Uykulu Sürücü Tespiti
Sleepy Driver Detection

Özgenur Saygı

Bilişim Sistemleri Mühendisliği
Kocaeli Üniversitesi
nursyg.99@gmail.com

Özet

Bu projede amaçlanan görüntü işleme ile araç sürücüsünün uyuşukluk tespiti ve sürücüyü uyaran bir sistem oluşturmaktır. Uyuşukluk algılama ile sürüş sırasında uykuya dalmış sürücülerin neden olduğu kazalar önlenebilecektir. Kameradan alınan görüntü ile sürücünün uykulu olup olmadığı görüntü işlenerek tespit edilmektedir. Daha sonra sürücünün gözleri kapalıysa yani uyuyor durumunda ise Firebase üzerinden gerçek zamanlı olarak NodeMCU ile haberleştirildi. Firebase üzerinden alınan bilgi ile sürücünün uyanması için buzzer ile bir ses verilmektedir.Aynı zamanda arkasındaki ve önündeki arabaları uyarmak için arabanın içerisinden belirli aralıklarla ışık yakıp söndürülmektedir. Sürücü kendisi uyanıp butona basıp durdurana kadar ses ve ışık aktif halde olmaktadır.

Anahtar kelimeler: Görüntü İşleme,Firebase,NodeMCU

Abstract

The aim of this project is to detect driver drowsiness with image processing and create a system that warns the driver.Accidents caused by drivers who fell asleep while driving can be prevented with drowsiness detection.With the image taken from the camera, it is determined whether the driver is sleepy or not by processing the image. Then, if the driver's eyes were closed, that is, she was sleeping, she was communicated to NodeMCU in real time via Firebase.With the information I get from Firebase, a buzzer is given to wake up the driver. At the same time, lights are turned on and off at certain intervals from inside the car to warn the cars behind and in front of them.Sound and light are activated until the driver wakes up, presses the button and stops.

Keywords: Image Process,Firebase,NodeMCU

1. Gereksinimler

Bu projede ki işlemler için kurulması gereken bir takım programlar ve paketler vardır. Sisteminizde Python kurulu olmalıdır, ardından console ekranından pip komutu kullanarak gerekli paketleri kurabilirsiniz.Bu projede IDE olarak Jupyter Notebook kullanılmıştır. Jupyter Notebook, çeşitli programlama dilleri için etkileşimli bir ortam sağlayan açık kaynak kodlu bir programdır. Kurulması gereken paketler şunlardır;

1. OpenCV - pip install opencv-python (Yüz ve göz algılama)

2. TensorFlow - pip install tensorflow (Keras arka planda TensorFlow kullanır)

3. Keras - pip install keras (Sınıflandırma modelimizi oluşturmak için)

1.1 OpenCv Nedir ?

OpenCV (*Open Source Computer Vision Library*, anlamı *Açık Kaynak Bilgisayar Görüsü Kütüphanesi*) gerçek-zamanlı bilgisayar görüsü uygulamalarında kullanılan açık kaynaklı kütüphanedir.C++, Python ve Java arayüzlerine sahiptir ve Windows, Linux, Mac OS, iOS ve Android'i destekler.OpenCV, yüzleri ve nesneleri algılama ve tanımlama, videolarda insani eylemleri sınıflandırma, kamera hareketlerini ve hareketli nesneleri izleme, nesneleri 3 boyutlu modellerine ayıklama, görüntüleri yüksek çözünürlükte birleştirme gibi alanlarda başarı ile kullanılabilir.

1.2 TensorFlow Nedir ?

Açık kaynak kodlu bir deep learning(derin öğrenme) kütüphanesidir. Esnek yapısı sayesinde, tek bir API ile platform farketmeksizin hesaplamaları, bir veya birden fazla CPU, GPU kullanarak uygulamanıza olanak sağlar. Temelinde Python kullanılarak geliştirilen bu framework, günümüzde Python'ın yanısıra C++, Java, C#, Javascript ve R gibi birçok dili desteklemektedir.

1.3 Keras Nedir ?

Keras, Python'da yazılmış açık kaynaklı bir sinir ağı kütüphanesidir.Derin öğrenme konusunda Keras oldukça önemli paketlerden biri. Aslında Keras kendi başına bir derin öğrenme kütüphanesi değildir. Keras, size Google Tensorflow, Microsoft CNTK, ve Theano derin öğrenme kütüphanelerini kullanabileceğiniz üst seviye bir API (application programming interface, uygulama programlama arayüzü) sunuyor. Bu sayede oluşturduğunuz derin öğrenme mimarisini farklı paketleri kullanarak eğitmeniz mümkündür.

2. Görüntü Üzerindeki Nesne Tespiti

Adım 1 : Proje dosyasına öncelikle OpenCv kütüphanesini kullanabilmek için cv2 sınıfı import edildi. Görüntü anlık olarak kameradan girdi olarak alınıyor. Bu yüzden web kamerasında her kareyi yakalayabilmek için sonsuz bir while döngüsü oluşturduk. Kameraya erişmek için OpenCV, cv2.VideoCapture (0) tarafından sağlanan yöntemi kullanıyoruz .Kameraya erişimi sağladıktan sonra sonsuz döngü içerisinde webcam.read () yöntemi ile her kare okunmaktadır.Bu read() fonksiyonu geriye iki tane değer döndürür.Bunlardan ilki kameradan görüntü alındığını kontrol edilmesini sağlayan boolean bir değerdir,ikincisi ise görüntüdür.

```
import cv2
import numpy as np
from keras.models import load_model

labels_dict={0:'Open',1:'Close'}
color_dict={0:(0,255,0),1:(0,0,255)}

size = 4
webcam = cv2.VideoCapture(0)

classifier = cv2.CascadeClassifier('hcf\haarcascade_frontalface.xml')
score=0
while True:
    (rval, im) = webcam.read()
```

Şekil 2: Projeden kod bloğu 1

Adım 2: Kameradan alınan görüntüyü kameranın aynalama özelliği yüzünden çevirmemiz gerekmektedir.Görüntüyü yatay eksende zıt yönde

