YAPAY ZEKAYI BİLSEM

Erasmus+

"YAPAY ZEKAYI BİLSEM" ETKİNLİK KİTAPÇIĞI

2020-1-TR01-KA101-086521

Erasmus+ Okul Eğitimi Personel Hareketliliği Konsorsiyum Projesi

Co-funded by the Erasmus+ Programme of the European Union

YAPAY ZEKAYI BİLSEM
PROJE HAKKINDA / ÖNSÖZ

Avrupa Komisyonu Yapay Zeka Yaklaşımı'na göre; ekonomik devrimlere neden olan buhar motoru ve elektrik gibi, toplumu dönüştürüyor olarak görülen Yapay Zeka, ekonomik kalkınmanın temel itici gücü haline gelmiş, stratejik öneme sahip dijital bir alandır. Ulusal bağlamda her alanda gelişim rotamızı belirleyen 11.Kalkınma Planı, küresel eğilimleri ve Türkiye etkileşimini belirtirken; "Teknolojinin hız kazanmasına bağlı olarak değişen ihtiyaçlar için becerilerin edinilmesine yönelik hayat boyu öğrenme yaklaşımı her alanda çeşitlenerek yaygınlaşmakta" şeklinde değişen teknoloji ile artan belirsizlikler ve yeni arayışlar ifade edilmiştir. Ayrıca plandaki "Kritik Teknolojiler" yatay politika alanında dijital teknoloji ve yeterlilikle ilgili "Öncelikli alanlarda teknolojik dönüşümün sağlanabilmesi ve rekabet gücünün artırılması amacıyla önümüzdeki dönemde yüksek katma değer oluşturması beklenen kritik teknoloji alanlarında teknoloji üretme ve adaptasyon yeteneğinin geliştirilmesi temel amacı kapsamında Ülkemizde Milli Teknoloji Hamlesi'nin gerçekleştirilmesine yönelik; yapay zeka, nesnelerin interneti, artırılmış gerçeklik, büyük veri, siber güvenlik, robotik teknolojilerine ilişkin gelişim yol haritalarının hazırlanması, ihtiyaç duyulan nitelikli insan kaynağının yetiştirilmesi ve toplumsal yönelimin bu alanlara odaklanması sağlanmakla birlikte kritik teknolojilerde insan gücü kapasitesi artırılacak." politikaları planlanmıştır.

T.C. Milli Eğitim Bakanlığı yatırımı ve sponsorlar aracılığıyla, 7 konsorsiyum üyesinin yer aldığı 14 Bilim ve Sanat Merkezi'nde Yapay Zeka atölyeleri kurulmuştur. Böyle mevcut dijital teknolojik kaynakları etkin kullanarak konsorsiyum okullarında dijital içerikli disiplinler arası katma değer sağlayan projelerin üretilmesini sağlamak stratejimiz doğrultusunda, Avrupa Komisyonu ve Türkiye Ulusal Ajansı'nın Erasmus+ programı kapsamında sağladığı hibe desteği ile Hadiye Kuradacı BİLSEM koordinatörlüğünde; Trabzon Faruk Başaran BİLSEM, Çetin Şen BİLSEM, Fahrettin Kırzıoğlu BİLSEM, Halil İnalcık BİLSEM, Diyarbakır BİLSEM, Karşıyaka Aydoğan Yağcı BİLSEM ortaklığında 7 Bölge 7 BİLSEM sloganıyla bu konsorsiyum projesi gelişmiştir.

Dijital içerikli disiplinler arası katma değer sağlayan projeler üretmek genel amacı doğrultusunda hazırlanan 12 aylık bu projenin amacı; 21 konsorsiyum üyesi personelin Avrupa'da 10 günlük kurs yoluyla derin öğrenme Yapay Zeka alanında mesleki bilgi, beceri ve yeterlilik geliştirmesini sağlamaktı. Almanya'da Derin Öğrenme Yapay Zeka alanında kursa katılan katılımcılar, öğrenme ve öğretmede daha fazla dijital teknolojileri kullanmak üzere konu alanında mesleki bilgi ve beceri edinmişlerdir. Ayrıca disiplinler arası çalışmaların artmasıyla katma değer yaratan projelerin ortaya çıkması için pedagojik olarak da çeşitli kazanımlar elde etmişlerdir.

Projenin yaygın etkisini artırmak ve ilgililerin alanla ilgili kapasitelerini geliştirmelerine fırsat sunmak amacıyla konsorsiyum tarafından geliştirilen ve çeşitli Yapay Zeka etkinliklerini içeren bu somut çıktıyı Yapay Zeka alanında çalışma yapan / alana ilgi duyan kurum / kuruluş ve kişilerin ilgisine sunmaktan onur duyarız.

Konsorsiyum Üyesi Kurumlar

Web Sitesi: http://www.yapayzekayibilsem7.com/
Facebook: Yapay Zekayı BİLSEM
Twitter: @YapayZekay
İnstagram: @yapayzekayibilsem7

Co-funded by the Erasmus+ Programme of the European Union

KATKIDA BULUNANLAR

SIRA NO	OKULU	KATILIMCILAR
1	Hadiye Kuradacı Bilim ve Sanat Merkezi	Hüseyin GÜREL
2	Hadiye Kuradacı Bilim ve Sanat Merkezi	Mahmut KÜÇÜKOĞLU
3	Hadiye Kuradacı Bilim ve Sanat Merkezi	Mustafa Çağlar YORULMAZ
4	Çetin Şen Bilim ve Sanat Merkezi	Tuğba KARAOĞLU
5	Çetin Şen Bilim ve Sanat Merkezi	Hüseyin Başar TURHAN
6	Çetin Şen Bilim ve Sanat Merkezi	Suat ŞAHİN
7	Trabzon Faruk Başaran Bilim ve Sanat Merkezi	Fatih GÜNGÖR
8	Trabzon Faruk Başaran Bilim ve Sanat Merkezi	Kadir İSMAİLOĞLU
9	Trabzon Faruk Başaran Bilim ve Sanat Merkezi	Fatma ATASU
10	Diyarbakır Bilim ve Sanat Merkezi	Turğay ÇEKEN
11	Diyarbakır Bilim ve Sanat Merkezi	Mehmet KAYA
12	Diyarbakır Bilim ve Sanat Merkezi	Sait ATLI
13	Halil İnalcık Bilim ve Sanat Merkezi	Hasibe ÖZER
14	Halil İnalcık Bilim ve Sanat Merkezi	Taliha KELEŞ
15	Halil İnalcık Bilim ve Sanat Merkezi	İzzet UZUNARDALI
16	Fahrettin Kırzıoğlu Bilim ve Sanat Merkezi	Sevinç ÖZDEMİR
17	Fahrettin Kırzıoğlu Bilim ve Sanat Merkezi	Cumali Güney SABUR
18	Fahrettin Kırzıoğlu Bilim ve Sanat Merkezi	Soner KAÇAN
19	Karşıyaka Aydoğan Yağcı Bilim ve Sanat Merkezi	Selcan KAYAHAN
20	Karşıyaka Aydoğan Yağcı Bilim ve Sanat Merkezi	Emre TOSUN
21	Karşıyaka Aydoğan Yağcı Bilim ve Sanat Merkezi	Arzu ŞAHİN

SIRA NO	OKULU	MERKEZ MÜDÜRÜ
1	Hadiye Kuradacı Bilim ve Sanat Merkezi	Emine ÖZDEMİR
2	Çetin Şen Bilim ve Sanat Merkezi	Sezer İNAN
3	Trabzon Faruk Başaran Bilim ve Sanat Merkezi	Abdulvahap ALTIN
4	Fahrettin Kırzıoğlu Bilim ve Sanat Merkezi	Ali Kasım BULUT
5	Karşıyaka Aydoğan Yağcı Bilim ve Sanat Merkezi	Emine ECE GÜLEÇ
6	Diyarbakır Bilim ve Sanat Merkezi	Mehmet Raci AKSOY
7	Halil İnalcık Bilim ve Sanat Merkezi	Erol ERTÜRK

Co-funded by the Erasmus+ Programme of the European Union

ETKİNLİK NO: 1

Ders	Bilişim Teknolojileri ve Yazılım
Program	ÖYG, PROJE
Modül / Öğrenme Alanı	Derin Öğrenme
Önerilen Süre	2 Ders Saati
Öğrenci Kazanımları	• Görüntü işlemede kullanılacak Tensorflow, OS, matplotlib kütüphanesini bilir.
Öğretme-Öğrenme Yöntem ve Teknikleri	Soru Cevap, Düz Anlatım, Gösterip Yaptırma, Uygulama
Araç-Gereçler ve Kaynaklar	• Python geliştirme ortamının kurulu olduğu internet bağlantılı bilgisayar (öğrenci sayısı kadar) • Colab Yazılımı • Etkileşimli Tahta veya Projeksiyon • Keras Örnek Uygulamalar
Öğretme-Öğrenme Süreci	Bilgisayarlara internetin bağlı olması ve Keras kütüphanesi derin öğrenme örnek uygulamalar sayfası üzerinden derin öğrenme uygulamasının Colab bağlantı linki ile açılması gerekmektedir. https://colab.research.google.com/github/keras-team/keras-io/blob/master/examples/vision/ipynb/image_classification_from_scratch.ipynb Ardından uygulamaya ilişkin işlem adımları şu şekilde olmalıdır: Adım-1: Kütüphanelerin içeri aktarılması ▾ Setup `[ ] import tensorflow as tf` `from tensorflow import keras` `from tensorflow.keras import layers` Adım-2: Ham veri arşivinin indirilmesi

Load the data: the Cats vs Dogs dataset

Raw data download

First, let's download the 786M ZIP archive of the raw data:

```
[ ]  !curl -O https://download.microsoft.com/download/3/E/1/3E
```

```
[ ]  !unzip -q kagglecatsanddogs_3367a.zip
     !ls
```

Now we have a `PetImages` folder which contain two subfolders, `Ca`

```
[ ]  !ls PetImages
```

Adım-3: Ham verilerdeki bozuk verilerin filtrelenmesi

Filter out corrupted images

When working with lots of real-world image data, corrupted images are a comr
feature the string "JFIF" in their header.

```
[ ]  import os

     num_skipped = 0
     for folder_name in ("Cat", "Dog"):
         folder_path = os.path.join("PetImages", folder_name)
         for fname in os.listdir(folder_path):
             fpath = os.path.join(folder_path, fname)
             try:
                 fobj = open(fpath, "rb")
                 is_jfif = tf.compat.as_bytes("JFIF") in fobj.peek(10)
             finally:
                 fobj.close()

             if not is_jfif:
                 num_skipped += 1
                 # Delete corrupted image
                 os.remove(fpath)

     print("Deleted %d images" % num_skipped)
```

Adım-4: Verilerin görselleştirilmesi ve eğitim verisindeki 9 görüntünün etiket değeri(Köpek :1 ve kedi:0)

Visualize the data

Here are the first 9 images in the training dataset. As you can see, label 1

```
[ ]  import matplotlib.pyplot as plt

     plt.figure(figsize=(10, 10))
     for images, labels in train_ds.take(1):
         for i in range(9):
             ax = plt.subplot(3, 3, i + 1)
             plt.imshow(images[i].numpy().astype("uint8"))
             plt.title(int(labels[i]))
             plt.axis("off")
```

Adım-5: Veri çoğaltılması ve çoğaltılan görüntünün görüntülenmesi

Using image data augmentation

When you don't have a large image dataset, it's a good practice to artificially in
transformations to the training images, such as random horizontal flipping or s
aspects of the training data while slowing down overfitting.

```
[ ]  data_augmentation = keras.Sequential(
         [
             layers.RandomFlip("horizontal"),
             layers.RandomRotation(0.1),
         ]
     )
```

