Traffic Sign Recognition

Writeup

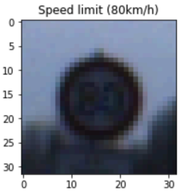
Data Set Summary & Exploration

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.**

Ans: I used pandas and numpy to calculate the number of training examples, number of validation numbers, number of testing numbers, shape of image, and unique classes. There are 34799 training examples, 4410 testing examples, image shape as [32,32,3], and unique labels of 43.

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

Ans:



Randomly picked a image and get a sense of what problems we are dealing with.

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.)**

Ans:

1. Turn it into grey to reduce training needs by using X_train = np.sum(X_train/3, axis=3, keepdims=True)
2. Using the quick way of (pixel-128)/128， I then shuffle it to have a better mix sequence of all labels images.

3. 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.

Ans:

This is adjusted based on LeNet

Layer	**Description**
Input	[32,32,1]
Conv1	Convolution layer, output_depth = 16, ksize=[5,5], strides = [1,1], padding = SAME
Pool	max_pooling, strides=[2,2], ksizes = [2,2]
Relu	Activation layer
Conv2	Convolution layer, output_depth = 64, ksize=[5,5], strides = [1,1], padding = SAME
Pool	max_pooling, strides=[2,2], ksizes = [2,2]
Relu	Activation layer
Conv3	Convolution layer, output_depth = 128, ksize=[5,5], strides = [1,1], padding = SAME
Pool	max_pooling, strides=[2,2], ksizes = [2,2]
Relu	Activation layer
Conv4	Convolution layer, output_depth = 512, ksize=[3,3], strides = [2,2], padding = SAME
Pool	max_pooling, strides=[2,2], ksizes = [2,2]
Relu	Activation layer
Flatten	Flatten layer
fc1	Fully connected,logits+Relu+dropout, output_depth=256
fc2	Fully connected,logits+Relu+dropout, output_depth=128
Final	Fully connected, logits, output_depth=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.

Ans:

Batch size = 156

Epoches = 20

Optimizer = AdamOptimizer

Learning rate = 0.00075

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.

Ans:

1. Start with LeNet structure, I need to change simple things such as output number, which should be 43, and input shape as [32,32,1].
2. The initial result with LeNet is not good enough, provided the sign is more complex and abstract than numbers, meaning we have to use more layers to get our abstract image.
3. After adding more layers, it's easy to get validation accuracy more than 80%, but below 93%. Maybe adding some regularization method is a good way.
4. After adding dropout layers, score goes up to 90%, but still no 93%. I was having 2048 inputs before FC layer, I reduce the size of the inputs basically by reducing the wide and high in the last convolution layer to left only 1x1, but a very deep of 512 output. The result is better now, maybe too many neurons make the net focus too much on noises.

5.The submission includes five new German Traffic signs found on the web, and the images are visualized. Discussion is made as to particular qualities of the images or traffic signs in the images that are of interest, such as whether they would be difficult for the model to classify.

Ans:

I download new data from GTSRB training set. I picked 6 images of different category. When I pick it, I actually picked images with clear show. The image size was very different, so I have to resize them before preprocessing. The preprocessing is the same as before: Turing grey and normalizing.

6. The submission documents the performance of the model when tested on the captured images. The performance on the new images is compared to the accuracy results of the test set.

Ans:
The new images test accuracy is 100%, which is high than the test set accuracy of 92.9%. It should be interesting to find out what's the sign that the model is having trouble with, Due to the time constrain, I hope I can implement that next time.

7. The top five softmax probabilities of the predictions on the captured images are outputted. The submission discusses how certain or uncertain the model is of its predictions.

Ans:
The top five softmax probabilities is quite certain, the first guess is very close to 1.