Traffic Sign Recognition

Writeup

The steps of this project are the following:

- Load the 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

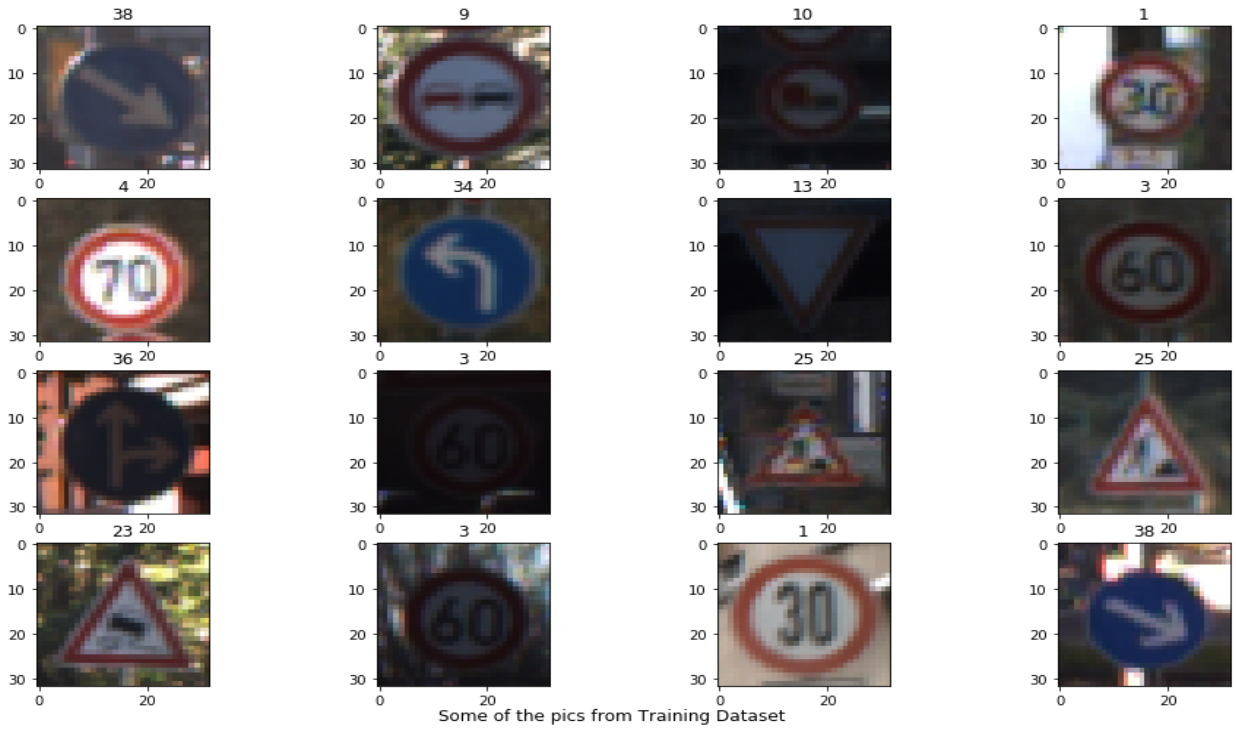
Rubric Points

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

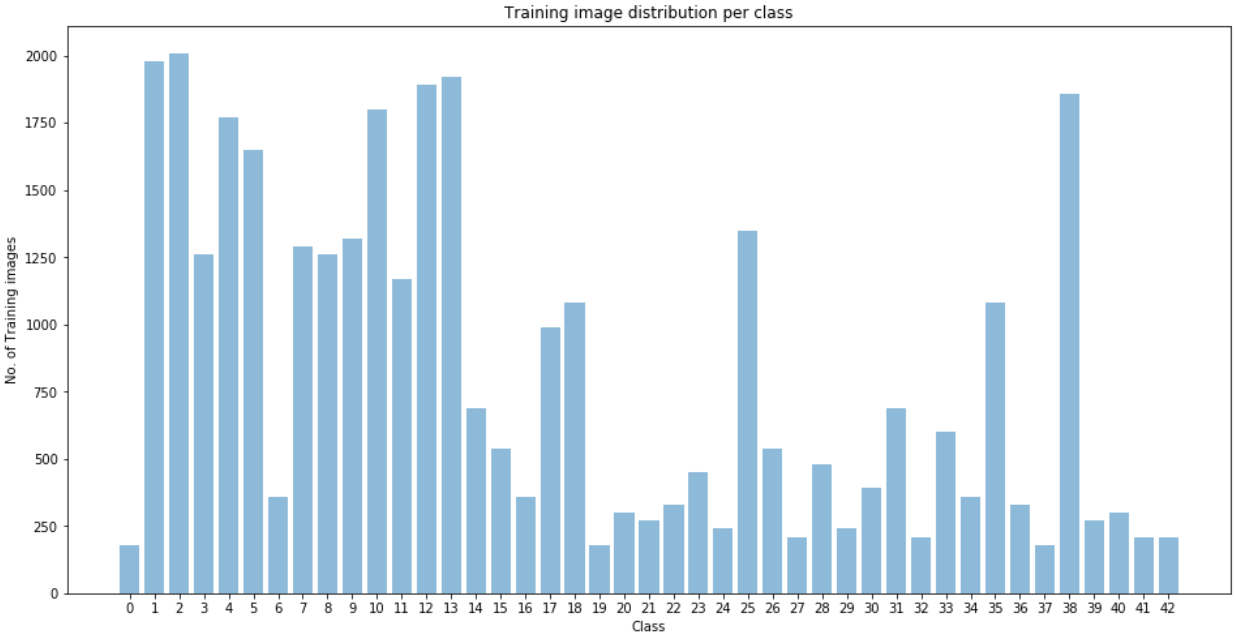
1. Data Set Summary & Exploration:

