Marco Intermite
Self Driving Car
Term 1 - Project 2
18 March, 2017

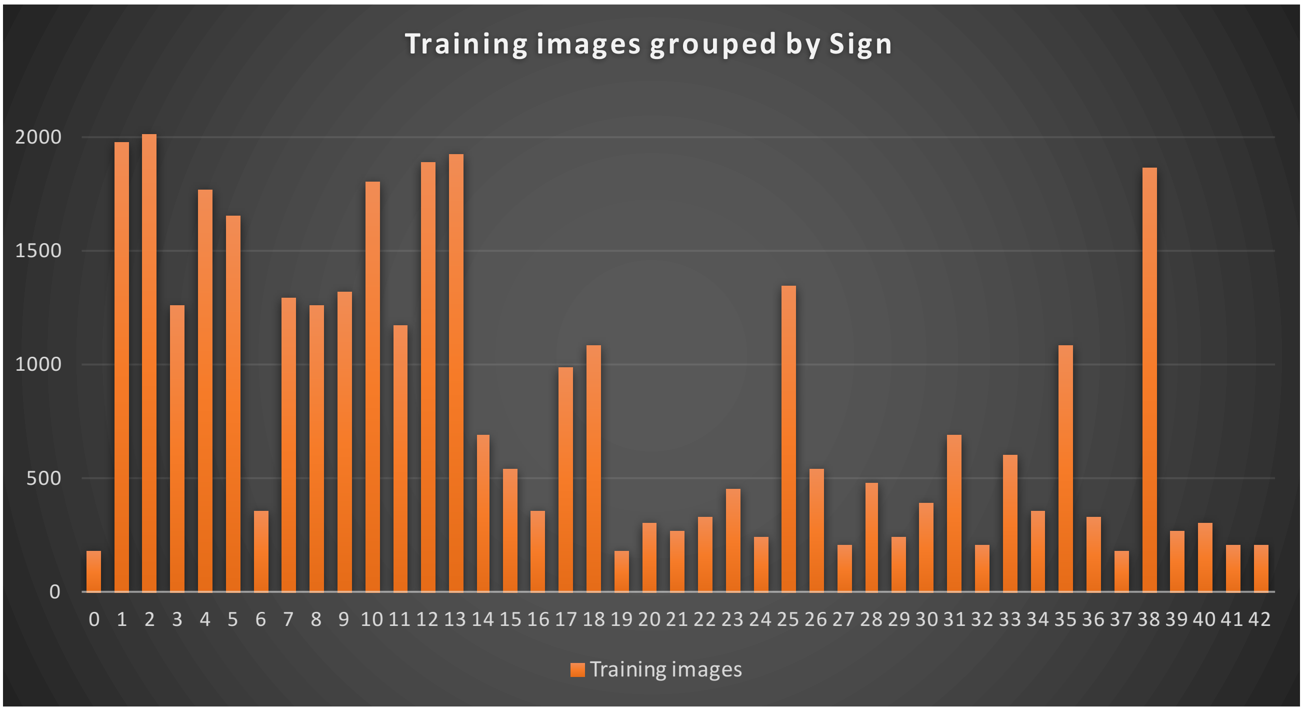
Traffic Sign Recognition

Data Set Summary & Exploration

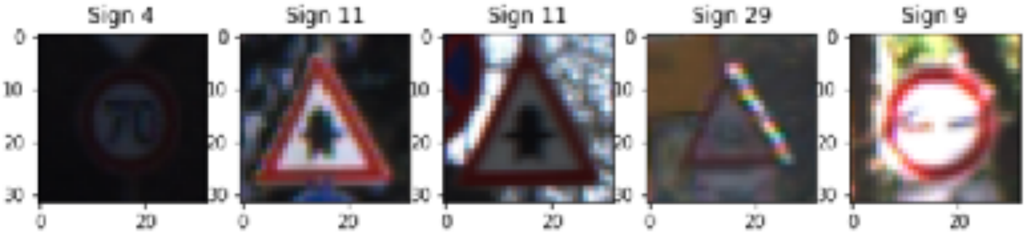
To provide a basic summary of data set I've used numpy, key numbers below:

