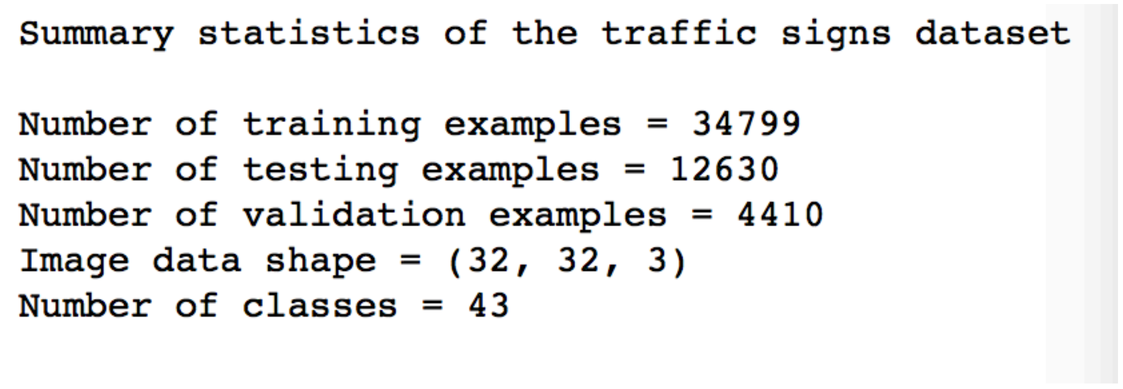
Self Driving Car Engineer NanoDegree
Project 2: Traffic Sign Recognition
By: Mano Paul

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

###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 regular python length 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 x 32 x 3
• The number of unique classes/labels in the data set is 43



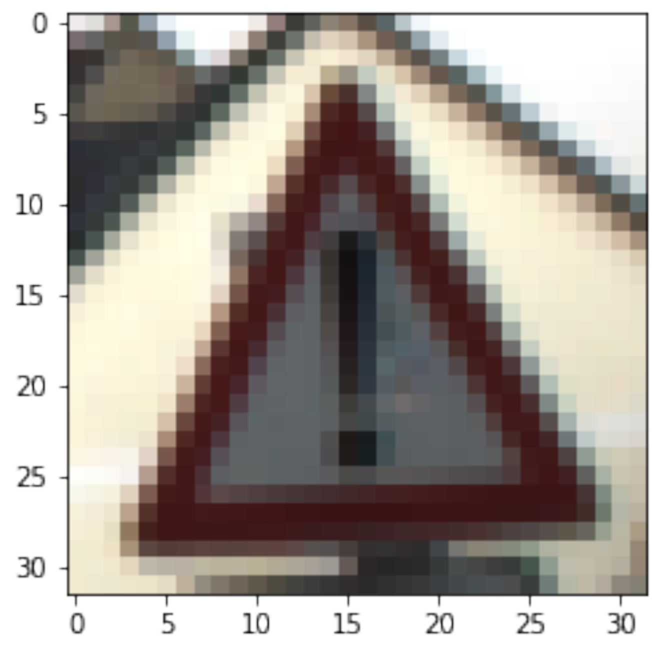
```
Summary statistics of the traffic signs dataset

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
```

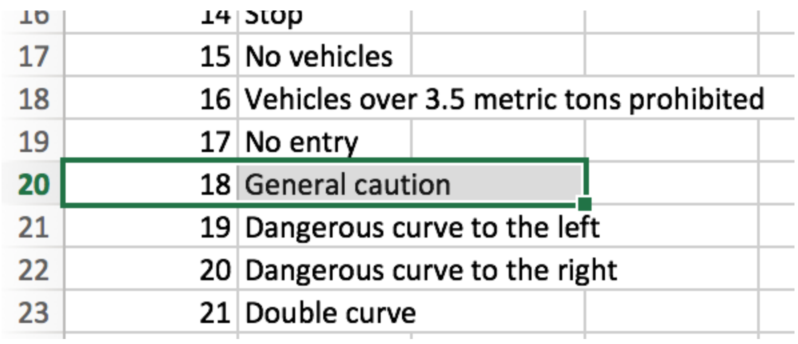
####2. Include an exploratory visualization of the dataset.