I used the in-built python functions 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.



Some of the pics from Training Dataset

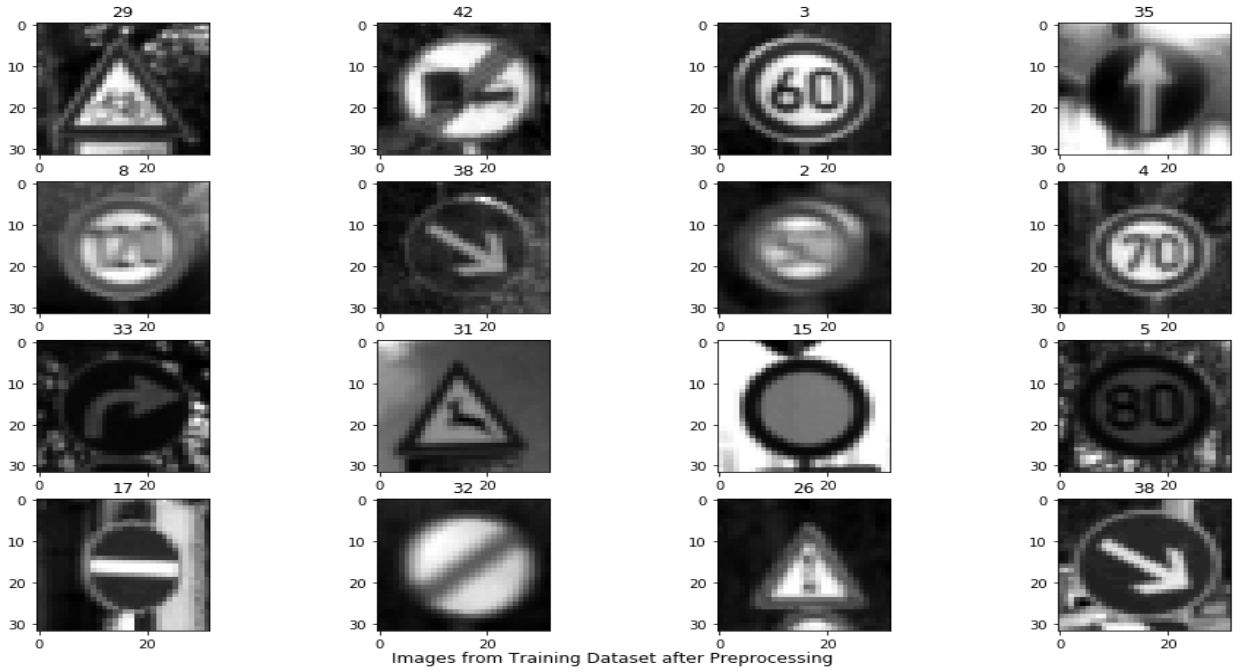


Here is an exploratory visualization of the data set. Above is a grid of few random images from the training dataset, and below that there is a bar chart showing how the training image data is distributed per class.

