

TÜRKİYE CUMHURİYETİ
YILDIZ TEKNİK ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ



DERİN ÖĞRENME İLE SENTETİK AÇIKLIKLI RADAR
GÖRÜNTÜLERİNDE GEMİ TESPİTİ

18011092 — Arda KAŞIKÇI
19011050 — Ömer Batuhan ÖZBAY

BİLGİSAYAR PROJESİ

Danışman
Dr. Öğr. Üyesi Ali Can KARACA

Haziran, 2022

TEŞEKKÜR

Proje kapsamında bizden desteğini esirgemeyen, bilgi ve tecrübelerini bizimle paylaşan ve sorularımızı asla cevapsız bırakmayan hocamız Dr. Öğr. Üyesi Ali Can KARACA'ya teşekkür ederiz.

Arda KAŞIKÇI
Ömer Batuhan ÖZBAY

İÇİNDEKİLER

SİMGE LİSTESİ	v
KISALTMA LİSTESİ	vi
ŞEKİL LİSTESİ	vii
TABLO LİSTESİ	viii
ÖZET	ix
ABSTRACT	x
1 Giriş	1
2 Ön İnceleme	2
3 Fizibilite	4
3.1 Teknik Fizibilite	4
3.1.1 Yazılım Fizibilitesi	4
3.1.2 Donanım Fizibilitesi	4
3.2 İş Gücü ve Zaman Planlaması	4
3.3 Yasal Fizibilite	5
3.4 Ekonomik Fizibilite	5
4 Sistem Analizi	6
5 Sistem Tasarımı	7
5.1 Yazılım Tasarımı	7
5.1.1 YOLO	7
5.1.2 EfficientDet	9
5.1.3 SSD MobileNet	10
5.1.4 Faster R-CNN	11
5.2 Veritabanı Tasarımı	11
5.3 Girdi - Çıktı Tasarımı	13

6	Uygulama	15
7	Deneysel Sonular	17
8	Performans Analizi	19
9	Sonu	20
	Referanslar	21
	Özgemiř	23

SİMGE LİSTESİ

Az	Azimuth
Rg	Ground Range

KISALTMA LİSTESİ

AP	Ortalama Kesinlik (Average Precision)
AR	Ortalama Duyarlılık (Average Recall)
CFAR	Yanlış Alarm Oranı (Constant False Alarm Rate)
FPN	Özellik Piramit Ağı (Feature Pyramid Network)
FSI	İnce Şerit Haritalama (Fine Stripmap)
FSII	Geniş İnce Şerit Haritalama (Wide Fine Stripmap)
IOU	Birleşim Üzerinde Kesişim (Intersection over Union)
IW	İnterferometrik Geniş Tarama (Interferometric Wide Swath)
mAP	Ortalama Kesinlik Ortalaması (Mean Average Precision)
NMS	Maksimum Olmayan Bastırma (Non Maximum Supression)
P	Kesinlik (Precision)
QPSI	Quad-Pol Şerit Haritalama (Quad-Pol Stripmap)
QPSII	Geniş Quad-Pol Şerit Haritalama (Wide Quad-Pol Stripmap)
R	Duyarlılık (Recall)
SAR	Sentetik Açıklıklı Radar (Synthetic Aperture Radar)
SM	Şerit Haritalama (Strip Map)
TN	Hatalı Onaylanmış (True Negative)
TP	Doğru Onaylanmış (True Positive)
UFS	Ultra İnce Şerit Haritalama (Ultra-Fine Stripmap)
YOLO	Sadece Bir Kere Bak (You Only Look Once)

ŞEKİL LİSTESİ

Şekil 3.1	İş gücü ve zaman planlamasını gösteren Gantt çizelgesi	4
Şekil 5.1	YoloV3 model karşılaştırması [9]	7
Şekil 5.2	Intersection Over Union örneği [17]	8
Şekil 5.3	EfficientDet mimarisinin genel gösterimi [13]	9
Şekil 5.4	Standart konvolüsyon filtrenin yerine kullanılan mimari [14] . .	10
Şekil 5.5	Faster R-CNN ağının örnek gösterimi [15]	11
Şekil 5.6	Veri setindeki görüntülerden bazı örnekler	12
Şekil 5.7	Gemi görüntüsü ve örnek etiketleri	13
Şekil 6.1	Her model için tespit sonuçlarının karşılaştırılması	16

TABLO LİSTESİ

Tablo 5.1	YOLOv3, YOLOv4 ve YOLOv5 karşılaştırma tablosu [18]	9
Tablo 5.2	Orijinal SAR görüntüleri için ayrıntılı bilgiler [5]	12
Tablo 6.1	YOLO modellerinde kullanılan hiperparametreler	15
Tablo 6.2	EfficientDet D0, SSD-MobileNet V2 ve Faster R-CNN yöntemlerinde kullanılan hiperparametreler	15
Tablo 7.1	YOLO algoritmalarının performans kıyası	17
Tablo 7.2	Faster R-CNN, SSD-MobileNet V2, EfficientDet-D0 ve YOLOv5 yöntemlerini kıyaslayan tablo	18

DERİN ÖĞRENME İLE SENTETİK AÇIKLIKLI RADAR GÖRÜNTÜLERİNDE GEMİ TESPİTİ

Arda KAŞIKÇI
Ömer Batuhan ÖZBAY

Bilgisayar Mühendisliği Bölümü
Bilgisayar Projesi

Danışman: Dr. Öğr. Üyesi Ali Can KARACA

Sentetik açıklıklı radar (SAR), günümüzde uzaktan algılamada kullanılan en önemli aktif görüntüleme sistemlerinden biridir. SAR, bulutlardan, gece ve gündüzden bağımsız olarak çalıştığından gemi tespitinde kullanılabilecek en iyi görüntüleri vermektedir.

Bu makalede, Sentinel-1 ve Gaofen-3 uydularından alınan ve yaklaşık 40.000 görüntüden oluşturulmuş SAR veri setinin yapısı anlatılmaya çalışılmıştır. Veri setinden gemi tespiti yapılabilmesi için uygun olan derin öğrenme algoritmaları belirlenmiştir. Bu veri setinden gemi tespiti için ilk olarak YOLO algoritmasının farklı versiyonları (v3Tiny, v4, v5) kullanılmış olup bu algoritmalar kendileri arasında average precision (AP), average recall (AR) ve tespit süresi gibi belirli parametreler kullanılarak hız ve performans açısından kıyaslanmıştır. Bu kıyaslamalara SSD-MobileNet, EfficientDet D0 ve Faster R-CNN algoritmaları da eklenip gemi tespiti için en uygun algoritma bulunmaya çalışılmıştır.

Anahtar Kelimeler: Gemi tespiti, derin öğrenme, sentetik açıklıklı radar, SAR, nesne tespiti

SHIP DETECTION ON SYNTHETIC APERTURE RADAR IMAGES WITH DEEP LEARNING

Arda KAŞIKÇI
Ömer Batuhan ÖZBAY

Department of Computer Engineering
Computer Project

Advisor: Assist. Prof. Dr. Ali Can KARACA

Nowadays, synthetic aperture radar (SAR) is one of the most important active imaging systems in remote sensing. Since SAR is not affected by clouds, day and night, it gives the best images that can be used in ship detection.

