Derin Öğrenme İle Denizdeki Atık Maddelerin Tespiti

Umut Özkan
İSTANBUL TOPKAPI ÜNİVERSİTESİ
İstanbul, Türkiye
umuttozkannis@gmail.com
22040301039

Efe İngin
İSTANBUL TOPKAPI ÜNİVERSİTESİ
İstanbul, Türkiye
inginefe44@gmail.com
21040301056

Enes Çakır
İSTANBUL TOPKAPI ÜNİVERSİTESİ
İstanbul, Türkiye
enescakir121@gmail.com
22040301053

I. GİRİŞ

Projenin amacı, denizlerdeki atık maddeleri tespit etmek ve denizlerin daha temiz olmasını sağlamaktır. Bununla beraber, insan yaşamının daha kaliteli hale gelmesi amaçlanmıştır. Projede derin öğrenme modellerinden birisi olan YOLOV8 kullanılmıştır. YOLOV8 hakkında makale içerisinde bilgilendirme yapılmıştır. Makalenin son kısmında, proje üzerinde çalışan kişilerin CV'leri verilmiştir.

A. YOLO

YOLO açılımı 'You only look once' dır. Yani Türkçe karşılığı ile 'Yalnızca Bir Kez Bak' anlamına gelmektedir. Adından ve açılımından da anlaşıldığı gibi verdiğimiz ya da gördüğü anlık bir görseli tek bir seferde işleyerek içerisinde görünen nesneleri verdiğimiz etiketler ile algılar ve bunları sınıflandırır . Yolodan farklı bir çok modelde resmi anlamlı parçaları ayırma yerine resmin bütününe hitap ederek bir kerede analiz eder. Bu özelliği YOLO'yu diğer farklı eş değer algoritmalarından ayırarak çok daha hızlı hale getirir ve aynı şekilde hızlı sonuç elde edilmesini sağlar. Nesne tespiti için oldukça yaygın kullanılan derim öğrenme algoritmalarından biridir.

B. YOLO Nerelerde Kullanılır?

1) Güvenlik Sistemleri: Çağımızda nüfüs çok fazla arttığından güvenlik birimleri adına denetim ve sorgu ciddi anlamda fazla yük düşmekte. Bununla birlikte artık neredeyse her sosyal alan dahil, sokaklar, mağazalar, AVM'ler , hatta kişilerin artık evlerinde dahil güvenlik kameraları ile gözlenmekte. Bu eğitilen modeller ile hareketli nesneler alıgılanabilir hale geldi. Herhangi bir tehdit oluşturabilecek anlar için derin öğrenme modelleri kullanılabilir ve kullanılmaktadır.

2) Otonom Araçlar: Günümüzde artık elektirikli ve otonom araçların sayısı hızla artmakta. Otonom araçların popülerliğinin en büyük nedenlerinden biri sunduğu kolaylıklar. Bunda ise yine en çok payı yapay zeka öğrenimi. Yollarda giderken nesnelerin , trafik ışıkları ve levhaların, yolda yürüyen yayaların, trafikteki araçların tespitinde kullanılmakta.

3) Tıp: Artık günümüz çağında tıp alanı da çok ileri bir seviyeye ulaştı. Hastanelerde ve tedavi yöntemleri için kullanılan cihazlarda kişinin anormalliklerini, sağlıklı bireye göre hastalık geçiren bireylerin arasındaki farkını tespit ederek çok daha önceden önlem alınmasına olanak sağlamaktadır. Bu sayede canlıların can kaybına neden olabilcek tehlikeli hastlaıkları kısa sürede önceden tespit etmek sağlık alanındaki en büyük faydalarından biri haline geldi.

II. OPTİMİZASYON ALGORİTMALARI

A. SGD (Stochastic Gradient Descent)

SGD temel bir optimizasyon algoritmasıdır. Makine öğreniminde ve derin öğrenme modellerinin eğitilmesinde kullanılır. Kayıp fonksiyonunun türevini alır. Bu şekilde parametreler güncellenir. Parametreler, her adımda kayıp fonksiyonunun azaldığı yönde adım atar. SGD, parametrelerini yineleyerek ayarlar ve modelin maliyetini düşürür. Bu, SGD algoritmasının asıl amacıdır. Aynı zamanda SGD algoritması, kayıp fonksiyonunu minimize etmeyi amaçlar.

