TRAFFIC SIGN CLASSIFIER

Introduction

This project used German traffic signs dataset to train a model that classifies German Traffic Signs. The model is a modified LeNet Neural Network architecture.

The project steps include the following:

- Load the datasets
- Explore, summarize and visualize the datasets
- Design, train and test of the model architecture
- Use the model to make inference on the new images acquired from the web.

1. Explore, summarize and visualize the datasets

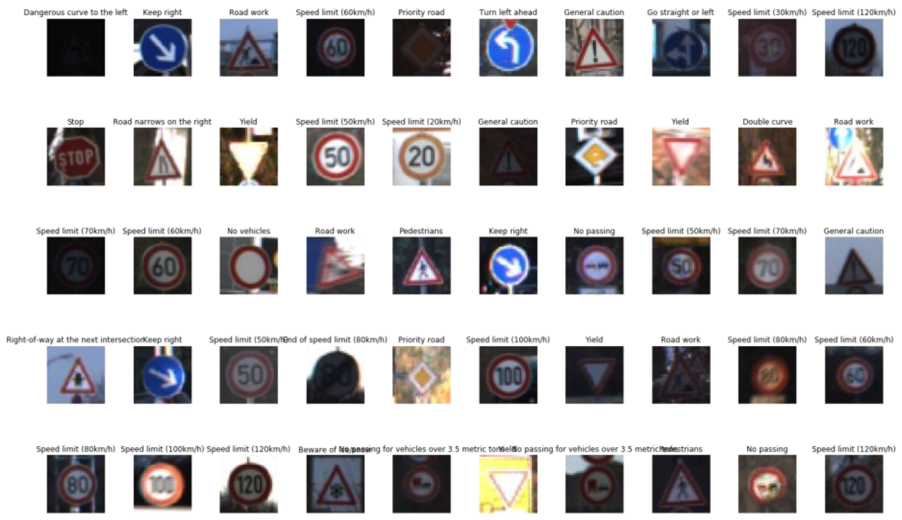
1.1 **Dataset Summary**

Python and numpy library were used to calculate summary statistics of the traffic sign dataset. The dataset is split into training, validation and test sets. Below is the summary statistics:

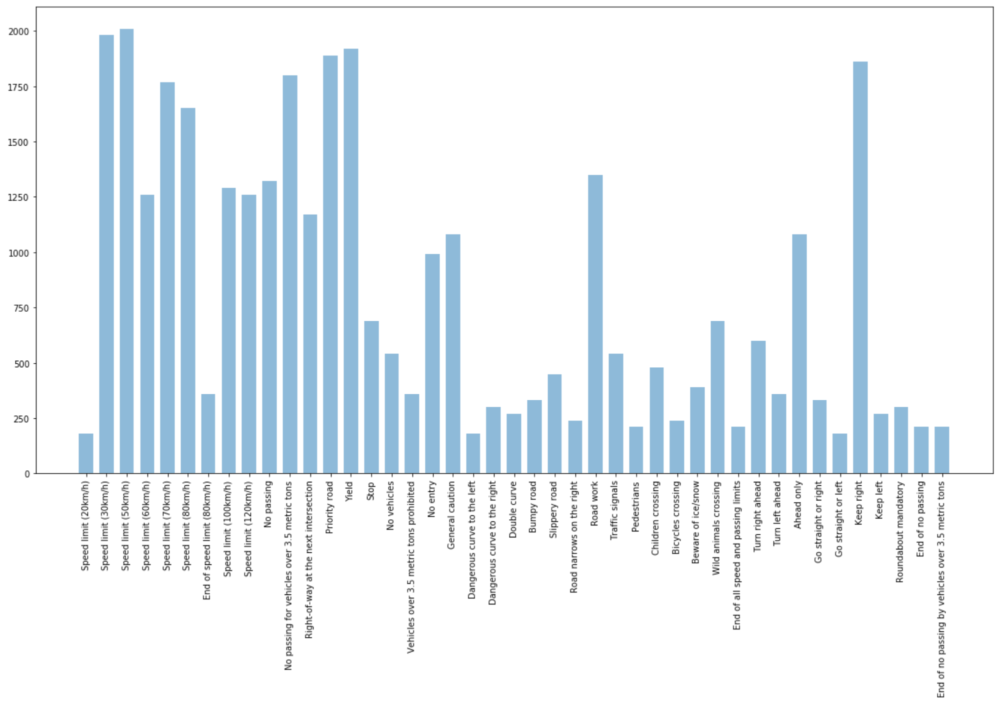
- Images are 32x32x3
- Training set contains 34,799 images
- Validation set contains 4,410 images
- Test set contains 2,630 image
- There are 43 classes for the images.