çevrilmesi gerekmektedir.Bu işlemi cv2.flip() yöntemi ile yapıyoruz.İlk girdi olarak bu işlev, çevirmeyi uygulamak istediğimiz görüntüyü alır. İkinci girdi olarak, istediğimiz çevirme yönünü temsil eden bir tamsayı alır.Bu projede yatay eksende çevrildiği için 1 değerini aldı.

```
while True:
    (rval, im) = webcam.read()
    im=cv2.flip(im,1,1)
```

Şekil 2: Projeden kod bloğu 2

Adım 3: Bundan sonraki işlem ise görüntünün boyutunu küçültmektir.Görüntüyü daha küçük boyuta getirilmesinin nedeni ise yazdığımız programın daha hızlı işlem yapmasını sağlanmıştır.Çünkü alınan her görüntü bir matristen oluşmaktadır.Görüntüyü küçülterek matrisin boyutunu da küçültmüş oluruz.Bir görüntüyü yeniden boyutlandırmak için OpenCV, cv2.resize () işlevi kullanılır. Görüntüdeki algılanması gereken nesne yüzdür ve ilgili alanın olduğu bölgeye (ROI) denilmektedir. Orijinal görüntü üzerinde sadece ile ilgili nesnenin olduğu bölümün alınmasıdır. Görüntüdeki yüzü algılamak için, haar cascade sınıflandırıcı kullanacağız. Bu, makine öğrenimine dayalı bir yaklaşımdır. Cascade işlevi, birçok pozitif ve negatif görüntüden eğitilir. Daha sonra diğer görüntülerdeki nesneleri tespit etmek için kullanılır. Bu kod parçacağı, classifier= cv2.CascadeClassifier('hcf\haarcascade_frontalface .xml') sınıflan dırıcıyı ayarlamak için kullanılır. Ardından, face= classifier.detectMultiScale (mini) kullanarak algılamayı gerçekleştiririz . Bu fonksiyon algılanan nesnenin sınırlarını yani genişliği, yüksekliği x, y koordinatları ile ilgili bir dizi döndürür. Artık yüzler üzerinde yineleme yapabilir ve her yüz için sınır kutuları çizebiliriz.

```
mini = cv2.resize(im, (im.shape[1] // size, im.shape[0] // size))

faces = classifier.detectMultiScale(mini)

for f in faces:
    (x, y, w, h) = [v * size for v in f]
    face_img = im[y:y+h, x:x+w]
```

Şekil 3: Projeden kod bloğu 3

Adım 4: Göz durumunu tahmin etmek için CNN sınıflandırıcı kullanıyoruz . Görüntümüzü modele beslemek için belirli işlemleri gerçekleştirmemiz gerekir çünkü modelin input katmanı için sabit boyutlara ihtiyacı vardır. Ardından, modelimiz 100*100 piksel görüntüler üzerinde eğitildiği için görüntüyü 100 * 100 piksel olarak yeniden boyutlandırılır.Bu işlemi cv2.resize (face_img, (100,100)) fonksiyonu ile gerçekleştirildi. Verilerimizi daha iyi yakınsama için normalize ediyoruz.Bu işlem bu kod satırı normalized= resized/255.0 (Tüm değerler 0-1 arasında olacaktır) ile gerçekleşmektedir.

Modelimizi model = load_model ('models/ model_generator.h5') kullanarak yükledik .Şimdi her bir yüzü modelimize predict fonksiyonu ile göndererek tahmin etmesini sağlıyoruz . Bu kod satırı result = model.predict (image) ile bu işlemi gerçekleştiriyoruz.Gelen değer result= 0 ise gözlerin açık olduğunu, result=1 ise gözlerin kapalı olduğunu belirtir.Gelen değer “0” ise yani gözler kapalı ise sınır çerçevesi kırmızı renk ve close etiketi yazmaktadır.Eğer “1” ise yani göz açık ise sınır çerçevesi yeşil renk ve open etiketi yazmaktadır.

```
for f in faces:
    (x, y, w, h) = [v * size for v in f]
    face_img = im[y:y+h, x:x+w]
    resized=cv2.resize(face_img,(100,100))
    normalized=resized/255.0
    reshaped=np.reshape(normalized,(1,100,100,3))
    reshaped = np.vstack([reshaped])
    result=model.predict(reshaped)
    print(result[0][0])
    label=result[0][0]
    n=label
    etiket=0
    d=float(f'{n:f}')
    if d < 0.000001:
        #print("küçük")
        etiket=1
    else:
        #print("büyük")
        etiket=0
    cv2.rectangle(im,(x,y),(x+w,y+h),color_dict[etiket],2)
    cv2.rectangle(im,(x,y-40),(x+w,y),color_dict[etiket],-1)
    cv2.putText(im, labels_dict[etiket]+str(score), (x, y-10),
```

Şekil 4: Projeden kod bloğu 4

Adım 5 : Cv2 kütüphanesi imshow() fonksiyonu ile alınan görüntülerin gösterilmesi sağlandı.Yani webcam den alınan görüntü anlık olarak gösterilmektedir. Cv2.waitKey() bir klavye bağlama fonksiyonudur. Bu fonksiyon herhangi bir klavye olayı için belirtilen milisaniyeleri bekler. O süre zarfında bir tuşa basarsanız program devam eder. Eğer "0" ataması yapılırsa sonsuza kadar bekler. Burada seçtiğimiz 'esc' tuşu ile kamerayı kapatabilirsiniz. Release komutu VideoCapture sınıfını kapatır.

```
    cv2.imshow('LIVE',   im)
    key = cv2.waitKey(5)

    if key == 27: #ESC
        break
webcam.release()

cv2.destroyAllWindows()
```

