Project Writeup

In this project, knowledge of deep neural networks and convolutional neural networks are applied in designing a neural network to classify traffic signs. This neural network is trained and validated on German Traffic sign dataset. Implementation code of this project is available **Traffic_Sign_Classifier.ipynb.** This Project comprises of below steps

* Load the data set
* Explore, summarize and visualize the data set
* Data Augmentation and Preprocessing
* Design, train and test a model architecture
* Use the model to make predictions on new images
* Analyze the softmax probabilities of the new images

#1 Load the data set

German traffic dataset is used for validation and testing of neural network. This dataset is available at http://benchmark.ini.rub.de/?section=gtsrb&subsection=dataset. Input dataset is split into 3 parts (training data, validation data and test data). These 3 datasets are provided to user in picked files (test.p,training.p,valid.p). These picked files are loaded into global variables and their sizes are read. Below is the output of first section. Training set consists of 34799 images and test set consists of 12360 images. Each image is color thumbnail with dimensions of 32 * 32 pixels. There are 43 unique traffic signs.

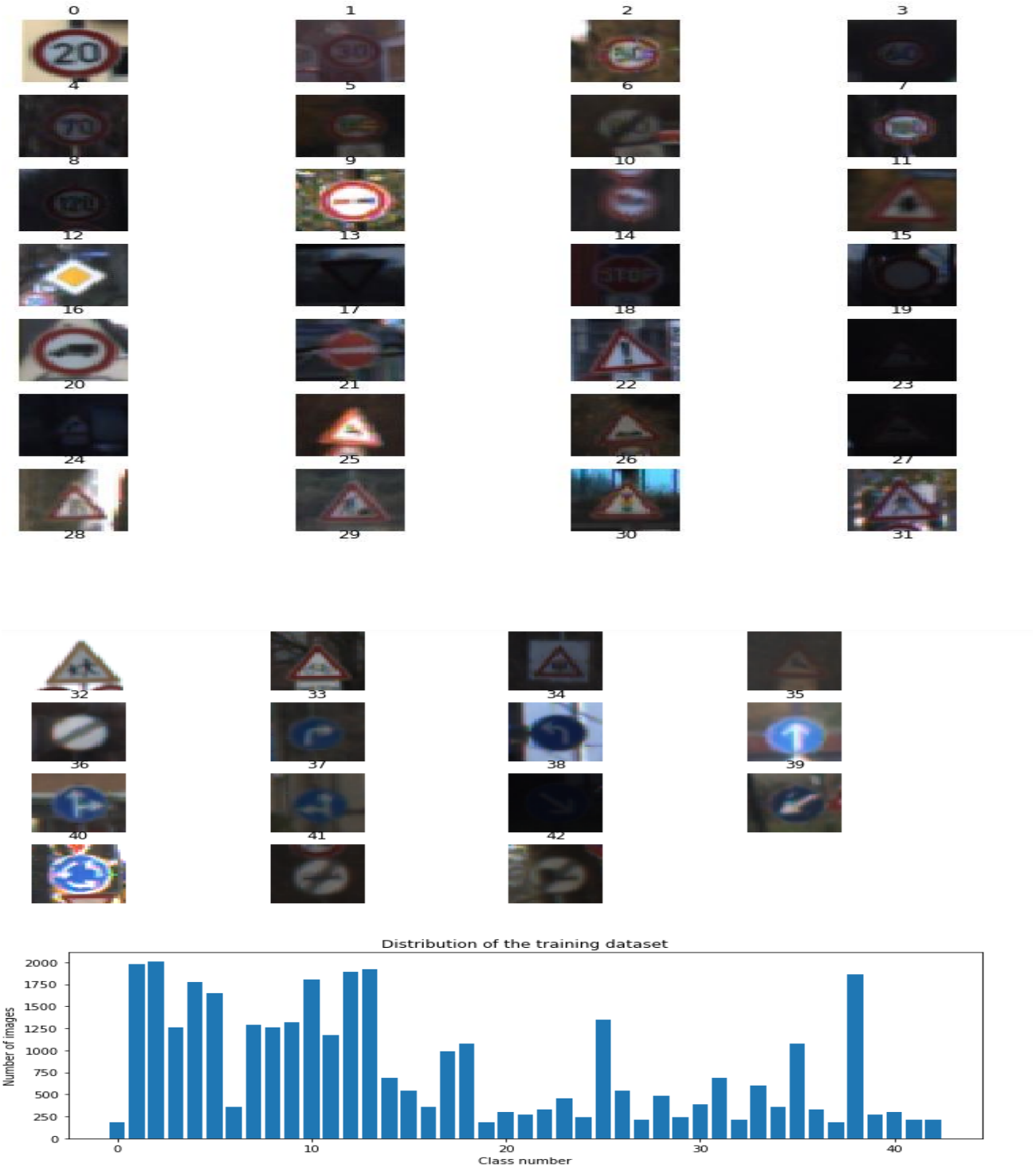
```
Output
Training size:- 34799
Test size:- 12630
Image shape:- (32, 32, 3)
Classes:- 43
```

