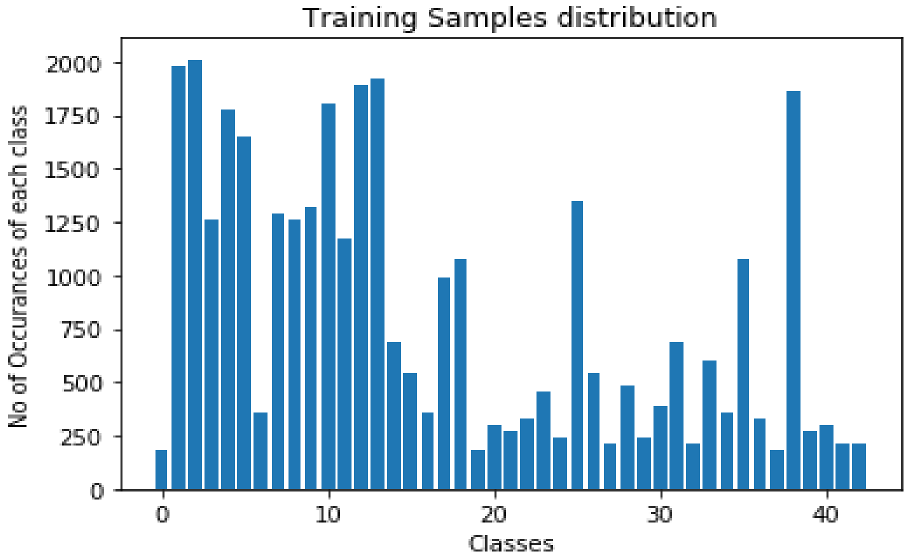
Traffic Sign Classifier

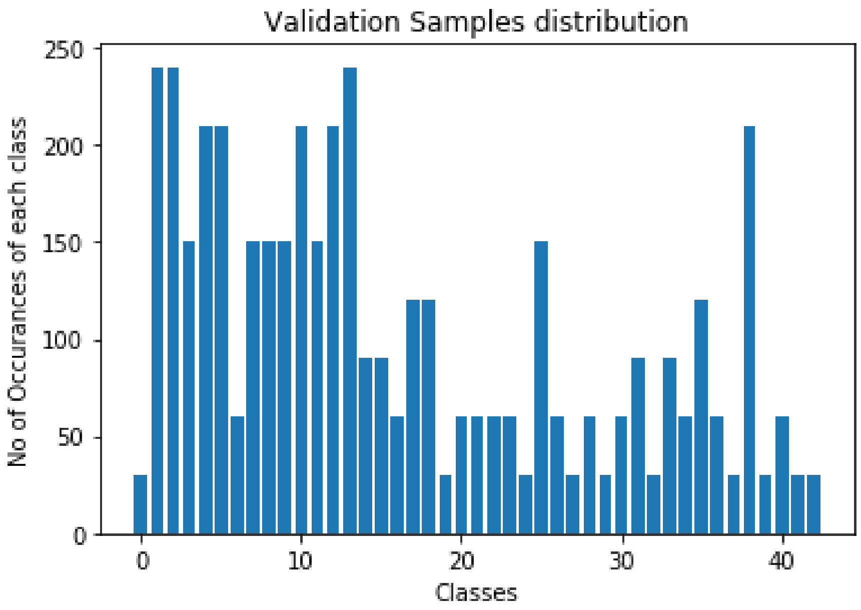
Data Set Summary & Exploration

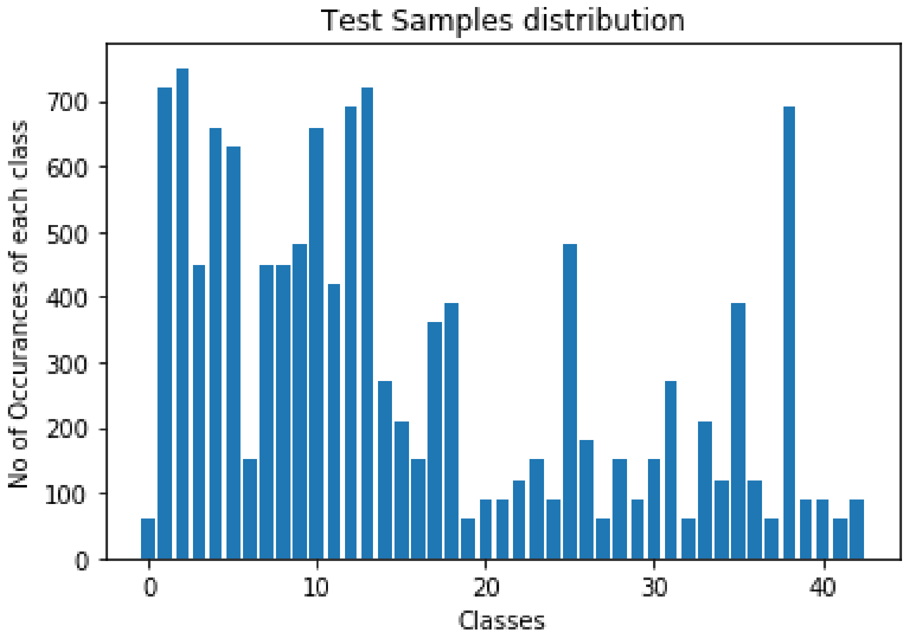
After loading the data sets, explored them to know how many features we have, the size of the image and number of examples provided.

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

Here is an exploratory visualization of the data set



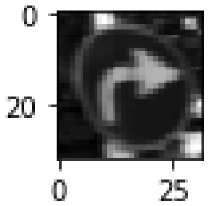




Design and Test a Model Architecture

Data preprocessing

I converted them to gray scale, then compute the mean and the variance to make training the network later easier, and improve performance.