In this paper, the structure of the SAR dataset, which was obtained from Sentinel-1 and Gaofen-3 satellites and composed of approximately 40,000 images, was tried to be explained. Deep learning algorithms fitting for ship detection from the dataset have been decided. First, different versions of the YOLO algorithm (v3Tiny, v4, v5) were used for ship detection. Then these algorithms were compared among themselves in terms of speed and performance using certain parameters like average precision (AP), average recall (AR) and detection time. SSD-MobileNet, EfficientDet D0 and Faster R-CNN algorithms were added to these comparisons and the most suitable algorithm for ship detection was tried to be found.

Keywords: Ship detection, deep learning, synthetic aperture radar, SAR, object detection

1

Giriş

Radarlar aktif olarak yüzeye mikrodalga ışınları göndererek hedefleri yakalar ve böylece sürekli görüntülemeye yol açarlar. Bir radar türü olan sentetik açıklıklı radar (SAR), gemi tespiti için en uygun olanıdır çünkü çözünürlüğü gözlemlenen hedeflerden uzaktayken bile sabittir [1, 2].

SAR aktif bir uzak mesafe sensörüdür yani kendi aydınlatmasını taşır ve güneş ışığına bağlı değildir. Bu özelliği onu her türlü hava koşulunda, gündüz ve gece çalışmalarında işlevsel kılar [3]. SAR sensörleri, kısa sürede büyük miktarda veri üretebilirler. Bu verilere ilgilendiğimiz hedeflerin otomatik tespiti için ihtiyaç duyarız. Bu veriler özellikle bir çalışma alanı olarak, çoğunda açık alan görüntüleri içeren, denizlerde ve okyanuslardaki gemilerin tespiti için kullanılabilir [4]. Gemi tespiti; petrol sızıntısı tespiti, yasadışı balıkçılık, deniz trafiğinin yönetimi ve deniz korsanlığı gibi alanlarda deniz gözetimi için önemlidir [5].

Bu gemi tespiti çalışma alanı için kullanılan iki çeşit yöntem kullanılmaktadır. Bunlardan ilki genelde eskiden kullanılan geleneksel yöntemlerdir. Bu yöntemler temel olarak sürekli yanlış alarm (CFAR) dayanır ve bu yöntemin genel yaklaşımı görüntülerde kara ve okyanusu birbirinden ayırmaya dayanır [3, 6]. Bu yaklaşım da gemilerin tespit hızında yavaşlamaya sebep olur [7, 8]. Diğer bir yöntem olan derin öğrenme ile yapılan çalışmalar ise önceden eğitilmiş nesne tespit araçlarını, gemi tespiti için değiştirmek ve onları SAR görüntüleriyle eğitmektir [5].

Bu raporun geri kalanı şu şekilde organize edilmiştir; 2. Bölümde önceden yapılan SAR görüntülerinde gemi tespiti çalışmaları incelenmiştir. Sonrasında 3. Bölümde bu projenin fizibilite çalışmaları aktarılmıştır. 4. Bölümde sistemin analizi, 5. Bölümde ise sistemin tasarımı üzerinde durulmuştur. 6. Bölümde yaptığımız uygulamalardan örnekler ve çıktılar gösterilmiş, 7. Bölümde ise alınan sonuçlar sunulmuştur. 8. Bölümde bu sonuçların değerlendirmesi yapılmıştır, son bölümde size çalışmanın özeti bulunmaktadır.

2 Ön İnceleme

Öncelikle genel hatlarıyla radar sistemlerinin nasıl çalıştığı ve SAR sisteminin farkının ne olduğu konusunda araştırmalarda bulunduk. Yaptığımız araştırmaların ardından kullandığımız veri setinin makalesi olan "A SAR Dataset of Ship Detection for Deep Learning under Complex Backgrounds" [5] bize yol gösteren ve projenin kalıplarını şekillendiren bir kaynak oldu. Bu makaleyi okuyarak üzerinde çalıştığımız veri seti hakkında genel hatlarıyla bilgi sahibi olduk. Görüntüler içinde farklı radar görüntüleri olduğundan bunların ne gibi hatalara sebep olabileceğini tartıştık. "Yolov3: An incremental improvement" [9] ve "An Improved Tiny-yolov3 Pedestrian Detection Algorithm" [10] adlı makalelerden YoloV3 modeli hakkında bilgi sahibi olduk. YoloV4 modeli için "Yolov4: Optimal Speed and Accuracy of Object Detection" [11] makalesini inceledik. YoloV5 modeli için "A Real-time Apple Targets Detection Method for Picking Robot Based on Improved Yolov5" [12] makalesini inceledik. EfficientDet-D0 için "Efficientdet: Scalable and Efficient Object Detection" [13], SSD MobileNetV2 için "Mobilenets: Efficient Convolutional Neural Networks For Mobile Vision Applications" [14] ve Faster R-CNN için "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks" [15] makalelerini inceleyip genel model yapıları hakkında fikir sahibi olduk.

Kompleks arkaplanlı görüntüler için hangi yöntemlerin daha iyi olabileceği konusunda araştırmalar yaptık. Veri setinde bulunan görüntü ve etiket koordinat dosyalarının eğitimde nasıl kullanılacağı konusunda bilgi sahibi olduk.

Ardından aynı makalede yapılmış olan derin öğrenme çalışmaları sayesinde hangi modellerin denenmiş olduğunu gördük. Bu modellerin analizleri sayesinde ise kendi uygulayacağımız modellerin gereksinimleri hakkında fikir sahibi olduk. Modellerin doğruluğun yanında, modellerin hızlı tespitinin önemini kavradık. Kullanılan modellerin hangi metrikler ile değerlendirildiğini görüp bu değerlendirme metriklerinin nasıl çalıştığı hakkında araştırmalar yaptık.

Projede, makalelerde hem daha önce kullanılmış olan hem de kullanılmamış olan yöntemler eğitilerek karşılaştırılacaktır. Belli modellerin eski halleriyle yeni halleri arasında ne gibi gelişmeler olduğu hakkında değerlendirmeler yapılacaktır. Uygulanan modellerin sonuçları üzerinde değerlendirme yapılacak ve genel olarak modellerin hatalı tespitleri üzerinde bir ortak nokta bulunursa bu durumlar için farklı bir çözüm arayışına girilecektir. Bu çözüm arayışı çalışmanın doğruluğunu arttırmak için kullanılacaktır.

Proje için yapılan ön incelemenin sonucunda projenin ana hatları bu şekilde belirlenmiş olup, çalışmalara bu amaçla başlanmıştır.