Here is an exploratory visualization of the data set.
It is a random image from the dataset along with its index. The index maps to the label in the reference signnames file that is comma-separated.

```
18
```



In this example, the index was 18. The reference file has the class id of 18 maps to “General Caution”



16	14	Stop
17	15	No vehicles
18	16	Vehicles over 3.5 metric tons prohibited
19	17	No entry
20	18	General caution
21	19	Dangerous curve to the left
22	20	Dangerous curve to the right
23	21	Double curve

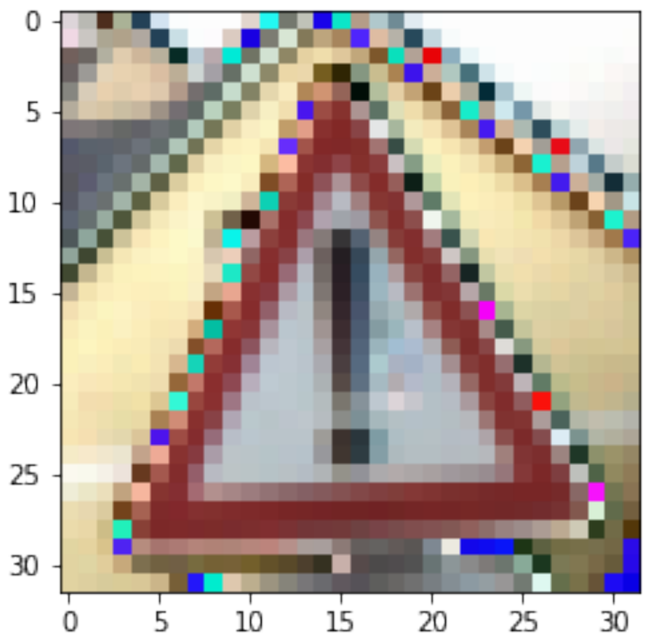
Design and Test a Model Architecture
1. Describe how you preprocessed the image data. What techniques were chosen and why did you choose these techniques? Consider including images showing the output of each preprocessing technique. Pre-processing refers to

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.)

Pre-processing Image Data
Conversion to grayscale reduced the channel layers from 3 (RGB) to 1 and this posed an issue with array dimensions not matching correctly when training the data using tensorflow. Need to reconcile this but in the interest of time, I resorted to normalization instead of grayscale conversion for preprocessing. The channel reduction would have optimized the training time.

I normalized the image data because this makes the mean value of the image to go toward zero and this helps in optimizing the training. For images since the pixel value is between 0 and 255, subtracting 128 from the pixel value and dividing by 128 can be used to normalize the data. However, I used the
normalized_grayscale_image = a + (((image_data - grayscale_min)*(b - a))/(grayscale_max - grayscale_min))
formula, where a=0.1, b=0.9, grayscale_min = 0 and grayscale_max=255

To test normalization, the random image selected earlier was normalized and the normalized image looked as shown below:



At the end of the preprocessing the training data was shuffled. Shuffling was accomplished using the function shuffle in the sklearn.utils library.

####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 architecture leveraged was the LeNet model which has the following sequence.
INPUT -> Convolution 1 -> RELU activation -> Pooling -> Convolution 2 -> RELU activation -> Pooling -> Flatten -> Fully Connected -> Activation -> Fully Connected

The image below shows each layer and the shape of the tensor it outputs. In total, we have 2 convolutional layers and three fully connected layers that output a prediction of 43 logits for the 43 different classes of traffic signs.

My final model consisted of the following layers as shown in the image:

```
ConvNet 1:  Tensor("add:0", shape=(?, 28, 28, 6), dtype=float32)
ConvNet 2:  Tensor("add_1:0", shape=(?, 10, 10, 16), dtype=float32)
Fully Connected 0:  Tensor("Flatten/Reshape:0", shape=(?, 400), dtype=float32)
Fully Connected 1:  Tensor("Relu_2:0", shape=(?, 120), dtype=float32)
Fully Connected 2:  Tensor("Relu_3:0", shape=(?, 84), dtype=float32)
Logits:  Tensor("add_4:0", shape=(?, 43), dtype=float32)
```

####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 the following:
Model: LeNet
Optimizer: AdamOptimizer
Number of epochs: 100
Batch size: 150
Learning rate of 0.001
Mean: 0
Sigma: 0.1

####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:
• training set accuracy of 99.3
• test set accuracy of 91.8
• validation set accuracy of 93.3

• The LeNet architecture was chosen for its reputation in the industry to be a robust architecture. Since the LeNet architecture takes a 32x32xC where C is the number of channels input and the images in the German Training Dataset is 32x32x3 (RGB) dataset, the LeNet architecture is a good starting architecture for traffic sign classification.

