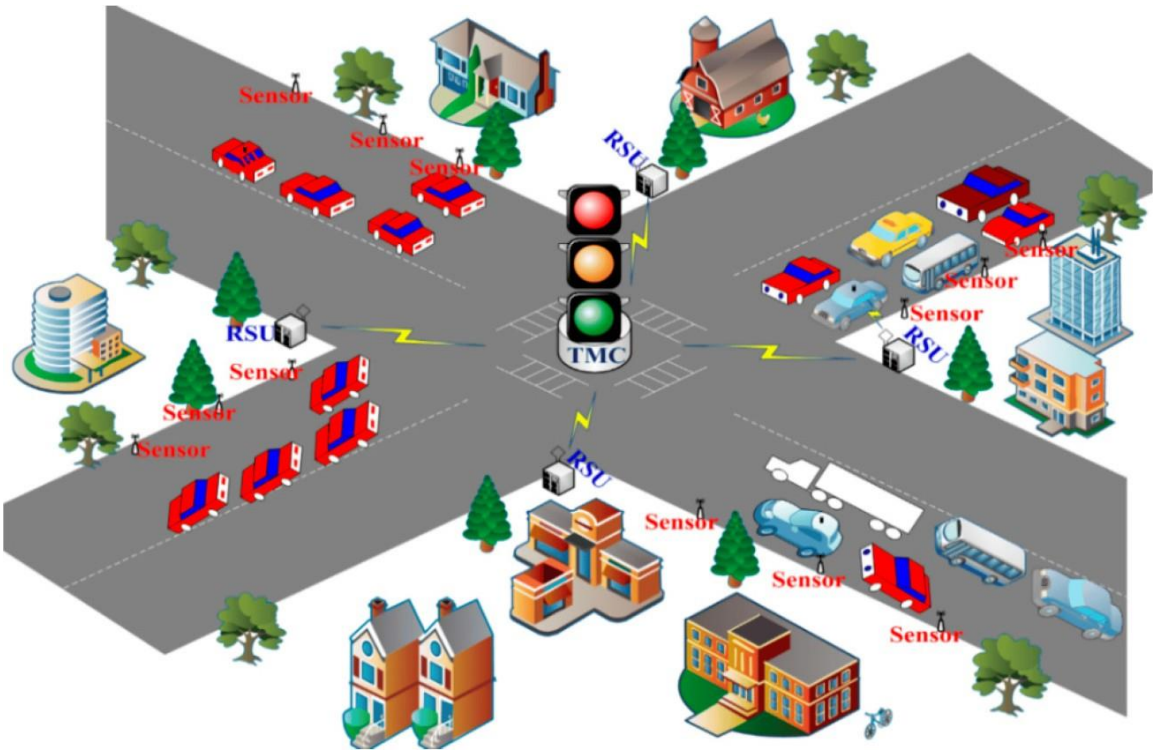
# IOT-TRAFFIC MANAGEMENT

NAME: A.SUJI

ID :aut962921104711

EMAIL: suji76531@gmail.com

# TABLE OF CONTENTS



# DEVELOPMENT PART 1

## Start Building the IOT Enable Traffic

## Management in systems:

# 1) Introduction:

Traffic Management Is the combination of measures that serve to preserve traffic capacity and improve the security , safety , And reliability of the overall road transport system. These measures make

Use of ITS systems , services and projects in day-to-day operations that impact on road network performance.

# 2)Overall design:

***Identify the problem**: The first step is to identify the problem that the traffic control system is intended to solve.

***conduct a traffic study** : A traffic study is an analysis of the traffic patterns and volume at a particular location.

# 3)Software Design:

Chetu designs customizable Traffic Management Systems (TMS) dashboards with modules for features such as Quick Response , GIS – powered map recording And playback , report retrieval ,

field communication , variable speed limit signage, route guidance technologies , intersection control tools including automated traffic – light and pedestrian signal sequencing. Business needs vary , and so should the software ! With our custom traffic management

software development , have full control of your solution's design , functionality , performance ,scalability , and maintenance .