Let's visualize what the augmented samples look like, by applying `data_augmer`

```
[ ]  plt.figure(figsize=(10, 10))
     for images, _ in train_ds.take(1):
         for i in range(9):
             augmented_images = data_augmentation(images)
             ax = plt.subplot(3, 3, i + 1)
             plt.imshow(augmented_images[0].numpy().astype("uint8"))
             plt.axis("off")
```

Adım-6: Performans için veri kümesinin yapılandırılması

Configure the dataset for performance

Let's make sure to use buffered prefetching so we can yie

```
[ ]  train_ds = train_ds.prefetch(buffer_size=32)
     val_ds = val_ds.prefetch(buffer_size=32)
```

Adım-7: Bir model oluşturma

```
def make_model(input_shape, num_classes):
    inputs = keras.Input(shape=input_shape)
    # Image augmentation block
    x = data_augmentation(inputs)

    # Entry block
    x = layers.Rescaling(1.0 / 255)(x)
    x = layers.Conv2D(32, 3, strides=2, padding="same")(x)
    x = layers.BatchNormalization()(x)
    x = layers.Activation("relu")(x)

    x = layers.Conv2D(64, 3, padding="same")(x)
    x = layers.BatchNormalization()(x)
    x = layers.Activation("relu")(x)

    previous_block_activation = x  # Set aside residual

    for size in [128, 256, 512, 728]:
        x = layers.Activation("relu")(x)
        x = layers.SeparableConv2D(size, 3, padding="same")(x)
        x = layers.BatchNormalization()(x)

        x = layers.Activation("relu")(x)
        x = layers.SeparableConv2D(size, 3, padding="same")(x)
        x = layers.BatchNormalization()(x)

        x = layers.MaxPooling2D(3, strides=2, padding="same")(x)
```

```
            # Project residual
            residual = layers.Conv2D(size, 1, strides=2, padding="same")(
                previous_block_activation
            )
            x = layers.add([x, residual])  # Add back residual
            previous_block_activation = x  # Set aside next residual

        x = layers.SeparableConv2D(1024, 3, padding="same")(x)
        x = layers.BatchNormalization()(x)
        x = layers.Activation("relu")(x)

        x = layers.GlobalAveragePooling2D()(x)
        if num_classes == 2:
            activation = "sigmoid"
            units = 1
        else:
            activation = "softmax"
            units = num_classes

        x = layers.Dropout(0.5)(x)
        outputs = layers.Dense(units, activation=activation)(x)
        return keras.Model(inputs, outputs)


    model = make_model(input_shape=image_size + (3,), num_classes=2)
    keras.utils.plot_model(model, show_shapes=True)
```

Adım-8: Model eğitme:

```
epochs = 50

callbacks = [
    keras.callbacks.ModelCheckpoint("save_at_{epoch}.h5"),
]
model.compile(
    optimizer=keras.optimizers.Adam(1e-3),
    loss="binary_crossentropy",
    metrics=["accuracy"],
)
model.fit(
    train_ds, epochs=epochs, callbacks=callbacks, validation_data=val_ds,
)
```

	Adım-9: Eğitilmiş modelin yeni veriler ile başarımının test edilmesi `img = keras.preprocessing.image.load_img(` `"PetImages/Cat/6779.jpg", target_size=image_size` `)` `img_array = keras.preprocessing.image.img_to_array(img)` `img_array = tf.expand_dims(img_array, 0)  # Create batch axis` `predictions = model.predict(img_array)` `score = predictions[0]` `print(` `"This image is %.2f percent cat and %.2f percent dog."` `% (100 * (1 - score), 100 * score)` `)`
Değerlendirme	Farklı ham veriler kullanılarak görüntüler üzerinde veri seti oluşturma, bozuk verilerin ayrılması, model oluşturma, modelin eğitilmesi ve modelin farklı veriler ile başarımının test edilmesi beklenir.

ETKİNLİK NO: 2

Ders	Bilişim Teknolojileri ve Yazılım
Program	*Özel Yetenekleri Geliştirme Programı*
Modül / Öğrenme Alanı	*Yapay Zeka/Makine Öğrenmesi*
Önerilen Süre	*4 x 40dk*
Öğrenci Kazanımları	• *Yapay zekanın kullanım alanlarından resim içerisindeki insan yüzünü tespit edebilir.* • *Yüzü oluşturan temel yapıların işaretlenmesini sağlayabilir.* • *Yüz hatlarını oluşturan 68 noktanın hangileri olduğunu, hangi numaraların, hangi noktanın nereyi ifade ettiğini açıklar.*
Öğretme-Öğrenme Yöntem ve Teknikleri	• *Analitik düşünme* • *Keşfetme* • *Disiplinlerarası düşünme* • *Uygulama Geliştirme*
Araç-Gereçler ve Kaynaklar	*Bilgisayar* *İnternet* *Web Kamera*
Öğretme-Öğrenme Süreci	HAZIRLIKLAR: • Yüz tanıma algoritmaları ve uygulama alanları sunu üzerinden gösteririlir. Bunlardan birisi olan shape_predictor_68_face_landmarks ağırlıkları ile yüz tanıma gerçekleştirilir. • Her noktanın yeri görüşülür. Farklı resimler ile testler yapılır. • Bu uygulamaları yapabilmek için, Temel seviye python bilinmesi gerekli ve bilgisayarda python, dlib ve opencv kütüphaneleri kurulu olmak zorundadır. • Örnek kod ve ilgili veri yapısı öğretmen tarafından öğrencilere verilmelidir. KAVRAMLAR: • ***Eğitilmiş bir model üzerinden yüz tanıma yapar.*** • ***Yüz tanıma işlemini 68 nokta ile yapar*** • ***İlgili Yüz Noktalarını tespit eder***

ÖN BİLGİLER:

- ***Python programlama dilini bilir.***
- ***Python ilg grafik komutlarını kullanır.***

UYGULAMA:

Dikkat Çekme:

Örnek uygulamalar ve uygulama alanları resim ve videolarla desteklenerek gösterilir. Sahip olunan algoritma ile noktaların tespiti sonrası nerelerde uygulama alanı olabileceği beyin fırtınası ile tartışılır.

Etkinlik:

Etkinlik Akışı

1. Ders :

Yüz tanıma algoritmaları gösterilir.

Yüz Tanıma Algoritmaları

Geleneksel algoritmalar iki temel yaklaşımı esas alır;

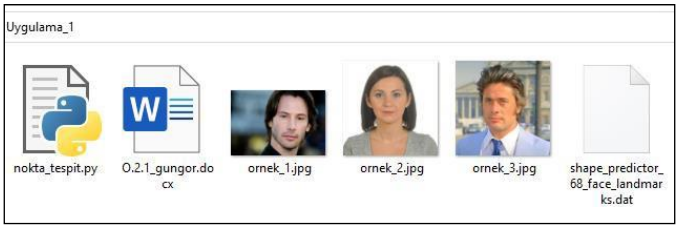
- Geometrik (özellik tabanlı) Yaklaşım
- Fotometrik (görünüm tabanlı) Yaklaşım

Fiziksel ya da geometrik yaklaşımda amaç, değişebilen özellikleri kullanmaktır. Bu da yüz özelliklerinin yapılandırılmasına dayanır. Göz, burun, ağız gibi yüzde ilk olarak konumlandırılan yerlerine, birbirlerine olan mesafelerine, açılarına göre farklı sınıflandırmalara dayanır. Bu yaklaşım, yüzün tanımlanmasının ardından ilk olarak alınan pozlama görüntüsü ile karşılaştırma yapılırken, olabilecek herhangi bir gölge, aydınlatma değişiklikleri ya da pozlamalardan kaynaklı oluşabilecek farklılıklardaki yer işaretlerinin tespitine dayanır. Bunun yanında görünüm tabanlı yaklaşımda, tüm yüzün şablonu kullanılır.

Kullanılan Yüz Tanıma Teknikleri

- *Temel Bileşen Analizi (PCA)*
- *Doğrusal Ayrımcılık Analizi (LDA)*
- *Elastik Demet Grafik Eşleştirme (Elastic Bunch Graph Matching= EBGM)*
- *3D Yüz Tanıma Teknikleri üzerinde durulur.*

2. Ders :



Uygulama Klasörü İçi

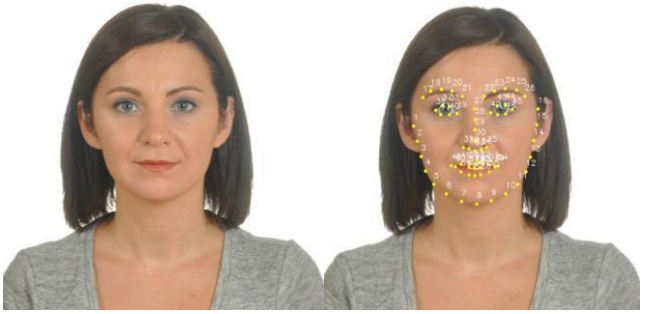
Uygulama klasörü içinde bulunan **shape_predictor_68_dace_landmarks.dat** dosyası hazırlanmış olan derin öğrenme ile hazırlanmış ağırlık dosyasıdır. Nokta_tespit.py bu dosya olmadan çalışmayacaktır. İşlenecek resimlerin ve bu dosyanın aynı klasörde olması işlemleri kolaylaştıracaktır.

Hazırlanmış kodlar üzerinden 68 noktalı, yüz tanıma uygulaması olan nokta_tespit.py herhangi bir editörde açılır. İçerisinde dosya adı kısmına işlenecek resim dosyasının adı yazılır ve kod çalıştırılır. Her örnek resim için aynı işlem yapılır ve aşağıdaki şekilde çıktılar üretilir.

```
import cv2
import dlib
import time

# Kullanılacak Resim Dosyasının Adı

dosya_adi ="ornek_3"

# Adı verilen dosyaların okunması ve i
frame = cv2.imread(dosya_adi+'.jpg')

# Noktaları tanımlayacak değişkenin ta
```

Örnek Resim_1, Resim_2 ve Resim_3 kullanılarak nokta_tespit.py dosyası çalıştırılır.

Örnek 1



Ornek 2

Örnek 3

3. *Ders :*

Kodların çalışma şekli öğrencilere ifade edilir. Nokta_tespit.py dosyasındaki açıklamalar doğrultusunda yüz noktalarının nasıl bulunduğu ve koordinatlarının nasıl kullanıldığı açıklanır. Örnek verilen resim dosyaları haricinde öğrencilerin kendi buldukları resimlerin işlemesi yapması beklenir.

