import cv2

import math

import argparse

import mysql.connector

from tkinter import \*

a=0

g=""

def setdata(n,ag,gen):

mydb=mysql.connector.connect(host="localhost",user="root",passwd="sql@111",database="agegen")

mycursor=mydb.cursor()

sqlformula="INSERT INTO details (name,gender,age) VALUES(%s,%s,%s)"

data=(n,gen,ag)

mycursor.execute(sqlformula,data)

mydb.commit()

def get\_data():

# print(n.get(),ag,gen)

mydb=mysql.connector.connect(host="localhost",user="root",passwd="sql@111",database="agegen")

mycursor=mydb.cursor()

mycursor.execute("SELECT \* FROM details")

myresult=mycursor.fetchall()

showdetails=Toplevel()

showdetails.config(bg="#8F43EE")

heading=Label(showdetails,text="Analyse Age and Gender",bg="#8F43EE",fg="white",font=("Berlin Sans FB",30)).place(x=600,y=30)

l=Listbox(showdetails,width=50,height=20,font=("Berlin Sans FB",20),fg="#8F43EE",bd=5)

p=2

l.insert(1," NAME GENDER AGE ")

for i,j,k in myresult:

l.insert(p,f" {i} {j} {k} ")

p=p+1

l.place(x=400,y=100)

showdetails.geometry('1920x1080')

showdetails.mainloop()

def cam(user\_name):

global a

global g

def highlightFace(net, frame, conf\_threshold=0.7):

frameOpencvDnn=frame.copy()

frameHeight=frameOpencvDnn.shape[0]

frameWidth=frameOpencvDnn.shape[1]

blob=cv2.dnn.blobFromImage(frameOpencvDnn, 1.0, (300, 300), [104, 117, 123], True, False)

net.setInput(blob)

detections=net.forward()

faceBoxes=[]

for i in range(detections.shape[2]):

confidence=detections[0,0,i,2]

if confidence>conf\_threshold:

x1=int(detections[0,0,i,3]\*frameWidth)

y1=int(detections[0,0,i,4]\*frameHeight)

x2=int(detections[0,0,i,5]\*frameWidth)

y2=int(detections[0,0,i,6]\*frameHeight)

faceBoxes.append([x1,y1,x2,y2])

cv2.rectangle(frameOpencvDnn, (x1,y1), (x2,y2), (0,255,0), int(round(frameHeight/150)), 8)

return frameOpencvDnn,faceBoxes

parser=argparse.ArgumentParser()

parser.add\_argument('--image')

args=parser.parse\_args()

faceProto="opencv\_face\_detector.pbtxt"

faceModel="opencv\_face\_detector\_uint8.pb"

ageProto="age\_deploy.prototxt"

ageModel="age\_net.caffemodel"

genderProto="gender\_deploy.prototxt"

genderModel="gender\_net.caffemodel"

MODEL\_MEAN\_VALUES=(78.4263377603, 87.7689143744, 114.895847746)

ageList=['(0-2)', '(4-6)', '(8-12)', '(15-20)', '(25-32)', '(38-43)', '(48-53)', '(60-100)']

genderList=['Male','Female']

faceNet=cv2.dnn.readNet(faceModel,faceProto)

ageNet=cv2.dnn.readNet(ageModel,ageProto)

genderNet=cv2.dnn.readNet(genderModel,genderProto)

video=cv2.VideoCapture(args.image if args.image else 0)

padding=20

while cv2.waitKey(1)!=ord('q'):

hasFrame,frame=video.read()

if not hasFrame:

cv2.waitKey()

break

resultImg,faceBoxes=highlightFace(faceNet,frame)

if not faceBoxes:

print("No face detected")

for faceBox in faceBoxes:

face=frame[max(0,faceBox[1]-padding):

min(faceBox[3]+padding,frame.shape[0]-1),max(0,faceBox[0]-padding)

:min(faceBox[2]+padding, frame.shape[1]-1)]

blob=cv2.dnn.blobFromImage(face, 1.0, (227,227), MODEL\_MEAN\_VALUES, swapRB=False)

genderNet.setInput(blob)

genderPreds=genderNet.forward()

gender=genderList[genderPreds[0].argmax()]

print(f'Gender: {gender}')

g=gender

ageNet.setInput(blob)

agePreds=ageNet.forward()

age=ageList[agePreds[0].argmax()]

print(f'Age: {age[1:-1]} years')

a=age[1:-1]

cv2.putText(resultImg, f'{gender}, {age}', (faceBox[0], faceBox[1]-10), cv2.FONT\_HERSHEY\_SIMPLEX, 0.8, (0,255,255), 2, cv2.LINE\_AA)

cv2.imshow("Detecting age and gender", resultImg)

video.release()

cv2.destroyAllWindows()

setdata(user\_name.get(),g,a)

root=Tk()

root.title('Analyse Age and Gender')

root.config(bg="#8F43EE")

user\_name=StringVar()

heading=Label(root,text="Analyse Age and Gender",bg="#8F43EE",fg="white",font=("Berlin Sans FB",30)).place(x=33,y=30)

name=Label(root,text="Full Name",bg="#8F43EE",fg="white",font=("Berlin Sans FB",30)).place(x=150,y=100)

inputname=Entry(root,textvariable=user\_name,font=("Berlin Sans FB",20),bd=5).place(x=80,y=150)

submit=Button(root,text="Find Age and Gender",bg="white",fg="#8F43EE",font=("Berlin Sans FB",20),bd=5,command=lambda:[cam(user\_name)]).place(x=120,y=200)

details=Button(root,text="Show Details",bg="white",fg="#8F43EE",font=("Berlin Sans FB",20),bd=5,command=lambda:[get\_data()]).place(x=170,y=270)

root.geometry('500x500+500+100')

root.mainloop()