İZMİR KÂTİP ÇELEBİ ÜNİVERSİTESİ
2010
FEN BİLİMLERİ ENSTİTÜSÜ

Araç Plaka Numarası Tespiti

Yazılım Mühendisliği Ana Bilim Dalı

Dönem Projesi

Alp Timuçin Aksel

ORCID 0009-0002-9125-9415

Proje Danışmanı: Halis Can Koyuncuoğlu

Ağustos 2024

Araç Plaka Numara Tespiti

Öz

Bu proje, araç plakalarının otomatik olarak tanınması ve dijital ortamda işlenmesi için geliştirilen bir görüntü işleme sistemini kapsamaktadır. Geliştirilen sistem, plakaların hızlı ve doğru bir şekilde tespit edilmesini ve tanınmasını sağlayarak trafik yönetimi, güvenlik ve otomasyon alanlarında etkin bir çözüm sunmayı hedeflemektedir. Çalışma, Python programlama dili ile geliştirilmiş olup, herhangi bir hesaplama programı (Matlab vb.) kullanılmadan, yalnızca Python kodlarıyla bir yazılım tasarlanmıştır. Bu yazılım, araç görüntülerinden plakaların yerini bulmayı ve bu plakadaki karakterleri tanıyıp metin bilgisine dönüştürmeyi sağlar.

Anahtar Sözcükler: Araç Plaka Tanıma, Görüntü İşleme, Python, Algoritma, Plaka Tespiti, Karakter Tanıma, Makine Öğrenimi, Yazılım Tasarımı, Plaka Yer Tespiti

Vehicle License Plate Detection

Abstract

This project involves the development of an image processing system for the automatic recognition and digital processing of vehicle license plates. The developed system aims to provide an effective solution in the fields of traffic management, security, and automation by ensuring the rapid and accurate detection and recognition of license plates. The project has been developed using the Python programming language, without relying on any computation software (such as Matlab), and solely with Python code. This software enables the identification of the location of license plates from vehicle images and the recognition and conversion of characters on these plates into text information.

Keywords: Vehicle License Plate Recognition, Image Processing, Python, Algorithm, Plate Detection, Character Recognition, Machine Learning, Software Design, Plate Localization

Teşekkür

Proje çalışmasına katkılarından dolayı Sayın Umut Kaan BAŞER'e teşekkürlerimi sunarım.

İçindekiler

Şekiller Listesi

Bölüm 1

Giriş

Bu proje, araç plakalarının otomatik olarak tanınması ve dijital ortamda işlenmesi için geliştirilen bir görüntü işleme sistemini kapsamaktadır. Geliştirilen sistem, plakaların hızlı ve doğru bir şekilde tespit edilmesini ve tanınmasını sağlayarak trafik yönetimi, güvenlik ve otomasyon alanlarında etkin bir çözüm sunmayı hedeflemektedir.

Gerçekleştirilen bu çalışma ise, şu ana kadar yapılmış olan çalışmalardan farklı olarak herhangi bir hesaplama, görüntü işleme programı (Matlab vb.) kullanılmadan sadece python kodları bir yazılım tasarlanmıştır. Yazılım ile araç görüntülerinden aracın plakasının yerinin bulunması ve bu plakadaki karakterlerin tanınıp metin bilgisine dönüştürülmesi sağlanmaktadır. Bu görüntü işleme algoritmaları tabanlı plaka bulma algoritması yazılımı sayesinde plaka tanıma işlemi hızlı bir şekilde yapılmaktadır.

2010 yılında Afyon Kocatepe Üniversitesi Fen ve Mühendislik Bilimleri Dergisi'nde Kerim Kürşat Çevik ve Abdülkadir Çakır tarafından bir giriş kapısına gelen aracın plakasının görüntü işleme algoritmaları ile tanınarak kapının otomatik olarak açılıp kapanmasını sağlayan bir sistem geliştirilmiştir. Geliştirilen yazılım ile plakaların yerinin bulunmasında %98, plakaların doğru olarak okunmasında ise % 88,1 başarı elde edilmiştir.[1]

Yine 2010 yılında Bilişim Teknolojileri Dergisi'nde Süleyman Demirel Üniversitesi Teknik Eğitim Fakültesi Elektronik-Bilgisayar Eğitimi Bölümü'nden Okan BİNGÖL ve Ömer KUŞCU tarafından uygulama nesne yönelimli bir dil olan C# .NET dili kullanılarak gerçekleştirilmiştir. Çalışmada ilk olarak plaka bölgesi yerinin bulunması gerçekleştirilmiş olup kenar bulma algoritması kullanılmıştır. Daha sonra bulunan plaka bölgesinden karakter ayrıştırma işlemine geçilmiş olup blob coloring algoritması

kullanılmıştır. Son olarak karakterleri ayrıştırılan plakadan karakter tanıma işlemine geçilerek şablon ayrıştırma algoritması kullanılmıştır. Elde edilen sonuçlara göre plaka bölgesinin tespitinde % 95' lik doğruluk oranı, karakter ayrıştırmada %92' lik doğruluk oranı ve karakterlerin okunması işleminde %87' lik doğruluk oranı elde edilmiştir.[2]

Düzce Üniversitesi Bilim ve Teknoloji Dergisinde yayınlanan araştırma makalesinde Talip ÇAY, Emre ÖLMEZ ve Orhan ER tarafından yayınlanan araştırma makalesinde bölgesel Tabanlı Evrişimli Sinir Ağları (R-CNN) ile araç plaka lokasyonu belirleme ve belirlenen lokasyon içerisinden plaka okuma işlemi gerçekleştirilmiştir. İki aşamadan oluşan çalışmanın ilk aşamasında giriş görüntüleri üzerinden plaka lokasyonları R-CNN ile belirlenirken ikinci aşamada geleneksel görüntü işleme teknikleri ile belirlenen lokasyonlardan plaka okuma işlemi gerçekleştirilmektedir. Çalışmada tasarlanan R-CNN eğitiminde veri setinde bulunan 550 adet görüntüden 450 adedi eğitimde ve 100 adedi test işleminde kullanılmıştır. R-CNN ile plaka lokasyonu bulma işleminde test seti üzerinde %95 başarı oranına ulaşılırken doğru olarak belirlenen lokasyonlardan plaka okuma işleminde %97 başarı oranına ulaşılmıştır.[3]