SGD, tüm veri seti yerine her iterasyonda bir örnek kullanır. Bu şekilde daha az bellek kullanılır ve hesaplama hızlanır. Aynı zamanda SGD, karmaşık hesaplamalara ihtiyaç duymadığından dolayı uygulanması kolay olan bir algoritmadır.

B. ADAM

Makine öğrenimi ve derin öğrenme modellerinin eğitimi için kullanılan bir optimizasyon algoritmasıdır. ADAM, gradyan iniş yönteminin daha gelişmiş bir versiyonudur. Momentum ve öğrenme hızı özelliklerini birleştirir. Bu şekilde daha hızlı ve kararlı bir optimizasyon sağlar. ADAM, her bir parametre için gradyanların hem birinci momentini hem de ikinci momentini hesaplar. Birinci moment gradyanların ortalaması, İkinci moment ise gradyanların karelerinin ortalamasıdır. Bu bilgiler, öğrenme hızının otomatik olarak ayarlanmasını sağlar.

C. ADAM Ve SGD Arasındaki Farklar

SGD basit, hafif ve genelleme açısından etkilidir. Tüm parametreler için sabit öğrenme hızını kullanır. Her adımda kayıp fonksiyonunun gradyanına dayanarak parametreleri günceller. Minibatch kullanarak daha hızlı hale getirir. ADAM ise hızlı, kararlı ve adaptiftir, fakat bellek maliyeti yüksektir. Güncellemede sadece o anki minibatch'den elde

edilen gradyanı kullanır. Güncellemeler zigzag şeklinde olabilir. Ayrıca ADAM, öğrenme hızını adaptif olarak ayarlar, yani öğrenme hızı sabir değildir.
Sonuç olarak SGD, küçük veri setleri ve modellerde daha yaygın kullanılır. ADAM ise büyük ve karmaşık veri setleri ve modellerde daha yaygındır.

III. VERİ SETİ VE İŞLEMLERİ

A. *Veri Seti Seçimi Ve Toparlanması*

4) Karşılaşılan Zorluklar : Veri Setini araştırırması yapılırken deniz üstü çalışma yapıldığı için çok fazla düzgün veri bulunamadı çalışmalarımızı önceliği yeni bir veri seti oluşturmaya odakladık YOLO yaml formatı veri setini oluştururken önemli bir yere sahiptir hangi sınıflar olucağı kaç adet sınıf olucağı vs herşey oluşturuldu ve ortaya toplamda başta 11 ama daha sonra tavsiye sonrası 10 adet sayıya düşürdüğümüz yaml dosyası ortaya çıktı bu adetleri temsil eden sınıflar sırayla; Plastik Şişe, Naylon Poşet, Metal kutular, Paketli Ambalajlar, Balık Ağları, Cam Şişe, Sahil Topları, Maske, Eldiven Ve Sigara İzmariti olmak üzere oluşturuldu.

5) Veri Setinin Birleştirilmesi : Yaklaşık 15 veri seti birleştirmeye karar verildi ortalama 30k gibi büyük ve güzel veri seti oluşturucaktı fakat büyük sorunlardan biriside etiket farklılığı bize yeni bir algoritma oluşturmaya itti ve label etiketlerine bakarak YOLO'da atılan etiketler başlangıçta etiket kodu içeriyordu bunu kendi etiketimize uyarlamalıydı ve şu tarz bir yönteme gidildi örneğin birleştirilcek veri setinde plastik şişe 3 etikete sahipse bunu 0 ile değiştirip kaydet yoluna gidildi ve başarıyla dönüşüm işlemi yapıldı.