#2 Dataset Summary & Exploration

In section2 of **Traffic_Sign_Classifier.ipynb**, input dataset is analyzed in detail. Below is the output. We can see that training dataset is an unbalanced dataset. There are very few samples for some traffic signs (max number of training samples = 2010 and min number of training samples for a sign is 180)

```
Output
Number of training examples = 34799
Number of testing examples = 12630
Image data shape = (32, 32, 3)
Number of unique classes = 43
number of training samples per class = [ 180 1980 2010 1260 1770 1650  360
1290 1260 1320 1800 1170 1890 1920  690
  540  360  990 1080  180  300  270  330  450  240 1350  540  210  480  240
  390  690  210  599  360 1080  330  180 1860  270  300  210  210]
Max Number of training samples in a class = 2010
```

In section3, all 43 unique traffic signs are plotted. Histogram of number of samples of each traffic sign is plotted as well.



#4 Data Augmentation and Preprocessing.

There are 2 problems with existing training set. First training set is too small. This can result in network overfitting this small training set. Secondly, training set is highly unbalanced. Min to Max ratio of number of samples per traffic sign is almost 9. it almost 9.

During testing of neural network, it was found that increasing training set and balancing the training set resulted in significant improvement of validation accuracy. Plain training set of 34000 gave a validation accuracy of 89% and training set of 200000 images gave a validation accuracy of 96%. First, library of helper functions (#Histogram Equalization, normalize(), rotate_img(),sharpen_img(),brighten_img()) are defined in step4. These functions are used to generate additional data from existing dataset in step5.

In step5, additional data is generated by rotating the image with small angles ranging from -3 to +15 radians. Larger angle rotation(ex:- 90, 120) was not done as they resulted in images of different signs. Min/Max ratio of each sign is used for equalizing the dataset. This algorithm is coded in step5.

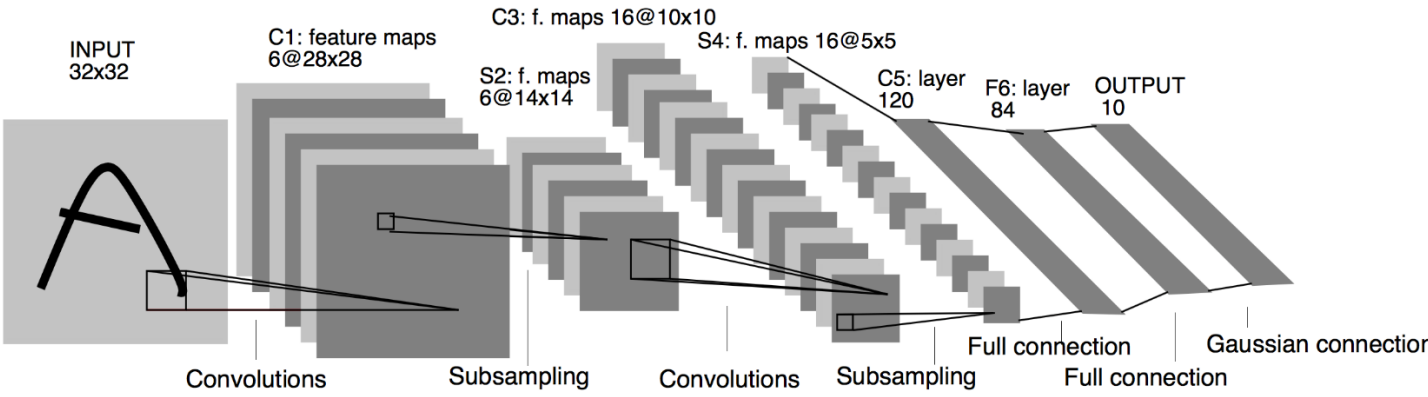
Below is the output of step5

```
Generating additional data from training set.
Data augmentation complete
Number of training examples after data augmentation = 106646
number of training samples per class = [2160 3960 4020 2520 3540 3300 2160
2580 2520 2640 3600 2340 3780 3840 1380
 2160 2160 1980 2160 2160 2100 2160 2310 2250 2160 2700 2160 2100 2400 2160
 2340 1380 2100 2396 2160 2160 2310 2160 3720 2160 2100 2100 2100]
Max Number of training sample in a class = 4020
```