Bölüm 2

Proje Planlaması

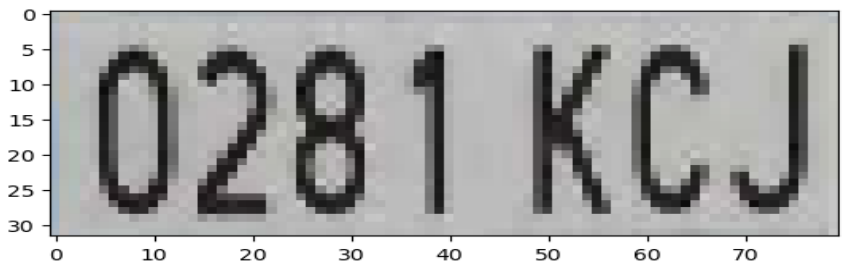
2.1 Gereksinimlerin Belirlenmesi ve Temini

Araç Plaka Numarası Tespiti'nin amacı belirli bir bölgeye giriş yapan araçların plakalarını tespit etmek ve kayıt altına almaktır. Binek araç plakalarını kapsamaktadır.

İki temel altyapıdan oluşuyor. İlk altyapı plakanın konumunu algılayan algoritma, ikinci algoritma plakanın üzerinde bulunan karakterleri sınıflandıran ve analiz eden algoritmadan oluşmaktadır. Projede kullanılacak teknolojiler ve araçlar Python, OpenCV, TensorFlow. Veri seti olarak daha önceden çekilmiş 23 adet araç fotoğrafı kullanılmıştır.

2.2 Veri Seti İnceleme

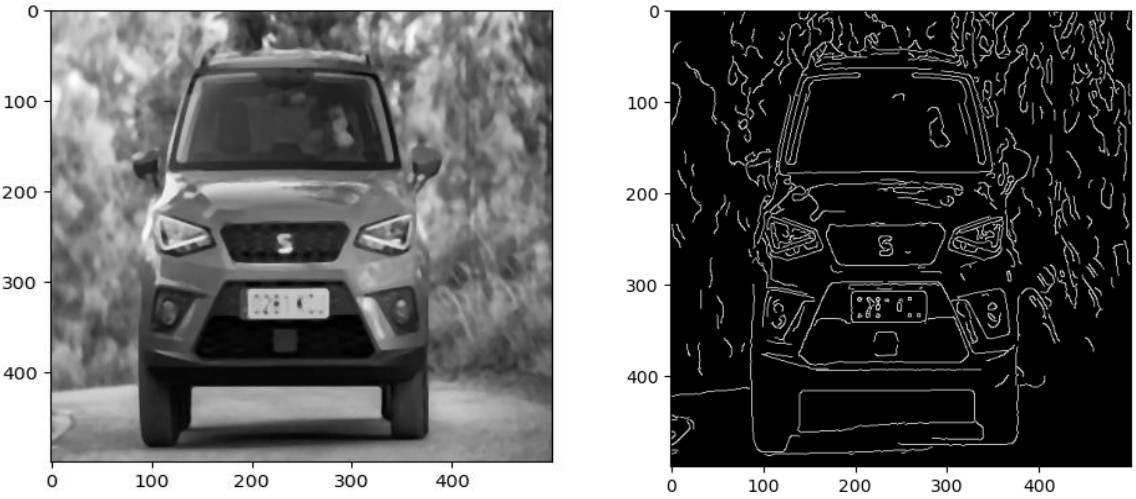
Veri seti incelemek için ilk olarak OS kütüphanesi ile diske bağlanıp diskten fotoğrafların adresleri alınmıştır. Matplotlib kütüphanesiyle fotoğraflara bakarken pixel pixel olarak görüntülememizi sağlamaktadır. Şekil Standart plakalar dikdörtgen yapıdadırlar. Yükseklik genişlik değerlerinde aralarında en az 2 kat fark olduğu görülmüştür. Yükseklik ve genişlik değerlerine baktığımızda Şekil 2.1'de yükseklik 50' nin altında genişlik 150'nin altında olduğu görülmüştür.