Şekil 5: Projeden kod bloğu 5

3. Sınıflandırma Modelini Oluşturma

3.1 Image Data Generator

ImageDataGenerator,görüntü büyütme tekniği, veri kümenizin boyutunu genişletmenin iyi bir yoludur. Orijinal veri kümesinden yeni dönüştürülmüş görüntüler elde edebilirsiniz. Derin ağlar, iyi bir performans elde etmek için büyük miktarda eğitim verisine ihtiyaç duyar. Az eğitim verisi kullanarak güçlü bir görüntü sınıflandırıcısı oluşturmak için, derin ağların performansını artırmak için görüntü büyütme genellikle gereklidir. Görüntü büyütme, rastgele döndürme, kaydırma, kesme ve çevirme gibi farklı işleme yöntemleri veya çoklu işlemlerin kombinasyonu yoluyla yapay olarak eğitim görüntüleri oluşturur.Keras'ta ImageDataGenerator API'si kullanılarak artırılmış bir görüntü oluşturucu kolayca oluşturulabilir. ImageDataGenerator, gerçek zamanlı veri artırma ile görüntü verisi yığınları oluşturur.ImageDataGenerator oluşturmak ve yapılandırmak ve artırılmış görüntülerle derin sinir ağını eğitebiliriz. Modele aktarılırken her eğitim görüntüsüne rastgele dönüşümler uygulayabilirsiniz. Bu görüntü büyütme teknikleri yalnızca veri kümesinin boyutunu artırmak ile kalmaz, aynı zamanda veri kümesinde modelinizin görünmeyen veriler üzerinde daha iyi tahmin yapmasına olanak sağlar.
Projemize öncelikle keras.preprocessing.image kütüphanesinden ImageDataGenerator sınıfını import edilmesi gerekmektedir. Daha sonra veri setimiz üzerinde kullanılmak üzere ImageDataGenerator sınıfından bir nesne oluşturuyoruz. Bu sınıf bir takım parametreler almaktadır.Bu parametrelerden bazıları şu şekildedir:

Random Shifts: Görüntüleri çevirmek de harika bir büyütme tekniğidir. ImageDataGenerator sınıfı, dikey veya yatay eksen boyunca çevirmek için horizontal_flip ve vertical_flip parametrelerine sahiptir .Bu projede sadece horizantal_flip uygulandı.

Random Zoom: Yakınlaştırma artırma, görüntüyü yakınlaştırır veya görüntüyü uzaklaştırır. ImageDataGenerator sınıf yakınlaştırma için zoom_range argümanı bir float değeri alır . 1'den küçük herhangi bir değer görüntüye yaklaşacaktır. 1'den büyük herhangi bir değer görüntüyü uzaklaştıracaktır.

Shear Range: Kayma yoğunluğu radyan cinsinden kayma açısı.

Rescale: Yeniden ölçekleme parametresidir. Veriler girilen değerle çarpılır.

Oluşturduğumuz nesneyi veri setimize Flow_from_directory fonksiyonu ile uygulayacağız.Yöntem, görüntüleri doğrudan dizinden okumanıza ve sinir ağı modeli eğitim verilerinden öğrenirken onları artırmanıza olanak tanır.Yöntem, farklı sınıflara ait görüntülerin farklı klasörlerde, ancak aynı ana klasörde bulunmasını bekler.Bu fonksiyon da bir takım parametreler almaktadır. Aşağıdakiler, bu yöntemin birkaç önemli parametresidir:

Directory: İlk parametre,eğitim için kullanılacak görüntülerin bulunduğu klasörün yoludur.

Target_size : Giriş resminin boyutudur.Klasördeki eğitim için kullanılacak görüntülerimizin hepsinin boyutlarını eşit hale getiriyoruz

Batch_size : Veri yığınlarının boyutudur.

Class_mode : Binary olarak ayarlandı. İki tane sınıf etiketimiz olduğu için ikili sınıflandırma uygulandı.

```
batch_size = 16
train_datagen = ImageDataGenerator(
        rescale=1./255,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True)
test_datagen = ImageDataGenerator(rescale=1./255)
train_generator = train_datagen.flow_from_directory(
        'data/train',
        target_size=(100, 100),
        batch_size=batch_size,
        class_mode='binary')
validation_generator = test_datagen.flow_from_directory(
        'data/validation',
        target_size=(100, 100),
        batch_size=batch_size,
        class_mode='binary')
```

```
Found 2002 images belonging to 2 classes.
Found 433 images belonging to 2 classes.
```

Şekil 6: Projeden kod bloğu 6

3.2 Evrişimsel Sinir Ağları

Konvolüsyonel sinir ağı (CNN- Convolutional Neural Networks) olarak da adlandırılan ESA görüntü tanıma ve nesne sınıflandırma gibi modellerin oluşturulmasında büyük rol oynamaktadır. ESA yapılan çalışmalarda insan ile kıyaslamada daha üstün sonuçlar elde etmiştir. Birincil amaç olarak görüntü sınıflandırma üzerine yoğunlaşan ESA'lar son yıllardır diğer alanlarda da yoğun bir şekilde kullanılmaktadır. CNN Modelini Oluşturan Katmanlar şu şekildedir;

Keras Conv2D Katmanı

2D evrişimli katmanlar, üç boyutlu bir girdi, tipik olarak üç renk kanallı bir görüntü alır. Görüntünün üzerine, aynı zamanda evrişim çekirdeği (kernel) olarak da adlandırılan bir filtre geçirirler.Bir seferde küçük bir piksel penceresini, örneğin 3 × 3 veya 5 × 5 piksel boyutlarını inceleyerek ve tüm görüntüyü tarayana kadar pencereyi hareket

ettirirler. Evrişim işlemi, filtrede tanımlanan ağırlıklarla mevcut filtre penceresindeki piksel değerlerinin nokta çarpımını hesaplar.Conv2D katmanı bir takım parametreler almaktadır.

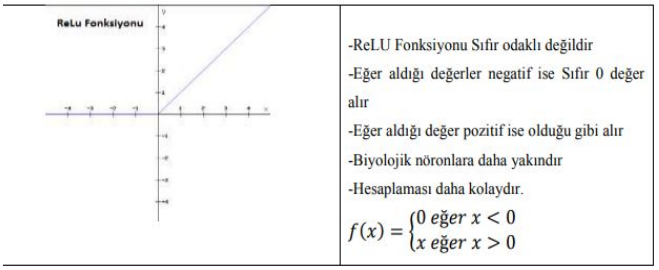
Filters: Bu katmanın ilk parametresidir. Evrişim işleminde kullanılan filtre sayısını ayarlar. Seçtiğiniz filtre sayısı, veri kümenizin karmaşıklığına ve sinir ağınızın derinliğine bağlı olmalıdır.Bu projede filtre 128 seçilmiştir.