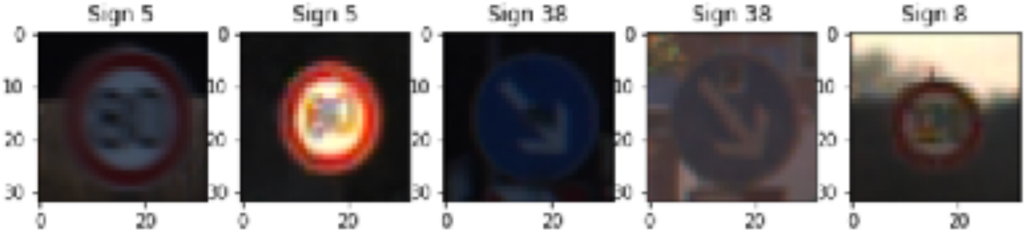
- Number of training examples = 34799
- Number of validation examples = 4410
- Number of testing examples = 12630
- Image data shape = (32, 32, 3)
- Number of unique signs (label) = 43

Interesting the training set numbers grouped by sign (label):



Here a visualization of the initial training set images:

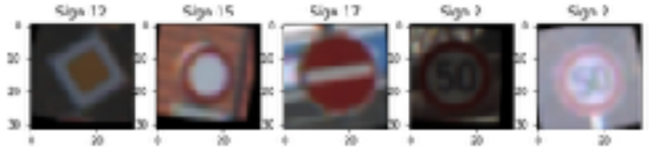


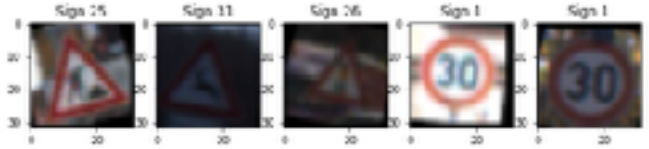


Pre-process the Data Set

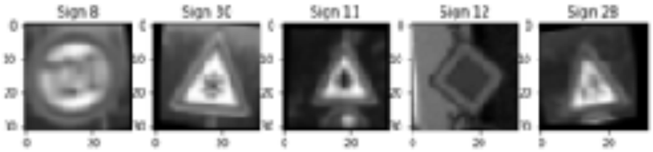
In order to better utilize the training set and improve the accuracy I've augmented the training set: I built a jittered dataset by adding 5 transformed versions of the original training set, yielding 173,995 samples in total. Samples are randomly perturbed:

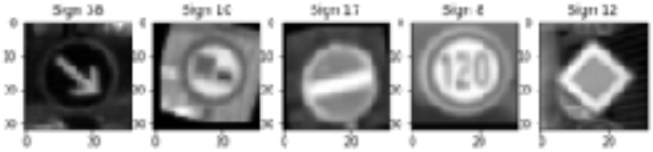
- in position ([-2,2] pixels)
- in scale ([.9,1.1] ratio)
- in rotation([-15,+15] degrees)





Further I've grayscaled and normalized the training set, validation and test set. Training set has been also shuffled.





All the images in the sets they have now shape (32,32,1)

Model Architecture

To train the machine learning I've used and modified LeNet CNN model and applying:

dropout technique to improve the generalization with the following composition

Learning rate dynamically decreasing.

