HUMAN FACE COLOR DETECTION

Name – Pankaj Kumar

Section – KM028

Roll Number – 59

Registration Number – 11910254

Course Name – Product Development and Testing (INT - 248)

Teacher Name – Ma'am, Jaspreet Kaur

CA-1

Introduction

In this project, we make human face colour detections. For detecting the human face, we used openCV2 and the face classifier. We can detect the face from an image, and we can also detect the face live from a webcam. For detecting the colour of an image, we used clustering, and from KMeans I got an accurate result. In KMeans clustering, we divide the colour into five levels and show the whole image's colour in five labels. From any image, show the maximum number of colours in that image. For human readable purposes, we draw the pie chart. which is easily understandable.

Libraries

In this many libraries used like –

- Numpy
- Matplotlib
- Scikit-learn
- CV2

Numpy -

A general-purpose toolkit for handling arrays is called NumPy. It offers a multidimensional array object with outstanding speed as well as capabilities for interacting with these arrays. It is the cornerstone Python module for scientific computing. The programme is open-source. It has a number of characteristics, including the following crucial ones:

a strong object for an N-dimensional array

Complex (broadcasting) operations

C/C++ and Fortran code integration tools

useful Fourier transform, random number, and linear algebra abilities

NumPy is a powerful multi-dimensional data container that has several applications outside of science. NumPy's ability to declare any data-types makes it possible for NumPy to quickly and easily interact with a broad range of databases.

Matplotlib -

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible. Create publication quality plots. Make interactive figures that can zoom, pan, update.

Scikit-learn -

The most effective and reliable Python machine learning package is called Skearn (Skit-Learn). Through a Python consistency interface, it offers a variety of effective tools for statistical modelling and machine learning, including classification, regression, clustering, and dimensionality reduction. This library is based on NumPy, SciPy, and Matplotlib and was created primarily in Python.

Tools for data mining and data analysis that are easy to use and effective. Support vector machines, random forests, gradient boosting, k-means, and other classification, regression, and clustering techniques are included.

everyone-friendly and reusable in several settings. built on top of Matplotlib, SciPy, and NumPy. Open source, useable for profit, BSD licence.

OpenCV is a sizable open-source library for image processing, machine learning, and computer vision. It currently plays a significant part in real-time operation, which is crucial in modern systems. Using it, one may analyse pictures and movies to find people, objects, and even human handwriting. Python is able to handle the OpenCV array structure for analysis when it is combined with other libraries, such as NumPy. We employ vector space and apply mathematical operations to these characteristics to identify visual patterns and their different features.

OpenCV's initial release was 1.0. OpenCV is free for both academic and commercial usage because it is distributed under a BSD licence. It is compatible with Windows, Linux, Mac OS, iOS, and Android and offers C++, C, Python, and Java interfaces. Real-time applications for improved processing efficiency were the primary consideration when OpenCV was developed. Everything is written in C/C++ that has been optimised to take advantage of multi-core processing.

Code Implementation

```
######################: Human Face Detection: ############################
######################: Detect face from Image: ##########################
    import cv2
    img = cv2.imread('SwamiVivekananda.jpg')
    gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    haar_cascade = cv2.CascadeClassifier('Haarcascade_frontalface_default.xml')
    faces_rect = haar_cascade.detectMultiScale(gray_img, 1.1, 9)
    for (x, y, w, h) in faces_rect:
    cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2)
    cv2.imshow('Detected faces', img)
    cv2.waitKey(0)
##################################---: End: ---################################
######################: Detect face from Camera: ##########################
    import cv2
    face_cascade = cv2.CascadeClassifier('Haarcascade_frontalface_default.xml')
    cap = cv2.VideoCapture(0)
    while True:
    _, img = cap.read()
    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray,1.1,4)
    for (x,y,w,h) in faces:
    cv2.rectangle(img,(x,y),(x + w,y + h),(255,0,0),2)
    cv2.imshow('img',img)
    k = cv2.waitKey(30) & 0xff
    if k == 27:
    break
```

```
    cap.release()
################################---: End: ---################################
########################: Color Dectection: ##############################
    from collections import Counter
    from sklearn.cluster import KMeans
    from matplotlib import colors
    import matplotlib.pyplot as plt
    import numpy as np
    import cv2
    def rgb_to_hex(rgb_color):
    hex_color = "#"
    for i in rgb_color:
    i = int(i)
    hex_color += ("{:02x}".format(i))
    return hex_color
    rgb_to_hex((255,0,0))
    img_name = 'SwamiVivekananda.jpg'
    raw_img = cv2.imread(img_name)
    raw_img = cv2.cvtColor(raw_img,cv2.COLOR_BGR2RGB)
    img = cv2.resize(raw_img,(900,600),interpolation = cv2.INTER_AREA)
    img.shape
    img = img.reshape(img.shape[0]*img.shape[1],3)
    img.shape
    img
    clf = KMeans(n_clusters=5)
    color_labels = clf.fit_predict(img)
    center_colors = clf.cluster_centers_
    color_labels
    center_colors
```

```
counts = Counter(color_labels)
counts
ordered_colors = [center_colors[i] for i in counts.keys()]
hex_colors = [rgb_to_hex(ordered_colors[i]) for i in counts.keys()]
hex_colors
plt.figure(figsize = (12,8))
plt.pie(counts.values(),labels = hex_colors,colors = hex_colors)
plt.savefig(f'{img_name[:-4]}-analysis.png')
def hex_to_rgb(hex):
rgb = []
for i in (0, 2, 4):
decimal = int(hex[i:i+2], 16)
rgb.append(decimal)
return tuple(rgb)
print(hex_to_rgb('a96139'))
```

