Traffic Sign Classification

James Brodovsky

Dataset Exploration

Starting off, I loaded the data from the pickle files and examined the shape of the data. The data was organized into a list of 32x32x3 images. I then found the length of the training (34,799), validation (4,410), and test (12,630) sets, and calculated the number of classes in the dataset to be 43 by finding the maximum value in the 'labels' list and adding 1 to account for zero-indexing. I found that the average size of the image files to be 27x27.5.

Exploring the dataset started with plotting a sample image and creating a histogram of the class counts. I found that images tended to be fairly blurry as a factor of them being low resolution. The training dataset contained an equal number of sample images for each class.

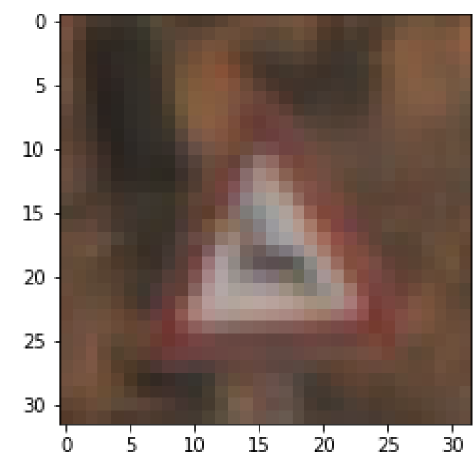

Figure 1: Sample dataset image.

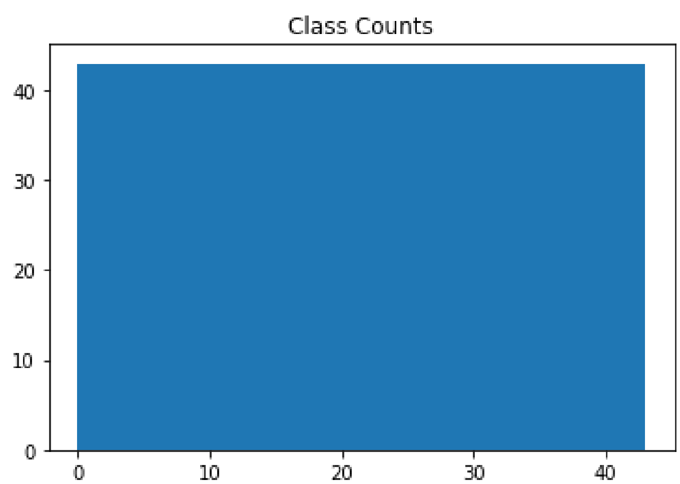

Figure 2: Class counts.

I examined a couple of preprocessing options. I scaled the RGB of the raw images to the range [0, 1], normalized the RGB to [-1, 1], converted to grayscale, and identified edges using Canny edge detection.

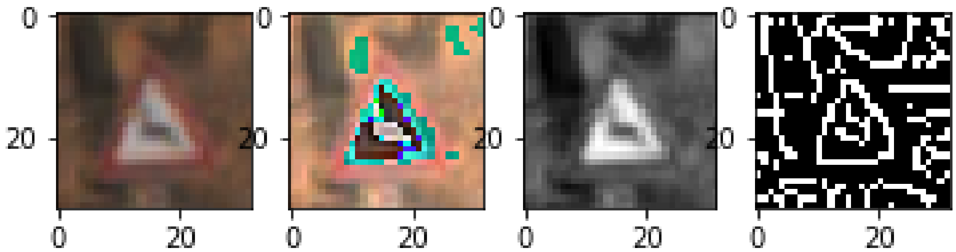

Figure 3: Preprocessing examples – scaling, normalization, grayscale, and edges.

I decided on scaling and normalization preprocessing as that ended up giving the best training results after a few preliminary training sessions. Preprocessing the datasets yielded an Nx32x32x6 structure.

Design and Test a Model Architecture

I used a standard LeNet-5 implementation with a few minor modifications. First, I modified the first convolution layer to take a 6 channel input, thus used a 5x5x6x6 convolution. Second, I modified the final fully-connected layer to have a shape of 86x43 to correspond the number of classes in the dataset. After a few preliminary training sessions it became apparent that the validation set would lag behind the accuracy achieved on the training set. This indicated a degree of over-fitting to the training set. To compensate for this a dropout activation replace the ReLu activation on the first two fully connected layers (layers 3 and 4, respectfully).