LAYER	SUB LAYER	DESCRIPTION
INPUT	Input	32x32x1 Grayscale Image
LAYER 1	Convolution - filter 5x5	1x1 Stride, Valid padding, output 28x28x6
	Activation	RELU
	Pooling	2x2 Stride, Output 14x14x6
LAYER 2	Convolution - filter 5x5	1x1 Stride, Valid padding, output 10x10x32
	Activation	RELU
	Pooling	2x2 Stride, Output 5x5x32
FLATTEN	Flatten	Output = 5x5x32 = 800
LAYER 3	Fully Connected	Output = 240
	Activation	RELU
	Dropout	Keep_prob = 0,5 const
LAYER 4	Fully Connected	Output = 168
	Activation	RELU
	Dropout	Keep_prob = 0,5 const
LAYER 5	Fully Connected	Output = 43
	Softmax	
OPTIMIZER	Adam Optimizer	Learning rate decreasing dinamically (0,001 - 0,0001)

Model Training and Solution Approach

I’ve trained the machine using the following hyper parameter:

Batch size: 128

EPOCHS: 1000

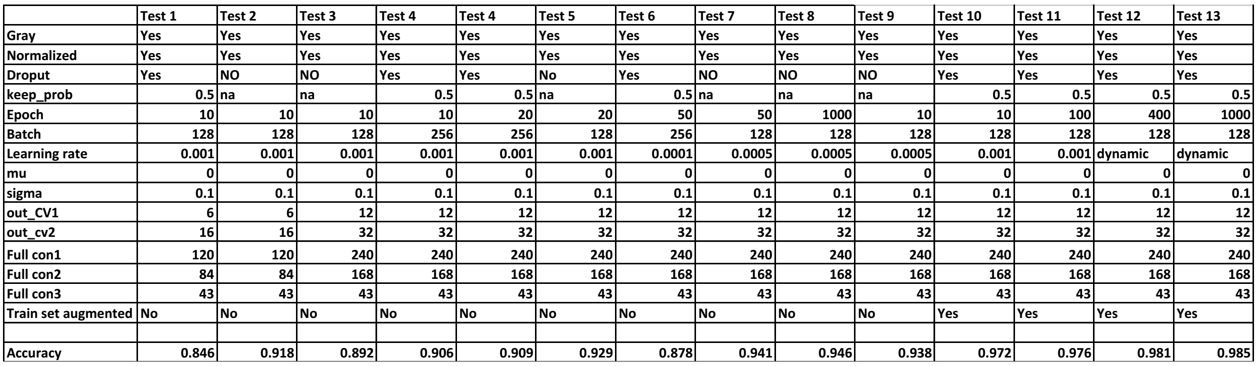
Mu = 0

Sigma = 0.1

Learning rate initial = 0.001

Learning rate final = 0.0001

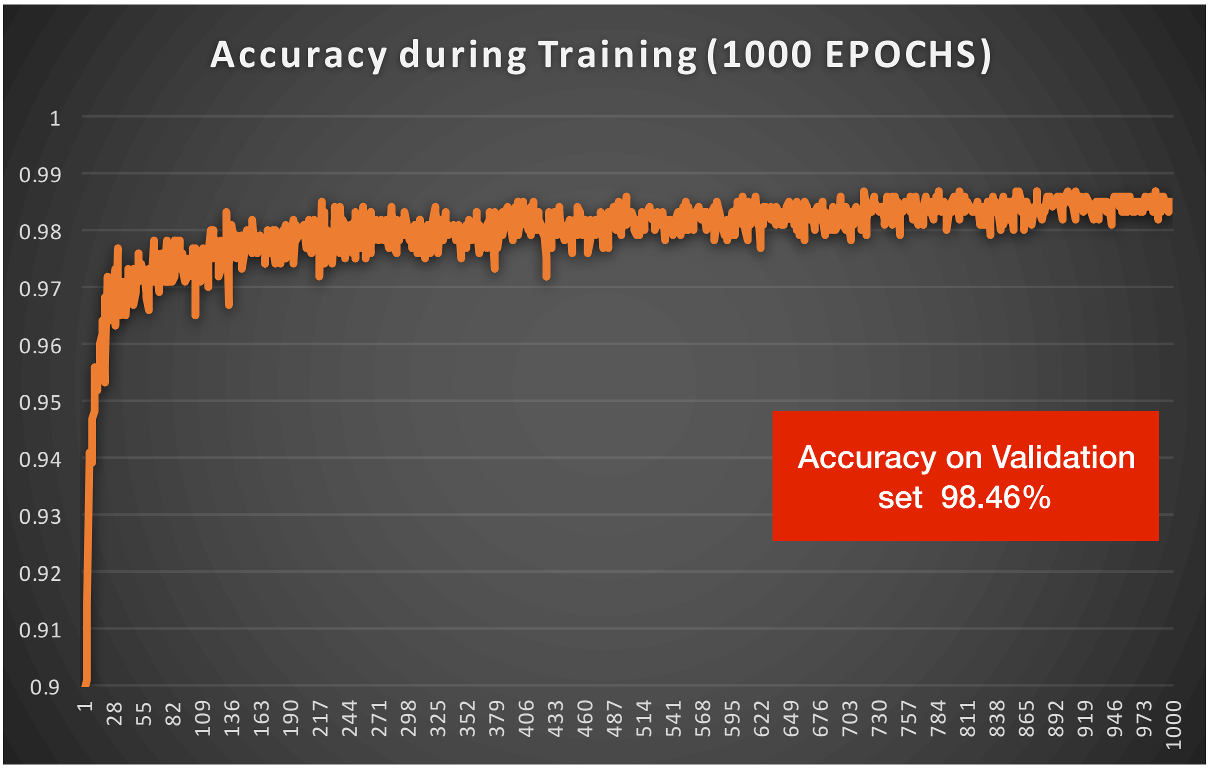
To choose the parameters I’ve trained the machine with different parameters and analyzed the results. By following the iteration I’ve done to decide the above parameters:



	Test 1	Test 2	Test 3	Test 4	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Test 11	Test 12	Test 13
Gray	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Normalized	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Droput	Yes	NO	NO	Yes	Yes	No	Yes	NO	NO	NO	Yes	Yes	Yes	Yes
keep_prob	0.5	na	na	0.5	0.5	na	0.5	na	na	na	0.5	0.5	0.5	0.5
Epoch	10	10	10	10	20	20	50	50	1000	10	10	100	400	1000
Batch	128	128	128	256	256	128	256	128	128	128	128	128	128	128
Learning rate	0.001	0.001	0.001	0.001	0.001	0.001	0.0001	0.0005	0.0005	0.0005	0.001	0.001	dynamic	dynamic
mu	0	0	0	0	0	0	0	0	0	0	0	0	0	0
sigma	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
out_CV1	6	6	12	12	12	12	12	12	12	12	12	12	12	12
out_cv2	16	16	32	32	32	32	32	32	32	32	32	32	32	32
Full con1	120	120	240	240	240	240	240	240	240	240	240	240	240	240
Full con2	84	84	168	168	168	168	168	168	168	168	168	168	168	168
Full con3	43	43	43	43	43	43	43	43	43	43	43	43	43	43
Train set augmented	No	No	No	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes
Accuracy	0.846	0.918	0.892	0.906	0.909	0.929	0.878	0.941	0.946	0.938	0.972	0.976	0.981	0.985