Kernel size: Evrişim filtresinin boyutunu piksel cinsinden belirtir.Filtre boyutu, kullandığınız CNN mimarisine göre belirlenebilir.Resimleriniz 128 × 128'den küçükse, daha küçük 1 × 1 ve 3 × 3 filtreleriyle çalışmak daha yaygındır.Bu projede filtre boyutu 3 x 3 seçilmiştir.

Input Shape : Giriş görüntüsünün genişliğini,yüksekliğini ve derinlik boyutunu belirlediğimiz parametredir.Örneğin, bir RGB görüntünün derinliği 3 olacaktır.Bu projedeki veri setindeki görüntüler 100 x 100 boyutunda ve renkli görüntüler olduğu için 3 kanallıdır.

Düzleştirilmiş Doğrusal Birim Katmanı(ReLu)

Aktivasyon fonksiyonlarının en büyük özelliği sinir ağının doğrusal olmayan özellik kazandırmasıdır. Gizli katmanlarda hesaplanan matris çarpımı sonucu ortaya çıkan nöron ağırlıkları bir sonraki katmana doğrusal olmayan bir yapıda giriş olarak hesaplanmaktadır. Yani aktivasyon fonksiyonları çok katmanlı yapay sinir ağlarında doğrusal olmayan dönüşüm işlemleri için kullanılmaktadır.Bu aktivasyon fonksiyonlarından en başarılı sonuç veren Relu fonksiyonudur.



Şekil 7: Relu fonksiyonu çalışma prensibi

Havuzlama (Pooling Layer) Katmanı

Havuzlama katmanı uygulamasında temel amaç, kendinden sonra gelen Konvolüsyon katmanı için giriş boyutunu azaltmaktır. Genellikle ReLu katmanından sonra gelir. Bu katman sonucu boyuttaki azalma bilgi kaybına sebep olur. Bilgi kaybından dolayı bir sonraki katmanlarda daha az hesap yükü oluşturur ve sistemin ezberlemesini önler.

Pool Size : 2 tamsayıdan oluşan bir tuple, maksimumun alınacağı pencere boyutu. (3, 3) maksimum değeri 3x3 havuz oluşturma penceresi üzerinden alır.

Flatten Katmanı

Bu katmanın görevi basitçe, son ve en önemli katman olan Fully Connected Layer'ın girişindeki verileri hazırlamaktır. Genel olarak, sinir ağları, giriş verilerini tek boyutlu bir diziden alır. Bu sinir ağındaki veriler ise Convolutional ve Pooling katmanından gelen matrislerin tek boyutlu diziye çevrilmiş halidir.

Dropout Katmanı

Evrişimsel sinir ağında eğitim aşamasında fazla eğitimden dolayı bazen ağ ezberleme yapar. Ağın ezber yapmasının önüne geçmesi için bu katman kullanılır.Bu katmana 0.5 değeri verildi.

Son eklediğimiz katman ise çıktı katmanıdır.Çıktı katmanındaki nöron sayısını veri setimizde ki sınıf sayısı kadar veriyoruz.Aktivasyon fonksiyonu olarak sigmoid kullanıldı.Projede kullanılan model de yukarıda açıklaması bulunan katmanlar kullanılmıştır.Toplamda 1 tane giriş katmanı 3 gizli katman 1 tane çıkış katmanı bulunmaktadır. Katmanların içerisinde ki parametrelerin bu projede hangi değerler verildiği yukarıda açıklamalarda belirtilmiştir.Model ile ilgili görsel aşağıdadır.

```
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D
from keras.layers import Activation, Dropout, Flatten, Dense

model = Sequential()
model.add(Conv2D(128, (3, 3), input_shape=(100,100,3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(32, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(64))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(1,activation="sigmoid"))

model.compile(loss='binary_crossentropy',optimizer='rmsprop', metrics=['accuracy'])
```

Şekil 8: Projeden kod bloğu 8

Bir sonraki aşamada modelimizi compile fonksiyonu ile derlemeye başlıyoruz.

optimizer : Eniyileme algoritmaları, kayıp fonksiyonuna göre ağın nasıl güncelleneceğini belirler. Optimizer, öğrenme oranını kontrol eder.Bu projede ağırlık katsayılarının güncellenmesi için kullanılacak optimizasyon yöntemi ‘rmsprop’ dır.

loss : Gerçek değer ile tahmin edilen değer arasındaki hatayı ifade eden metrik. Temelde **“**binary_crossentropy**”** , ikili doğruluğu gösterir. İkili sınıflı sınıflandırma problemleri için kullanılır.

metrics: Model değerlendirme kriterleri belirlenir. Eğitim ve test sırasında değerlendirmek istediğim metrik accuracy (doğruluk) değeridir.

Modeli derledikten sonra eğitim sürecine başlayabiliriz. Bunu, Keras’taki fit_generator () fonksiyonu kullanılarak yapılabilir.Bu fonksiyon da bir takım parametreler almaktadır.

İlk argüman, flow () veya flow_from_dataframe () veya flow_from_directory () yönteminden elde ettiğimiz görüntülerimizin oluşturduğu veri setidir.

Epoch: Bu değer, model eğitilirken verilerin modelden kaç kez geçiş yapacağını belirtir

Steps_per_epoch :bir eğitim dönemi bitmiş sayılmadan önceki toplu yineleme sayısıdır.

Validation_data : Doğrulama seti olarak bilinir. Bu parametreye veri setimizden test için ayırdığımız değerleri veriyoruz.

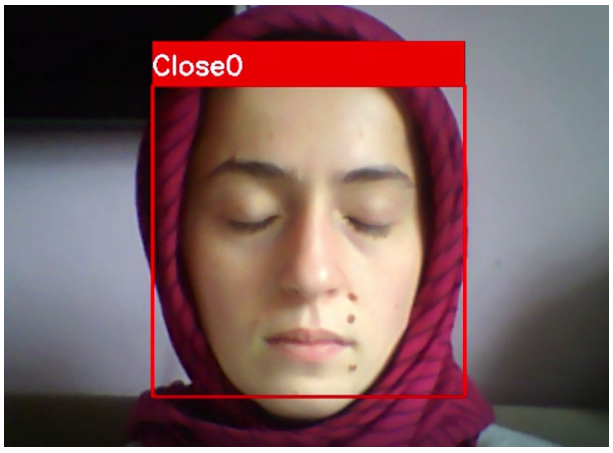
Validation_steps : steps_per_epoch'a benzer, ancak test veri kümesi üzerinde işlem yapar.

```
model.fit_generator(
        train_generator,
        steps_per_epoch=2002 // batch_size,
        epochs=50,
        validation_data=validation_generator,
        validation_steps=433 // batch_size)
```