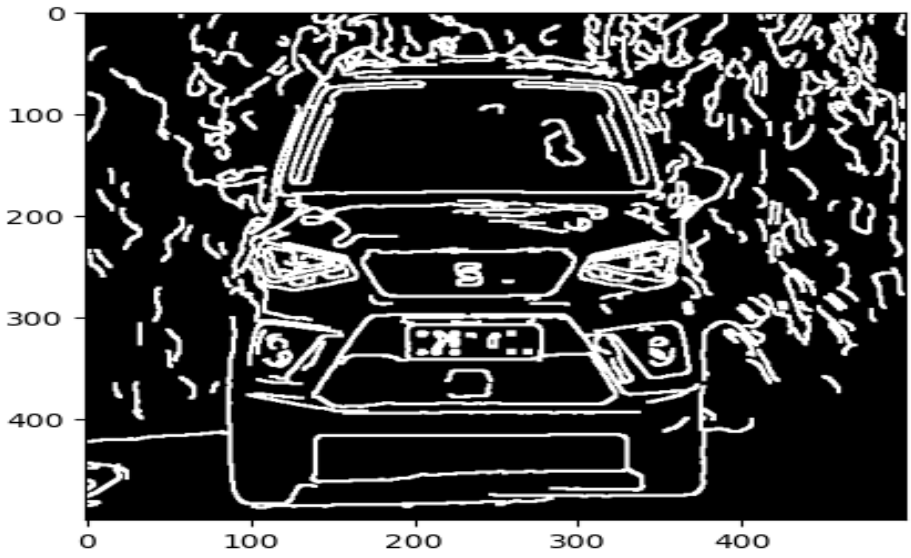
Şekil 2.1: Plaka tespiti

Araç plakalarının arka planı beyaz olmaktadır. Renk RGB yoğunluk değerleri 100 ile 200 arasında olduğu görülmüştür. Bu inceleme işlemleriyle öz nitelik çıkarma işlemi yapılmıştır. Resimler incelenirken RGB ve plakalar siyah beyaz oldukları için gri renk formatında incelenmiştir. Şekil 2.2'de Yeniden boyutlandırılmış, RGB ve gri format görülmektedir. Kenarlıklara baktığımızda siyah çizgiler 5 piksel boyutunda olduğu görülmüştür. Kenarlıkların daha belirgin olması için Şekil 2.3'te görüldüğü üzere Median blurlama yöntemi kullanılmıştır. Bu yöntem, birçok farklı blurlama yöntemi arasında en etkili olanıdır. Gürültü giderme ve kenarlık tespiti ve daha iyi kenarlık tespiti için Şekil 2.4'te genişletme işlemi uygulanmıştır ve kenarlar hep aynı piksel değerlerini almıştır.

Şekil 2.2: Yeniden boyutlandırılmış, RGB ve Gri seviye dönüştürülmüş format



Şekil 2.3: Median blurlama ve Kenarlık tespiti Canny algoritması



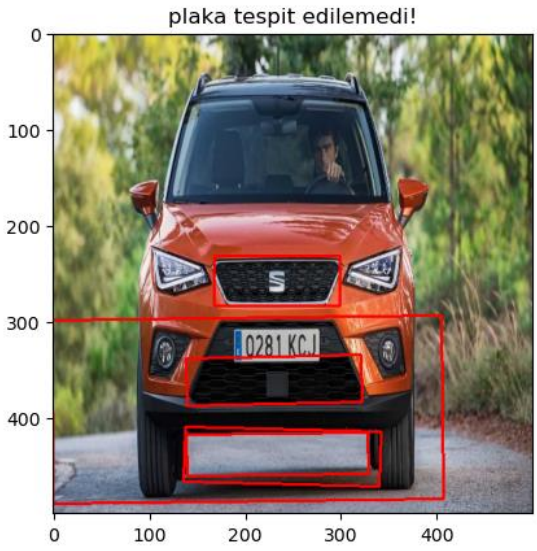
Şekil 2.4: Kenarları genişletme ve kontur bulma

Bölüm 3

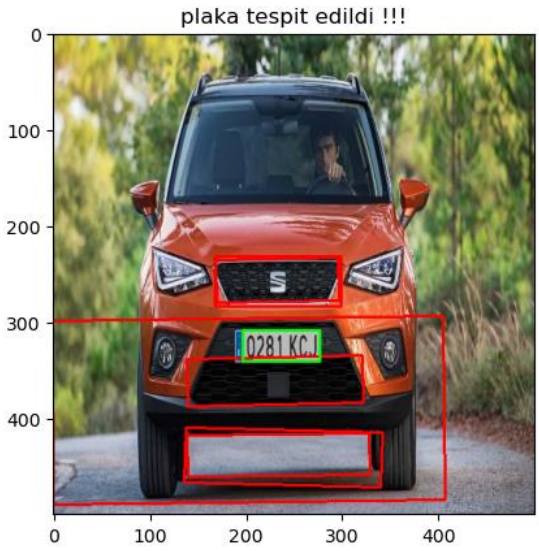
Algoritmalar

3.1 Plaka Tespiti Algoritması

Plaka tanıma ve kordinat bulmada ilk adım dikdörtgen bulma koşulu ile yükseklik ve genişlik oranı 2 koşulu, yoğunluk değerleri 85 ile 200 arasında olması ve boyut sınırlamanın 50 ve 150 arasında olma koşullarını sağlaması gereklidir. Tespit edilen dikdörtgenlerin hangisinin plaka hangisinin plaka olmadığı bu koşulları sağlayıp sağlamadığını tespit edilmesi gereklidir. Koşulları gerçekleştiren ve gerçekleştirmeyenler için plaka tespit edilemedi ! ve plaka tespit edildi !!! ifadesi Şekil 3.1 ve 3.2'de görüldüğü gibi yazdırılmıştır.



Şekil 3.1: plaka tespit edilemedi!



Şekil 3.2: plaka tespit edildi !!!

3.2 Sınıflandırma Algoritması ve Ayrıştırma

3.2.1 Eşikleme (Threshold)

Plaka da rakamlar ve harfleri ilk olarak ayrıştırmamız gerekiyor. Rakamlar siyah olduğu için sadece siyahlıklar ortaya çıkması için eşikleme işlemi gereklidir. Harf ve rakamların genişlik değeri plakanın genişlik değerinin 8'de birinden küçük olmalıdır. Rakamların veya harflerin alanları yaklaşık olarak birbiriyle aynı olduğu görülmüştür.

3.2.2 Karakter Ayrıştırma (Segmentation)

3.2.2.1 Görüntü Dosyasını Okuma ve Yeniden Boyutlandırma

Bu adımda görüntü dosyası okunuyor ve 500x500 piksel boyutunda yeniden boyutlandırılıyor.

3.2.2.2 Plaka Konumunu Belirleme ve Kesme

Bu kısımda plaka_konum_don fonksiyonu ile plakanın konumu belirleniyor.