Nokta_tespit.py

```
import  cv2
import  dlib
import time

# Kullanılacak Resim Dosyasının Adı
dosya_adi ="ornek_3"

# Adı verilen dosyaların okunması ve ilgili değişkene atılması
frame = cv2.imread(dosya_adi+'.jpg')

# Noktaları tanımlayacak değişkenin tanımlanması.
detector = dlib.get_frontal_face_detector()

# Hazırlanmış Yüz Tanıma Ağırlıklarının olduğu model dosyanın tanıtılması.
predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")

# Resimin Gri Tonlamaya Çevrilmesi
```

Co-funded by the Erasmus+ Programme of the European Union

```
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

# Histaogram eğrisi ile gereksiz parlak pikseller çıklartılır.
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8)) # Ortalama Değerleri
clahe_image = clahe.apply(gray)

# nokta koordinatları resim ile aynı isimli bir dosyaya kaydedilmesi için açılır.
coordinates_txt = open(dosya_adi+".txt","w")


# Resim Üzerinde Yüz Tanıma Yapılır
detections = detector(clahe_image, 1)

# For döngüsü tespi edilen her yüz için çalışır. birden fazla yüz tespit edilebilir.
for k,d in enumerate(detections):

    # Noktaların koordinatları shape dizinine aktarılır.
    shape = predictor(clahe_image, d)
    for i in range(0,68):

        if(i>=36 and i<=46):

            #68 nokta openCV komutları ile resim üzerine işaretlenir.
            cv2.circle(frame, (shape.part(i).x, shape.part(i).y), 1, (0,255,255),
thickness=2) #For each point, draw a red circle with thickness2 on the frame

            #koordinatlar dosyaya yazılır.
            coordinates_txt.write(str(shape.part(i).x) + " " + str(shape.part(i).y) +
"\n")

            # her koordinatın üzerine numarası yazılıyor
            cv2.putText(frame, str(i), (shape.part(i).x, shape.part(i).y-10),
cv2.FONT_HERSHEY_SIMPLEX,.4,(255, 255, 255), 1)

# Koordinatlar İlgili Dosyaya Yazılıyor ve Final Resim Görüntüleniyor
print(coordinates_txt)
cv2.imshow("image", frame)
coordinates_txt.close()
```

Bu sırada

- *Farklı açılara ve farklı yapıdaki resimlerin nasıl işlendiği*
- *Nasıl hataların oluştuğu,*
- *Oluşan hataların giderilmesi süreçleri işletilir.*

4. Ders :

Numaralandırılmış noktaların hangilerinin nereleri ifade ettiği çıkartılır.

- 0-16 Yüz Çehresi
- 22-26 Sol Kaş
- 17-21 Sağ Kaş
- 27-35 Burun
- 36-41 Sağ Göz
- 42-46 Sol Göz

İlgili noktaları ifade eden kod üzerinde sadece bu noktaların görünmesini sağlayan ifade eklenerek yüz tanıma işlemi özelleştirilebilir.

Örneğin sadece gözeleri ifade eden noktaların görüntülenmesi için; sadece i değişkeninin ifade ettiği noktaların sahip olduğu koordinat değerleri işlenir. Bu işlem yazılan if komutu ile sadece ilgili aralıktaki noktaların şartları yazılarak yapılabilir.

```
# Noktaların koordinatları shape dizinine aktarılır.
shape = predictor(clahe_image, d)
for i in range(0,68):

    if(i>=36 and i<=46):

        #68 nokta openCV komutları ile resim üzerine işaretl
        cv2.circle(frame, (shape.part(i).x, shape.part(i).y)

        #koordinatlar dosyaya yazılır.
        coordinates_txt.write(str(shape.part(i).x) + " " + s

        # her koordinatın üzerine
        cv2.putText(frame, str(i), (shape.part(i).x, shape.p

        #drawing the landmarks with thickness 3 - yellow in
```

Koda Eklenen If Satırı

Sadece Gözlerin Seçimi

Yönerge:

Etkinliğe uygun öğrencilerin bireysel veya grup çalışması ile yapabilecekleri bir görevin sunulması ve aşağıdaki gibi örnek bir yönergenin öğrencilere sunulması

- *Yüz tanıma algoritmalarını inceleyin.*
- *Uygulama alanlarını açıklayın.*
- *Verilen örneği çalıştırın.*
- *Kendi resminizi veya farklı resimler için programı çalıştırın.*
- *İstenilen noktaların seçimini sağlayın*

Co-funded by the Erasmus+ Programme of the European Union

	KAYNAKÇA • *Torun, B.,Yurdakul, M., Duygulu, P. (2007), Benzer yüzlerin bulunması, ayar Mühendisliği, Bilkent Üniversitesi* • *https://www.veribilimiokulu.com/cv2-ile-yuz-tanima-ve-belirleme/* • *https://github.com/codeniko/shape_predictor_81_face_landmarks* • *https://www.elektrikport.com/makale-detay/yuz-tanima-algoritmalari-ve-uygulamalari/22059#ad-image-0* ***ÖNEMLİ NOT:*** *Uygulamaların çalışması için aşağıda adresi yazılı ağırlık dosyasının uygulama klasörlerine ayrı ayrı kopyalanması gereklidir. Aksi halde hiç bir uygulama çalışmayacaktır.* *//////////////* *https://github.com/italojs/facial-landmarks-recognition/blob/master/shape_predictor_68_face_landmarks.dat* *//////////////*
Değerlendirme	*Öğrencilerin;* • *Programı çalıştırması* • *Programın çıktılarının incelemesi* • *Farklı bir resim üzerinde denemesi* • *Noktaların piksek koordinatları olduğu ve bu noktalara nasıl erişebilemesi* • *Noktalar arası bir bağlantı bulması beklenmektedir.*

ETKİNLİK NO: 3

Ders	Bilişim Teknolojileri ve Yazılım
Program	*Özel Yetenekleri Geliştirme Programı*
Modül / Öğrenme Alanı	*Yapay Zeka/Makine Öğrenmesi*
Önerilen Süre	*2 x 40 dk*
Öğrenci Kazanımları	• *Yapay zekanın kullanım alanlarından resim içerisindeki insan yüzünü tespit edebilir.* • *Yüzü oluşturan temel yapıların işaretlenmesini sağlayabilir.* • *Yüz hatlarını oluşturan 68 noktanın hangileri olduğunu, hangi numaraların, hangi noktanın nereyi ifade ettiğini açıklar.* • *Canlı video üzerinde yüz tanıma yapar.* • *Belirlenen noktaları kullanarak yüz çizer.*
Öğretme-Öğrenme Yöntem ve Teknikleri	• *Analitik düşünme* • *Keşfetme* • *Disiplinlerarası düşünme* • *Uygulama Geliştirme*
Araç-Gereçler ve Kaynaklar	*Bilgisayar* *İnternet* *Web Kamera*
Öğretme-Öğrenme Süreci	HAZIRLIKLAR: • Yüz tanıma algoritmaları ve uygulama alanları sunu üzerinden gösterilir. Bunlardan birisi olan shape_predictor_68_face_landmarks ağırlıkları ile yüz tanıma gerçekleştirilir. • Her noktanın yeri tespit edilir. • Canlı video üzerinde yüz tanımlaması uygulamaları gösterilir • Bu uygulamaları yapabilmek için, Temel seviye python bilinmesi gerekli ve bilgisayarda python, dlib ve opencv kütüphaneleri kurulu olmak zorundadır. • Örnek kod ve ilgili veri yapısı öğretmen tarafından öğrencilere verilmelidir. • Her öğrencide bir kamera olmalıdır.

Co-funded by the Erasmus+ Programme of the European Union

KAVRAMLAR:

- ***Eğitilmiş bir model üzerinden yüz tanıma yapar.***
- ***Yüz tanıma işlemini 68 nokta ile yapar***
- ***Canlı video ile yüz tanıma uygulaması yapar***
- ***Yüzün istenen bölümlerini gösterebilir.***

ÖN BİLGİLER:

- *Python programlama dilini bilir.*
- *Python ile grafik komutlarını kullanır.*

UYGULAMA:

Dikkat Çekme:

- *Örnek uygulamalar ve uygulama alanları resim ve videolarla desteklenerek gösterilir.*
- *Yüz yüze canlı bir örnek yaptırılır.*
- *Sonrasında yüz tanıma ile ilgili kullanım alanları tartışılır.*

Etkinlik:

Etkinlik Akışı

1. Ders :

Bir önceki ders uygulamasında, kullanılacak nokta_tespiy.py komutları güncellenerek kamera ile çalışır hale getirilmiştir. Genel yapısı itibarı ile resim üzerinde yüz hatlarını belirten algoritma video görüntüsü üzerine uygulanmaktadır.

```
import  cv2
import  dlib
import time

predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")
detector = dlib.get_frontal_face_detector()

frame = cv2.VideoCapture(0)

while(True):

    _, orjinal = frame.read()

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

    gray = cv2.cvtColor(orjinal, cv2.COLOR_BGR2GRAY)
```

```
    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))

    clahe_image = clahe.apply(gray)

    detections = detector(clahe_image, 1)

    for k,d in enumerate(detections):


        shape = predictor(clahe_image, d)

        for i in range(0,68):

            cv2.circle(orjinal, (shape.part(i).x, shape.part(i).y), 1, (0,255,255), thickness=2) #For each point, draw a red circle with thickness2 on the frame

# nokta numaraları
            cv2.putText(orjinal, str(i), (shape.part(i).x, shape.part(i).y-5), cv2.FONT_HERSHEY_SIMPLEX,.4,(255, 255, 255), 1)

    cv2.imshow("frame",orjinal)


frame.release()
cv2.destroyAllWindows()
```

Yukarıdaki kodlamada kırmızı ile işaretli yerler eklemiştir. Bu komutlar video görüntüsünün anlık alınarak anlık görüntü işleme yapılması sağlanacaktır.

Örnek Video Tespit Görüntüsü

Nokta_tespit_video.py dosyası içerisinde aşağıdaki satır aktif edilerek her noktanın numara değeri üzerinde görülmektedir.

```
# nokta numaraları
            # cv2.putText(orjinal, str(i), (shape.part(i).x, shape.part(i).y-5), cv2.FONT_HERSHEY_SIMPLEX,.4,(255, 255, 255), 1)
```

Co-funded by the Erasmus+ Programme of the European Union

Nokta Numaraları ile Birlikte

Örnek çalışma alanında görüleceği üzere ve daha önceki etkinliğimizde numaraların hangi aralıklarda kullanıldığı çıkartılmıştı.

Numaraların hangi yüz bölgesini ifade ettiği;

- 0-16 Yüz Çehresi
- 22-26 Sol Kaş
- 17-21 Sağ Kaş
- 27-35 Burun
- 36-41 Sağ Göz
- 42-46 Sol Göz
- 47-67 Ağız

Yüz tanıma işlemi noktalarla yapıldığında tamamen bu noktaların birbirlerine olan uzaklık ve oranlarına bağlı olarak yapılmaktadır.

Aşağıdaki kodlar kullanılarak tüm noktalar çizgilerle birleştirilmiştir.

```
import   cv2
import   dlib
import time

predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")
detector = dlib.get_frontal_face_detector()

frame = cv2.VideoCapture(0)

while(True):

    _, orjinal = frame.read()

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
    gray = cv2.cvtColor(orjinal, cv2.COLOR_BGR2GRAY)
```

Co-funded by the Erasmus+ Programme of the European Union

```
    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))

    clahe_image = clahe.apply(gray)

    detections = detector(clahe_image, 1)

    for k,d in enumerate(detections):


        shape = predictor(clahe_image, d)
        for i in range(0,68):
            #cv2.circle(orjinal, (shape.part(i).x, shape.part(i).y), 1, (0,255,255), thickness=2)
#For each point, draw a red circle with thickness2 on the frame
            #cv2.putText(orjinal, str(i), (shape.part(i).x, shape.part(i).y-5),
cv2.FONT_HERSHEY_SIMPLEX,.4,(255, 255, 255), 1)

        endx = shape.part(i).x
        endy = shape.part(i).y

        for i in range(0,68):
            image = cv2.line(orjinal, (endx, endy),(shape.part(i).x, shape.part(i).y), (255,
255, 255), 2)
            endx = shape.part(i).x
            endy = shape.part(i).y

    cv2.imshow("frame",orjinal)

frame.release()
cv2.destroyAllWindows()
```

Kodlar çalıştırıldığında ve webcam bağlı ise aşağıdaki gibi bir görüntü oluşacaktır.

Sadece yüzün ilgili noktaları gösterilmesini istediğimizde, sadece o noktaları ifade eden bir koşul yeterli olacaktır.

Örnek Kodumuzda aşağıdaki değişiklik yapılarak sadece kaçların görüntülenmesi sağlanmıştır.