3.1 Teknik Fizibilite

3.1.1 Yazılım Fizibilitesi

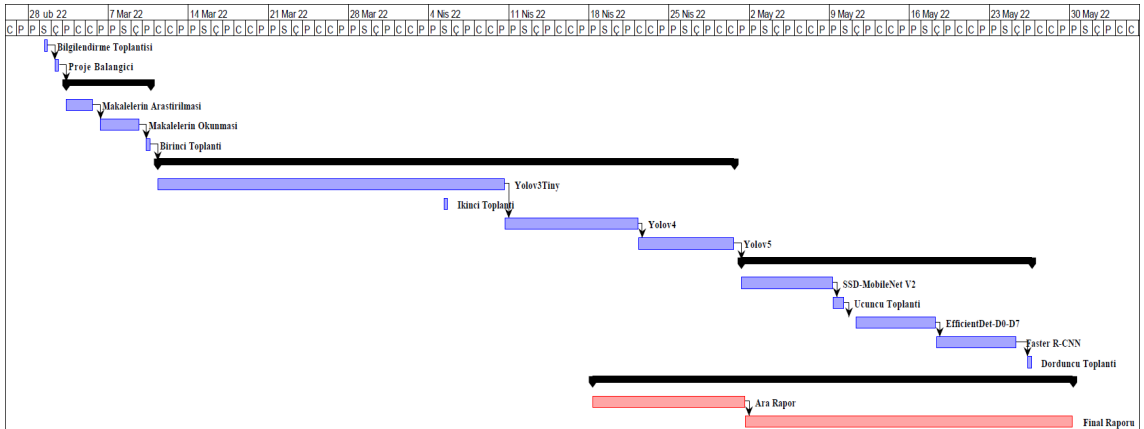
Projede derin öğrenmeye özel kütüphaneleri bulunması sebebiyle Python kullanılmıştır. Python'ın basit ve okunabilir olması, karışık algoritmaların daha kolay kurulmasını sağlar ve bu sayede daha hızlı sonuçlar alınır.

3.1.2 Donanım Fizibilitesi

Veri boyutu ve algoritmaların eğitimi göz önüne alındığında 13GB'lık bellek alanının yeterli olacağı düşünülmektedir. Hafıza için ise minimum 15GB'lık alan uygun görülmüştür. Modeller üzerinde yapılan her türlü çalışma Google Colab üzerinde gerçekleştirilmektedir.

3.2 İş Gücü ve Zaman Planlaması

Belirlenen iş gücü ve zaman planı çizelgesi Şekil 3.1'deki gibidir.



Şekil 3.1 İş gücü ve zaman planlamasını gösteren Gantt çizelgesi

3.3 Yasal Fizibilite

Çalışmamızı gerçekleştirdiğimiz veri seti, halka açık şekilde paylaşılmıştır. Bu nedenle yaptığımız çalışmaların herhangi bir yasal yükümlülüğü yoktur.

3.4 Ekonomik Fizibilite

Bu proje kapsamında herhangi bir gelir söz konusu değildir. Proje sürecinde gider olarak bilgisayarların tükettiği elektriğin faturaları ve 2 aylık Google Colab Pro üyeliği (19,98 \$) bulunmaktadır. Bunun yanında Google Drive’da hafıza sınırına gelinmesi durumunda 3 TL ödeme yapılarak 85GB ek hafıza satın alınmalıdır.

4 Sistem Analizi

Projede kullanılan veri setinde gemi notasyonları YOLO formatında olduğu için projede ilk olarak YOLO algoritmalarının kullanılmasına karar verilmiştir. Başlangıç için YOLOv3tiny algoritmasına karar verilmiştir, bu algoritma eğitilip doğruluk oranları incelendikten sonra daha sonraki sürümler olan YOLOv4 ve YOLOv5 algoritmaları eğitilecek ve karşılaştırılacaktır. Algoritmaların birbirleri arasındaki farklar için ise algoritmaların makaleleri incelenecektir.

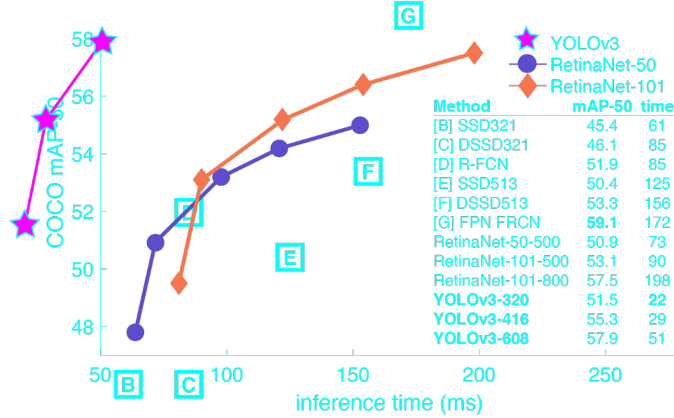
Bir sonraki adımda ise incelenen makalelerde kullanılmış olan Faster R-CNN, SSD-MobileNet v2 ve EfficientDet D0 algoritmaları eğitilip doğruluk oranları hem birbirleriyle hem de makalelerde verilen doğruluk oranlarıyla karşılaştırılacaktır. Bu karşılaştırmanın yapılabilmesi için ise veri setinin notasyon dosyaları algoritmalara uygun hale getirilecektir. Doğruluk oranları arasında ciddi farkların oluşması durumunda bu farkların nedenleri incelenecektir.

Bu karşılaştırmalar yapılırken Rafael Padilla tarafından yazılan "A Comparative Analysis of Object Detection Metrics with A Companion Open-Source Toolkit" makalesinde tanıtılmış arayüz kullanılacaktır [16]. Kullanılan metrikler ise PASCAL-VOC ve COCO Challenge'larında kullanılan average precision, average recall ve bu metriklerin varyasyonlarıdır.

5.1 Yazılım Tasarımı

5.1.1 YOLO

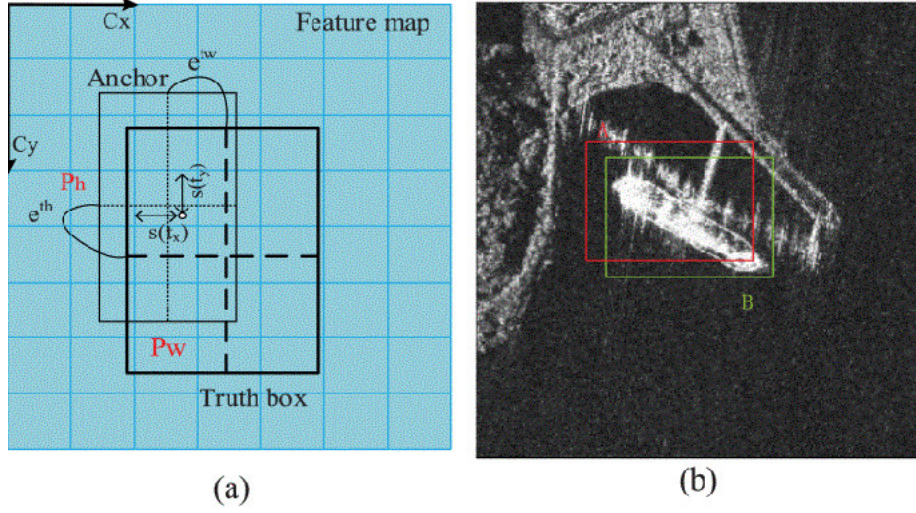
YOLO modeli İngilizce açılımıyla "You Only Look Once", son zamanlarda nesne tespiti için sık kullanılan bir modeldir. Bu model konvolüsyonel sinir ağlarını kullanarak nesne tespiti yapar. Popüler olmasının sebebi de aslında isminde yatar. "Yalnızca bir kez bak" şeklinde çevrilebilen isminden de anlaşılabilceği gibi bu model bir görüntüye sadece bir kez bakarak yani görüntü üzerinde tespit ve sınıflandırma işlemini tek adımda gerçekleştirir. Kısaca modelin avantajı, modelin küçük boyutunda ve hızlı hesaplama hızında yatmaktadır.



