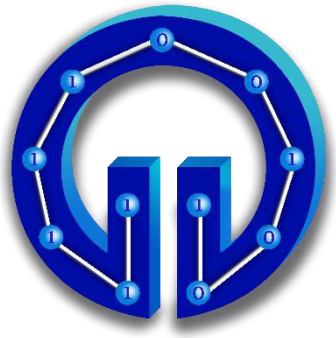
KARADENİZ TEKNİK ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
WİNDOWS PROGRAMLAMA DERSİ



MOBİL CİNSİYET VE YAŞ TAHMİNİ UYGULAMASI

DÖNEM ÖDEVİ RAPORU

338396 Orkhan ALİYEV
330202 Furkan ÖNER

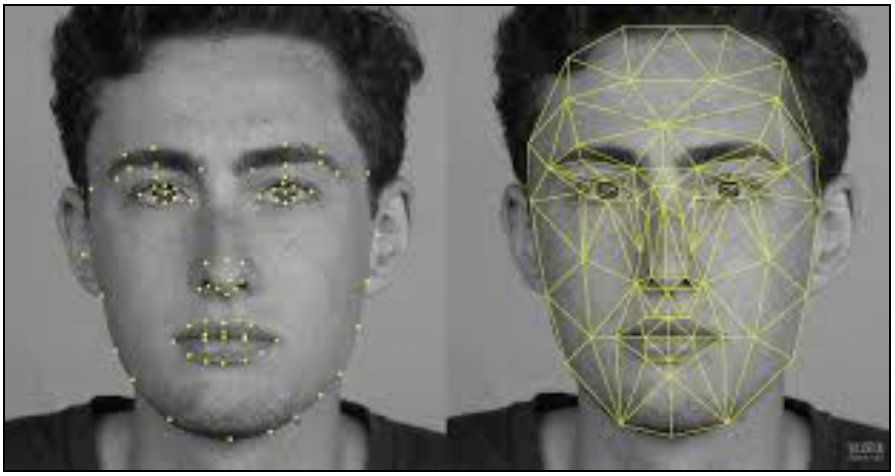
2019-2020 GÜZ DÖNEMİ

İÇİNDEKİLER

1. GENEL BİLGİLER

1.1. Giriş

Günümüz teknolojilerinde yüz tanıma sistemleri insan hayatında önemli bir yer kaplamaktadır. Görüntü işleme algoritmalarının gelişmesi ile yükselişe geçen yüz tanıma sistemleri bankacılıkta, günlük hayatta, güvenlik ve askeri sistemlerde kullanılmaktadır. Hemen hemen 1960'lara dayanan yüz tanıma sistemleri ilk kez *Woodrow Wilson Bledsoe* tarafından geliştirilmiştir. O zamandan beri gelişen yüz tanıma sistemleri bir çok alt başlıklara ayrılmıştır. Sadece yüz özelliklerine bakılarak kişilerin cinsiyet ve yaş tahminleri yapılabilmektedir. Derin öğrenmenin gelişmesiyle de bu tahminlerin doğruluk oranları büyük ölçekte artmıştır. Bu proje yukarıda bahsedilen iki alt başlığın tek bir ana başlık altında gerçeklenmesine ve mobil platforma uygun bir şekilde uyarlanmasına dayanmaktadır.



