Traffic Sign Recognition Project

The goals / steps of this project are the following:

1. Load the data set (see below for links to the project data set)
2. Explore, summarize and visualize the data set
3. Design, train and test a model architecture
4. Use the model to make predictions on new images
5. Analyze the softmax probabilities of the new images
6. Summarize the results with a written report

Image reference:

[image1]: ./new_image/visualization.jpg "Visualization"
[image2]: ./ new_image /grayscale.jpg "Grayscaling"
[image3]: ./ new_image / 2_image.ppm "Traffic Sign 2"
[image4]: ./ new_image / 17_image.ppm "Traffic Sign 17"
[image5]: ./ new_image / 18_image.ppm "Traffic Sign 18"
[image6]: ./ new_image / 27_image.ppm "Traffic Sign 27"
[image7]: ./ new_image / 28_image.ppm "Traffic Sign 28"

Rubric Point

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.

Here is an exploratory visualization of the data set. It is a bar chart showing how the data ...

![alt text][see jupyter notebook]

3. 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.)

1.Grayscaling
As a first step, I decided to convert the images to grayscale because grayscaling can reduce data set that is handling. So it can be faster than before. Also, for 5 new images, accuracy of correct predictoon with grayscling(correct prediction:4/5) higher than without grayscaling(correct prediction:3/5) but it is not scientifical. It is because it has only 5 image that are regarded as simple.
However, accuracy of validationf of train data set with grayscling can archieve to 97% while archieve to 96% without grayscaling. So it may less noise image with grayscling than without grayscaling.

Here is an example of a traffic sign image before and after grayscaling.

2.Normalized
After the last step, I normalized the image data. Accuracy of correct prediction with normalized(98%) is higher than without normalized(97%). All pixels are mapping into[1, -1] and it

close to zero. It can improve the convergence of training model.

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(c1)	1x1 stride, VALID padding, outputs 28X28X6
RELU	
Max pooling(s2)	2x2 stride, outputs 14x14x6
Convolution 5x5(c3)	1x1 stride, VALID padding, outputs 10X10X16
RELU	
Max pooling(s4)	2x2 stride, outputs 5x5x16 = 400
Fully connected	output 120
dropout	0.5
Fully connected	output 84
Fully connected	output 43
Softmax	

From the table, we can see the modol is simliar as the LENET model. It contains 2 Convolution and pooling layers, 3 Fully connected layers and one droptout layer. At the beginning, I add dropout layer behind each layers, as a result, the accuacy of validation always keep small and have no trend of convergence. It is because the convolutional layers usually don't have all that many parameters. So they need less regularization to begin with. Then I only add the dropout behind the first Fully connected.

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
optimizer: Adam;
BATCH_SIZE:125;
EPOCHS:30
learning rate:0.002

1. Type of optimizers:

In the tensorflow, there are many optimizers such as GradientDescentOptimizer, AdadeltaOptimizer, AdagradOptimizer,MomentumOptimizer, AdamOptimizer, RMSPropOptimizer and so on. From the comparison with different types of optimizers(From web site https://www.leiphone.com/news/201706/e0PuNeEzaXWsMPZX.html), Adagrad, Adadelta,

RMSprop and Adam are better than others in rate of convergence and learning rate
In additional, Adam that based on RMSprop add bias-correct and momentum. When the gradient become sparse, the effect of Adam is better than RMSprop. Totally, Adam is the best choices than others.

2. EPOCHS:

when EPOCHS is very small, it will affact the result of train model. Because the parameters of model need be trainned by many times until the loss and accuracy aim to a setting value(such as accuracy at least 0.93) or keep unconspicuous change. So the minimum of EPOCHS is at least 10. Also, the choices of size of EPOCHS is affacted by the size of BATCH_SIZE. The more suitable the size of BATCH_SIZE is choosed, the smaller the size of EPOCHS. In my train model, it is 30 because it can have more steps to observe the effect on result of train model.

3. BATCH_SIZE:

a. Full Batch Learning:

As size of training set is 34799 that belong to big training set, it can not suppose to be using Full Batch Learning. when I set the BATCH_SIZE is 10000, the beginning of the loss is 7.3 (the good beginning of loss is smaller than 2, even is smaller than 1).
the accuracy of validation still keep small(such as 0.05) and no obvious trend on convergence. It is because too many samples in the same batch cause too many differences, then the correct value from different layers are offseted by each other. So it is difficult to correct the parameters.

b. Online Learning:

When the size of BATCH_SIZE is set to 1, the beginning of the loss is 3.3 and the beginning of accuracy of validation is also very small(such as 0.04~0.05). Normally, the loss will become small, however, in the Online Learning ,the change of loss do not have a obvious trend(sometimes it become small, sometimes it become big).

c. Suitable scale:

From training model, when the EPOCHS is 30 and the validation set accuracy aim to at least 0.93, the scale of the suitable BATCH_SIZE is 100~150.
In my train model, 125 of BATCH_SIZE is most suitable

4. Learning rate

Learning rate has a big effect on the speed of gradient descent. When the learning rate is 0.1, it is hard to get good rate of convergence. When learning rate is very small, it happens the same situation. So a optimal learning rate that is not too big or too small must be choosed.

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.

Outline the steps:

1. Softmax and logisticregression

After getting the result from LENET model, using the function (softmax_cross_entropy_with_logits) that Computes softmax cross entropy between logits and labels. The reason why the model choose the softmax_cross_entropy_with_logits function but not choosing the softmax is that the return of softmax is predict value that may can be used in the test validataion model. However, using softmax_cross_entropy_with_logits that based on softmax compute softmax cross so that it serves for the loss fuction

2. Minimize the the loss and training the model(train model)

The result of loss function is smaller, the better it is ,. Using AdamOptimizer minimizes the loss from the result of last step by tensorflow. It updates the model parameters when the result of loss function is smaller than before.