In step6, preprocessing of the images is done. First, Histogram equalization is done on the images to improve the contrast of images and then pixel values of images are normalized to a mean of 0. Normalization helps with gradient descent and avoids local maxima problems. In testing of the project, Normalization step improved validation accuracy by 4 percent.

#5 Model Architecture & Test results

Modified version of LeNet model is used for traffic sign classification. LeNet is a convolutional neural network designed to recognize visual patterns directly from pixel images with minimal preprocessing.



Our model is adapted from the LeNet as follows.

- The inputs are 32x32 (RGB - 3 channels) images
- The activation function is ReLU except for the output layer which uses Softmax
- The output has 43 classes
- Dropouts are added to avoid overfitting the training data and to increase redundancy in the model. They improved validation accuracy by 3% from 94 to 97%.

Layer	Shape
Input	32 * 32 * 3
Convolutions (5*5*6, valid)	28*28*6
Max Pooling (2*2, valid)	14* 14* 6
Activation (ReLU)	14* 14* 6
Dropout (keep_prob = 0.7 for training)	14* 14* 6
Convolutions (5*5*16, valid)	10*10 *16
Max Pooling (2*2, valid)	5* 5*16
Activation (ReLU)	5* 5*16
Dropout (keep_prob = 0.7 for training)	5* 5*16
Flatten	400
Dense	120
Activation (ReLU)	120
Dropout (keep_prob = 0.7 for training)	120
Dense	84
Activation (ReLU)	84
Dropout (keep_prob = 0.7 for training)	84
Dense	10
Activation (Softmax)	10

Below selection of hyper parameters were used and below is the output of training stage.

Hyper parameters
EPOCHS = 30
BATCH_SIZE = 128
rate = 0.001

Validation accuracy: - 97%

Output log

```
Training...

EPOCH 1 ...
Validation Accuracy = 0.836

EPOCH 2 ...
Validation Accuracy = 0.903

EPOCH 3 ...
Validation Accuracy = 0.930

EPOCH 4 ...
Validation Accuracy = 0.950

EPOCH 5 ...
Validation Accuracy = 0.949

EPOCH 6 ...
Validation Accuracy = 0.956

EPOCH 7 ...
Validation Accuracy = 0.955

EPOCH 8 ...
Validation Accuracy = 0.962

EPOCH 9 ...
Validation Accuracy = 0.965

EPOCH 10 ...
Validation Accuracy = 0.961

EPOCH 11 ...
Validation Accuracy = 0.965

EPOCH 12 ...
Validation Accuracy = 0.964

EPOCH 13 ...
Validation Accuracy = 0.961

EPOCH 14 ...
Validation Accuracy = 0.963
```

```
EPOCH 15 ...
Validation Accuracy = 0.963

EPOCH 16 ...
Validation Accuracy = 0.961

EPOCH 17 ...
Validation Accuracy = 0.963

EPOCH 18 ...
Validation Accuracy = 0.970

EPOCH 19 ...
Validation Accuracy = 0.955

EPOCH 20 ...
Validation Accuracy = 0.969

EPOCH 21 ...
Validation Accuracy = 0.971

EPOCH 22 ...
Validation Accuracy = 0.968

EPOCH 23 ...
Validation Accuracy = 0.969

EPOCH 24 ...
Validation Accuracy = 0.964

EPOCH 25 ...
Validation Accuracy = 0.966

EPOCH 26 ...
Validation Accuracy = 0.964

EPOCH 27 ...
Validation Accuracy = 0.967

EPOCH 28 ...
Validation Accuracy = 0.970

EPOCH 29 ...
Validation Accuracy = 0.967

EPOCH 30 ...
Validation Accuracy = 0.970

Model saved
```