2. Ders :

Yönerge:

```
        endx = shape.part(17).x
        endy = shape.part(17).y

        for i in range(0,68):

            if (i>=17) and (i<=26):
                image = cv2.line(orjinal, (endx, endy),(shape.part(i).x, shape.part(i).y), (0, 0,
0), 10)
                endx = shape.part(i).x
                endy = shape.part(i).y
```

```
        endx = shape.part(0).x
        endy = shape.part(0).y

        for i in range(0,68):

            if (i>=0) and (i<=16):
                image = cv2.line(orjinal, (endx, endy),(shape.part(i).x, shape.part(i).y), (0,
255, 255), 10)

                endx = shape.part(i).x
                endy = shape.part(i).y
```

```
endx = shape.part(27).x
        endy = shape.part(27).y

        for i in range(0,68):

            if (i>=31) and (i<=35):
                image = cv2.line(orjinal, (endx, endy),(shape.part(i).x, shape.part(i).y), (0, 255, 255), 10)

                endx = shape.part(i).x
                endy = shape.part(i).y
```

Bu uygulamada noktların konumları yüz üzerinde istediğimiz bölgede istediğimiz işlemi yapabilmemizi sağlamaktadır.

Örneğin, burmumuzun uç nokatası 33 numaralıdır. Bu numaranın koordinatına kırmızı bir top yerleştirdiğimizde burnumuz bir palyaço burnu olarak görünecektir.

cv2.circle(orjinal, (shape.part(30).x, shape.part(30).y), 45, (0,255,255), thickness=-1)

KAYNAKÇA

- *Torun, B.,Yurdakul, M., Duygulu, P. (2007), Benzer yüzlerin bulunması, Bilgisayar Mühendisliği, Bilkent Üniversitesi*
- *https://www.veribilimiokulu.com/cv2-ile-yuz-tanima-ve-belirleme/*
- *https://github.com/codeniko/shape_predictor_81_face_landmarks*
- *https://www.elektrikport.com/makale-detay/yuz-tanima-algoritmalari-ve-uygulamalari/22059#ad-image-0*

ÖNEMLİ NOT:

Uygulamaların çalışması için aşağıda adresi yazılı ağırlık dosyasının uygulama klasörlerine ayrı ayrı kopyalanması gereklidir. Aksi halde hiç bir uygulama çalışmayacaktır.

//////////////

https://github.com/italojs/facial-landmarks-recognition/blob/master/shape_predictor_68_face_landmarks.dat

//////////////

Değerlendirme	*Öğrencilerin;* • *Programı çalıştırması* • *Programın çıktılarının incelemesi* • *Farklı bir resim üzerinde denemesi* • *Noktaların piksek koordinatları olduğu ve bu noktalara nasıl erişebilemesi* • *Noktalar arası bir bağlantı bulması beklenmektedir.*

ETKİNLİK NO: 4

Ders	Bilişim Teknolojileri ve Yazılım
Program	*Özel Yetenekleri Geliştirme Programı*
Modül / Öğrenme Alanı	*Yapay Zeka/Makine Öğrenmesi*
Önerilen Süre	*2 x 40 dk*
Öğrenci Kazanımları	• *Yapay zekanın kullanım alanlarından resim içerisindeki insan yüzünü tespit edebilir.* • *Yüzü oluşturan temel yapıların işaretlenmesini sağlayabilir.* • *Yüz hatlarını oluşturan 68 noktanın hangileri olduğunu, hangi numaraların, hangi noktanın nereyi ifade ettiğini açıklar.* • *Canlı video üzerinde yüz tanıma yapar.* • *Belirlenen noktaları kullanarak gözün açık kapalı olduğunu tespit eder.*
Öğretme-Öğrenme Yöntem ve Teknikleri	• *Analitik düşünme* • *Keşfetme* • *Disiplinlerarası düşünme* • *Uygulama Geliştirme*
Araç-Gereçler ve Kaynaklar	*Bilgisayar* *İnternet* *Web Kamera*
Öğretme-Öğrenme Süreci	HAZIRLIKLAR: • Yüz tanıma algoritmaları ve uygulama alanları sunu üzerinden gösterilir. Bunlardan birisi olan shape_predictor_68_face_landmarks ağırlıkları ile yüz tanıma gerçekleştirilir. • Her noktanın yeri tespit edilir. Farklı resimler ile testler yapılır. • Bu uygulamaları yapabilmek için, Temel seviye python bilinmesi gerekli ve bilgisayarda python, dlib ve opencv kütüphaneleri kurulu olmak zorundadır. • Örnek kod ve ilgili veri yapısı öğretmen tarafından öğrencilere verilmelidir. • Her öğrencide bir kamera olmalıdır.

Co-funded by the Erasmus+ Programme of the European Union

KAVRAMLAR:

- ***Eğitilmiş bir model üzerinden yüz tanıma yapar.***
- ***Yüz tanıma işlemini 68 nokta ile yapar***
- ***Canlı video ile yüz tanıma uygulaması yapar***
- ***Yüzün istenen bölümlerini gösterebilir.***
- ***Gözlerin açık ve kapalı olduğunu algılatabilir.***

ÖN BİLGİLER:

- *Python programlama dilini bilir.*
- *Python ile grafik komutlarını kullanır.*

UYGULAMA:

Dikkat Çekme:

- *Örnek uygulamalar ve uygulama alanları resim ve videolarla desteklenerek gösterilir.*
- *Yüz yüze canlı bir örnek yaptırılır.*
- *Sonrasında yüz tanıma ile ilgili kullanım alanları tartışılır.*
- *Göz kapaklarının konumu ve hareketlerini tespit eder.*

Etkinlik:

Etkinlik Akışı

1. Ders :

Bir önceki ders uygulamasında, kullanılacak nokta_tespiy_video.py komutları güncellenerek kamera ile çalışır hale getirilmiştir. Genel yapısı itibarı ile resim üzerinde yüz hatlarını belirten algoritma video görüntüsü üzerine uygulanmaktadır.

```
import  cv2
import  dlib
import time

predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")
detector = dlib.get_frontal_face_detector()

frame = cv2.VideoCapture(0)

while(True):

    _, orjinal = frame.read()

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

    gray = cv2.cvtColor(orjinal, cv2.COLOR_BGR2GRAY)

    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))

    clahe_image = clahe.apply(gray)

    detections = detector(clahe_image, 1)
```

```
    for k,d in enumerate(detections):


        shape = predictor(clahe_image, d)

        for i in range(0,68):

            cv2.circle(orjinal, (shape.part(i).x, shape.part(i).y), 1, (0,255,255), thickness=2)
#For each point, draw a red circle with thickness2 on the frame

# nokta numaraları
            cv2.putText(orjinal,          str(i),          (shape.part(i).x,          shape.part(i).y-5),
cv2.FONT_HERSHEY_SIMPLEX,.4,(255, 255, 255), 1)

    cv2.imshow("frame",orjinal)


frame.release()
cv2.destroyAllWindows()
```

Yukarıdaki kodlamada kırmızı ile işaretli yerler eklemiştir. Bu komutlar video görüntüsünün anlık alınarak anlık görüntü işleme yapılması sağlanacaktır.

Örnek Video Tespit Görüntüsü

Nokta_tespit_video.py dosyası içerisinde aşağıdaki satır aktif edilerek her noktanın numara değeri üzerinde görülmektedir.

Örnek çalışma alanında görüleceği üzere ve daha önceki etkinliğimizde numaraların hangi aralıklarda kullanıldığı çıkartılmıştı.

Numaraların hangi yüz bölgesini ifade ettiği;

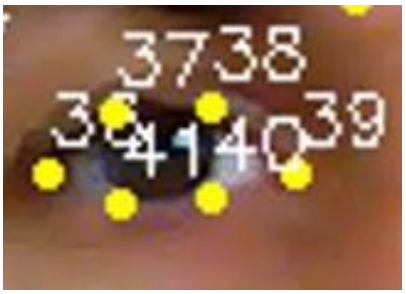
- 0-16 Yüz Çehresi
- 22-26 Sol Kaş
- 17-21 Sağ Kaş
- 27-35 Burun
- 36-41 Sağ Göz
- 42-46 Sol Göz
- 47-67 Ağız

Co-funded by the Erasmus+ Programme of the European Union

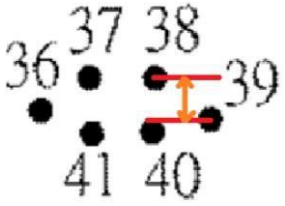
2. Ders :

Sadece yüzün ilgili noktaları gösterilmesini istediğimizde, sadece o noktaları ifade eden bir koşul yeterli olacaktır.

Örnek Kodumuzda aşağıdaki değişiklik yapılarak sadece gözlerin görüntülenmesi sağlanmıştır.

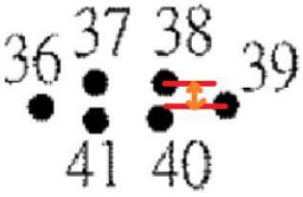


Sol Gözü İfade Eden Noktaların Konumları



Göz Açık Olma Durumu

İki piksel y değeri arasındaki fark 4 pikselden fazla olursa göz açık.



Göz Kapalı Olma Durumu

İki piksel y değeri arasındaki fark 4 pikselden az olursa göz kapalı.

Burada yapılan 38 ve 39 nolu noktalar arası piksel farkının gözün açık ve kapalı olma durumunu ifade etmesidir.

İlgili Kodumuz :

```
import  cv2
import  dlib
import time

predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")

detector = dlib.get_frontal_face_detector()

frame = cv2.VideoCapture(0)

while(True):

    _, orjinal = frame.read()
```

```
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

    gray = cv2.cvtColor(orjinal, cv2.COLOR_BGR2GRAY)

    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))

    clahe_image = clahe.apply(gray)

    detections = detector(clahe_image, 1)

    for k,d in enumerate(detections):


        shape = predictor(clahe_image, d)

        #for i in range(0,68):
            #print("")
            #cv2.circle(orjinal, (shape.part(i).x, shape.part(i).y), 1, (0,255,255), thickness=2)
#For each point, draw a red circle with thickness2 on the frame
            #cv2.putText(orjinal, str(i), (shape.part(i).x, shape.part(i).y-5),
cv2.FONT_HERSHEY_SIMPLEX,.4,(255, 255, 255), 1)

        solendx = shape.part(36).x
        solendy = shape.part(36).y


        for i in range(0,68):

            if (i>=36) and (i<=41):
                image = cv2.line(orjinal, (solendx, solendy),(shape.part(i).x, shape.part(i).y),
(0, 255, 255), 1)

                solendx = shape.part(i).x
                solendy = shape.part(i).y


        image = cv2.line(orjinal, (solendx, solendy),(shape.part(36).x, shape.part(36).y),
(0, 255, 255), 1)


        if (shape.part(39).y-shape.part(38).y)<4 :

            cv2.putText(orjinal, " GOZ KAPALI ", (5, 30),
cv2.FONT_HERSHEY_SIMPLEX,1,(255, 0, 0), 3)
        else:
            cv2.putText(orjinal, " GOZ ACIK ", (5, 30),
cv2.FONT_HERSHEY_SIMPLEX,1,(0, 0, 255), 3)

    cv2.imshow("frame",orjinal)


frame.release()
cv2.destroyAllWindows()
```

Göz Tespiti Yapan Uygulamanın Çalışan Hali

KAYNAKÇA

- *Torun, B.,Yurdakul, M., Duygulu, P. (2007), Benzer yüzlerin bulunması, Bilgisayar Mühendisliği, Bilkent Üniversitesi*
- *https://www.veribilimiokulu.com/cv2-ile-yuz-tanima-ve-belirleme/*
- *https://github.com/codeniko/shape_predictor_81_face_landmarks*
- *https://www.elektrikport.com/makale-detay/yuz-tanima-algoritmalari-ve-uygulamalari/22059#ad-image-0*

ÖNEMLİ NOT:

Uygulamaların çalışması için aşağıda adresi yazılı ağırlık dosyasının uygulama klasörlerine ayrı ayrı kopyalanması gereklidir. Aksi halde hiç bir uygulama çalışmayacaktır.

