Traffic sign classifier

This is my solution report for Project 2 of Udacity's Self Driving Car Nanodegree.

1. Goal

- Dataset Exploration, sum basic properties of data set
- Exploratory Visualization
- Design a Model Architecture
- Test a Model Architecture
- Test a Model on New Images

2. Submission Files

- Ipython notebook with code
- HTML output of the code
- A writeup report (either pdf or markdown)

3. Basic and supported documents

- download dataset: traffic-signs-data.zip
- test new image
- signnames.csv

4. Process

step 0：load the data

three .p documents would be load

step 1: dataset summary anf exploration

print the number of training examples, testing examples, valid examples, image data shape and number of classes.

We obtain the basic dataset as figure 1

```
Number of training examples = 34799
Number of testing examples = 12630
Number of valid examples = 4410
Image data shape = (32, 32, 3)
Number of classes = 43
```

Include an exploratory visualization of the dataset

In the first time I start with something simple.

① plotting traffic sign images

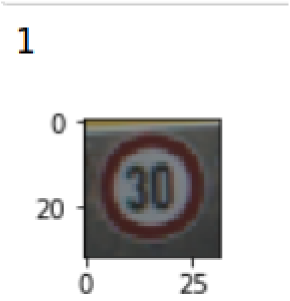

Fig 1 sample of training data

② plotting the count of each sigh

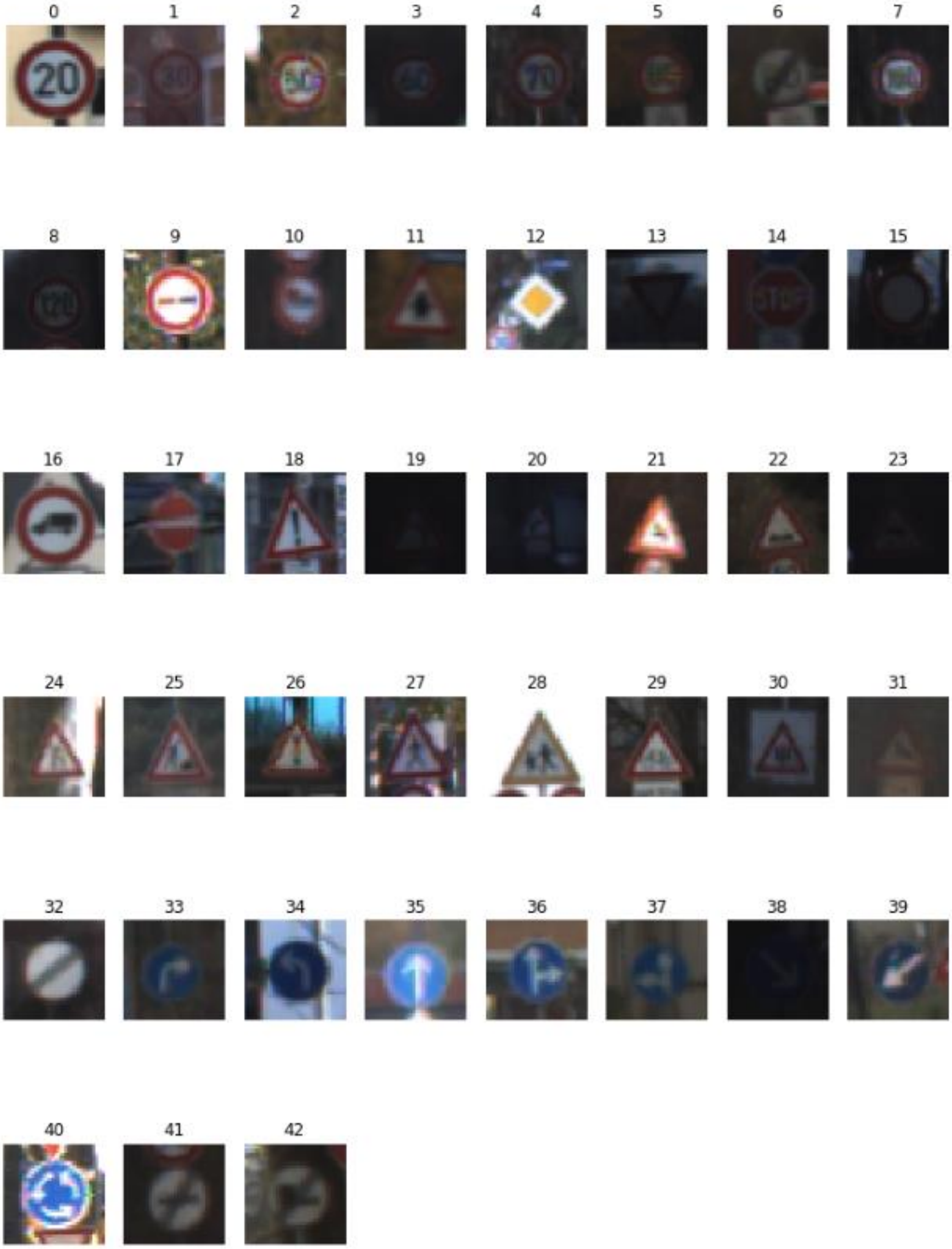

Fig 2 sample of training data for each count number

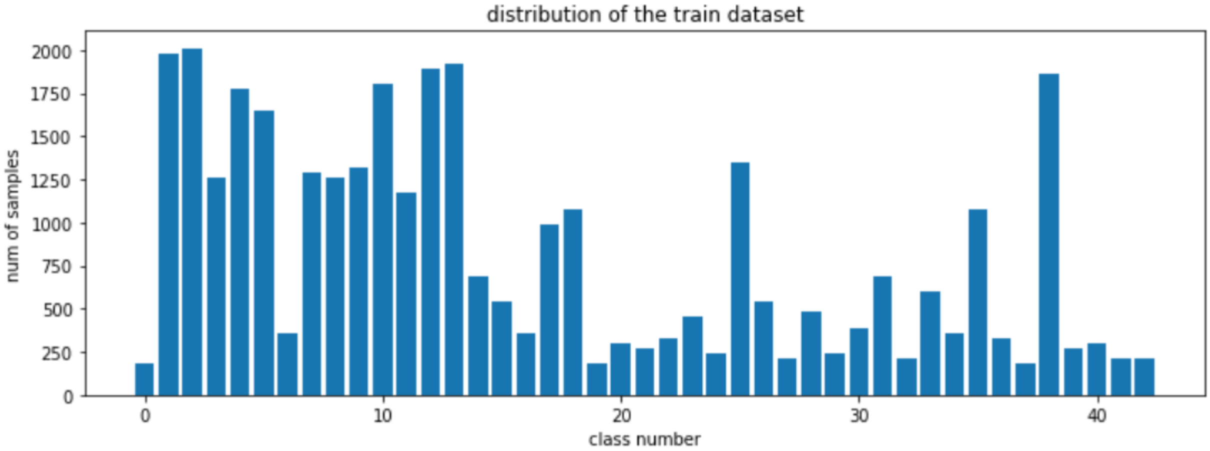

Fig 3 distribution of train dataset