Şekil 5.1 YoloV3 model karşılaştırması [9]

Genel olarak YOLO algoritmasından bahsedilmesi gerekirse, bu algoritma görüntüyü her biri eşit $S \times S$ boyutlu N adet ufak bölgeye ayırarak çalışmaktadır. Bu N adet bölgenin her biri, kendi bölgesindeki nesnenin varlığını sorgular. Bölge içinde nesne tespit edildiyse ve bu nesnenin merkezi kendi alanındaysa, algoritma nesne için sınırlayıcı kutu çizer. Fakat aynı nesneyi farklı sınırlayıcı kutu tahminleriyle tespit etmiş olan birden fazla bölge yinelenen tahmin sonucunu ortaya çıkarır. Bu durumda sınırlayıcı kutular oluşur. Bunları filtrelemek için de Non Maximum Supression (NMS) yöntemi kullanılır.

Bu yöntem sınırlayıcı kutuların sahip oldukları güven skorlarına bakarak, en büyük güven skoruna sahip sınırlayıcı kutuyu kendine baz olarak alır. Sonrasında bu yüksek güven skorlu sınırlayıcı kutular arasında IOU değerine göre bir kıyas yapılır ve IOU skorları bir eşik değerinin üzerinde olan sınırlayıcı kutular bastırılır.



Şekil 5.2 Intersection Over Union örneği [17]

Yolov3Tiny, YOLOv3'ten çok daha az sayıda evrişim katmanına sahip olan, YOLOv3'ün basitleştirilmiş bir sürümüdür. Bu durum, modelin büyük miktarda bellek işgal etmesine gerek olmadığı ve donanım ihtiyacını azalttığı anlamına gelir. Ayrıca tespiti büyük ölçüde hızlandırır, ancak tespit doğruluğunun bir kısmını kaybeder [10].

YOLOv4'te CSPDarknet53 özellik çıkarım modelinin daha iyi sonuç verdiği görülmüştür ve bu model kullanılmaya başlanmıştır. YOLOv3'te kullanılan FPN omurgası yerine PANet kullanılmıştır. Head yapısı ise YOLOv3 ile aynıdır [11]. Birçok geliştirme ile model doğruluğu ve hızı arttırılmaya çalışılmıştır.

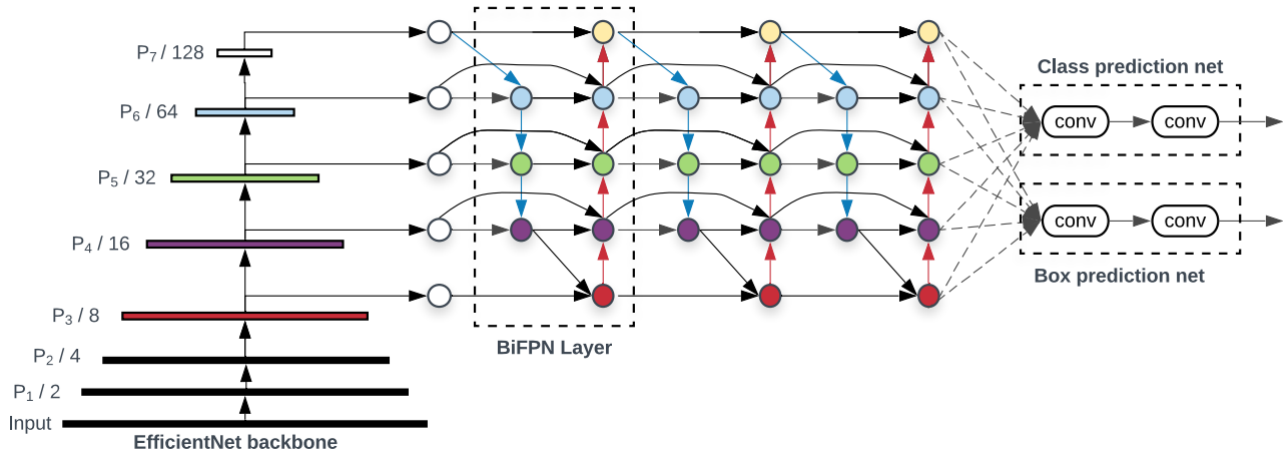
YOLOv5, YOLO algoritmalarının en son ürünüdür. YOLOv5'te PyTorch implementasyonu kullanılmıştır ve aktivasyon fonksiyonlarında geliştirme yapılmıştır. YOLOv5'in ağırlık dosyası YOLOv4'ün ağırlık dosyasından yaklaşık %90 daha küçüktür. Bu nedenlerden ötürü YOLOv5'in avantajları yüksek tespit doğruluğu, hafif olması ve aynı zamanda hızlı olması olarak söylenebilir [12].

Tablo 5.1 YOLOv3, YOLOv4 ve YOLOv5 karşılaştırma tablosu [18]

	YOLOv3	YOLOv4	YOLOv5
Yapay Sinir Ağı Tipi	Tam Konvolüsyon	Tam Konvolüsyon	Tam Konvolüsyon
Backbone Özellik Tanımlayıcı	Darknet-53	CSPDarknet53	CSPDarknet53
Kayıp Fonksiyonu	İkili Çapraz Entropi	İkili Çapraz Entropi	İkili Çapraz Entropi ve Logit Kayıp Fonksiyonu
Neck	FPN	SSP ve PANet	PANet
Head	YOLO katmanı	YOLO katmanı	YOLO katmanı

5.1.2 EfficientDet

EfficientDet, Google Research'ün Beyin Takımı tarafından geliştirilmiştir. Temelde 8 adet EfficientDet modeli vardır, bu modeller D0'dan D7'ye gidecek şekilde isimlendirilmiştir. D7 en güçlü donanım için tasarlanmıştır ve en büyük EfficientDet modelidir. Şekil 5.3'te mimarinin genel yapısı görselle açıklanmıştır.

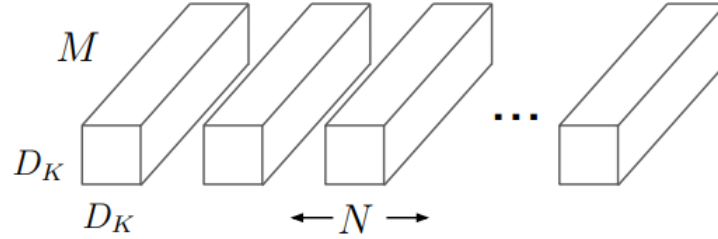


Şekil 5.3 EfficientDet mimarisinin genel gösterimi [13]

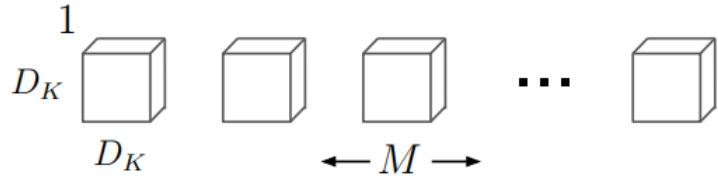
Şekil 5.3 incelendiğinde mimarinin üç ana başlığa ayrılabilir olduğu görülür. Backbone olarak ImageNet-pretrained EfficientNets kullanılmıştır. Modelin ikinci ana parçası Bidirectional feature pyramid network (BiFPN), normal FPN'lerin farklı çözünürlükteki girdileri kaynaştırma konusundaki başarısızlığını gidermek üzerine tasarlanmıştır. BiFPN top-down ve bottom-up olarak çift yönlü özellik füzyonu uygulayarak özelliklerin önemini daha doğru şekilde vermektedir. Füzyonlaşmış özellikler sınıf ve kutu ağına girerek sınıf ve sınır kutusu tahmini yapılır [13].

