CHAPTER 10

APPENDICES

Source Code

[ ]:
```
!pip install kaggle
!pip install IPython
!pip install seaborn
!pip install plotly
!pip install -U scikit-learn scipy matplotlib
!pip install keras
```

[15]:
```
! mkdir ~/.kaggle
```

```
mkdir: cannot create directory ‘/root/.kaggle’: File exists
```

[16]:
```
!cp /content/kaggle.json ~/.kaggle/
```

[17]:
```
! chmod 600 ~/.kaggle/kaggle.json
```

[18]:
```
! kaggle datasets download -d odins0n/ucf-crime-dataset
```

```
Downloading ucf-crime-dataset.zip to /content
100% 11.0G/11.0G [04:44<00:00, 45.6MB/s]
100% 11.0G/11.0G [04:44<00:00, 41.5MB/s]
```

[19]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import os

import tensorflow as tf
from tensorflow.keras.preprocessing import image_dataset_from_directory
from tensorflow.keras.applications import DenseNet121
from sklearn.preprocessing import LabelBinarizer
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D,␣
 ↪Dropout,MaxPooling2D , Conv2D,Flatten
from tensorflow.keras.models import Sequential

from IPython.display import clear_output
import warnings
```

```
warnings.filterwarnings('ignore')
```

[ ]:
```
! unzip ucf-crime-dataset.zip
```

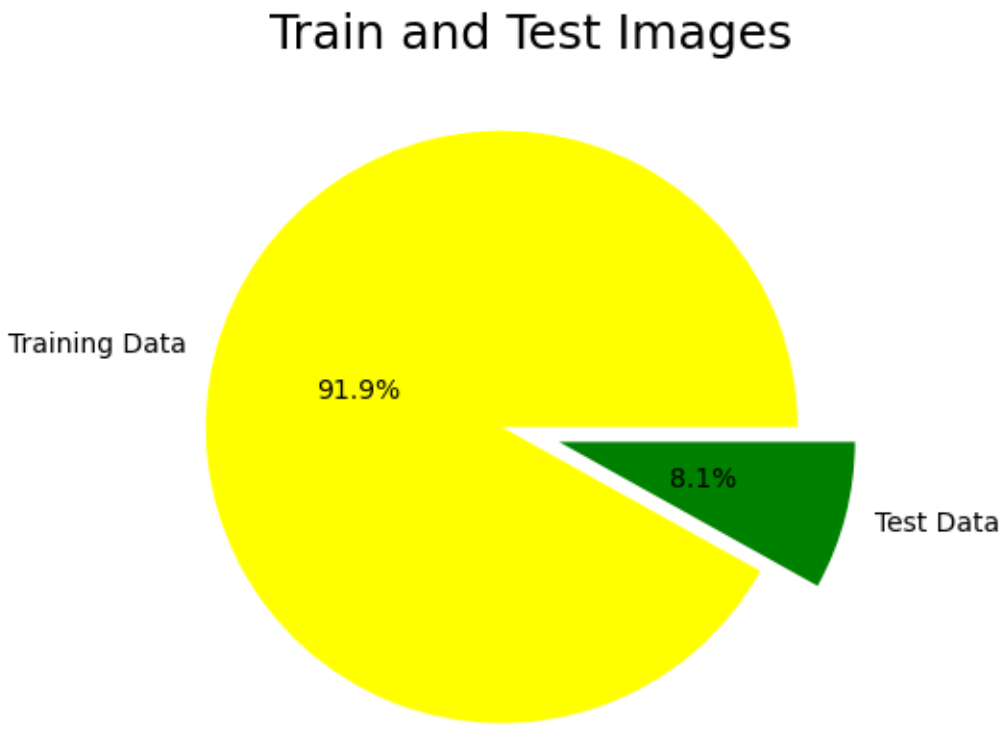
[21]:
```
train_dir="/content/Train"
test_dir="/content/Test"

