Traffic Sign Recognition

1. Dataset Exploration&Summary

1.1 basic summary of the data set

I used the numpy library, and pickle.load() function to open and read the files for training and test. And then I use .shape() function and numpy to calculate summary statistics of the traffic signs data set.Here is the Data's specification:

- The size of training set is 34799
- 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

1.2 Exploratory Visualization

Based on the matplotlib library, all the pictures and their label were printed. It was for confirming the label and figure match to each other. (fig1)

Fig 1. label and figure match

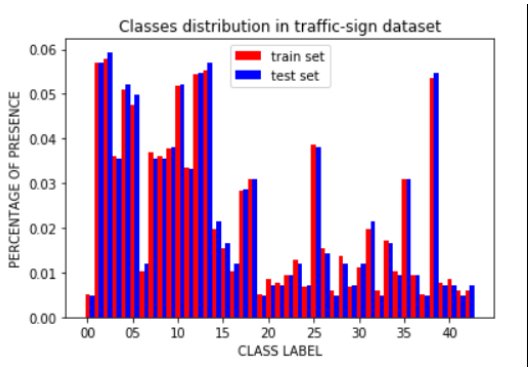

Fig 2.Classes distribution

The Fig2 is the Classes distribution in traffic-sign dataset. This was used for confirm the frequency of the each labels.

2. Design and Test a Model Architecture

2.1 Preprocessing data

The following preprocessing metheds were used: At first, I decided to convert the images to grayscale. Then I normalized the images to -1 and 1 by the recommended fomula (pixel - 128)/ 128, because it can make different features have the same scale, and the convergence can be accelerated. Lastly I use the shuffle function from sklearn.utils to disrupt the order of the data that will be trained. I think that will make my model more generalized.

2.2 Model Architecture

The training model was made based on LeNet and modified refer to the paper(Traffic Sign Recognition with Multi-Scale Convolutional Networks, Pierre Sermanet and Yann LeCun Courant Institute of Mathematical Sciences, New York University)

Pieree's paper presented a method which is very appropriate to train a model for traffic recognition. The architecture of the model is showed as Fig 3. It has 2 maxpoling layers , 3 convolution layers and 3 steps of flatten layers. They considered the a number of difficult challenges due to real world variabilities such as viewpoint variations, lighting condition(saturations, low-contrast),motion-blur, occlusions, sun glare, physical damage, colors fading, graffiti, stickers and an input resolution as low as 15x15.

The traditional ConvNet architecture was modified by feeding 1st stage features in addition to 2nd stage features to the classifier. And their experiments conducted after phase 1 produced a new record of 99.17%. This good enough to train a model for traffic sign. What's more, it has 97.33% accuracy when feed with random features. This paper provided a good solution to solve the recognizition problem of the traffic signs.

At first I tried using LeNET5. However, its performance was not good enough. I can only get 80%~90% validation accuracy. After that I used Pieree's model , the accuracy of validation step reached 96% and the test accuracy reached 94% . I thought overfitting still happened but the results was good enough to meet the specification.