- The final model's validation accuracy was 90 and this provides evidence that the model is working well with just normalized data. If the dataset is converted to grayscale, it would speed up the training time.

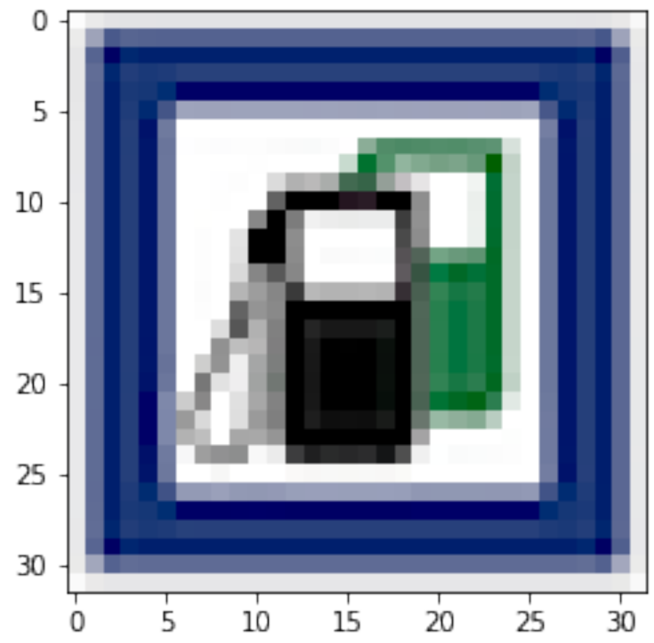
###Test a Model on New Images
####1. Choose five German traffic signs found on the web and provide them in the report. For each image, discuss what quality or qualities might be difficult to classify.

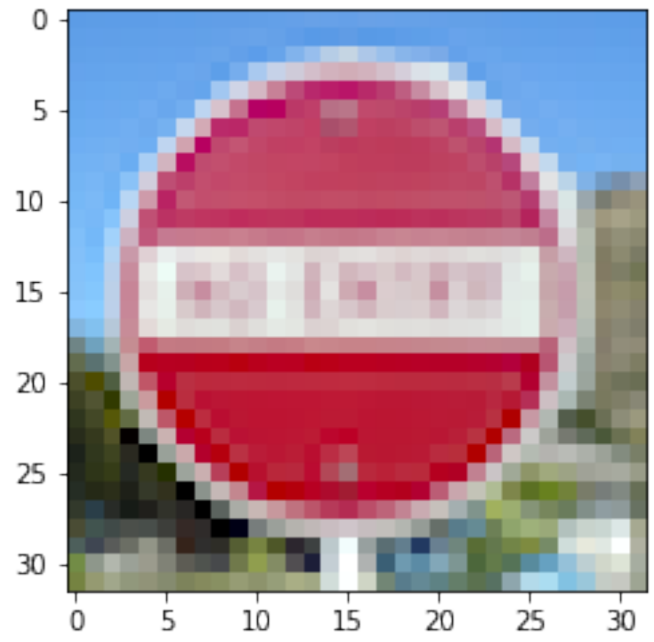
Here are six German traffic signs that I found on the web.
1st image: Gasoline
2nd image: No entry sign
3rd image: Pedestrian Crossing
4th image: Railroad Crossing
5th image: Speed Limit 30km/h
6th image: Stop