2.2 Exploratory **Dataset Visualization**

Below are the samples images from the training set.



The bar chart below shows the distribution of the training images across classes. There is significant imbalance in the dataset as some images are in hundreds while others are in thousand. This could result in the model being biased toward over represented classes. However, the data imbalance can be mitigated by augmenting under represented classes using data augmentation.



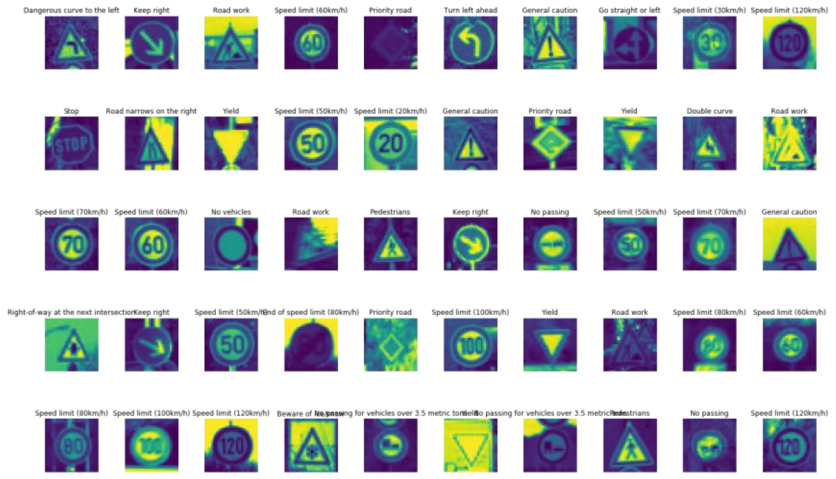
2. **Design, Train and Test of the Model Architecture**

2.1 **Preprocessing steps**

Three preprocessing are applied to the images

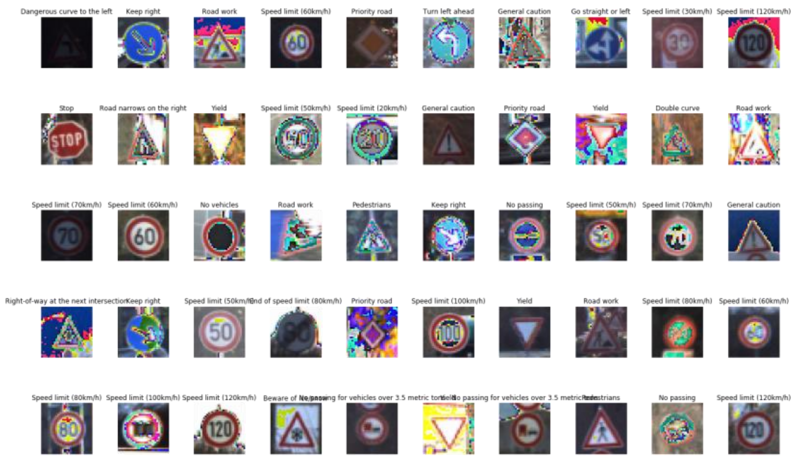
- ✓ **Gray scaling**

  We convert the color images to grayscale as several images are dark, hence, contain only little color information. The grayscale images are less computational intensive.



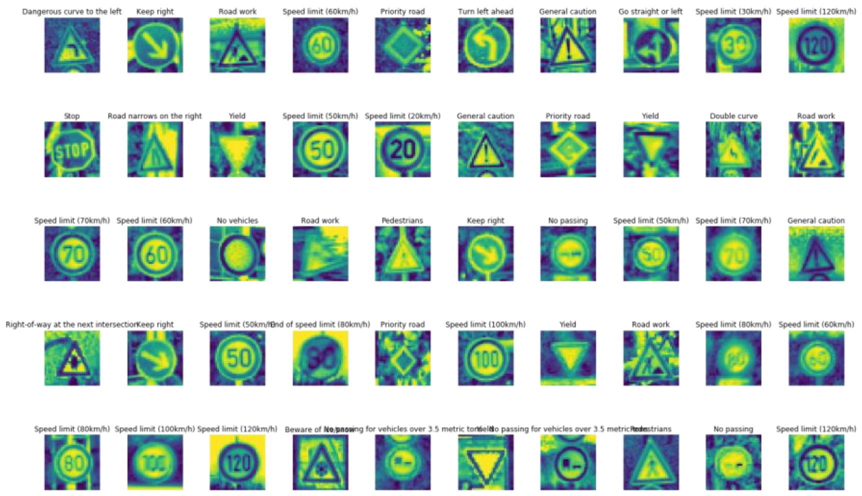
- ✓ **Image Normalization**

  We center the distribution of the images by subtracting each by the mean and divide by its standard deviation. This helps our model treat the images uniformly.