2. Design and Test a Model Architecture

As a first step, I decided to convert the images to grayscale and normalize the image data between (0.1,0.9) because the color was not a strong parameter for detecting the road signs and it is much more easy to normalize the grayscale images than RGB. I performed min/max normalization on all the images for optimization and it increased the accuracy significantly.

Here are the few examples of the traffic sign images after grayscaling and normalization.



Images from Training Dataset after Preprocessing

My final model consisted of the following layers:

Layer	Input	Description	Output
Preprocessing	32x32x3 RGB Image	Grascaling and Normalization done all the images	32x32x1 Normalized Grayscaled Image
Convolution	32x32x1 Normalized Grayscaled Image	Convolution filter of shape (5, 5, 32) with a stride of [1,1,1,1]	28x28x32
ReLU	28x28x32	Activation Function	28x28x32
Max Pooling	28x28x32	Max Pooling with k-size=[1,2,2,1]	14x14x32

Convolution	14x14x32	Convolution filter of shape (5, 5, 64) with a stride of [1,1,1,1]	10x10x64
ReLU	10x10x64	Activation Function	10x10x64
Max Pooling	10x10x64	Max Pooling with k-size=[1,2,2,1]	5x5x64
Flatten	5x5x64	Flattening to 1-D for transferring data to fully connected layers	1600
Dropout	1600	Dropout with keep_prob=0.5	1600
Fully Connected Layer	1600	Fully Connected layer with output window size=400	400
ReLU	400	Activation Function	400
Dropout	400	Dropout with keep_prob=0.5	400
Fully Connected Layer	400	Fully Connected layer with output window size=160	160
ReLU	160	Activation Function	160
Dropout	160	Dropout with keep_prob=0.5	160
Fully Connected Layer	160	Fully Connected layer with output window size=80	80
ReLU	80	Activation Function	80
Fully Connected Layer	80	Fully Connected layer with output window size=43	43

To train the model, I used:

- Type of Optimizer :- AdamOptimizer
- Batch Size :- 128
- Number of Epochs :- 20
- Learning Rate :- 0.001
- Dropout Probability :- 0.5
- Mu (Mean of truncated normal function) :- 0
- Sigma (Std. Deviation of truncated normal function) :- 0.1

My final model results were:

- training set accuracy of 0.999.
- validation set accuracy of 0.977.
- test set accuracy of 0.964.

If a well-known architecture was chosen:

- What architecture was chosen?
  LeNet-5 architecture was chosen, but I modified a little bit. I added dropouts in between fully connected layers, and added another fully connected layer and ReLU before the output.

- Why did you believe it would be relevant to the traffic sign application?
  Because it was based on image recognition for which LeNet-5 architecture is very popular.

- How does the final model's accuracy on the training, validation and test set provide evidence that the model is working well?
  I checked on 7 new images from internet and it worked well.