Test set accuracy

Once the model is finalized, it is finally verified with test dataset. On test dataset, this model achieved an accuracy of 94.3%. Below is the snippet of the Ipython codecell

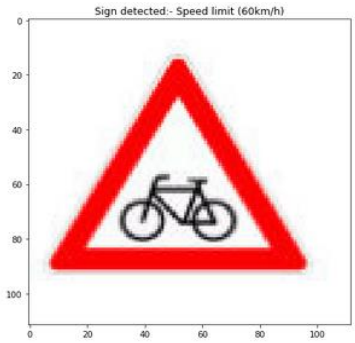
Step3.1 Run the model on test set

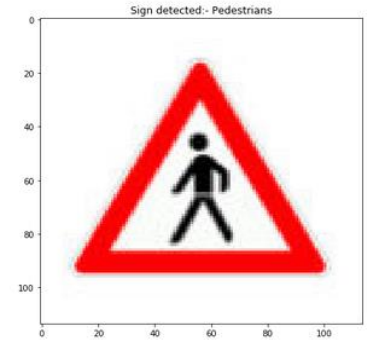
```
### Train your model here.
### Calculate and report the accuracy on the training and validation set.
### Once a final model architecture is selected,
### the accuracy on the test set should be calculated and reported as well.
### Feel free to use as many code cells as needed.

with tf.Session() as sess:
    saver.restore(sess, tf.train.latest_checkpoint('.'))

    test_accuracy = evaluate(X_test, y_test)
    print("Test Accuracy = {:.3f}".format(test_accuracy))
```

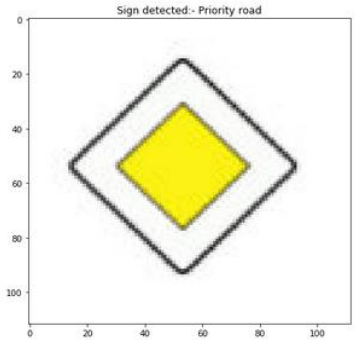
```
Test Accuracy = 0.943
```

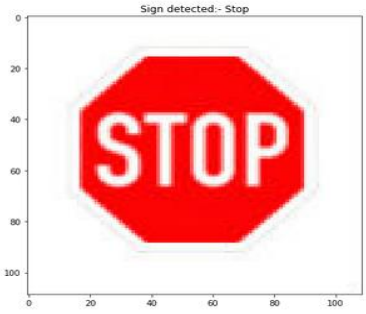
#6 : Test a Model on New Images

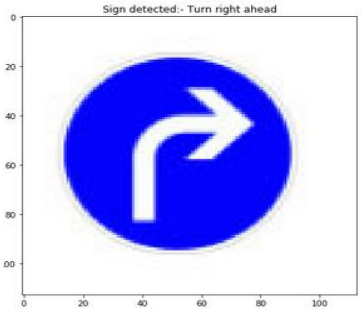
8 high resolution German traffic signs were downloaded from internet to verify the model on new images. These images are converted into thumbnail images and then preprocessed with histogram equalization and normalization algorithm. Below is the output of the model with these images. All 8 images were successfully classified resulting in 100% accuracy.

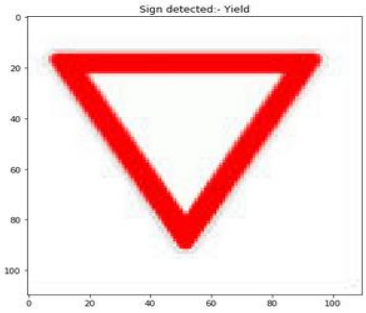






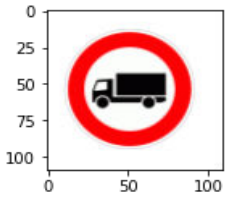


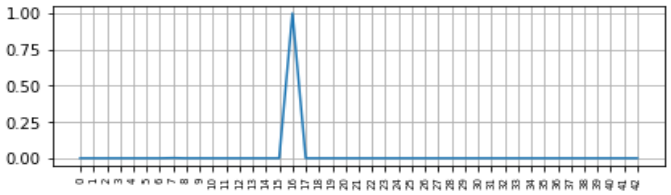




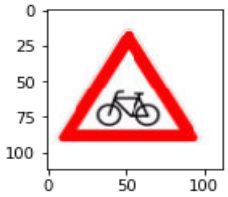
#7: Output Top 5 Softmax Probabilities For Each Image Found on the Web