SEED = 12
IMG_HEIGHT = 64
IMG_WIDTH = 64
BATCH_SIZE = 128
EPOCHS = 5
LR = 0.00003
```

[22]:
```
crime_types=os.listdir(train_dir)
n=len(crime_types)
print("Number of Crime Categories : ",n)
```

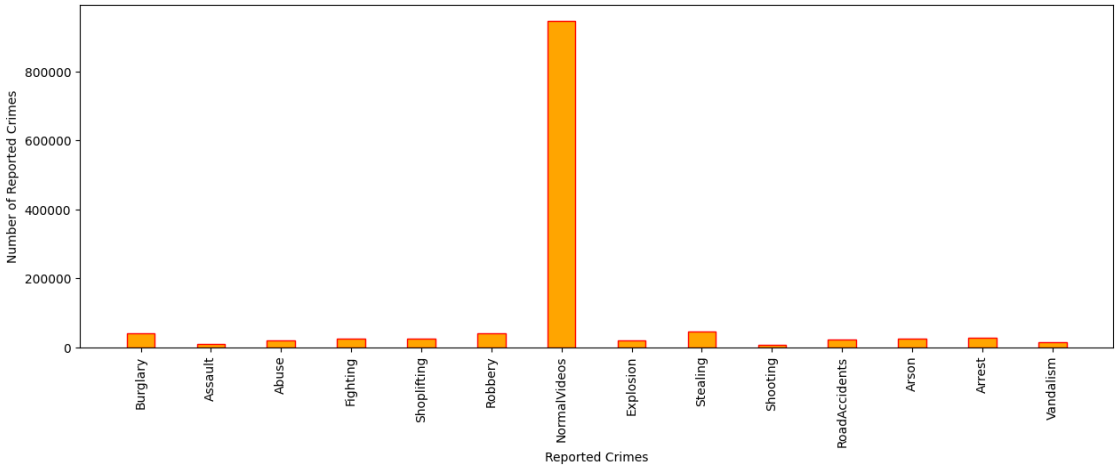
```
Number of Crime Categories :  14
```

[23]:
```
crimes={}
train=test=0
for clss in crime_types:
  num=len(os.listdir(os.path.join(train_dir,clss)))
  train+=num
  test+=len(os.listdir(os.path.join(test_dir,clss)))

  crimes[clss]=num
```

[24]:
```
plt.figure(figsize=(8,5))
plt.pie(x=np.array([train,test]),autopct="%.1f%%", explode=[0.1,0.1],␣
 ↪labels=["Training Data", "Test Data"], pctdistance=0.5,␣
 ↪colors=['yellow','green'])
plt.title("Train and Test Images", fontsize=18);
```



[25]:
```
plt.figure(figsize=(15,5))
plt.bar(list(crimes.keys()),list(crimes.values()),width=0.4, align="center",␣
  ↪edgecolor=['red'], color=['orange'])
plt.xticks(rotation=90)

plt.xlabel("Reported Crimes")
plt.ylabel("Number of Reported Crimes")
plt.show()
```



[26]:
```
IMG_SHAPE=(IMG_WIDTH,IMG_HEIGHT)
```

[27]:
```
train_set=image_dataset_from_directory(
    train_dir,
    label_mode="categorical",
    batch_size=BATCH_SIZE,
    image_size=IMG_SHAPE,
    shuffle=True,
    seed=SEED,
    validation_split=0.2,
    subset="training"
)
```

```
Found 1266345 files belonging to 14 classes.
Using 1013076 files for training.
```

[28]:
```
val_set=image_dataset_from_directory(
    train_dir,
    label_mode="categorical",
    batch_size=BATCH_SIZE,
    image_size=IMG_SHAPE,
    shuffle=True,
    seed=SEED,
    validation_split=0.2,
    subset="validation"
)
```

```
Found 1266345 files belonging to 14 classes.
Using 253269 files for validation.
```

[29]:
```
test_set=image_dataset_from_directory(
     test_dir,
     label_mode="categorical",
     image_size=IMG_SHAPE,
     shuffle=False,
     class_names=None,
     batch_size=BATCH_SIZE,
     seed=SEED
)
```

```
Found 111308 files belonging to 14 classes.
```

[30]:
```
INPUT_SHAPE=(64,64,3)
```

[31]:
```
def transfer_learning():
  base_model=DenseNet121(include_top=False,
                         input_shape=INPUT_SHAPE,
                         weights="imagenet")

  thr=14

  for layers in base_model.layers[:thr]:
    layers.trainable=False
  for layers in base_model.layers[thr:]:
    layers.trainable=False

    return base_model
```

[32]:
```
def create_model():
  model=Sequential()

  base_model=transfer_learning()
  model.add(base_model)

  model.add(GlobalAveragePooling2D())

  model.add(Dense (256, activation="relu"))
  model.add(Dropout (0.2))

  model.add(Dense (512, activation="relu"))
  model.add(Dropout (0.2))

  model.add(Dense (1024, activation="relu"))

  model.add(Dense (n, activation="softmax"))

  model.summary()
```

```
  return model
```

[33]:
```
model=create_model()

