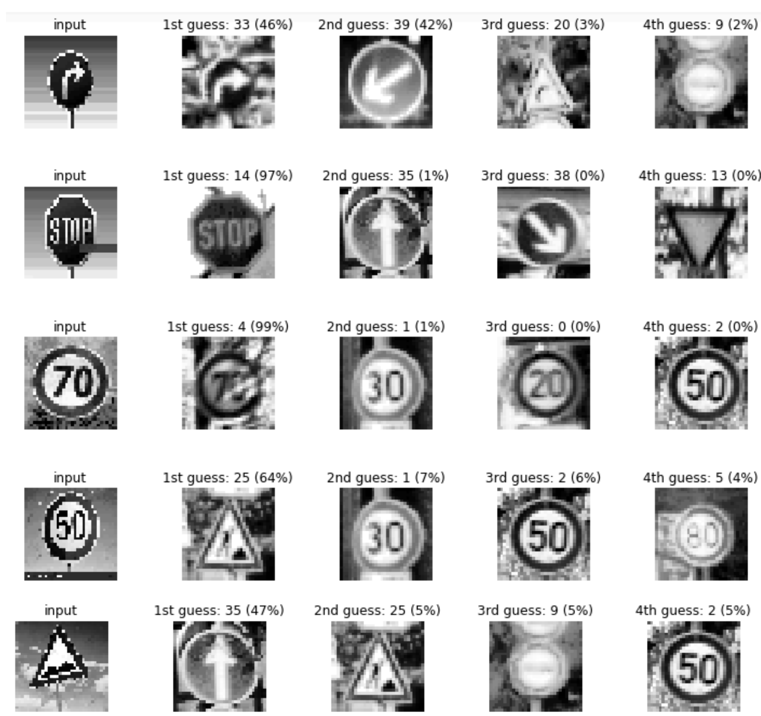
Here are five German traffic signs that I found on the web:



The last image might be difficult to classify because the pattern is not very complex and the quality of the image can provide misdirection for the algorithm resolution.

The results obtained was this dataset is only 0.6 which mean that only 3 on the 5 images were correctly classified.

Softmax Probabilities on new images



As expected, the bumby road was one of the missclassified image. Strangly, the 50 km limitation was not well correlated.

Features Map Analysis

It is interesting to see for example that the network characterized by himself the shape of sign: