İNSANSIZ HAVA ARACI GÖRÜNTÜLERİNDEN DERİN ÖĞRENME YÖNTEMLERİYLE NESNE TANIMA

Semih Murat
171402109

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Yüksek Lisans Programı
Danışman: Dr. Öğr. Üyesi Erdal Güvenoğlu

İstanbul
T.C. Maltepe Üniversitesi
Lisansüstü Eğitim Enstitüsü
Şubat, 2021

İNSANSIZ HAVA ARACI GÖRÜNTÜLERİNDEN DERİN ÖĞRENME YÖNTEMLERİYLE NESNE TANIMA

Semih Murat
171402109
Orcid: 0000-0002-8491-1690

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Yüksek Lisans Programı
Danışman: Dr. Öğr. Üyesi Erdal Güvenoğlu

İstanbul
T.C. Maltepe Üniversitesi
Lisansüstü Eğitim Enstitüsü
Şubat, 2021



JÜRİ VE ENSTİTÜ ONAYI

Bu belge, Yükseköğretim Kurulu tarafından 19.01.2021 tarihli "*Lisansüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge*" ile bildirilen 6689 Sayılı Kişisel Verilerin Korunması Kanunu kapsamında gizlenmiştir.

ETİK İLKE VE KURALLARA UYUM BEYANI

Bu belge, Yükseköğretim Kurulu tarafından 19.01.2021 tarihli "*Lisansüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge*" ile bildirilen 6689 Sayılı Kişisel Verilerin Korunması Kanunu kapsamında gizlenmiştir.

TEŞEKKÜR

Tez çalışmam boyunca her zaman olumlu, yol gösterici ve motive edici katkılarından dolayı değerli danışmanım Dr. Öğr. Üyesi Erdal Güvenoğlu'na teşekkür ederim.

Yüksek lisans eğitimim süresince beni her zaman destekleyen başta sevgili eşim Sedanur Murat ve oğlum Ömer Alp Murat olmak üzere aileme ve de özellikle tez çalışması süresince desteklerinden ötürü kardeşlerim Beytullah Murat ve Pervin Acar ile meslektaşım Sertaç Çetinkaya'ya teşekkür ederim.

Semih Murat

Şubat, 2021

ÖZ

İNSANSIZ HAVA ARACI GÖRÜNTÜLERİNDEN DERİN ÖĞRENME YÖNTEMLERİYLE NESNE TANIMA

Semih Murat
Yüksek Lisans Tezi
Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Yüksek Lisans Programı
Danışman: Dr. Öğr. Üyesi Erdal Güvenoğlu
Maltepe Üniversitesi Lisansüstü Eğitim Enstitüsü, 2021

İnsansız hava araçlarının kullanımı ve önemi sürekli artmaktadır. Her geçen gün hayatın farklı bir alanında bu araçların kullanımıyla ilgili yeni bir gelişme yaşanmaktadır. Yaşanan bu gelişmeler sayesinde havadan alınan görüntülerde hem nitelik hem nicelik anlamında da artış olmaktadır. Havadan alınan görüntüler kullanılarak nesne algılama çalışmaları için en önemli ihtiyaçlardan birisi olan kaliteli ve büyük veri setleri insansız hava araçları ile giderilmiştir. Hem bu alanda hem de bilgisayarlı görü dünyasında yaşanan gelişmeler hava fotoğraflarından nesne algılama çalışmalarını daha da önemli bir hale getirmiştir. Özellikle evrişimsel sinir ağları dünyasında sürekli daha hızlı ve daha başarılı modeller geliştirilmektedir. Bu modellerin başarısının hava fotoğrafları ile de anlaşılması literatür için oldukça önemli ve gereklidir. Bu tez çalışmasında alanda eksik olduğu görülen son dönemde geliştirilmiş hızlı ve başarılı evrişimsel sinir ağları kullanılarak hava fotoğraflarından nesne algılama çalışmaları yapılmıştır. Deneysel çalışmalar sonucunda elde edilen sonuçlar grafiklerle anlaşılır bir şekilde analiz edilmiştir.

Anahtar Sözcükler: İnsansız Hava Araçları, Hava Fotoğrafı, Nesne Algılama, Evrişimsel Sinir Ağları.

ABSTRACT

DEEP LEARNING METHODS OBJECT RECOGNITION FROM UNMANNED AERIAL VEHICLE IMAGESPHOTOS

Semih Murat
Master's Thesis
Department of Computer Engineering
Computer Engineering Programme
Advisor: Asst. Prof. Erdal Güvenoğlu
Maltepe University Graduate School, 2021

The use and importance of unmanned aerial vehicles is constantly increasing. Every day there is a new development regarding the use of these tools in a different area of life. Thanks to this increase, there is an increase in both quality and quantity of aerial images. Using aerial images, one of the most important needs for object detection studies, high quality and large data sets were met with unmanned aerial vehicles. Developments in both this field and the world of computer vision have made object detection studies from aerial photographs even more important. Especially in the world of convolutional neural networks, faster and more successful models are constantly being developed. Understanding the success of these models with aerial photographs is very important and necessary for the literature. In this thesis, object detection studies from aerial photographs have been carried out using the recently developed fast and successful convolutional neural networks, which are seen to be lacking in the field. The results obtained as a result of experimental studies were analyzed with graphics in an understandable way.

Keywords: Unmanned Aerial Vehicle, Air Photo, Object Detection, Convolutional Neural Network.

İÇİNDEKİLER

JÜRİ VE ENSTİTÜ ONAYI ii
ETİK İLKE VE KURALLARA UYUM BEYANI iii
TEŞEKKÜR iv
ÖZ v
ABSTRACT vi
İÇİNDEKİLER vii
ŞEKİLLER LİSTESİ ix
TABLOLAR LİSTESİ xi
KISALTMALAR LİSTESİ xi
ÖZGEÇMİŞ xiv
BÖLÜM 1. GİRİŞ 1
1.1. İnsansız Hava Araçları 2
1.2. Hava Fotoğrafı 5
BÖLÜM 2. LİTERATÜR ARAŞTIRMASI 6
BÖLÜM 3. MATERYAL VE YÖNTEM 12
3.1. Yapay Zekâ 12
3.2. Makine Öğrenmesi 14
3.2.1. Denetimli Öğrenme 14
3.2.2. Denetimsiz Öğrenme 15
3.3. Derin Öğrenme 15
3.3.1. Yapay Sinir Ağları 16
3.3.2. Tek Katmanlı Algılayıcılar 18
3.3.3. Çok Katmanlı Algılayıcılar 19
3.3.4. İleri Yönde Yayılım 20
3.3.5. Maliyet Fonksiyonun Hesaplanması 21
3.3.6. Geri Yönde Yayılım 24
3.3.7. Hiperparametreler 25
3.3.7.1. Öğrenme Hızı 26
3.3.7.2. Aktivasyon Fonksiyonları 26
3.3.7.3. Optimizasyon ve En İyileme 28
3.3.7.4. Devir Sayısı ve Yığın Boyutu 30
3.4. Evrişimsel Sinir Ağları 31
3.4.1. Evrişim Katmanı 32
3.4.2. Havuzlama Katmanı 32
3.4.3. Tam Bağlantılı Katman 33
3.4.4. Hiperparametreler 33
3.4.4.1. Filtre Boyutu 33
3.4.4.2. Dolgulama ve Adım Kaydırma 34
3.4.4.3. Havuzlama Türü ve Boyutu 35
3.5. Veri Setleri 35
3.5.1. ImageNet Veri Seti 35
3.5.2. Pascal VOC Veri Seti 36
3.5.3. MS COCO Veri Seti 36
3.5.4. MNIST Veri Seti 36

3.5.5. CIFAR Veri Seti 36
3.6. Evrişimsel Sinir Ağı Modelleri 37
3.6.1. YOLOv3 37
3.6.2. YOLOv5 40
3.6.3. RetinaNet 42
3.6.4. EfficientDet 43
BÖLÜM 4. GERÇEKLEŞTİRİLEN UYGULAMA VE PERFORMANS SONUÇLARI 45
4.1. Kullanılan Veri Setleri 45
4.2. Değerlendirme Metrikleri 52
4.2.1. Hata Matrisi 52
4.2.2. Kesinlik 53
4.2.3. Duyarlılık 54
4.2.4. Doğruluk 56
4.2.5. Ortalama Doğruluk Oranı 56
4.3. Veri Setlerinin Aynı Modele Göre Performans Karşılaştırması 57
4.4. Hava Fotoğrafları için Modellerin Karşılaştırılması 62
4.4.1. YOLOv5 Modelinin Çalışma Sonuçları 62
4.4.2. YOLOv3 Modelinin Çalışma Sonuçları 63
4.4.3. RetinaNet Modelinin Çalışma Sonuçları 64
4.4.4. EfficientDet Modelinin Çalışma Sonuçları 66
4.4.5. Tüm Modellerin Çalışma Sonuçlarının Karşılaştırılması 68
BÖLÜM 5. SONUÇLAR 83
KAYNAKÇA 86

ŞEKİLLER LİSTESİ

Şekil 1.1. Ülkelerin insansız hava aracı üretim tablosu. 4
Şekil 3.1. Yapay zekâ ve alt alanları hiyerarşisi gösterimi 13
Şekil 3.2. Denetimli makine öğrenmesi süreci 15
Şekil 3.3. İnsan sinir hücresinin yapısı. 16
Şekil 3.4. Yapay Sinir Hücresi. 17
Şekil 3.5. Tek katmanlı yapay sinir ağı yapısı. 19
Şekil 3.6. Çok Katmanlı Yapay Sinir Ağı Yapısı 20
Şekil 3.7. Dereceli alçalma fonksiyonu. 24
Şekil 3.8. Aktivasyon fonksiyonları. 27
Şekil 3.9. Eğim inişi gösterimi. 29
Şekil 3.10. Momentum kullanılan ve kullanılmayan olasılıksal eğim inişi. 30
Şekil 3.11. Evrişim işlemi 32
Şekil 3.12. Maksimum havuzlama katmanı. 33
Şekil 3.13. Evrişim işlemi öncesinde dolgulama uygulanan giriş görüntüsü. 34
Şekil 3.14. YOLO modelinin mimari görünümü. 38
Şekil 3.15. YOLOv3 modelinin kullandığı Darknet-53 mimarisi. 40
Şekil 3.16. Tek Aşamalı Nesne Tespit Dedektörü. 41
Şekil 3.17. RetinaNet modelinin mimarisi. 43
Şekil 3.18. EfficientNet omurgası üzerine kurulan EfficientDet mimarisi. 44
Şekil 4.1. DOTA veri setinden örnek görüntüler 48
Şekil 4.2. Stanford veri setinin eğitim, doğrulama ve test dağılım grafiği 50
Şekil 4.3. Stanford Drone veri setinden örnek görüntüler 51
Şekil 4.4. YOLOv5 modelinin kesinlik grafiği 54
Şekil 4.5. YOLOv5 modelinin duyarlılık grafiği 55
Şekil 4.6. YOLOv5 modelinin 1000 epok eğitiminde kesinlik-duyarlılık eğrisi 55
Şekil 4.7. Birleşim Üzerinden Kesişim Gösterimi 57
Şekil 4.8. YOLOv5 modelinin farklı çözünürlüklerle eğitim sonuçları 60
Şekil 4.9. YOLOv5 modelinin farklı çözünürlük ve epok sayıları ile eğitim sonuçları. 61
Şekil 4.10. YOLOv5 modelinin 1000 epok eğitim sonucu 62
Şekil 4.11. YOLOv3 modelinin 100 epok eğitim sonucu 63

Şekil 4.12. RetinaNet modelinin 100 epok eğitim sonucu 65
Şekil 4.13. EfficientDet modeli 250 epok eğitim sonucu..... 67
Şekil 4.14. YOLOv3 ile EfficientDet modellerinin karşılaştırılması 69
Şekil 4.15. Modellerin Başarı Oranları..... 73
Şekil 4.16. Test görüntüsünün YOLOv5 çıkarım görüntüsü ile karşılaştırılması 75
Şekil 4.17. Test görüntüsünün YOLOv3 çıkarım görüntüsü ile karşılaştırılması 76
Şekil 4.18. Test görüntüsünün RetinaNet çıkarım görüntüsü ile karşılaştırılması 77
Şekil 4.19. Test görüntüsünün EfficientDet çıkarım görüntüsü ile karşılaştırılması..... 78
Şekil 4.20. Test görüntüsünün YOLOv5 çıkarım görüntüsü ile karşılaştırılması 79
Şekil 4.21. Test görüntüsünün YOLOv3 çıkarım görüntüsü ile karşılaştırılması 80
Şekil 4.22. Test görüntüsünün RetinaNet çıkarım görüntüsü ile karşılaştırılması 81
Şekil 4.23. Test görüntüsünün EfficientDet çıkarım görüntüsü ile karşılaştırılması..... 82

TABLOLAR LİSTESİ

Tablo 3.1. Aktivasyon fonksiyonları. 27
Tablo 4.1. DOTA Veri Setinin Etiket Dağılımı 47
Tablo 4.2. Stanford Drone Veri Setinin Etiket Dağılımı 49
Tablo 4.3. Hata Matrisi 53
Tablo 4.4. Modellere ait hiperparametrelerin gösterimi 58
Tablo 4.5. İki farklı veri setinin YOLOv5 modeli ile eğitim sonuçları 58
Tablo 4.6. YOLOv5 modeli farklı çözünürlük ve epok değerleri ile çalışma sonuçları 59
Tablo 4.7. YOLOv3 modelinin test veri seti ait ortalama doğruluk oranı tablosu 64
Tablo 4.8. RetinaNet modeli değerlendirme veri seti ortalama doğruluk oranı tablosu 66
Tablo 4.9. EfficientDet modelinin test veri seti ortalama doğruluk oranı tablosu 68
Tablo 4.10. Tüm modellerin karşılaştırılması 70
Tablo 4.11. Tüm modellerin test görüntüleri ile karşılaştırılması 72
Tablo 4.12. Modellerin tahmin yüzdesi 72

KISALTMALAR

CIFAR	**:**	Kanada İleri Araştırma Enstitüsü Veri Seti (Canadian Institute For Advanced Research Dataset)
CNN	**:**	Evrişimsel Sinir Ağları (Convolutional Neural Network)
CPU	**:**	Merkezi İşlem Birimi (Central Process Unit)
DOTA	**:**	Havadan Görüntülerde Nesne Algılama için Büyük Ölçekli Veri Kümesi (A Large-scale Dataset for Object Detection in Aerial Images)
EPCC	**:**	Çok Yüzlü Konik Sınıflandırıcı (Extended Polyhedral Classifier)
Faster R-CNN	**:**	Daha Hızlı Bölge Tabanlı Evrişimsel Sinir Ağları (Faster Region Based Convolutional Neural Network)
GPU	**:**	Grafik İşlemci Birimi (Graphics Processing Unit)
ILSVRC	**:**	ImageNet Büyük Ölçekli Görsel Tanıma Yarışması (ImageNet Large Scale Visual Recognition Challenge)
IoU	**:**	Birleşim üzerinden Keşisim (Intersection over Union)
İHA	**:**	İnsansız Hava Aracı
JSON	**:**	JavaScript Nesne Gösterimi (JavaScript Object Notation)
mAP	**:**	Ortalama Doğruluk Oranı (Mean Average Precision)
MNIST	**:**	Değiştirilmiş Ulusal Standartlar ve Teknoloji Enstitüsü Veri Seti (Modified National Institute of Standards and Technology Dataset)
MOR-UAV	**:**	İnsansız Hava Aracından Hareketli Nesne Tanıma (Motion Object Recognition Unmanned Aerial Vehicles)
MS COCO	**:**	Microsoft Bağlam İçinde Ortak Nesneler (Microsoft Common Object in Context)
PASCAL VOC	**:**	Örüntü Analizi, İstatistiksel Modelleme ve Hesaplamalı Öğrenme Görsel Nesne Sınıfları (Pattern Analysis, Statistical Modelling and Computational Learning Visual Object Classes)
R-CNN	**:**	Bölge Tabanlı Evrişimsel Sinir Ağları (Region based Convolutional Neural Network)
ReLU	**:**	Düzleştirilmiş Doğrusal Birim (Rectified Linear Units)
RMSProp	**:**	Kök Ortalama Kare Yayılımı (Root Mean Square Propagation)

RPN	**:**	Bölge Teklif Ağı (Region Proposal Network)
SGD	**:**	Olasılıksal Dereceli Azalma (Stochastic Gradient Descent)
SSD	**:**	Tek Atış Dedektörü (Single Shot Detector)
SVM	**:**	Destek Vektör Makinesi (Support Vector Machine)
UAS	**:**	İnsansız Hava Sistemi (Unmanned Aerial System)
UAV	**:**	İnsansız Hava Aracı (Unmanned Aerial Vehicle)
VEDAI	**:**	Hava Görüntülerinde Araç Algılama Veri Seti (Vehicle Detection in Aerial Imagery Dataset)
YOLO	**:**	Sadece Bir Kez Bakarsınız (You Only Look Once)

ÖZGEÇMİŞ

Semih MURAT

Bilgisayar Mühendisliği Anabilim Dalı

Eğitim

Derece	*Yıl*	*Üniversite, Enstitü, Anabilim/Anasanat Dalı*
Y.Ls.	2021	Maltepe Üniversitesi, Lisansüstü Eğitim Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı
Ls.	2014	Marmara Üniversitesi, Atatürk Eğitim Fakültesi Bilgisayar ve Öğretim Teknolojileri Eğitimi
Lise	2005	Kayseri Sümer Lisesi

İş/İstihdam

Yıl	*Görev*
2015 - 2020	Bilişim Teknolojileri ve Yazılım Öğretmeni - Ataşehir Emlak Konut Ortaokulu
2020 - …	Kayseri İl Milli Eğitim Müdürlüğü

Kişisel Bilgiler

Doğum yeri ve yılı : Yozgat, 1988 Cinsiyet: E

Yabancı diller : İngilizce (Orta Seviye)

GSM / e-posta : smhmurat@gmail.com

BÖLÜM 1. GİRİŞ

Dijitalleşen ve dijitalleşme yolunda attığı adımları her geçen gün daha da hızlandıran dünyada birkaç yıl öncesine kadar sadece ülkelerin savunma sistemlerinde kullanılan insansız hava araçları artık birçok sektörde kullanılmaya başlanmıştır. Gökyüzünde belirli bir mesafede istenilen her noktayı gösterebilme kabiliyetine sahip olan hareketli kameralara sahip insansız hava araçları güvenlik, tarım, ulaşım, lojistik hatta arama-kurtarma çalışmalarında dahi işleri çok kolaylaştıran ve hızlandıran araçlar haline gelmiştir. Teknolojinin gelişmesi ve yapay zekâ özelinde derin öğrenme algoritmalarının da çok daha etkili olması ile insansız hava araçlarından alınan fotoğraf ya da video görüntüleri kullanılarak ülke sınırlarında milli güvenliği, trafikte yol durumunu ve güvenliğini, tarlada ekili ürünlerin hastalığını tespit etmek bilgisayarlı görünün konusu olmuştur. İnsansız hava araçları görsel gözetim veya arama kurtarma gibi birçok alanda kullanıldığından, gelecekteki araştırmalara kaynak olabilmesi için bu araçlardan elde edilen görüntüler kullanılarak kapsamlı bir kıyaslamanın yapılması gerekmektedir.

Son yıllarda dünya genelinde kullanılan insansız hava aracı sayısı sürekli artış göstermektedir. Bu araçlar toplumu ilgilendiren farklı alanlarda kullanılmaktadır. Kullanım alanlarının en dikkat çekici amaçlarından biri de özellikle doğal afetlerin sık görüldüğü Japonya'da afet sonrası etkilenen bölgeleri insansız hava araçları ile taramak ve tahliye edilenleri tespit etmektir. Başka bir kullanım amacı da dezavantajlı bölgelerde acil hastalara tıbbi malzeme ulaştırmak ve gıda malzemesine ulaşma imkânı olmayanlara gıda tedariki sağlamaktır [1].

Yapay zekanın yaşadığı gelişim bilgisayarla görme dünyasında daha verimli ve etkili görme sistemlerinin geliştirilmesi talebine yol açmıştır. Anlambilim içeren nesne tanıma, davranış analizi ya da sahne anlama gibi birçok görev içi farklı performans değerlendirme araçları geliştirilmiştir. Bu araçlardan ImageNet yarışması nesne algılama ve sınıflandırma, MOTChallenge2015 yarışması çoklu nesne takibi ya da VOT2016 yarışması tek nesne takibi için öne çıkan performans değerlendirme görevleridir. Bu yarışmalarda kullanılan veri setleri genellikle sabit kameralar ya da yer hizasında bulunan kameralardan alınan görüntülerden oluşmaktadır. Ancak bir insansız hava aracından

alınan görüntüler aracın esnek olması, kamera hassasiyeti ve arka planların karmaşık olması ya da nesnelere olan uzaklık nedeniyle daha zorlu performanslar gerektirmektedir. Dolayısıyla insansız hava araçlarından elde edilen görüntülerden nesne tanıma ve algılama çalışması yaparken farklı parametrelerden dolayı diğer veri setleri ile alınan performans ve verim alınamamaktadır [2].

Teknolojiyi insan yararına ve insan hayatını kolaylaştıracak şekilde kullanmayı amaç edinen yapay zekâ ve derin öğrenme çalışmaları; bir askeri görevlinin bilgisayar başında saatlerce kalarak ve üst düzey dikkat gerektiren görüntü izleme ve nesne tespit etme çalışmalarında çok başarılı sonuçlar üretmektedir. Bu sayede insan gücünün verimli kullanılması ve insan kaynaklı hata ve ihmallerin azaltılması sağlanmıştır. Ancak yapılan uygulamalar ve çalışmalar göz önüne alındığında derin öğrenme tabanlı nesne tanıma çalışmalarındaki geliştirmeler genellikle yer mesafesinde insan gözüyle aynı hizada kullanılan kameralardan alınan görüntülerle sağlanmaktadır.

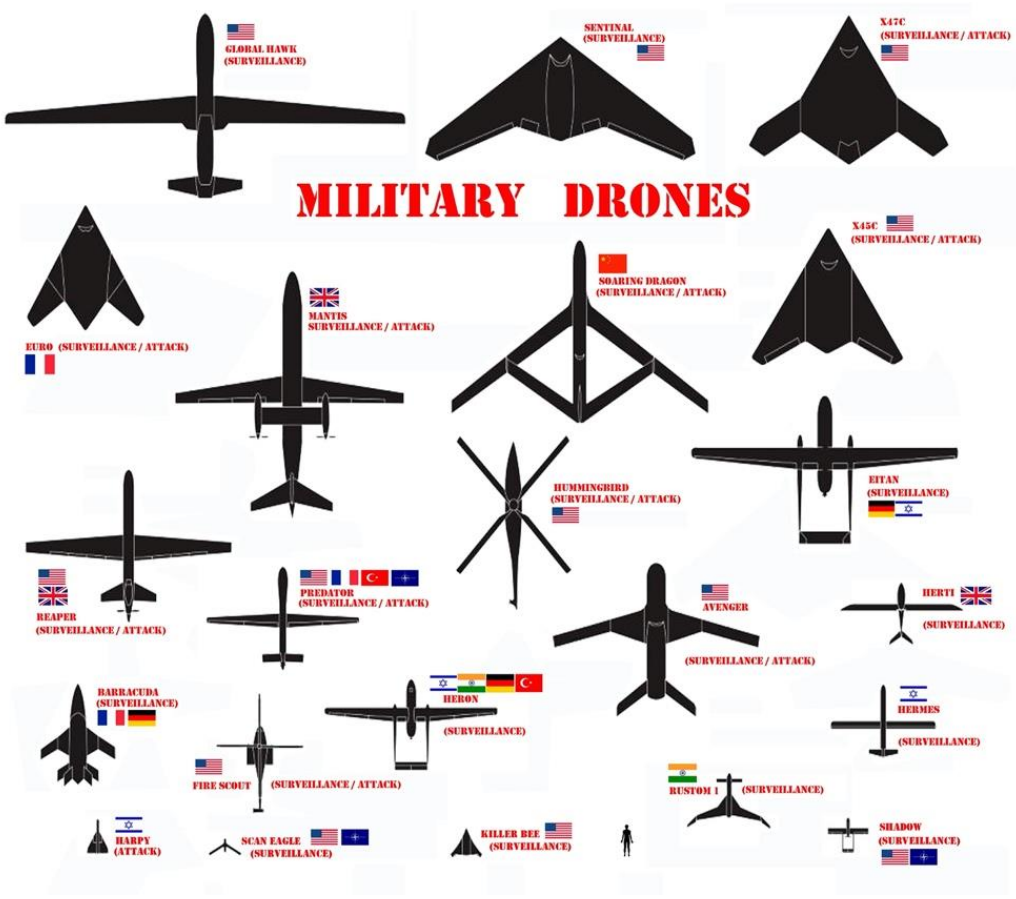
İnsansız hava araçlarından alınan görüntülerde bir nesnenin sadece uzay açı olarak adlandırılan belirli bir perspektifi görüntülenmektedir. Bu görüntüleme açısının bazı dezavantajları ve zorlukları bulunmaktadır. İnsansız hava araçlarının farklı yüksekliklerden aldığı fotoğraflarda nesneler farklı büyüklüklerde ve açı olarak farklı yönlerde olmaktadır. Alınan görüntülerin arka planının karmaşık olması nedeniyle zayıf algılama performansı ortaya çıkabilir. Tüm bu nedenler nesne tanıma çalışmalarının performansını düşürmektedir. Dolayısıyla nesne tanıma çalışmaları yer mesafesinden alınan görüntülere göre daha düşük öznitelikler kullanılarak yapılmaktadır. İnsansız hava araçlarından alınan görüntüler perspektif olarak daha çok özellik içerse de tanınacak nesneye dair daha az özellik içermektedir.

1.1. İnsansız Hava Araçları

Günümüzde insansız hava araçları yerde bulunan bir pilot tarafından gönderilerek uzaktan kumanda ile kontrol edilen ya da önceden oluşturulmuş uçuş programı yüklenerek belirlenen rotasında otomatik olarak uçurulan araçlardır. İnsansız hava araçları teknik özelliklerine göre ve kullanım amaçlarına göre genel olarak iki kategoriye ayrılır. Teknik özelliklerine göre insansız hava araçları ağırlık, kanat yapısı, kontrol şekli, kullandığı yakıt ya da enerji kaynağı olarak alt kategorilere ayrılırken kullanım amaçları

olarak da askeri ve sivil amaçlı olmak üzere iki alt kategoriye ayrılmaktadır. Askeri amaçlı insansız hava araçları keşif, imha, silahlı saldırı gibi farklı görevlerde kullanılabilmektedir. Uluslararası literatürde drone, insansız hava araçları ya da insansız hava sistemleri gibi farklı isimlendirmeler yapılsa da belirli teknik özellikler dışında gerçekte aynı anlama gelmektedir. Diğer taraftan, ilk versiyonları drone olarak bilinen ve bugünkü kadar yüksek teknik özelliklere sahip olmayan insansız hava araçları günümüze kadar genelde askeri amaçlı ve silah olarak kullanıldıklarından ve bu bağlamda toplumların hafızasında olumsuz bir imaja sahip olduklarından, günümüzde sivil sektörde daha çok UAV/UAS (İHA) isimleri kullanılmaktadır. Aradaki farklılığa teknik açıdan bakıldığında ise, drone otomatik olarak gidebilen insansız hava veya deniz araçlarını ifade ederken UAV ise insansız olarak uçabilen, motoru olan, kontrol edilebilen ve kendisi silah olmayan uçağı ifade etmektedir [3].

Birinci Dünya Savaşı yıllarından itibaren askeri alanda ülkelerin savaş kabiliyetini önemli ölçüde artıran hava araçları hem kapasite ve güç olarak geliştirilmeye devam etmiş hem de insansız askeri hava aracı icat etme fikri gerçekleştirilmeye çalışılmıştır. Özellikle ülkemizde de son yıllarda yaşanan gelişmeler gözetleme ve bilgi toplama faaliyetleri yapan insansız hava araçlarının üretilmesinin yanı sıra silahlı insansız hava araçlarının da üretilebilmesini sağlamıştır. Şekil 1.1'de ülkelerin yakın dönemde sahip oldukları askeri anlamda kullanılan hava araçları gösterilmektedir.



Şekil 1.1. Ülkelerin insansız hava aracı üretim tablosu [4].

Dünya nüfusunun hızla artmasına paralel olarak özellikle metropol ve büyük şehirlerde trafik yoğunluğu ülkelerin öncelikli problemlerinden biri olmuştur. Sivil alanda kullanımı artan insansız hava araçları yenilikçi ve öncü teknoloji firmaları ve e-ticaret platformları tarafından kargo taşımacılığında kullanılmaya başlanmıştır. Ancak asıl amacın kargo taşımacılığının da ötesinde uçan hava araçları ile yeryüzündeki araç trafiğini azaltmak ve gökyüzü trafiğine geçmek olduğu bir gerçektir. Bu kapsamda özellikle ülkemizde ve gelişmiş ülkelerde yapılan uçan araba tasarımları ve yarışmaları dikkate alındığında teknolojideki ilerlemelerin ve değişikliklerin çok daha hızlı olacağı görülmektedir. Teknolojik gelişmeler sadece hava aracının uçuş kabiliyeti üzerinde değil aynı zamanda yapay zekâ gelişmeleri kullanılarak güvenliği sağlayacak şekilde de olmaktadır. Bu zamanda kara araçlarında kullanılan şerit takip sistemi, mesafe kontrol sistemi ve akıllı uyarı sistemi gibi yenilikler derin öğrenme mimarileri sayesinde hava araçlarında da kullanılabilecektir.

Tüm bu gelişmeler takip edildiğinde insansız hava araçları hem kapasite, amaç, güvenlik ve teknolojik olarak gelişmekte hem de derin öğrenme mimarileri ve algoritmaları ile desteklenip insan hayatında çok daha fazla alanda kullanılmaya başlanmaktadır.

1.2. Hava Fotoğrafı

Hava fotoğrafı çeşitli amaçlar için kullanılmak üzere hava taşıtları kullanılarak gökyüzünden kuşbakışı fotoğrafların çekilmesidir. Çekim için uçak, helikopter, balon, insansız hava araçları ya da uydu araçları kullanılabilmektedir. Hava fotoğrafları özellikle askeri alanda, haritacılık sektöründe, arkeoloji çalışmalarında, afet sonrası arama ve kurtarma faaliyetlerinde ya da tarımsal faaliyetlerde hızlı veri toplama amacıyla kullanılmaktadır. Bunun yanı sıra şehir içi ya da şehirler arası yol güzergahlarında trafik durumunu izleme ve kontrol amaçlı, turizm bölgelerinde tanıtım amaçlı da kullanılabilmektedir [5].

Gökyüzünden fotoğraf çekme fikri ilk olarak 1794 yılında Fleurus savaşında bilgi toplama amaçlı gerçekleştirilmiştir. Savaş esnasında yere bağlı bir balon kullanılarak havadan görüntü alınmıştır. Daha sonra 1840 yılında Fransız bilim insanları topoğrafik harita yapma metodunu açıklamışlar ve havadan serbest uçan bir balon kullanılarak fotoğraf almışlardır. 1860 yılında yere bağlı bir balonla 1200 metre yükselerek Amerika Birleşik Devletleri'nde Boston eyaletinin fotoğrafı çekilmiştir. 1906 yılında Rusya'da bir rokete monte edilen kamera kullanılarak 800 metre yükseklikten hava fotoğrafı alınmıştır. Hava fotoğraflarında dikkat çeken gelişmelerden bir diğeri de 1909 yılında bir İtalyan subayın uçaktan bugünkü gibi sıralı hava fotoğrafı çekebilmesi olmuştur [5].

BÖLÜM 2. LİTERATÜR ARAŞTIRMASI

İnsansız hava araçları kullanılarak elde edilen görüntüler üzerinden derin öğrenme teknikleriyle nesne tanıma çalışmaları genel olarak iki ayrı kategoriye ayrılmaktadır. Bu kategoriler; tek aşamalı nesne tespit algoritmaları ve iki aşamalı nesne tespit algoritmalarıdır. Tek aşamalı nesne tespiti çalışmalarında YOLO, RetinaNet, EfficientNet ve SSD modeli ön plana çıkarken, iki aşamalı nesne tespit çalışmalarında R-CNN, Fast R-CNN ve Faster R-CNN modelleri en başarılı ve hızlı sonuçları vermektedir.

Sarıbaş, Çevikalp ve Kahveci insansız hava araçlarından çekilen imgelerde arabaların algılanması ve konumlarının bulunması problemi için çok yüzlü konik sınıflandırıcı kullanan nesne konum bulma algoritması ve literatürde en hızlı ve iyi sonuçları veren YOLOv2 algoritmasını karşılaştırmak amacıyla bir çalışma yapmışlardır. Çalışmada kendi oluşturdukları ve içerisinde insansız hava aracından elde edilmiş yaklaşık 10000 imgenin bulunduğu veri setini kullanmışlar ve elde edilen sonuçlara göre ortalama doğruluk oranı olarak en iyi sonucu 8 kök sezicili EPCC algoritmasının hız olarak ise en iyi sonucu YOLO-Tiny algoritmasının verdiğini tespit etmişlerdir [6].

Toraman, 2018 yılında yapmış olduğu çalışmada 2012 yılının ImageNet yarışmasını kazanan AlexNet ve 2014 yılında aynı yarışmada ikinci olan VGG16 derin öğrenme modellerini karşılaştırarak insansız hava aracından alınan görüntülerden yaya tespiti yapmıştır. Çalışmada nesne tanıma amacıyla kullanılan algoritmanın yanı sıra transfer öğrenme ve veri artırımı yöntemlerinin de etkisi incelenmiştir. Stanford Üniversitesi drone veri setinin kullanıldığı çalışma sonucu her iki modelde de yaklaşık %99 yaya sınıflandırması başarı oranı yakalanmıştır [7].

Wang ve diğerleri yaptıkları çalışmada yine aynı şekilde Stanford üniversitesi drone veri setini kullanarak RetinaNet algoritması üzerinde Faster-RCNN ve SSD mimarilerini karşılaştırmışlar ve çalışma sonucunda ortalama doğruluk oranı değerleri olarak tek aşamalı SSD mimarisinin %80,42 iki aşamalı Faster-RCNN mimarisinin %83,63 oranını yakaladığı tespit etmişlerdir. Çalışma sonuçlarına göre tek aşamalı mimarilerin başarı olarak iki aşamalı mimarilere yaklaştığı sonucuna ulaşmışlardır [8].

Benjdira ve arkadaşları araç gözetimi ve trafik izleme senaryosunu gerçekleştirmek için tek aşamalı YOLOv3 ile iki aşamalı Faster R-CNN modelini

karşılaştırmışlardır. İnsansız hava araçlarından alınan görüntülerden oluşan veri seti kullanılarak yaptıkları çalışmada performans değerlendirmesini doğruluk ve işlem süresi açısından ele almışlardır. Çalışma sonuçlarına göre her iki algoritma da kullanılan veri setinde nesne tahmin doğruluğunu en az %99 ve üzerinde gerçekleştirmiştir. Ancak YOLOv3 algoritmasının Faster-RCNN modeline göre daha sağlam ve daha yüksek hatırlamaya sahip olduğu sonucuna ulaşılmıştır [9].

Nassim ve arkadaşları insansız hava aracından alınan görüntülerden araba tespitini alandaki diğer algoritmalarla karşılaştırmak amacıyla yaptıkları çalışmada 4 önemli adım uygulamışlardır. Birinci adım olarak alınan görüntü de araba olup olmadığının tespiti için ortalama kaydırma algoritması kullanılarak görüntü farklı bölgelere ayrılmıştır. İkinci adımda ayrılan bölge önceden eğitilmiş evrişimsel sinir ağına girdi olarak verilerek özellik çıkarma işlemi yapılmış ve üçüncü adımda SVM algoritması kullanılarak sınıflandırma yapılmıştır. Son adımda ise tespit edilen nesneler üzerinde morfolojik işlemler ile ince ayarlar yapılmıştır. Uygulanan çalışmada elde edilen sonuçlar diğer başarılı algoritmalar ile karşılaştırıldığında daha yüksek doğruluk oranına ulaştığı görülmüştür [10].

Lan ve diğerleri insansız hava aracından alınan yaklaşık 6000 görüntüden oluşturdukları veri setini kullanarak Faster R-CNN mimarisi ile elektrik şebekelerindeki arızaları tespit etmeye yönelik çalışma yapmışlardır. Çalışmalarında Faster R-CNN mimarisi üzerinde değişiklikler yaparak VGG16 ve ResNet101 ağlarında denemeler yapmışlar ve sonuçları karşılaştırmışlardır. Elde edilen sonuçlara göre test değerleri ortalama doğruluk oranı VGG16 ağında %81,11 ve ResNet101 ağında %67,92 olarak tespit edilmiştir. Yani üzerinde değişiklik yaptıkları Faster R-CNN mimarisi elektrik şebekelerindeki hata ve arızaları tespit etmede başarılı sonuçlar üretmiştir [11].

Xu ve diğerleri özellikle insansız hava araçlarından alınan görüntülerde çok yönlü nesne tespitinde yatay sınırlayıcı kutu kullanmanın etkisini araştırdıkları ve kayan tepe önerisinde bulundukları çalışmalarında yatay sınırlayıcı kutu kullanmanın farklı yönlerde olan nesneler için dezavantajlarını belirtmişlerdir. Çalışmaya göre havadan alınan görüntülerde farklı yönlerde olan nesneler için belirli bir yatay sınırlayıcı kutu kullanmanın nesne değişikliği algılamada açıdan kaynaklı doğru yönlendirme ve ölçek bilgisi sağlamayacağı ve açı sapması nedeniyle düşük IoU değeri oluşturacağı, bunun da

tahmin doğruluğunu düşüreceği belirtilmektedir. Önerilen yöntemde yatay sınırlayıcı kutuda var olan koordinat bilgileri, genişlik, yükseklik (x, y, w, h) gibi özelliklere ek olarak eğiklik katsayısı ile belirlenen döndürülmüş kutu değerleri de ağın eğitimi sırasında kullanılmıştır. Çalışma sonucunda tek aşamalı nesne tespit modeli olan ResNet101 mimarisinde döndürülmüş yatay sınırlayıcı kutu kullanmanın diğer çalışmalara göre hem daha hızlı hem de ortalama doğruluk oranı olarak kullanılan veri setlerine göre %75,02 ile %88,2 arasında daha iyi sonuçlar ürettiği tespit edilmiştir [12].

Lee ve diğerleri insansız hava araçları için gerçek zamanlı ve bulut tabanlı nesne tanıma çalışmalarında insansız hava aracının aldığı görüntülerden gerçek zamanlı nesne tanıması yaptığı ve hesaplama işlem maliyetinin daha düşük olduğu bir öneride bulunmuşlardır. İnsansız hava araçlarının yapı olarak gerçek zamanlı nesne tanımasının önünde bazı dezavantajlar olduğunu vurgulamışlardır. Bunlar; insansız hava aracının hareketli olması nedeniyle aldığı görüntülerin gürültülü ve bulanık olması, yerleşik kameralar nedeniyle düşük çözünürlüğe sahip olması ve hedef nesnelerin genellikle küçük olmasıdır. Bunun yanında nesne tanıma çalışmalarında tek bir nesneyi algılamak dahi oldukça yüksek işlem gücü gerektirir ve son zamanlarda yapılan çalışmalar bu hesaplama maliyetini ekran kartları kullanılarak karşılamaktadır. İnsansız hava aracından alınan görüntülerin bulut servisler tarafından nesne tanımasının yapılması bir avantaj olarak görülse de muhtemel iletişim gecikmeleri yaşanabilmektedir. Bulut bilişim sistemleri neredeyse sınırsız hesaplama kaynaklarına isteğe bağlı erişim sağlamaktadır ve bu periyodik olarak büyük hesaplama gerektiren iş yükleri için yararlı olmaktadır. Yapılan çalışmada tüm işlerin bulut platformlarda yapılması tercih edilmemiş hafif bir nesne dedektörü çalıştırılarak ilgilenilen nesneyi içermesi muhtemel görüntüler ve bilgiler bulut platforma gönderilmiştir. Gerçek zamanlı nesne tespiti için kullanılan drone kamerası 320 x 240 piksel çözünürlüğe sahiptir. İşlem maliyetini azaltmak için bulut servislerde sadece Faster R-CNN algoritması kullanılarak nesne tespiti yapılmıştır. Algoritma nesne tespiti yapabilmesi için Pascal VOC2012 veri seti ile eğitilmiştir. Çalışmada elde edilen sonuçlara göre bulut serviste çalışan Faster R-CNN mimarisi %83,9 ortalama doğruluk olarak oranına ulaşırken; kullanılan SSD mimarilerinden SSD300 %81,6 ve SSD500 %82,6 ortalama doğruluk oranı elde etmişlerdir. Kullanılan diğer mimarilerden YOLO modeli %78,3 ve Fast YOLO modeli %79,4 ortalama

doğruluk oranına ulaşmıştır. Sonuç olarak çalışma ile işlem yükü bulut servislerce karşılanmasına rağmen daha doğru ve verimli bir mimari önerilmiştir [13].

Ophoff ve diğerleri tarafından yapılan farklı nesne algılama ağlarının uydu görüntülerinden araç ve gemi algılama ve sınıflandırma görevini inceleyen çalışmalarında yüksek performanslı tek atış dedektörü olan YOLO modeli kullanılmıştır. Uydu görüntülerinden oluşan veri setinde araç ve gemi algılama ve sınıflandırma deneysel çalışması ImageNet ve DOTA önceden eğitilmiş ağırlık verileriyle yapılmış ve elde edilen sonuçlara göre karşılaştırılan YOLO modelleri (YOLOv2, YOLOv3, D-YOLO ve YOLT) arasından hem çalışma zamanı hem de doğruluk olarak en uygun modelin D-YOLO olduğu tespit edilmiştir [14].

Mandal ve diğerleri alanda eksik olarak gördükleri insansız hava aracından alınan video görüntülerinden oluşturulan veri seti ve hareketli nesne algılama çalışmalarında; hareketli nesnelerin tespit edilmesi için 30 video görüntüsünden ve toplam 10.948 frame karesinden oluşan MOR-UAV adında yeni bir veri seti oluşturmuşlardır. Bu veri setini kullanarak hareketli nesneleri eşzamanlı olarak yerelleştirmek ve sınıflandırmak için MOR-UAVNet adını verdikleri derin öğrenme çerçevesini kullanarak yaptıkları deneysel çalışmalarda en az %50 IoU eşiği üzerinden %58,56 ortalama doğruluk oranı elde etmişlerdir [15].