//////////////

https://github.com/italojs/facial-landmarks-recognition/blob/master/shape_predictor_68_face_landmarks.dat

//////////////

Değerlendirme	*Öğrencilerin;* • *Programı çalıştırması* • *Programın çıktılarının incelemesi* • *Farklı bir resim üzerinde denemesi* • *Noktaların piksek koordinatları olduğu ve bu noktalara nasıl erişebilemesi* • *Noktalar arası bir bağlantı bulması beklenmektedir.*

ETKİNLİK NO: 5

Ders	Bilişim Teknolojileri ve Yazılım
Program	ÖYG, PROJE
Modül / Öğrenme Alanı	5. Görüntü İşleme 5.3.Nesne yönelimli yüksek seviyeli dilde kütüphane işlemleri
Önerilen Süre	2 Ders Saati
Öğrenci Kazanımları	• Görüntü işlemede kullanılacak OpenCV kütüphanesini bilir. • Görüntünün işlenmesi için gereken algoritma ve kütüphaneyi kullanır.
Öğretme-Öğrenme Yöntem ve Teknikleri	Soru Cevap, Düz Anlatım, Gösterip Yaptırma, Uygulama
Araç-Gereçler ve Kaynaklar	• Python geliştirme ortamının kurulu olduğu internet bağlantılı bilgisayar (öğrenci sayısı kadar) • Anaconda Yazılımı • Etkileşimli Tahta veya Projeksiyon • https://ab.org.tr/ab10/kitap/eristi_AB10.pdf (Erişim Tarihi:16.11.2021)
Öğretme-Öğrenme Süreci	Öncelikle bilgisayarlara Anaconda yazılımını yükleyelim https://www.anaconda.com/products/individual Anaconda Prompt programını açalım Gerekli kütüphaneleri yükleyelim, OpenCV kütüphanesini yüklemek için aşağıdaki satırı Prompt ekranına yazalım. conda install -c conda-forge opencv Konunun anlaşılması için OpenCV kütüphanesi ve bileşenleri anlatılır. OpenCV(Open Source Computer Vision) , “bir resim ya da video içindeki anlamlı bilgileri çıkarıp işleyebilmek için INTEL tarafından C ve C++

dilleri kullanılarak geliştirilmiş, açık kaynak kodlu görüntü işleme kütüphanesi" şeklinde açıklanabilir.

OpenCV kütüphanesi, beş temel bileşenden oluştuğu belirtilerek, bileşenler sırayla açıklanır.

CV(Computer Vision (Bilgisayarla Görü/Görme) bileşeni, temel resim işleme fonksiyonları ve Bilgisayarla Görü/Görme için kullanılan yüksek seviyeli algoritmaları bünyesinde barındıran kütüphaneden biridir.
MLL(Machine Learning Library) bileşeni, Makina Öğrenmesi dalı için gerekli istatistiksel verilere ulaşmak, mevcut verileri sınıflandırmak için kullanılan fonksiyonları/araçları içeren kütüphanedir.
HighGUI bileşeni, slider, form gibi OpenCV kütüphanesi içerisinde tanımlanmış pek çok nesneyi yaratabilmemizi sağlayan bir grafik arabirimi olmakla beraber, resim ve videoları kaydetmek, yüklemek, hafızadan silmek için gerekli giriş/çıkış (I/O) fonksiyonları içerir.
CXCore bileşeni, OpenCV'ye ait IpImage, cvPoint, cvSize, cvMat, cvHistogram... vs gibi veri yapılarını bünyesinde barındıran, XML desteği de sağlayan bir kütüphanedir.
CvAux bileşeni, şablon eşleştirme (template-matching), şekil eşleştirme (shape matching), bir objenin ana hatlarını bulma (finding skeletons), yüz tanıma (face-recognition), ağız hareketleri izleme (mouth-tracking), vücut hareketlerini tanıma (gesture recognition) ve kamera kalibrasyonu gibi daha pek çok deneysel algoritmaları bünyesinde barındıran kütüphanedir

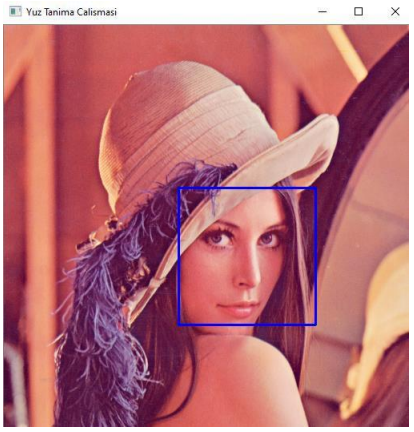
OpenCV bileşenleri açıklandıktan sonra, öğrencilere OpenCV kütüphanesi kullanılarak yüz tanıma etkinliği yapacağımız söylenir.
Haar Cascade kullanan yüz algılama, kademeli bir işlevin bir dizi girdi verisi ile eğitildiği, makine öğrenimi tabanlı bir yaklaşım olduğu belirtilir. Yüz algılama için daha önce eğitilmiş bir Haar Cascade veri seti indirilir.

Veri setini indirmek için aşağıdaki linki kullanabilirsiniz.
https://raw.githubusercontent.com/opencv/opencv/master/data/haarcascades/haarcascade_frontalface_default.xml

Yukarıdaki linkte açılan sayfanın üzerine sağ tıklayarak Farklı Kaydet seçeneği ile dosyayı, uygulama klasörünüzün içerisine "haarcascade_frontalface_default.xml" ismi ile kaydedelim.

Haar Cascade dosyası indirildikten sonra, Anaconda Spyder uygulaması açılır ve öğrencilerin de kendi bilgisayarlarında aynı uygulamayı açmaları sağlanır. Aşağıdaki kodlar açıklanır ve öğrencilerin de kodları adım adım yazmaları ve programı çalıştırmaları sağlanır.

```
import cv2
# Cascade XML dosyasını yükleme
face_cascade =
cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
# Giriş görüntüsünü okuma
#Görüntü dosyasını uygulama klasörü içerisinde bulunmasına dikkat
ediniz.
img = cv2.imread('lenna.png')
# Gri renk seviyesine dönüştürme
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
# Yüz algılama
faces = face_cascade.detectMultiScale(gray, 1.05, 4)
'''
MultiScale işlevi yüzleri algılamak için kullanılır ve 3 argüman alır:
Giriş görüntüsü, ScaleFactor ve MinNeighbours.
ScaleFactor, her ölçekle görüntü boyutunun ne kadar küçültüleceğini
belirtir.
MinNeighbours, her aday dikdörtgenin onu korumak için kaç komşusu
olması gerektiğini belirtir.
'''
# Yüz çevresinin dikdörtgen ile gösterilmesi
for (x, y, w, h) in faces:
    cv2.rectangle(img, (x, y), (x+w, y+h), (255, 0, 0), 2)
# Çıkış görüntüsünü oluşturma
```

	cv2.imshow('Yuz Tanima Calismasi', img) cv2.waitKey() Program çıktısı aşağıdaki şekilde olacaktır:  *Görüntünün Plot sekmesinde görüntülenmesi* Spyder programında plot sekmesinde görüntü sonuçlarını görmek için öncelikle Matplotlib kütüphanesini yüklememiz gerekir. Matplotlib; veri görselleştirmesinde kullandığımız temel python kütüphanesidir. Anaconda Prompt ekranından aşağıdaki kodu yazarak Matplotlib kütüphanesini yüklenmesini sağlayabilirsiniz. conda install -c conda-forge matplotlib Son olarak aşağıdaki kod satırını, daha önce yazdığımız kodun sonuna ekleyerek görüntümüzü plot sekmesinde görüntüleyebiliriz. plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
Değerlendirme	Farklı yüz görüntüleri üzerinde; Scalefactor ve minNeighbours değerlerini değiştirerek ideal sonuçlara ulaşmaları beklenir.

ETKİNLİK NO: 6

Ders	Bilişim Teknolojileri ve Yazılım
Program	ÖYG, PROJE
Modül / Öğrenme Alanı	5. Görüntü İşleme 5.3.Nesne yönelimli yüksek seviyeli dilde kütüphane işlemleri
Önerilen Süre	2 Ders Saati
Öğrenci Kazanımları	• Görüntü işlemede kullanılacak OpenCV kütüphanesini bilir. • Görüntünün işlenmesi için gereken algoritma ve kütüphaneyi kullanır.
Öğretme-Öğrenme Yöntem ve Teknikleri	Soru Cevap, Düz Anlatım, Gösterip Yaptırma, Uygulama
Araç-Gereçler ve Kaynaklar	• Python geliştirme ortamının kurulu olduğu internet bağlantılı bilgisayar (öğrenci sayısı kadar) • Anaconda Yazılımı • Etkileşimli Tahta veya Projeksiyon
Öğretme-Öğrenme Süreci	Önceki etkinlikte Anaconda yazılımı yükleme ve OpenCV kütüphanelerinin yüklenmesinden bahsedilmiştir. Öğrencilere OpenCV kütüphanesi kullanılarak kameradaki görüntüde yüz yakalama etkinliği yapacağımız söylenir. Yüz algılama için daha önce eğitilmiş bir Haar Cascade veri seti indirilir. Veri setini indirmek için aşağıdaki linki kullanabilirsiniz. https://raw.githubusercontent.com/opencv/opencv/master/data/haarcascades/haarcascade_frontalface_default.xml Yukarıdaki linkte açılan sayfanın üzerine sağ tıklayarak Farklı Kaydet seçeneği ile dosyayı, uygulama klasörünüzün içerisine "haarcascade_frontalface_default.xml" ismi ile kaydedelim. Haar Cascade dosyası indirildikten sonra, Anaconda Spyder uygulaması açılır ve öğrencilerin de kendi bilgisayarlarında aynı uygulamayı açmaları sağlanır.

Aşağıdaki kodlar açıklanır ve öğrencilerin de kodları adım adım yazmaları ve programı çalıştırmaları sağlanır.

```
import time
import cv2

# Cascade XML yükle
face_cascade =cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

# Webcam üzerinden videonun açılması
cap = cv2.VideoCapture(0)

'''
Videoyu giriş verisi olarak kullanma
Web Kamerasından alınan görüntü yerine aşağıdaki kod ile herhangi bir
video dosyasından da veriler alınabilir.
'''
# cap = cv2.VideoCapture('filename.mp4')

sayac=0
while True:
    # Her karenin okunması
    _, img = cap.read()

    # Gri seviyeye çevrilmesi
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # Yüzlerin tanınması
    faces = face_cascade.detectMultiScale(gray, 1.1, 4)

    # Her yüzün çevresinde dikdörtgen oluşturulması
    for (x, y, w, h) in faces:
        sayac+=1
        cv2.rectangle(img, (x, y), (x+w, y+h), (255, 0, 0), 2)
```

```
        #Yakalanan yüz görüntüsünün kaydedilmesi
        cv2.imwrite("yuz" + str(sayac) +".jpg", img[y:y+h,x:x+w])
        #cv2.imwrite("yuz" + str(sayac) +".jpg", gray[y:y+h,x:x+w])
        #gray yerine img yazıldığında görüntü renkli kaydedilir

    # Çıkış görüntüsü
    cv2.imshow('img', img)
    time.sleep(1) #1 saniye aralıklarla görüntü alacaktır

    # ESC tuşu veya sayaç 4'ten büyük olduğunda çıkış
    k = cv2.waitKey(30) & 0xff
    if k==ord('c') or sayac>4:
        break

cap.release() # VideoCapture objesinin serbest bırakılması

cv2.destroyAllWindows() # tüm ekranları kapatılması
```

Program çalıştırıldığında uygulama klasörünün içine; 5 adet web kamerasından yakalanan yüz görüntüsü kaydedilecektir.

