GÖRÜNTÜ İŞLEME İLE DUYGU ANALİZİ

BOLU ABANT İZZET BAYSAL ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

GÖRÜNTÜ İŞLEME DERSİ

Oğuz Kağan Dönmez 203405025

Ahmet Öztürk 203405054

Emirhan Topcuoğlu 203405064

BAŞLIKLAR

- Projenin Amacı
- Projede Kullanılan Teknolojiler
- Proje İçeriği
- Gerekli Kütüphanelerin ve Kullanılacak Modelin Yüklenmesi
- Görüntüden Özellik (Feature) Çıkarmak İçin Kullanılan Fonksiyonun Tanımlanması
- Kameradan Anlık Görüntü Alma ve Yüz Tanıma

PROJENİN AMACI

Bu proje ile hedeflemekte olduğumuz adımlar şu şekildedir:

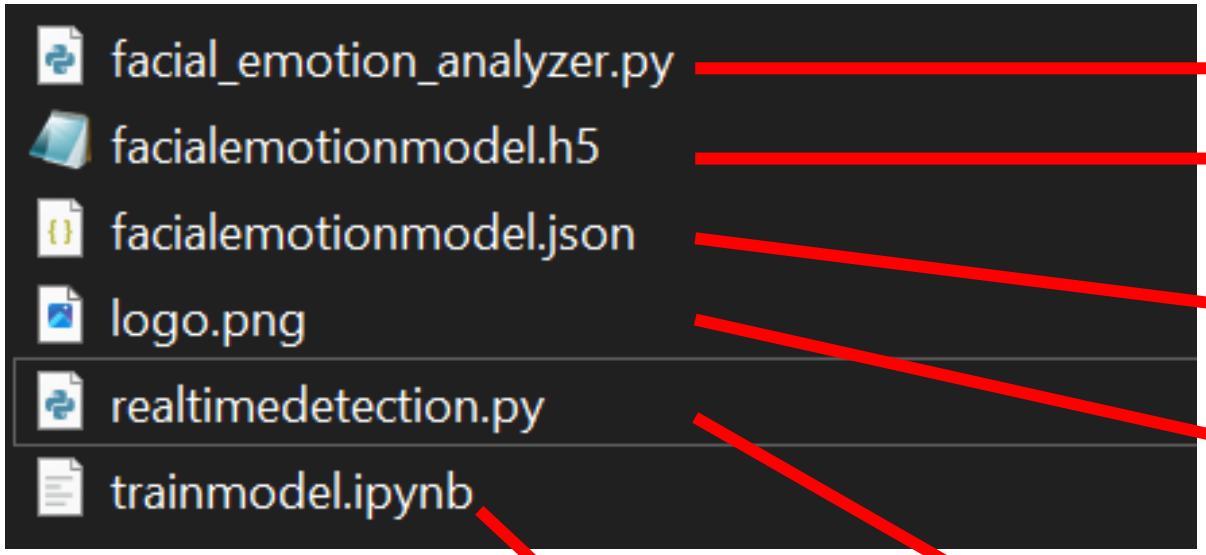
- Bilgisayar kamerasından alınan görüntüleri işleyerek insan yüzlerini tanımak.
- Tanınan yüzler üzerinde duygu analizi yaparak duygu durumunu belirlemek.
- Kullanıcı dostu bir arayüz üzerinden sonuçları göstermek ve kullanıcıyla etkileşim sağlamak.

PROJEDE KULLANILAN TEKNOLOJİLER

Projemizde kullanmış olduğumuz teknolojiler:

- Python programlama dili
- OpenCV
- PyQt5
- Tensorflow
- Keras

PROJE İÇERİĞİ



PyQt5 ile arayüz tasarımı

Yüz hareketlerini (facial motion) tanıma ve duygu analizi için eğitilmiş bir model

Modelin mimarisini tanımlayan yapısal bilgileri içerir.

Proje logosu

Kameradan anlık görüntü alarak görüntü işlemenin gerçekleştirilmesi

Modelin hazırlanması ve eğitilmesi

GEREKLİ KÜTÜPHANELERİN VE KULLANILACAK MODELİN YÜKLENMESİ

```
1   import cv2
2   from keras.models import model_from_json
3   import numpy as np
4   # from keras_preprocessing.image import load_img
5   json_file = open("facialemotionmodel.json", "r")
6   model_json = json_file.read()
7   json_file.close()
8   model = model_from_json(model_json)
9
10  model.load_weights("facialemotionmodel.h5")
11  haar_file=cv2.data.haarcascades + 'haarcascade_frontalface_default.xml'
12  face_cascade=cv2.CascadeClassifier(haar_file)
```

GÖRÜNTÜDEN ÖZELLİK (FEATURE) ÇIKARMAK İÇİN KULLANILAN FONKSİYON

```
14    def extract_features(image):
15        feature = np.array(image)
16        feature = feature.reshape(1,48,48,1)
17        return feature/255.0
```