✓ **Histogram Equalization**

Histogram Equalization is a computer vision technique used to increase the contrast in images. As some of our images suffer from low contrast (blurry, dark), we will improve visibility by applying OpenCV's Contrast Limiting Adaptive Histogram Equalization (aka CLAHE) function.



2.2 **Model Architecture**

The model architecture is based on LeNet model architecture. We modified the model by adding dropout at each fully connected layer to prevent over fitting.

Enrollment in local colleges, 2005

Layer	Description
Input	32x32x1 grayscale image
Convolution 5x5	1x1 stride, valid padding, output 28x28x6
Relu	
Max pooling	2x2 stride, output 14x14x16
Input	4x14x16
Convolution 5x5	1x1 stride, valid padding, output 28x28x6
Relu	
Max pooling	2x2 stride, output 5x5x16
flatten	Output 400
Dropout	0.5%
Fully connected	Output 120
Relu	
Dropout	0.5%
Fully connected	Output 84
Relu	
Dropout	0.5%
Fully connected	Output 43
softmax	

2.3 **Model training**

The model was trained using Adam optimizer with the following hyper parameters:

Batch size	128
Epochs	200
Learning rate	0.0005
dropout	0.5

The model achieve the following accuracy

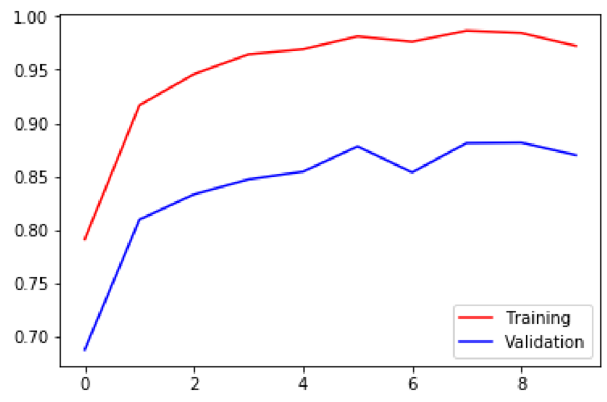
- ✓ Training accuracy 99.9%
- ✓ Validation accuracy 96.3%
- ✓ Test accuracy 93.2%

2.4 **Solution Methodology**

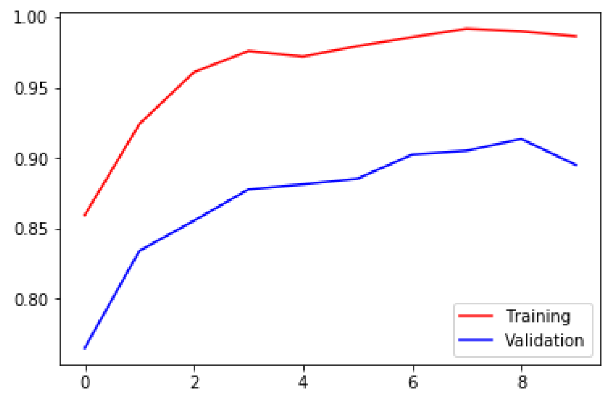
The iterative method was used to optimize the model to achieve the required accuracy. Different combinations of hyper parameters were tried.

1. In the first attempt, we used original LeNet architecture from Udacity course. The input shape was changed to (32x32x3) to reflect color image from German Traffic Sign dataset. And also, the output class was changed to 43 to fit 43 unique labels in the training set.

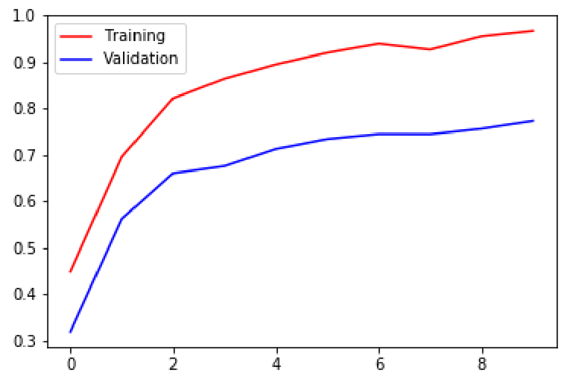
   We obtained training and the validation accuracies as **99.0%** and **87.3%** respectively with the following parameters**: EPOCHS:10, BATCH_SIZE:128, learning_rate:0.001, mu:0, sigma:0.1**