Make traffic infrastructure serve drivers and businesses with our top-notch traffic management software development services ! - We collect and process traffic, Vehicle , road , and weather data to help your

business with navigation, data –driven decisions ,maintenance management , and more ….

# 4)System Testing:

The software test program is intended to cover the software testing from design reviews through hardware/software integration testing . All software proposed for use in your TMS should be subjected to

testing before it is accepted and user to support operations. The extent and through ness of that testing should be based on the maturity of that software and the risk you are assuming in including it in your operations.

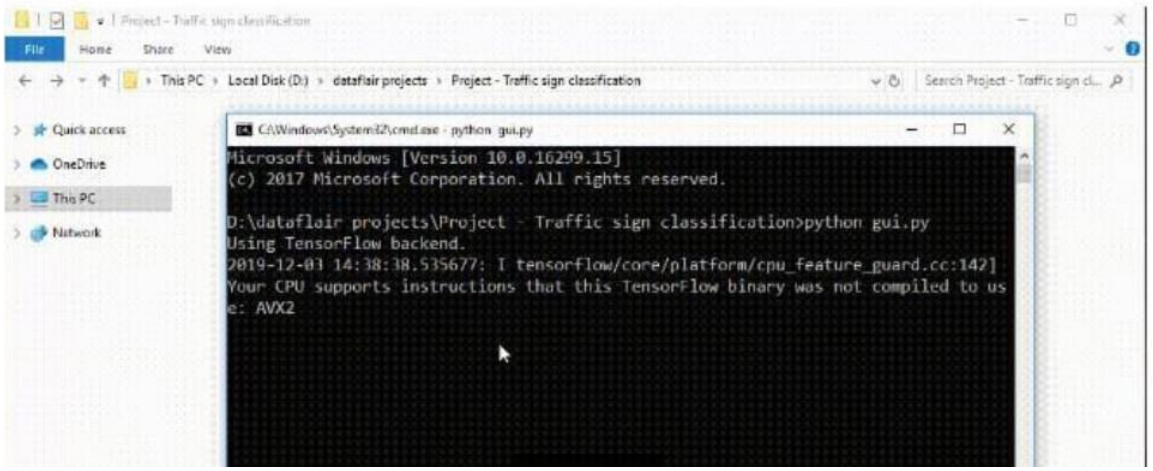
# 5)Python Script :

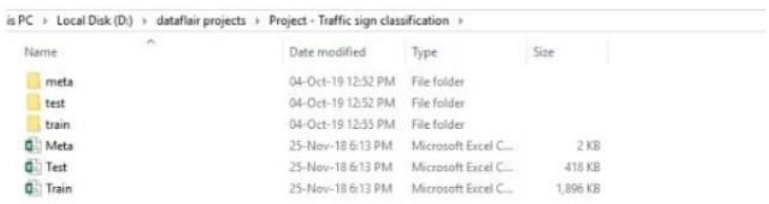
In this Python project example , we will build a deep neural network model that can classify traffic signs present in the image into different

categories . With this model, we are able to read and understand traffic signs which are a very important task for all autonomous vehicles.

And extract the files into a folder such that you will

have a train, test and a meta folder.




Create a python script file and name it traffic _ signs.py in the

project folder .

Our approach to building this traffic sign classification model is discussed in four

