METAL DAMGALAMA GÖRÜNTÜLERİ ÜZERİNDEN KARAKTER TANIMLAMASI

Ali Atilla Aydemir 180202063

Melih Yeşilyurt 180202060

Erdem Nayın 180202050

Barış Arslan 180202112

Problem

- Günümüzde birçok fabrikada ürün numaraları metal yüzey üzerine yazılmaktadır. Bu firmalara örnek olarak Aygaz ve birçok otomotiv firması tarafından kullanılmaktadır.
- Bu projede çözülmesi hedeflenen sorun ise yazı metni ile arka plan renginin aynı olmasından dolayı OCR modellerinin yazıyı tespit etmemesi.

Karşılaşılan Sorunlar

- Metin Rengi ve Arka Planın Aynı Renk Olması
- Veri Seti Yetersizliği
- Tecrübe Yetersizliği
- Test Edilmiş filtrelerin Kodlarının Bulunmaması
- Benzer Projelerin Bulunmaması

Denediğimiz Başarısız Çalışmalar

- Tesseract Kütüphanesi Kullanarak Fotoğraftaki Yazı Karakterlerinin Tespiti
- EasyOCR ve Torch Kütüphanesi Kullanılarak Fotoğraftaki Yazı Karakterlerinin Tespiti
- OpenCV ve Tesseract Kütüphanesi Kullanılarak Fotoğraftaki Yazı Karakterlerinin Tespiti
- Tesseract Kütüphanesi Kullanılarak Metal Yüzeyindeki Kabartmalı Yazı Karakterlerinin Okunması
- Knn kullanılarak harflerin ve sayıların tek tek tespit edilip analizlerinin yapılması.

BULDUĞUMUZ ÇÖZÜMLER

Tesseract OCR Kullanılarak

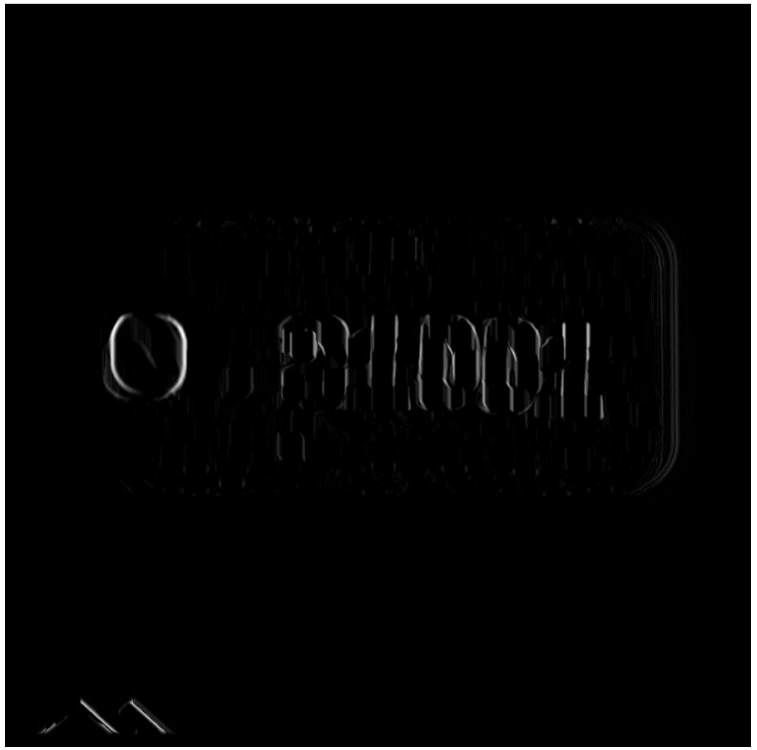
Yazı Tespitinin Gerçekleştirilmesi

- Bu çözümü yaparken daha önceki denemelerimizden faydalandık. İlk önce OCR teknolojisini araştırdık ve nasıl gerçekleştirildiğini öğrendik. OCR kütüphanelerini incelediğimiz zaman bize uygun olanın Tesseract olduğuna karar verdik. Veri setimiz kalitesiz olduğundan dolayı internette kaliteli fotoğraflar aradık ve sadece 3 tane kaliteli ve uygun olan fotoğraf bulabildik.
- Öncelikle Çözümümüzde fotoğraflardaki yazının konumunu tespit etmek gerekiyordu. Bu yüzden onunla ilgili detectTextForCrop() methodu hazırlandı. Kullandığımız fotoğrafın orijinal hali yandaki şekildeki gibidir.

