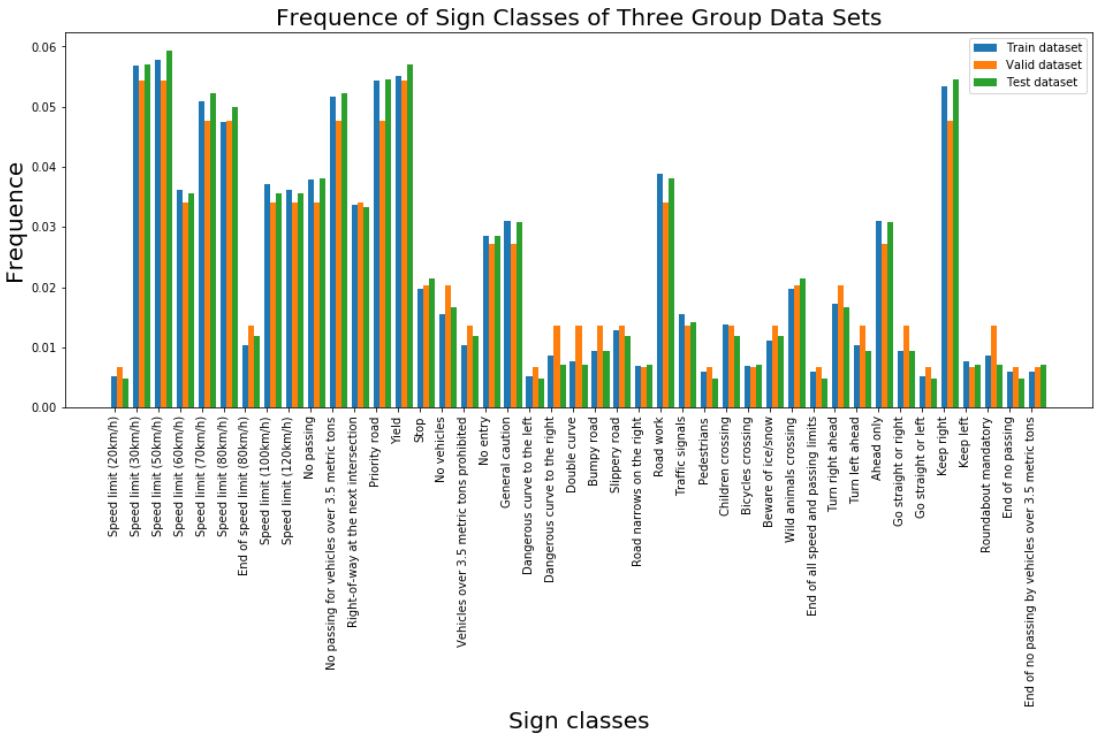
1. Data Set Summary & Exploration

1.Provide a basic summary of the data set.

I used the python 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 the frequency of of every single traffic sign of three datasets. The table show that three datasets have the similar distribution.

2. Design and Test a Model Architecture

1. preprocessed the image data

Considered that traffic signs have different colors and the colors may provide some information to CNN,I didn't grayscale the images.
I normalized the data because this will improve the performance of CNN.
So, the difference between the original data set and the augmented data set is augmented data got normalized.

2. Describe the final model architecture looks like

Layer	Description
Input	32*32*3 RGB Image
Conv 5*5	1x1 stride, valid padding, outputs 28*28*18
RELU	
Max pooling 2*2	2*2 stride, valid padding, output 14*14*18
Conv 5*5	1*1 stride, valid padding, outputs 10*10*54
RELU	
Max pooling 2*2	2*2 stride, valid padding, outputs 5*5*54
Fully connected	Inputs 1350, outputs 800
RELU	
Dropout	
Fully connected	Inputs 800, outputs 300
RELU	
Dropout	
Fully connected	Inputs 300, outputs 43
SoftMax	

3. Describe how you trained your model.

* Optimizer. Adam optimization combines Momentum and RMSprop algorithm, and it performance good on different architectures. So I used that.
* Batches. Considered train speed and computer ability, I chose 128.
* Epochs. Train data set is big, so I set epochs 20.But truth is the net work have a good performance earlier, so I set it 5.
* Learning rate. I chose a normal value 0.001 and try many values like 0.01, 0.005,0.0001. Finally I set it 0.001.

3.How Train

I use AdamOptimizer.
After I tried several times,I set
learning_rate 0.001
epochs = 5
batch_size = 128

4. Describe the approach taken for finding a solution and getting the validation set accuracy to be at least 0.93.

My final model results were:
* training set accuracy of 99.7%
* validation set accuracy of 94.2%
* test set accuracy of 93.6%

*** What was the first architecture that was tried and why was it chosen?**
In my first architecture, the conv1 had depth of 6 and the conv2 had depth of 18.

*** What were some problems with the initial architecture?**
The architecture did performance good enough.

*** 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.**
My architecture has two conv layers and two fully connected layers just like LeNet. The difference is that I set the depth of input layer 3 so the input information is much more. To use the input information I set the conv layers deeper.

*** Which parameters were tuned? How were they adjusted and why?**
I thought I'd set Epochs bigger because I set the architecture more complicated. The truth is this architecture is good at learning, so I didn't change it.

*** 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?**
Same traffic signs have same colors and shapes, so a convolution layer work well with this problem.

3. Test a Model on New Images

1. Choose five German traffic signs found on the web and provide them in the report.

Here are five German traffic signs that I found on the web:




Picture No1, No3, No4 have Higher brightness than train dataset. The position of the sign in No5 is on the bottom. The sign in No4 has a little shape transformation. These factors may confuse the neural network.

2. Here are the results of the prediction:

Image	Prediction
No passing	No passing
Priority road	Priority road
Road work	Road work
Speed limit (70km/h)	Speed limit (70km/h)
Wild animals crossing	Wild animals crossing

The model was able to correctly guess all the traffic signs, which gives an accuracy of 100%.

I normalize image data before use the neural network, so the network can handle with brightness change.
CNN has a advantage that translation invariance, so Picture.5 get right prediction.
Picture4 has a little shape transformation. As the degree of transformation is minor and three channels image can provides more information to the network than LeNet, my network get the right prediction. But I'm not sure exactly I got the right explaination.

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.

Here are the top5 sofmax probabilities:

```
TopKV2(values=array([[1.0000000e+00, 4.4054065e-25, 4.4410720e-27, 4.0130622e-27,
        1.0942888e-27],
       [1.0000000e+00, 0.0000000e+00, 0.0000000e+00, 0.0000000e+00,
        0.0000000e+00],
       [1.0000000e+00, 1.5696413e-16, 2.0515417e-19, 1.6381097e-20,
        1.5863595e-21],
       [1.0000000e+00, 2.1992904e-24, 1.8941058e-25, 0.0000000e+00,
        0.0000000e+00],
       [9.9999630e-01, 3.7305508e-06, 1.4658128e-08, 2.0837981e-11,
        1.7471168e-14]], dtype=float32), indices=array([[ 9, 10, 20,  2, 41],
       [12,  0,  1,  2,  3],
       [25, 29, 24,  5, 31],
       [ 4,  1,  0,  2,  3],
       [31, 25, 11, 27, 24]]))
```

Obviously, my model almost 100% certain the predict result.