model.compile(optimizer="adam",
              loss='categorical_crossentropy',
              metrics = ['accuracy'])
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-
applications/densenet/densenet121_weights_tf_dim_ordering_tf_kernels_notop.h5
29084464/29084464 [==============================] - 2s 0us/step
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 densenet121 (Functional)    (None, 2, 2, 1024)        7037504

 global_average_pooling2d (G  (None, 1024)             0
 lobalAveragePooling2D)

 dense (Dense)               (None, 256)               262400

 dropout (Dropout)           (None, 256)               0

 dense_1 (Dense)             (None, 512)               131584

 dropout_1 (Dropout)         (None, 512)               0

 dense_2 (Dense)             (None, 1024)              525312

 dense_3 (Dense)             (None, 14)                14350

=================================================================
Total params: 7,971,150
Trainable params: 7,832,334
Non-trainable params: 138,816
_________________________________________________________________
```

[34]:
```
history = model.fit(x=train_set,validation_data=val_set, epochs=EPOCHS)
```

```
Epoch 1/5
7915/7915 [==============================] - 1729s 207ms/step - loss: 0.0970 -
accuracy: 0.9741 - val_loss: 0.1867 - val_accuracy: 0.9640
Epoch 2/5
7915/7915 [==============================] - 1625s 205ms/step - loss: 0.0279 -
accuracy: 0.9931 - val_loss: 0.0464 - val_accuracy: 0.9874
Epoch 3/5
```

```
7915/7915 [==============================] - 1627s 205ms/step - loss: 0.0202 -
accuracy: 0.9951 - val_loss: 0.0158 - val_accuracy: 0.9962
Epoch 4/5
7915/7915 [==============================] - 1594s 201ms/step - loss: 0.0161 -
accuracy: 0.9961 - val_loss: 0.0119 - val_accuracy: 0.9968
Epoch 5/5
7915/7915 [==============================] - 1593s 201ms/step - loss: 0.0151 -
accuracy: 0.9964 - val_loss: 0.0084 - val_accuracy: 0.9977
```

[35]:
```
model.save('crime.h5')
```

Home.html

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Navigation Bar</title>
    <style type="text/css" >

        body {
            background-color: rgb(163, 170, 173);
        }


        *{
            text-decoration: none;
        }
        .navbar{
            background: rgb(122, 123, 120); font-family: calibri; padding-right:
15px;padding-left: 15px;
        }
        .navdiv{
            display: flex; align-items: center; justify-content: space-between;
        }
        .logo a{
            font-size: 35px; font-weight: 600; color: white;
        }
        li{
            list-style: none; display: inline-block;
        }
        li a{
            color: white; font-size: 18px; font-weight: bold; margin-right: 25px;
        }
        button{
            background-color: black; margin-left: 10px; border-radius: 10px;
padding: 10px; width: 90px;
        }
        button a{
            color: white; font-weight: bold; font-size: 15px;
        }

        body {
      margin: 0;
      padding: 0;
      font-family: 'Times New Roman', Times, serif;
```

```
      font-size: 20px
    }

    .container {
      display: flex;
      flex-wrap: wrap;
    }

    .column {
      flex: 1;
      padding: 20px;
      box-sizing: border-box;
    }

    .column img {
      max-width: 100%;
      height: auto;
      display: block;
      margin-bottom: 20px;
    }

    @media (max-width: 768px) {
      .column {
        flex-basis: 100%;
      }
    }



    </style>
</head>
<body>
    <nav class="navbar">
        <div class="navdiv">
            <div class="logo"><a href="#">Crime Classification</a> </div>
            <ul>
                <button><a href="home.html">Home</a></button>
                <button><a href="predict.html">Predict</a></button>
            </ul>
        </div>
    </nav>

    <div class="container">
        <div class="column">
```

```
          <img src="https://blog.ipleaders.in/wp-
content/uploads/2019/01/Fraud.jpg" alt="Image">
        </div>
        <div class="column">
          <h2 >Stop Crime</h2>
          <p>Crime refers to any act that violates the established laws and
regulations of a society, resulting in harm to individuals or the community as a
whole. It encompasses a wide range of behaviors, from petty offenses to serious
felonies. Crimes are classified based on their nature and severity, allowing for
a systematic understanding and handling of different criminal acts. One common
classification system categorizes crimes into four main types: personal crimes,
property crimes, inchoate crimes, and statutory crimes. Personal crimes involve
direct harm or threat to an individual's physical or psychological well-being,
such as assault, robbery, or murder. Property crimes, on the other hand, focus on
offenses against possessions, including burglary, theft, or arson. Inchoate
crimes refer to actions that indicate an intention to commit a crime, like
conspiracy or attempted robbery. Lastly, statutory crimes involve violations of
specific laws and regulations, such as drug offenses, traffic violations, or
white-collar crimes. Understanding the classification of crimes is essential for
law enforcement agencies, legal professionals, and policymakers in determining
appropriate punishments, prevention strategies, and rehabilitation programs to
maintain social order and protect the welfare of individuals and communities</p>
        </div>
      </div>

</body>
</html>
```