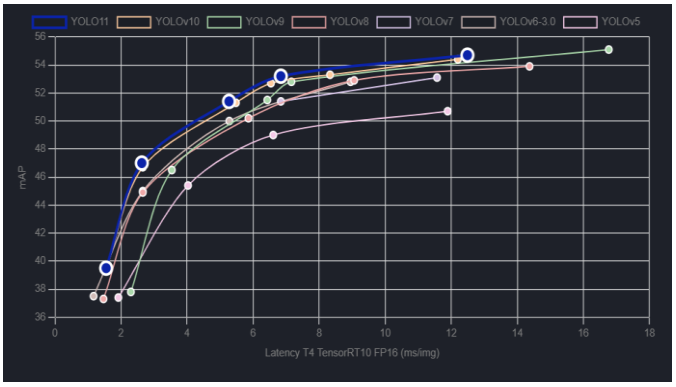
6) Veri Seti İsimlendirilmesi Hatası : Veri setleri Roboflowdan toplandıldığı için veri manipülasyonu yaparken isimlendirme yapılırken ekstra hash şifreleri eklenince windowsa ait sınırlama aşıldı buna karşın txt dosyaları okunamaz hale geldi bunu aşmak için yeni bir kod parçası yapıldı YOLO'da resim ile txt dosyaları aynı isme sahipti oluşturduğumuz algoritma bunları beraber değiştirerek veri kaybına uğramadan dönüşütürmeye yaradı.

IV. YOLO İLE ÇALIŞMA

C. *YOLO PARAMETRELERİ*

Yoloda bir çok parametre bulunmakla beraber bir çok değişik parametre denenmiştir ama ciddi verim artışı gördüğümüz yolları belirmekte fayda var.

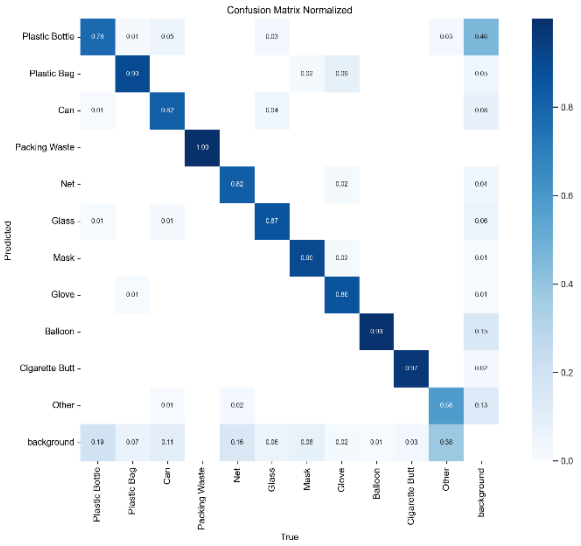
1) Model Seçimi: Yolonun bir çok sürümü olmakla beraber bu sürümlerde parametre farklılığı var örnek verirsek birkaç ay önce çıkan YOLOV11 Nano modelinde 2.5 milyon param varken YOLO'nun en optimal sürümü olarak kabul edilen YOLOV8 de bu sayı 3 milyon param sayısına kadar çıkabiliyor biz daha köklü bulduğumuz için ve param sayısı daha çok olduğu için YOLOV8 kullandık.



Şekil 1 : YOLO sürümlerinin doğruluk oranları

2) YOLO'da Small Ve Nano Farklılığı : Model eğitimine başlarken Nanoda başladık fakat bu da belirli bir noktadan sonra tıkanmaya başlıyordu yaklaşık %80 civarına kadar ulaşabildik Nano modelde fakat daha sonra small modelini keşettik Small model daha ağır yaklaşık 11.5 milyon paramdan oluşan bir model bununla beraber %90 civarı doğruluk oranına ulaştık hemen hemen Nano modelin 4 kat fazla parama sahip ancak bu da nano modelin birçok pcde çalışabilme özelliğinden azda olsa feragat ettiriyor iki modelde testler yapıldı ve denendi.

3) Kod içi parametreler : Kod ile değiştirilebilen birkaç parametre vardır örneğin epoch 30,batch 9 (sistemimiz buna el verdi) optimizer SGD , lr0 0.001 , lrf 0.1, momentum 0.9… diye gider detaylara github hesabı üzerinde ulaşabilirsiniz.



Şekil 2 : Doğruluk matriksi

V. RAPOR UYGULAMASI VE SONUÇLAR

Herşeyin raporunu detaylıca çıkartılması için "Tkinter" kütüphanesi üzerinden Rapor oluşturma PDF olarak çıktısını alma opsiyonelini ekledik peki ne yapıyor drone görüntüsünü yükledikten sonra bize program her saniyenin raporunu detaylıca ne bulduğunu kaç adet bulduğunu tek tek grafiklerle yazılarla raporunu verir ve bunu pdf olarak çıkartır.