Model architecture

I used the LeNet model but added dropout layer, the first one after activation of convolution layer 2

conv2 = tf.nn.dropout(conv2, keep_prob1)

The second one after the fully connected layer, fc1

fc1 = tf.nn.dropout(fc1, keep_prob1)
The third one after 2nd fully connected layer
fc2 = tf.nn.dropout(fc2, keep_prob2)

I used regularization concept in the architecture to avoid over fitting

regularizers = tf.nn.l2_loss(conv1_W)
+tf.nn.l2_loss(conv2_W)+tf.nn.l2_loss(fc1_W)+tf.nn.l2_loss(fc2_W)+tf.nn.l2_loss(fc3_W)

and used beta = 0.001

loss_operation = tf.reduce_mean(cross_entropy + beta*regularizers)

Layers	Description
Input	32x32x1 gray images
Conv 5x5	1x1 stride, valid padding, output 28x28x6
Relu	
Pooling	2x2 stride, output 14x14x6
Conv 5x5	1x1 stride, same padding, output 10x10x16
Relu	
Drop-out	keep_prob1 = 0.8
Pooling	2x2 stride, 5x5x16
Fully connected	120 layer
Relu	
Drop-out	keep_prob1 = 0.8
Fully connected	120 layer
Relu	
Drop out	Keep_prob2 = 0.5
Output	43

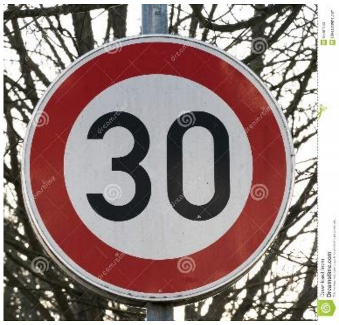
```
I used learning rate = 0.001, beta = 0.001, used adam optimizer, EPOCHS = 20
BATCH_SIZE = 150, I added keep_prob 1 and keep prob2 to avoid over fitting, I tried diff
erent combinations then finally choose 0.8 and 0.5
```

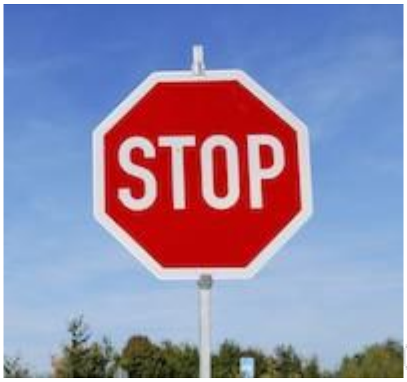
My final model results were:

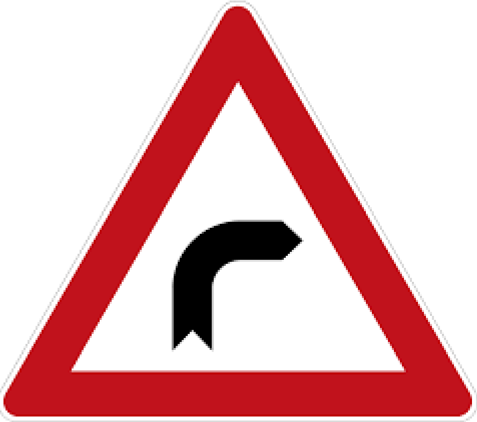
- validation set accuracy of 0.951
- test set accuracy of 0.933

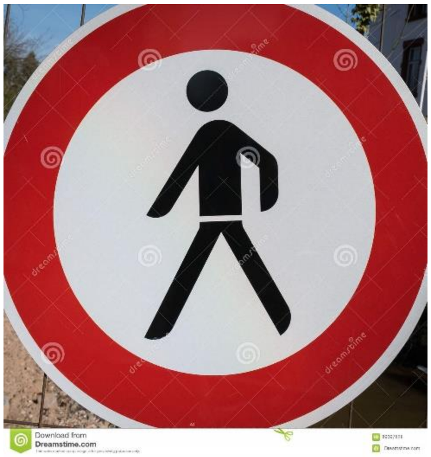
Test a Model on New Images

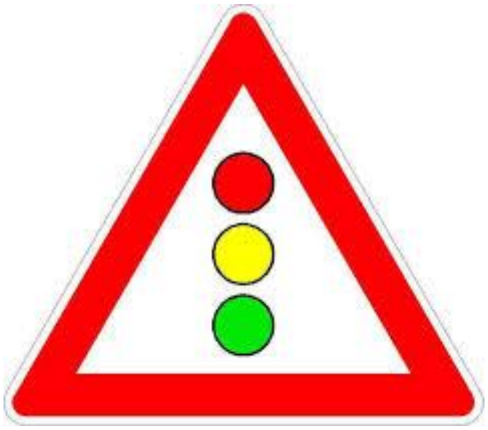
Here are five German traffic signs that I found on the web:











I resized the images to 32x32 and applied the same preprocessing, then I run the model the accuracy is 0.6

Image	prediction

Speed limit (30km/h)	Speed limit (30km/h)
Stop	Stop
pedestrian	pedestrian
Traffic signal	General caution
Dangerous curve to right	Dangerous curve to right

Output Top 5 Softmax Probabilities For Each Image Found on the Web

Here below the probability for each image among the 5 provided :

Each row represent image and each value is the probability so for example the first image

```
24.154957 , 19.865532 , 14.743233 , 14.656722 , 13.933894,
20, 23, 27, 30, 41
```

Reflection:I believe accuracy could be improved if we increased the number of the training set using Data augmentation