**Steps:**

* Explore the dataset

* Build a CNN model

* Train and Validate the model

Test the model with test dataset

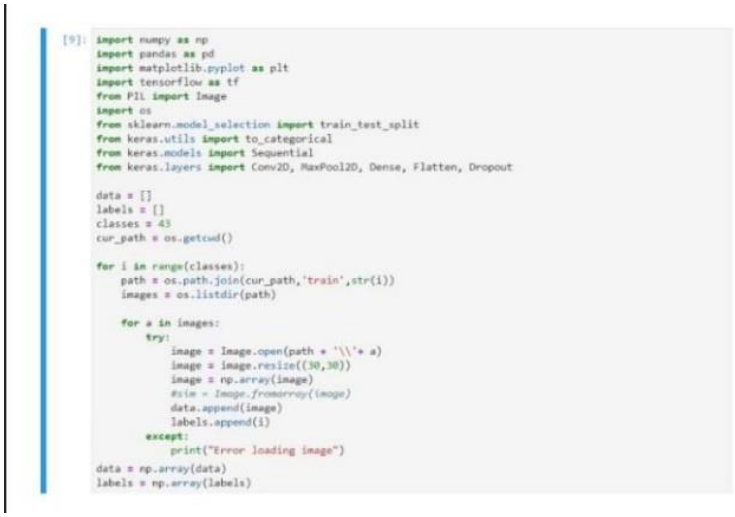
**Step 1** : Explore the dataset

Our ‘train’ folder contains 43folders each representing a different class. The range of the folder is from 0 to 42.with the help of the OS module we iterate and their respective labels in the data and labels list.

The PIL library is used to open image content into an array.

Finally ,we have stored all the images and their label

into lists.

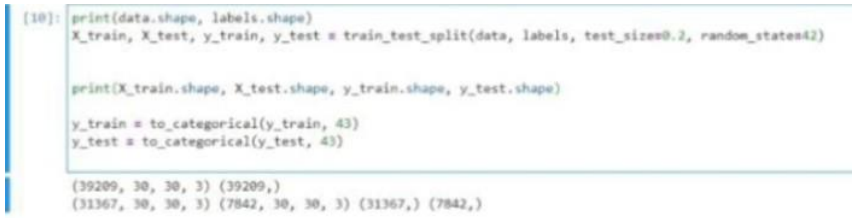


```
[9]: import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import tensorflow as tf
     from PIL import Image
     import os
     from sklearn.model_selection import train_test_split
     from keras.utils import to_categorical
     from keras.models import Sequential
     from keras.layers import Conv2D, MaxPool2D, Dense, Flatten, Dropout

     data = []
     labels = []
     classes = 43
     cur_path = os.getcwd()

     for i in range(classes):
         path = os.path.join(cur_path,'train',str(i))
         images = os.listdir(path)

         for a in images:
             try:
                 image = Image.open(path + '\\'+ a)
                 image = image.resize((30,30))
                 image = np.array(image)
                 #sim = Image.fromarray(image)
                 data.append(image)
                 labels.append(i)
             except:
                 print("Error loading image")
     data = np.array(data)
     labels = np.array(labels)
```

We need to convert the list into numpy arrays for feeding to the model.

The shape of data is(39209,30,30,3) which meansthat there are 39,209 images of size 30*30pixels and the last 3 means the data contain colored images. We use the train _ test _ split() training and testing

data.From the keras . Utils package, we use to_categorical method to convert the labels presentin y _ train and t_test into one –hot encoding.




**Step 2**: Build a CNN Model To classify the images into their

respective categories , we will build a CNN model .

The architecture of our model is :

.2 Conv2D Layer

(filter=32,kernel_size=(5,5),activation="relu")

. Maxpool2d layer(pool_size=(2,2))

.Dropout layer(ratio=0.25)

.2Conv2D layer

(filter=64,kernel_size=(3,3),activation="relu")

.Maxpool2D layer (pool_size=(2,2))

.Dropout layer (rate=0.25)

.Flatten layer to squeeze the layers into 1dimension

.Dense Fully connected layer(256 nodes,activation=“relu”)

.Dropout layer (rate=0.5)

.Dense layer (43 nodes,activation=“softmax”)

We Compile the model with adam optimizer which performs well and loss is “categorical _ crossentropy ”because we have multiple classes to categorise.