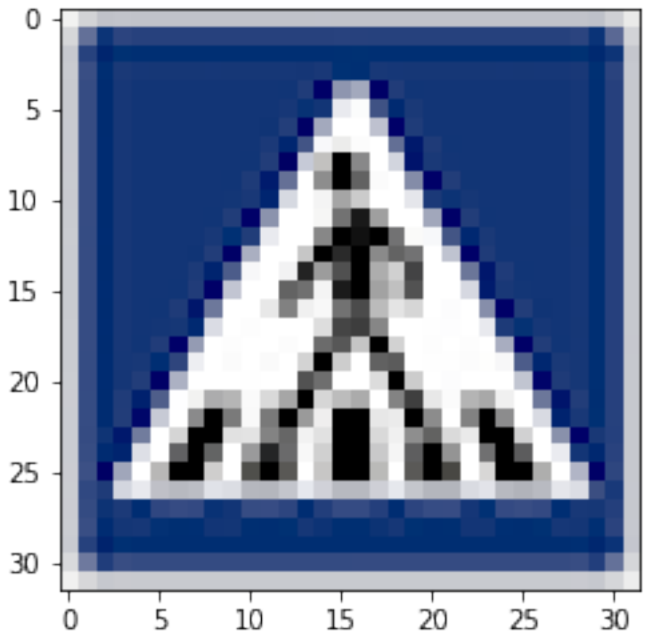
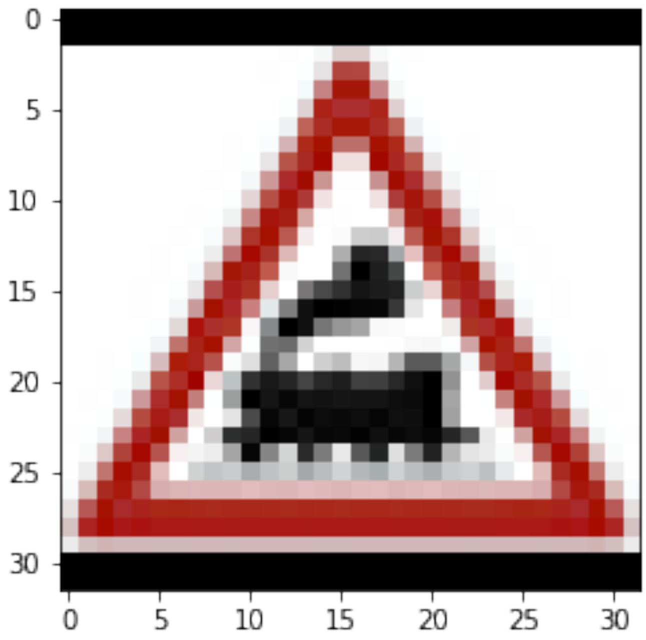
#	**Image**
1	Gasoline
2	No-entry sign
3	Pedestrian crossing
4	Railroad crossing
5	Speed Limit 30km/h
6	Stop

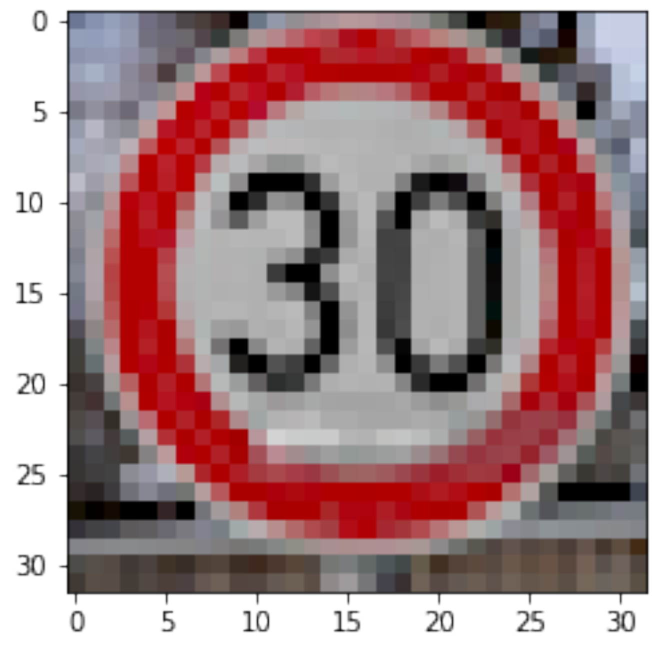
They were stored into a local directory and loaded for processing.
After normalize these images, they were tested against the