Algancı ve diğerleri uydu görüntülerinden alınan fotoğraflardan uçak tanıma üzerine yaptıkları çalışmada; YOLOv3, SSD, Faster R-CNN modellerinin Pascal VOC ve COCO gibi çok perspektifli veri setlerinde elde ettikleri doğrulama başarısını düşük perspektif ve sınırlı veri kümesi üzerine uygulamışlardır. Çalışmada veri seti olarak bu tez çalışmasında da kullanılan DOTA veri seti kullanılmıştır. Üç evrişimsel sinir ağı modelinde kullanılan hiperparametreler optimize edilerek modellerin birkaç kez eğitimi sağlanmıştır. Eğitim sonunda modeller hem DOTA hem de başka bir uydu görüntüsü veri seti ile test edilmiştir. Elde edilen sonuçlara göre ortalama doğruluk oranı olarak Faster R-CNN, işlem süresi olarak da YOLOv3 diğer modellere göre ön plana çıkmaktadır. SSD modeli en kötü algılama performansı göstermesine karşılık nesne yerelleştirmesinde daha iyi sonuçlar üretmiştir [16].

Hinzmann ve diğerleri insansız hava araçlarından alınan optik ve kızıl ötesi görüntülerin kullanılarak derin öğrenme yöntemleriyle insan tespiti üzerine çalışma yapmışlardır. Çalışmada amaç evrişimsel sinir ağı mimarilerinde hem kızılötesi hem optik görüntülerin birleştirilerek kullanılması ve sonuçların izlenmesidir. Oluşturulan veri seti farklı RetinaNet ve YOLO modelleri ile çalıştırılarak sonuçları karşılaştırılmıştır. Ayrıca karşılaştırmalarda RetinaNet modelinin kullandığı odak kaybı optimizasyonuna karşılık YOLOv3 modelinin kullandığı çapraz entropi kaybı dikkate alınmıştır. Elde edilen sonuçlarda RetinaNet modelinin YOLOv3 modeline göre daha başarılı olduğu tespit edilmiştir. RetinaNet mimarisinin daha başarılı olmasının sebebi olarak ön plan ve arka plan arasındaki büyük dengesizlik durumuyla başa çıkabilmek için kullandığı odak kaybı fonksiyonu olduğu belirtilmiştir [17].

Carlet ve Abayowa VEDAI ve benzer hava fotoğraflarından oluşan veri setlerini kullanılarak YOLOv2 ile Faster R-CNN modellerini karşılaştıran bir çalışma yapmışlardır. Çalışmalarında ilk tercihleri olan Faster RCNN modelinin hızı ve doğruluğu nedeniyle 2015'teki başlangıcından bu yana yoğun bir şekilde kullanıldığı vurgulanmıştır. Hava fotoğraflarının kullanıldığı çalışma sonucunda YOLOv2 modelinin hız ve doğruluk açısından Faster R-CNN modeline göre yaklaşık 4 kat üstün olduğu sonucuna ulaşılmıştır [18].

Onim vd. tarafından YOLOv4 modeli kullanılarak Bangladeş'te trafikte bulunan araçların plakalarını tespit ederek karakter tanıma uygulaması geliştirilmiştir. Modelin eğitimi esnasında veri artırma, öğrenimin transferi (transfer learning) kullanılmış ve omurga olarak MobileNetv2 mimarisi seçilmiştir. Bangladeş trafiğinden elde edilen video görüntüsü kullanılarak fotoğraf veri seti oluşturulmuş ve model 6000 epok eğitim sonundan yaklaşık %90 ortalama doğruluk oranı başarısı göstermiştir. Elde edilen sonuçlar modelin hem hızlı hem başarılı olduğunu doğrulamıştır [19].

İnsansız hava araçlarından alınan görüntüler kullanılarak nesne tespiti için yapılan çalışmalar incelendiğinde; genel olarak hızı ile ön plana çıkan tek aşamalı nesne tespit dedektörleri ve doğruluk oranının yüksek olduğu görülen iki aşamalı nesne tespit dedektörleri karşılaştırılmaktadır. Modeller karşılaştırılırken genellikle ImageNet, Pascal VOC ve MS COCO olmak üzere iki ya da üç veri seti üzerinde çalışılarak sonuçlar değerlendirilmektedir.

Literatürde eksik olduğu değerlendirilen iki durum bu tez çalışmasının kapsamını oluşturmaktadır. İlk olarak hava fotoğraflarından nesne tanıma çalışmalarında tek bir veri seti kullanılarak yapılan değerlendirme yerine daha sağlam dayanak oluşturabilmek için her ikisi de insansız hava aracından elde edilen birisi nispeten daha çok nesne sınıfı barındıran diğeri ise daha çok görüntü barındıran iki farklı veri seti kullanılarak sonuçların değerlendirmesi yapılacaktır. Burada amaç nesne tanıma mimarilerinin hava fotoğrafları üzerindeki başarısını aynı amaca hizmet eden iki farklı veri seti kullanarak tespit edebilmektir.

Çalışmaya konu olacak ikinci amaç ise son dönemde geliştirilen ve henüz literatürde üzerinde yeterince çalışma yapılmamış YOLOv5 tek aşamalı nesne tespit dedektörünün YOLOv3, RetinaNet ve EfficienDet modelleri karşısındaki başarısını ve performansı değerlendirmektir. Çalışma sonunda YOLOv5 modeli ile tasarlanan bir nesne tespit uygulaması geliştirilmiş olacaktır.

BÖLÜM 3. MATERYAL VE YÖNTEM

Bilgisayarlı görme uygulamaları genel olarak yapay zekâ ve gelişimi bağlamında incelense de spesifik anlamda yapay zekanın derin öğrenme alt alanı içerisinde gelişmiştir. Kullanılan algoritmalar ve mimariler neredeyse tamamen derin öğrenme başlığı altında toplanmıştır. Özellikle son yıllarda dikkat çeken başarılı modeller evrişimsel sinir ağları kullanılarak geliştirilmektedir.

Bu nedenle tezin bu başlığı altında alana dair genel bilgilendirmeler yapılmıştır.

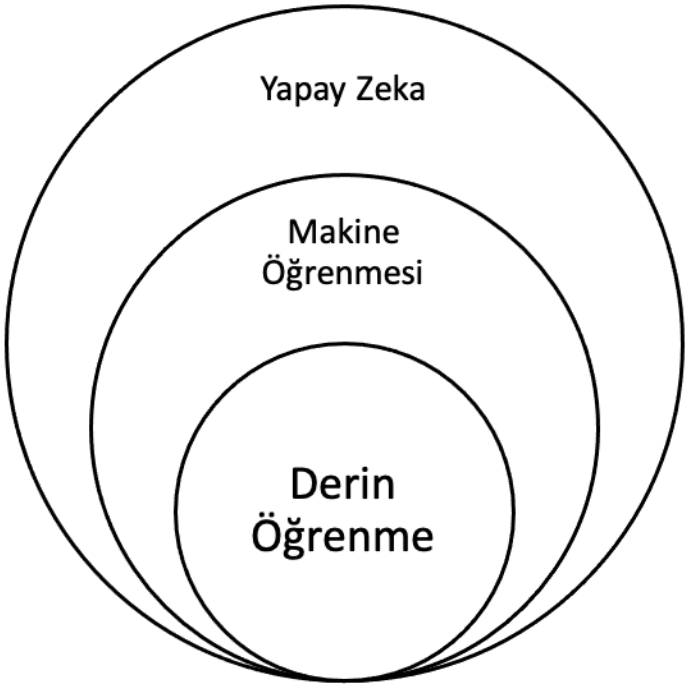
3.1. Yapay Zekâ

İnsan zekâsı bulunduğu ortamda doğru düzgün ve olan biteni öngörme ve duruma uygun hareket edebilme kabiliyetidir. Yapay zekâ ise makinelere zeki davranabilme kabiliyeti kazandırma çabasıdır. Tanıma göre insanlar, hayvanlar hatta makineler dahil olmak üzere birçok varlık zeki olarak nitelendirilmektedir. Akıllı kamera kullanan makineler ya da hayvanlar zekâ kademelerine göre ilkel zekâ basamağında yer almaktadır. İnsanlar ise gelişmiş zekâ kabiliyetleri ile mantık yürütebilen, koyduğu hedeflere ulaşabilen, bir dili anlayıp cümle kurabilen, duyusal girdileri ile algılayıp tepki verebilen canlılardır. Yine matematik teoremlerini ispatlayabilmek, zorlu oyunlar oynamak, bir bilgiyi sentezleyip özetleyebilmek, sanat ya da müzik eserleri oluşturabilmek insanoğlunun zekâsı ile gerçekleştirebildiği ona özgü yeteneklerdir [20].

Bilgisayar biliminin atası olarak kabul edilen matematikçi ve bilim adamı Alan Turing düşünebilen makineler fikrini ortaya attığı 1950'li yıllardan beri dünyanın çeşitli ülkelerinde insana ait zekâ becerisi ve onu kullanarak yapabildiği görme, konuşma, anlama gibi davranışlarını makinelerin de yapabilmesi çalışmaları yapay zekâ olarak adlandırılmıştır. Bu genel başlık altında makine öğrenmesi ve derin öğrenme özel başlıkları ise bilgisayarlı görme gibi önemli alanları içermektedir.

İnsan beyni matematiksel bir işlemi zorluk derecesine göre birkaç dakikada ya da daha uzun bir sürede yapabilirken; görme, konuşma, anlama, yorumlama, kavrama gibi olayları çok kısa bir sürede gerçekleştirebilir. Bunun yanı sıra edindiği tecrübelerden faydalanarak yeni durumlar karşısında davranış tarzını belirleyebilir. Örneğin sıcak bir cisme elini değdirdiğinde elinin yandığını tecrübe ederse daha sonraki davranışlarına bu

bilgiyi yansıtabilir. Bilgisayarlar ise çok daha karışık ve üst düzey matematiksel hesaplamaları insanlara göre çok kısa sürede yapabilme kapasitelerine sahiptirler. Ancak insanlara ait görece daha kolay olan görme, konuşma vs. gibi becerilere sahip değillerdir. İşte yapay zekâ olarak adlandırılan çalışmalar insanların tecrübe edinerek otomatikleştirdiği bu davranışları bilgisayarların da yapabilme becerisi kazanmasıdır. Yapay zekâ tüm bu alanda yapılan çalışmalara verilen genel bir isimlendirme olmuştur. Şekil 3.1'de yapay zekâ ve alt alanları genelden özele doğru hiyerarşik bir şekilde gösterilmektedir. Makinelere düşünebilme kabiliyeti kazandırma çalışmaları genellikle yapay zekâ gelişmeleri olarak adlandırılmaktadır. Ancak makine öğrenmesi çalışmaları ve algoritmaları daha alt alanı kapsamaktadır. Makine öğrenmesinin en dikkat çekici uygulamalarını içeren evrişimsel sinir ağları ise derin öğrenme alt alanında yer alan çalışmaları oluşturur.



Şekil 3.1. Yapay zekâ ve alt alanları hiyerarşisi gösterimi

3.2. Makine Öğrenmesi

Bilgisayar bilimlerinde yapay zekanın öğrenme ve tanıma görevlerini yerine getirebilmesi için geliştirilen uygulamalar makine öğrenmesi olarak adlandırılmaktadır. Makine öğrenmesi; bilgisayarın öğrenmesi ve sonrasında da tahminde bulunabilmesi için en uygun algoritmayı oluşturmayı amaçlamaktadır. Bu algoritmalar klasik bilgisayar programlaması için oluşturulan algoritmalar gibi sıralı talimatları uygulamak yerine eğitim setinde karşılaştığı örneklerden öğrenip genelleme yaparak öğrendiği bilgiyi yeni durumlara uyarlayabilmektedir. Bu şekilde algoritmaların geliştirilerek bilgisayarlara öğrenme işinin kazandırılması makine öğrenmesi olarak bilinmektedir [21].

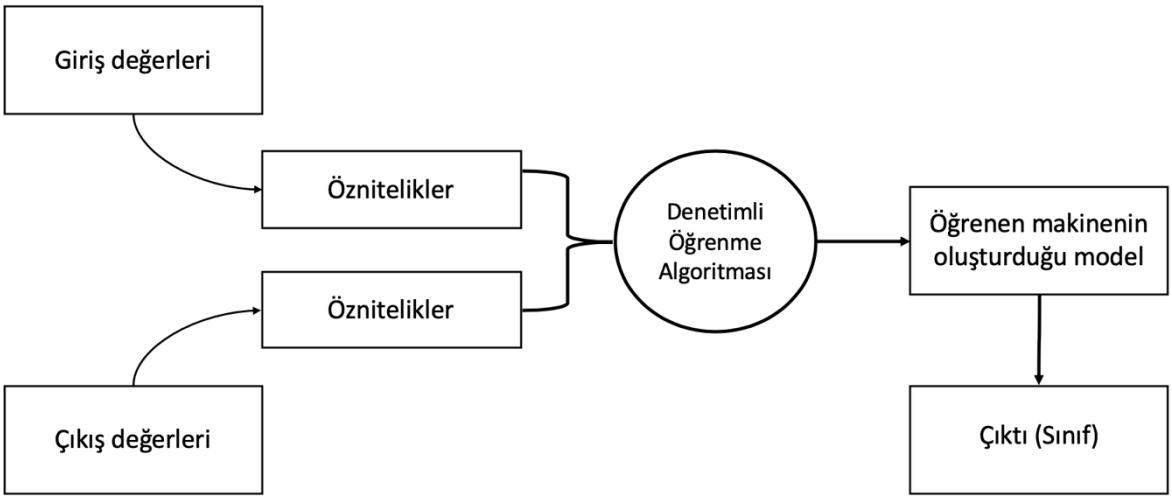
Klasik programlamada kurallar ve veriler makineye verilir ve cevaplar istenirken makine öğrenmesinde cevaplar ve veriler makineye verilir ve kurallar makineden istenir. Yani makinenin cevabını bildiği örüntülerden bir kural çıkarması beklenir.

3.2.1. Denetimli Öğrenme

Denetimli öğrenme tıpkı insan öğrenmesi gibi makineye sonucu belirli olan problem durumunu deneyimleyip bilinen sonuca götüren yolu öğrenmesini sağlamaktır. Makinenin öğrenip deneyimlemesi için kullanılacak veri setinde girdiler ve elde edilecek çıktılar belirlidir. Makineden istenen bu girdileri kullanarak çıktılara götürecek yolu öğrenmesidir. Örneğin; özellikle yapay zekâ alanında en çok kullanılan verilerden birisi olan elektronik postaların spam ya da spam değil olarak belirlenmesi probleminde ilk olarak hangi postanın spam olduğu hangi postanın spam olmadığı bilinmektedir. Makinenin burada yapması gereken girdilerle verilmiş olan öznitelikleri kullanarak çıktılara götüren yolun örüntüsünü çıkarabilmesidir. Ancak dikkat edilmesi gereken nokta sisteme hem giriş değerleri hem de çıkış değerleri verilir. Burada bir insanın bir öğretmen gözetiminde öğrenmesi deneyiminden yola çıkılmıştır.

Makine öğrenmesinde denetimli (gözetimli) öğrenme süreci genellikle iki problem durumunda kullanılır. Eldeki veri seti çıktı olarak sayısal değerden oluşuyorsa yani makine sayısal bir çıktı değeri öğrenmeye çalışıyorsa tahmin çalışması yapılmaktadır. Ancak makine çıktı olarak kategorik değerleri öğreniyorsa sınıflandırma çalışması yapılmaktadır. Şekil 3.2 denetimli öğrenme sürecindeki genel işleyişi görselleştirmektedir. Öznitelik olarak giriş ve çıkış değerleri algoritmaya sunulur. Çıkış

değerini bilen algoritma örüntüyü öğrenerek tahminde bulunur ve tahmininin doğru olup olmadığına göre değerlendirilip geliştirilir.



Şekil 3.2. Denetimli makine öğrenmesi süreci

3.2.2. Denetimsiz Öğrenme

Denetimsiz makine öğrenmesi algoritmasında makinenin ya da bilgisayarın öğrenme sürecini takip eden ve denetleyen bir sistem yoktur. Makineye sadece giriş verileri sunulur. Çıkış verileri ise etiketsiz olarak makine tarafından etiketlenecek şekilde sunulur. Bu öğrenme algoritmasına göre giriş özniteliklerini kullanarak bilgisayarın öğrenme yolunda bir örüntü geliştirmesi ve sonucunda bu örüntüye uygun olacak şekilde bir çıktı üretmesi beklenir. Burada genel olarak kategorize edilmeyen bir veri seti sunularak verilerin anlamlı bir bütün şeklinde kategorize edilmiş çıktısı alınır. Bu işleme kümeleme adı verilir.

3.3. Derin Öğrenme

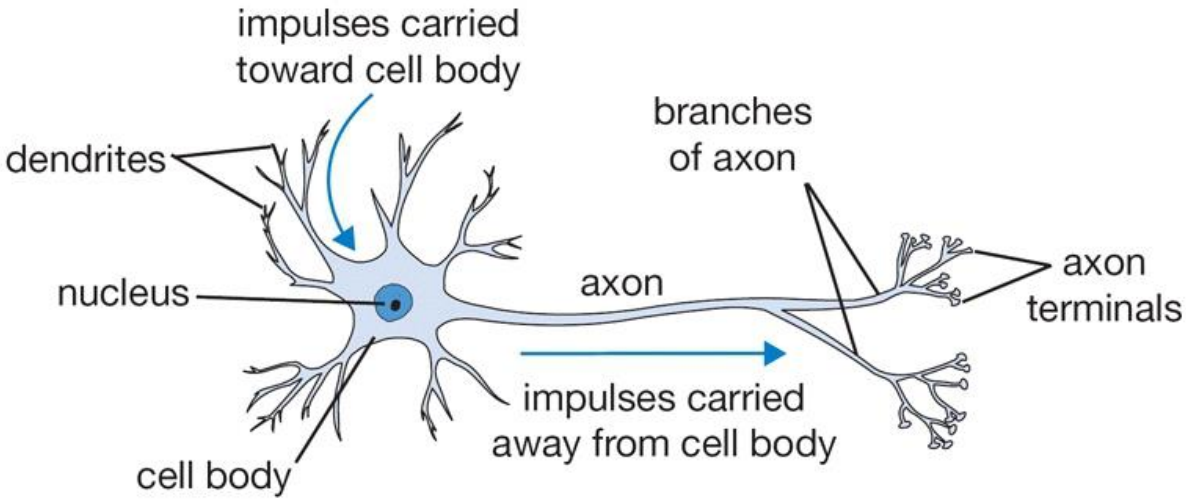
Yapay zekâ fikrinin ortaya çıktığı ilk zamanlarda insanlar açısından güçlükle çözülebilen ve zaman alan matematiksel hesaplama gerektiren problemler bilgisayarlar tarafından çok hızlı bir şekilde çözüme kavuşturuldu. Ancak asıl sorun yüz tanıma, nesne algılama ya da takip etme, konuşma tanıma gibi insanların kolay bir şekilde ve sürekli yapabildiği hatta otomatikleştirdiği davranışları bilgisayarlar tarafından oldukça büyük bir problem olmasıdır. Bilgisayarın kavramlar hiyerarşisi mantığı karmaşık kavramları

daha basit kavramlardan yola çıkarak örüntü kurma yoluyla bulabilmesini sağlar. Bu kavramların birbiri üzerine inşa edilerek birden fazla katmandan oluşan bir mimari inşa edilmesi ile derin öğrenme adı verilen yaklaşım ortaya çıkmıştır [22].

Bilgisayarlı görme çalışmaları derin öğrenme tabanlı mimariler ve algoritmalar üzerinden geliştirilmektedir. Dolayısıyla bu tez çalışmasında yapay zekâ ve makine öğrenmesinden ziyade ağırlıklı olarak derin öğrenme başlığı incelenmiştir.

3.3.1. Yapay Sinir Ağları

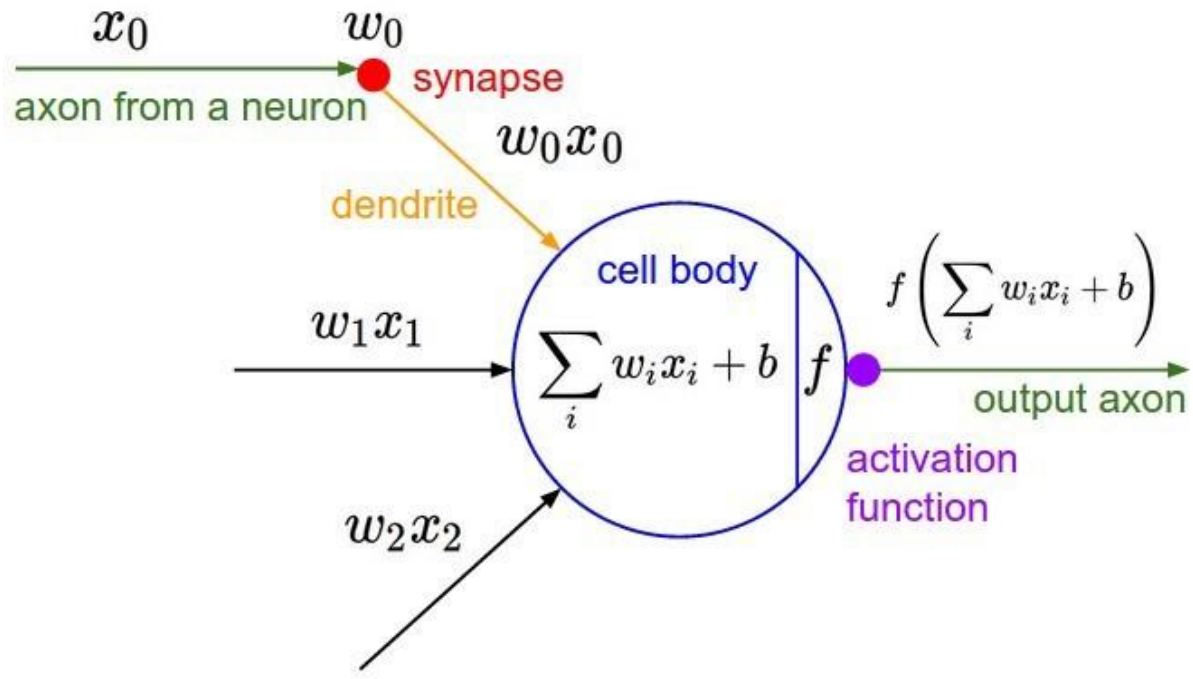
Yapay zekanın en temel amacını oluşturan insan beynini ve çalışma mantığını bilgisayarlara aktarabilme fikri insan sinir sistemini taklit ederek gerçekleşmektedir. Bu nedenle yapay sinir ağları insan sinir sistemi ile birlikte incelenerek anlaşılabilir. Şekil 3.3 bir insana ait sinir hücresinin yapısını göstermektedir. İnsan sinir hücresi giriş verilerini alan dendritlerden ve elektriksel işlem sonucu çıkış verisini üreten akson adı verilen yapılardan oluşmaktadır. Diğer sinir hücrelerinin aksonlarından gelen veriler dendrit vasıtasıyla alınır ve işlenerek elde edilen çıktı yine akson vasıtasıyla iletilir. Bu şekilde milyonlarca sinir hücresinin birbirine bağlı olarak çalışması sonucu görme, konuşma, anlama, yorumlama, kavrama gibi davranışlar yerine getirilebilmektedir.



Şekil 3.3. İnsan sinir hücresinin yapısı [23].

Yapay sinir ağları kavramın ilk olarak bu alanda çalışmalar yapan Mc Cullogh ve Pitts tarafından 1943 yılında yayınlanan bir makale ile ortaya atılmıştır. Ancak o dönem teknolojinin ve şartların yeterince gelişmemiş olması nedeniyle çok fazla ilgi görmemiştir. 1954 yılında Farley ve Clark tarafından bir ağ içerisinde uyarılara tepki veren model geliştirilmiştir. 1960 yılına gelindiğinde bu alanda kullanılabilecek bilgisayarlar geliştirilmeye çalışılmıştır. Belirli bir dönem yine ilgi görmeyen alanda 1980 yılına gelindiğinde zor problemlerin çözümü için yapay sinir ağlarını genelleştirme sayesinde çözüm üretilebileceği fikri ortaya atılmıştır. 1980'li yılların sonlarına doğru yapay sinir ağları daha fazla ilgi görmeye başlamıştır [24].

Şekil 3.4 insan sinir sisteminin model alınarak bilgisayar ya da makineler için zekâ sistemi tasarımında kullanılan yapay sinir ağını göstermektedir. Başka bir sinir hücresinden aldığı giriş verisini (x_0) kullanarak işlemler yapan ve elde ettiği çıktıyı yine akson benzeri bir yapı ile diğer nörona ya da katmana aktaran bir işleyişi vardır. Derin öğrenme mimarisinde yapay sinir ağları bu mantık üzerine çalışmaktadır.



Şekil 3.4. Yapay Sinir Hücresi [23].

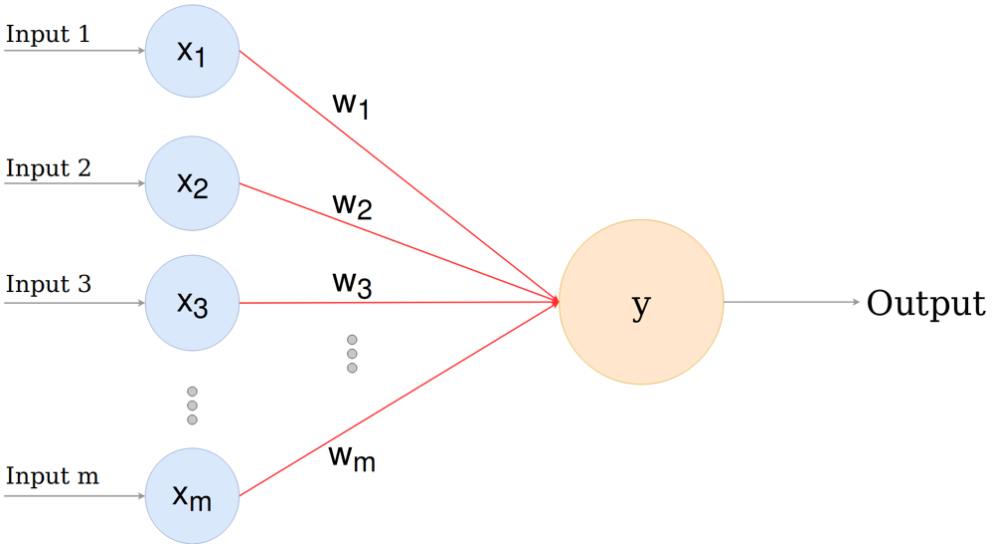
Bir yapay sinir ağı yapısı genel olarak giriş katmanı (girdi değerleri ya da öznitelikler), gizli katmanlar ve çıktı katmanı olmak üzere üç ana bölümden oluşur. Girdi katmanında başka bir sinir ağından gelen çıktı değerleri ya da işleme tabi tutulacak nesneye ait öznitelikler bulunur. Gizli katmanlar bu öznitelikleri ve belirli ağırlık değerlerini kullanarak özellik haritası çıkarırlar. Bu özellikler aktivasyon fonksiyonları ile çıktı katmanında anlamlı bir sonuç üretilmesini sağlarlar.

3.3.2. Tek Katmanlı Algılayıcılar

Tek katmanlı yapay sinir ağı yapısı girdi katmanı, hesaplama ve çıktı katmanından oluşmaktadır. Bu yapıda giriş değerleri belirlenmiş ağırlıklar ile çarpılarak genel bir toplama işlemine tabi tutulur ve çıktı katmanına iletilir. Elde edilen sonuç genellikle iki durum üretir. Yani basit sınıflandırma çalışmaları için kullanılabilir. Denklem 3.1 bir yapay sinir ağının yaptığı matematiksel işlemin formülünü göstermektedir. Yapay sinir ağı denklemde x olarak gösterilen girdi öznitelikleri ile denklemde w ile gösterilen ağırlık değerlerini çarpma işlemine tabi tutar ve hesaplanan değer denklemde b ile ifade edilen yanlılık değeri ile toplanır. Bulunan sonuç aktivasyon fonksiyonunda işleme tabi tutulduktan sonra elde edilen değer sıfırdan büyük ise 1, değil ise 0 sonucunu üretir. Sonucun sadece iki durum olması bu sinir ağı yapısının sınıflandırma çalışmalarında kullanılabildiğini göstermektedir.

$$f(x) = \begin{cases} 1, & w.x + b > 0 \; ise \\ 0, & değilse \end{cases} \tag{3.1}$$

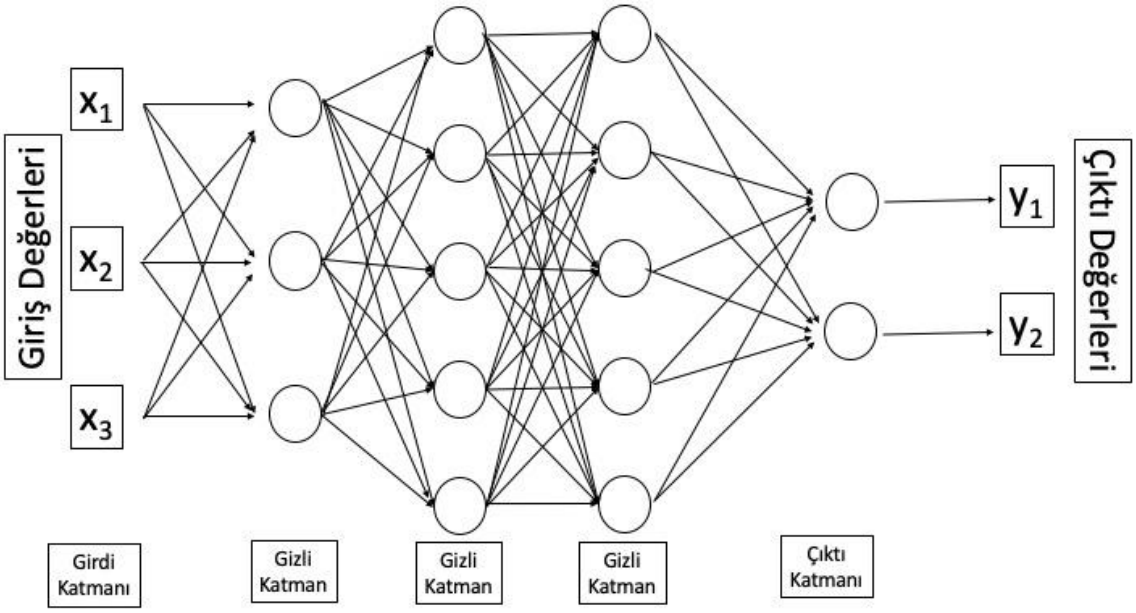
Şekil 3.5'te tek katmanlı bir yapay sinir ağı gösterilmektedir. Denklem 3.2'de gösterilen formüle uygun olarak x_1, x_2,….,x_m şeklinde gösterilen her girdi değeri w_1, w_2,…., w_m şeklinde ifade edilen ağırlık değerleriyle işleme tabi tutulur ve bir çıktı değeri üretilir.



Şekil 3.5. Tek katmanlı yapay sinir ağı yapısı [23].

3.3.3. Çok Katmanlı Algılayıcılar

Tek katmanlı yapay sinir ağları üzerine geliştirmeler yapılarak daha derin ağlarla daha karışık problemlerin çözülebilmesi için çok katmanlı algılayıcılar geliştirilmiştir. Girdi ve çıktı katmanları arasında gizli katmanların bulunduğu sinir ağlarıdır. Gizli katmanlardaki nöronların görevi girdi değerlerinin yani özniteliklerin, ağırlık değerleriyle çarpılması sonucu elde edilen matematiksel ifadenin aktivasyon fonksiyonuna tabi tutularak sonraki katman için girdi değeri olarak oluşmasını sağlamaktır. Şekil 3.6 çok katmanlı bir yapay sinir ağının yapısını göstermektedir. Tek katmanlı yapay sinir ağında yapılan işlemin daha fazla katman kullanarak daha derin ağ oluşturulması sağlanmaktadır. Girdi katmanında bulunan her girdi değeri ilk gizli katmanda bulunan tüm nöronlarla işleme tabi tutulur. Bu katmanda elde edilen sonuçların hepsi bir sonraki katmanda bulunan tüm nöronlarla işleme tabi tutulur. Çok katmanlı yapay sinir ağında kaç adet gizli katman varsa işlem o kadar tekrarlanarak ilerler. Çıktı katmanında ise giriş değerlerine uygun olarak çıktı değerleri üretilir.



Şekil 3.6. Çok Katmanlı Yapay Sinir Ağı Yapısı

3.3.4. İleri Yönde Yayılım

Bilgisayarlı görüde gri tabanlı ya da renkli fotoğraf karesinden nesne tanıyabilmek için girdi görüntüsüne ait piksel değerlerinin matematiksel olarak ifade edilmesi gerekir. Elde edilen öznitelikler ister rastgele olsun ister başka bir derin öğrenme uygulamasından transfer edilen ağırlık değerleriyle olsun hesaplamaya tabi tutulur. Gizli katmanlarda elde edilen sonuçlar bir sonraki katmana iletilerek ileri yönlü bir hesaplama yapılmaktadır. Hesaplamalar sonucu elde edilen değerlerin doğrusal olmaması ve bir anlam ifade edebilmesi için aktivasyon fonksiyonlarıyla yeniden hesaplaması yapılır. Aktivasyon fonksiyonlarıyla yapılan hesaplama sonucunda elde edilen negatif piksel değerleri önemsiz olarak kabul edilir ve sıfıra eşitlenir. Bu şekilde gizli katmanların yaptığı hesaplamaları bir sonraki katmana aktarması ileri yönde yayılım olarak adlandırılmaktadır. Bir yapay sinir ağının gizli katmanlarının sayısının artırılarak hesaplama kapasitesinin derinleştirilmesi sonucunda derin ağlar oluşturulmaktadır.

Yapay sinir ağlarında katmanların sayılması gizli katmanlardan başlayarak yapılır. Yani girdi katmanlı yapay sinir ağının derinliği hesap edilirken hesaplamaya dahil edilmez. Yapay sinir ağının derinliğini gizli katmanların ve çıktı katmanlarının sayısı

belirler. Denklem 3.2 çok katmanlı bir yapay sinir ağının ilk katmanında yapılan işlemi formülize etmektedir. Denklemde ilk olarak hesaplanan $A^{[0]}$ değeri girdi değerlerini ifade etmektedir. Denklem 3.3'te bir önceki hesaplamadan elde edilen $A^{[1]}$ değeri kullanılarak bir sonraki gizli katmana ait matematiksel işlemler yerine getirilir.

$$Z^{[1]} = W^{[1]} * A^{[0]} + b^{[1]}$$
$$A^{[1]} = G^{[1]}(Z^{[1]}) \tag{3.2}$$

$$Z^{[2]} = W^{[2]} * A^{[1]} + b^{[2]}$$
$$A^{[2]} = G^{[2]}(Z^{[2]}) \tag{3.3}$$

3.3.5. Maliyet Fonksiyonun Hesaplanması

Çok katmanlı yapay sinir ağlarında girdiye ait özellikler ileri yayılımın ilk adımında rastgele ya da belirli bir kurala göre seçilen ağırlık değerleriyle matematiksel olarak hesaplanır. İlk katmanın yaptığı hesaplama aktivasyon fonksiyonundan geçtikten sonra diğer katmana girdi olarak aktarılır ve ileri yayılım bu şekilde devam ederek son katmanda bir çıktı üretir. Üretilen bu tahmin değerinin gerçek değere olan uzaklığının belirlenmesi ve tahmin başarısının ölçülmesi gerekir. Burada modele en iyi tahmini yaptıracak olan parametrelerin değerleri bulunmaya çalışılır. Maliyet fonksiyonu tahmin edilen değerler ile gerçek değerler hatayı nicel olarak hesaplar ve bunu bir gerçek sayı değeri ile ifade eder. Soruna bağlı olarak maliyet fonksiyonu birçok farklı şekilde oluşturulmaktadır. Minimize edilmiş maliyet fonksiyonunun amacı mümkün olduğu kadar küçük bir sayı döndürecek model parametrelerini bulmaktır. Burada döndürülen değere maliyet, kayıp ya da hata fonksiyonu denmektedir. Maksimize edilmiş maliyet fonksiyonunun amacı ise mümkün olduğu kadar büyük bir sayı döndürecek model parametrelerini bulmaktır. Burada döndürülen değere ise ödül adı verilmektedir [25].

Denklem 3.4'te ileri yayılım için n katmanlı bir yapay sinir ağında n. katmanda yapılacak hesaplama formülü gösterilmiştir. Bu hesaplama sonucunda elde edilen değer

$\hat{y}$ olarak ifade edilen tahmin değeridir. Bu tahmin değerinin gerçek değere olan uzaklığı maliyet ya da kayıp adı verilen fonksiyon ile hesaplanmaktadır.

$$Z^{[n]} = W^{[n]} * A^{[n-1]} + b^{[n]}$$
$$A^{[n]} = G^{[n]}(Z^{[n]}) \tag{3.4}$$

Maliyet fonksiyonu hesaplanırken iki farklı yöntem benimsenmiştir. Bu hesaplama yöntemlerinden biri ortalama mutlak hata, diğeri ise ortalama kare hatasıdır. Ortalama mutlak hata yönlerini dikkate almadan tahmin grubundaki hataların ortalama büyüklüğünü ölçer. Burada tüm bireysel sapmaların daha önemli olduğu varsayılarak tahminler ve beklenen sonuçlar arasındaki mutlak farklılıkların ortalaması hesaplanır. Denklem 3.5 ortalama mutlak hata formülünü göstermektedir. Formülde m ifadesi tahmin edilen örnek sayısını göstermektedir.

$$\text{Ortalama Mutlak Hata} = \frac{1}{m}\sum_{i=1}^{m} |\hat{y}^{(i)} - y^{(i)}| \tag{3.5}$$

Bir diğer maliyet fonksiyonu hesaplama yöntemi ortalama kare hatasında ise; tahmin edilen ve beklenen sonuçlar arasındaki ortalama kare hataları hesaplanır. Bu hata türünde farklılıkların mutlak değerinin alınması yerine kareleri alınır. Denklem 3.6 ortalama kare hatası formülünü göstermektedir.

$$\text{Ortalama Kare Hatası} = \frac{1}{2m}\sum_{i=1}^{m} (\hat{y}^{(i)} - y^{(i)})^2 \tag{3.6}$$

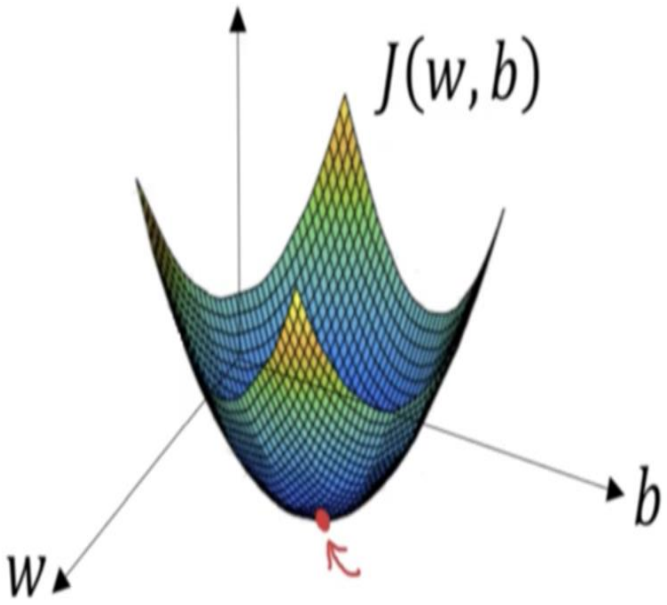
Derin ağlarda maliyet fonksiyonu hesaplaması her katman ve her nörondaki ağırlık değerleri (w) ile yanlılık değerleri (b) için yapılmaktadır. Denklem 3.7'de girdi özelliklerinin ağırlık ve yanlılık değerleriyle elde edilen çıktısı aktivasyon fonksiyonuyla

doğrusal olmaktan çıkarılır. Daha sonra denklem 3.8'de maliyet fonksiyonunun hesaplaması yapılır. Diğer denklemlerde kullanılan yöntemlerden farklı olarak kayıp fonksiyonu üzerinden maliyet fonksiyonu hesaplamasında tahmin edilen ve beklenen değerlerin kayıp fonksiyonu sonuçları kullanılmaktadır. Denklemin tamamı maliyet fonksiyonunu hesaplamaktadır. Burada maliyet fonksiyonunu mümkün olduğunca küçük yapacak ağırlık ve yanlılık değerleri bulunmaya çalışılmaktadır. Bunun için dereceli alçalma yöntemi kullanılmaktadır [26].

$$\hat{y}=\sigma(w^{T}x+b),\ \sigma(z)=\frac{1}{1+e^{-z}} \tag{3.7}$$

$$J(w,b)=\frac{1}{m}\sum_{i=1}^{m}\left(L\left(\hat{y}^{(i)},y^{(i)}\right)=-\frac{1}{m}\sum_{i=1}^{m}y^{(i)}log\hat{y}^{(i)}+(1-y^{(i)})\log(1-\hat{y}^{(i)})\right. \tag{3.8}$$

Şekil 3.7'de dış bükey olarak gösterilen dereceli alçalma eğrisi maliyet fonksiyonu için ileri yönde yayılımın ilk adımlarında grafiğin tepesi olarak görünür. Amaç en uygun ağırlık ve yanlılık değerlerini bularak maliyet fonksiyonunun grafiğin tabanına doğru inmesini sağlamaktır. Şekilde gösterilen grafik tabanı maliyet fonksiyonunu en düşük yapan ağırlık ve yanlılık değerleridir.



Şekil 3.7. Dereceli alçalma fonksiyonu [27].

3.3.6. Geri Yönde Yayılım

Geri yönde yayılım yapay sinir ağlarının eğitiminin özünü teşkil eder. Bir yapay sinir ağının ağırlıklarının önceki dönemde elde edilen hata oranına göre ince ayarının yapılarak iyileştirilmesi gerekir. Ağırlıkların doğru şekilde ayarlanması daha düşük hata oranları sağlar ve modeli güvenilir kılar. Geri yönde yayılım sürecinde her birim ve düğümdeki kaybın bulunması gerekir. Çünkü derin öğrenme modelinin ulaştığı her kayıp tüm düğümlerin sebep olduğu ve toplanarak elde edilen bir sayıya karşılık gelir. Hangi düğümün daha çok kayba neden olduğunu bulmak daha küçük ağırlık değeri vererek modeli daha etkin şekilde optimize etmeyi sağlar [28].