3. Test a Model on New Images

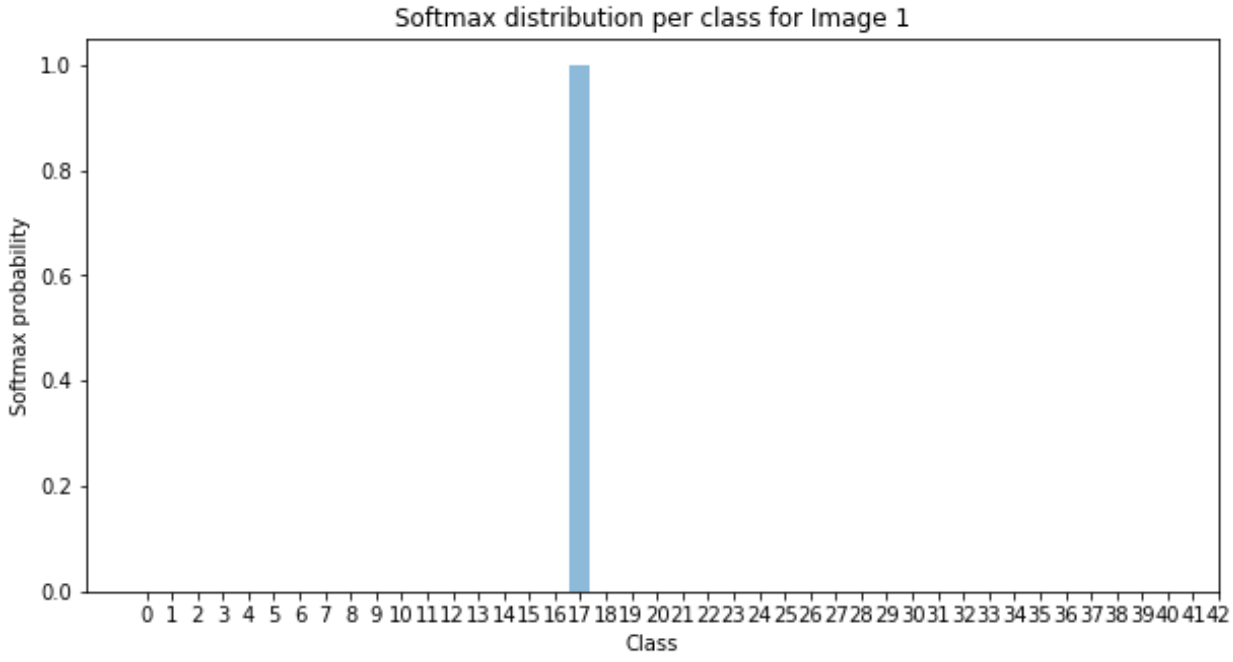
Here are the seven German traffic signs that I found on the web:

Here are the results of the prediction:

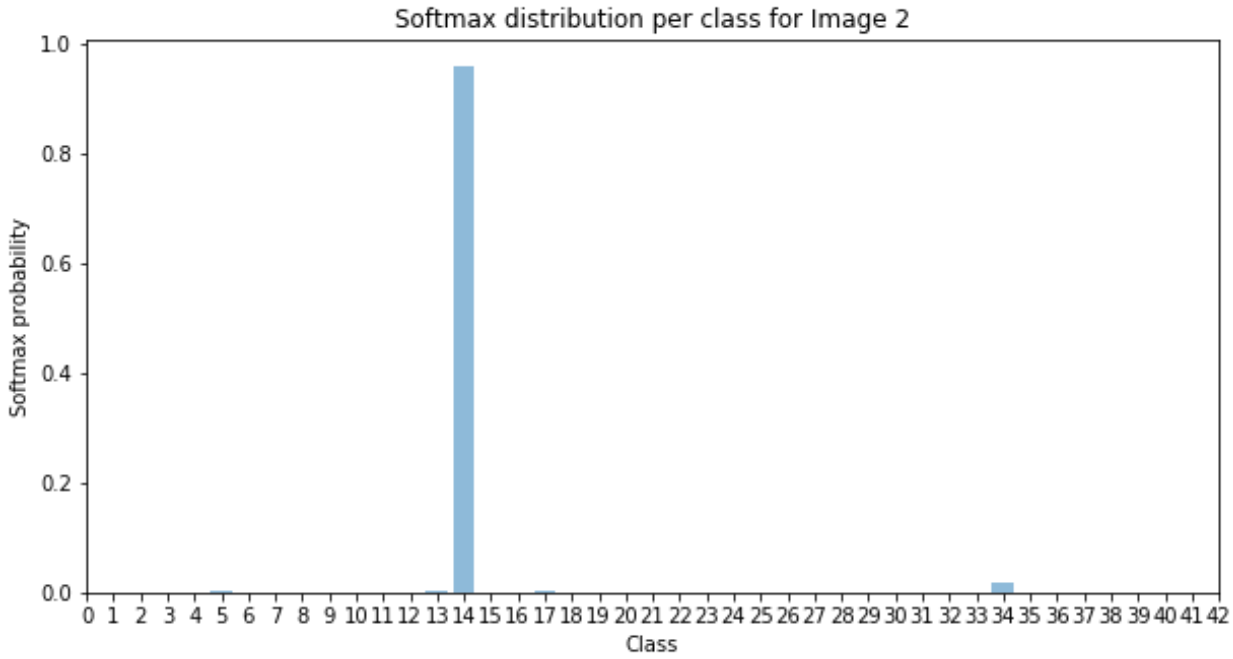
Image	Prediction
No entry	No entry
Stop	Stop
Slippery road	Beware of ice/snow
Speed limit (70km/h)	Speed limit (70km/h)
Wild animals crossing	Wild animals crossing
Yield	Yield
Yield	Yield