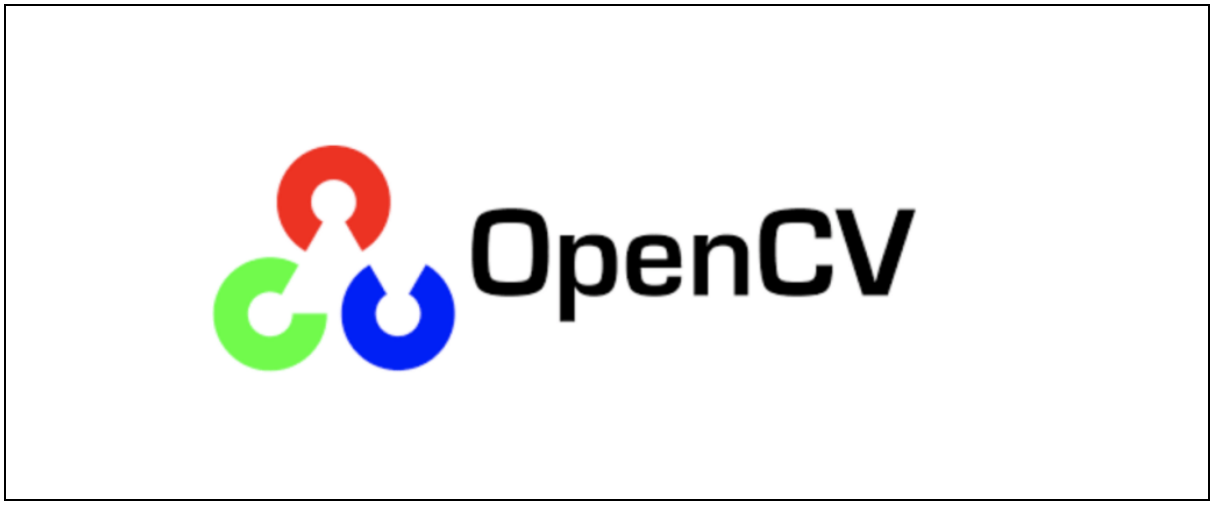
1.2 Kullanılan Teknolojiler

Projede görüntü işleme algoritmalarını gerçekleyebilmek için OpenCV kütüphanesinden, cinsiyet ve yaş tahmini yapılabilmesi için Caffe modellerinden yararlanılmıştır. Projenin mobil arayüzünde ise Python Kivy arayüz programlama framework'ü kullanılmıştır.

2. PROJE TASARIMI

2.1 OpenCV

OpenCV(Open Source Computer Vision Library), esas olarak gerçek zamanlı bilgisayar görmeyi hedefleyen programlama fonksiyonlarının görüntü işleme kütüphanesidir . C++, Python ve Java arayüzlerine sahiptir ve Windows, Linux, Mac OS, iOS ve Android'i destekler. Kullanıldığı alanlar yüz tanıma, işaret dili tanıma, hareket yakalama gibi görüntü işleme algoritmalarında sıklıkla kullanılır. 1999 yılında intel firması ilk sürümünü piyasaya sürmüştür. SourceForge geliştirilmesine devam etmektedir.



2.2 CaffeModel

Caffe derin öğrenme yapısı hızlı ve modüler olacak şekilde tasarlanmıştır. Berkeley Vision and Learning Center – BVLC (Berkeley Görüntü ve Öğrenme Merkezi) ve kullanıcı topluluğu tarafından geliştirilmiştir. Yangqing Jia tarafından UC Berkeley'de doktora döneminde hazırlanmıştır. Caffe BSD 2-Clause license altında kullanıma sunulmuştur. GPU makine üzerinde eğitim işlemini yapmak için CPU ve GPU değişimi bir etiket ayarı ile gerçekleştirilebilmekte böylece küme bilgisayarlara veya mobil cihazlara yayılım sağlanabilmektedir. Genişletilebilir kod yapısı aktif geliştirmeyi desteklemektedir.

2.3 Yüz Algılama

Yüz tanıma teknolojisi minimum düzeyde sorun üreten ve en hızlı biyometrik teknolojidir. En belirgin bireysel kimlikle çalışır: İnsan yüzü. Yüz tanıma alanındaki yapılan çalışmalarda alınan başarıların neticesinde yüz tanıma teknolojisi son yıllarda büyük atılım göstermiştir. Yüz tanıma teknolojisi ile kart kaybetme, şifre unutma gibi sıkıntılar tamamen ortadan kalkmış ve kullanıcılar için büyük bir kullanım kolaylığı sağlanmıştır. İnsanların ellerini bir okuyucuya yerleştirmeleri ya da gözleriyle bir tarayıcıya bakmak zorunda olmaları yerine yüz tanıma sistemi belirlenen alanlarda kişilerin resmini sessizce çeker. Yüz tanıma sistemi bir dijital video kamera ile bir kişinin yüz görüntülerini analiz eder. Gözler, burun, ağız ve çene kenarlarındaki mesafeler de dahil olmak üzere bütün yüz yapısını ölçer. Bu ölçümler bir veritabanında saklanır ve bir kullanıcı kamera önüne geldiği zaman yapılacak karşılaştırmalar için kullanılır. Her yüz farklı karakteristik özelliklere sahiptir. Her insan yüzü yaklaşık 80 düğüm noktasına sahiptir. Yüz tanıma teknolojisiyle gözler arasındaki mesafe, burun genişliği, göz çukurlarının derinliği, elmacık kemiklerinin şekli, çene hattının uzunlukları vs. ölçülür.