2. We converted the images to grayscale while leaving hyper parameters unmodified. The training accuracy was **98.6%** and the validation accuracy was **90.5%**

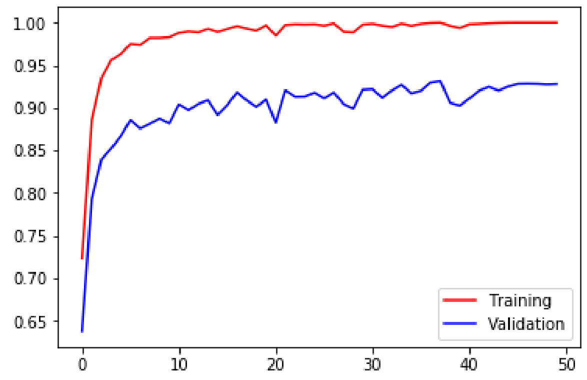


3. We added normalization to the preprocessing steps while hyper parameters still unmodified. The training accuracy was **97.6%** and the validation accuracy was **88.2%**



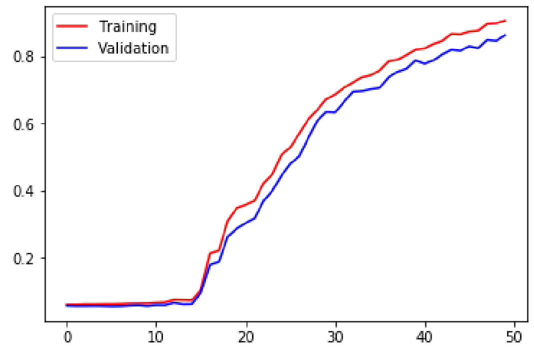
4. Considering both the training and the validation accuracies dropped by adding normalization, hence, the normalization step was removed. Then we modified some hyper parameters as followed: **EPOCHS:50, BATCH_SIZE:128, learning_rate:0.0005, mu:0, sigma:0.1.**

The training and the validation accuracies were **99.8%** and **92.0%** respectively.



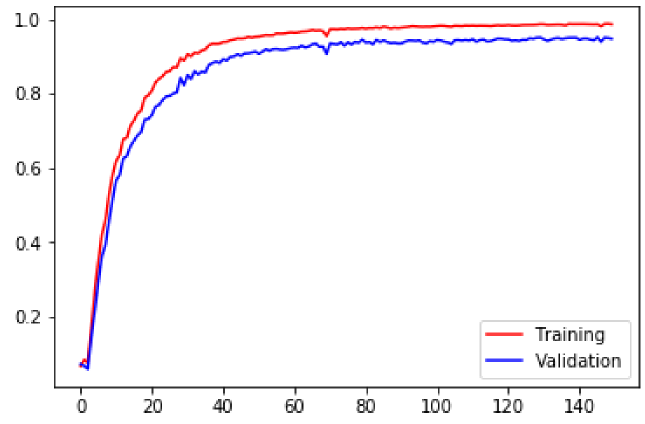
5. We added dropout layer to the three fully connected layers with these hyper parameters; **EPOCHS:150, BATCH_SIZE:128, learning_rate:0.0005, mu:0, sigma:0.1.**

The training accuracy was **99.9%** and the validation accuracy was **96.0%**



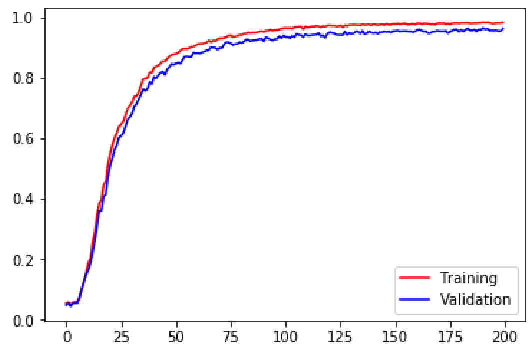
6. We further reduced the learning rate and increased the epochs but the accuracies remain the same; **EPOCHS:200, BATCH_SIZE:128, learning_rate:0.0004, mu:0, sigma:0.1.**

The training accuracy was **99.9%** and the validation accuracy was **96.0%**



7. Finally, we added the equalization to the preprocessing steps.

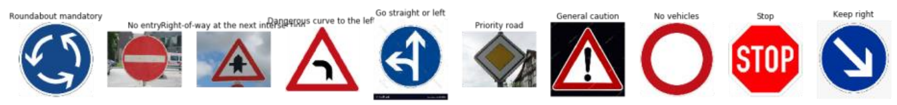
   The training accuracy was **98.8%** and the validation accuracy was **96.3%**