Final Results

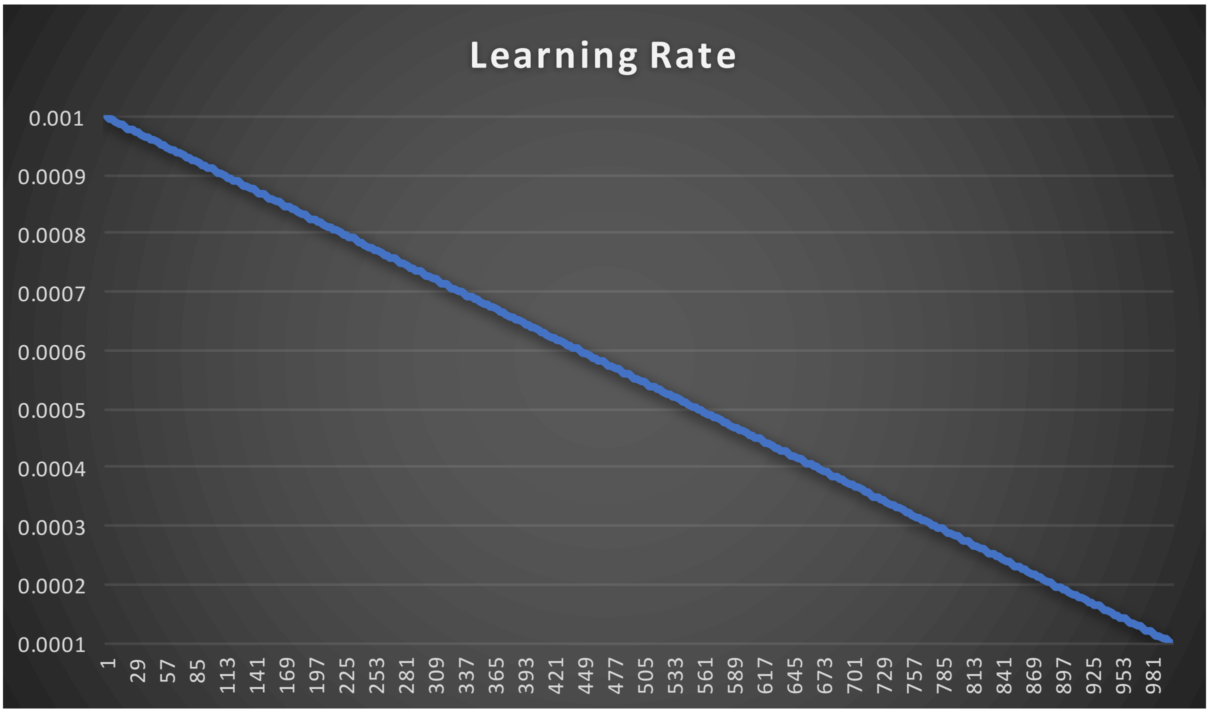
Below there is visualized the accuracy during training with 1000 EPOCHS:



Accuracy on Validation set: 98.46% (average of the last 50 iterations)