Derin öğrenme mimarisinde girdi vektörü olan x kullanılarak ileri beslemeli bir sinir ağı oluşturulduğunda çıktı değeri olan $\hat{y}$ elde edilmesi için bilgi ağ içerisinde ileri doğru akar. Mimariye ilk bilgiyi sağlayan girdi değeridir. Daha sonra bu bilgi bütün gizli katmanlardaki gizli birimlere yayılır ve çıktı üretilir. İşte bu ileri yönde yayılım aşaması matematiksel bir maliyet değeri üretilene kadar devam eder. Maliyet fonksiyonunun elde edilmesi ile geri yayılım adımı başlar ve bilgi tam ters yönde geriye doğru akar ve eğim hesaplanır. Geri yayılım algoritması sinir ağlarında tüm ağın öğrenme algoritması olarak

kullanılmaz. Aslında geri yönde yayılım ifadesi eğim hesaplama metoduna verilen bir isimlendirmedir. Bu hesaplama ile elde edilen eğimi kullanan olasılıksal eğim inişi gibi algoritmalar öğrenmeyi gerçekleştirmektedir [29].

Eğim inişi hesaplama yapılarak maliyet fonksiyonu hesaplaması ve daha sonrasında geri yönde yayılım ile ağırlık ve yanlılık değerlerinin güncellenmesi için Denklem 3.9'daki formüller kullanılmaktadır. Bu aşamada ilk olarak ayarlanan ağırlık değerinden öğrenme oranı ile maliyet fonksiyonunun ağırlık değerine göre türevinin çarpımı çıkarılır. Aynı hesaplama yanlılık değeri için de yapılır. Bu şekilde yeni ağırlık ve yanlılık değerleri elde edilerek sonraki ileri yönde yayılım aşamasına geçilmiş olur.

$$w = w - \alpha \cdot \frac{\partial J(w,b)}{\partial w}$$
$$b = b - \alpha \cdot \frac{\partial J(w,b)}{\partial b} \tag{3.9}$$

3.3.7. Hiperparametreler

Çok katmanlı yapay sinir ağ mimarisi tasarımında modelin veriden en etkili şekilde öğrenebilmesi için tasarımcı tarafından kullanılan model ve tekniğe uygun olarak bazı parametrelerin de en uygun olacak şekilde belirlenmesi gerekir. Bir sınıflandırma probleminde en yakın komşu algoritması kullanılıyorsa en yakın komşu sayısına ya da destek vektör makineleri algoritması kullanılıyorsa hangi çekirdek fonksiyonunun kullanılacağına tasarlayan tarafından karar verilmesi gerekir. Aynı şekilde çok katmanlı yapay sinir ağlarında; seyreltme değeri, katman sayısı, nöron sayısı, aktivasyon fonksiyonları, öğrenme oranı gibi parametreler hiperparametre olarak adlandırılmaktadır. Hiperparametrelerin özelliği tasarımcı tarafından belirlenen, problem durumuna ve veri setine göre değişen değerler olmasıdır [30].

Bu bölümde tez çalışmalarında kullanılan evrişimsel sinir ağı modellerine ait hiperparametrelerden bahsedilmiştir.

3.3.7.1. Öğrenme Hızı

Öğrenme hızı modelin her geri yayılım sonucunda ağırlık ve yanlılık değerlerinin güncelleme hızı olarak tanımlanmaktadır. Zira geri yayılım sonucunda zincirleme türev yöntemiyle ağırlık ve yanlılık yeni değerleri Denklem 3.9'da da gösterildiği gibi ilk değerlerinden, türev alınan ve öğrenme hızı ile çarpılan değerlerin çıkarılması sonucu bulunmaktadır. Öğrenme hızı mevcut ağırlıklara kayıp eğiminin ne kadarının uygulanacağını belirleyen hiperparametredir. Öğrenme hızının yüksek bir değer olarak belirlenmesi ağırlıkların yeni değerlerinin daha hızlı değişmesine sebep olurken düşük bir değer olarak belirlenmesi ise optimum eğim noktasına gidiş süresini uzatır. Genellikle derin sinir ağı modellerinde öğrenme hızı değerleri 0,1 ile 0,001 olarak belirlenmektedir [31].

3.3.7.2. Aktivasyon Fonksiyonları

Yapay sinir ağlarında yapılan işlemler sonucu elde edilen değerlerin anlamlı bir aralıkta olması gerekmektedir. Girdi değerleriyle çarpılacak ağırlık değerleri literatürde ve yapılan çalışmalarda genel olarak ilk başta rastgele değerler olarak belirlenir. İleri yayılım ve geri yayılım işlemi sonucunda ağırlık değerleri güncellenerek hangi nöronun sistem için daha önemli olduğu belirlenir. Ancak yapılan bu matematiksel işlemin bir anlam ifade etmesi için belirli bir aralıkta kalması gerekmektedir. Yani anlamsız derecede çok küçük ya da çok büyük değerler çıktı olarak kullanıldıklarında hatalı sonuç üretirler. Bu sebeple aktivasyon fonksiyonları kullanılarak bu değerlerin belirli aralıkta kalması sağlanmaktadır.

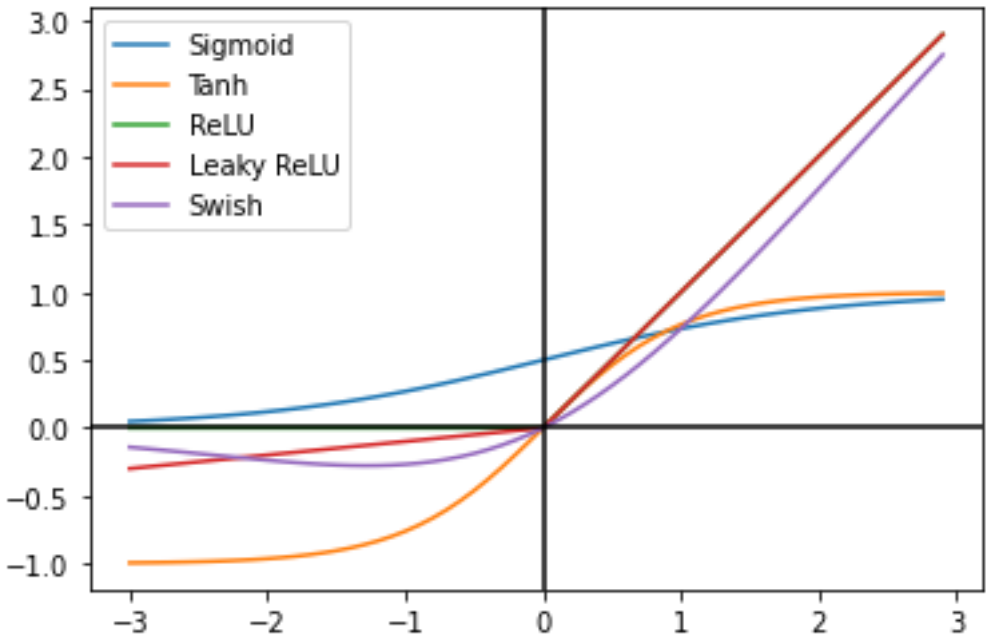
Aktivasyon fonksiyonlarının yapay sinir ağı içerisinde en temel görevi elde edilen çıktı değerinin doğrusal olmasını önlemektir. Kullanılan sinir ağı yapısına göre elde edilen çıktı değerleri sadece sınıflandırma yapmak için değil daha derin yapılar içeren doğrusal olmayan değerler üretecektir.

Bu bölümde tez çalışmaları esnasında kullanılan evrişimsel sinir ağı modellerine ait aktivasyon fonksiyonlarına değinilmiştir. Tablo 3.1'de aynı zamanda evrişimsel sinir ağlarında da en çok kullanılan aktivasyon fonksiyonlarına ait denklemler ve değer aralıkları birlikte gösterilmiştir.

Tablo 3.1. Aktivasyon fonksiyonları [32].

Foksiyon Adı	Fonksiyon Denklemi	Değer Aralığı
Sigmoid	$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$	(0,1)
ReLU	$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}$	[0,∞)
Leaky ReLU	$f(x) = \begin{cases} 0.01, & x < 0 \\ x, & x \geq 0 \end{cases}$	(-∞, ∞)
Swish	$f(x) = 2x\sigma(\beta x) = \begin{cases} \beta = 0, & f(x) = x, \\ \beta \to \infty, & f(x) = 2max(0, x) \end{cases}$	(-∞, ∞)

Şekil 3.8'de ise denklemleri ve değer aralıkları verilmiş olan aktivasyon fonksiyonlarının koordinat sistemi üzerindeki grafiği gösterilmiştir.

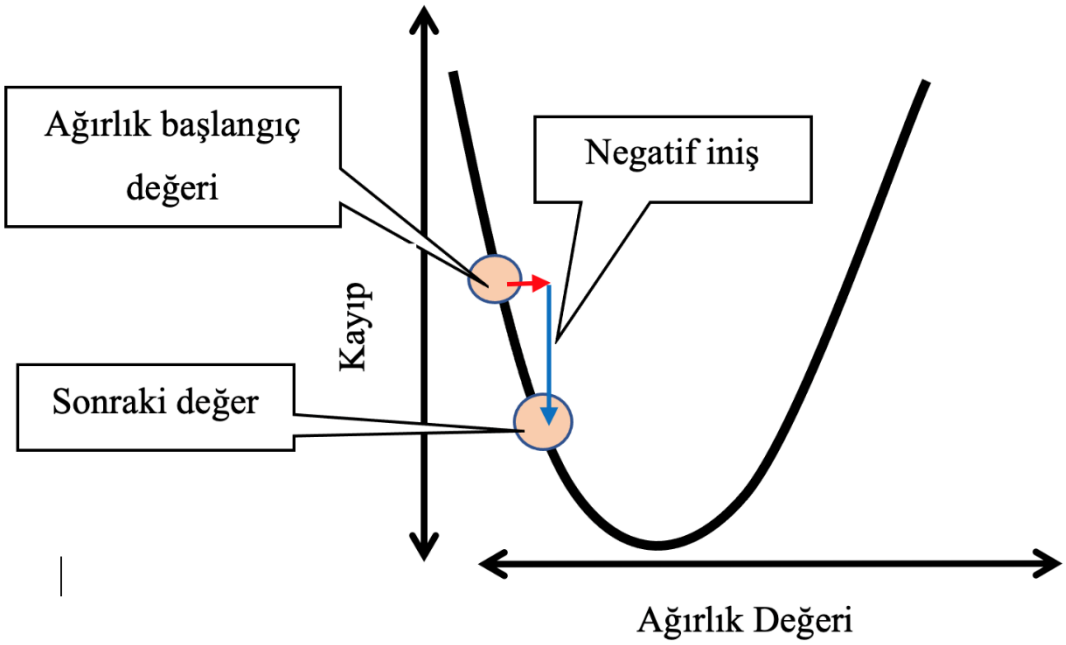


Şekil 3.8. Aktivasyon fonksiyonları.

3.3.7.3. Optimizasyon ve En İyileme

Derin öğrenme mimarisinde çok katmanlı yapay sinir ağının amacı eğitimde kullanılan modelin iyi bir genelleme elde etmesidir. Daha sonra bu genellemeyi doğrulama ve test verilerine uygulayarak performansını ölçmek mümkün olmaktadır. Eğitim sırasında oluşabilecek hataları minimum seviyeye indirebilmek için optimizasyon ve en iyileme yöntemleri olarak adlandırılan eğimli iniş (gradient descent) yöntemleri kullanılmaktadır. Optimizasyon süreci öğrenme hızının da dahil olduğu adım adım işleyen bir süreçtir [33].

İleri yayılım aşamasında ağırlık parametresi seçim yöntemlerinden biri rastgele ağırlık değeri seçmektir. Seçilen değerin kayıp fonksiyonunda elde edilen sonuca göre zincirleme türev yoluyla güncellenmesi beklenir. Ağırlık değerinin ilk belirlenmesi aşaması eğimli iniş algoritmasının ilk aşamasını oluşturur. Genel görüş başlangıç değerinin çok önemli olmadığı yönündedir. Şekil 3.9 rastgele belirlenen ilk ağırlık değerinin ardından geri yayılım sonucunda maliyet fonksiyonuna göre ağırlık değerinin güncellenmesini ve negatif yönde azalan bir eğim inişini göstermektedir. Eğim inişinde amaç en uygun ağırlık değerini bularak minimum eğim noktasına ulaşmaktır. Normal bir eğim inişi algoritmasında her adımda minimum noktasına biraz daha yaklaşılması beklenmektedir. Şekil 3.9'da da görüldüğü gibi eğimin hem yönü hem de büyüklüğü vardır. Eğim iniş algoritmalarında bir sonraki noktayı belirlemek için eğim büyüklüğü ve öğrenme oranı kullanılmaktadır. Örneğin, eğim büyüklüğünün 2,5 ve öğrenme oranının 0,01 olarak belirlendiği eğim iniş algoritmasında bir sonraki nokta bulunulan noktadan sadece 0,025 birim uzakta olacaktır [34].

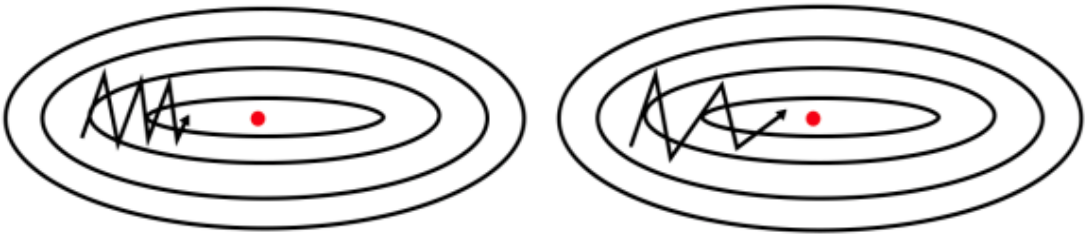


Şekil 3.9. Eğim inişi gösterimi [33].

Eğim iniş algoritmasında maliyet fonksiyonu eğitim veri setinde bulunan bütün örnekler kullanılarak hesaplanmaktadır. Bilgisayarlı görüde nesne tanıma gibi büyük ölçekli veri kümeleri kullanılarak yapılan eğitimlerde her örneğe göre hesaplama yapmak oldukça maliyetli ve zaman alıcı bir çalışma gerektirir. Eğitim setinin büyük olması algoritmanın ağırlıkları o derece yavaş güncellemesine ve eğitimin oldukça uzun sürmesine neden olur. Bu maliyetten kurtulmak için veri setinde bulunan tüm örnekleri kullanmak yerine belirli sayıda örneğin kullanıldığı olasılıksal eğim inişi (stochastic gradient descent) algoritması önerilmiştir. İsimde olasılıksal ifadesinin kullanılmasının nedeni ise her yinelemede eğitim seti içerisinden bir önceki yinelemede kullanılan verileri önemsemeden rastgele örnekler seçilmesinden kaynaklanmaktadır [35].

Maliyet ya da kayıp fonksiyonunun hesaplanması ile geri yayılım sürecinin amacı belirlenen ağırlık değerlerinin maliyeti minimum yapacak değeri bulmasını sağlamaktır. Bu süreçte olasılıksal eğim inişini minimum noktasına götüren öğrenme hızı ile birlikte diğer hiperparametre ise momentumdur. Momentum önceki ağırlık güncellemesi ile mevcut olana basitçe bir işaret ekler. Eğim aynı yönü göstermeye devam ettiğinde bu minimuma doğru atılan adımların boyutunu artırmaktadır. Dolayısıyla ağırlık ve yanlılık değerlerinin en uygun olduğu noktayı belirlemek için eğim inişinin daha hızlı ilerlemesi

sağlanmaktadır [36]. Şekil 3.8.(a)'da momentum kullanılmadan olasılıksal eğim inişi gösterilmiştir. Şekil 3.8.(b)'de ise olasılıksal eğim inişine momentumun eklenmesiyle önceki şekilde gösterilen inişe göre yanlara doğru çok daha az adım atılarak minimum noktasına doğru iniş olduğu görülmektedir.



(a) (b)

Şekil 3.10. Momentum kullanılan ve kullanılmayan olasılıksal eğim inişi [37].

Çok katmanlı yapay sinir ağlarında kullanılan bir diğer optimizasyon yöntemi Adam algoritmasıdır. RMSProp ve SGD algoritmaları kullanılarak türetilen bu algoritmaya göre öğrenme hızı uyarlanabilir şekilde ayarlanmaktadır. Yani farklı parametreler için bireysel öğrenme oranı ayarlanabilmektedir. Bu yüzden adı uyarlanabilir moment tahmininden türetilmiştir. Algoritma ismi kısaltma değildir ve uyarlanabilir moment tahmini (adaptive moment estimation) kelimelerinden türetilmiştir. Algoritma sinir ağının her ağırlık değerinin öğrenme hızını uyarlamak için birinci ve ikinci eğim anlarının tahminini kullanmaktadır [38].

3.3.7.4. Devir Sayısı ve Yığın Boyutu

Optimizasyon, algoritmanın yinelemeli olarak devam ettiği bir arama sürecidir. Çok katmanlı yapay sinir ağlarında her ileri ve geri yönde yayılımda optimizasyon algoritması daha iyi sonucu üretebilecek parametre değerlerini aramaya devam eder. Eğim inişi algoritması da optimizasyonda kullanılan temel algoritmalardan biridir. Eğim iniş algoritmasında tüm eğitim örnekleri üzerinde çalışmak yerine belirli parti (batch) grupları üzerinde çalışarak daha hızlı parametre güncellemeleri yapar. Burada parti olarak bahsedilen eğitim örneklerinin belirli sayılarda gruplandırılmasıdır. Tüm eğitim örnekleri tek bir grup olarak kullanıldığında öğrenme algoritması toplu eğim inişi olarak adlandırılır. Sadece rastgele bir parti grubu seçildiğinde olasılıksal eğim inişi, eğitim seti

boyutunu aşmayacak şekilde birden fazla parti grubu seçilirse mini toplu eğim inişi adını almaktadır. Bunun yanı sıra modelin tüm eğitim ve doğrulama veri setini kullanarak bir kez ileri ve geri yönde yayılım yapması bir devir (epok) olarak adlandırılır. Model eğitim setindeki örnekleri kullanarak ileri yönde yayılım yapar ve elde ettiği maliyet hatasına göre geri yönde yayılım ile o epok tamamlanmış olur [39].

Her modelin ulaşabileceği en üst başarı oranını yakalayabilmesi için belirli epok sayısı ile eğitilmesi gerekmektedir. Bu tez çalışmasında YOLOv3, EfficientDet ve RetinaNet modelleri 100 epok olarak eğitim sürecinden geçerken YOLOv5 modeli 500 ve 1000 epok olarak eğitim sürecinden geçmiştir. YOLOv5 modeli diğer modellere göre 5-10 kadar daha fazla epok sayısında eğitime tabi tutulmasına rağmen eğitimi daha kısa sürede tamamlamıştır. Bu nedenle eğitim süresi üzerinde yığın boyutu, devir sayısı ve adım sayısı gibi parametrelerin ayrı ayrı etkileri bulunmaktadır.

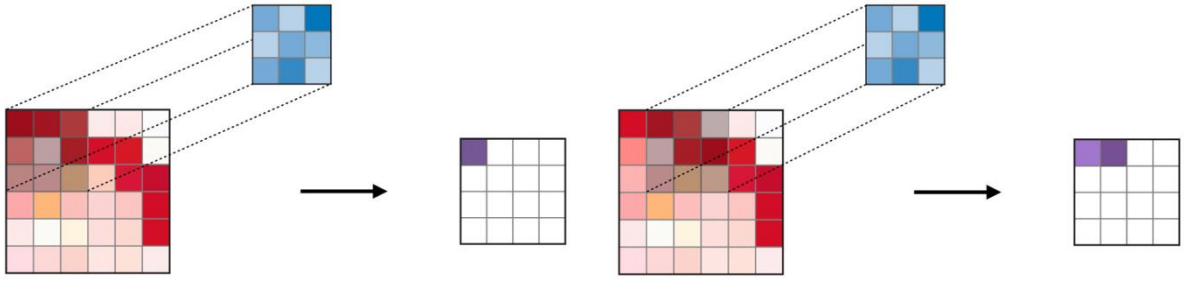
3.4. Evrişimsel Sinir Ağları

Derin öğrenme ya da derin sinir ağı kavramı çok katmanlı yapay sinir ağlarını ifade etmektedir. En popüler derin sinir ağlarından biri evrişimsel sinir ağları olarak görülmektedir. Evrişimsel sinir ağlarının en önemli özelliği örüntü tanıma çalışmalarında yapay sinir ağındaki parametre sayısını azaltmasıdır. Bir evrişimsel sinir ağı uygulamasında amaca ulaşmak için görüntünün tamamının aynı anda kullanılmasına gerek yoktur. Kullanılan farklı boyutlardaki filtreler sayesinde her katmanda evrişim işlemi yapılarak aynı anda işlenebilecek parametre sayısı büyük oranda azaltılabilmektedir. Evrişimsel sinir ağlarının bir diğer önemli yönü giriş görüntüsü daha derin katmanlara doğru yayıldığında daha soyut özellikler öğrenilebilmektedir. Yüz tanıma çalışmalarında ilk katmanlarda kenar özellikleri tespit edilirken daha derin katmanlara gidildikçe yüz hatları gibi spesifik özellikler tespit edilebilmektedir [40].