2.4 Cinsiyet ve Yaş Tahmini

Daha önce bu alanda yapılmış araştırmalar doğrultusunda PyTorch, TensorFlow, CaffeModel üzerinden cinsiyet ve yaş tahmini yapabilen modeller açık kaynak olarak paylaşılmaktadır. Cinsiyet ve yaş tahmini yapan sistemler genellikle CaffeModel üzerinden üretilmiş hazır modelleri kullanmaktadır.Bu proje OpenCV'nin geliştirdiği DNN (Deep Neural Network) ile kolaylıkla bu modelleri yükleyip kullanmaktadır.

2.5 Kivy

Kivy mobil cihazlarda da çalışabilecek programların yazılabileceği bir Python modülüdür. Diğer bir deyişle Mobil GUI Toolkit (Mobil Grafik Kullanıcı Arayüzü Aracı) diyebiliriz. Python ile Mobil Uygulama geliştirmek isteyenlerin çoğunluğu Kivy'i tercih etmektedir. Kivy ile yazacağınız programlar hemen her platformda çalışabilir. Bu platformları şöyle sıralayabiliriz;

- Masaüstü Bilgisayarlar: Linux, Mac OS X, Windows
- Tabletler: Android cihazlar, iPad, iPhone

Kivy dokunmatik ekranlar için optimize edilmiş olmasına rağmen, geliştirilen uygulamalar masaüstü bilgisayarlarda da rahatlıkla çalışabilmektedir. Bununla birlikte masaüstü bilgisayarlarda kullanılan diğer GUI araçlarındaki birçok özelliği bulma şansınız yok. Kivy aslında Pygame üzerine kurulmuş bir yapıdır. Tüm widgetler (grafik bileşenleri) Pygame ile çizilmektedir. Kivy ile yazdığınız programlar, bir Linux makina (veya sanal makinada çalışan bir Linux) ile kolaylıkla Android paketleri haline getirilebilmektedir. Getirilen paketler içerisinde Python ve diğer bileşenler eklendiğinden, uygulama paketi kurulduğunda başka herhangi bir eklentiye gerek kalmadan çalışmaktadır.

2.6 Projenin Gerçeklenmesi

```
import cv2 as cv
import time
import math
import random
```

Projenin başlangıcında gerekli kütüphaneleri tanımlıyoruz. OpenCV modülünü indirmek için;

```
pip install opencv-python
```

komutunu koşuyoruz.Kivy'i projeye dahil etmek için de;

```
pip install Kivy
```

komutunu koşuyoruz.

```
from kivy.app import App
from kivy.lang import Builder
from kivy.uix.widget import Widget
from kivy.uix.boxlayout import BoxLayout
from kivy.uix.image import Image
from kivy.clock import Clock
from kivy.uix.button import Button
from kivy.uix.label import Label
from kivy.graphics.texture import Texture
from kivy.uix.camera import Camera
from kivy.config import Config
from kivy.uix.togglebutton import ToggleButton
```

Kullanacağımız kivy araçlarının tanımlamasını projeye dahil ediyoruz. Mobil uygulamanın arayüz boyutlarının setlemesini aşağıdaki gibi gerçekleştiriyoruz.

```
Config.set('graphics', 'width', '360')
Config.set('graphics', 'height', '600')
```

Girdi olarak alınan karelerde yüz tespit edilmesi durumunda yüzlerin kare içerisine alınıp noktasal koordinatlarını döndürmek için fonksiyon:

```
def getFaceBox(net, frame, conf_threshold=0.7):
     frameOpencvDnn = frame.copy()
    frameHeight = frameOpencvDnn.shape[0]
    frameWidth = frameOpencvDnn.shape[1]
```

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

    net.setInput(blob)
    detections = net.forward()
    bboxes = []
    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)
            bboxes.append([x1, y1, x2, y2])
            cv.rectangle(frameOpencvDnn, (x1, y1), (x2, y2),
                         (255, 0, 0), int(round(frameHeight/150)), 8)
    return frameOpencvDnn, bboxes
```

Yüz modeli ve bu modelin şeması projeye dahil edilir.

```
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"
```

Modellerin ortalama değerleri, cinsiyet ve yaş aralıkları statik olarak tanımlanır.

```
MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744, 114.895847746)
ageList = ['(0-3)', '(4-7)', '(8-14)', '(15-20)',
           '(21-35)', '(36-45)', '(46-56)', '(57-100)']
genderList = ['Erkek', 'Kadin']
```

Yapay sinir ağı girdiler verilerek tanımlanır .