3.2.2.3 Plaka Görüntüsünü Alma ve Görselleştirme

```
if w > h:
    plaka_bgr = img[y:y+h, x:x+w].copy()
else:
    plaka_bgr = img[y:y+w, x:x+h].copy()
plt.imshow(plaka_bgr)
plt.show()
```

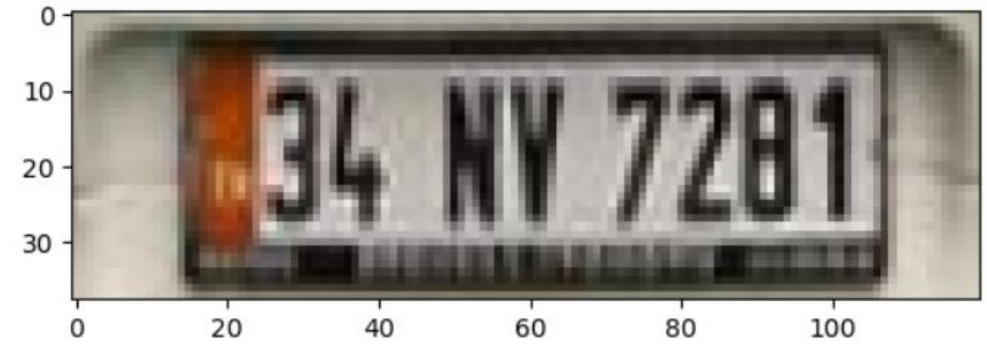
Bu adımda plaka bölgesi kesiliyor ve görüntüleniyor.

3.2.2.4 Plaka Görüntüsünü Yeniden Boyutlandırma

Şekil 3.3'te plaka görüntüsü yeniden boyutlandırılıyor.

```
H, W = plaka_bgr.shape[:2]
```

```
print("orjinal boyut:", W, H)

H, W = H * 2, W * 2

print("yeni boyut:", W, H)

plaka_bgr = cv2.resize(plaka_bgr, (W, H))
```




```
orjinal boyut: 120 38
yeni boyut: 240 76
```

Şekil 3.3: Yeniden boyutlandırma

3.2.2.5 Gri Seviyeye Dönüştürme ve Görüntü İşleme

Şekil 3.4: Gri seviye ve eşikleme işlemi

Şekil 3.4'te plaka gri seviyeye dönüştürülüyor, eşikleme işlemi uygulanıyor ve Şekil 3.5'te morfolojik işlemlerle gürültü gideriliyor.

Şekil 3.5: Gürültü giderme

3.2.2.6 Karakterlerin Bulunması ve Kesilmesi

Şekil 3.6'da görüldüğü gibi bu adımda plaka üzerindeki karakterler bulunup kesiliyor ve kaydediliyor.

cnt=cv2.findContours(th_img,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

```
cnt = cnt[0]
cnt = sorted(cnt, key=cv2.contourArea, reverse=True)[:15]
for i, c in enumerate(cnt):
    rect = cv2.minAreaRect(c)
    (x, y), (w, h), r = rect
    kon1 = max([w, h]) < W / 4
    kon2 = w * h > 200
    if kon1 and kon2:
        print("karakter ->", x, y, w, h)
        box = cv2.boxPoints(rect)
        box = np.int64(box)
        minx = np.min(box[:, 0])
        miny = np.min(box[:, 1])
        maxx = np.max(box[:, 0])
        maxy = np.max(box[:, 1])
        odak = 2
        minx = max(0, minx - odak)
        miny = max(0, miny - odak)
        maxx = min(W, maxx + odak)
        maxy = min(H, maxy + odak)
        kesim = plaka_bgr[miny:maxy, minx:maxx].copy()
```

```
try:
    cv2.imwrite(f"karakterseti/{isim}_{i}.jpg", kesim)
except:
    pass
yaz = plaka_bgr.copy()
cv2.drawContours(yaz, [box], 0, (0, 255, 0), 1)
plt.imshow(yaz)
plt.show()
```

Şekil 3.6: Karakter ayrıştırma

Bölüm 4

Makine Öğrenimi ile Karakter Tanıma

4.1 Gerekli Kütüphanelerin İçe Aktarılması

Bu adımda gerekli kütüphaneler import ediliyor. opencv (cv2), numpy, pandas, pickle, tensorflow ve scikit-learn kütüphaneleri kullanılıyor.

```
import cv2

import numpy as np

import pandas as pd

import pickle

import tensorflow as tf

from sklearn.metrics import accuracy_score

from sklearn.ensemble import RandomForestClassifier

import os
```

4.2 Görüntü Dosyalarının ve Etiketlerinin Yüklenmesi

Bu bölümde, karakterseti klasöründeki tüm resim dosyaları ve bunların sınıfları (etiketleri) bir listeye ekleniyor. Bu bilgiler daha sonra bir DataFrame yapısında saklanıyor.

```
path = "karakterseti/"
```

```
siniflar = os.listdir(path)

tek_batch = 0

urls = []

sinifs = []

for sinif in siniflar:

    resimler = os.listdir(path+sinif)

    for resim in resimler:

        urls.append(path+sinif+"/"+resim)

        sinifs.append(sinif)

        tek_batch+=1

df = pd.DataFrame({"adres":urls,"sinif":sinifs})
```

4.3 Görüntü İşleme Fonksiyonları

```
def islem(img):

    yeni_boy = img.reshape((1600,5,5))

    orts = []

    for parca in yeni_boy:

        ort = np.mean(parca)

        orts.append(ort)

    orts = np.array(orts)

    orts = orts.reshape(1600,)

    return orts
```

```
def on_isle(img):
    return img/255
```

- islem fonksiyonu, her görüntüyü belirli bir şekle yeniden boyutlandırıp, her parçanın ortalamasını alarak bir özellik vektörü oluşturur.
- on_isle fonksiyonu ise görüntüleri normalleştirir.

