Traffic Sign Recognition

Writeup

Data Set Summary & Exploration

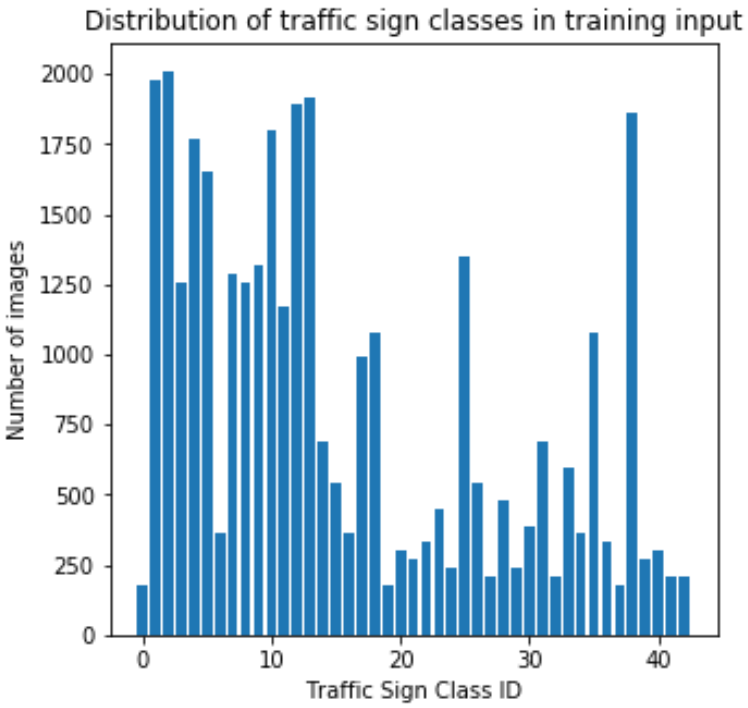
1. Summary of the data set. In the code, the analysis should be done using python, numpy and/or pandas methods rather than hardcoding results manually.

I used the numpy library 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 32x32x3.
- The number of unique classes/labels in the data set is 43.

2. Exploratory visualization of the dataset.

Here is an exploratory visualization of the data set. It is a bar chart showing the distribution of the traffic sign classes in the training input images. Here it can be seen that classes 1, 2, 12, 13 and 38 have the most images in the training set.



Design and Test a Model Architecture

1. Image Preprocessing

I felt that the color information of the different traffic signs was a valuable parameter for distinguishing the traffic signs. Hence, I did not covert the images from RGB to Grayscale. However, I did normalize the input image to have zero mean.

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.

My final model consisted of the following layers:

Layer	Description
Input	32x32x3 RGB image
Convolution 5x5	1x1 stride, valid padding, outputs 28x28x12
RELU	
Max pooling	2x2 stride, outputs 14x14x12
Convolution 5x5	1x1 stride, valid padding, outputs 10x10x28
RELU	
Convolution 5x5	1x1 stride, valid padding, outputs 6x6x36
RELU	
Max Pooling	2x2 stride, outputs 3x3x36
Fully connected	Input: 324 Output: 240
RELU	
Fully connected	Input: 240 Output: 98
RELU	
Fully connected	Input: 98 Output: 43

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 training strategy that was used for training LeNet in the previous exercise. I used the Adam optimizer as it was mentioned to be a a good extension of SGD. I kept the learning rate at 0.001, as I wanted my model to settle slowly, over 10 epochs into its final state. For training the model, I used a batch size of 128 images.

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 99.8%

- Validation set accuracy of 97%
- Test set accuracy of 95.1%

I chose an iterative approach as the original network architecture was already behaving pretty well with the changed images too. Hence, I decided to build on that. However, I realized that the accuracy of validation was not going beyond 89%. This indicated that the network did not have sufficient parameters to characterize the traffic sign images, and that it was underfitting.

Hence, I decided to add more parameters by making the second convolutional layer deeper. This increased the validation dataset accuracy to 91%, but it was still short of the required 93%.

At this point of time, I decided to add more parameters and normalize all the data. I added another convolutional layer so that it could add more effective depth to the network for storing more parameters after the first convolutional layer. I dropped one max pooling layer as it was leading to data loss. I also added a dropout layer in between the fully connected layers so that the network will be robust to some inputs not being present so that it could generalize the inference.