```
ageNet = cv.dnn.readNet(ageModel, ageProto)
genderNet = cv.dnn.readNet(genderModel, genderProto)
faceNet = cv.dnn.readNet(faceModel, faceProto)
```

OpenCV'nin kamera erişimi için fonksiyonu çağırılır.

```
cap = cv.VideoCapture(0)
padding = 20
```

Ana arayüz penceresi **`BoxLayout`** formatında tanımlanıyor.

```
class MainWindow(BoxLayout):
    def __init__(self, **kwargs):
        super(MainWindow, self).__init__(**kwargs)
```

CamApp sınıfı tanımlanıyor.

```
class CamApp(App):
```

"Initialization" fonksiyonu aracılığıyla arayüz pencereleri ve nesneleri tanımlanır:

```
def build(self):
        self.camera = Camera()

        superLayout = MainWindow(orientation='vertical')
        verticalLayout = MainWindow(orientation='vertical')

        verticalLayout.add_widget(self.camera)
        self.lblOutput = Label(text="yukleniyor...",
                               font_size=10,
                               color=(0, 0, 1, 1),
                               size=(50, 50),
                               size_hint=(.4, .1))

        verticalLayout.add_widget(self.lblOutput)

        superLayout.add_widget(verticalLayout)
        self.capture = cap
        Clock.schedule_interval(self.update, 1.0/13.0)
        return superLayout
```

Yüz algılanmadığı zaman çalışacak fonksiyon:

```
def no_face(self):
        self.lblOutput.text = "Yuz algilanmadi, bir sonraki kare kontrol
ediliyor"
```

Her kareyi **.png** formatında döndürecek fonksiyon:

```
def capture(self):
        timestr = time.strftime("%Y%m%d_%H%M%S")
        self.camera.export_to_png("IMG_{}.png".format(timestr))
```

Çıkarılan sonuçlar doğrultusunda yaş ve cinsiyet tahminlerini ekrana yazan fonksiyon:

```
def output(self, gender, genderPreds, age, agePreds):
        self.lblOutput.text = "Cinsiyet : {}, oran = {:.3f}".format(
            gender, genderPreds[0].max()) + "\n" + "Yas : {}, oran =
{:.3f}".format(
            age, agePreds[0].max())
```

Her karede çalışacak fonksiyon :

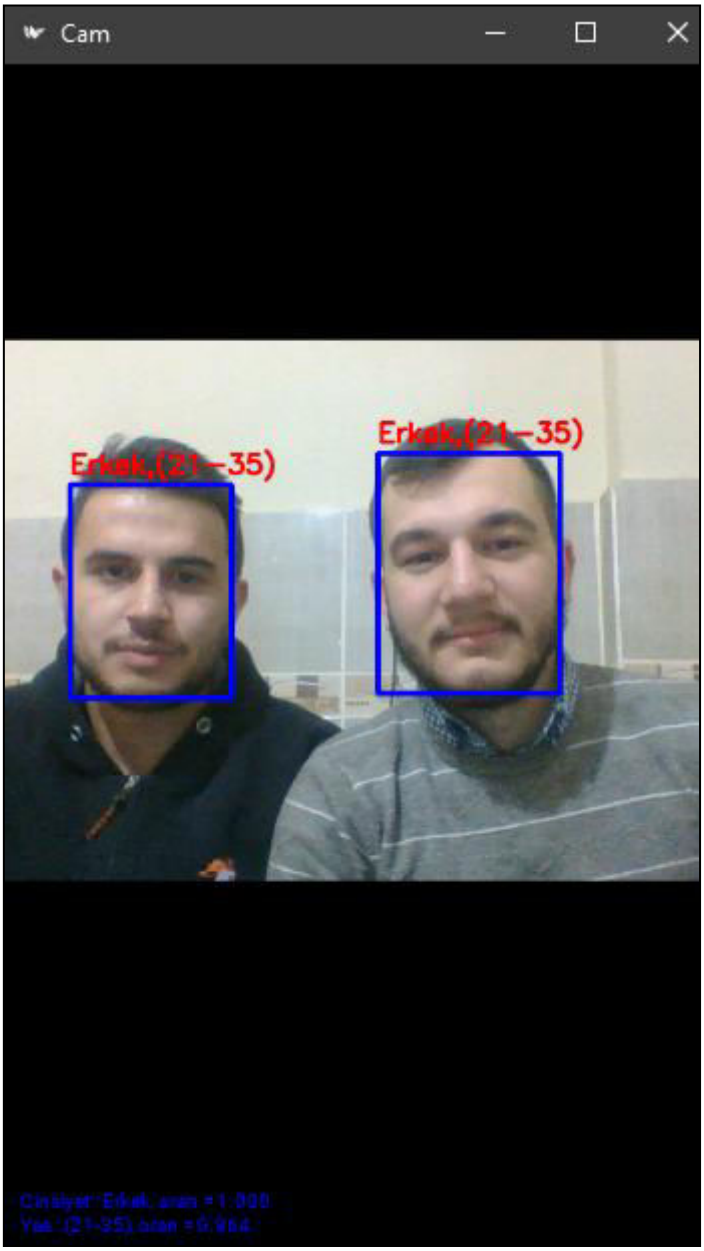
```
def update(self, dt):