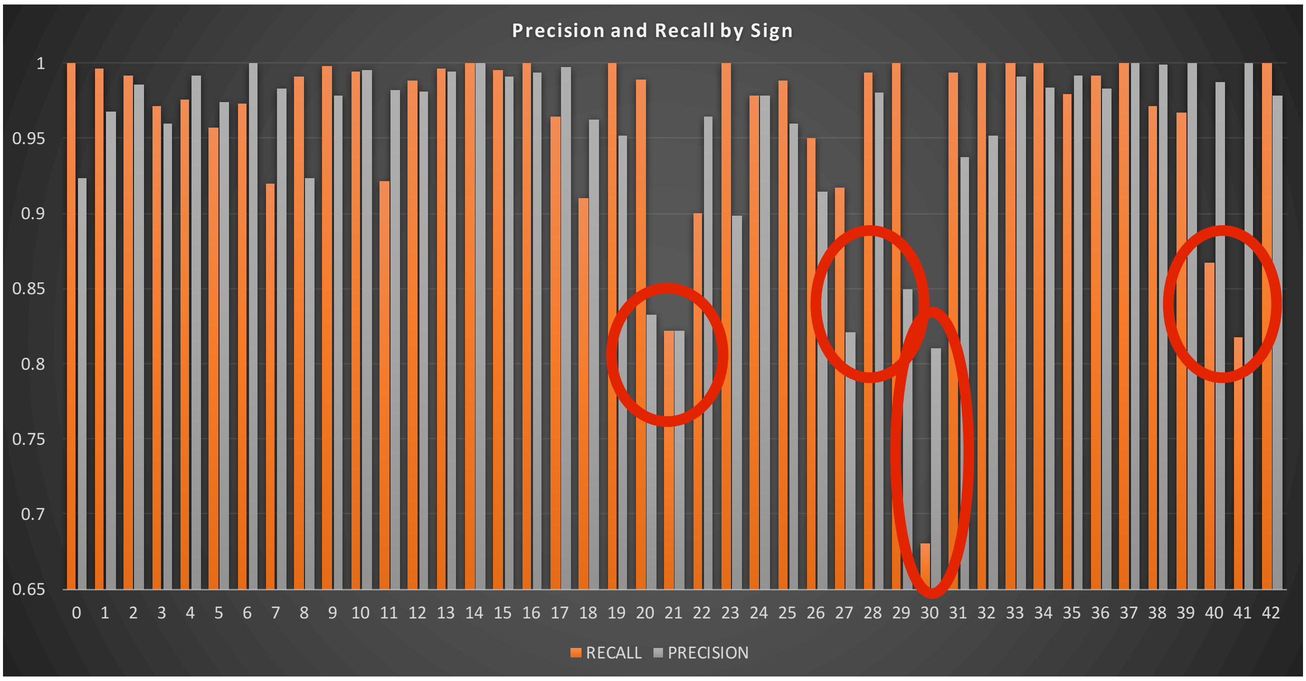
Accuracy on Test Set: 97.1%

By following the representation of the learning rate during the training:



Precision and Recall on Test Data

By following the precision and recall calculated for each sign:



To calculate the precision and recall I've used the confusion_matrix from sklearn library.

As main consideration the signs 21 and 30 are the signs where the model is more uncertain. Also in sign 40 and 41 the Recall is low and 29 and 27 they have a low Precision.

All these four signs are coming with low volume of training data (bar chart Training Images grouped by Sign above). Augment the training set on these signs would be an effective action to improve the accuracy.

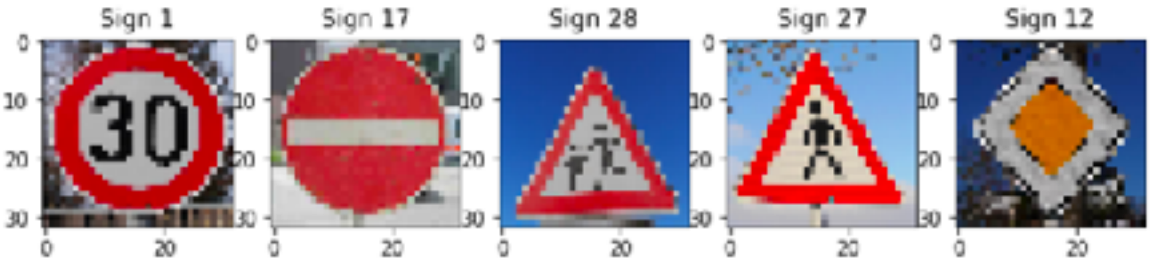
Test Model On New Images

Acquiring New Images

I’ve collected the following 5 images on Internet:

I’ve loaded this images, scaled them and added them into a np array with shape (32,32,3). I’ve also created a one_hot_y array with their label to verify the accuracy.

By following the images loaded are visualized along with their label.



I’ve then preprocessed the images (grayscale and normalize) in order to be ready to be recognized.

Performance on New Images

I’ve run the model against these 5 new images with the following result:

Accuracy on new test images: 100%

Each image has been correctly recognized. The accuracy reached is greater then the accuracy reached on the test set, this is justified for the fact that:

- the precision and recall for these images is very high
- these new images are well defined and very clear.

Model Certainty - Softmax Probabilities

The model is pretty much certain in predicting the 5 images found on the web. By following the top 5 Softmax probabilities:

```
Image 1:
Labels: [ 1,  2,  4,  5,  0]
Softmax: [1.00000000e+00,6.95114805e-24,  2.09909004e-24,
2.66068069e-27,9.03008215e-30]

Image 2:
Labels:[17,  0,  1,  2,  3]
[1.00000000e+00,0.00000000e+00,0.00000000e+00,
0.00000000e+00,0.00000000e+00]

Image 3:
Labels:[28, 27, 20, 18, 29]
Softmax: [1.00000000e+00,7.42377580e-16,2.50818972e-19,
3.46577368e-21,2.76854726e-21]

Image 4:
Labels:[27, 18, 24, 11, 28]
Softmax:[1.00000000e+00,6.91652356e-13,1.25352907e-24,
1.64956054e-27,3.92520523e-33]
```

```
Image 5:
Labels:[12,  0,  1,  2,  3]
Softmax:[1.00000000e+00,0.00000000e+00,0.00000000e+00,
0.00000000e+00,0.00000000e+00]
```