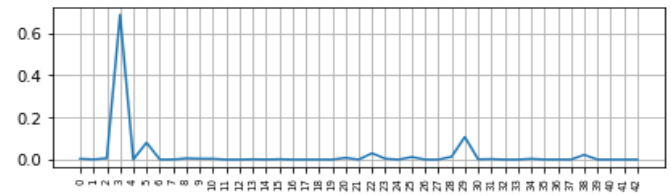
Below is the output of top5 SoftMax probabilities for 8 images downloaded from the internet.



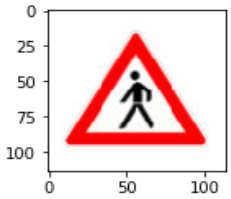


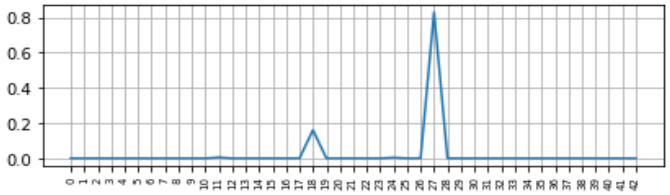
```
Top 5 probabilities predicted by softmax algoritm along with their values
Index number: 0 , Probability value: 0.9977023005485535, Sign label index : 16
Index number: 1 , Probability value: 0.0019752460066229105, Sign label index : 7
Index number: 2 , Probability value: 0.00019305653404444456, Sign label index : 9
Index number: 3 , Probability value: 8.244316268246621e-05, Sign label index : 5
Index number: 4 , Probability value: 2.7710313588613644e-05, Sign label index : 8
```



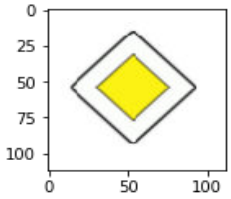


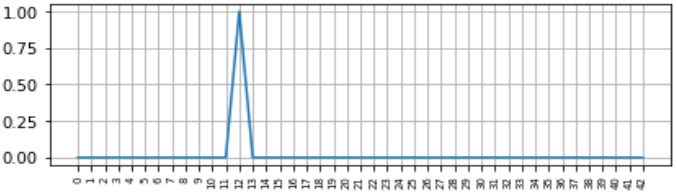
```
Top 5 probabilities predicted by softmax algoritm along with their values
Index number: 0 , Probability value: 0.6874043345451355, Sign label index : 3
Index number: 1 , Probability value: 0.10716386139392853, Sign label index : 29
Index number: 2 , Probability value: 0.08015187084674835, Sign label index : 5
Index number: 3 , Probability value: 0.029604023322463036, Sign label index : 22
Index number: 4 , Probability value: 0.022121185436844826, Sign label index : 38
```



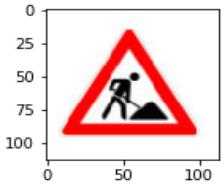


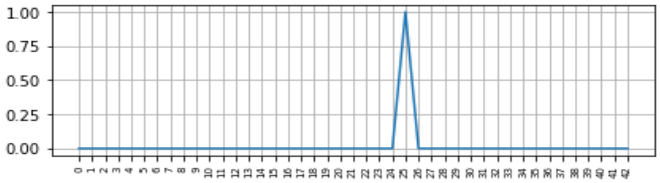
```
Top 5 probabilities predicted by softmax algoritm along with their values
Index number: 0 , Probability value: 0.8299410343170166, Sign label index : 27
Index number: 1 , Probability value: 0.16052211821079254, Sign label index : 18
Index number: 2 , Probability value: 0.005286495666950941, Sign label index : 11
Index number: 3 , Probability value: 0.004067474976181984, Sign label index : 24
Index number: 4 , Probability value: 0.00012029030040139332, Sign label index : 28
```