The model was able to correctly guess 6 of the 7 traffic signs, which gives an accuracy of 85.7%. The one that was wrong was the trickiest for the model, it was partly covered with snow.

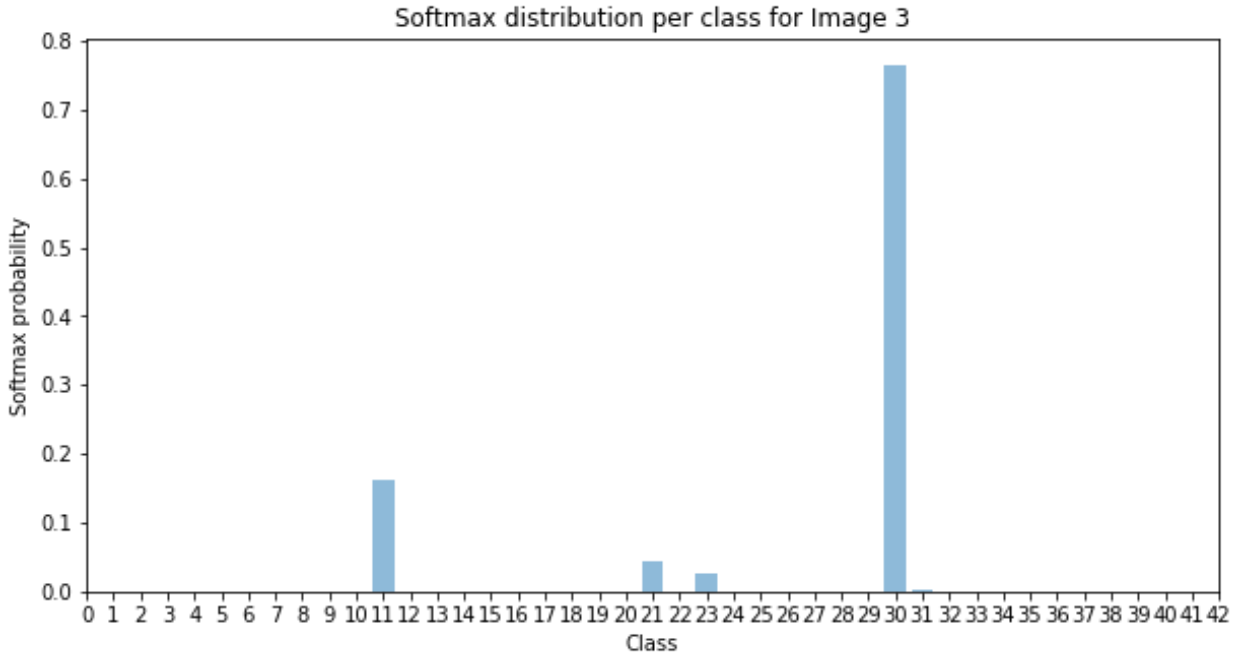
For the first image, the model is fully sure that this is a No entry sign (probability of 1.0). The top five soft max probabilities distribution were