Evrişimsel sinir ağları ile yapay sinir ağları arasındaki dikkate değer tek fark evrişimsel sinir ağlarının öncelikle görüntü işlemede örüntü tanıma alanında kullanılmasıdır. Bu, görüntüye özgü özelliklerin mimariye kodlanmasına olanak tanır ve ağı görüntü odaklı görevler için daha uygun hale getirir. Ayrıca modeli kurmak için gerekli parametre sayısını da oldukça azaltır. Evrişimsel sinir ağları evrişim, havuzlama ve tam bağlantılı olmak üzere üç tür katmandan oluşmaktadır [41].

3.4.1. Evrişim Katmanı

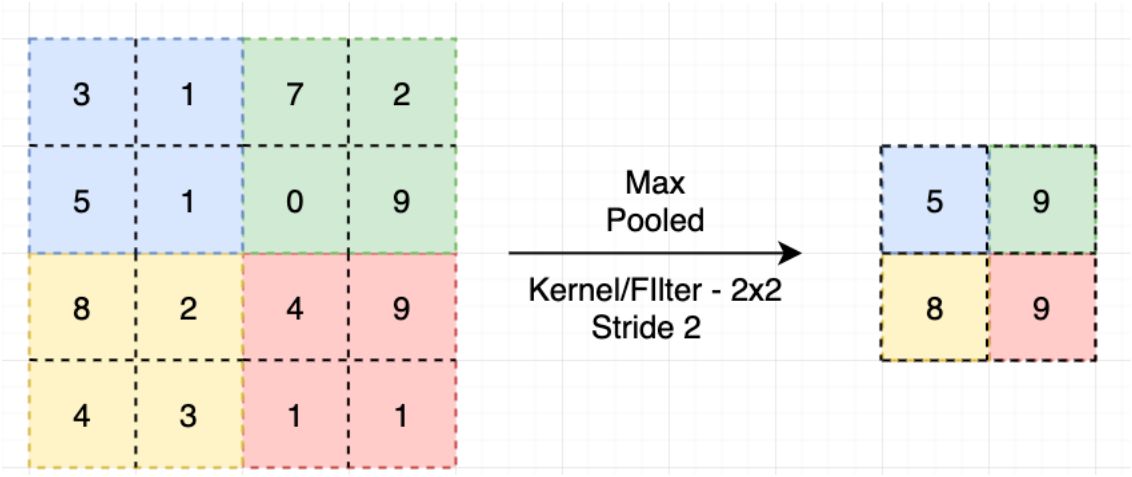
Evrişimsel sinir ağlarının ana yapısını oluşturan evrişim katmanı girdi görüntüsünü alarak evrişim işlemine tabi tutan filtreleri barındıran katmandır. Filtreler giriş görüntüsünü boyutlarına göre tararken kullanılır. Evrişim katmanı filtre boyutu, dolgulama ve adım kaydırma gibi hiperparametreleri içerir. Bu katmanda elde edilen çıktı öznitelik haritası ya da aktivasyon haritası olarak adlandırılmaktadır [42]. Şekil 3.11'de evrişim işleminin ilk iki adımı gösterilmektedir. Giriş görüntüsünün sol üst köşesinden başlayarak filtrenin sırayla uygulanması evrişim işlemini oluşturmaktadır.



Şekil 3.11. Evrişim işlemi [43].

3.4.2. Havuzlama Katmanı

Bir evrişimsel sinir ağı mimarisindeki ardışık evrişim katmanlarından sonra periyodik olarak havuzlama katmanı eklenir. Bunun amacı ağdaki parametre ve hesaplama miktarını azaltmak için hem temsilin uzamsal boyutunu kademeli olarak küçültmek hem de aşırı öğrenmeyi kontrol etmektir. Şekil 3.12'de maksimum havuzlama katmanına ait örnek gösterilmektedir [44].



Şekil 3.12. Maksimum havuzlama katmanı [45].

3.4.3. Tam Bağlantılı Katman

Evrişimsel sinir ağlarının en önemli ve en son katmanı tam bağlantılı katmanlardır. Bu katmanların görevi evrişimsel sinir ağını ileri yönde beslemektir. Tam bağlantılı katman kendisinden önceki evrişim, aktivasyon ve havuzlama katmanlarının çıktısını alır ve bu çıktıyı bir vektör olacak şekilde düzleştirir. Bu düzleştirilmiş vektör daha sonra yapay sinir ağlarının hesaplama mantığı ile aynı şekilde ağırlık ve yanlılık değerleriyle matematiksel olarak hesaplanır [46].

3.4.4. Hiperparametreler

Evrişimsel sinir ağlarında kullanılan katman sayısı, her bir katmanda kullanılan filtre sayısı, filtre boyutu, dolgu ve adım kaydırma değerleri, havuzlama türü ve boyutu gibi model başarısını etkileyen ve en uygun değerlere ayarlanması gereken hiperparametreler bulunmaktadır.

Bu kısımda çalışmalarda kullanılan sinir ağı modellerine ait hiperparametre özelliklerine değinilmiştir.

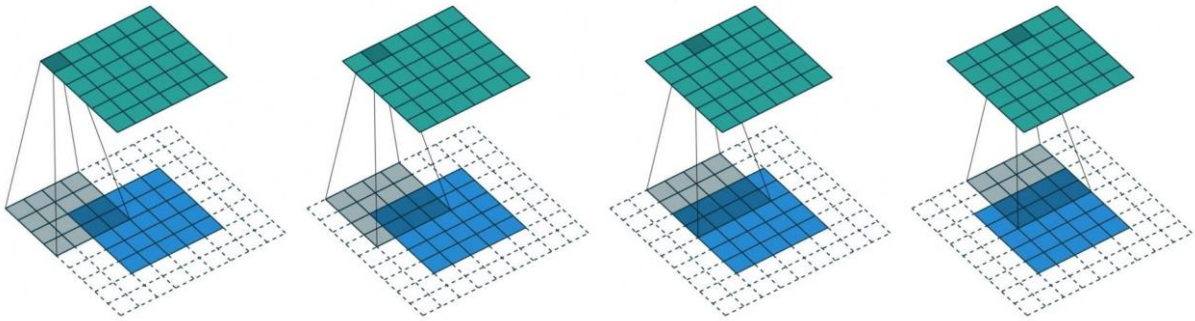
3.4.4.1. Filtre Boyutu

Evrişimsel sinir ağlarında girdi katmanını oluşturan görüntüler yükseklik ve genişlik olarak birçok kez bölünebilir piksel değerlerine sahip olmalıdır. Genellikle 32, 64, 224 ve 512 gibi piksel değerleri seçilmektedir. Evrişim katmanında kullanılan filtreler

genel olarak 3x3 ve 5x5 piksel olarak seçilir. Evrişim katmanının en önemli özelliği girdi görüntüsünün uzamsal boyutunu küçültmemesidir [47].

3.4.4.2. Dolgulama ve Adım Kaydırma

Evrişim işlemi esnasında giriş görüntüsü üzerinde sol üst köşede bulunan pikselden başlayarak kullanılan filtre önce yatay sırada sonra dikey sırada ilerleyerek tüm görüntüyü tarama yapar. Ancak buradaki tarama işlemi filtrenin boyutlarına ve özelliğine bağlı olarak üzerinde bulunduğu görüntünün genellikle merkez pikseline ait özellik çıkarma olarak işler. Bu nedenle genellikle kenarlarda ve köşelerde bulunan piksel değerleri merkeze alınamadığı için özellik çıkarma işlemine dahil edilemezler. Ayrıca çıktı boyutu kullanılan filtrenin giriş görüntüsü üzerindeki olası yerleşimlerinin sayısı kadar olmaktadır. Yani 32x32 piksel değerine sahip bir görüntü üzerinde 3x3 piksel değerine sahip bir filtre görüntünün sol kısmından başlayarak sağ tarafına dokunana kadar ilerlemeye devam eder ve çıktı görüntüsünün boyutu 30x30 piksel boyutlarında olur. Hem köşe ve kenar piksellerindeki özellikleri tespit edebilmek hem de her evrişim işlemi sırasında görüntünün sürekli küçülmesini engellemek için dolgulama (padding) adı verilen işlem uygulanır [48]. Şekil 3.7'de giriş görüntüsü üzerinde evrişim işlemi uygulanmadan önce kenarlarına dolgulama yapılmıştır.



Şekil 3.13. Evrişim işlemi öncesinde dolgulama uygulanan giriş görüntüsü [48].

Dolgulama işleminde giriş görüntüsünün bütün kenarlarına genellikle sıfır pikselleri eklenir. Bu sayede evrişim işleminde matrislerin matematiksel hesaplamasında herhangi bir değişiklik olmaz ve giriş görüntüsü aynı boyutta çıktı olarak elde edilir. Adım kaydırma işleminde ise giriş görüntüsü üzerinde filtrenin evrişim işlemi sırasında

her defasında sağa ve aşağıya kaç piksel atlayarak gideceği belirlenir. Burada amaç evrişim işlemi sonunda özellik kaybına uğramadan çıktı görüntüsü boyutunu küçülterek parametre sayısını azaltmaktır.

3.4.4.3. Havuzlama Türü ve Boyutu

Genellikle kullanılan iki farklı havuzlama katmanı bulunmaktadır ve her havuzlama katmanı görüntünün tüm derinlik dilimlerine uygulanır. Havuzlama katmanının en yaygın kullanılan şekli 2x2 boyutlu olan kare boş filtredir. Bu filtre evrişim görüntüsü üzerinde tıpkı evrişim işleminde olduğu gibi sağa ve aşağıya giderek bulunduğu her 4 piksel değerini alır. Kullanılan havuzlama katmanı ortalama havuzlama ise bu 4 piksel değerinin ortalamasını alarak ilgili piksele yerleştirir. Kullanılan havuzlama katmanı maksimum havuzlama ise bu 4 piksel değerinden en büyük piksel değerini ilgili piksele yerleştirir. Havuzlama katmanının uygulanmasıyla evrişim işlemi sonunda elde edilen görüntü büyüklüğü %75 oranında azaltılmış olur. Son yıllarda evrişimsel sinir ağı çalışmalarında genellikle maksimum havuzlama türü kullanılmaktadır.

3.5. Veri Setleri

Literatür araştırmasında net bir şekilde görülmüştür ki geliştirilen evrişimsel sinir ağı modelleri genellikle üç veri seti kullanılarak eğitim ve test sürecine tabi tutulmaktadır. En çok kullanılan veri setlerinin daha iyi tanınması tez sürecinde hazırlamış olduğumuz veri setinin farklı yönlerini ve özelliklerini daha iyi gösterecektir. Bu nedenle bu başlık altında evrişimsel sinir ağı modellerinde en çok kullanılan veri setlerine değinilmiştir.

3.5.1. ImageNet Veri Seti

ImageNet veri seti 27 üst kategoriden ve bu kategorilerle ilişkili olarak en az 20 bin alt kategoriden oluşan yaklaşık 14 milyon tam çözünürlüklü görüntü barındıran görüntü veri kümesidir. WordNet tarafından sağlanan hiyerarşik yapı üzerine inşa edilmiştir. Toplamda 14 milyondan fazla görüntü barındıran veri setinde 1 milyondan fazla sınırlayıcı kutu notlarına sahip resim bulunmaktadır. 2010 yılından beri her yıl düzenlenen ILSVRC yarışmasında kullanılmaktadır [49, 50].

3.5.2. Pascal VOC Veri Seti

Pascal VOC veri seti 2005 – 2012 yılları arasında geliştirilerek yayınlanmış her yıl aynı isimle düzenlenen yarışmada kullanılan ve en son haliyle toplam 20 nesne sınıfı olmak üzere 11.530 görüntüden ve 27.450 nesne etiketi verisinden oluşan görsel veri setidir. Kullanılan sınıflar genel olarak araba, bisiklet, otobüs, insan, motosiklet, gemi, kedi, köpek gibi gerçek hayat nesnelerinden oluşmaktadır [51].

3.5.3. MS COCO Veri Seti

MS COCO veri kümesi, 82 nesne kategorisi 5 binden fazla etiketli içeriğe sahip toplam 91 ortak nesne kategorisi içerir. Toplamda veri kümesinde 328 bin görüntüde 2 milyon 500 bin etiketli örneği bulunmaktadır. ImageNet veri setinin aksine, COCO daha az kategoriye ve kategori başına daha fazla örneğe sahiptir. Bu durum, hassas iki boyutlu yerelleştirme yapabilen ayrıntılı nesne modellerinin öğrenilmesine yardımcı olmaktadır. Veri kümesi, kategori başına giriş sayısı bakımından PASCAL VOC ve diğer veri kümelerinden önemli ölçüde daha büyüktür. Veri setinde kullanılan sınıflar; insan, bisiklet, araba, tren, kamyon, trafik ışığı vb. gibi gerçek hayat nesnelerinden oluşmaktadır [52].

3.5.4. MNIST Veri Seti

MNIST, el yazısıyla yazılmış rakamlardan oluşan ve 60 bin eğitim, 10 bin test olmak üzere toplamda 70 bin görüntüden oluşan bir veri setidir. Görüntüler 28x28 piksel boyutlarında olup gri seviyelidir [53].

3.5.5. CIFAR Veri Seti

CIFAR, 80 milyon küçük boyutlu görüntü içeren veri setidir. İki farklı versiyonu olan veri setinde; CIFAR-10 veri seti her sınıfta 6000 görüntü bulunan toplam 10 sınıftan oluşmaktadır. Görüntü boyutları 32x32 piksel boyutlarında olup 50 bin görüntü eğitim, 10 bin görüntü test kümesi olarak ayrılmıştır. CIFAR-100 veri seti ise her sınıfta 600 görüntü bulunan toplam 100 sınıftan oluşmaktadır [54].

3.6. Evrişimsel Sinir Ağı Modelleri

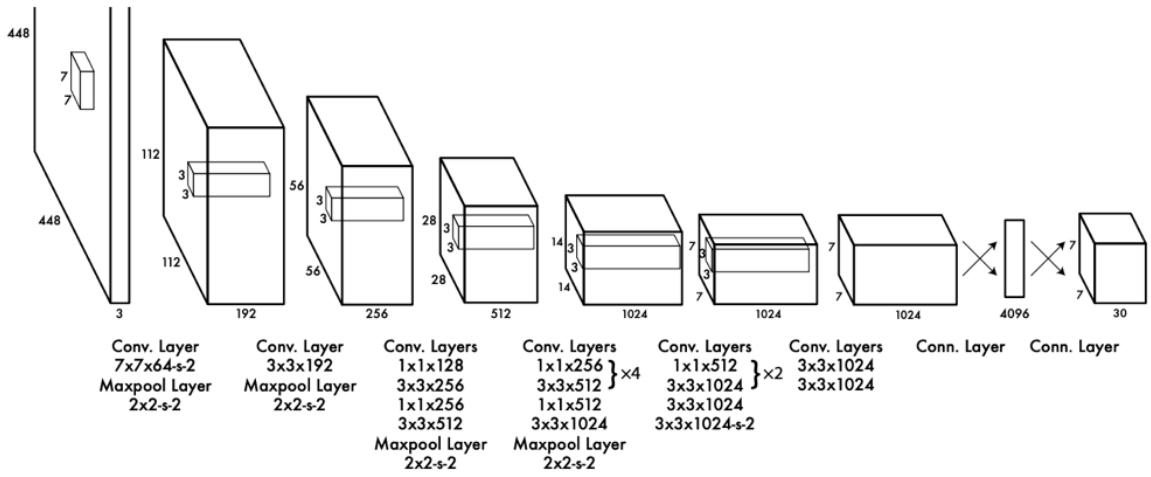
Evrişimsel Sinir Ağları kullanılarak yapılan nesne tanıma çalışmaları genel olarak iki ayrı kategoride değerlendirilmektedir. İki aşamalı nesne tespit dedektörleri önce bölge önerisi daha sonra önerilen bölgeler içerisinde nesne algılama çalışmaları yapmaktadır. R-CNN, Fast-RCNN, Faster-RCNN iki aşamalı nesne tespit modellerindendir. Tek aşamalı nesne tespit dedektörleri ise doğrudan evrişimsel sinir ağı çalışmaları ile nesne algılama çalışmaları yapmaktadır. RetinaNet, SSD ve YOLO modelleri tek aşamalı nesne tespit modellerindendir.

Tez çalışmasının amacı tek aşamalı nesne tespit modellerinin başarı ve performans karşılaştırması olduğu için ve özellikle YOLO modelleri üzerinde geliştirmeler yapıldığı için kullanılan evrişimsel sinir ağı modelleri bu başlık altında anlatılmıştır.

3.6.1. YOLOv3

YOLO modeli nesne algılama problemini doğrudan görüntü piksellerinden sınırlayıcı kutu koordinatlarına ve olasılık sınıflarına kadar tek bir regresyon sorunu olarak değerlendirmektedir. Bu model kullanıldığında hangi nesnelerin mevcut olduğunu ve nerede olduklarını tahmin etmek için bir resme yalnızca bir kez bakmak yeterli olmaktadır. Tek bir evrişimsel ağ aynı anda birden çok sınırlayıcı kutuları ve bu kutular için sınıf olasılıklarını tahmin eder. YOLO, tam görüntüler üzerinde eğitim alır ve algılama performansını doğrudan optimize eder. Bu birleşik model, iki aşamalı modellerle kıyaslandığında; hız, daha az arka plan hatası yapma ve daha genellenebilir çıkarımda bulunma gibi özellikleri ile ön plana çıkmaktadır. YOLO görüntülerde nesneleri hızlı bir şekilde belirleyebilmesine rağmen yerelleştirme yani belirlediği nesnenin sınıfını belirleyip sınırlayıcı kutu çizme konusunda diğer modellere göre daha az başarılıdır. YOLO mimarisinde ağın ilk evrişimsel katmanları görüntüden öznitelikleri çıkarırken tamamen bağlı katmanlar çıktı olasılıklarını ve koordinatlarını tahmin eder. YOLO modeli GoogLeNet mimarisinden esinlenilerek oluşturulmuştur. Model, 24 evrişimsel katman ve 2 tam bağlantılı katmandan oluşmaktadır. Tahmin başarısının artırılabilmesi için giriş görüntüsü 224x224 piksel olarak ImageNet sınıflandırma görevinde önceden eğitilmiş daha sonra görüntü 448x448 piksel olarak ağa girdi olarak

sunulmuştur. Nesne içermeyen sınırlayıcı kutu tahminlerinin eğimi düşürmesini engellemek için nesne içeren kutular için kullanılan değişken (λ_{coord}) değeri 5, nesne içermeyen kutular için kullanılan değişken (λ_{noobj}) değeri ise 0.5 olarak ayarlanmış ve bu şekilde toplam-kare hatası işlemine tabi tutulmuştur. VGG-16 ağı kullanılarak tasarlanan YOLO ve Faster R-CNN modelleri Pascal VOC 2007 ve 2012 veri seti kullanılarak eğitime tabi tutulmuş ve YOLO hem eğitim sürecinde hem de gerçek zamanlı tahminde daha hızlı olmasına karşın ortalama doğruluk oranı olarak daha az başarılıdır. Şekil 3.14'te ilk geliştirilen YOLO modeline ait mimari tasarım gösterilmiştir [55].



Şekil 3.14. YOLO modelinin mimari görünümü [55].

Genel amaçlı nesne tespiti hem hızlı ve doğru olmalı hem de çok çeşitli nesneleri tanıyabilmelidir. Mevcut nesne algılama veri kümeleri, sınıflandırma ve etiketleme gibi diğer görevler için veri kümelerine kıyasla sınırlıdır. Hem halihazırda sahip olunan büyük miktardaki sınıflandırma verisinden yararlanmak hem de nesne dedektörlerini sadece sınıflandırma için değil aynı zamanda algılama konusunda da eğitmeye olanak tanıyan ortak bir eğitim algoritması önerilmiştir. YOLO modeli geliştirilerek gerçek zamanlı nesne algılama modeli YOLOv2 üretilmiştir. Geliştirilen model YOLOv2, ImageNet ve MS COCO veri setlerinin birleştirilmesiyle oluşturulan veriler kullanılarak eğitilmiş ve 9000'den fazla nesne kategorisini algılayabilen YOLO9000 modeli üretilmiştir. İlk geliştirilen YOLO modeli, Faster R-CNN modeli ile karşılaştırıldığında daha çok yerelleştirme hatası yapan ve daha az hata analizi yapan bir modeldir. YOLOv2 modeli geliştirilirken hızlı olma özelliğinin korunması öncelik olurken başarıyı artırmak için ağı

derinleştirmek yerine basitleştirerek temsilin öğrenilmesi kolaylaştırılmıştır. YOLOv2 modeli toplu normalleştirme kullanarak ortalama doğruluk oranını yaklaşık %2 artırmıştır. Tüm son teknoloji tespit yöntemleri, ImageNet üzerinde önceden eğitilmiş sınıflandırıcı kullanmaktadır. AlexNet'ten başlayarak çoğu sınıflandırıcı 256x256'dan daha küçük girdi görüntülerinde çalışır. Orijinal YOLO, sınıflandırıcı ağını 224x224'te eğitir ve algılama için çözünürlüğü 448x448'e çıkarır. Giriş görüntüsü çözünürlüğünün artırılması modelin ortalama doğruluk oranını yaklaşık %4 artırmıştır. İlk geliştirilen YOLO modelinden tamamen bağlı katmanlar çıkarılmış ve sınırlayıcı kutuları tahmin etmek için bağlantı kutuları kullanılmıştır. Ayrıca giriş görüntüsü 448x448 piksel yerine 416x416 piksel olarak ayarlanmış, büyük nesnelerin görüntüde merkezde yer alabileceği tek bir konum tahmin edecek ağ tasarımı yapılmıştır. YOLOv2 modelinin farklı bir özelliği de ağın her 10 rastgele partide görüntü boyutlarının 32x32 ile 608x608 arasında 32 değerinin katları olacak şekilde değiştirilmesidir. Bu şekilde ağ farklı görüntü boyutlarında iyi tahmin yapmayı öğrenmeye zorlanır. Bu sayede hız ve doğruluk olarak denge sağlanır. YOLOv2 modeli sınırlayıcı kutu koordinat tahmini ve yerelleştirmenin yanı sıra ortak nesnelerin nasıl sınıflandırılacağı gibi algılamaya özgü bilgileri de öğrenmek üzerine tasarlanmış bir algoritma olduğu için eğitim sırasında hem algılama hem de sınıflandırma kümelerinden veriler karıştırılmıştır. YOLOv2 modeli Pascal VOC 2007 ve 2012 veri seti üzerinde eğitilen Faster R-CNN ve SSD modeli ile karşılaştırıldığında hem daha hızlı hem daha doğru sonuçlar üretmiştir [56].