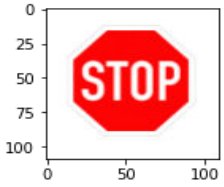


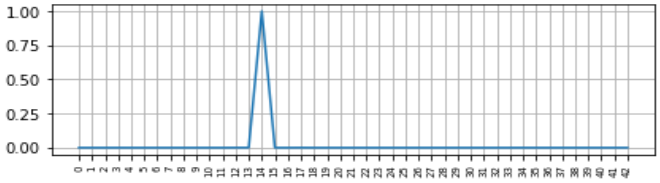
```
Top 5 probabilities predicted by softmax algoritm along with their values
Index number: 0 , Probability value: 0.9998452663421631, Sign label index : 12
Index number: 1 , Probability value: 7.035648013697937e-05, Sign label index : 13
Index number: 2 , Probability value: 4.9918864533538e-05, Sign label index : 10
Index number: 3 , Probability value: 1.0065809874504339e-05, Sign label index : 17
Index number: 4 , Probability value: 4.00716089643538e-06, Sign label index : 5
```



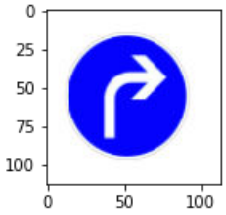


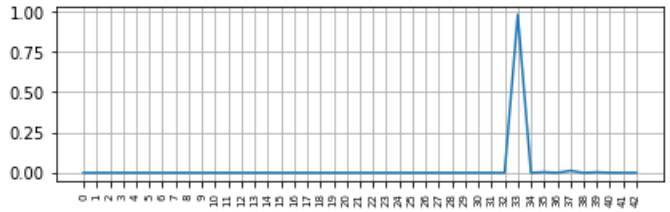
```
Top 5 probabilities predicted by softmax algoritm along with their values
Index number: 0 , Probability value: 0.9999734163284302, Sign label index : 25
Index number: 1 , Probability value: 7.514418939535972e-06, Sign label index : 31
Index number: 2 , Probability value: 4.634826836991124e-06, Sign label index : 5
Index number: 3 , Probability value: 4.385459305922268e-06, Sign label index : 30
Index number: 4 , Probability value: 3.8253133425314445e-06, Sign label index : 20
```



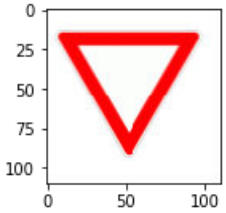


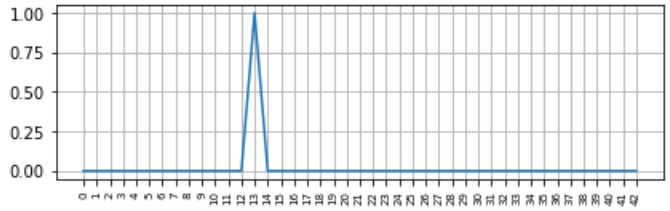
```
Top 5 probabilities predicted by softmax algoritm along with their values
Index number: 0 , Probability value: 0.9985758066177368, Sign label index : 14
Index number: 1 , Probability value: 0.00040239302325062454, Sign label index : 12
Index number: 2 , Probability value: 0.00038515363121405244, Sign label index : 17
Index number: 3 , Probability value: 0.0003402652801014483, Sign label index : 13
Index number: 4 , Probability value: 0.00014899863163009286, Sign label index : 34
```





```
Top 5 probabilities predicted by softmax algoritm along with their values
Index number: 0 , Probability value: 0.9791160225868225, Sign label index : 33
Index number: 1 , Probability value: 0.012669154442846775, Sign label index : 37
Index number: 2 , Probability value: 0.004086029715836048, Sign label index : 35
Index number: 3 , Probability value: 0.0035606676246970 89, Sign label index : 39
Index number: 4 , Probability value: 0.0005167543422430754, Sign label index : 40
```





```
Top 5 probabilities predicted by softmax algoritm along with their values
Index number: 0 , Probability value: 0.999711811542511, Sign label index : 13
Index number: 1 , Probability value: 0.0001898791961139068, Sign label index : 34
Index number: 2 , Probability value: 3.0048810003791004e-05, Sign label index : 12
Index number: 3 , Probability value: 2.407706415397115e-05, Sign label index : 38
Index number: 4 , Probability value: 2.3784306904417463e-05, Sign label index : 9
```