4.4 Görüntü Verisetinin Hazırlanması

Bu kısımda, ImageDataGenerator kullanılarak görüntüler normalleştirilir ve DataFrame'den alınan veri seti hazırlanır. Daha sonra görüntüler işlenir (islem fonksiyonu ile) ve özellik vektörleri elde edilir.

```
target_size=(200,200)
batch_size=tek_batch
train_gen = tf.keras.preprocessing.image.ImageDataGenerator(
    preprocessing_function=on_isle)
train_set = train_gen.flow_from_dataframe(df, x_col = "adres", y_col="sinif",
target_size=target_size,
color_mode = "grayscale",
shuffle=True,
class_mode='sparse',
batch_size=batch_size)
images, train_y = next(train_set)
train_x = np.array(list(map(islem, images))).astype("float32")
train_y= train_y.astype(int)
```

4.5 Modelin Eğitilmesi

Bu adımda, RandomForestClassifier modeli train_x ve train_y verileri ile eğitilir. Model 10 ağaç kullanarak eğitiliyor.

print("svm egitiliyor")

rfc = RandomForestClassifier(n_estimators=10, criterion="entropy")

rfc.fit(train_x, train_y)

4.6 Modelin Performansının Değerlendirilmesi

Eğitilen model, eğitim verisi üzerinde değerlendirilir ve başarı oranı yazdırılır. Şekil 4.1'de başarı oranı görülmektedir.

pred = rfc.predict(train_x)

acc = accuracy_score(pred, train_y)

print("başarı:", acc)

```
Python 3.12.4 (tags/v3.12.4:8e8a4ba, Jun  6 2024, 19:30:16) [MSC v.1940 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>
= RESTART: C:/Users/alpti/Desktop/İKÇÜ/3.Dönem/Bitirme Proje/plakatanimasistemi-
main/alg2_model_m_o(2).py
Found 176 validated image filenames belonging to 37 classes.
svm egitiliyor
başarı: 1.0
>
= RESTART: C:/Users/alpti/Desktop/İKÇÜ/3.Dönem/Bitirme Proje/plakatanimasistemi-
main/alg2_model_m_o(2).py
Found 176 validated image filenames belonging to 37 classes.
svm egitiliyor
başarı: 0.9943181818181818
>
```

Şekil 4.1: Başarı oranı

Bölüm 5

Model Geri Yükleme ve Başarı Değerlendirmesi

Bu bölümde daha önce eğitilmiş bir RandomForestClassifier modelini kullanarak karakterseti dizinindeki karakter görüntülerini sınıflandırıyor. Kod, model tarafından tahmin edilen karakterleri ve resimlerini görüntüler.

5.1 Kütüphanelerin İçe Aktarılması

Bu adımda gerekli kütüphaneler import ediliyor.

```
import cv2

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import os

import pickle
```

5.2 Görüntü İşleme Fonksiyonu

```
def islem(img):

    yeni_boy = img.reshape((1600,5,5))
```

```
    orts = []
    for parca in yeni_boy:
        ort = np.mean(parca)
        orts.append(ort)
    orts = np.array(orts)
    orts = orts.reshape(1600,)
    return orts
```

islem fonksiyonu, verilen bir görüntüyü alır ve belirli bir Şekilde yeniden boyutlandırarak her parçanın ortalamasını hesaplar. Bu ortalamaları bir diziye dönüştürerek döndürür.

5.3 Modelin Yüklenmesi

```
dosya = "rfc_model.rfc"
rfc = pickle.load(open(dosya, "rb"))
```

Bu bölümde daha önce eğitilmiş model pickle kullanılarak yüklenir.

5.4 Görüntü Dosyalarının ve Etiketlerinin Yüklenmesi

```
path = "karakterseti/"
siniflar = os.listdir(path)
tek_batch = 0
urls = []
sinifs = []
for sinif in siniflar:
```

```
    resimler = os.listdir(path + sinif)
    for resim in resimler:
        urls.append(path + sinif + "/" + resim)
        sinifs.append(sinif)
        tek_batch += 1
df = pd.DataFrame({"adres": urls, "sinif": sinifs})
```

karakterseti dizinindeki tüm resim dosyaları ve bunların sınıfları bir listeye eklenir. Bu bilgiler daha sonra bir DataFrame yapısında saklanır.

5.5 Sınıf Etiketlerinin Tanımlanması

```
sinifs = {'0': 0, '1': 1, '2': 2, '3': 3, '4': 4, '5': 5, '6': 6, '7': 7, '8': 8, '9': 9, 'A': 10,
        'B': 11, 'C': 12, 'D': 13, 'E': 14, 'F': 15, 'G': 16, 'H': 17, 'I': 18, 'J': 19, 'K': 20,
        'L': 21, 'M': 22, 'N': 23, 'O': 24, 'P': 25, 'Q': 26, 'R': 27, 'S': 28, 'T': 29, 'U': 30,
        'V': 31, 'W': 32, 'X': 33, 'Y': 34, 'Z': 35, 'arkaplan': 36}
ayrisma_sinifs = ["arka", "harf", "sayi"]
index = list(sinifs.values())
siniflar = list(sinifs.keys())
```