I then set up a training session that check the training accuracy, validation accuracy, and the change in both. This continued to train while either change in accuracy was positive and the validation accuracy was less than the desired 0.93. This resulted in some early termination, so I modified the training parameters to continuing training even if both changes were negative while the validation accuracy was still over 0.8. I then tuned the keep probability of the dropout layers and the learning rate. I settle on a keep probability of 75% and a learning rate of 0.0005. I was able to achieve a 99.1% accuracy on the training set and a 93.1% accuracy on the validation set after 37 epochs.

This model was saved and run through the test set. It achieved a 93.4363% accuracy on the test set.

Test a Model on New Images

Using a Google image search I downloaded an image of a German “Stop”, “Bicycle Crossing”, “Keep Left”, “Keep Right”, and “No Entry” signs.

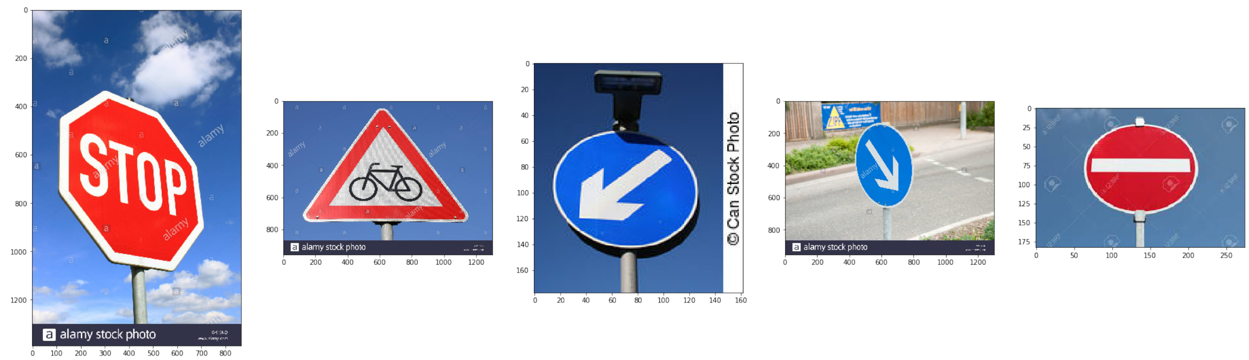

Figure 4: New images.

These five images were then resized to be 32x32 pixels, preprocessed, and run through the model to generate predictions.

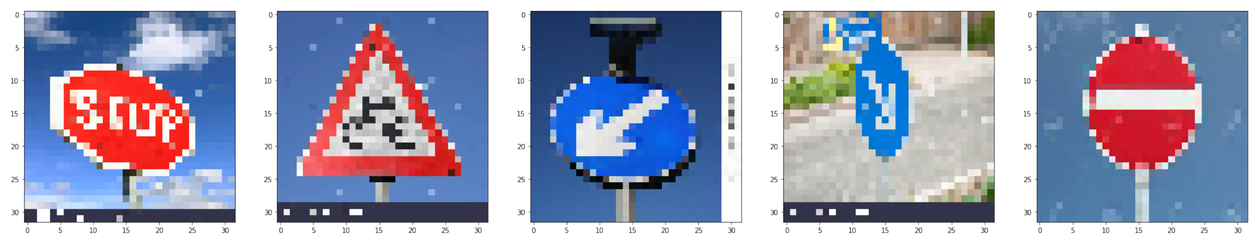

Figure 5: Resized new images.

These new images were then run through the model. The “Keep Left” and “No Entry” signs were successfully predicted for a total accuracy of 40%. Softmax probabilities of the predictions are given below.

Table 1: New image prediction confidence indicators.

New Image Predictions				
Stop Sign	Prediction	17	14	29
14	Confidence	100.00%	0.00%	0.00%
Bicycle Crossing	Prediction	25	29	30
29	Confidence	100.00%	0.00%	0.00%
Keep Left	Prediction	39	33	37
39	Confidence	100.00%	0.00%	0.00%
Keep Right	Prediction	11	40	31
38	Confidence	99.995%	0.005%	0.00%
No Entry	Prediction	17	10	11
17	Confidence	99.972%	0.025%	0.004%

The high degree of certainty, even in the misclassifications, indicates that there is something amiss. What I think is likely is the resizing and framing of the image. The two signs that were successfully classified had similar orientation to the camera and position in the frame relative to the training dataset. Further, it appears that the training images were crops from larger images rather than a wholesale scaling of the entire image. To improve the performance of this model additional processing should be done to standardize the raw images in to more closely match those of the training set. Alternatively, include more image such as these that are less standardized so that the model becomes more robuts.