5.1.3 SSD MobileNet

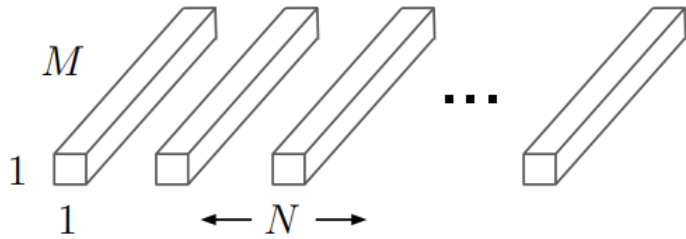
MobileNet, mobil ve gömülü görüntü uygulamaları için tasarlanan bir konvolüsyonel sinir ağıdır. Düşük gecikme süresine sahip olan hafif derin sinir ağıları oluşturmak için "depthwise separable convolutions" kullanan bir mimariye dayanmaktadır. Hafif olması nedeniyle kıyaslama yapılırken sık sık kullanılmaktadır.



(a) Standard Convolution Filters



(b) Depthwise Convolutional Filters



(c) 1×1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Separable Convolution

Şekil 5.4 Standart konvolüsyon filtrenin yerine kullanılan mimari [14]

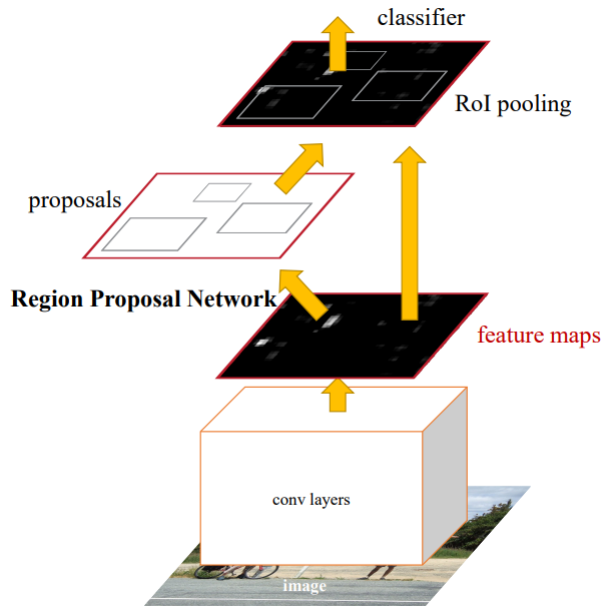
Şekil 5.4'te Standart konvolüsyon filtresi yerine depthwise konvolüsyonel filtresi ve pointwise konvolüsyon ekleyerek MobileNet modelinin diğer yöntemlerden daha az maliyetli bir çözüm sunmasını sağlamıştır.

"MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications" [14] makalesinde yapılan kıyaslamalara göre yapılan işlem sayısı azalmış fakat doğruluk korunmuştur.

5.1.4 Faster R-CNN

R-CNN metotlarının başarılı olmasından sonra devam edilen geliřtirmelerle maliyetleri son derece dūřmūřtur, Fast R-CNN derin aęlar kullanarak neredeyse gerek zamanlı sonular vermektedir. Genelde bōlge önerisi iin Selective Search kullanılsa da bu metot efektif tespit aęlarına gōre resim bařına yaklařık 2 saniye daha yavař kalmaktadır [15].

Faster R-CNN iki modūlden oluřmaktadır. Bu modūllerin ilki olan tam konvolūsyonel aę, yukarıda bahsedilen Selective Search ile aynı iřlevi gōrmektedir. İkinci modūl ise önerilen bōlgeleri kullanan Fast R-CNN dedektōrūdūr. řekil 5.5’den anlařılacaęı ūzere tūm sistem birleřmiř tek bir aędan oluřmaktadır.



řekil 5.5 Faster R-CNN aęının ōrnek gōsterimi [15]

Konvolūsyon aęlarından oluřturulmuř feature map’ın sliding window teknięiyle taranır. Bu özellik iki adet tamamen baęlanmış katmanlara beslenir. Bu katmanlar box-regression ve box-classification katmanlarıdır [15]. Gelen parametrelere gōre bōlge önerisi yapıldıktan sonra řekil 5.5’te gōrūldūęı ūzere Region of Interest (RoI) pooling’e gōnderilir. RoI’de sınıflandırma yapılarak nesne tespiti tamamlanmıř olur.

5.2 Veritabanı Tasarımı

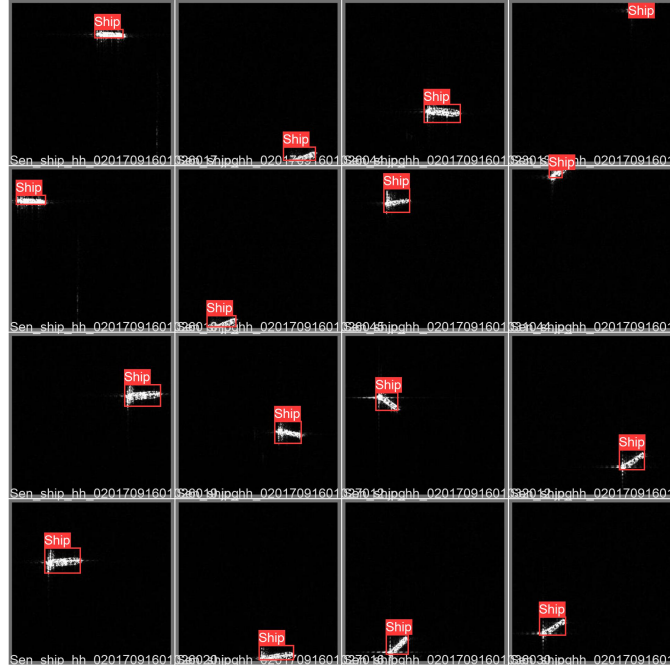
Projede kullanılan veri seti, "A SAR Dataset of Ship Detection for Deep Learning under Complex Backgrounds" [5] makalesinde oluřturulan veri setidir. Veri seti, Github sayfasında herkese aık olarak paylařılmaktadır. Veri setinde 102 Gaofen-3 ve 108 Sentinel-1 gōrūntülerinden 59.535 gemi bulunmaktadır. Bu gōrūntüler 39,729

görüntü ortaya çıkacak şekilde kırılmıştır ve 256 x 256 pikselliktedir. Bu görüntüler çözünürlük, geliş açısı ve arkaplan açısından çeşitlidir.