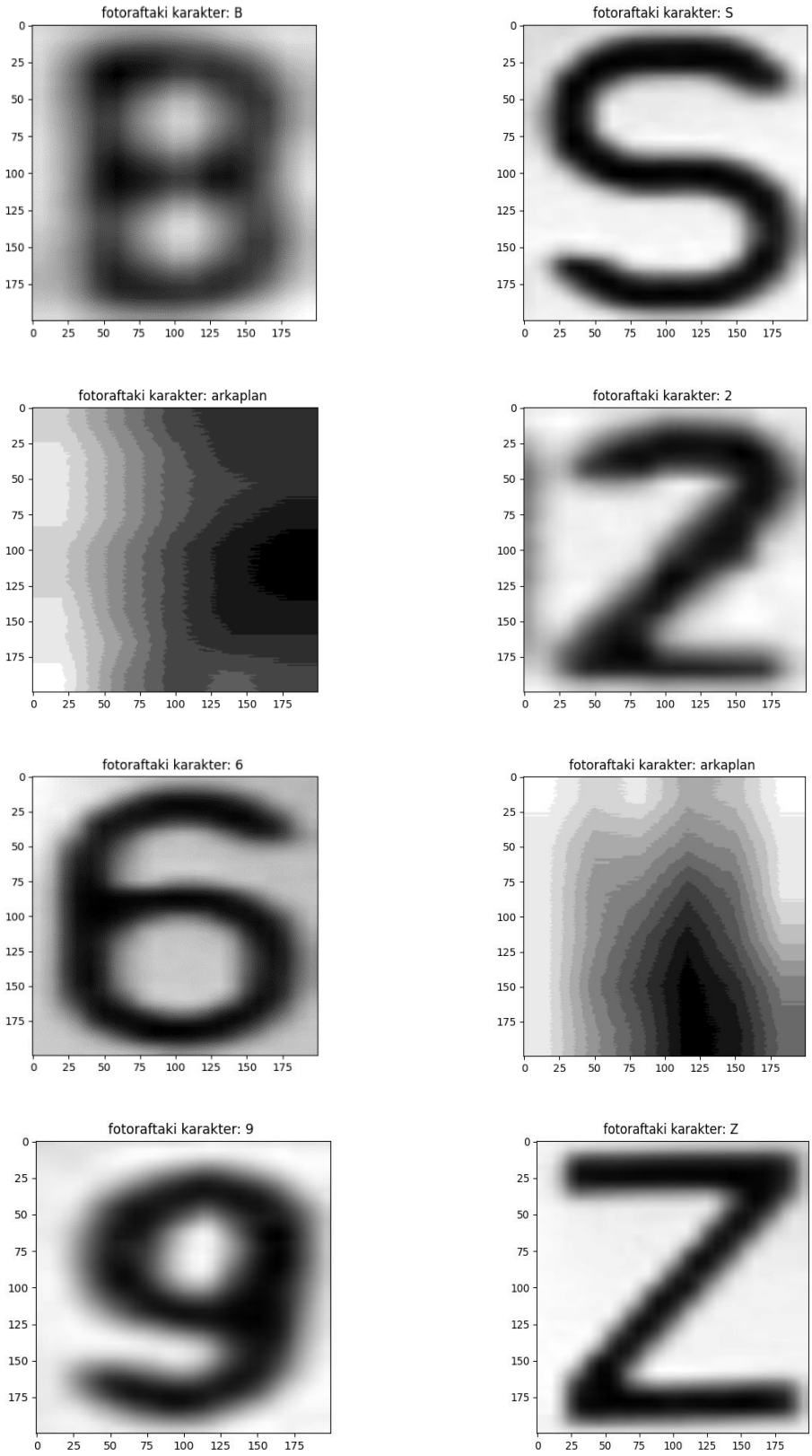
Bu bölümde, karakterlerin sınıfları ve bunların sayısal karşılıkları bir sözlükte tanımlanır.

5.6 Görüntülerin Karıştırılması ve Sınıflandırılması

```
df = df.sample(frac=1)
for adres, sinif in df.values:
```

```
st_adres = adres.split("/")

st0 = st_adres[-2][0].lower()

st1 = st_adres[-1][0].lower()

if(st0 == st1):

    continue

image = cv2.imread(adres, 0)

resim = cv2.resize(image, (200, 200))

resim = resim / 255

oznitelikler = islem(resim)

sonuc = rfc.predict([oznitelikler])[0]

print(sonuc)

ind = index.index(sonuc)

sinif = siniflar[ind]

plt.imshow(resim, cmap="gray")

plt.title(f"fotoraftaki karakter: {sinif} ")

plt.show()
```

Bu bölümde DataFrame karıştırılarak her bir resim tek tek işlenir. Resim 200x200 boyutunda yeniden boyutlandırılır ve normalleştirilir. İşlem fonksiyonu kullanılarak görüntü özellikleri elde edilir ve model tarafından sınıflandırma yapılır. Tahmin edilen sınıf etiketi ve görüntü Şekil 5.1'de ki gibi matplotlib ile gösterilir.



Şekil 5.1: Tahmin edilen sınıf etiketi ve görüntüler

Bölüm 6

Modüler Yapma

Bu bölümde veriseti dizinindeki görüntülerden plaka konumlarını ve karakterlerini tanımlayan bir süreç izler. Kod, belirli bir görüntü üzerinde plaka konumunu bulur, tanımlar ve plakadaki karakterleri çıkararak ekranda gösterir. Bu işlemi her görüntü için tekrarlar.

6.1 Kütüphanelerin İçe Aktarılması

```
import cv2

import numpy as np

import matplotlib.pyplot as plt

import os

from alg1_plaka_tespiti import plaka_konum_don

from alg2_plaka_tanima import plakaTani
```

- cv2: OpenCV kütüphanesi, görüntü işleme fonksiyonları için kullanılır.
- numpy: NumPy kütüphanesi, matematiksel işlemler ve dizi işlemleri için kullanılır.
- matplotlib.pyplot: Görüntüleri grafiksel olarak göstermek için kullanılır.
- os: Dosya ve dizin işlemleri için kullanılır.
- alg1_plaka_tespiti ve alg2_plaka_tanima: Bu modüller, plaka konumlandırma ve plaka tanıma işlemlerini gerçekleştiren fonksiyonları içerir.

6.2 Veri Dizinindeki Görüntülerin Listelenmesi

veriler = os.listdir("veriseti") veriseti dizinindeki tüm dosyalar listelenir ve veriler değişkenine atanır.

6.3 Görüntülerin İşlenmesi ve Plaka Tanımlama

```
for isim in veriler:

    print("resim:", "veriseti/" + isim)

    img = cv2.imread("veriseti/" + isim)

    img = cv2.resize(img, (500, 500))

    plaka = plaka_konum_don(img)

    plakaImg, plakaKarakter = plakaTani(img, plaka)

    print("resimdeki plaka:", plakaKarakter)

    plt.imshow(plakaImg)

    plt.show()
```

- for döngüsü ile veriseti dizinindeki her bir görüntü işlenir.
- cv2.imread ile görüntü okunur ve cv2.resize ile 500x500 piksel boyutlarına yeniden boyutlandırılır.
- plaka_konum_don fonksiyonu, görüntüdeki plakanın konumunu belirler.
- plakaTani fonksiyonu, belirlenen konumdan plaka görüntüsünü çıkarır ve plaka karakterlerini tanımlar.