Değerlendirme	Öğrencilerden farklı Haar Cascade veri setlerini ulaşmaları istenir. Öğrencilerin sadece gülen yüzleri yakalayan bir çalışma yapmaları istenir.

ETKİNLİK NO: 7

Ders	Bilişim Teknolojileri ve Yazılım
Program	ÖYG, PROJE
Modül / Öğrenme Alanı	5. Görüntü İşleme 5.3.Nesne yönelimli yüksek seviyeli dilde kütüphane işlemleri
Önerilen Süre	2 Ders Saati
Öğrenci Kazanımları	• Görüntü işlemede kullanılacak OpenCV kütüphanesini bilir. • Görüntünün işlenmesi için gereken algoritma ve kütüphaneyi kullanır.
Öğretme-Öğrenme Yöntem ve Teknikleri	Soru Cevap, Düz Anlatım, Gösterip Yaptırma, Uygulama
Araç-Gereçler ve Kaynaklar	• İnternet bağlantılı bilgisayar (öğrenci sayısı kadar) • Anaconda Yazılımı • Etkileşimli Tahta veya Projeksiyon
Öğretme-Öğrenme Süreci	Bu etkinlikte Web kamerasından gülen yüzler yakalanacak ve elde edilen yüzler uygulama klasörüne kaydedilecektir. Bu etkinlik öncesinde OpenCV kütüphanesi kullanılarak kameradaki görüntüde yüz yakalama etkinliğinin yapılması önerilmektedir. Yüz tanıma için haarcascade_frontalface_default.xml dosyası daha önce indirilmediyse indirilmelidir. https://raw.githubusercontent.com/opencv/opencv/master/data/haarcascades/haarcascade_frontalface_default.xml Ayrıca gülen yüzleri tanıması için daha önce eğitilmiş smile cascade dosyası da indirilmelidir. https://raw.githubusercontent.com/opencv/opencv/master/data/haarcascades/haarcascade_smile.xml Yukarıdaki linkte açılan sayfaların üzerine sağ tıklayarak Farklı Kaydet seçeneği ile dosyaları, uygulama klasörünüzün içerisine sırasıyla “haarcascade_frontalface_default.xml” “haarcascade_smile.xml” isimleriyle kaydedelim

Haar Cascade dosyaları indirildikten sonra, Anaconda Spyder uygulaması açılır ve öğrencilerin de kendi bilgisayarlarında aynı uygulamayı açmaları sağlanır. Aşağıdaki kodlar açıklanır ve öğrencilerin de kodları adım adım yazmaları ve programı çalıştırmaları sağlanır.

```
import cv2
# Cascade XML yükleme
face_cascade =cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
smileCascade = cv2.CascadeClassifier('haarcascade_smile.xml')

# Webcam üzerinden videonun açılması
cap = cv2.VideoCapture(0)

sayac = 0
while True:
# Her karenin okunması
_, img = cap.read()
image = img
# Gri seviyeye çevrilmesi
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
# Yüzlerin tanınması
faces = face_cascade.detectMultiScale(gray, 1.1, 4)
# Her yüzün çevresinde dikdörtgen oluşturulması
for (x, y, w, h) in faces:
#tespit edilen yüzler mavi çerçeveye alınacak
cv2.rectangle(img, (x, y), (x+w, y+h), (255, 0, 0), 2)
smile = smileCascade.detectMultiScale(
gray[y:y+h,x:x+w],
scaleFactor= 1.5,
minNeighbors=15,
minSize=(25, 25),
)
```

	for (xx, yy, ww, hh) in smile: *#tespit edilen yüzdeki gülümsemeyi yakalama* *cv2.rectangle(img[y:y+h,x:x+w], (xx, yy), (xx + ww, yy + hh), (0, 255, 0), 2)* *#gülümseme çerçevesi oluşturma, çerçeve yeşil olacak* *sayac =sayac+1* *cv2.imwrite("gulenyuz" + str(sayac) + ".jpg", image[y:y+h,x:x+w])* *# Çıkış görüntüsü* *cv2.imshow('img', img)* *# ESC ve sayaç 10 olduğunda çıkış* *k = cv2.waitKey(1) & 0xff* *if k== ord('c') or sayac>9:* *break* *cap.release() # VideoCapture objesinin serbest bırakılması* *cv2.destroyAllWindows() # tüm ekranları kapat* Program çalıştırıldığında uygulama klasörünün içine; 10 adet web kamerasından yakalanan gülen yüz görüntüsü kaydedilecektir.
Değerlendirme	Öğrencilerin sadece gülen yüzlerdeki gözleri yakalayan bir çalışma yapmaları istenir.

ETKİNLİK NO: 8

Ders	Bilişim Teknolojileri ve Yazılım
Program	ÖYG, PROJE
Modül / Öğrenme Alanı	5. Görüntü İşleme 5.3.Nesne yönelimli yüksek seviyeli dilde kütüphane işlemleri
Önerilen Süre	2 Ders Saati
Öğrenci Kazanımları	• Görüntü işlemede kullanılacak OpenCV kütüphanesini bilir. • Görüntünün işlenmesi için gereken algoritma ve kütüphaneyi kullanır.
Öğretme-Öğrenme Yöntem ve Teknikleri	Soru Cevap, Düz Anlatım, Gösterip Yaptırma, Uygulama
Araç-Gereçler ve Kaynaklar	• İnternet bağlantılı bilgisayar (öğrenci sayısı kadar) • Anaconda Yazılımı • Etkileşimli Tahta veya Projeksiyon
Öğretme-Öğrenme Süreci	Bu etkinlikte Web kamerasından gülen yüzler yakalanacak ve elde edilen yüzler uygulama klasörüne kaydedilecektir. Bu etkinlik öncesinde OpenCV kütüphanesi kullanılarak kameradaki görüntüde yüz yakalama etkinliğinin yapılması önerilmektedir. Yüz tanıma için haarcascade_frontalface_default.xml dosyası daha önce indirilmediyse indirilmelidir. https://raw.githubusercontent.com/opencv/opencv/master/data/haarcascades/haarcascade_frontalface_default.xml Ayrıca gülen yüzleri tanıması için daha önce eğitilmiş smile cascade dosyası da indirilmelidir. https://raw.githubusercontent.com/opencv/opencv/master/data/haarcascades/haarcascade_smile.xml Yukarıdaki linkte açılan sayfaların üzerine sağ tıklayarak Farklı Kaydet seçeneği ile dosyaları, uygulama klasörünüzün içerisine sırasıyla “haarcascade_frontalface_default.xml” “haarcascade_smile.xml” isimleriyle kaydedelim

Co-funded by the Erasmus+ Programme of the European Union

Haar Cascade dosyaları indirildikten sonra, Anaconda Spyder uygulaması açılır ve öğrencilerin de kendi bilgisayarlarında aynı uygulamayı açmaları sağlanır. Aşağıdaki kodlar açıklanır ve öğrencilerin de kodları adım adım yazmaları ve programı çalıştırmaları sağlanır.

```
import cv2
# Cascade XML yükleme
face_cascade =cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
smileCascade = cv2.CascadeClassifier('haarcascade_smile.xml')

# Webcam üzerinden videonun açılması
cap = cv2.VideoCapture(0)

sayac = 0
while True:
    # Her karenin okunması
    _, img = cap.read()
    image = img
    # Gri seviyeye çevrilmesi
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    # Yüzlerin tanınması
    faces = face_cascade.detectMultiScale(gray, 1.1, 4)
    # Her yüzün çevresinde dikdörtgen oluşturulması
    for (x, y, w, h) in faces:
        #tespit edilen yüzler mavi çerçeveye alınacak
        cv2.rectangle(img, (x, y), (x+w, y+h), (255, 0, 0), 2)
        smile = smileCascade.detectMultiScale(
            gray[y:y+h,x:x+w],
            scaleFactor= 1.5,
            minNeighbors=15,
            minSize=(25, 25),
            )
```

	`` for (xx, yy, ww, hh) in smile: #tespit edilen yüzdeki gülümsemeyi yakalama cv2.rectangle(img[y:y+h,x:x+w], (xx, yy), (xx + ww, yy + hh), (0, 255, 0), 2) #gülümseme çerçevesi oluşturma, çerçeve yeşil olacak sayac =sayac+1 cv2.imwrite("gulenyuz" + str(sayac) + ".jpg", image[y:y+h,x:x+w]) # Çıkış görüntüsü cv2.imshow('img', img) # ESC ve sayaç 10 olduğunda çıkış k = cv2.waitKey(1) & 0xff if k== ord('c') or sayac>9: break cap.release() # VideoCapture objesinin serbest bırakılması cv2.destroyAllWindows() # tüm ekranları kapat ``` Program çalıştırıldığında uygulama klasörünün içine; 10 adet web kamerasından yakalanan gülen yüz görüntüsü kaydedilecektir.
Değerlendirme	Öğrencilerin sadece gülen yüzlerdeki gözleri yakalayan bir çalışma yapmaları istenir.

ETKİNLİK NO: 9

Ders	Bilişim Teknolojileri ve Yazılım
Program	ÖYG
Modül / Öğrenme Alanı	6. Yapay Zekâ 6.9. Derin Öğrenme
Önerilen Süre	6 (Altı) Ders Saati
Öğrenci Kazanımları	Derin öğrenme algoritmalarını kullanarak bir bilgisayarlı görü probleminin çözümüne ilişkin uygulama geliştirir.
Öğretme-Öğrenme Yöntem ve Teknikleri	Soru Cevap, Düz Anlatım, Beyin fırtınası, Gösterip Yaptırma, Uygulama
Araç-Gereçler ve Kaynaklar	• Python geliştirme ortamının kurulu olduğu internet bağlantılı bilgisayar (öğrenci sayısı kadar) • Python geliştirme ortamının kurulu olduğu internet bağlantılı etkileşimli tahta • Türkçe Kitap: Prof. Dr. Vasif V. NABİYEV – Yapay Zekâ • İngilizce Kitap: Stuart Russel & Peter Norvig – Artificial Intelligence: A Modern Approach • https://www.ibm.com/cloud/learn/deep-learning (Erişim tarihi: 09.11.2121) • https://keras.io/ (Erişim tarihi: 09.11.2021) • https://en.wikipedia.org/wiki/Digital_image_processing (Erişim tarihi: 09.11.2021)
Öğretme-Öğrenme Süreci	Önceki etkinliklerde öğrenilen veri, veri seti, derin öğrenme gibi kavramların hatırlatılmasıyla etkinliğe başlanır. **Derin öğrenme**; bir makine öğrenmesi yöntemidir. ‘Derin’ öğrenme olarak adlandırılmasının nedeni öğrenme algoritmalarının birden fazla katmandan oluşmasıdır. Girişin ardından, yapılacak uygulamanın amacından bahsedilir: Makinemizin, el yazısı ile yazılmış rakamları tanıması sağlanacaktır. Bu amaç doğrultusunda, ilk olarak farklı insanlar tarafından el yazısı ile yazılmış rakamların resimlerinden oluşan **MNIST** veri seti detaylı

olarak incelenir (http://yann.lecun.com/exdb/mnist/). Resim dosyalarının, aslında sayılardan oluşan birer matris olduğu açıklanır. Örneğin uygulamamızda, makinemizin girdi olarak bir resmi okuyabilmesi için 28x28 boyutlu bir matrisi (resmi) 28x28=784 elemanlı bir vektöre dönüştüreceğiz (dizi, matris, vektör vb. kavramların önceki öğrenmelerde öğrenildiği kabul edilmektedir).