Tablo 5.2 Orijinal SAR görüntüleri için ayrıntılı bilgiler [5]

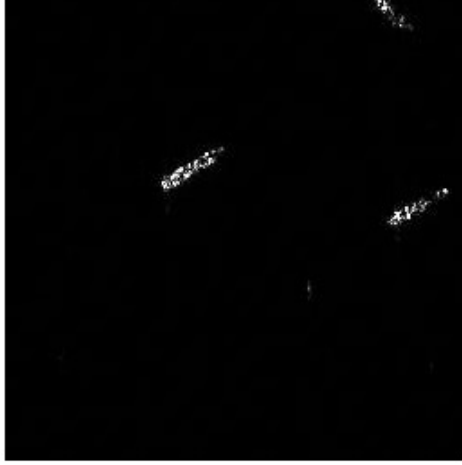
Sensör	Görüntü Modu	Çözünürlük Rg. x Az. (m)	Alan (km)	Açı	Polarizasyon	Görüntü Sayısı
GF-3	UFS	3 x 3	30	20~50	Tek	12
GF-3	FSI	5 x 5	50	19~50	Çift	10
GF-3	QPSI	8 x 8	30	20~41	Tam	5
GF-3	FSII	10 x 10	100	19~50	Çift	15
GF-3	QPSII	25 x 25	40	20~38	Tam	5
Sentinel-1	SM	1.7 x 4.3~3.6 x 4.9	80	20~45	Çift	49
Sentinel-1	IW	20 x 22	250	29~46	Çift	10

Şekil 5.6’de veri setinden bazı örnek görüntüler etiketlenmiş halde verilmiştir.



Şekil 5.6 Veri setindeki görüntülerden bazı örnekler

Veri setinde bulunan görüntüler SAR uzmanları tarafından etiketlenmiştir. Bu etiketlemeler YOLO formatındadır. YOLO formatında her bir resim dosyası için bir txt dosyası mevcuttur. Her bir txt dosyasında her bir obje için beş adet değer vardır. Bu değerler sırayla; sınıf numarası, objenin x eksenine göre merkezi (x), objenin y eksenine göre merkezi (y), genişlik (w) ve yüksekliktir (h). Veri setinde sınıf sayısı 1 olduğu için ilk değer her zaman sıfırdır.



0 0.88671875 0.443359375 0.1328125 0.08203125
 0 0.3984375 0.361328125 0.140625 0.08984375
 0 0.826171875 0.029296875 0.05859375 0.05859375

Şekil 5.7 Gemi görüntüsü ve örnek etiketleri

Şekil 5.7’te görüldüğü üzere görüntüde 3 gemi mevcuttur ve sınıf numarası, x, y, w ve h değerleri yazılmıştır. Bu değerlerin normalize edilmesi formatın önemli bir parçasıdır, görüntünün sol üst köşesi (0,0), sağ alt köşesi (1,1) olacak şekilde hareket edilmelidir.

Veri seti modellerde kullanılmak üzere eğitilirken %73 eğitim, %27 test olacak şekilde ayrılmıştır. 28933 görüntü eğitim, 10796 görüntü ise test amaçlı kullanılmıştır. İdeal durumda görüntüleri eğitim ve test için ayırırken görüntülerin hangi uydudan geldiğine de dikkat edilmelidir. Bununla birlikte veri setindeki bazı görüntülerin "shipxxxxxxx" ve "newshipxxxxxxx" şeklinde isimlendirilmesi görüntülerin hangi uydudan geldiğinin anlaşılmasına yol açmıştır. Bu sebepten ötürü görüntüler eğitim ve test olarak ayırırken uydular dikkate alınmamıştır.

5.3 Girdi - Çıktı Tasarımı

Yapılan eğitimlerin değerlendirilmesi için 14 farklı kriter kullanılmıştır. Bu kriterlerin çoğu Precision (Kesinlik) ve Recall (Duyarlılık) değerleri baz alınarak oluşturulmuştur. Bu yüzden bu değerlerin hesaplanması için ilk olarak Precision ve Recall değerleri hesaplanmalıdır.

$$Precision = \frac{TP}{TP + FP} \quad (5.1)$$

$$Recall = \frac{TP}{TP + FN} \quad (5.2)$$

Yukarıdaki denklemlerde Precision değeri, doğru pozitif değerinin doğru pozitif ve yanlış pozitif değerleri toplamına bölünmesiyle elde edilir. Doğru tahmini yapılan değerlerin doğruluk oranını verir. Recall ise doğru pozitif değerinin doğru pozitif ve yanlış negatif değerlerinin toplamına bölünmesiyle elde edilir. Doğru tahmini yapılan

değerin toplam doğru miktarına oranını verir.

İdeal durumda hem Precision hem de Recall'ın 1 olması beklenir. Buna karşın hem yanlış pozitifin hem de yanlış negatifin 0 olması mümkün olmadığı için bu iki değeri dikkate alan ayrı bir metrik gerekliliği oluşmuştur.

Mean Average Precision (Ortalama Kesinlik Değerlerinin Ortalaması), nesne tespiti çalışmalarında kullanılan en popüler metriklerden biridir, standart hale gelmiştir. Bu değer, average precision, yani ortalama kesinlik (AP) değerinin ortalamasıdır. MAP'ın anlaşılabilmesi için ilk olarak AP açıklanmalıdır. Yapılan bu çalışmada sadece gemi sınıfı olduğu için AP değeri mAP değeri ile eşdeğerdir. AP, aşağıdaki gibi hesaplanmaktadır:

$$AP = \sum_n (R_n - R_{n-1}) P_n \quad (5.3)$$

Bu çalışmada farklı AP metrikleri kullanılmıştır. Bu metriklerden bahsetmek gerekirse, AP, AP^{50}, AP^{75} metrikleri [%50:%5:%75], %50 ve %75 gibi farklı IOU değerlerine göre hesaplanır ve bu çalışmada dikkate alınmıştır. Bu metriklere ek olarak küçük, orta ve büyük boyutlu gemilerin AP değerini gösteren AP^S, AP^M, AP^L metrikleri de karşılaştırma yapılırken kullanılmıştır. Bu metrikler COCO değerlendirme metriklerince hesaplanmıştır. Sınıfa göre AP ve mAP metrikleri de PASCAL challenge'da olduğu gibi değerlendirilmiştir.

Average Recall (AR) değeri aşağıdaki gibi hesaplanır:

$$AR = 2 \int_{0.5}^1 R_{IOU}(o) do \quad (5.4)$$

Bu denklemde, $R_{IOU}(o)$ "o" IOU değerine göre recall değerini veren bir fonksiyondur. AP'da olduğu gibi, AR'ın da farklı varyasyonları bu çalışmada kullanılmıştır. AR^S, AR^M, AR^L AP karşılıklarında olduğu gibi boyutlara göre recall değerlerini göstermek için kullanılmıştır. Bu çalışmada kullanılan diğer 3 AR varyasyonu ise AR^1, AR^{10} ve AR^{100} metrikleridir. AR^1 metriği her görüntü için en fazla bir tespiti dikkate alır, AR^{10} ve AR^{100} metrikleri de görüntü başına 10 ve 100 tespiti dikkate alan metriklerdir.

Bu çalışmada kullanılan 13 metrik yukarıda açıklanmıştır, bu metriklerin yanında bizim için en önemli metriklerden biri tespit süresidir. Bu süre hesaplanırken tüm test veri seti dikkate alınmıştır. Performans analizi yapılırken hem zaman hem de değerlendirme metrikleri incelenmiştir.

6 Uygulama

Tablo 6.1’de YOLO algoritmalarının eğitimi için verilen hiperparametreler gösterilmiştir.