3. Validation model

Using validation data from training set to test the accuracy of the predict result of model by tensorflow

My final model results were:

*** training set accuracy of 0.98**

*** validation set accuracy of 0.97**

*** test set accuracy of 0.98**

If an iterative approach was chosen:

* What was the first architecture that was tried and why was it chosen?
* What were some problems with the initial architecture?
* How was the architecture adjusted and why was it adjusted? Typical adjustments could include choosing a different model architecture, adding or taking away layers (pooling, dropout, convolution, etc), using an activation function or changing the activation function. One common justification for adjusting an architecture would be due to overfitting or underfitting. A high accuracy on the training set but low accuracy on the validation set indicates over fitting; a low accuracy on both sets indicates under fitting.
* Which parameters were tuned? How were they adjusted and why?
* What are some of the important design choices and why were they chosen? For example, why might a convolution layer work well with this problem? How might a dropout layer help with creating a successful model?

My first model architecture is the following:

Layer	Description

Input	32x32x3 RGB image
Convolution 5x5	1x1 stride, SAME padding, outputs 32X32X32
RELU	
Max pooling	2x2 stride, outputs 16x16x6
Convolution 5x5	1x1 stride, SAME padding, outputs 16X16X64
RELU	
Max pooling	2x2 stride, outputs 8x8x64 = 4096
Convolution 5x5	1x1 stride, SAME padding, outputs 8X8X64
RELU	
Max pooling	2x2 stride, outputs 4x4x64 = 1024
Fully connected	output 700
dropout	0.5
Fully connected	output 84
Fully connected	output 43
Softmax	

1. Pooling, dropout, convolution layers choose:

As this model architecture can recognize the image such as verification code(from web site:http://news.hiapk.com/internet/s597361a665d8.html) and its accuacy aim to 99%, I choose this model architecture to design. Of course, I change initial architecture that have 2 Fully connected to 3 Fully connected layers because the output(1024) of the third Convolution layer and Max pooling is too big. So it design 3 Fully connected layers to reduce dimensions. The second changing is that delete most of dropout layers except after the first Fully connected layer. It is because the result of loss and accuracy rate with many dropout layers are still at one value and has no convergence. So too may dropouts cause underfitting and no dropouts may cause overfitting. As a result, leaving a droppout after the first Fully connected layer.

2. SAME padding and VALID padding

SAME means that the output feature map has the same spatial dimensions as the input feature map. Zero padding is introduced to make the shapes match as needed, equally on every side of the input map.

VALID means no padding. Therefore, when the pixel in the edge of image has no affects on the classification of image, VALID padding is more suitable. On the contrary, if the pixel in the edge has a significant affect on the classification of image, SAME padding is more suitable. For the traffic sign application, VALID padding is better than SAME padding

If a well known architecture was chosen:

* What architecture was chosen?
* Why did you believe it would be relevant to the traffic sign application?
* How does the final model's accuracy on the training, validation and test set provide evidence that the model is working well?

The reason why LENET moel choose in final :

Compare above model architecture with LENET5 model, the final accuracy of LENET5 model and above model architecture are 98% and 97%,respectively. At the meaning time, training model for LENET5 spend less time on the the traffic sign application than above model architecture. Therefore, for the the traffic sign, LENET5 is more suitable.

4.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:

There are 5 German traffic signs(int the new_image directory) in the flowing:
![alt text][image3] ![alt text][image4] ![alt text][image5]
![alt text][image6] ![alt text][image7]

Obviously, these 5 images are not very clear. As the accuacy of validation of this model can be about 98%, it can classfy most of images. So I am intentional to choose fuzzy images to verify.

The number of name of image means the sign number. For example, 17_image.ppm means NO entry(sign number is 17).
From the 5 images, we can see 18_image, 27_image and 28_image are difficult to classfy than others. 18_image is too dark so that it cant be not seen. 28_image and 27_image are a little fuzzy and noise.

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
Children crossing(28_image)	Children crossing(28)
Pedestrians(27_image)	General caution(18 wrong)
No entry(17_image)	No entry(17)
General caution(18_image)	General caution(18)

| 50km/h(2_image) | 50km/h(2) |

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 98% is a little lower. As I said in the preprocessed, this comparison is not scientific. One reason is that 5 images is a versy small sample. Secondly, I am intentional to choose these fuzzy images to verify.

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 Children crossing and do not have other traffic signs.

Probability	Prediction
1	Children crossing

For the second image the model is 56% sure that this is a Right-of-way at the next intersection. Howerver, the correct traffic sign is Pedestrians. As I said before, this image has a bit noise. Other probabilities are Speed limit (20km/h)(28.9%), General caution(14.6%), Go straight or left(0.3%), Keep left(0.2%).

Probability	Prediction
0.559	Right-of-way at the next intersection
0.289	Speed limit (20km/h)
0.146	General caution
0.003	Go straight or left
0.002	Keep left

For the third image, the model is 100% sure that this is a No entry sign and do not have other traffic signs.

Probability	Prediction
1	No entry

For the forth image the model is 99.7%% sure that this is a General caution and the next is Traffic signals(0.2%)

Probability	Prediction
0.997	General caution
0.002	Traffic signals

For the fifth image, the model is 100% sure that this is a Speed limit (50km/h) and do not have other traffic signs.

Probability	Prediction
1	Speed limit (50km/h)