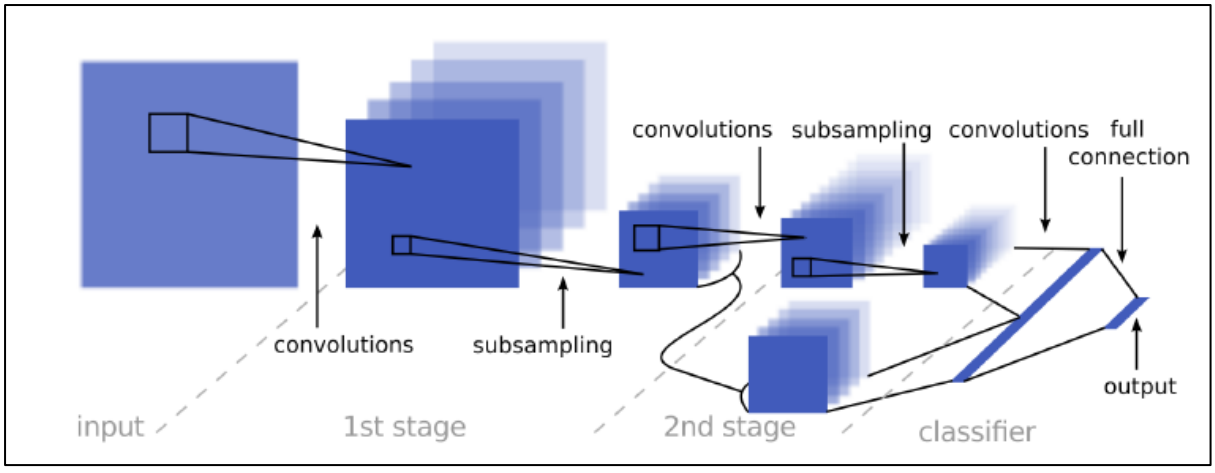

Fig 3. Architecture of multi-scale convolutional networks

- 1st layer : Input : 32x32x1, output: 28x28x6, valid padding, stride [1,1], filter size[5,5]
  - Almost same with LeNet
- Relu layer
- Max_pooling layer, stride [2,2], input: 28x28x6, output : 14x14x6
- 2nd layer: Input :14x14x6 , output : 10x10x16, filter size: [5,5], valid padding, stride[1,1]
- Relu layer
- Max_pooling layer, stride[2,2], input: 10x10x16, output: 5x5x16, filter size: [2,2], stride[2,2]
- Flatten layer(for 2nd stage which is direct to full connection), size : 400
- Convolution layer(2nd stage subsampling)
- Flatten layer(for full connection), size : 400
- Concatenate layer of full connection , 400+400= 800
- Full connection to logits output . input: 800 output: 43

2.3 Model Training

The epochs, keep_prob, learning rate and batch size was decided by many attempts. I use AdamOptimizer as many other students did. And then decreased the loss function was executed by minimize() function of the AdamOptimizer. The followings are hyperparameters that I chose:

Batch_size : 128

Epochs : 30

Learning_rate : 0.00085 (it was very sensitive. Once if the value change into 0.001 or 0.0001, the model's performance changed a lot)

2.4 Solution Approach

After training, the model's test accuracy was about 94%.

```
In [7]: with tf.Session() as sess:
            saver.restore(sess, tf.train.latest_checkpoint('.'))

            test_accuracy = evaluate(X_test, y_test)
            print("Test Accuracy = {:.2f}".format(test_accuracy))

        Test Accuracy = 0.94
```

Fig 4. Accuracy test

3. Test a Model on New Images

3.1 Acquiring New Images

I found some images for test in google by searched " german traffic signs". I choose some clear images and can be exactly labeled. So I found the images shown in Fig 5

Some of them are with blue sky and some of them are totally a standard without any background. And some of them have green background.

Fig 5 German traffic images found in google

3.2. Performance on New Images

For evaluating my model's performance with new images from download, I tested the model that I built. The images were pre-processed by some functions. The labels of my download images are described as a [1,3,14,23,25]. However, the accuracy of the trained model was only 60%.Maybe my model is overfitting.Another possible reason is that my pre-processed function is too easy.

3.3 Model Certainty - Softmax Probabilities

By using the tensorflow.nn.top_k(softmax_logits)function, I found the 'stop' and 'slippy' labels prediction was totally wrong. So the model only get 60% accuracy. It seems that my model needs more training to catch the features of images.

```
Top 5 Softmax Probabilities For Image 1 :
label: 1
probability:1.00
label: 5
probability:0.00
label: 2
probability:0.00
label: 6
probability:0.00
label: 11
probability:0.00
```

Correct

```
Top 5 Softmax Probabilities For Image 2 :
label: 3
probability:1.00
label: 2
probability:0.00
label: 5
probability:0.00
label: 6
probability:0.00
label: 1
probability:0.00
```

Correct

```
Top 5 Softmax Probabilities For Image 3 :
label: 5
probability:0.99
label: 7
probability:0.01
label: 2
probability:0.00
label: 42
probability:0.00
label: 10
probability:0.00
```

Wrong(correct label:14)

```
Top 5 Softmax Probabilities For Image 4 :
label: 11
probability:0.93
label: 31
probability:0.06
label: 18
probability:0.01
label: 21
probability:0.00
label: 27
probability:0.00
```

Wrong(correct label:23)

```
Top 5 Softmax Probabilities For Image 5 :
label: 25
probability:1.00
label: 21
probability:0.00
label: 24
probability:0.00
label: 26
probability:0.00
label: 19
probability:0.00
```

Correct

Fig 7. Softmax Probabilities of the prediction

References：

[1] Traffic Sign Recognition with Multi-Scale Convolutional Networks, Pierre Sermanet and Yann LeCun, Courant Institute of Mathematical Sciences, New York University

[2] jeremy-shannon, https://github.com/jeremy-shannon

[3] Param Aggarwal, https://medium.com/computer-car/intricacies-of-traffic-sign-classification-with-tensorflow-8f994b1c8ba

[4] MehdiSv, https://github.com/MehdiSv

[5] vxy10, https://github.com/vxy10