Tablo 6.1 YOLO modellerinde kullanılan hiperparametreler

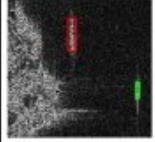
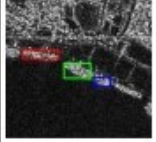

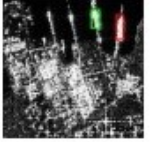
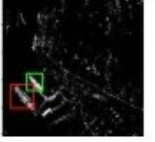
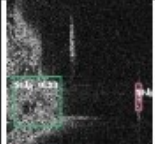
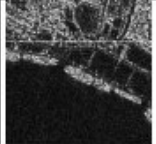

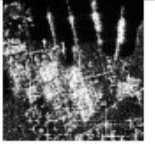
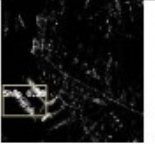
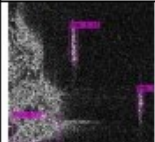
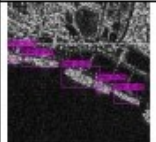
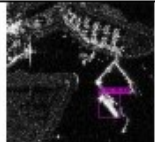
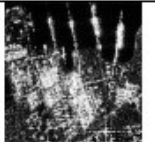
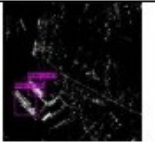

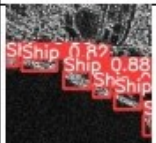

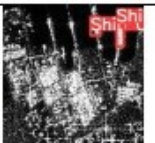

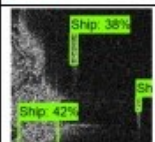
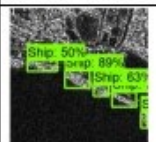
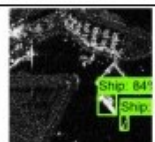
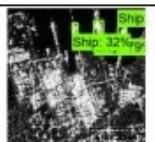


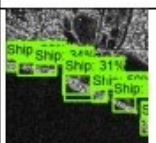
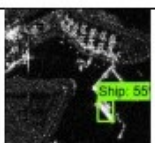



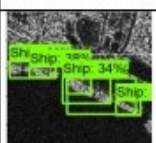
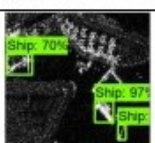


Parametreler	YOLOv3Tiny	YOLOv4	YOLOv5
Batch Size	64	16	32
Momentum	0.9	0.949	0.937
Max Batches	2000	2000	30(Epoch)
Learning Rate	0.001	0.001	0.01

Tablo 6.2’de çalışmada kullanılan diğer algoritmaların hiperparametreleri verilmiştir.

Tablo 6.2 EfficientDet D0, SSD-MobileNet V2 ve Faster R-CNN yöntemlerinde kullanılan hiperparametreler

Parametreler	EfficientDet D0	SSD-MobileNet V2	Faster R-CNN
Batch Size	8	16	2
Number of Steps	30000	30000	30000
Learning Rate Base	0.01	0.01	0.05
Warmup Learning Rate	0.0001	0.0001	0.0001
Warmup Steps	2000	2000	2000
Momentum Optimizer	Cosine Decay	Cosine Decay	Cosine Decay
Momentum Value	0.9	0.9	0.9

Bu hiperparametrelerle oluşturulan modellerin sonuçları Deneysel Sonuçlar bölümünde açıklanmıştır. Daha sonraki bölüm olan Performans Analizi bölümünde ise bu sonuçlar yorumlanıp değerlendirilmiştir. Şekil 6.1’de bir önceki bölümde verilmiş olan hiperparametrelerle oluşturulmuş modellerin elle seçilmiş görüntülerdeki performansı verilmiştir.

Orijinal					
YoloV3Tiny					
Yolov4					
Yolov5					
EfficientDet-D0					
SSD-MobileNet V2					
Faster R-CNN					

Şekil 6.1 Her model için tespit sonuçlarının karşılaştırılması

Şekil 6.1’de ilk satırdaki görüntüler uzmanlarca etiketlenmiş doğruluğu kesin orijinal verilerdir. Alt satırlarda ise kullanılan modellerin tespit sonuçları verilmiştir. Bu görüntülerden de anlaşılacağı üzere modeller karmaşık arkaplanlı radar görüntülerinde de çalışıp, doğru sonuçlara yakın sonuçlar vermiştir.

7 Deneysel Sonuçlar

Çalışmanın ilk adımında YOLO yöntemleri kendi içlerinde kıyaslanmıştır. Tespit süreleri belirlenirken Şekil 6.1’da kullanılan görüntülerin ortalama tespit süresi dikkate alınmıştır.

Tablo 7.1 YOLO algoritmalarının performans kıyası

Model	mAP	Tespit Süresi (ms)
Yolov3Tiny	0.7518	85
Yolov4	0.85	22
Yolov5	0.92	50

Tablo 7.1’de görüldüğü üzere Yolov3Tiny hem zaman hem de mAP açısından en kötü yöntemdir. Yolov4 ve Yolov5 kıyasında ise yöntemin başarısı ile tespit süresi arasında bir ödünleşim söz konusudur. Bu çalışmada diğer algoritmalarla kıyas yapılması için mAP’ı daha yüksek olan Yolov5 seçilmiştir.

Tablo 7.2, Rafael Padilla ve arkadaşları tarafından kaleme alınan "A Comparative Analysis of Object Detection Metric with a Companion Open-Source Toolkit" makalesiyle birlikte yayınlanan nesne tespiti değerlendirme yazılımıyla hesaplanmıştır [16]. Tabloda kullanılan metriklerin hepsi çalışmanın Sistem Tasarımı bölümünün Girdi-Çıktı Tasarımı alt-bölümünde açıklanmıştır.

Tablo 7.2 Faster R-CNN, SSD-MobileNet V2, EfficientDet-D0 ve YOLOv5 yöntemlerini kıyaslayan tablo

Metrikler	Faster R-CNN	SSD-MobileNet V2	EfficientDet-D0	YOLOv5
mAP(%)	80.03	86.98	90.49	92.00
Tespit Süresi (sn)	0.09	0.017	0.04	0.05
AP (%)	35.18	40.02	46.40	52.82
AP^{50} (%)	78.58	85.89	89.63	90.91
AP^{75} (%)	25.35	30.67	42.44	55.68
AP^S (%)	32.39	36.31	43.48	49.70
AP^M (%)	39.39	45.04	50.60	57.81
AP^L (%)	16.30	44.42	49.56	66.66
AR^1 (%)	36.58	40.24	43.37	48.06
AR^{10} (%)	44.69	50.81	54.50	60.88
AR^{100} (%)	44.71	50.94	54.71	61.17
AR^S (%)	41.99	47.08	51.22	57.12
AR^M (%)	48.76	56.32	59.61	66.80
AR^L (%)	3.67	62.67	61.00	71.67

Tablodaki parametrelerin incelenip değerlendirilmesi Performans Analizi bölümünde yapılmıştır.

8 Performans Analizi

Değerlendirmeye başlamadan önce dikkat edilmesi gereken ilk nokta bu yöntemlerin eşit parametrelerle eğitilmemiş olmalarıdır. Bu durumun tek sebebi kullanılan algoritmaların ideal parametrelerde çalışma ortamını oldukça zorlamasıdır. Örnek vermek gerekirse, Tablo 6.2’de Faster R-CNN’in batch size’ının 2 olduğu dikkat çekmektedir. Bu durumun yaşanmaması için çalışma ortamında Faster R-CNN’in batch size’ı sırayla 16-8-4-2 olarak düzenlenmiştir fakat ortamı fazla zorlamasından ötürü eğitim batch size 2 olacak şekilde yapılmıştır. Açıkladığımız durumlar adil bir analiz yapamayacağımızı düşündürse de yöntemlerin aynı ortamda eğitilebilir olmasının da bir kıstas olduğunu düşünüyoruz ve bu yüzden Tablo 6.2’nin değerlendirme yapmamıza engel olmaması gerektiğine kanaat getirdik.

