**Computer Vision based Traffic Sign Recognition**

**Motivation:**

The motivation for the algorithm is to develop a **TRAFFIC SIGN RECOGNITION SYSTEM** that is compatible with **RASPBERRY PI.**

**IMPLEMENTATION:**

The Implementation of the algorithm is divided into 2 parts.

1. Image pre-processing
2. Image Recognition

**TOOLS USED:**

The following tools are for applying the algorithm in Python and Google Collab

1. Python
2. Visual Studio Code (Editor for many types of software like Python, C, Java etc.,)
3. List of libraries in python (Installation commands)
   1. OS (default library installed while installing python)
      1. This library is used to access files and folders available in the system (windows)
   2. OpenCV (pip install opencv)
      1. This library is used to apply image pre-processing methods on images
   3. Pandas (pip install pandas)
      1. This library is used for reading files like excel, csv etc.,
   4. Numpy (pip install numpy)
      1. This library is used to perform matrix operations easily on data
   5. TensorFlow and Keras (pip install tensorflow)
      1. This library allows user to write their own version of Deep Learning Model.
4. **Google Collab:**
   1. This web service is useful in training Deep Learning models with GPU.
      1. GPU is similar to CPU but does work a lot faster than CPU.

**PRE-PROCESSING:**

There are several steps in this process

1. Applying Histogram Equalization on the image
2. Converting Image to Gray Scale

**Histogram-Equalization:**

This is a image processing technique that helps to equalize the pixel Intensity throughout the image. This method is helpful to recognize images in low lighting conditions.

**Gray Scale:**

Grayscaling is a method where the image is converted from Color to Gray. This reduces the Image size

In a Color Image there are 3 different matrices.

1. Red intensity matrix
2. Green intensity matrix
3. Blue intensity matrix

But a gray image has only 1 matrix.

Hence we can decrease large amount of data by converting image to greyscale. This decrease in size helps algorithm to run faster.

**RECOGNITION:**

In this phase the pre-processed images are sent into a Deep Learning model for recognition.

The model was inspired from:

1. The idea of using Convolutional Neural Network.
2. The idea of Alex Net and VGG Net.

**METHOD FOLLOWED:**

To implement the algorithm we have used the GERMAN TRAFFIC SIGN DATASET (<https://www.kaggle.com/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign>). The Dataset was divided into train and test folders. A total of 43 classes of different Traffic signs are taken for training. Along with images 2 different csv files are given for Train and test data in which the region of interest of each image is specified.

We have imported all the training images into an array using the OS Library specified above. The images are then resized to 30x30. This resizing helps in training the model fast. The resized image is pre-processed as mentioned in the PRE-PROCESSING. The pre-processed images are stored in an file which was uploaded to google drive. This uploaded file is then used in google collab for training the model

The pre-processed training images are then sent into a Deep learning model in order to train the model. Here we have used various models mentioned in RECOGNNITION. The **ACCURACY** of entire model is as at 97% which is a huge change from the previous Machine Leaning Models.

**Improvements:**

1. The current model is trained upon entire image. To decrease the computation the model need to be trained on Region of Interest.
2. Finding Region of Interest with certain level of uniqueness.