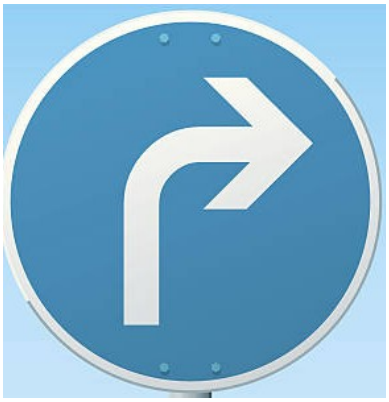
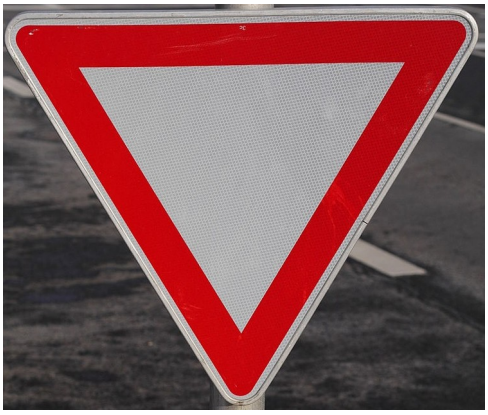
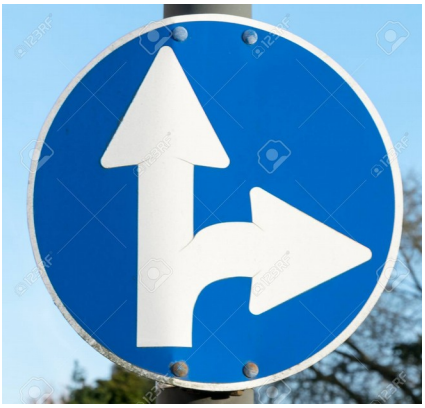
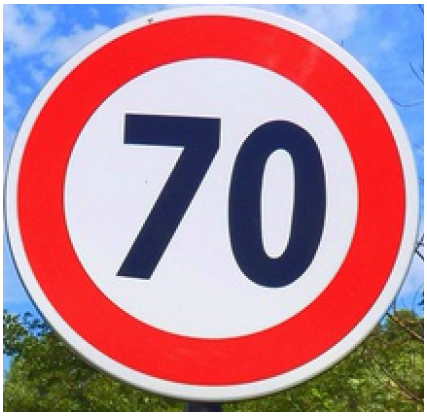
I stuck with the default learning rate and epoch count as they worked for me.

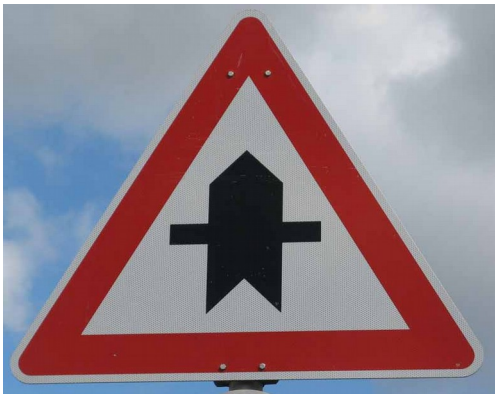
Using the revamped architecture, I was able to achieve a pretty high accuracy.

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 five German traffic signs that I found on the web:



The first image is pretty clear and should be simple to classify. The second and fourth images should be challenging as the environments of the signs could throw off the network. The third image should be difficult to classify because it is pretty similar to some signs. Also, the image has watermarks on it which can serve as noise. The fifth image will test the network's response to signs with less illumination. Overall, the set tests the response of the network to brightness too.

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).

Here are the results of the prediction:

Image	Prediction
Turn right ahead	Turn right ahead
Yield	Yield
Go straight or go right	Go straight or go right
70 km/h	60 km/h
Right of way at next intersection	Right of way at next intersection

The model was able to correctly guess 4 of the 5 traffic signs, which gives an accuracy of 80%. This compares unfavorably with the test set accuracy. However, we need to keep into consideration that these are just 5 images and just a single wrong prediction can drop accuracy from 100% to 80%.

In this case, the network performed well in almost all cases. However, it got confused with the sign with the 70 km/h speed limit and predicted it to be 60 km/h. This can be due to the similar shapes of the signs along with the contents.

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)

For the first image, the model is extremely sure that this is a "Turn Right Ahead" sign (probability of 0.99), and the image does contain the same sign. The top five soft max probabilities were

Probability	Prediction
.99	Turn Right Ahead
5.2261566e-06	Ahead Only
7.3018022e-08	70 km/hr
5.4251206e-08	Keep right
2.3633202e-08	80 km/hr

For the second image, the network is completely sure that the image is a yield sign, with a probability of nearly 1.

The top five softmax probabilities were:

Probability	Prediction
1.00000e+00	Yield
2.3822607e-22	No Vehicles
4.8681307e-23	Road work
9.0203673e-24	No passing
2.0791327e-26	Ahead only

For the third image as well, the network is pretty sure that the image is a “Go straight or go right” sign, with a probability of nearly 0.99.

The top five softmax probabilities were:

Probability	Prediction
9.99999976e-01	Go straight or go right
1.8436631e-07	Keep right
1.2037944e-10	End of no passing
9.2007055e-11	Ahead only
7.8432115e-11	Dangerous curve to the right

For the fourth image too, the network is pretty sure that the image is a “60 km/hr” sign, with a probability of nearly 0.99. However, it is relatively more unsure than the other four images about the remaining images. The actual sign gets the lowest of the top 5 softmax probabilities.

The top five softmax probabilities were:

Probability	Prediction
9.9997795e-01	60 km/hr
2.1118827e-05	80 km/hr
8.9822697e-07	Wild animals crossing
9.7055183e-11	Ahead only
8.1984232e-11	70 km/hr

The network predicted the fifth image correctly to be a “Right of way on next intersection” with a probability of 0.99. The top five softmax probabilities are:

Probability	Prediction
9.9999583e-01	Right of way on next intersection
4.1915223e-06	Beware of ice/snow
5.3396528e-12	Slippery road ahead
1.2637004e-12	Pedestrians
8.3137639e-14	Dangerous curve to right