YOLOv2 modeli üzerinde birtakım geliştirmeler yapılarak ağ daha derin hale getirilmiş ve YOLOv3 modeli üretilmiştir. YOLOv3 modeli de diğer modellerde olduğu gibi ilk olarak görüntü üzerinde sınırlayıcı kutuyu çizer ve ardından kutu içerisinde bulunabilecek nesne sınıfını tanımlayabilmek için nesne algılamayı iki aşamalı bir problem durumu olarak ele almaktadır. YOLOv3 modeli özellik çıkarma ağı olarak Darknet-53 mimarisini kullanmaktadır. Geliştirme aşamasında YOLOv2'de kullanılan havuzlama katmanı çıkarılmıştır ve tespit edilen nesne büyüklüğüne göre kullanılan özellik haritasının büyüklüğü de değişmektedir. Geliştirilen modelin hem SSD hem de RetinaNet modeline göre daha hızlı ve daha başarılı olduğu tespit edilmiştir [57,58]. Şekil 3.15'te YOLOv3 mimarisinde kullanılan Darknet-53 kütüphanesinin filtre sayısı, filtre boyutu ve çıktı boyutları gösterilmektedir.

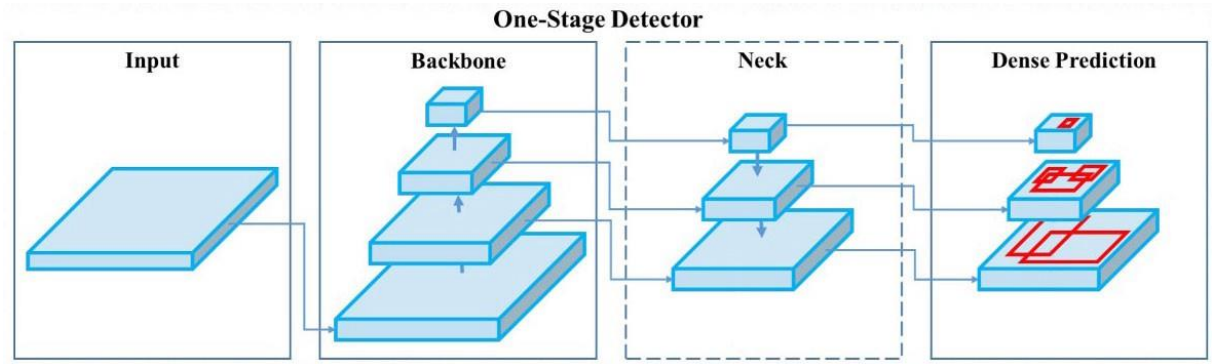
	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Şekil 3.15. YOLOv3 modelinin kullandığı Darknet-53 mimarisi [58].

3.6.2. YOLOv5

Modern bir nesne algılama dedektörü genellikle ImageNet üzerinde önceden eğitilmiş bir omurga (backbone) ve nesnelerin sınıfları ile sınırlayıcı kutularını tahmin etmek için kullanılan bir kafa (head) olmak üzere iki parçadan oluşur. Omurga olarak VGG, ResNet, ResNeXt, DenseNet, MobileNet kullanılır. Tek aşamalı nesne tespit dedektörlerinde kafa olarak YOLO, SSD, RetinaNet, CenterNet, CornerNet ve iki aşamalı nesne tespit dedektörlerinde ise kafa olarak Faster R-CNN, R-FCN kullanılabilir. Son yıllarda geliştirilen nesne dedektörleri omurga ile kafa arasında boyun (neck) adı verilen ve farklı katmanlardan özellik haritalarını çıkaran yeni katmanlar eklemişlerdir. Genellikle bir boyun birkaç aşağıdan yukarıya ve yukarıdan aşağıya yoldan oluşur. Bu

şekilde oluşturulmuş ağlara Özellik Piramit Ağı (FPN), Yol Birleştirme Ağı (PAN) ve Çift Yönlü Özellik Piramit Ağı (BiFPN) örnek olarak verilebilir. YOLOv4 modeli; kafa olarak kendisinden önce geliştirilmiş YOLOv3 modeli, omurga olarak CSPDarknet53 mimarisi ve boyun olarak SPP ve PAN kullanılarak geliştirilmiştir. Geliştirilen modelin ImageNet ve MS COCO veri seti ile eğitimi sırasında etkinliğini artırmak için farklı veri artırma, sınıf etiketi yumuşatma, aktivasyon fonksiyonlarını değiştirme gibi teknikler uygulanmıştır. Geliştirilen modelin diğer yeni teknoloji gerçek zamanlı nesne algılama modellerine göre daha hızlı ve başarılı olduğu tespit edilmiştir [59]. Şekil 3.16'da tek aşamalı nesne tespit dedektörünün genel şeması gösterilmiştir. YOLOv5 modelinin de kullandığı mimari omurga, boyun ve kafa yapılarından oluşmaktadır. Giriş görüntüsü omurga (backbone) üzerinde özellik çıkarma işlemine tabi tutulur. Omurgadan aldığı özellikleri kullanarak oluşturduğu özellik haritalarının kafa bölümüne taşıyan boyun (neck) kısmı aşağıdan yukarıya ve yukarıdan aşağıya birkaç yoldan oluşur. Tek aşamalı nesne tespit dedektörleri kafa (head) bölümünde yoğun tahmin (dense prediction) sonuçlarını çıkarmaktadır. Tahminler sınırlayıcı kutu ile yerelleştirilmektedir.



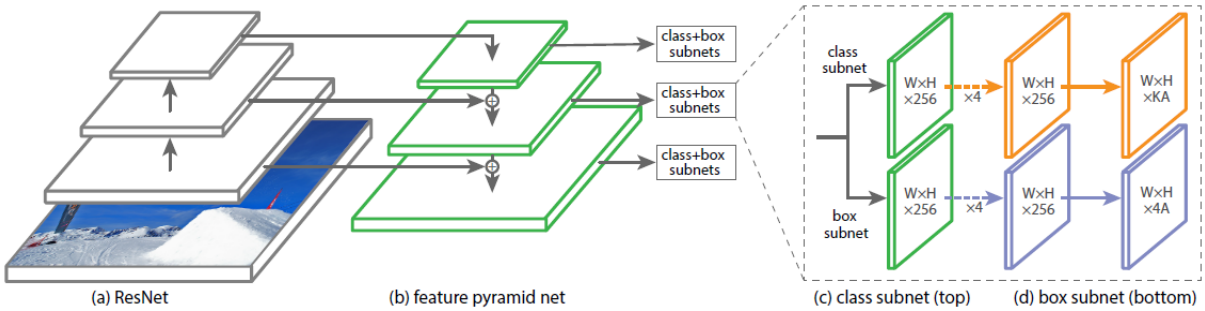
Şekil 3.16. Tek Aşamalı Nesne Tespit Dedektörü [60].

Darknet kütüphanesi üzerine geliştirilen YOLOv4 modelinin Pytorch kütüphanesi üzerine yapılan yeni bir geliştirmesi ile YOLOv5 modeli üretilmiştir. Model, YOLOv4 modeli ile boyun ve kafa olarak aynı yapıları kullanmaktadır. YOLOv5 modelinin diğer YOLO sürümlerinden en önemli farkı geliştirildiği kütüphane olmuştur. Ayrıca diğer YOLO sürümlerinden farklı olarak YOLOv5 modeli için geliştiricileri tarafından resmi bir makale yayımlanmamıştır.

3.6.3. RetinaNet

Günümüzde kullanılan son teknoloji ürünü nesne algılama dedektörleri Faster R-CNN modelinin popüler hale getirdiği iki aşamalı modellerdir. Bu teknolojide ilk aşama seyrek bir aday nesne konumlar kümesi oluşturur ve ikinci aşama ise bu aday konumları ön plan sınıflarından ya da evrişimsel sinir ağı kullanarak arka plan olarak sınıflandırır. Bu modellerin doğruluk oranı yüksek olmakla birlikte tek aşamalı dedektörlere göre oldukça yavaştır. RetinaNet ile yeni bir kayıp işlevi ekleyerek Faster R-CNN ya da Mask R-CNN nüshası taşıyan yeni bir dedektör oluşturulmuştur. Burada tek aşamalı dedektörlerin yaşadığı ana sorun olan eğitim aşamasındaki sınıf dengesizliği iki aşamalı dedektörlerde aday bölge önerisiyle giderilmektedir. İki aşamalı nesne algılama dedektörleri belirli sayıda aday bölge ile ilgilenirken tek aşamalı dedektörler binlerce aday nesne konumunu işlemelidir. YOLO ve SSD gibi tek aşamalı dedektörlerde görüntü başına ortalama 104-105 sınıf aday konumu değerlendirilirken yalnızca birkaç tanesi nesne içermektedir. Bu durum hem eğitimin verimsiz olmasına hem de modelin üretken olmamasına neden olur. RetinaNet modeli ile doğru sınıfa olan güven arttıkça ölçeklendirme faktörünün sıfıra düştüğü dinamik olarak ölçeklenmiş bir çapraz entropi kaybı önerilmiştir. Bu öneriye göre ölçeklendirme faktörüne kolay örneklerin katkısı azaltılırken eğitim zor örneklerin üzerine yoğunlaşmaktadır. ResNet-101-FPN omurgasına inşa edilen RetinaNet modeli MS COCO test setinde 39.1 doğruluk elde ederek önceki tüm tek aşamalı ve iki aşamalı detektörlerden daha iyi performans göstermiştir [61].

Şekil 3.7'de tek aşamalı nesne tespit dedektörlerinden olan RetinaNet modelinin yapısı gösterilmektedir. RetinaNet bir omurga iki alt ağdan oluşmaktadır. Omurga giriş görüntüsünün tamamı üzerinde bir evrişimsel özellik haritası hesaplamaktan sorumludur. Birinci alt ağ omurganın çıktısı üzerinde evrişimsel nesne sınıflandırması gerçekleştirir. İkinci alt ağ ise evrişimsel sınırlayıcı kutu regresyonu gerçekleştirir.

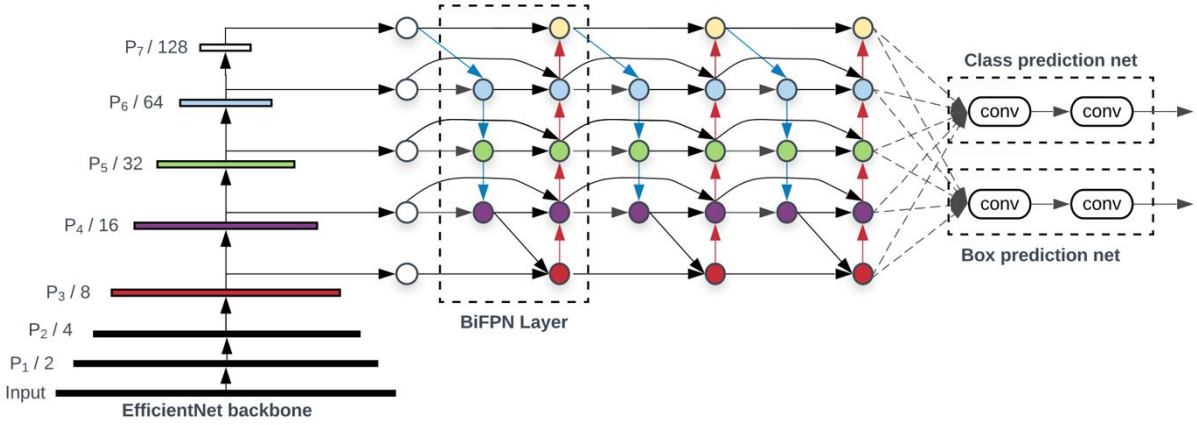


Şekil 3.17. RetinaNet modelinin mimarisi [61].

Diğer modellerle son performans değerlendirmesine göre RetinaNet modeli, %39,1 ortalama başarı oranı elde ederken DSSD513 tek aşamalı nesne tespit dedektörü %33,2 ve Faster R-CNN iki aşamalı nesne tespit dedektörü %36,8 ortalama başarı oranı elde etmiştir [61].

3.6.4. EfficientDet

Tek aşamalı ve çapasız nesne tespit dedektörleri yüksek verimlilik elde etme eğiliminde olsalar da doğruluktan feragat etmektedirler. Mobil cihazlardan yüksek veri merkezlerine kadar geniş bir yelpazede kullanılan nesne tespit dedektörleri için hem yüksek verimlilik hem de yüksek doğruluk sağlayacak kaynak kısıtlarının ölçekli kullanılması yöntemi EfficientDet mimarisinin temelini oluşturmaktadır. EfficientDet modeli iki önemli geliştirme sağlamaktadır. Birincisi farklı giriş özelliklerinin önemini öğrenmek için öğrenilebilir ağırlıklar sunan basit ancak etkili çift yönlü bir özellik piramit ağı önerilmektedir. İkincisi modelin hem omurga ağı olarak hem de kutu/sınıf tahmin ağı olarak derinlik, genişlik ve çözünürlük olmak üzere bileşik ölçekleme yöntemi önerilmektedir. Geliştirilen EfficientDet modeli YOLOv3 ve RetinaNet modeline kıyasla daha az kaynak kullanmasına rağmen daha başarılı sonuçlar üretmektedir. Şekil 3.18'de ImageNet üzerine önceden eğitilmiş EfficientNet omurga ağı üzerine geliştirilen EfficientDet mimarisi gösterilmektedir. EfficientNet omurgası P1 ve P7 arasında 7 farklı seviyeden oluşmaktadır. P3 – P7 arasındaki seviyelerden elde edilen özellikler daha sonraki katmanda tekrar tekrar özellik füzyonuna tabi tutulmaktadır. Bu birleştirilmiş özellikler sırasıyla nesne sınıfları ve kutu tahminleri üretmek için sınıf ve kutu ağına beslenmektedir. [62].



Şekil 3.18. EfficientNet omurgası üzerine kurulan EfficientDet mimarisi [63].

EfficientDet modelinin ikinci önerisi olan bileşik ölçekleme yönteminde ImageNet üzerinde önceden eğitimiş EfficientNet omurga ağı derinlik, genişlik ve çözünürlük olarak ölçeklendirmeye tabi tutulmuştur. Ölçeklendirmede derinlik doğrusal olarak, genişlik üssel olarak artırılmış ve en iyi genişlik değeri 1,35 olarak belirlenmiştir. Çözünürlük ölçeklemesinde ise D0 modelinden başlayarak D7 modeline doğru her yükselmede çözünürlük değeri $2^7 = 128$ olacak şekilde artırılmıştır. En düşük çözünürlük değeri D0 modelinde 540 piksel en yüksek çözünürlük değeri D7 modelinde 1536 piksel olacak şekilde ayarlanmıştır. EfficientDet modeli doğruluk incelemesinde YOLOv3 modelinden 28 kat daha az kaynak kullanmasına karşılık aynı doğruluğu elde etmiştir. Ayrıca RetinaNet ve Detectron2 ile karşılaştırıldığında 8 kat daha az parametre ve 21 kat daha az kaynak kullanımına rağmen aynı doğruluğu elde etmiştir. Ayrıca EfficientDet modelleri GPU'da en az 4 kat, CPU'da en az 10 kat daha hızlıdır. Ancak bu çalışmada tek aşamalı nesne tespit dedektörleri karşılaştırılırken sadece RetinaNet modeli yeniden üretilmiş, diğer modellerin performansına atıfta bulunulmuştur [62].

BÖLÜM 4. GERÇEKLEŞTİRİLEN UYGULAMA VE PERFORMANS SONUÇLARI

Yapılan kapsamlı literatür araştırmasına göre son yıllarda geliştirilen tek aşamalı nesne tespit modellerinin hız ve başarı konusunda elde ettikleri üstün performans gerçek zamanlı nesne tespitinde öncelikli olarak kullanılmalarını sağlamıştır. Geliştirme sürecinde aynı fikirden yola çıksa da tek aşamalı nesne tespit modellerinin tasarım olarak farklılıkları bulunmaktadır. Özellikle son zamanlarda geliştirilen modellerden YOLOv5 modelinin RetinaNet ve EfficientDet ile karşılaştırmaları yapılmaktadır. Bu çalışmada geliştirilen YOLOv5 modelinin hava fotoğrafları kullanılarak YOLOv3, RetinaNet ve EfficientDet modelleri karşısında performansı değerlendirilmiştir.

4.1. Kullanılan Veri Setleri

Evrişimsel sinir ağı kullanılarak nesne tanıma çalışmaları için son yıllarda farklı veri setleri ve farklı algoritmalar/mimariler geliştirilmiştir. Her ne kadar farklı mimariler geliştirilerek daha verimli ve daha doğru sonuçlar elde edilse de nesne tanıma çalışmalarında başarılı olunmasının en önemli sebeplerinden biri algoritmalara kaynak olarak sunulan veri miktarının çok fazla olmasıdır. Derin öğrenme alanında yapılan çalışmalar incelendiğinde genel olarak ImageNet, Pascal VOC, MNIST, CIFAR ya da MS COCO veri seti ile çalışıldığı tespit edilmiştir. Bu veri setleri Bölüm 3.5'te anlatıldığı üzere hem çok fazla veriden oluşmakta hem de nesne sınıfı olarak ayırıcı özelliklere sahip olmaktadır. Ayrıca görüntü boyutları oldukça küçük olduğu için eğitim sürecinin çok daha kısa sürede tamamlanmasını sağlamaktadır. Ancak bu tez çalışması insansız hava aracından alınan görüntüler kullanılarak oluşturulan YOLO mimarisi üzerinde eğitim süreci ve sonrası için performans değerlendirmesi yapmaktadır. Çalışmalarda modellerin verimliliğini daha iyi anlayabilmek açısından insansız hava aracından alınan görüntülere ait iki farklı veri seti kullanılmıştır.