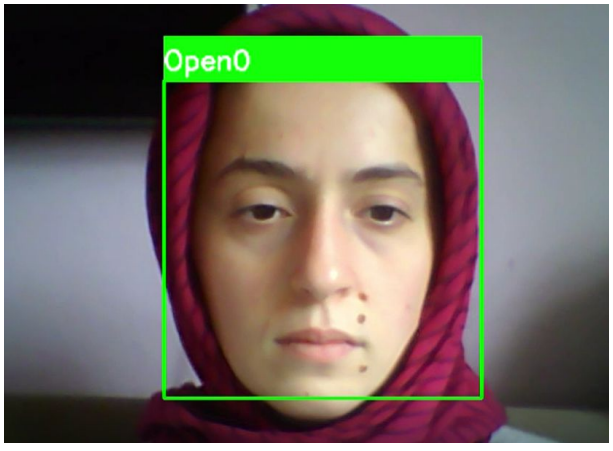
```
125/125 [==============================] - 90s 722ms/step - loss: 0.0890 - accuracy: 0.9718 - val_loss: 0.0358 - val_accurac
y: 0.9664
Epoch 46/50
125/125 [==============================] - 90s 719ms/step - loss: 0.0946 - accuracy: 0.9770 - val_loss: 0.1166 - val_accurac
y: 0.9448
Epoch 47/50
125/125 [==============================] - 93s 742ms/step - loss: 0.0636 - accuracy: 0.9802 - val_loss: 0.2631 - val_accurac
y: 0.9688
Epoch 48/50
125/125 [==============================] - 90s 716ms/step - loss: 0.0899 - accuracy: 0.9753 - val_loss: 0.1038 - val_accurac
y: 0.9712
Epoch 49/50
125/125 [==============================] - 90s 718ms/step - loss: 0.0790 - accuracy: 0.9740 - val_loss: 0.0083 - val_accurac
y: 0.9712
Epoch 50/50
125/125 [==============================] - 89s 713ms/step - loss: 0.0853 - accuracy: 0.9768 - val_loss: 1.2389e-05 - val_accu
racy: 0.9640

<keras.callbacks.callbacks.History at 0x135bdfc9b88>
```

Şekil 9: Modelin başarı değerleri

Aşağıdaki resimler projenin çalışır halinden çekilmiştir.Gözler kapalı durumda ise sınır çerçevesi kırmızı renk ve close etiketi yazmaktadır.Gözler açık ise sınır çerçevesi yeşil renk ve open etiketi yazmaktadır.



Şekil 10: Gözler kapalı iken verilen uyarı



Şekil 11: Gözler açık iken verilen uyarı

Sürücünün durumu ile ilgili bilgi Firebase veri tabanında saklanmaktadır. Sürücünün gözleri belirli süre 5-6 sn yakın kapalı kaldıysa Firebase sunucusunda "nodemculed" veritabanı içerisinde ilgili alana "Close" değeri yazılmaktadır.Firebase içerisindeki verileri json formatında saklamaktadır. Öncelikle Python'da Firebase'i kullanmak için bazı paketler yüklemeniz gerekir. Bu paketler sırasıyla şunlardır ;

- ➔ *pip install requests*
- ➔ *pip install python-firebase*

Paketleri yükledikten sonra projemize aşağıdaki görseldeki gibi dahil ediyoruz.

```
from firebase import firebase
import json
```

FirebaseApplication fonksiyonu iki parametre alıyor; biri veritabanının URL'si ve ikincisi veritabanı için kimlik doğrulama ayrıntıları ile ilgilidir. Veritabanı ile bağlantı işlemini gerçekleştirdikten sonra put komutu veritabanımızda "Value" alanını sürücünün durumuna göre "Close" değerini yazdırıyoruz.

```
firebase = firebase.FirebaseApplication('https://nodemculed-f73f0.firebaseio.com/',
                                        None)
```

```
if(sure>30):
    firebase.put('/RandomVal/','Value',"Close")
    print("Record updated")
```

4.NodeMCU İle Sürücü Uyarı Sistemi

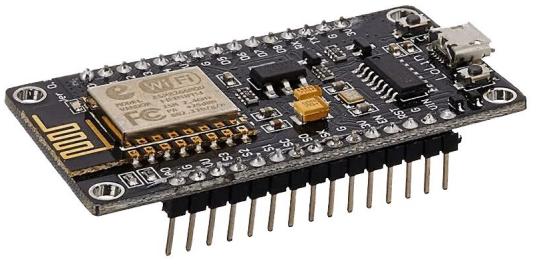
Projenin son aşamasında ise Firebase veritabanından sürücünün durumu ile ilgili bilgiyi buluttan çekebilmek için NodeMCU kartı

kullanıldı ve gerekli uyarı sistemi yapıldı. NodeMCU ile sürücünün durumu ile ilgili bilgiyi çekiyoruz ve bu bilgiyi kullanarak gerekli kodlarımızı yazarak sürücüyü uyarı için kullanılacak ışık ve ses aktif hale getirilmektedir. NodeMCU bağlanılan buzzer ile ayarlanan alarm çalmaya başlıyor,çıkan ses ile sürücünün uyanması hedeflenmektedir. Belirli süre ile yanıp sönen ışık ile diğer araçlarda ki sürücülerin dikkatli olması için uyarılması hedeflenmektedir. Sürücü kendisi uyandıktan sonra ister butona basıp uyarı sistemini durdurabilir yada mobil uygulamadan da uyarı sistemini pasif hale getirebilir.

Bu bölümde öncelikle NodeMCU internete ve firebase bağlamak için gereken ayarlar ve kodlar anlatılmaktadır. Daha sonra NodeMCU ile buzzer, şerit led ve buton bağlantısı yani devre şeması anlatılmaktadır. Gelen bilgiye göre de sırasıyla hangi devre elemanlarının ne kadar süre ile aktif olacağı kodlar ile de kontrol etmektedir.Bundan sonraki bölümde bu işlemler ile ilgili detaylı bilgiye yer verildi.