3. Use the model to make inference on the new images acquired from the web.

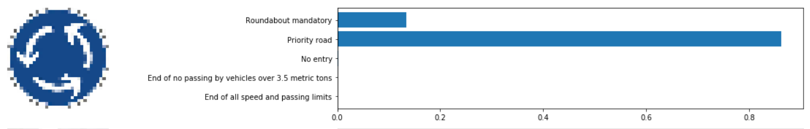
3.1 Model Performance

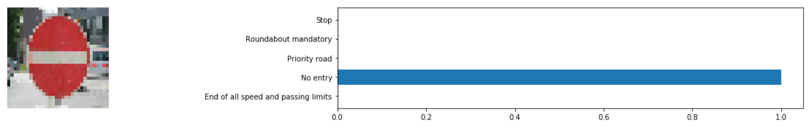
We downloaded ten (10) German Traffic Signs images from web. We performed inference these images. The model was able to correctly guess nine (9) out of ten (10) given accuracy of 90%

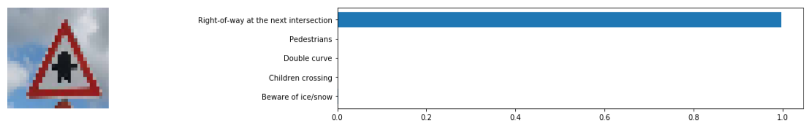


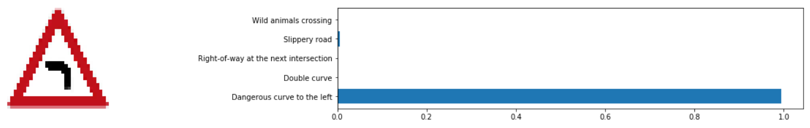
3.2 Model Certainty

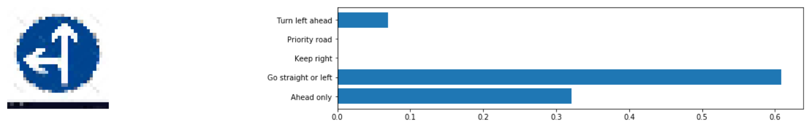
Below we showed model certainty on each new image with top 5 softmax probabilities.

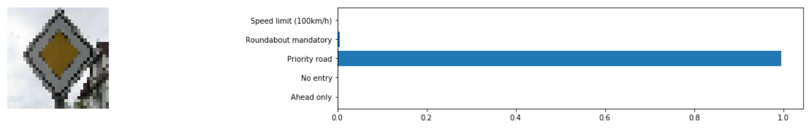


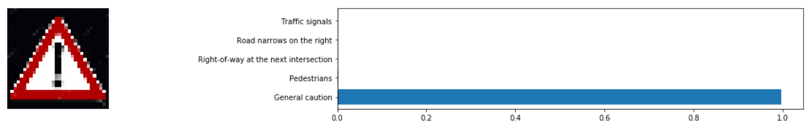


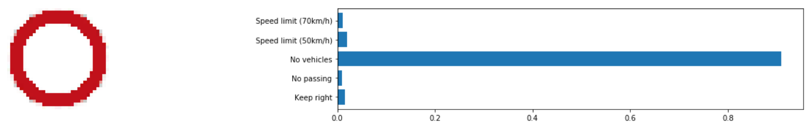


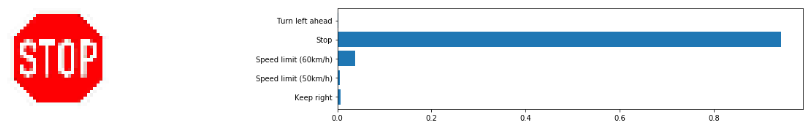


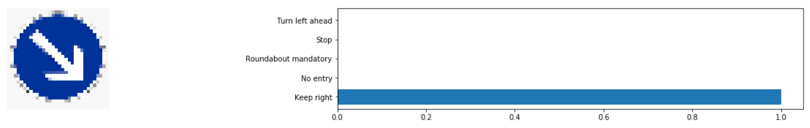












4. **Conclusion**

In this project, we trained a model that classify German traffic signs. We started by loading the data and then calculate summary statistics. We also explore the images by visualization and finally trained and test the model.

The model achieved 96.3% validation accuracy and 90% test accuracy on ten (10) German traffic signs acquired from the web.

In future, we will improve the model performance by incorporating data augmentation as well as exploring some model architecture such as VGG, ResNet, AlexNet.