Sonuçlar ekrana yazdırılır ve matplotlib kullanılarak görüntülenir.

Bölüm 7

Karakter Dizme ve Kontrol Algoritması

Bu kod, görüntülerden plaka tanımlama ve karakter tanıma işlemlerini gerçekleştiren bir plaka tanıma sistemini içerir. Kod, plakanın konumunu tespit eder, karakterlerini çıkarır ve karakterleri sınıflandırarak plakayı belirler.

7.1 Kütüphanelerin İçe Aktarılması ve Modelin Yüklenmesi

```
import cv2

import numpy as np

import pickle

dosya = "rfc_model.rfc"

rfc = pickle.load(open(dosya, "rb"))
```

- cv2: OpenCV kütüphanesi, görüntü işleme fonksiyonları için kullanılır.
- numpy: NumPy kütüphanesi, matematiksel işlemler ve dizi işlemleri için kullanılır.
- pickle: Pickle kütüphanesi, eğitimli modeli dosyadan yüklemek için kullanılır.

Model:

rfc: Daha önce eğitim verilmiş RandomForestClassifier modeli, plaka karakterlerini sınıflandırmak için kullanılır.

7.2 Sınıf Etiketlerinin Tanımlanması

```
sinifs = {'0': 0, '1': 1, '2': 2, '3': 3, '4': 4, '5': 5, '6': 6, '7': 7, '8': 8, '9': 9, 'A': 10,
        'B': 11, 'C': 12, 'D': 13, 'E': 14, 'F': 15, 'G': 16, 'H': 17, 'I': 18, 'J': 19, 'K': 20,
        'L': 21, 'M': 22, 'N': 23, 'O': 24, 'P': 25, 'Q': 26, 'R': 27, 'S': 28, 'T': 29, 'U': 30,
        'V': 31, 'W': 32, 'X': 33, 'Y': 34, 'Z': 35, 'arkaplan': 36}
index = list(sinifs.values())
siniflar = list(sinifs.keys())
```

Plaka karakterlerinin sınıf etiketleri tanımlanmıştır.” sinifs” sözlüğü her karaktere bir sınıf numarası atar.

7.3 Görüntü İşleme ve Özellik Çıkartma Fonksiyonu

```
def islem(img):
    yeni_boy = img.reshape((1600, 5, 5))
    orts = []
    for parca in yeni_boy:
        ort = np.mean(parca)
        orts.append(ort)
    orts = np.array(orts)
    orts = orts.reshape(1600,)
    return orts
```

“islem” fonksiyonu, görüntüyü yeniden boyutlandırır ve her bir parçanın ortalama yoğunluğunu hesaplar. Bu ortalamalar özellik vektörleri olarak kullanılır.

7.4. Plaka Ayrıştırma Fonksiyonu

```
def plakaAyristir(mevcutPlaka):
    mevcutPlaka = sorted(mevcutPlaka, key=lambda x: x[1])
    mevcutPlaka = np.array(mevcutPlaka)
    mevcutPlaka = mevcutPlaka[:, 0]
    mevcutPlaka = mevcutPlaka.tolist()
    karakterAdim = 0
    for i in range(len(mevcutPlaka)):
        try:
            int(mevcutPlaka[i])
            karakterAdim += 1
        except:
            if karakterAdim > 0:
                if i - 2 >= 0:
                    mevcutPlaka = mevcutPlaka[i - 2:]
                break
            mevcutPlaka.pop(i)
    karakterAdim = 0
    for i in range(len(mevcutPlaka)):
```

```
        kontrolIndex = -1 + (-1 * karakterAdim)
        try:
            int(mevcutPlaka[kontrolIndex])
            karakterAdim += 1
        except:
            if karakterAdim > 0:
                karIndex = len(mevcutPlaka) - karakterAdim
                mevcutPlaka = mevcutPlaka[:karIndex + 4]
                break
            mevcutPlaka.pop(kontrolIndex)
    return mevcutPlaka
```

plakaAyristir fonksiyonu, plaka karakterlerini doğru sırayla ayrıştırır. Plakanın başında en fazla 2 rakam ve sonunda en fazla 4 rakam olmasını sağlar.

7.5 Plaka Tanıma Fonksiyonu

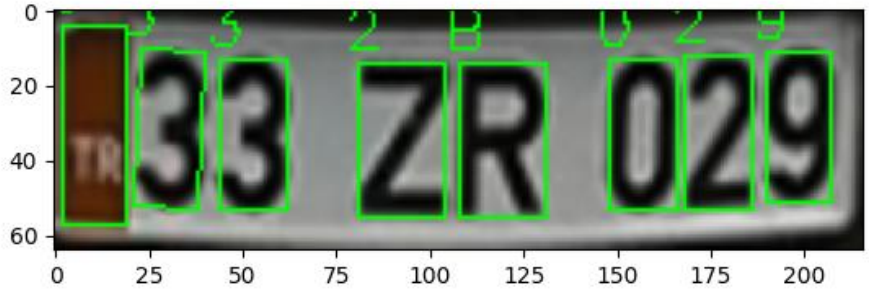
```
def plakaTani(img, plaka):
    global index, siniflar
    x, y, w, h = plaka
    if (w > h):
        plaka_bgr = img[y:y + h, x:x + w].copy()
```

```
else:

    plaka_bgr = img[y:y + w, x:x + h].copy()

H, W = plaka_bgr.shape[:2]

H, W = H * 2, W * 2

plaka_bgr = cv2.resize(plaka_bgr, (W, H))

plaka_resim = cv2.cvtColor(plaka_bgr, cv2.COLOR_BGR2GRAY)

th_img = cv2.adaptiveThreshold(plaka_resim, 255,
    cv2.ADAPTIVE_THRESH_MEAN_C, cv2.THRESH_BINARY_INV, 11, 2)

kernel = np.ones((3, 3), np.uint8)

th_img = cv2.morphologyEx(th_img, cv2.MORPH_OPEN, kernel, iterations=1)

cnt = cv2.findContours(th_img, cv2.RETR_EXTERNAL,
    cv2.CHAIN_APPROX_SIMPLE)

cnt = cnt[0]

cnt = sorted(cnt, key=cv2.contourArea, reverse=True)[:15]

yaz = plaka_bgr.copy()

mevcutPlaka = []

for i, c in enumerate(cnt):

    rect = cv2.minAreaRect(c)
```

```
(x, y), (w, h), r = rect

kon1 = max([w, h]) < W / 4

kon2 = w * h > 200

if (kon1 and kon2):

    box = cv2.boxPoints(rect)

    box = np.int64(box)

    minx = np.min(box[:, 0])

    miny = np.min(box[:, 1])

    maxx = np.max(box[:, 0])

    maxy = np.max(box[:, 1])

    odak = 2

    minx = max(0, minx - odak)

    miny = max(0, miny - odak)

    maxx = min(W, maxx + odak)

    maxy = min(H, maxy + odak)

    kesim = plaka_bgr[miny:maxy, minx:maxx].copy()

    tani = cv2.cvtColor(kesim, cv2.COLOR_BGR2GRAY)
```