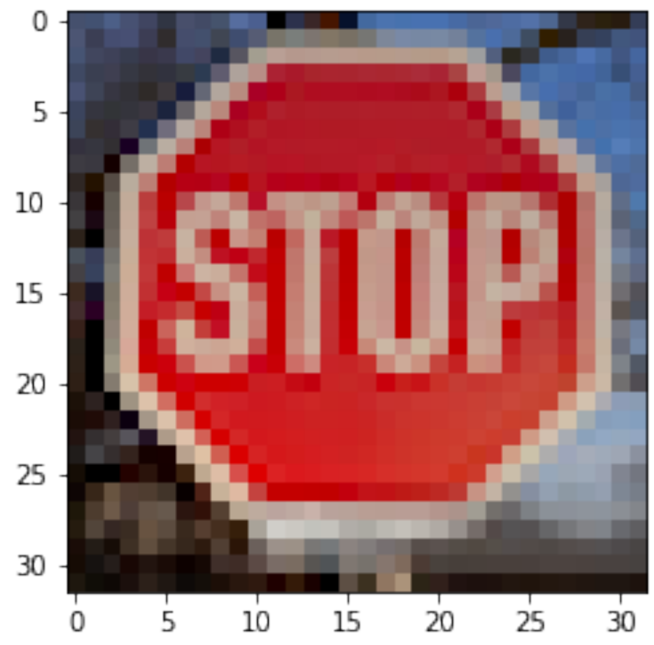
previously trained LeNet neural network (nn) to see if the nn would predict/classify these images correctly.






0
5
10
15
20
25
30
0
5
10
15
20
25
30

0
5
10
15
20
25
30
0
5
10
15
20
25
30





The first (Gasoline) and fourth (Railroad Crossing) image might be difficult to classify because they are not there in the 43 classes of traffic signs.

####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).

The Sign Names file that was read displayed
```
(0, b'Speed limit (20km/h)')
(1, b'Speed limit (30km/h)')
(2, b'Speed limit (50km/h)')
(3, b'Speed limit (60km/h)')
(4, b'Speed limit (70km/h)')
(5, b'Speed limit (80km/h)')
(6, b'End of speed limit (80km/h)')
(7, b'Speed limit (100km/h)')
(8, b'Speed limit (120km/h)')
(9, b'No passing')
(10, b'No passing for vehicles over 3.5 metric tons')
(11, b'Right-of-way at the next intersection')
(12, b'Priority road')
(13, b'Yield')
(14, b'Stop')
(15, b'No vehicles')
(16, b'Vehicles over 3.5 metric tons prohibited')
(17, b'No entry')
(18, b'General caution')
(19, b'Dangerous curve to the left')
(20, b'Dangerous curve to the right')
(21, b'Double curve')
(22, b'Bumpy road')
(23, b'Slippery road')
(24, b'Road narrows on the right')
(25, b'Road work')
(26, b'Traffic signals')
(27, b'Pedestrians')
(28, b'Children crossing')
(29, b'Bicycles crossing')
(30, b'Beware of ice/snow')
(31, b'Wild animals crossing')
(32, b'End of all speed and passing limits')
(33, b'Turn right ahead')
(34, b'Turn left ahead')
(35, b'Ahead only')
(36, b'Go straight or right')
(37, b'Go straight or left')
(38, b'Keep right')
(39, b'Keep left')
(40, b'Roundabout mandatory')
(41, b'End of no passing')
(42, b'End of no passing by vehicles over 3.5 metric tons')
```

The predictions of the images

Here are the results of the prediction:

The possible predictions for the 6 images were one of the following.
Speed limit (20 km/h) (index -1)
Speed limit (50 km/h) (index – 2)
Speed limit (60 km/h) (index – 3)
Speed limit (70 km/h) (index – 4)
Speed limit (80 km/h) (index - 5)
Right of way at next intersection (index -11)
Priority Road (index -12)
Yield (index – 13)
General Caution - (index – 18)
Wild animals crossing (index - 31)
Beware of ice/snow (index -30)
End of all speed and passing limits (index -32)

Although the validation accuracy was over 93% the predictions were inaccurate in predicting the images correctly.

#	**Image**
1	Gasoline
2	No-entry sign
3	Pedestrian crossing
4	Railroad crossing
5	Speed Limit 30km/h
6	Stop

Interestingly the speed limit sign of 30km/h was not in even one of the predictions. It is likely that the No entry sign was classified as End of all speed and passing limits.