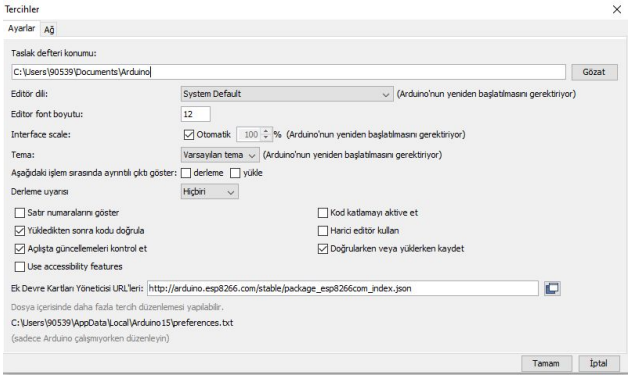
4.1 NodeMCU Ayarları ve Firebase Bağlantısı

NodeMCU; minik bir elektronik devredir. Açık kaynaktır, ucuzdur ve yeteneklidir. Düşük gerilimli enerjiyle çalışır. Üzerinde çok sayıda bağlantı noktaları vardır. Bu bağlantı noktalarını kullanarak bağlayacağınız başka elektonik bileşenleri yönetebilirsiniz. Barındırdığı WiFi entegresi sayesinde kolayca IOT yani internet şeyleri olarak bilinen cihazlar yapmanıza olanak sağlamaktadır.Arduino IDE ve Arduino'nun kullandığı dille de programlanabilir. USB kablosuyla bilgisayara kolayca bağlanır, programlanabilir ve veri iletişim kurulabilir.



*Şekil 12:*NodeMCU devre kartı

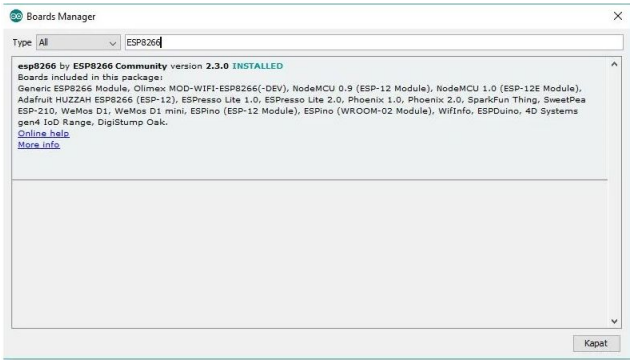
Öncelikle diğer geliştirme kartlarını Arduino IDEye tanımlarken kullandığımız yöntem ile kartımızı Arduino IDEye ekleyeceğiz. Önce Arduino IDE mizi açıyoruz ve Dosya—Tercihler bölümüne giriyoruz burada en altta Additional Board manager URLs in yanındaki kutucuğa ***http://arduino.esp8266.com/stable/package_esp8266com_index.json*** linkini yapıştırıyoruz.



*Şekil 13:*Arduino IDE tercihler menüsü

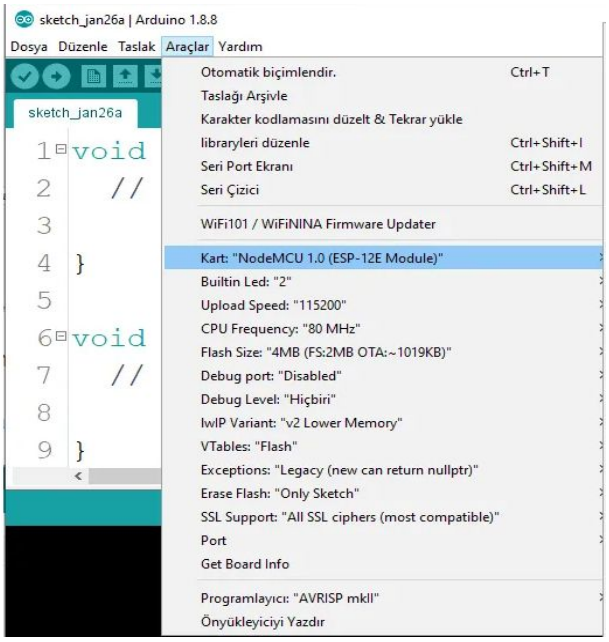
ESP8266 kütüphanesini IDEye kurduktan sonra programımız, kartımızı kodlamaya uygun hale gelmiş oluyor.

Kütüphaneyi de ekledikten sonra artık kodlarımızı NodeMCU üzerinde ki hafızaya yükleyebiliriz.



*Şekil 14:*Board manager penceresi

Kodumuzu ise NodeMCU içerisine yükleyebilmek için de bir takım ayarlar yapmaktadır.Kart ve port ayarlarınızı doğru seçtikten sonra kodu yükle butonuna basıyoruz ve genel bilgileri log ekranında görüyoruz, kod aktarımı bittikten sonra Log penceresinde "Yükleme Tamamlandı" yazmaktadır.



*Şekil 15:*Araçlar menüsü kart seçimi

İlk olarak proje dosyamıza esp8266 kütüphanesini import ediyoruz. Bu kod satırı "#include <ESP8266WiFi.h>" ESP8266 kütüphanesi dahil ediliyor. NodeMCU ile internete bağlanmak için internet ağınızın kullanıcı ismi ve şifresini (SSID , PASSWORD) tanımlıyoruz.

```
const char* ssid = "[redacted]"; //--> Wifi adı
const char* password = "[redacted]"; //--> Wifi şifresi
```

*Şekil 16:*Wifi kullanıcı adı ve şifresi

Serial.begin() komutu ile seri haberleşme başlatılıyor.Wifi.begin() komutu ile bağlanılacak Ağın adı ve şifresi girilen internet ağına bağlanılıp ardından WiFi başlatılıyor . Bu kod ile "while (WiFi.status() != WL_CONNECTED)" döngüsü ile ağa bağlanılıncaya kadar "." gönderilir

```
Serial.begin(115200);
delay(500);

WiFi.begin(ssid, password); //--> Connect
Serial.println("");

  pinMode(ON_Board_LED,OUTPUT); //--> On
  digitalWrite(ON_Board_LED, HIGH); //--
  pinMode(led, OUTPUT);
  pinMode(14, OUTPUT);
  digitalWrite(14,HIGH);
  pinMode(button, INPUT);


//Bağlantı bekleniyor
Serial.print("Connecting");
while (WiFi.status() != WL_CONNECTED) {
  Serial.print(".");
  digitalWrite(ON_Board_LED, LOW);
  delay(250);
  digitalWrite(ON_Board_LED, HIGH);
  delay(250);
  //----------------------------------
}
```

*Şekil 17:*NodeMCU kod bloğu

NodeMCU wifi bağlantısı gerçekleştikten sonra Firebase ilgili işlemleri gerçekleştirebiliriz.

