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

Data Set Summary & Exploration

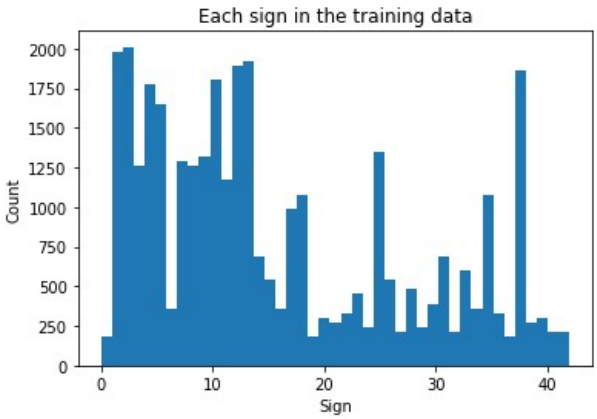
1. Provide a basic summary of the data set. In the code, the analysis should be done using python, numpy and/or pandas methods rather than hardcoding results manually.

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 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.

2. Include an exploratory visualization of the dataset.

The bar chart shows how many training data is available for training data in each different traffic sign.In general, the data is not very uniform for each traffic sign.



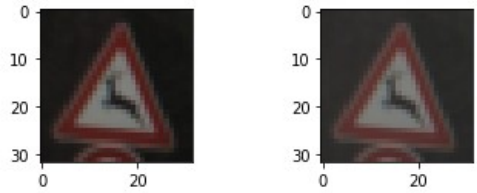
Design and Test a Model Architecture

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.)

For traffic signs, the recognition rate of gray image is high, but this time I want to keep the color and see if the color can improve the recognition rate.Implement Min-Max scaling in the normalize_grayscale() function to a range of a=0.1 and b=0.9. After scaling, the values of the pixels in the

input data should range from 0.1 to 0.9.

The effects are as follows:



Min-Max Scale equation:

$$\text{Min-Max Scaling: } X' = a + \frac{(X - X_{\min})(b-a)}{X_{\max} - X_{\min}}$$

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, same padding, outputs 28x28x32 | | RELU | | | Max pooling | 2x2 stride, outputs 14x14x32 | | Convolution 5x5 | 1x1 stride, same padding, outputs 10x10x64 | | RELU | | | Max pooling | 2x2 stride, outputs 5x5x64 | | Fully connected | inputs 5x5x64, outputs 1600 | | Dropout | keep*prob 0.5* | | *Fully connected* | *wx+b, inputs 1600, outputs 128* | | *RELU* | | | *Dropout* | *keep*prob 0.5 | | Output | wx+b, inputs 128, outputs 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.

I choose the following parameters to train the model:

Type of optimiser: AdamOptimizer

Batch size: 128

Training Epochs: 30

Learning rate: 0.001

Dropout (fully connected layers): 0.5

Padding: SAME

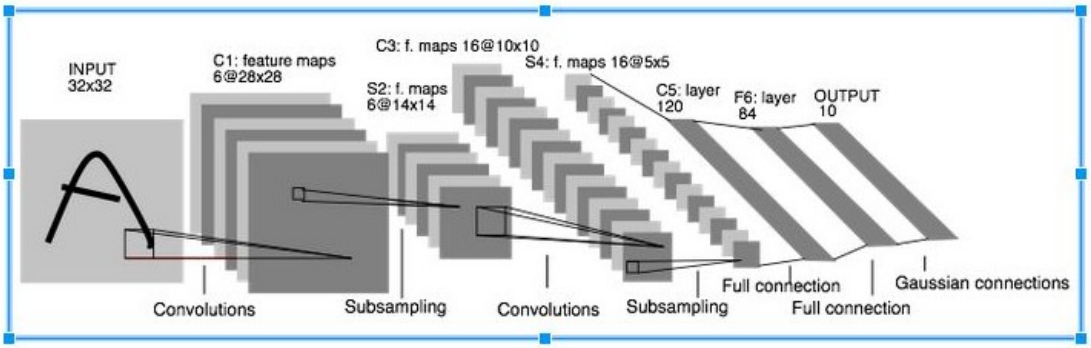
AdamOptimizer is the SGD optimizer, but it's more complicated than SGD. Due to insufficient computer memory, batch*size is 128. Epochs and learning*rate, which are self-adjusting to train the model many times to determine; Dropout (fully connected layers), I set up in the fully connected layer to improve the recognition accuracy of the model and prevent overfitting. Padding, for full Padding, is also to improve the recognition rate.