- **image parametresi:** Bu, işlevin girdisi olarak alınan bir görüntüdür.
- **np.array(image):** Görüntü, NumPy dizisine dönüştürülür. Bu, görüntüyü daha sonra işleyebilmek için uygun bir veri yapısı sağlar.
- **feature = feature.reshape(1,48,48,1):** Görüntü dizisi, belirli bir şekle (shape) yeniden şekillendirilir. Burada, 48x48 piksel boyutunda ve tek bir renk kanalı olan bir görüntü olarak yeniden şekillendirilir. Bu, tipik olarak bir derin öğrenme modelinin girişine uygun bir formattır. Yani, görüntü bir örnekte olduğu gibi bir dizi haline getirilir.
- **return feature/255.0:** Son olarak, bu işlev, yeniden şekillendirilmiş görüntüyü 0 ile 1 arasında ölçeklemek için görüntüyü 255'e böler. Bu, görüntü piksellerinin değerlerini normalize eder ve genellikle derin öğrenme modellerinin daha iyi performans göstermesine yardımcı olur.

KAMERADAN ANLIK GÖRÜNTÜ ALMA VE YÜZ TANIMA

```
webcam=cv2.VideoCapture(0)
labels = {0 : 'angry', 1 : 'disgust', 2 : 'fear', 3 : 'happy', 4 : 'neutral', 5 : 'sad', 6 : 'surprise'}
while True:
    i,im=webcam.read()
    gray=cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
    faces=face_cascade.detectMultiScale(im,1.3,5)
    try:
        for (p,q,r,s) in faces:
            image = gray[q:q+s,p:p+r]
            cv2.rectangle(im,(p,q),(p+r,q+s),(255,0,0),2)
            image = cv2.resize(image,(48,48))
            img = extract_features(image)
            pred = model.predict(img)
            prediction_label = labels[pred.argmax()]
            # print("Predicted Output:", prediction_label)
            # cv2.putText(im,prediction_label)
            cv2.putText(im, '% s' %(prediction_label), (p-10, q-10),cv2.FONT_HERSHEY_COMPLEX_SMALL,2, (0,0,255))
        cv2.imshow("Output",im)
        cv2.waitKey(27)
    except cv2.error:
        pass
```

KAMERADAN ANLIK GÖRÜNTÜ ALMA VE YÜZ TANIMA

- **cv2.VideoCapture(0):** Webcam'i başlatmak için OpenCV'nin VideoCapture sınıfı kullanılır. Parametre olarak "0" verilerek bilgisayarınızın birincil kamera cihazına erişilir.
- **labels:** Duygu tahminlerini anlamlı etiketlerle eşlemek için bir sözlük tanımlanır.
- **while True:** Sonsuz bir döngü oluşturulur, bu da webcam'den sürekli olarak görüntü alınmasını sağlar.
- **webcam.read():** Webcam'den bir kare alır.
- **cv2.cvtColor(im,cv2.COLOR_BGR2GRAY):** Alınan kare, gri tonlamalı bir görüntüye dönüştürülür. Bu genellikle yüz tanıma algoritması için daha iyi performans sağlar.

KAMERADAN ANLIK GÖRÜNTÜ ALMA VE YÜZ TANIMA

- **face_cascade.detectMultiScale(im,1.3,5):** Gri tonlamalı görüntüde yüzleri tespit etmek için kullanılan bir yüz tespitme algoritması olan "detectMultiScale" fonksiyonu çağrılır.
- **for (p,q,r,s) in faces:** Tespit edilen her yüz için bir döngü oluşturulur ve yüzlerin koordinatları (p, q) ve boyutları (r, s) alınır.
- **image = gray[q:q+s,p:p+r]:** Yüz bölgesi gri tonlamalı görüntüden alınır.
- **cv2.rectangle(im,(p,q),(p+r,q+s),(255,0,0),2):** Her yüzün etrafına bir dikdörtgen çizilir.
- **image = cv2.resize(image,(48,48)):** Yüz görüntüsü, duygu tanıma modeline beslemek için 48x48 piksel boyutuna yeniden boyutlandırılır.
- **img = extract_features(image):** Yüz görüntüsünden özellikler çıkarılır (örneğin, normalize edilir).
- **pred = model.predict(img):** Model, verilen görüntü için bir tahmin yapar.

KAMERADAN ANLIK GÖRÜNTÜ ALMA VE YÜZ TANIMA

- **prediction_label = labels[pred.argmax()]:** Modelin tahmin ettiği duyguyu etiketler sözlüğünden alır.
- **cv2.putText(im, '% s' %(prediction_label), (p-10, q-10),cv2.FONT_HERSHEY_COMPLEX_SMALL,2, (0,0,255)):** Tanımlanan yüzün etrafına, tahmin edilen duygu etiketi yazılır.
- **cv2.imshow("Output",im):** Görüntü işlenir ve ekranda gösterilir.
- **cv2.waitKey(27):** "ESC" tuşuna basıldığında döngüyü kırar ve programı sonlandırır.
- **except cv2.error:** Herhangi bir OpenCV hatası alındığında, kod devam eder ve geçer. Bu, hata oluştuğunda programın çökmesini önler.

DİNLEDİĞİNİZ İÇİN TEŞEKKÜR EDERİZ