Arduino IDE de firebase sunucusunu kullanabilmek için indirilmesi gereken bir takım kütüphaneler vardır. Firebase ve arduino json kütüphaneleri lazım bu sefer ise taslak>library ekle>kütüphaneleri yönet yolunu izliyoruz buradan Arduinojson ile arama yaptırıp ilk kütüphanenin 5.7.3 sürümünü kuruyoruz. firebase kütüphanesini https://github.com/FirebaseExtended/firebase-arduino buradaki github linkinden clone or download butonuna basarak .zip şeklinde inirin ve taslak>library ekle>.zip kitaplığı yükle yolunu izleyip .zip ile indirmiş olduğunuz library i kurun.

Bu kütüphaneleri ile indirdikten sonra projemize dahil "#include <FirebaseArduino.h>, #include <ArduinoJson.h>" kütüphanelerini ediyoruz. Veritabanına bağlanabilmek için veritabanı şifremizi ve host adresini tanımlamanız gerekmektedir. Veritabanınızın şifresini FIREBASE_AUTH değişkenine tanımlıyoruz. NodeMCU nun veritabanına erişmesine izin veriyoruz. Koddaki FIREBASE_HOST kısmınıda database imizin host'unu kaydediyoruz.

```
#include <ESP8266WiFi.h>
#include <FirebaseArduino.h>
#include <ArduinoJson.h>
#include <ESP8266HTTPClient.h>

#define FIREBASE_HOST "nodemculed-f73f0.firebaseio.com" //--> URL
#define FIREBASE_AUTH "[redacted]"
```

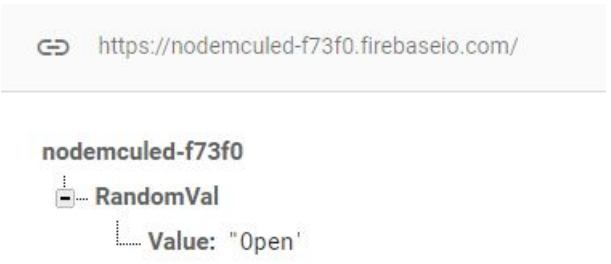
*Şekil 18:*Firebase auth ve host adı

Firebase.begin() komutu ile bağlanılacak veritabanın hostu ve şifresi ile firebase bağlanılıp ardından Firebase başlatılıyor .

```
Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
```

*Şekil 19:*NodeMCU kod bloğu 2

Bu işlemlerde tamamlandığı için artık veritabanımızda ki verilere erişim sağlayıp işlem yapabiliriz.



*Şekil 20:*Firebase veritabanı

Yukarıdaki görselde Firebase realtime database içerisinde verilerin kaydedilme biçimi gösterilmiştir. Veriler her zaman anahtar-değer (key-value) şeklinde tutuluyordu.Value anahtarı içerisinde ki değeri çekmemiz için Firebase kütüphanesinin sağlamış olduğu getString() komutunu kullanacağız.Bu fonksiyon parametre olarak veritabanında çekilecek olan verinin bulunduğu path (yol) değerini almaktadır.Veri çekildikten sonra eğer değer "Close" ise ilk olarak alarm çalması için buzzer aktif hale getiriliyor . Alarm belirli süre (5sn) çaldıktan sonra durduruluyor ve ışık aktif hale yanıp sönmeye başlıyor.

Daha sonra alarm tekrardan aktif oluyor.Bu şekilde veritabanından “Close” değeri geldiği sürece devam ediyor. Open değeri geldiğinde sistem duracaktır. Value değişkeninin değeri iki şekilde “Open” olmaktadır.Bunlardan ilki devreye eklenilen push butona 5 sn basarak alarm ve ışık sistemini durduruyor aynı zamanda Value değişkeni sürücü uyanıp butona bastığı için “Open” değerini almış oluyor. İkincisi mobil uygulamadan uyarı sistemini durdurabilir ve Value değişkeni yine “Open” değerini almış olmaktadır.NodeMcu da butonun takılı olduğu dijital pinden butonun durumu ile ilgili değerimizi okuyoruz ve “temp” değişkenine atılıyor. Eğer “temp=0” yani LOW ise butona basılmamıştır.Butona basılmış ise “temp=1” yani HIGH değerini almaktadır.Butonun durumu “1” olduğunda Firebase kütüphanesinin setString() fonksiyonun kullanarak “Open” değerini veritabanına yazdırılıyor. Böylelikle butona basılma durumunu kontrol edebiliyoruz.

```
temp = digitalRead(button);
if (temp == HIGH) {
      dataSend = "Open";
      Firebase.setString("RandomVal/Value",dataSend);
}

Serial.print("Value : ");
String deger=Firebase.getString("RandomVal/Value"); //--> Ver
Serial.println(deger);
if(deger=="Close" ){
   cikis=0;
       for(int hz = 440; hz < 1000; hz++){
            if(temp==HIGH){cikis=1;break;}
            tone(14, hz, 20);
            delay(5);

        }
        noTone(14);
        digitalWrite(14,HIGH);

      for(int dutyCycle = 0; dutyCycle < 1023; dutyCycle++){
          if(temp==HIGH || cikis==1){cikis=1;break;}
          analogWrite(led, dutyCycle);
          delay(1);
       }

      for(int hz = 1000; hz > 440; hz--){
          if(temp==HIGH || cikis==1){break;}
          tone(14, hz, 20);
          delay(5);
      }
      noTone(14);
      digitalWrite(14,HIGH);
}
```

*Şekil 21:*Buzzer ve şerit led için kodlar

4.2 NodeMCU Devre Bağlantıları

Gerekli Malzemelerimiz:

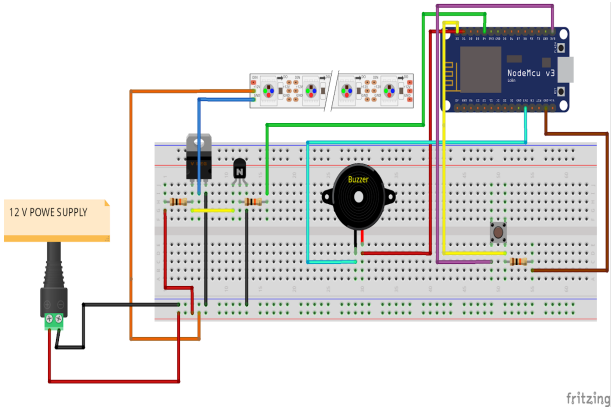
- NodeMCU
- 12 V Şerit led
- BC547 NPN Transistör
- IRFZ44N MOSFET
- 12V 2A adaptör
- 3 adet 10 kΩ direnç
- Buzzer
- 1 adet push-buton
- Jumper kablo

Şerit ledler, büyük çoğunlukla 12V gerilim ile kullanılmaya hazır olarak akım limitleyici dirençler ile birlikte hazırlanırlar. Böylelikle besleme gerilimini 0-12V arasında değiştirerek LED şeritimizin istediğimiz parlaklık seviyesinde yanmasını sağlayabiliriz. Bu işlem için IRFZ44N MOSFET kullanıyoruz. MOSFET’in düzgün şekilde çalışabilmesi için, ayrıca bir NPN tipinde transistöre ihtiyaç duymaktayız. Bunun sebebi, kullanmış olduğumuz MOSFET’in, lojik seviyesi (3.3V veya 5V) sinyaller ile tetiklemeye uygun olmamasıdır. NodeMCU’dan alacağımız PWM çıkışı BC547 NPN transistöre 10kΩ değerinde direnç ile bağlıyoruz ve bu çıkışı güçlendirerek daha büyük yükleri anahtarlayabileceğimiz MOSFET’imizi sürüyoruz.

Buzzer üzerinde iki adet bağlantı pini bulunmaktadır. Uzun olan pin + pini, kısa olan pin ise – pinidir. Buzzer NodeMCU bağlantısı yapılırken + pini NodeMCU dijital pinlerinden birine, – pini ise 3v bağlandı.

Pull-down direnci (10 kΩ), dijital pinleri giriş olarak kullandığımızda sinyalin bozulmamasını sağlar. Bu projemizde buton basılı değilken dijital pinden okunan değer 0V yani lojik LOW seviyesidir. Pull-down direnci, buton basılıp değer HIGH'a çekilmediği sürece bu pindeki gerilimin 0V'ta sabit kalmasını sağlar.Buton NodeMCU üzerinde D0 pinine bağlandı.

Aşağıdaki görselde devredeki tüm bağlantılar gösterilmektedir.



*Şekil 22:*Proje devre şeması

Aşağıdaki görselde uyarı sistemini durdurulabilmesi için yapılan mobil uygulamanın arayüzü gösterilmektedir. Android Studio IDE geliştirilmiştir.Firebase veritabanını android uygulamalarda kullanabilmek içinde bir takım ayarlar gerekmektedir. Firebase ayarlar bölümünden android uygulamayı eklememiz gerekmektedir. Oluşturulan SDK dosyasını android proje dosyası içerisine de ekliyoruz.

Şekil 23: Firebase uygulamalar bölümü

Uygulamanın arayüzünde ses ve ışığı kapatması için bir buton bulunmaktadır.

*Şekil 24:*Mobil uygulama

Yukarıdaki mobil uygulamayı oluşturmak için yazılan kodlar aşağıdadır. Buton tıklandığında click fonksiyonu içerisinde kod tetiklemektedir.

```
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

public class MainActivity extends AppCompatActivity implements View.OnClickListener {
    Button btnkapat;
    FirebaseDatabase database=FirebaseDatabase.getInstance();
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        btnkapat=(Button) findViewById(R.id.btnkapat);
        btnkapat.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                DatabaseReference myref=database.getReference().child("RandomVal").child("Value");
                myref.setValue("Open");
                |
            }
        });
    }
```

*Şekil 25:*Mobil uygulama kodları

5. Kaynaklar

[1]https://randerson112358.medium.com/classify-images-using-convolutional-neural-networks-python-a89cecc8c679

[2]https://medium.com/@yasincakicioglu/android-studio-firebase-kurulum-6b0f857ba952

[3]https://roboindia.com/tutorials/buzzer-nodemcu/

[4]https://machinelearningmastery.com/how-to-load-convert-and-save-images-with-the-keras-api/#:~:text=Keras%20provides%20the%20img_to_array(),data%20into%20a%20PIL%20image.

[5]https://machinelearningmastery.com/how-to-make-classification-and-regression-predictions-for-deep-learning-models-in-keras/

[6]https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-to-classify-photos-of-dogs-and-cats/

[7]https://www.analyticsvidhya.com/blog/2020/02/learn-image-classification-cnn-convolutional-neural-networks-3-datasets/

[8]https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html

[9]https://keras.io/api/models/model_saving_apis/

[10]https://www.analyticsvidhya.com/blog/2020/08/image-augmentation-on-the-fly-using-keras-imagedatagenerator/

[11]https://towardsdatascience.com/understanding-input-and-output-shapes-in-convolution-network-keras-f143923d56ca

[12]https://missinglink.ai/guides/keras/keras-conv2d-working-cnn-2d-convolutions-keras/

[13]https://ayyucekizrak.medium.com/deri%CC%87ne-daha-deri%CC%87ne-evri%C5%9Fimli-sinir-a%C4%9Flar%C4%B1-2813a2c8b2a9

[14]https://keras.io/api/layers/pooling_layers/max_pooling2d/

[15]https://maker.robotistan.com/kizilotesi-kumanda-ile-led-kontrolu/

[16]https://maker.robotistan.com/arduino-ile-buton-ve-led-uygulamasi/

[17]https://www.youtube.com/watch?v=rKuGCQda_Qo

[18]https://medium.com/@yasincakicioglu/android-studio-firebase-kurulum-6b0f857ba952