####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 image 0 is about [ 0.6922667   0.30023319  0.00215537  0.001801    0.00096669] chance to be the sign with index:
[ 3  5 32 11 30]
The image 1 is about [ 0.82363033  0.08360636  0.04133265  0.02295604  0.02141334] chance to be the sign with index:
[ 3  1 32 13  5]
The image 2 is about [ 0.95035684  0.01710486  0.01516867  0.00725479  0.00688135] chance to be the sign with index:
[18  1 12 11 31]
The image 3 is about [ 0.88734788  0.09415069  0.01257696  0.00161749  0.00147223] chance to be the sign with index:
[ 3  5 18 30 11]
The image 4 is about [  9.28528726e-01   6.53521493e-02   1.69922085e-03   9.00210172e-04
   8.63650115e-04] chance to be the sign with index: [ 3  5 31 12 25]
The image 5 is about [ 0.47229764  0.24313045  0.13881393  0.04215979  0.0407932 ] chance to be the sign with index:
[ 3  5  2 12  1]
```

#	Image
1	Gasoline
2	No-entry sign
3	Pedestrian crossing
4	Railroad crossing
5	Speed Limit 30km/h
6	Stop

For the first image (Gasoline), the model is relatively sure that this is a Speed limit of 60km/h sign (probability of 0.69). The top five soft max probabilities were:

Probability	Prediction
.69	Speed limit 60km/h
.30	Speed limit 80km/h
.002	End of all speed and passing limits
.001	Right of way at the next intersection
.009	Beware of ice/snow

For the second image (No entry sign), the model is relatively sure that this is a Speed limit of 60km/h sign (probability of 0.82). The top five soft max probabilities were:

Probability	Prediction
.82	Speed limit 60km/h

.08	Speed limit 30km/h
.04	End of all speed and passing limits
.02	Yield
.02	Speed limit 80km/h

For the third image (Pedestrian Crossing), the model is relatively sure that this is a General Caution (probability of 0.95). The top five soft max probabilities were:

Probability	**Prediction**
.95	General Caution
.017	Speed limit 30km/h
.015	End of all speed and passing limits
.007	Right of way at the next intersection
.006	Wild animal crossing

For the fourth image (Railroad Crossing), the model is relatively sure that this is a Speed limit 60 km/h sign (probability of 0.89). The top five soft max probabilities were:

Probability	**Prediction**
.89	Speed limit 60km/h
.094	Speed limit 80km/h
.012	General Caution
.001	Beware of ice/snow
.001	Right of way at the next intersection

For the fifth image (Speed Limit 30km/h), the model is relatively sure that this is a Speed limit 60 km/h sign (probability of 0.92). The top five soft max probabilities were:

Probability	**Prediction**
.92	Speed limit 60km/h
.065	Speed limit 80km/h

.016	Wild animals crossing
.009	Priority Road
.008	Road work

For the fifth image (Speed Limit 30km/h), the model is relatively sure that this is a Speed limit 60 km/h sign (probability of 0.92). The top five soft max probabilities were:

Probability	**Prediction**
.92	Speed limit 60km/h
.065	Speed limit 80km/h
.016	Wild animals crossing
.009	Priority Road
.008	Road work

For the sixth image (Stop sign), the model is relatively sure that this is a Speed limit 60 km/h sign (probability of 0.47). The top five soft max probabilities were:

Probability	**Prediction**
.47	Speed limit 60km/h
.24	Speed limit 80km/h
.13	Speed limit 50km/h
.04	Priority Road
.04	Speed limit 30km/h

These incorrect classifications could have possibly due to the low quality and granulated images.

DID NOT USE THIS
Converting to Grayscale

Using a random image from the dataset, first I converted the image from color to grayscale using opencv library in python. Then I plotted the same color image in grayscale as shown here.
Reading the shape of the color image and the grayscale image indicated that the channel for color image was 3 while there was no value for the grayscale image. Assuming that the channel is 1 for grayscale image.
The reason for converting to grayscale is because the training time on 1 channel would be faster than an image with 3 channels.

Here is an example of a traffic sign image before and after grayscaling.