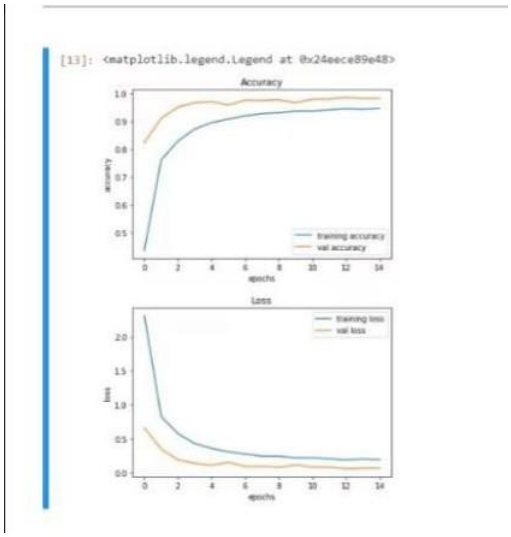
```
[11]: model = Sequential()
      model.add(Conv2D(filters=32, kernel_size=(5,5), activation='relu', input_shape=X_train.shape[1:]))
      model.add(Conv2D(filters=32, kernel_size=(5,5), activation='relu'))
      model.add(MaxPool2D(pool_size=(2, 2)))
      model.add(Dropout(rate=0.25))
      model.add(Conv2D(filters=64, kernel_size=(3, 3), activation='relu'))
      model.add(Conv2D(filters=64, kernel_size=(3, 3), activation='relu'))
      model.add(MaxPool2D(pool_size=(2, 2)))
      model.add(Dropout(rate=0.25))
      model.add(Flatten())
      model.add(Dense(256, activation='relu'))
      model.add(Dropout(rate=0.5))
      model.add(Dense(43, activation='softmax'))

      #Compilation of the model
      model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

**step 3**: Train and Validate the model After building the model architecture, we then train the model using model .fit(). I tied with batch size 32 and 64 .our model performed better with 64

batch size . And after 15 epochs the accuracy wasstable.

```
[13]: plt.figure(0)
      plt.plot(history.history['accuracy'], label='training accuracy')
      plt.plot(history.history['val_accuracy'], label='val accuracy')
      plt.title('Accuracy')
      plt.xlabel('epochs')
      plt.ylabel('accuracy')
      plt.legend()

      plt.figure(1)
      plt.plot(history.history['loss'], label='training loss')
      plt.plot(history.history['val_loss'], label='val loss')
      plt.title('Loss')
      plt.xlabel('epochs')
      plt.ylabel('loss')
      plt.legend()

[13]: <matplotlib.legend.Legend at 0x24eece89e48>
```

## Plotting Accuracy




## Accuracy and loss Graphs

**Step 4**: Test our model with test dataset Our dataset contains a test folder and in a test. CSV file ,we have the details related to the image path and their respective class labels using pandas . Then to predict the model , we have to resize our images to 30*30 pixels and make a numpy array containing all image data. From the sk learn.metrics , we imported the accuracy_score and observed how our model predicated the actual labels . Weachieved a 95%accuracy in this model .

```
[14]: from sklearn.metrics import accuracy_score
      import pandas as pd
      y_test = pd.read_csv('Test.csv')

      labels = y_test["ClassId"].values
      imgs = y_test["Path"].values

      data=[]

      for img in imgs:
          image = Image.open(img)
          image = image.resize((30,30))
          data.append(np.array(image))

      X_testnp.array(data)

      pred = model.predict_classes(X_test)

      #Accuracy with the test data
      from sklearn.metrics import accuracy_score
      accuracy_score(labels, pred)
[14]: 0.9532066508313539
```

In the end, we are going to save the model that we have trained using the model that we have trained using the keras model. Save() function.

**model . Save (‘ traffic _ classifier . h5’)**

# Conclusion:

.With the help of ITS , the traffic congestions , rate of road accidents , wastage of fuels will be decreased to a large extent.

.This gives the people of the country a more economic mean of transportation withadvanced information of transits.

Hence with much more interest and advanced research in the field of ITS , it can be 100% efficiency in our country and can prove to be an effective solution to the traffic problems.