        ret, frame = self.capture.read()
        frameFace, bboxes = getFaceBox(faceNet, frame)
        if not bboxes:
            self.no_face()

        for bbox in bboxes:
            face = frame[max(0, bbox[1]-padding):min(bbox[3]+padding,
frame.shape[0]-1), max(
                0, bbox[0]-padding):min(bbox[2]+padding, frame.shape[1]-1)]

            blob = cv.dnn.blobFromImage(
                face, 1.0, (227, 227), MODEL_MEAN_VALUES, swapRB=False)
            genderNet.setInput(blob)
            genderPreds = genderNet.forward()
            gender = genderList[genderPreds[0].argmax()]

            ageNet.setInput(blob)
            agePreds = ageNet.forward()
            age = ageList[agePreds[0].argmax()]
            #print("Age Output : {}".format(agePreds))
            self.output(gender, genderPreds,age,agePreds)

            label = "{},{}".format(gender, age)
            cv.putText(frameFace, label, (bbox[0], bbox[1]-10),
                  cv.FONT_HERSHEY_SIMPLEX, 0.8, (0, 255, 255), 2, cv.LINE_AA)
```

Bu fonksiyon öncelikle **`capture`**() fonksiyonunun döndürdüğü kareyi okur. Alınan kare parametre olarak **`getFaceBox()`** fonksiyonuna gönderilir ve yüz tespiti yapılır. **`getFaceBox()`** fonksiyonu algılanan yüzleri döndürür eğer yüz bulunmaz ise **`no_face()`** fonksiyonu çağırılır. Yüz algılanması olasılığında for döngüsü ile tüm yüzlerin içerisinde gezilir. Kare içerisine alınan yüzlerin noktasal koordinatları bir dizi içerisinde tutularak **`cv.dnn.blobFromImage()`** fonksiyonuna parametre olarak gönderilir. Fonksiyonun döndürdüğü **blob** (matematiksel olarak n boyutlu bir dizi) içerisine atanır. Bu **blob** parametre olarak **genderNet** ve **ageNet** için girdi değerleri olarak setlenir. Her ikisi için tahminler yapılır ve oranı en fazla olan argüman döndürülür. Döndürülen yaş ve cinsiyet tahminleri **output**() fonksiyonuna gönderilir ve ekranda yazdırılır.

Kameradan alınan karenin ekranda dokusal olarak işlenmesi:

```
buf1 = cv.flip(frameFace, 0)
buf = buf1.tostring()
texture1 = Texture.create(
size=(frame.shape[1], frame.shape[0]), colorfmt='bgr')
texture1.blit_buffer(buf, colorfmt='bgr', bufferfmt='ubyte')
self.camera.texture = texture1
```

2.7 Sonuç



Sonuç çıktı yukarıdaki gibidir. Kamera erişimi otomatik yapılmıştır(***OpenCV VideoCapture()***)

Uygulamaya erişim;
https://github.com/aliyevorkhan/Mobile_Age_Gender_Recognition
bağlantısı üzerinden sağlanabilir.

2.8 Kaynaklar

[1] - https://ieeexplore.ieee.org/document/5455084(Face Recognition System and It's Application -Xuehong Tian)

[2] - https://docs.opencv.org/2.4/ (OpenCV Dökümanı ve Kullanım Kılavuzu)

[3] - https://caffe.berkeleyvision.org/ (Berkley University Computer Vision FrameWork)

[4] - https://kivy.org/#home (Kivy Dökümanı ve Kullanım Kılavuzu)

[5] - https://www.python.org/doc/ (Python Dökümanı ve Kullanım Kılavuzu)