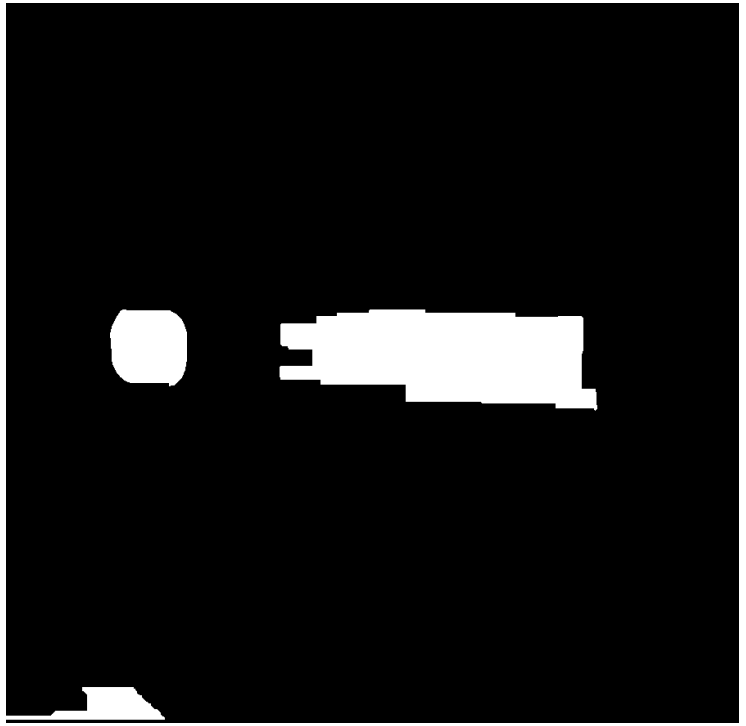
- detectTextForCrop() methodu öncelikle fotoğrafın boyutunu hesaplıyor eğer genişlik 180'den büyük ve yükseklik 60'tan büyük ise kırpma işlemi yapıyor. Kırpma işlemi yaparken ilk önce genişliği 90'a ayarlıyor. Daha sonra renk paletini gri hale getiriyor ve Gauss bulanıklaştırma yöntemi uygulanıyor.

- Dilate operatörüne blurlanmış son görüntümüzü veriyoruz. Bu operatör giriş olarak verilen görüntü üzerinde parametreler ile verilen alan içerisindeki sınırları genişletmektedir. Bu genişletme sayesinde piksel grupları büyür ve pikseller arası boşluklar küçülür.
- Numpy yardımıyla manuel yapılandırma elemanı oluşturmaya yarar çekirdeğin şeklini ve boyutunu giriyoruz ve matris içi tamamen 1 dolduruyor.

- MorphologyEx fonksiyonu yardımıyla giriş görüntüsü ve görüntünün açıldıktan sonraki halini elde ediyoruz. Sobel filtresi yardımıyla kenarları tespit ediyoruz. 1. sırada fotoğraf, 2. sırada image matrisindeki dışta kalan piksellerin nasıl üstesinden gelineceği işlemi yapılıyor. Sobel fonksiyonuna verdiğimiz parametreler sayesinde dikey yönde kenarları tespit ediyoruz.



- Daha sonra MorphologyEx ve Threshold fonksiyonları yardımıyla dilation ile genişletilen fotoğrafta, yazının olabileceği yatay bölgelerin belirtildiği fotoğraf elde ediliyor ve yazdırılıyor. Burada kullandığımız FindContours yöntemi sebebiyle fotoğraf kullanılamaz hale geliyor bu yüzden fotoğrafımızı yedekliyoruz.



Kırpılmış Hali

- Konturlar aynı renk ve yoğunluğa sahip olan tüm kesintisiz noktaları sınır boyunca birleştiren bir eğri olarak basit bir şekilde açıklanabilir. Çeşitli fonksiyonlar sayesinde en büyük kontur alanını tespit ediyoruz.
- Çünkü kontur alanının en büyük olduğu yerlerde yazı olma ihtimali en yüksek oluyor. Bu alanın sınırları tespit edilir ve fotoğraf bu sınırlara göre kesilir, kaydedilir ve yazdırılır. En son yandaki fotoğrafı elde ediyoruz.

Bulanıklaşmış ve Renk paleti değişmiş hali

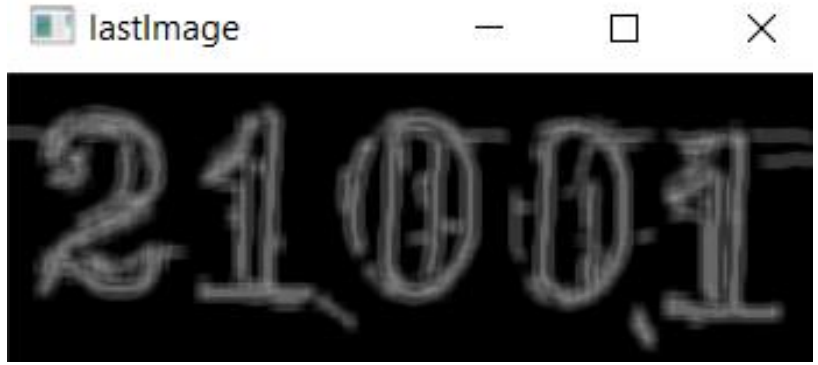
- Kırpma işlemleri bittikten sonra yazının tespit edilmesi için detectLettersFromImage() isimli bir method hazırladık.
- Method içerisinde ilk önce fotoğraftaki gürültüyü gidermek için cv2.fastNlMeanDenoisingColored() kullanarak fotoğrafı bulanıklaştırıyoruz. Daha sonra renk paletini gri yapıyoruz.