Veri setinin incelenmesinin ardından, Python derin öğrenme kütüphanelerinden biri olan **keras** kütüphanesinden bahsedilir (kütüphane kavramının ve kurulumunun önceki öğrenmelerde öğrenildiği kabul edilmektedir). Keras, derin öğrenmeye giriş için çok kullanılan bir kütüphanedir. Öğrencilerin keras kütüphanesini kurmaları sağlanır.

Keras açıklandıktan sonra, uygulamada el yazısı rakamları sınıflandırmak için kullanılacak derin öğrenme algoritması **MLP (Multi Layer Perceptron – Çok Katmanlı Algılayıcı)** konusuna giriş yapılır. MLP; ileri beslemeli ve çok katmanlı bir yapay sinir ağıdır (İlgili kavramlar modülün önceki öğrenme alanlarında işlenmiştir). Bu noktada tahtaya/etkileşimli tahtaya yapay sinir ağı ve MLP yapısı çizilebilir.

Uygulamaya geçmek üzere, etkileşimli tahtada Python geliştirme ortamı açılır ve öğrencilerin de kendi bilgisayarlarında açmaları sağlanır. Aşağıdaki kodlar etkileşimli tahtada adım adım açıklanır ve öğrencilerin de kodları adım adım yazmaları ve programı çalıştırmaları sağlanır. Kodlarda geçen kavramlar, not satırlarında basitleştirilmiş düzeyde açıklanmıştır. İhtiyaç duyulursa, kavramların farklı kaynaklardan araştırılarak anlatılması önerilmektedir.

```
#İlgili kütüphanelerin, sınıflandırıcıların ve modellerin yüklenmesi
import keras
from keras.datasets import mnist
```

```
from keras.models import Sequential
from keras.layers import Dense
from keras.optimizers import RMSprop

iterasyon_sayisi = 64 # her bir iterasyonda "64" resim alınsın
rakam_sayisi = 10 # tanınmak istenen 0-9 rakam (10 sınıf)
epok_sayisi = 2 # eğitim 2 epok (eğitim devir sayısı) sürsün

#mnist veriseti rastgele karıştırılmış şekilde eğitim ve test verileri olarak yükleniyor
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# Sinir ağımız her eğitim örneği için tek bir vektör kullanacak, bu nedenle girdiyi 28x28 resim tek bir 784 boyutlu vektör olacak şekilde yeniden şekillendiriyoruz
x_train = x_train.reshape(60000, 784)
x_test = x_test.reshape(10000, 784)
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')

print(x_train.shape[0], 'eğitim verisiörnek sayısı')
print(x_test.shape[0], 'test verisi sayısı')

#sınıf vektörlerini ikili sınıf matrislerine dönüştürüyoruz
y_train = keras.utils.to_categorical(y_train, rakam_sayisi)
y_test = keras.utils.to_categorical(y_test, rakam_sayisi)

# Ağımızı kuralım
# Burada basit bir 3 katmanlı tam bağlantılı ağ yapacağız (fully connected network ("Dense"))
model = Sequential()
model.add(Dense(512, activation='relu', input_shape=(784,)))
```

	#Softmax katmanı ile çıktı değerlerimizin bir olasılık dağılımı olarak hesaplanmasını sağlıyoruz model.add(Dense(rakam_sayisi, activation='softmax')) model.summary() #Ağ derlenirken kayıp fonksiyonunu ve optimizatörünü belirtmemiz isteniyor #Optimizasyon algoritmalarından "RMSprop" ve yitim (loss) fonksiyonu olarak "categorical_crossentropy" kullanıyoruz #Değerlendirme parametresi olarak "accuracy" yi kullanıyoruz. Accuracy bize 0-1 arasında bir doğruluk değeri verecektir model.compile(loss='categorical_crossentropy', optimizer=RMSprop(), metrics=['accuracy']) #Şimdi ağımızı eğiteceğiz #Daha önce yüklenmiş eğitim verileri ile ağımız sınıflandırmayı öğrenecek #Fit fonksiyonu eğitim sırasındaki yitim (loss), başarım (accuracy) değerleri ile bir çok ayrıntı döndürür history = model.fit(x_train, y_train, batch_size=iterasyon_sayisi, epochs=epok_sayisi, verbose=1, validation_data=(x_test, y_test)) #Son olarak eğitilmiş ağımızın test verimiz üzerindeki performans değerlerini hesaplayarak ekrana yazdıralım. #Yani oluşturduğumuz yapay zekanın yüzde kaç oranında doğru tahmin yapabildiğini görelim score = model.evaluate(x_test, y_test, verbose=0) print('Test loss:', score[0]) print('Test accuracy:', score[1]) ***Not-1:*** Bu etkinlik, ilgili modülün önceki öğrenme alanları öğrenildikten sonra uygulanmalıdır. *Not-2:* Paylaşılan kodlar Python 3.8.8 sürümünde yazılmıştır. Farklı sürümlerde kodlar değişkenlik gösterebilir.
Değerlendirme	Farklı bir veri seti ve derin öğrenme algoritması ile benzer bir program yazdırarak öğrencilerin değerlendirilmesi önerilmektedir.

ETKİNLİK NO: 10

Ders	Bilişim Teknolojileri ve Yazılım
Program	*Özel Yetenekleri Geliştirme Programı*
Modül / Öğrenme Alanı	*Yapay Zeka/Makine Öğrenmesi*
Önerilen Süre	*2 Ders Saati*
Öğrenci Kazanımları	• Yapay zekâ kavramını tanır. • Yapay zekâ uygulamalarını ve günümüzdeki yapı örneklerini bilir. • Yapay zekâ kavramlarını listeler. • Görüntü işleme nedir bilir ve arka planda çalışma mantığı hakkında bilgi edinir. • Ses işleme nedir bilir ve arka planda çalışma mantığı hakkında bilgi edinir. • Veri işleme nedir bilir ve arka planda çalışma mantığı hakkında bilgi edinir. • Mblock 5 programını açmayı bilir. • Mblock 5 programının ara yüzünü tanır. • Mblock 5 programının web ortamında veya indirilebilir yapılarda geliştirilebildiğini fark eder. • Mblock 5 programındaki kod bloklarını bilir ve kullanır. • Mblock 5 programında eklenti yüklemeyi bilir. • Mblock 5 programı ile yapay zekâ arasındaki işbirliğini fark eder. • Mblock 5 programında geliştirdiği kodları çalıştırmayı öğrenir ve sahne ortamında gerçekleşmesini sağlar.
Öğretme-Öğrenme Yöntem ve Teknikleri	• *Problem çözme* • *Analitik düşünme* • *Keşfetme*
Araç-Gereçler ve Kaynaklar	*Bilgisayar* *İnternet* *Kamera (1080p)* *mBlock Uygulaması* *Renkli A4 Yazıcı*
Öğretme-Öğrenme Süreci	HAZIRLIKLAR: Burada kullanacağımız mblock 5 programı blok komutlar mantığıyla öğrencilerin kolay bir şekilde kodlama yapmalarına imkân verecektir. Aynı zamanda makeblock ve elektronik yapıların da kolayca kodlanmasını sağlar. Programımızı açmak veya indirmek için: https://www.mblock.cc/en-us/download/ web sayfasını tıklayalım. Programı ister burada online olarak geliştirebilir, istersek de bilgisayarımıza (Windows, Linux, MACOS)indirerek kurabiliriz. Mblock 5 sayfasını açtığımızda “download” kısmında” karşımıza gelen ekranda “Create in the browser” ile online kodlama sayfasına yönlendirileceksiniz. “Download” tıklayarak da bilgisayarınıza programı indirebilirsiniz. Bilgisayarımıza indirdikten sonra kurulum dosyasını açıyoruz.

	Kurulumu tamamlıyoruz. Kurulum tamamlandıktan sonra eklentiler/uzantılar bölümünden "Makine Öğrenmesi" adlı eklentiyi kuruyoruz. KAVRAMLAR: • ***Veri Seti*** • ***Model eğitimi*** • ***Makine Öğrenmesi*** ÖN BİLGİLER: • *Scratch ve mBlock vs. gibi ortamlarda orta düzey blok tabanlı kodlama bilgisi.* UYGULAMA: *Dikkat Çekme:* *Dersin dikkat çekme aşamasında yapay zeka ile neler yapılabileceği hakkında kısa ve ilginç videolar izlettirilerek bir beyin fırtınası yapılır.* *Etkinlik:* Uygulamayı bilgisayara yükleme ve kurma. Model eğitimi için internet üzerindeki kaynaklardan geç, dur, sağa dön ve sola dön vs. gibi el işaretleri resimleri bulunarak a4 renkli yazıcı ile kağıt üzerine çıktılar alınır. Daha sonra bu resimleri kamera yardımıyla tanıtılır. Model eğitimi sağlanır. Modele isimler verilir. Kodlaması yapılır. Yönerge: Her öğrenciye bilgisayar, kamera ve çıktı alabileceği bir bilgisayar sağlanır. Öğrenciler kendi elleri ile işaret yapabilir bunu çıktı alabilir veya internet üzerinden görsel kaynak taraması da yapabilir. İlgili çıktı alınan görseller mBlock Makine Öğrenimi eklentisi üzerinden tanıtılı ve makine öğrenmesinin sağlanabilmesi için en az 20 adet resmini kamera yardımıyla kaydeder. mBlock üzerinde bulunan veya internet üzerinde bulunan resimlerden trafiği işaret eden veya herhangi bir kavşağı gösteren arka plan resmi ve bir adet kuklaya giydirmek üzere otomobil resmi bulunur. Kodlama kısmında ise ilgili işaretler için kodlamaları yapılır. Uygulama çalıştırılır ve kameraya gösterilen el işaretleri ile kuklanın yani otomobilin hareket etmesi, durması, sağa dönmesi veya sola dönmesi sağlanır.
Değerlendirme	Farklı ham veriler kullanılarak görüntüler üzerinde veri seti oluşturma, bozuk verilerin ayrılması, model oluşturma, modelin eğitilmesi ve modelin farklı veriler ile başarımının test edilmesi beklenir.

