Traffic Sign Recognition

Writeup

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

Rubric Points

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

Writeup / README

1. Provide a Writeup / README that includes all the rubric points and how you addressed each one. You can submit your writeup as markdown or pdf. You can use this template as a guide for writing the report. The submission includes the project code.

You're reading it! and here is a link to my [project code]()

Data Set Summary & Exploration

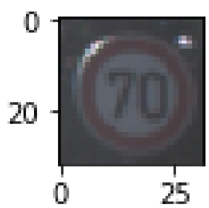
1. Provide a basic summary of the data set. In the code, the analysis should be done using python, numpy and/or pandas methods rather than hardcoding results manually.

I used the pandas library to calculate summary statistics of the traffic
signs data set:

- The size of training set is 34799
- The size of 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

2. Include an exploratory visualization of the dataset.

Here is sample visualization of the data set. It is a random sample image showing the dimensions and characteristics of an image in the dataset.



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.

I normalized the image data because gradient based algorithms need the ranges of possible feature values to be consistent to work properly.

I decided not to grayscale the data because traffic signs can be identified by their color. I believe that any real world traffic sign classifier on a car must use color in their classifier.

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.

I used the Lenet model and added nodes to the last 3 fully connected layers as well as adding 2 dropout layers in between the fully connected layers.

My final model consisted of the following layers:

Layer	Description
Convolution	Convolutional. Input = 32x32x3. Output = 28x28x20.
Relu Activation	
Max Pooling	Input = 28x28x20. Output = 14x14x20.
Convolution	Output = 10x10x32.
Relu Activation	
Max Pooling	Input = 10x10x32. Output = 5x5x32.
Flatten	Input = 5x5x32. Output = 800.
Fully Connected	Input = 800. Output = 400.
Relu Activation	

Dropout	.50 Keep Prob
Fully Connected	Input = 400. Output = 200.
Relu Activation	
Dropout	.50 Keep Prob
Fully Connected	Input = 200. 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 adam optimizer with 100 epochs with a batch size of 128 and a learning rate of 0.001.

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 99.9%
* validation set accuracy of 93.6%
* test set accuracy of 92.6%

If an iterative approach was chosen:

*I started by using the lenet model with no preprocessing as a baseline and I was getting around 88% validation. The lenet model is a CNN that was designed for the MNist data set. Signs are simple images like the MNIST data set and it seems like a suitable model. CNNs are particularly suited for computer vision applications.

*I implemented normalization and grayscale but was not seeing improvements so I decided to keep the rgb as I theorized that real world models would need to know the colors of signs.

*I felt like my model was underfitting so I decided to add nodes to the fully connected layers I double the size of each of the layers.

* I tried lowering and raising the learning rate but it seemed to work well at my original value of .001

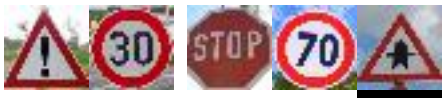
* I raised the number of epochs from 10 originally to 100 iteratively after trying many different values in between and still seeing improvement.

The final model's training accuracy was near 100 the validation and test accuracy were relatively similar which implies that I did not overfit to the validation data.

3. 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:



These images should all be relatively easy to classify. They are relatively clear compared to the samples that I saw from the training data and are cropped similarly.

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

Here are the results of the prediction:

Image	**Prediction**
General Caution	General Caution
Speed Limit 30	Speed Limit 30
Stop	Stop
Speed Limit 70	Road Work
Right of way	Right of Way

The model was able to correctly guess 4 of the 5 traffic signs, which gives an accuracy of 80%. This compares favorably to the accuracy on the test set of 92.6%.

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.

For the first image, the model is positive that this is a caution sign (probability of 100%), and the image does contain a caution sign. The top five soft max probabilities were:

Probability	Prediction
100%	General Caution
0	
0	
0	
0	

For the second image the model was relatively sure that this is a speed limit 30 sign (probability of 69%) and the image does contain a speed limit 30 sign. The top five soft max probabilities were:

Probability	Prediction
69%	Speed Limit 30
27%	Speed Limit 50
4%	Speed Limit 80
0	No Vehicles
0	Speed Limit 70

For the third image the model was positive that this was a stop sign (probability of 100%) and the image does contain a stop sign. The top five soft max probabilities were:

Probability	Prediction
100%	Stop
0	
0	
0	
0	

For the third image the model was almost positive that this was a road work sign (probability of 96%) and the image does not contain a road work sign. The top five soft max probabilities were:

Probability	Prediction
96%	Road Work
3%	Speed Limit 70
1%	Road Narrows
0	
0	

For the third image the model was positive that this was a right of way sign (probability of 100%) and the image does contain a right of way sign. The top five soft max probabilities were:

Probability	Prediction
100%	Right of way
0	
0	
0	
0	