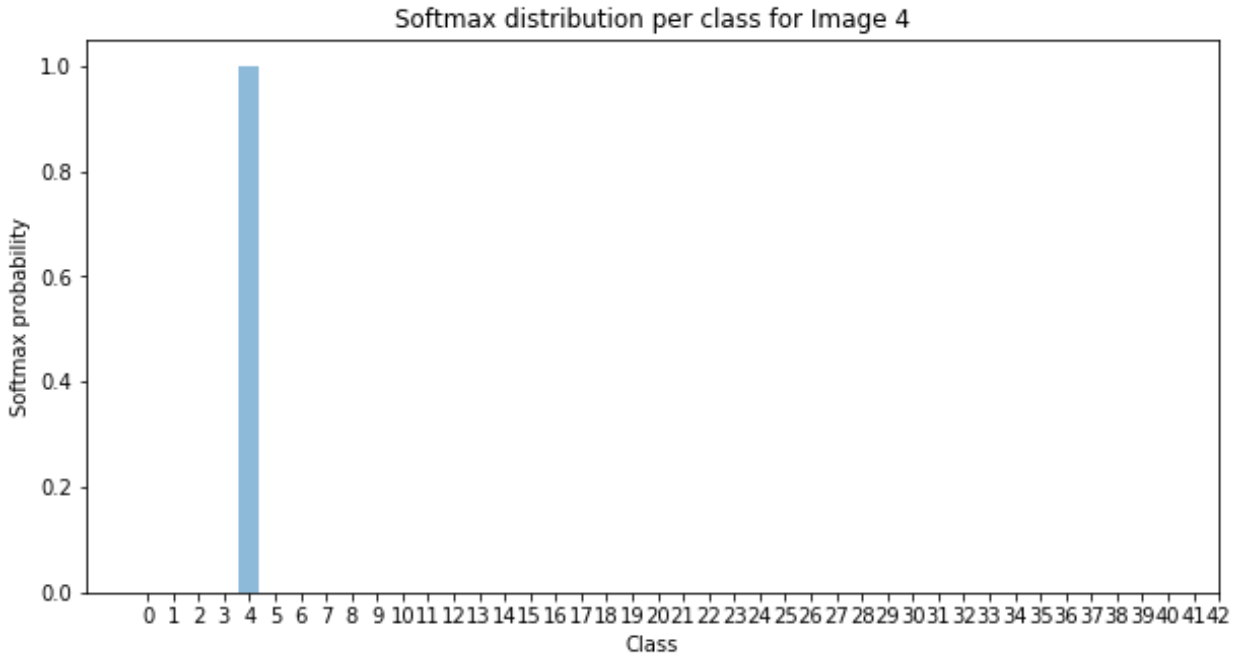
For the second image, the model is fully sure that this is a Stop sign (probability of 0.98). The top five soft max probabilities distribution were



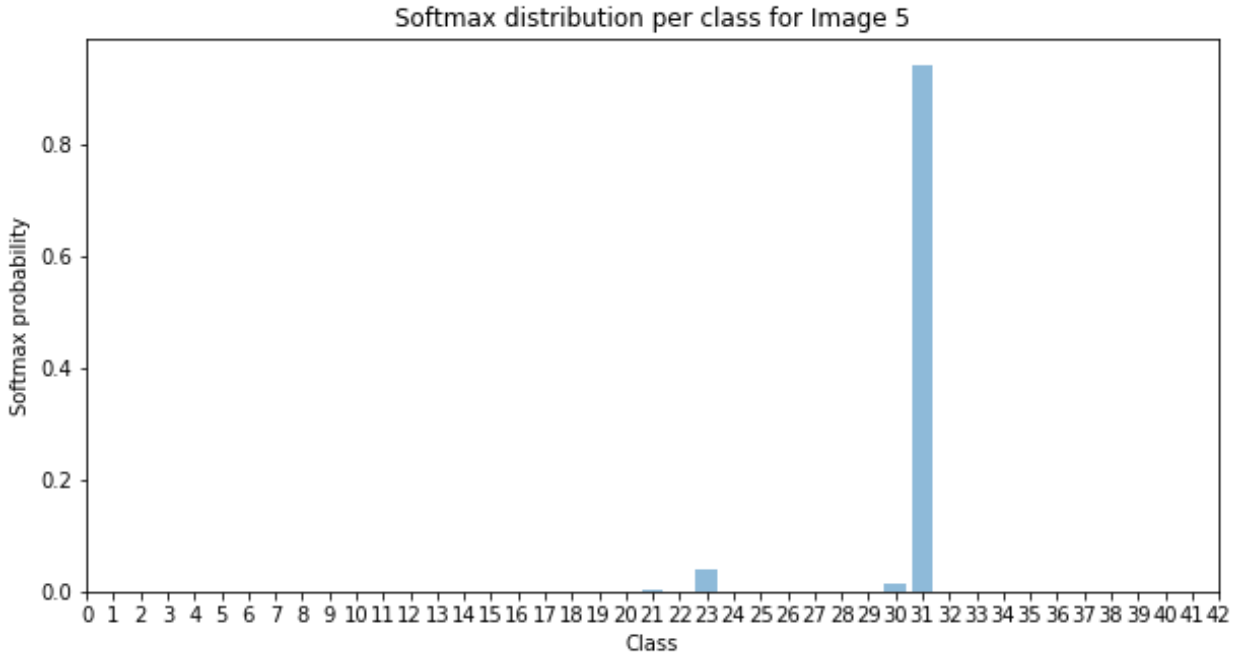
For the third image, the model is relatively sure that this is a Beware of ice/snow sign (probability of 0.76) but the sign was of Slippery road. The top five soft max probabilities distribution were