A) Sonuçlara gelirsek YOLO'da çıkartılan Sonuçlar

MAp50 (Ortalama Keskinlik): Modelin doğruluğunu ölçmek için kullanılan bir performans metriğidir. Ortalama olarak testlerimizde en düşük nano modelde 65 en yüksek 80 olacak şekilde small model ise 85 ila 92 arasında değişti.

MAp50-95 (Ortalama Keskinlik): Modelin doğruluğunu ölçmek için kullanılan bir performans metriğidir fakat Map50 e göre daha keskindir. Ortalama olarak testlerimizde en düşük nano modelde 50 en yüksek 65 olacak şekilde small model ise 70 ila 80 arasında değişti.

Box Loss: Label etikelerini boyutsal olarak bulma oranın kayıp oranı düşük olması daha iyidir. Nano model 0.95 ve 0.90 arasında değişiyor Small ise 0.90 ila 0.70 kadar düşüş görüldü.

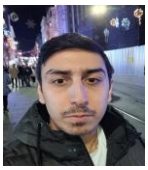
Instances : Nesne tespiti ve bilgisayarla görme (computer vision) alanlarında, bir görüntüdeki belirli bir nesnenin örneğini ifade eder. Nanoda 63 ila 75 arasında doğruluk oranı bulundu small modelde ise 75 ila 85 arasında oran değişti.

Geriye kalan tüm metrikleri GitHub Üzerinden Ulaşabilirsiniz.

B) Optimal Sonuçlara Göre Small VS Nano:

%	MAP50	MAP50-95	Box Loss	Instances
YOLOV8NANO	80	62	0.90	72
YOLOV8SMALL	90	70	0.68	88

CV



Ben Efe İngin, İstanbul Topkapı Üniversitesi 3. sınıf öğrencisiyim. Başta siber güvenlik ve yapay zeka olmak üzere, birçok yazılım alanına ilgim var. Python, Java ve Linux hakkında tecrübeye sahibim. Yaptığım projelerden bazıları:

-Deepfake videoları tespit eden bir yapay zeka modeli geliştirdim.

-Denizdeki atık maddeleri tespit etme projesinde yer aldım.

Detaylı CV ve proje linkleri için mail adresi : inginefe44@gmail.com

Projenin Git-Hub Linki : https://github.com/EfeCyber/WasteWave-Detector

Ben Umut Özkan,3. Sınıf yazılım mühendisliği öğrencisiyim Kotlin, Python , C#, Java gibi dillerde ufak çaplı yaptığım projelerim var sql ve nosql gibi yapılarda deneyimim var yaptığım birkaç projelerden örnek;

Sosyal medyada kullanıcıların günlük paylaşımlarını ve etkinliklerini analiz ederek duygu durumlarını belirleyen bir sosyal medya uygulaması geliştirdim. Bu uygulama, kullanıcıların içeriklerine göre duygu analizi yaparak kişiselleştirilmiş geri bildirimler ve öneriler sunan mobil uygulama.

Birkaç ufak çaplı derin öğrenme projesi ve etiketli resim manipülasyonunu artırma (augmentation) teknikleriyle uygulayan bir proje üzerinde çalıştım. Bu projelerde, verilerin çeşitliliğini artırarak model performansını iyileştirmeyi hedefledim.

Detaylı CV ve proje linkleri için mail adresi : umuttozkannis@gmail.com

Projenin Git-Hub Linki : https://github.com/hopepi/WasteWave-Detector

Ben Enes Çakır. İstanbul Topkapı Üniversitesi Yazılım Mühendisliği 3.Sınıf öğrencisiyim. C# , Python , HTML , CSS gibi derslerde eğitim aldım. İlgi alanlarım daha çok görsel programlama ve yapay zeka eğitimi üzerine. Küçük çaplı kendi ve ekip arkadaşları ile geliştirilen proje ve uygulamalar bulunmakta. Örnek vermek gerekirse ;

-Görsel programlama ile Hastane kişi giriş çıkışları SQL üzerinde tutulan bir otomasyon,

-Derin öğrenme dersinde ekip ile geliştirilen yapay zeka modelleri,gibi örnekleri verebilirim.

Detaylı CV ve proje linkleri için mail adresi : enescakir121@gmail.com

Projenin Git-Hub Linki : https://github.com/enesscakirr/WasteWave-Detector