ETKİNLİK NO: 11

Ders	Bilişim Teknolojileri ve Yazılım
Program	*Özel Yetenekleri Geliştirme Programı*
Modül / Öğrenme Alanı	*Yapay Zeka/Makine Öğrenmesi*
Önerilen Süre	*2 Ders Saati*
Öğrenci Kazanımları	• Yapay zekâ kavramını tanır. • Yapay zekâ uygulamalarını ve günümüzdeki yapı örneklerini bilir. • Yapay zekâ kavramlarını listeler. • Görüntü işleme nedir bilir ve arka planda çalışma mantığı hakkında bilgi edinir. • Ses işleme nedir bilir ve arka planda çalışma mantığı hakkında bilgi edinir. • Veri işleme nedir bilir ve arka planda çalışma mantığı hakkında bilgi edinir. • Mblock 5 programını açmayı bilir. • Mblock 5 programının ara yüzünü tanır. • Mblock 5 programının web ortamında veya indirilebilir yapılarda geliştirilebildiğini fark eder. • Mblock 5 programındaki kod bloklarını bilir ve kullanır. • Mblock 5 programında eklenti yüklemeyi bilir. • Mblock 5 programı ile yapay zekâ arasındaki işbirliğini fark eder. • Mblock 5 programında geliştirdiği kodları çalıştırmayı öğrenir ve sahne ortamında gerçekleşmesini sağlar.
Öğretme-Öğrenme Yöntem ve Teknikleri	• *Problem çözme* • *Analitik düşünme* • *Keşfetme*
Araç-Gereçler ve Kaynaklar	*Bilgisayar* *İnternet* *Kamera (1080p)* *mBlock Uygulaması* *Renkli A4 Yazıcı*
Öğretme-Öğrenme Süreci	HAZIRLIKLAR: Burada kullanacağımız mblock 5 programı blok komutlar mantığıyla öğrencilerin kolay bir şekilde kodlama yapmalarına imkân verecektir. Aynı zamanda makeblock ve elektronik yapıların da kolayca kodlanmasını sağlar. Programımızı açmak veya indirmek için: https://www.mblock.cc/en-us/download/ web sayfasını tıklayalım. Programı ister burada online olarak geliştirebilir, istersek de bilgisayarımıza (Windows, Linux, MACOS)indirerek kurabiliriz. Mblock 5 sayfasını açtığımızda “download” kısmında” karşımıza gelen ekranda “Create in the browser” ile online kodlama sayfasına yönlendirileceksiniz. “Download” tıklayarak da bilgisayarınıza programı

Co-funded by the Erasmus+ Programme of the European Union

	indirebilirsiniz. Bilgisayarımıza indirdikten sonra kurulum dosyasını açıyoruz. Kurulumu tamamlıyoruz. Kurulum tamamlandıktan sonra eklentiler/uzantılar bölümünden "Makine Öğrenmesi" adlı eklentiyi kuruyoruz. KAVRAMLAR: • ***Veri Seti*** • ***Model eğitimi*** • ***Makine Öğrenmesi*** ÖN BİLGİLER: *Scratch ve mBlock vs. gibi ortamlarda orta düzey blok tabanlı kodlama bilgisi.* UYGULAMA: *Dikkat Çekme:* *Dersin dikkat çekme aşamasında yapay zeka ile neler yapılabileceği hakkında kısa ve ilginç videolar izlettirilerek bir beyin fırtınası yapılır.* *Etkinlik:* Uygulamayı bilgisayara yükleme ve kurma. Model eğitimi için internet üzerindeki kaynaklardan geç, dur, sağa dön ve sola dön vs. gibi el işaretleri resimleri bulunarak a4 renkli yazıcı ile kağıt üzerine çıktılar alınır. Daha sonra bu resimleri kamera yardımıyla tanıtılır. Model eğitimi sağlanır. Modele isimler verilir. Kodlaması yapılır. Yönerge: Her öğrenciye bilgisayar, kamera ve çıktı alabileceği bir bilgisayar sağlanır. Öğrenciler kendi elleri ile işaret yapabilir bunu çıktı alabilir veya internet üzerinden görsel kaynak taraması da yapabilir. İlgili çıktı alınan görseller mBlock Makine Öğrenimi eklentisi üzerinden tanıtılı ve makine öğrenmesinin sağlanabilmesi için en az 20 adet resmini kamera yardımıyla kaydeder. mBlock üzerinde bulunan veya internet üzerinde bulunan resimlerden trafiği işaret eden veya herhangi bir kavşağı gösteren arka plan resmi ve bir adet kuklaya giydirmek üzere otomobil resmi bulunur. Kodlama kısmında ise ilgili işaretler için kodlamaları yapılır. Uygulama çalıştırılır ve kameraya gösterilen el işaretleri ile kuklanın yani otomobilin hareket etmesi, durması, sağa dönmesi veya sola dönmesi sağlanır.
Değerlendirme	Farklı ham veriler kullanılarak görüntüler üzerinde veri seti oluşturma, bozuk verilerin ayrılması, model oluşturma, modelin eğitilmesi ve modelin farklı veriler ile başarımının test edilmesi beklenir.

ETKİNLİK NO: 12

Ders	Bilişim Teknolojileri ve Yazılım / Not Defterimizi Yapıyoruz
Program	*Özel Yetenekleri Geliştirme Programı*
Modül / Öğrenme Alanı	*Yapay Zeka/Makine Öğrenmesi*
Önerilen Süre	*4 x 40dk*
Öğrenci Kazanımları	*Arama algoritmaları* *Makine Öğrenmesinde Sınıflandırma* *Makine Öğrenmesine Giriş*
Öğretme-Öğrenme Yöntem ve Teknikleri	• *Problem çözme* • *Analitik düşünme* • *Keşfetme* • *İşbirlikli öğrenme*
Araç-Gereçler ve Kaynaklar	*Bilgisayar* *İnternet* *Sözlük*
Öğretme-Öğrenme Süreci	HAZIRLIKLAR: Ders başlangıcında örnek bir metin editörü programının çalışma prensibi ile ilgili konuşulur. Yazım aşamasında kelimelerin nasıl algılatıldığı ve programın kelimeleri algılayıp ne şekilde kontrol ettiği anlatılır. Türkçe sözlük içinde bulunan bütün kelimelerin editör veri tabanına yüklendiği ve kontrol edildiği bilgisi verilir. Gerekli durumlarda kullanıcının kendi kelimelerini eklemesi için eklenti yapılacak bölümlerin ne şekilde hazırlanabileceği konusunda tartışılır. KAVRAMLAR: • ***Veri Seti:*** • ***Makine Öğrenmesi*** • ***Nesne Sınıflandırma*** ÖN BİLGİLER: • *Python programlama dilini bilir.* UYGULAMA: *Dikkat Çekme:*

	Bilgisayarda bir metin editöründe önceden belirlenmiş bir paragraf hızlı bir şekilde yazılır. Öğrenciden düzeltme yapmadan yazması istenir. Yazı tamamlandığında otomatik düzeltmeler ile hataların ayıklanması istenir. Tamamlama süresi not edilir. *İkinci bir öğrenciden de yazıyı yazarken hataları kendisinin düzeltmesi istenerek aynı metin yazdırılır. İki öğrencinin yazıyı tamamlama süreleri kontrol edilir ve hangi yöntemin daha kullanışlı olacağı hakkında fikirler beyan edilir.* *Etkinlik:* Öğrencilerin kendilerine ait basit bir not defteri uygulaması yapması istenir. Sözlükten veri tabanına aktarılan kelimelerin bilgisayar hafızasına alınması istenir. Daha sonra uygulama aşamasına geçilir. Yönerge: ● *İnternet ortamında not defteri ve metin editörü uygulamalarını inceleyiniz.* ● *Bu tür uygulamalar yapmak için genellikle kullanılan yazılım programlarını inceleyiniz.* ● *Kendinize ait bir arayüz oluşturunuz.* ● *Yazı fontlarının tanımlamasını yapınız.* ● *Büyük ve küçük harf tanım girişini yapınız.* ● *Metin programınızın çalışıp çalışmadığını kontrol ediniz.* ● *Hataları kontrol edip geliştirmeleri yapınız.* ● *Halihazırdaki diğer programlardan farkını açıklayınız.*
Değerlendirme	*Programın doğru şekilde çalışıp çalışmadığı öğrencilerle birlikte kontrol edilir.*

ETKİNLİK NO: 13

Ders	Bilişim Teknolojileri ve Yazılım / Bilgisayar Ekranına El Yazısı İle Çizilen Bir Numaranın Kaç Olduğunu Tahmin Eden Bir Uygulama
Program	Özel Yetenekleri Geliştirme Programı
Modül / Öğrenme Alanı	Yapay Zeka/Makine Öğrenmesi/ Derin Öğrenme
Önerilen Süre	4 x 40dk
Öğrenci Kazanımları	6.5.1.Yapay sinir hücresi kavramını açıklar. 6.5.4.Gerçek yaşam problemlerinin çözümü için yapay sinir ağlarını kullanır. 6.6.1.Temel makine öğrenmesi kavramlarını tanımlar. 6.6.2.Denetimli öğrenme kavramını açıklar. 6.6.3.Denetimsiz öğrenme kavramını açıklar. 6.6.4.Yarı-denetimli öğrenme kavramını açıklar. 6.6.5.Pekiştirmeli öğrenme kavramını açıklar. 6.6.6.Canlıların öğrenmesi ile makine öğrenmesi arasındaki ilişkiyi açıklar. 6.9. Derin Öğrenme 6.9.1. Veri seti kavramını açıklar. 6.9.2. Derin öğrenmenin temel kavramlarını açıklar. 6.9.3.Sık kullanılan derin öğrenme algoritmalarını açıklar. 6.9.4. Derin öğrenme algoritmalarını kullanarak bilgisayarlı görü probleminin çözümüne ilişkin bir uygulama geliştirir. 6.9.5. Derin öğrenme algoritmalarını kullanarak metin verisi probleminin çözümüne ilişkin bir uygulama geliştirir.
Öğretme-Öğrenme Yöntem ve Teknikleri	• Problem çözme • Analitik düşünme • Keşfetme • Disiplinlerarası düşünme
Araç-Gereçler ve Kaynaklar	> Bilgisayar > İnternet > Anaconda platformu bünyesinde veya ondan bağımsız bir IDE (Pycharm, Spider vs..)

Öğretme-Öğrenme Süreci	HAZIRLIKLAR: Dersin başlangıcında öğrencilere yaptırılacak olan uygulamanın kodları ve model eğitimi hazırlanarak ders öncesinde öğretmen tarafından uygulama yapılır. Her öğrencinin bilgisayarında yüklü bir IDE geliştirme ortamı olması gerekmektedir. KAVRAMLAR: ● ***Derin Öğrenme:*** Derin Öğrenme bir makine öğrenme yöntemidir. Verilen bir veri kümesi ile çıktıları tahmin edecek yapay zekayı eğitmemize olanak sağlar. ● ***OpenCV:*** OpenCV(Open Source Computer Vision Library), programlama fonksiyonlarının görüntü işleme kütüphanesidir . ● ***Keras Kütüphanesi:*** Keras, neredeyse her tür derin öğrenme modelini tanımlamak ve eğitmek için uygun bir yol sağlayan Python için bir derin öğrenme kütüphanesidir. ÖN BİLGİLER: ● Python programlama dilini bilir. UYGULAMA: *Dikkat Çekme:* Dersin başlangıcında öğrencilere "Bilgisayar, ekranına çizilen bir rakamı tanıyabilir mi?" sorusu sorularak öğrencilerde merak uyandırılması sağlanır. Öğrencilerden alınan dönütler öğretmen tarafından değerlendirilerek, yapay zeka, derin öğrenme kavramları ve işleyiş şekli ile ilgili temel bilgiler verilir. Ardından derin öğrenme kütüphanelerinden OpenCV, Keras kütüphaneleri ve fonksiyonları tanıtılır. Öğrencilere başlangıçta sorulan bu işlemin mümkün olup olmadığını gözlemlemek için uygulama yapmaya teşvik edilerek etkinliğe başlanır. *Etkinlik:* Bilgisayarda yüklü olan herhangi bir python IDE sine proje ve model kodları aktarılır. Modelin eğitilmesi beklenir. Model eğitimi tamamlanınca proje kodları çalıştırılarak ekrana fare yardımıyla bir rakam yazılması istenir. Yönerge: ● Keras kütüphanesinin ne olduğunu inceleyiniz. ● Yapılacak olan etkinlikte neden Keras kütüphanesinin kullanıldığını araştırınız. ● Projenin ve modelin kodlarını simülasyon ortamına aktarınız. ● Öğrenme modelinin oluşturulması için bir süre bekleyiniz. ● Modelinizi eğitme süreci bitince, ekrana yazdığınız numaraları tanıması için çalıştırınız.

Co-funded by the Erasmus+ Programme of the European Union

	• Modelin öngördüğü rakamın, yazdığınız rakamla aynı olup olmadığını kontrol ediniz. GÖRSEL KAYNAKÇA Video https://jn7.net/python-opencv-ve-tensorflow-ile-el-yazisi-tanima-uygulamasi/ KAYNAKÇA Proje ve Model Kodları: https://github.com/jagadishb1409/Digit-recognition Detaylı Bilgiler https://jn7.net/python-opencv-ve-tensorflow-ile-el-yazisi-tanima-uygulamasi/
Değerlendirme	*Değerlendirme formu*