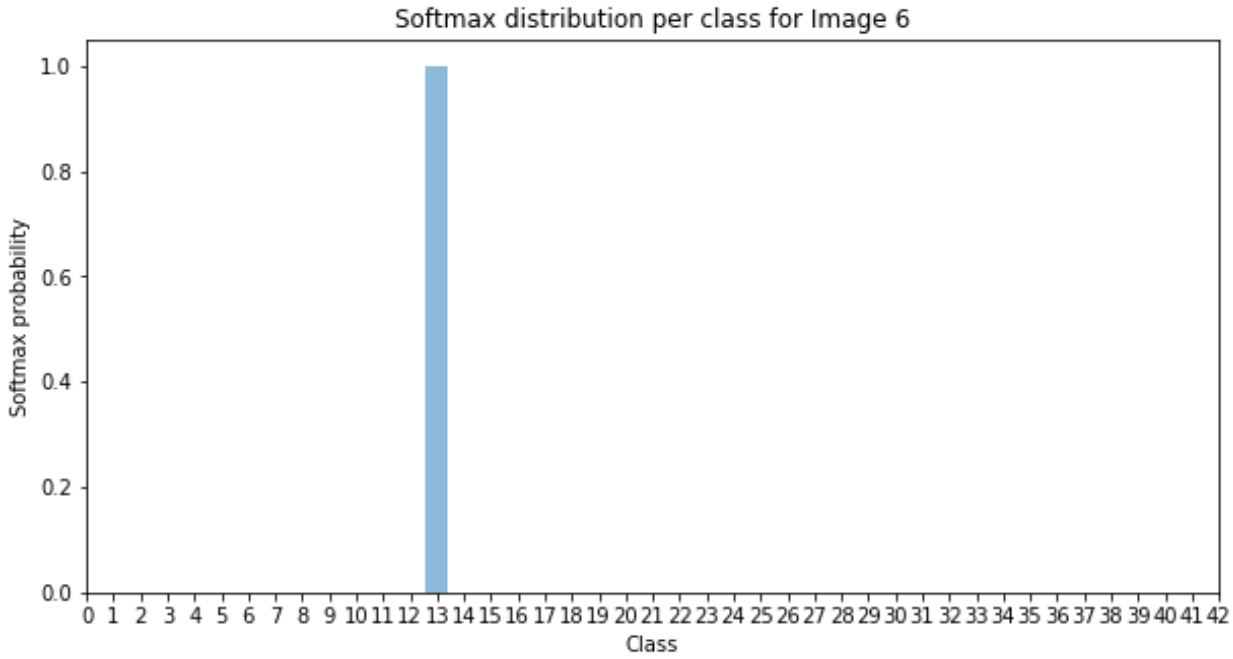
For the fourth image, the model is fully sure that this is a Speed limit (70km/h) sign (probability of 1.0). The top five soft max probabilities distribution were



For the fifth image, the model is fully sure that this is a Wild animals crossing sign (probability of 0.94). The top five soft max probabilities distribution were



For the sixth image, the model is fully sure that this is a Yield sign (probability of 1.0). The top five soft max probabilities distribution were



For the seventh image, the model is fully sure that this is a Yield sign (probability of 1.0). The top five soft max probabilities distribution were