Tablo 7.2’de ilk olarak en başarılı yöntemin artık standart hale gelmiş mAP metriğine göre YOLOv5 olduğu görülmektedir. EfficientDet-D0, YOLO’ya oldukça yakın sonuç vermiş olup gemileri biraz daha hızlı tespit etmiştir. Bu iki yöntem arasındaki en önemli farklar büyük gemilerin tespit edilmesinde kullanılan AP^L ve AR^L metrikleri ve 0.75’ten büyük IOU değerleri için hesaplanan AP^{75} metriğinde görülmektedir. Buradan YOLO algoritmasının EfficientDet’ten en önemli farkının gemi sınırlarını daha iyi çizmesi ve büyük gemileri tespit kabiliyetinin daha iyi olduğu anlaşılmaktadır.

SSD-MobileNet V2 daha hızlı sonuçlar verme üzerine tasarlanmıştır ve bu konudaki başarısını en yakın rakibine oranla 2 kattan daha hızlı şekilde gemileri tespit ederek kanıtlamıştır.

Tabloda dikkat çeken bir diğer nokta Faster R-CNN’in diğer yöntemlere kıyasla kötü bir performans göstermesidir. Faster R-CNN hem hız olarak en yakın yöntem olan YOLO’dan yaklaşık 2 kat daha yavaştır, hem de büyük gemileri bulma konusunda AP^L ve AR^L metriklerinden anlaşılabileceği üzere oldukça düşük başarı oranına sahiptir. Bu durumun ilk paragrafta açıklanan sebeplerden ötürü iyi eğitilememesinden kaynaklandığını düşünmekteyiz.

9 Sonuç

Bu çalışmada gemi veri seti [5] üzerinden hızlı ve doğru şekilde gemi tespiti yapılmaya çalışıldı. Bu amaç için çeşitli YOLO modelleri (v3Tiny,v4,v5), Faster R-CNN, SSD-MobileNet V2 ve EfficientDet-D0 modelleri eğitilip test sonuçları karşılaştırılarak değerlendirilmiştir. Bu değerlendirmeye için kullanılan metrikler, bu metrikleri kolayca ulaşılabilir kılan nesne tespiti aracıyla [16] elde edilmiştir.

Yapılan performans analizlerinin sonucunda gemi tespiti için donanım bilgilerini verdiğimiz ortamı göz önüne alarak YOLOv5 modelinin kullanılması gerektiği sonucuna vardık. Bunun yanında TensorFlow kütüphanelerinin kullanımının gerekli olabileceği durumlarda EfficientDet-D0'ın YOLO'ya iyi bir alternatif olacağını düşünmekteyiz.

- [1] A. Moreira, P. Prats-Iraola, M. Younis, G. Krieger, I. Hajnsek, and K. P. Papathanassiou, "A tutorial on synthetic aperture radar," *IEEE Geoscience and remote sensing magazine*, vol. 1, no. 1, pp. 6–43, 2013.
- [2] C. Oliver and S. Quegan, *Understanding synthetic aperture radar images*. SciTech Publishing, 2004.
- [3] K. El-Darymli, E. W. Gill, P. McGuire, D. Power, and C. Moloney, "Automatic target recognition in synthetic aperture radar imagery: A state-of-the-art review," *IEEE access*, vol. 4, pp. 6014–6058, 2016.
- [4] D. J. Crisp, "The state-of-the-art in ship detection in synthetic aperture radar imagery," Defence Science and Technology Organisation Salisbury (Australia) Info ..., Tech. Rep., 2004.
- [5] Y. Wang, C. Wang, H. Zhang, Y. Dong, and S. Wei, "A sar dataset of ship detection for deep learning under complex backgrounds," *remote sensing*, vol. 11, no. 7, p. 765, 2019.
- [6] K. El-Darymli, P. McGuire, D. Power, and C. R. Moloney, "Target detection in synthetic aperture radar imagery: A state-of-the-art survey," *Journal of Applied Remote Sensing*, vol. 7, no. 1, p. 071 598, 2013.
- [7] U. Kanjir, H. Greidanus, and K. Oštir, "Vessel detection and classification from spaceborne optical images: A literature survey," *Remote sensing of environment*, vol. 207, pp. 1–26, 2018.
- [8] K. Ouchi, "Current status on vessel detection and classification by synthetic aperture radar for maritime security and safety," in *Proceedings of the 38th Symposium on Remote Sensing for Environmental Sciences, Gamagori, Aichi, Japan*, 2016, pp. 3–5.
- [9] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.
- [10] Z. Yi, S. Yongliang, and Z. Jun, "An improved tiny-yolov3 pedestrian detection algorithm," *Optik*, vol. 183, pp. 17–23, 2019.
- [11] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.
- [12] B. Yan, P. Fan, X. Lei, Z. Liu, and F. Yang, "A real-time apple targets detection method for picking robot based on improved yolov5," *Remote Sensing*, vol. 13, no. 9, p. 1619, 2021.
- [13] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 781–10 790.

- [14] A. G. Howard *et al.*, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” *arXiv preprint arXiv:1704.04861*, 2017.
- [15] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *Advances in neural information processing systems*, vol. 28, 2015.
- [16] R. Padilla, W. L. Passos, T. L. B. Dias, S. L. Netto, and E. A. B. da Silva, “A comparative analysis of object detection metrics with a companion open-source toolkit,” *Electronics*, vol. 10, no. 3, 2021, ISSN: 2079-9292. DOI: 10.3390/electronics10030279. [Online]. Available: <https://www.mdpi.com/2079-9292/10/3/279>.
- [17] C. Chen, C. He, C. Hu, H. Pei, and L. Jiao, “A deep neural network based on an attention mechanism for sar ship detection in multiscale and complex scenarios,” *IEEE Access*, vol. 7, pp. 104 848–104 863, 2019.
- [18] U. Nepal and H. Eslamiat, “Comparing yolov3, yolov4 and yolov5 for autonomous landing spot detection in faulty uavs,” *Sensors*, vol. 22, no. 2, p. 464, 2022.

BİRİNCİ ÜYE

İsim-Soyisim: Arda KAŞIKÇI
Doğum Tarihi ve Yeri: 18.06.2000, İstanbul
E-mail: ardakskc@gmail.com
Telefon: 0534 613 51 83
Staj Tecrübeleri: Yok

İKİNCİ ÜYE

İsim-Soyisim: Ömer Batuhan ÖZBAY
Doğum Tarihi ve Yeri: 09.12.1999, Aşkabat
E-mail: batuhan.ozbay@std.yildiz.edu.tr
Telefon: 0551 111 23 53
Staj Tecrübeleri: Yok

Proje Sistem Bilgileri

Sistem ve Yazılım: Windows İşletim Sistemi, Python
Gerekli RAM: 13GB
Gerekli Disk: 25GB