Canny uygulanmış hali

- Daha sonra kenarlarını belirginleştirmek cv2.Canny uyguluyoruz. Canny'in ilk threshold değerini 40 ikincisini ise 20 olarak ayarlıyoruz.

Bulanıklaştırma uygulanmış hali

- Daha sonra oluşan görüntüdeki gürültülerin azaltılması için tekrar bulurlama işlemi uyguluyoruz ama bu bulurlama işlemini daha farklı şekilde yapıyoruz. Kendi ayarladığımız bir kernel matrisi kullanarak bulanıklaştırıyoruz.



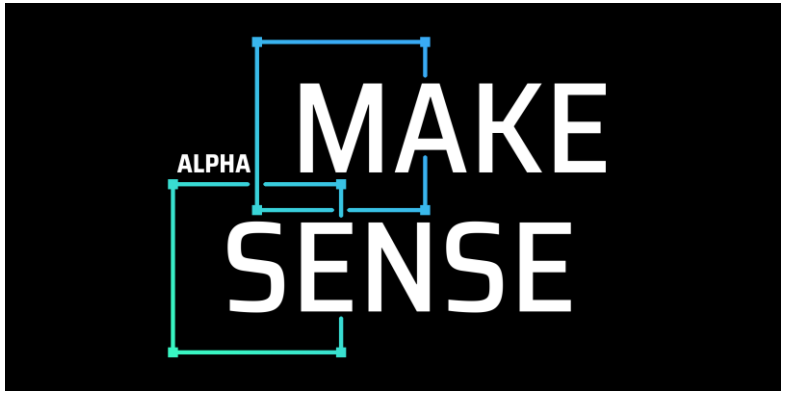
- Fotoğrafın son hali her ne kadar göze hoş gelmese de fotoğraf artık Tesseract tarafından tespit edilebilir hale gelmiştir.
- Tesseract'ın tespit edebilmesi için image_to_string() fonksiyonunu çalıştırmak yeterlidir. Son olarak çıkan yazıyı konsola çıktı olarak vermekteyiz.

BULDUĞUMUZ ÇÖZÜMLER

Yolov4 Kullanılarak

Yazı Tespitinin Sağlanması

- Yolov4'ü seçme sebebimiz rakiplerine kıyasla en başarılı olan olması. Makesense.ai adlı internet sitesini kullanarak yolo4 formatında fotoğraf etiketledik. Test için 31 fotoğraf kullandık. Veri setimizde 300 adet fotoğraf vardı, bu fotoğrafları veri madenciliği dersinde öğrendiğimiz bilgilere göre temizledik.
- Eğitim için ise 125 fotoğraf kullandık. Toplamda model için 156 fotoğraf kullandık yaklaşık 25 fotoğrafta kendimizin modeli test etmesi için ayırdık. Modeli eğitmek için Google colab kullandık. En çok yaşadığımız problem Google colab kullanırken işlemci kullanma hakkımızın bitmesiydi. Bunun birden fazla hesap açtık.

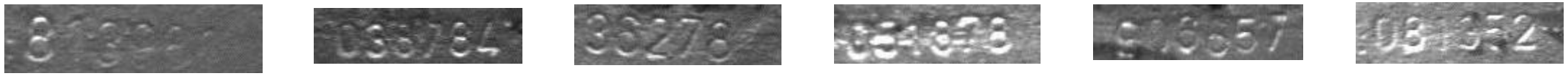


- Kodlarımız Google Colab içerisinde çalıştırdık yani bulut ortamında çalıştık. Veri setimiz çok kaliteli ve fazla olmadığından dolayı pek başarılı olamadık.
- Ama sonuç olarak bu problemin çözümünün bu şekilde sağlanabileceğini kanıtlamış olduk.

Dateset’te bulunan örnek fotoğraflar

- Dataset’teki fotoğraflar sadece rakamları içermektedir.
- Çözünürleri çok düşüktür.
- Çoğu insan gözüyle okunamamaktadır.