##################################---: End: ---##################################

Face Detections Steps

Step1: - First we import the library of openCV2

Step2: - Then for this project we used face classifier

Step3: - Then we convert RGB image to grayscale image

Step4: - haar_cascade.detectMultiScale(gray_img, 1.1, 9)

First parameter is grayscale image

Second parameter is scale factor

Third parameter is minimum neighbours

Step5: - Then we define dimensions of rectangle like length x breadth and thickness and colour

Step6: - Using openCV then we show the image using imshow

Step7: - And for live detection we used same method

Color Detections Steps

Step1: - First we import all important library like sklearn, matplotlib, numpy, openCV2

Step2: - Then we convert rgb to hex colour code

Step3: - Then we give the path image

Step4: - We convert into BGR to RGB

Step5: - Then we resize the size of image

Step6: - Then apply KMeans clustering on that image and make group of five

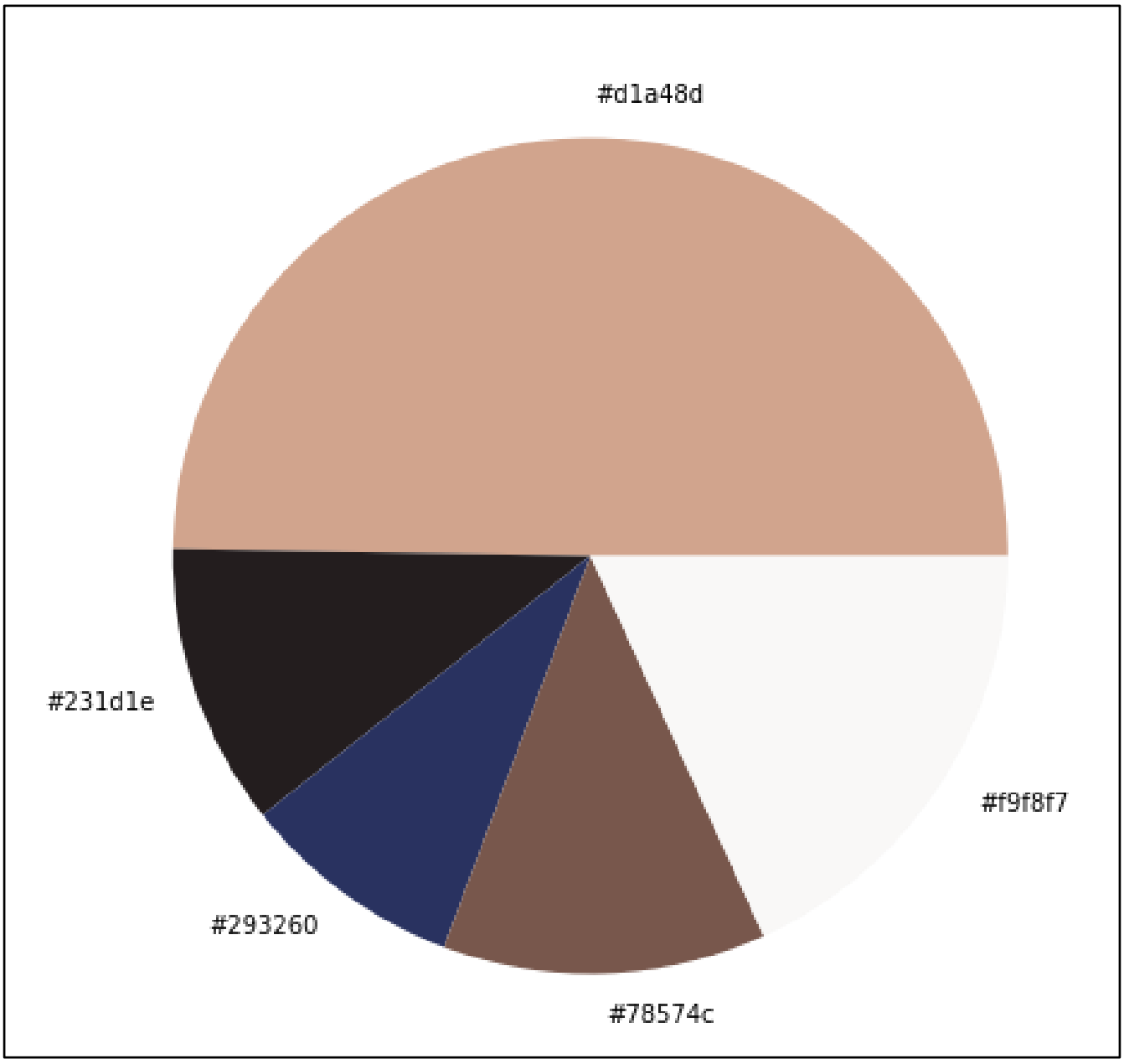
Step7: - Then we train the image

Step8: - We predict the colour of image

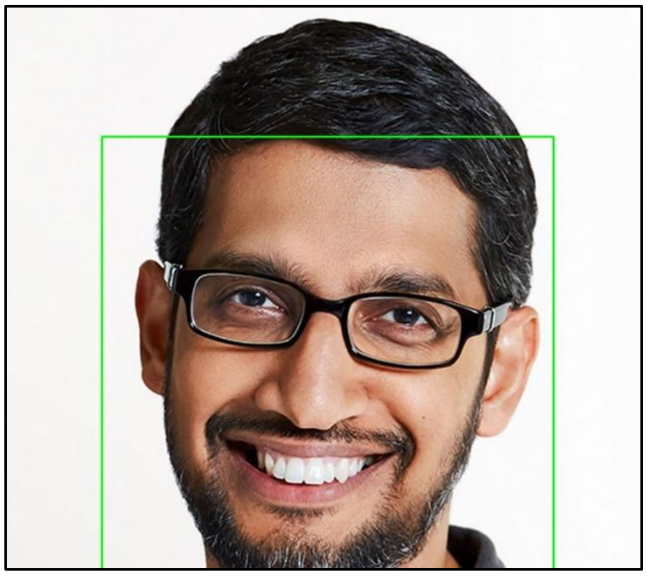
Step9: - After we plot the hex colour into the pie chart

Result and Analysis

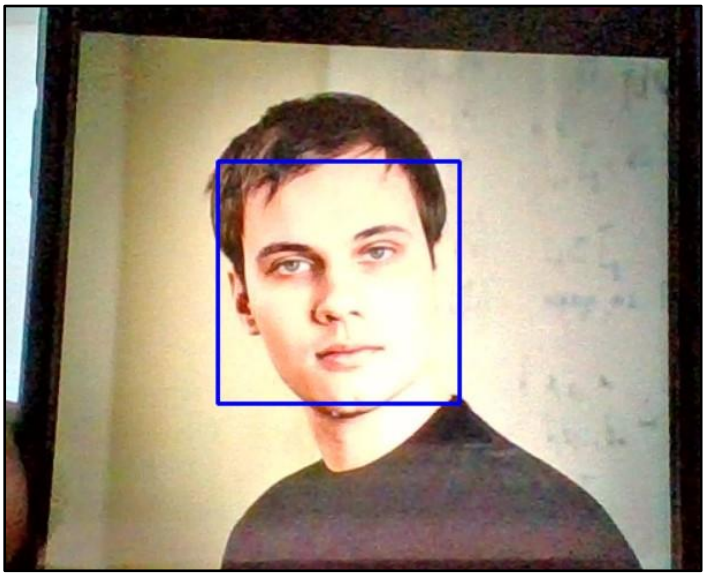
We test our model on different images we got accurate result in colour detections.



Face detection from image



Face detection from Webcam



Challenges

In the beginning of the project, I started learning about openCV2 from "freecodecamp." After learning about openCV2, I started implementing it in my project. In both models, I implement OpenCV2 for colour detection and image detection.

Learning Outcomes

In this project I learn opencv2. OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in commercial products.

References

1.) https://www.geeksforgeeks.org/numpy-in-python-set-1-introduction/
2.) https://matplotlib.org/