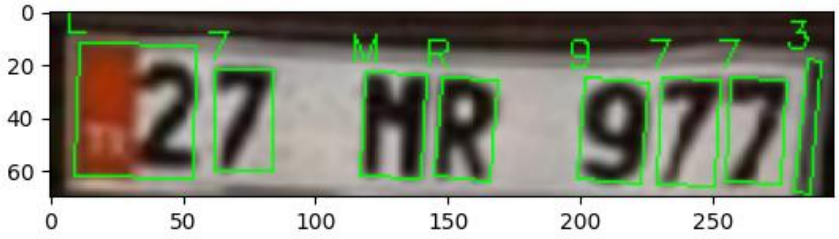
```
            tani = cv2.resize(tani, (200, 200))

            tani = tani / 255

            oznitelikler = islem(tani)

            karakter = rfc.predict([oznitelikler])[0]

            ind = index.index(karakter)

            sinif = siniflar[ind]

            if sinif == "arkaplan":

                continue

            cv2.putText(yaz, sinif, (minx - 2, miny - 2),
            cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 1)

            mevcutPlaka.append([sinif, minx])

            cv2.drawContours(yaz, [box], 0, (0, 255, 0), 1)

    if len(mevcutPlaka) > 0:

        mevcutPlaka = plakaAyristir(mevcutPlaka)

    return yaz, mevcutPlaka
```

plakaTani fonksiyonu, plakanın konumunu ve plaka karakterlerini tanımlar. Plakadan karakterleri çıkarır, her karakteri sınıflandırır ve plaka görüntüsü üzerinde Şekil 7.1, 7.2 ve 7.3'te görüldüğü gibi karakterleri işaretler.



Şekil 7.1: Plaka görüntüsü üzeri karakterler

Şekil 7.2: Plaka görüntüsü üzeri karakterler 2



Şekil 7.3: Plaka görüntüsü üzeri karakterler 3

7.6 Kullanım Adımları

`rfc_model.rfc` dosyasından daha önce eğitim verilmiş model yüklenir. `plakaTani` fonksiyonu, verilen görüntüdeki plaka konumunu ve karakterlerini tanımlar. `plakaAyristir` fonksiyonu, tanımlanan karakterleri sıralar ve plaka formatına uygun hale getirir.

7.7 İyileştirme ve Geliştirme

Görüntü İşleme Parametreleri: cv2.adaptiveThreshold ve cv2.morphologyEx gibi görüntü işleme parametreleri üzerinde deney yaparak daha iyi sonuçlar elde edilebilir.

Plaka tespiti ve karakter tanıma algoritmalarında daha fazla hassasiyet sağlamak için yeni modeller (Evrişimsel Katman gibi derin öğrenme tabanlı yöntemler) entegre edilebilir.

Optik Karakter Tanıma (OCR) algoritmalarını daha doğru ve verimli hale getirmek için makine öğrenimi ile eğitim verisi artırılabilir. Farklı plaka tipleri, yazı tipleri ve boyutlarına göre özelleştirilmiş karakter tanıma yöntemleri eklenebilir.

Sistem, gerçek zamanlı video analizini destekleyecek şekilde optimize edilebilir. Kamera ile doğrudan entegrasyon sağlanarak anlık plaka tanıma işlemleri yapılabilir.

Daha çeşitli veri setleri (farklı ülkelerden plaka görüntüleri, farklı ışık koşulları vb.) kullanılarak model yeniden eğitilebilir ve genel başarı oranı artırılabilir.

Araç görüntülerinin işlenmesi sırasında gizlilik önlemleri alınabilir. Verilerin şifrelenmesi ve güvenli bir şekilde saklanması gibi ek güvenlik özellikleri eklenebilir.

Bu öneriler, sistemin daha verimli, hızlı ve güvenilir çalışmasını sağlayarak geniş bir kullanım alanı için daha uygun hale getirilmesine katkıda bulunacaktır.

Bölüm 8

Kaynaklar

[1] Çevik, K. K., & Çakı, A. (2010). Görüntü İşleme Yöntemleriyle Araç Plakalarının Tanınarak Kapı Kontrolünün Gerçekleştirilmesi. Afyon Kocatepe Üniversitesi Fen Ve Mühendislik Bilimleri Dergisi, 10(1), 31-38.

[2] Bingöl, O., & Kuşcu, Ö. (2010). Bilgisayar Tabanlı Araç Plaka Tanıma Sistemi. Bilişim Teknolojileri Dergisi, 1(3).

[3] Çay, T., Ölmez, E., & Er, O. (2023). Bölgesel Tabanlı Evrişimli Sinir Ağı ile Araç Plaka Tanıma. Düzce Üniversitesi Bilim Ve Teknoloji Dergisi, 11(1), 10-20. https://doi.org/10.29130/dubited.1058850