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:

- validation set accuracy of 0.977
- test set accuracy of 0.973

Convolutional Neural Nets (CNNs): 1989



First of all, I used the LeNet from Yann LeCun in 1998. As shown in the figure, the results of the unnormalized data are satisfactory, but the recognition rate has not reached more than 95%.In order to prevent the fitting in the two biggest I pooling layer and all the output of the link layer to join droput funciton, after several training after find it difficult to more than 93% accuracy, my solution is to adjust the parameters and network structure to train the model, parameters include: vector, keep_prob, convolution layer dimensions, convolution thickness when accuracy is more than 93% (learned this process: 1. The convolution of dimension and thickness accuracy is higher, the greater the but will be a big amount of calculation.2. According to actual condition, appropriate adjust the depth of the network architecture can improve recognition rate), very excited at this moment, and then download five new German traffic signs pictures, make model to simulate the image recognition, the result let me lost, accuracy is only 60%, I have been thinking about this is the reason why..... Finally, I decided to adjust the structure of network, again the dropout before the link layer are removed, because I think the pooling is an important characteristic of access to images, if droput is not random, the redundancy of training is not enough, that is likely to lose some important characteristic information, resulting in a decline in recognition rate . finally, I verify my idea is right, the new five images were correctly identified, and I found a complicated picture of German traffic signs, and the results were also 100%.So I think the architecture is suitable for the current problem.

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:

Traffic Sign 1，2 and 3, The size of the logo in the image, the shape and the background color are slightly different from the training data.

Traffic Sign 4, The picture came from the windy road, with leaves covering the part.

Traffic Sign 5: The shape is small and fuzzy.

★★Here's a little experiment. When there are three signs of traffic signs on the road, what is the model prediction? The pictures are as follows:

Please keep reading, You can get the answer.

---

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 | |:---------------------:|:---:| | Priority road | Priority road | | Stop | Stop | | Children crossing | Children crossing | | Wild animals crossing | Wild animals crossing | | Keep right | Keep right |

The model was able to correctly guess 5 of the 5 traffic signs, which gives an accuracy of 100%. Now the model can identify the important features of the image, such as an image of eight curves, eight shapes, etc. and finally combine the whole image to identify more complex objects in the image.

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 code for making predictions on my final model is located in the 11th cell of the Ipython notebook.

For the first image, the model is relatively sure that this is a Priority road sign (probability of 1.0), and the image does contain a Priority road sign. The top five soft max probabilities were 1.0.

| Probability | Prediction | |:---------------------:|:---:| | 1.0 | Priority road | | 0.0 | No entry | | 0.0 | Yield | | 0.0 | Speed limit (80km/h) | | 0.0 | Stop |

For the second image(a stop sign), The top five soft max probabilities were 1.0.

| Probability | Prediction | |:---------------------:|:---:| | 1.0 | stop | | 0.0 | No entry | | 0.0 | Speed limit (80km/h) | | 0.0 | Speed limit (60km/h) | | 0.0 | Speed limit (30km/h) |

For the third image(a Children crossing sign), The top five soft max probabilities were 1.0.

| Probability | Prediction | |:---------------------:|:---:| | 1.0 | Children crossing | | 0.0 | Beware of ice/snow | | 0.0 | Right-of-way at the next intersection | | 0.0 | Bicycles crossing | | 0.0 | Speed limit (60km/h) |

For the fourth image(a Wild animals crossing sign), The top five soft max probabilities were 1.0.

| Probability | Prediction | |:---------------------:|:---:| | 1.0 | Wild animals crossing | | 0.0 | Double curve | | 0.0 | Dangerous curve to the left | | 0.0 | Slippery road | | 0.0 | Speed limit (50km/h) |

For the fifth image(a Keep right sign), The top five soft max probabilities were 1.0.

| Probability | Prediction | |:---------------------:|:---:| | 1.0 | Keep right | | 0.0 | Turn left ahead | | 0.0 | Roundabout mandatory | | 0.0 | Go straight or right | | 0.0 | Go straight or left |

★★Now let's look at three signs of the road sign, and the model recognizes the details:

| Probability | Prediction | |:---------------------:|:---:| | 1.0 | Yield | | 0.0 | Priority road | | 0.0 | No passing for vehicles over 3.5 metric tons | | 0.0 | Turn left ahead | | 0.0 | Ahead only |

I'm estimating that the model will predict the middle part of the image, and as expected, Excellent！！

(Optional) Visualizing the Neural Network (See Step 4 of the Ipython notebook for more details)

1. Discuss the visual output of your trained network's feature maps. What characteristics did the neural network use to make classifications?