step 2: design and test of the model architecture

Preprocessing

I normalized the data before training for mathematical reasons. Normalized data can make the training faster and reduce the chance of getting stuck in local optima.

Model architecture

I use a lenet5 to classify the traffic signs. The input of the network is an 32x32x3 image and the output is the probability of each of the 43 possible traffic signs.

My final model consisted of the following layers:

layer	operation	description	input	output
layer1	Convolution 5x5	1x1 stride, valid padding, RELU	32x32x3	28x28x6
	Max pooling	2x2 stride, 2x2 window	28x28x6	14x14x48
layer2	Convolution 5x5	1x1 stride, valid padding, RELU	14x14x48	10x10x16
	Max pooling	2x2 stride, 2x2 window	10x10x16	5x5x16
flatten		3 dimensions -> 1 dimension	5x5x16	400
Fully Connected		connect every neuron from layer above	400	120
Fully Connected		connect every neuron from layer above	120	84
		output = number of traffic signs in data set	84	10

Table 1 model architecture

Model training

here are my final training parameters

set epochs 30 means every batch would run 30 times

set batch size 150 means 150 samples are a batch.

Set rate 0.008 that means 0.008 is the leaning rate. In first time I set rate as 0.01 and the accuracy go up faster but limited to 0.9 while I set rate as 0.0001 the accuracy go up so slowly.