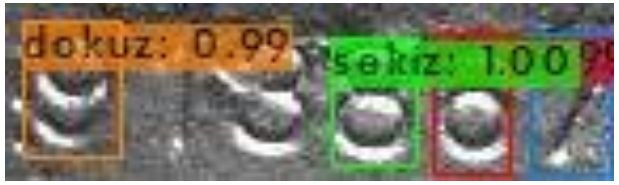
813981 038784 36278 051878 906657 081352



Modelin tespit ettiği fotoğraflar

699668

913867

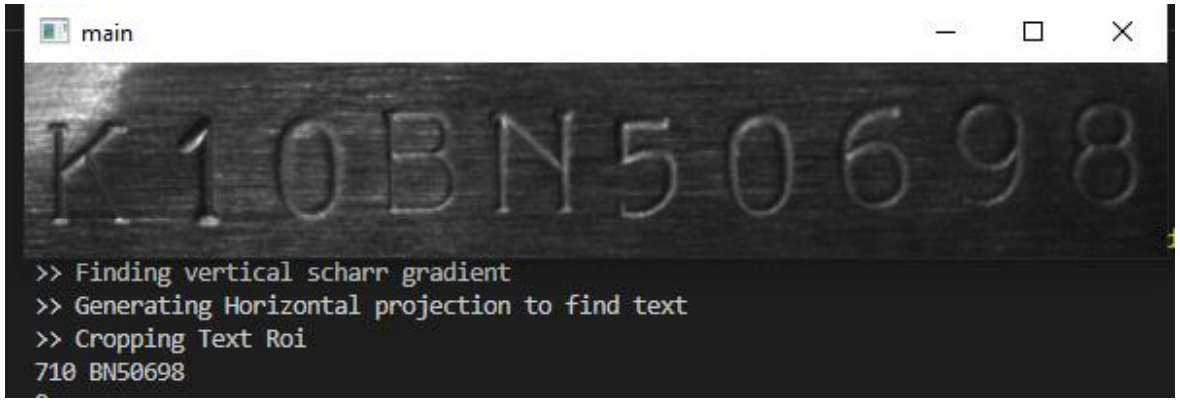


478378

139708

Deneysel Sonuçlar

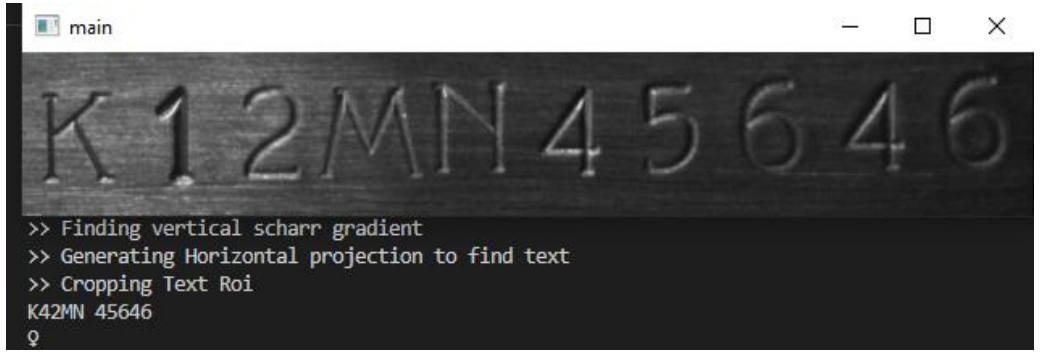
Orijinal Fotoğraf



Elde ettiğimiz Sonuç

Deneysel Sonuçlar

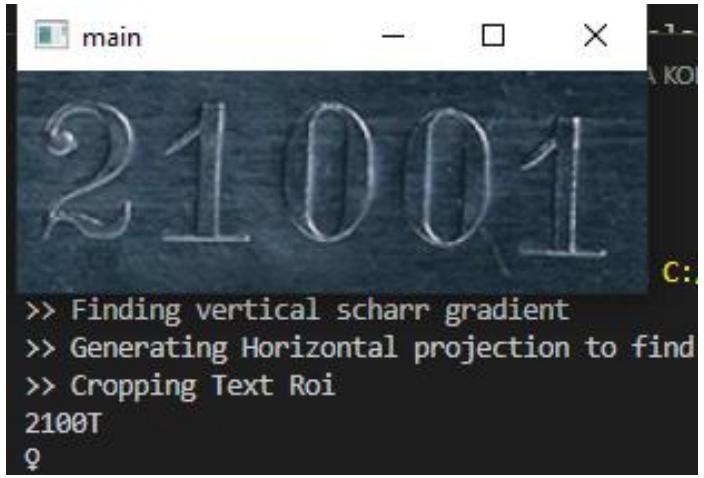
Orijinal Fotoğraf



Elde ettiğimiz Sonuç

Deneysel Sonuçlar

Orijinal Fotoğraf



Elde ettiğimiz Sonuç