As main consideration in all the images the model was pretty defined in predicting the right label.

Considering the precision and recall, as expected in the image 4 the second probability is low but higher than the remaining ones, let’s visualize the images:

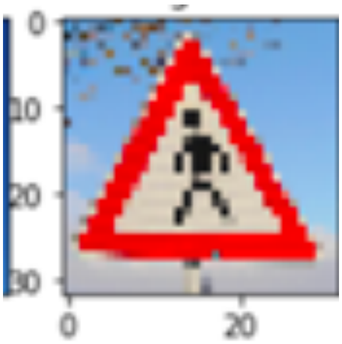
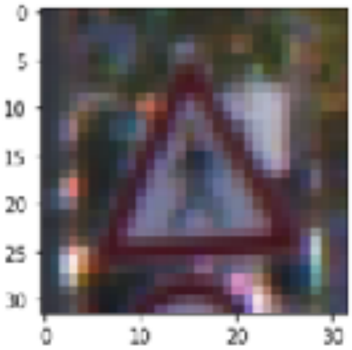
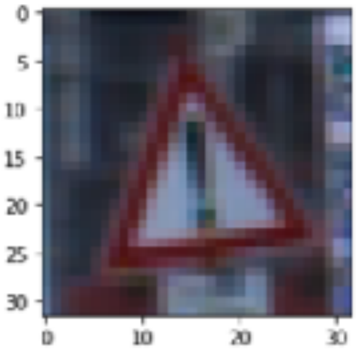


Image 4



Sign 27: ranked#1



Sign 18: ranked#2

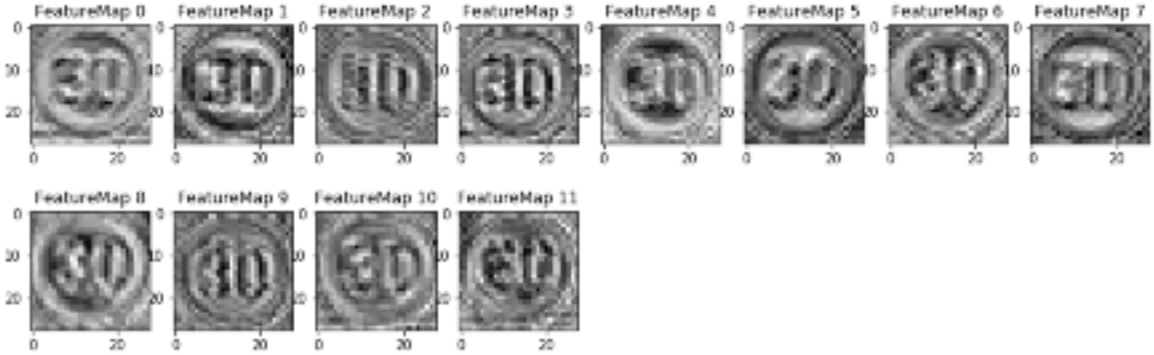
Indeed there is similarity between the 2 signs but the model is predicting quite well the right sign.

VISUALIZE LAYERS OF THE NEURAL NETWORK

The following is the visualization of the first layer of convolution, when stimulated with this image:

Input Image

As expect the first layer will output features with dimension (28,28) and output of 12, as visualized below:



Analyzing the features the model it seems is considering mainly the lines and curves of the image. The signal images are pretty basic images and the model is well recognized the image from the very first convolution, just looking at simple lines.