My results after training the model is

validation accuracy is equal to 0.935 and test accuracy is 0.916

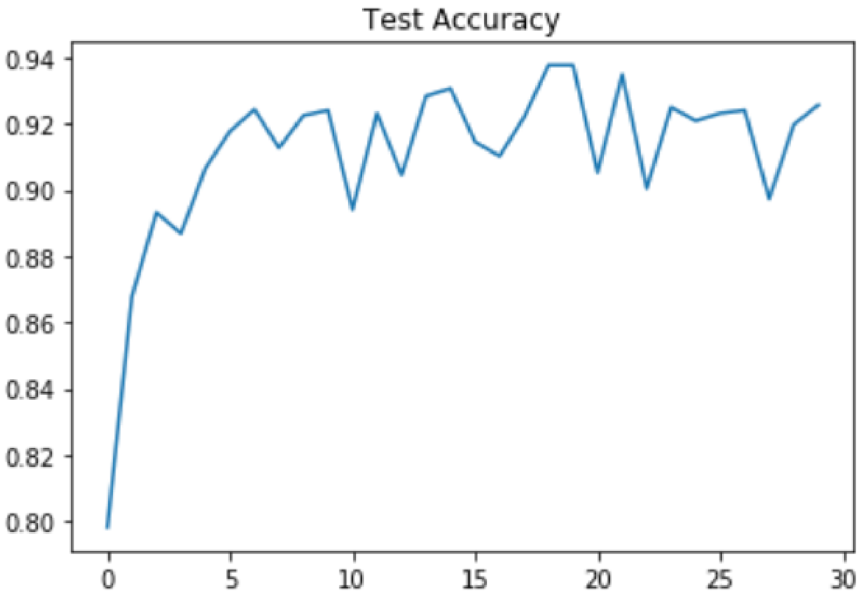

Fig 4 test accuracy with epoch ranges

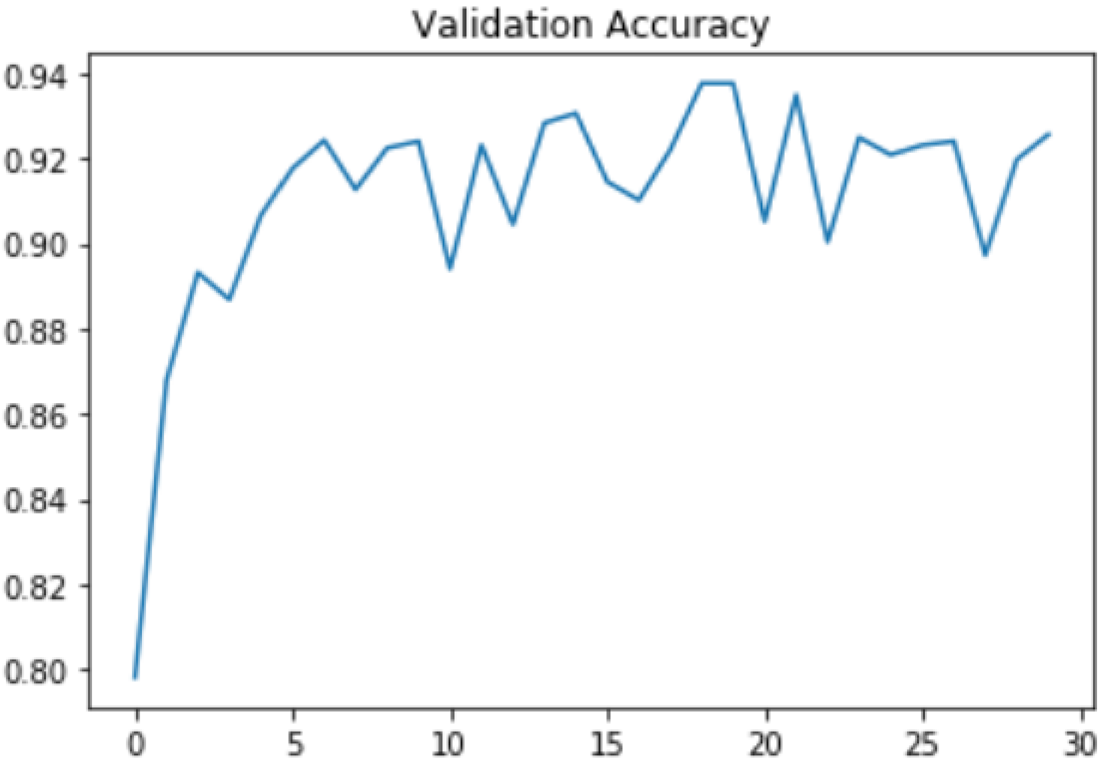

Fig 5 validation accuracy with epoch ranges step 3: test a model on new image

I used web new image for my test.

But I have trouble in softmax visualization.