Predict.html

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Navigation Bar</title>
    <style type="text/css" >

        body {
        background-color: rgb(163, 170, 173);
        }

        *{
            text-decoration: none;
        }
        .navbar{
            background: rgb(122, 123, 120); font-family: calibri; padding-right:
15px;padding-left: 15px;
        }
        .navdiv{
            display: flex; align-items: center; justify-content: space-between;
        }
        .logo a{
            font-size: 35px; font-weight: 600; color: white;
        }
        li{
            list-style: none; display: inline-block;
        }
        li a{
            color: white; font-size: 18px; font-weight: bold; margin-right: 25px;
        }
        button{
            background-color: black; margin-left: 10px; border-radius: 10px;
padding: 10px; width: 90px;
        }
        button a{
            color: white; font-weight: bold; font-size: 15px;
        }

        body {
            margin: 0;
            padding: 0;
            font-family: 'Times New Roman', Times, serif;
```

```
            font-size: 20px
        }

        .container {
            display: flex;
            flex-wrap: wrap;
        }

        .column {
            flex: 1;
            padding: 20px;
            box-sizing: border-box;
        }

        .column img {
            max-width: 100%;
            height: auto;
            display: block;
            margin-bottom: 20px;
        }

        @media (max-width: 768px) {
            .column {
                flex-basis: 100%;
            }
        }

        button a{
            color: rgb(255, 255, 255); font-weight: bold; font-size: 15px;
        }



    </style>
</head>
<body>
    <nav class="navbar">
        <div class="navdiv">
            <div class="logo"><a href="#">Crime Classification</a> </div>
            <ul>
                <button><a href="home.html">Home</a></button>

            </ul>
        </div>
    </nav>
```

```
    <div class="container">
        <div class="column">
          <img src="https://images.pexels.com/photos/923681/pexels-photo-
923681.jpeg" alt="Image">
        </div>
        <div class="column">
          <h2 >Stop Crime</h2>
          <p>Drop the Image to get the prediction</p>

          <form action="upload.php" method="post" enctype="multipart/form-data">
            Select image to upload:
            <input type="file" name="fileToUpload" id="fileToUpload">
            <input type="submit" value="Upload Image" name="submit">
          </form>

          <p>{{ text }}</p>


        </div>
      </div>

</body>
</html>
```

In [15]:

```
import re
import numpy as np
import pandas as pd
import os
import tensorflow as tf
from flask import Flask, app, request, render_template
from tensorflow.keras import models
from tensorflow.keras.preprocessing import image
from tensorflow.python.ops.gen_array_ops import concat
from tensorflow.keras.models import load_model
from werkzeug.utils import secure_filename
from flask import Flask, request
from google.colab import files
```

In [16]:

```
model=load_model(r"crime.h5",compile=False)
app=Flask(__name__)
```

In [17]:

```
@app.route('/')
def home():
  return render_template('home.html')
@app.route('/prediction')
def prediction():
  return render_template('predict.html')
```

In [18]:

```
@app.route('/predict', methods=['POST'])
def predict():
  if request.method=='POST':

      # Get the file from post request
      f = request.files['image']
      #Save the file to ./uploads
      basepath = os.path.dirname('upload.php')
      file_path = os.path.join(basepath, 'uploads', secure_filename(f.filename))
      f.save(file_path)
      img = image.load_img(file_path, target_size=(64, 64))
      x = image.img_to_array(img) # Converting image into array
      x = np.expand_dims(x,axis=0) # expanding Dimensions
      pred = np.argmax(model.predict(x)) # Predicting the higher probablity index
      op = ['Abuse','Arrest','Arson','Assault','Burglary','Explosion','Fighting','Normal
Videos','RoadAccidents','Robbery','Shooting','Shoplifting', 'Stealing','Vandalism']
      op[pred]
      result=op[pred]
      result='The predicted output is {}'.format(str(result))
      print(result)
  return render_template('predict.html',text=result)
```

In [19]:

```
if __name__=="__main__":
  app.run()
```

```
 * Serving Flask app '__main__'
 * Debug mode: off
```

```
INFO:werkzeug:WARNING: This is a development server. Do not use it in a production deploy
ment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
INFO:werkzeug:Press CTRL+C to quit
```