İlk olarak DOTA veri seti kullanılmıştır. Veri seti içerisinde her biri 4000 x 4000 piksel çözünürlüklere sahip toplamda 2806 havadan alınmış görüntü ve 15 nesne sınıfının kullanıldığı toplam 188.282 etiket bulunmaktadır. Çalışma içerisinde veri seti nesne sınıflarına ait isimlendirmelerde herhangi bir değişiklik yapılmamıştır. Nesne sınıfları;

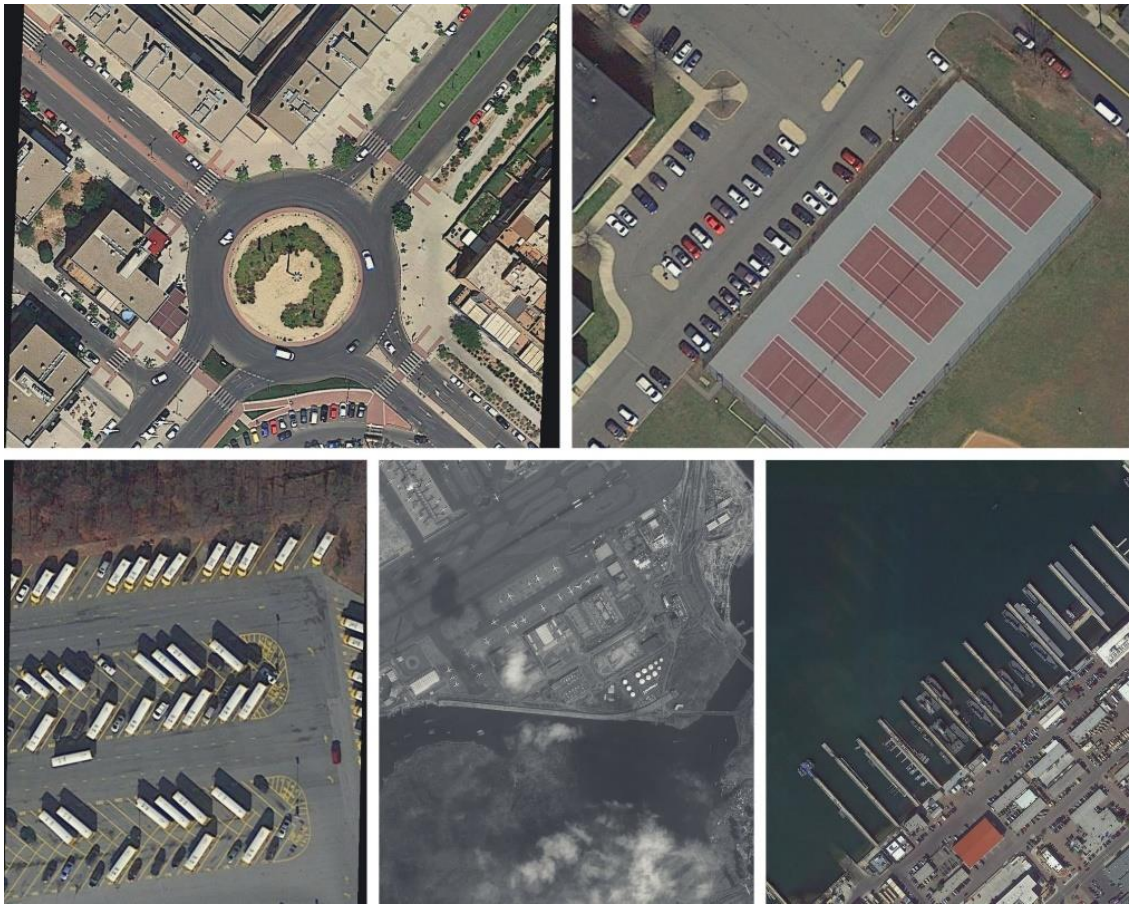
uzun araç(large-vehicle), yüzme havuzu (swimming-pool), helikopter (helicopter), köprü (bridge), uçak (plane), gemi (ship), futbol sahası (soccer-ball field), basketbol kortu (basketball court), yürüyüş alanı (ground track field), küçük araç (small-vehicle), liman (harbor), beyzbol alanı (baseball diamond), tenis kortu (tennis court), kavşak (roundabout), depolama tankı (storage tank) olarak isimlendirilmiştir. Veri setinden alınan 1410 görüntü eğitim, doğrulama ve test olarak üç parçaya bölünmüştür. Veri setinin üçe bölünmesinde bilgisayarlı görüde genel olarak takip edilen ve en uygun oran olarak belirtilen %70 eğitim, %20 doğrulama ve %10 test oranı kullanılmıştır. Sonuç olarak veri seti 987 adet eğitim, 282 adet doğrulama ve 141 adet test görüntüsünden oluşturulmuştur [64].

Tablo 4.1 DOTA veri setinde bulunan nesne sınıflarına ait etiket sayısının test veri seti ve toplam veri seti olarak ayrı ayrı dağılımını göstermektedir. Tablodan görüleceği üzere veri setinde bazı nesne sınıflarının etiketlenme oranı yüksek, bazı nesne sınıflarının ise oldukça düşüktür. Bu durum modellerin ortalama doğruluk oranı üzerinde olumsuz etki yaratmaktadır.

Tablo 4.1. DOTA Veri Setinin Etiket Dağılımı

Etiket Adı	Test Veri Setindeki Etiket Sayısı	Veri Setindeki Toplam Etiket Sayısı	Test Veri Setindeki Etiket Sayısının Toplam Etiket Sayısına Oranı
Ship (Gemi)	2056	28068	%7,33
Small-Vehicle (Küçük Araç)	2945	26126	%11,27
Large-Vehicle (Uzun Araç)	1970	16969	%11,61
Plane (Uçak)	859	7883	%10,90
Harbor (Liman)	1249	5983	%20,88
Storage-Tank (Depolama Tankı)	289	5015	%5,76
Tennis-Court (Tenis Kortu)	169	2367	%7,14
Bridge (Köprü)	202	2047	%9,87
Swimming-Pool (Yüzme Havuzu)	83	1736	%4,78
Helicopter (Helikopter)	38	630	%6,03
Basketball-Court (Basketbol Kortu)	32	515	%6,21
Baseball-Diamond (Beysbol Alanı)	25	415	%6,02
Roundabout (Kavşak)	28	399	%7,02
Soccer-Ball-Field (Futbol Alanı)	34	326	%10,43
Ground-Track-Field (Parkur)	31	325	%9,54
TOPLAM	**10010**	**98804**	**%10,13**

Çalışma esnasında evrişimsel sinir ağı modellerinde ekran kartının verimli kullanılması için ilk veri setindeki görüntüler 640 x 640 piksel olarak yeniden boyutlandırılmıştır. Daha yüksek değerlerde görüntü boyutlarında eğitim aşaması ekran kartının aşırı kullanımı nedeniyle tamamlanamamıştır. Şekil 4.1'de DOTA veri setinden alınan örnek görüntüler gösterilmektedir. Veri setinde bulunan görüntüler farklı yüksekliklerden elde edildiği için bazı nesneler oldukça küçük temsil edilmektedir.



Şekil 4.1. DOTA veri setinden örnek görüntüler

İkinci veri seti olarak Stanford Üniversitesi tarafından nesne algılama ve izleme çalışmalarına kaynak oluşturması için üniversite kampüsünün insansız hava aracı ile çekiminden oluşturulan veri seti kullanılmıştır [65]. Veri seti kampüs içerisinde farklı noktalardan alınan video görüntüsü karelerinden oluşmaktadır. Nesne sınıfı olarak bisikletli (biker), yaya (pedestrian), patenli (skater), araba (car), kampüs arabası (cart) ve otobüs (bus) sınıfları bulunmaktadır. Veri setinde bulunan toplam 3259 adet görüntü eğitim, doğrulama ve test olarak üç parçaya bölünmüştür. Veri setinin üçe bölünmesinde yine bilgisayarlı görüde genel olarak takip edilen ve en uygun oran olarak belirtilen %70 eğitim, %20 doğrulama ve %10 test oranı kullanılmıştır. Sonuç olarak kullanılan veri seti 2283 adet eğitim, 653 adet doğrulama ve 326 adet test görüntüsü olacak şekilde oluşturulmuştur. Tablo 4.2'de oluşturulan Stanford drone veri setinin test veri setindeki ve toplam veri setindeki nesne sınıflarına ait etiket dağılımlarını göstermektedir. Modellerin başarısını değerlendirirken ortalama doğruluk oranı üzerinde etiketli nesne

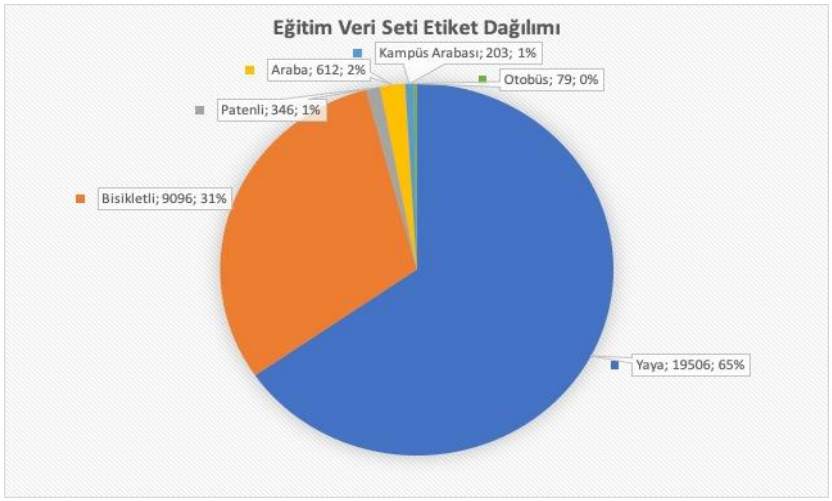
sayısının önemini anlayabilmek için test veri seti ve toplam veri setinin gösterimi tercih edilmiştir.

Eğitim esnasında aşırı öğrenmeye izin vermeyecek şekilde nesne sınıfına ait örnek gösterimi ne kadar fazla olursa modelin başarısı da o oranda yükselmektedir. Bu nedenle etiket sayısı model eğitimi için oldukça önemlidir. Veri setinde görüntü başına ortalama 13 olmak üzere toplamda 42,505 nesne etiketi bulunmaktadır. Ancak Tablo 4.2'de de görüldüğü üzere yaya ve bisikletli sınıfı dışındaki gösterim temsil düzeyi oldukça düşük olan sınıflar eğitimde doğruluk oranını düşüren etki oluşturmaktadır.

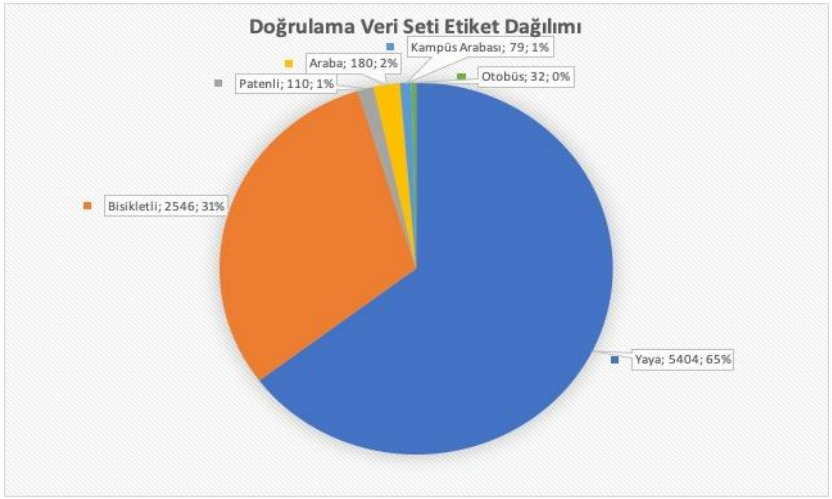
Tablo 4.2. Stanford Drone Veri Setinin Etiket Dağılımı

Etiket Adı	Test Veri Setindeki Etiket Sayısı	Veri Setindeki Toplam Etiket Sayısı	Test Veri Setindeki Etiket Sayısının Toplam Etiket Sayısına Oranı
Pedestrian (Yaya)	2773	27,683	%10,02
Biker (Bisikletli)	1322	12,964	%10,2
Car (Araba)	109	901	%12,1
Skater (Patenli)	52	508	%10,24
Cart (Araba)	46	328	%14,02
Bus (Otobüs)	10	121	%8,26
TOPLAM	**4213**	**42505**	**%10,14**

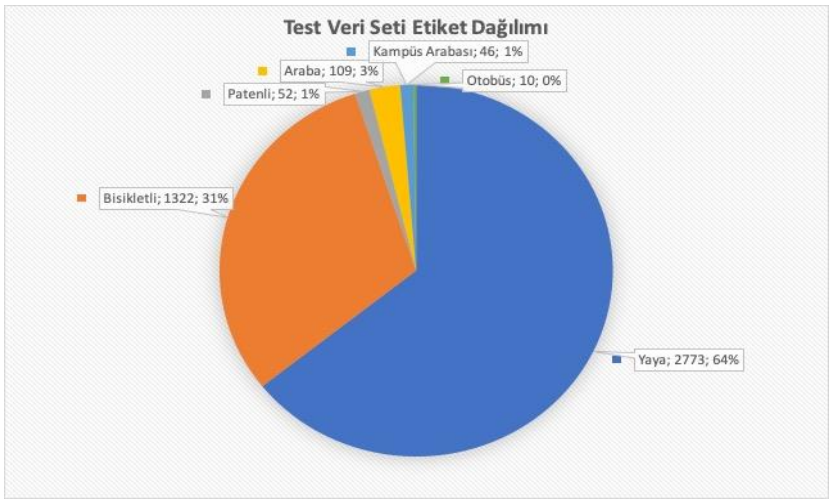
Şekil 4.2'de veri setinin eğitim, test ve doğrulama olarak dağılım grafiği gösterilmiştir. Veri seti özellikle yaya ve bisikletli sınıflarının yoğun olduğu diğer sınıfların oldukça düşük temsil edildiği bir yapıya sahiptir. Modellerin değerlendirme aşamasında bazı nesne sınıflarının test veri setinde temsilinin düşük olması başarı oranını olumsuz etkilemektedir.



(a) Eğitim Veri Setinin Etiket Dağılımı



(b) Doğrulama Veri Setinin Etiket Dağılımı



(c) Test Veri Setinin Etiket Dağılımı

Şekil 4.2. Stanford veri setinin eğitim, doğrulama ve test dağılım grafiği

Stanford Üniveristesi drone veri seti DOTA veri setine göre görüntü sayısı yaklaşık iki katı olmasına rağmen nesnelerin etiket sayısı olarak oldukça az etiket bulundurmaktadır. Ancak nesne sınıf sayısı daha az olduğu için veri setinin genel olarak model performansında olumlu etkisi gözlenmiştir. Şekil 4.3'te veri setine ait örnek görüntüler gösterilmiştir. Bu veri setinin modellerle eğitimi esnasında görüntü boyutları 640x640 piksel olarak yeniden ayarlanmıştır.



Şekil 4.3. Stanford Drone veri setinden örnek görüntüler

Sinir ağı mimarileri veri setini farklı formatlarda kullandıkları için veri seti etiketleri YOLOv5 Pytorch için .txt uzantılı olarak diğer mimariler için JSON dosyası olarak hazırlanmıştır.

Evrişimsel sinir ağı modellerinin eğitimi için yüksek performanslı işlemci ve ekran kartları kullanılmaktadır. Modellerin eğitimi ve test çalışmaları için Google tarafından sunulan bulut platform üzerinden özellikle yüksek seviye ekran kartı kullanımı sağlayan Google Colaboratory ortamı kullanılmıştır. Modellerin eğitimi için kullanılan Google Colaboratory uzak sunucu bilgisayarının özellikleri; işletim sistemi Linux-4.19.112+-x86_64-with-Ubuntu-18.04-bionic ve ekran kartı modeli Tesla P100-PCIE-16GB şeklindedir.

4.2. Değerlendirme Metrikleri

Her modelin eğitim süreci ve sonunda elde ettiği başarı ya da performansı ölçütlere uygun olarak değerlendirilmesi gerekmektedir. Modelin verimliliği ve performansı farklı değerlendirme araçları ile gösterilmektedir. Bu bölümde çalışmalarda kullanılan modellerin değerlendirilmesi için gerekli araçlara değinilmiştir.

4.2.1. Hata Matrisi

Nesne tespit ve algılama çalışmalarında değerlendirme aracı olarak hata matrisi ölçeği kullanılmaktadır. Tablo 4.3'te gösterilen bu matrise göre sınıflandırmada 4 durum söz konusudur. Sınıflandırmada başarıyı artıran iki durum vardır. Bunlar; doğru pozitif (true positive) ve doğru negatif (true negative) durumlarıdır. Bir mail sisteminde spam ve spam değil olarak iki durum barındıran sınıflandırma yaptığımızda doğru pozitif durumu spam olan mailleri modelin de spam olarak tahmin etmesidir. Doğru negatif durumu ise spam olmayan mailleri modelin de spam değil olarak tahmin etmesidir. Bunun dışında yanlış pozitif (false positive) durumu spam değil olarak etiketlenmiş maili modelin spam olarak tahmin etmesidir. Yanlış negatif (false negative) durumu ise spam olarak etiketlenmiş maili modelin spam değil olarak tahmin etmesidir. Bilgisayarlı görüde hem nesne algılama hem de nesne yerelleştirme yani algılanan nesnenin çevresinde sınırlayıcı kutu çizme çalışması söz konusu olduğu için hata matrisinin değerlendirme metriği olarak yetersiz kaldığı durumlar vardır. Ancak model başarısı için yeterli düzeyde değerlendirme yapma imkânı sağlamaktadır.

Tablo 4.3. Hata Matrisi

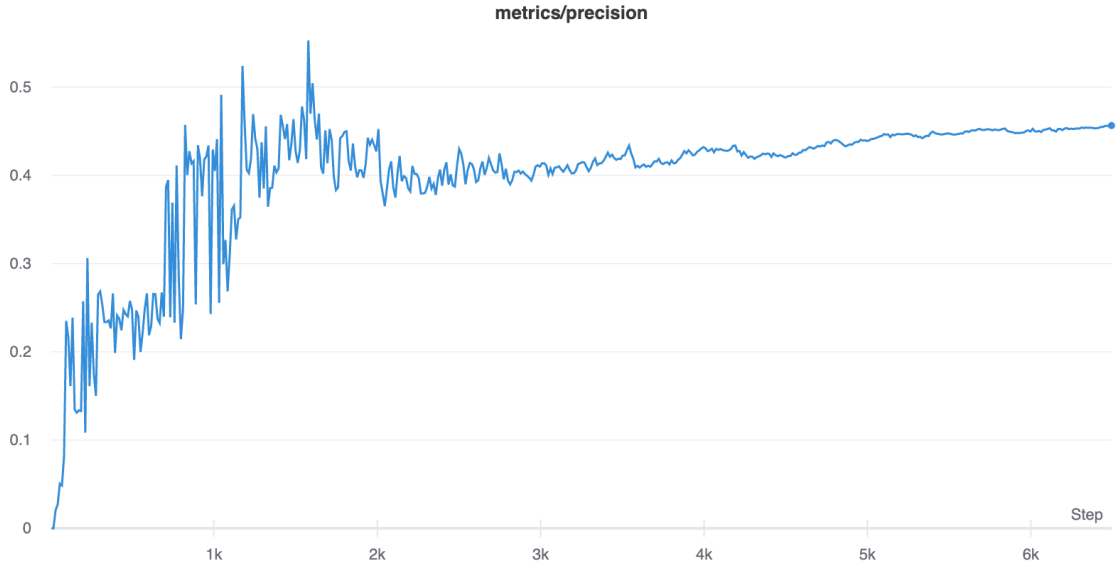
HATA MATRİSİ		**Tahmin Edilen Sınıf**	
		(+)	**(-)**
Gerçek Sınıf	**(+)**	**True Positive** Gerçek değerin ve tahmin edilen değerin 1 olduğu sınıf	**False Negative** Gerçek değerin 1 ve tahmin edilen değerin 0 olduğu sınıf
	(-)	**False Positive** Gerçek değerin 0 ve tahmin edilen değerin 1 olduğu sınıf	**True Negative** Gerçek değerin ve tahmin edilen değerin 0 olduğu sınıf

4.2.2. Kesinlik

Kesinlik (precision), değeri pozitif sınıfa ait olan ve pozitif sınıf olarak tahmin edilen örneklerin oranıdır. Denklem 4.1'de kesinlik değerinin hesaplanma formülü gösterilmiştir.

$$Kesinlik\ (Precision)\ = \frac{Doğru\ Pozitif}{(Doğru\ Pozitif\ +\ Yanlış\ Pozitif)} \tag{4.1}$$

Şekil 4.4'te YOLOv5 modelinin Stanford drone veri seti kullanılarak yapılan eğitim sürecine ait kesinlik grafiği gösterilmektedir. Kesinlik ölçütü doğruluk oranını etkileyen bir değer olduğu için artış göstermesi doğruluk oranının da arttığını göstermektedir.



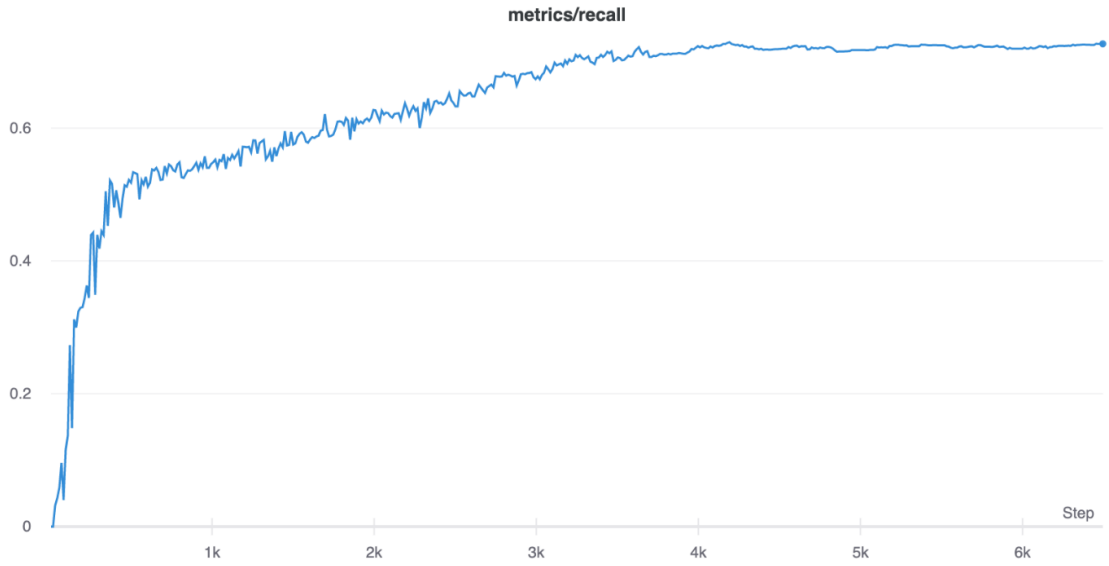
Şekil 4.4. YOLOv5 modelinin kesinlik grafiği

4.2.3. Duyarlılık

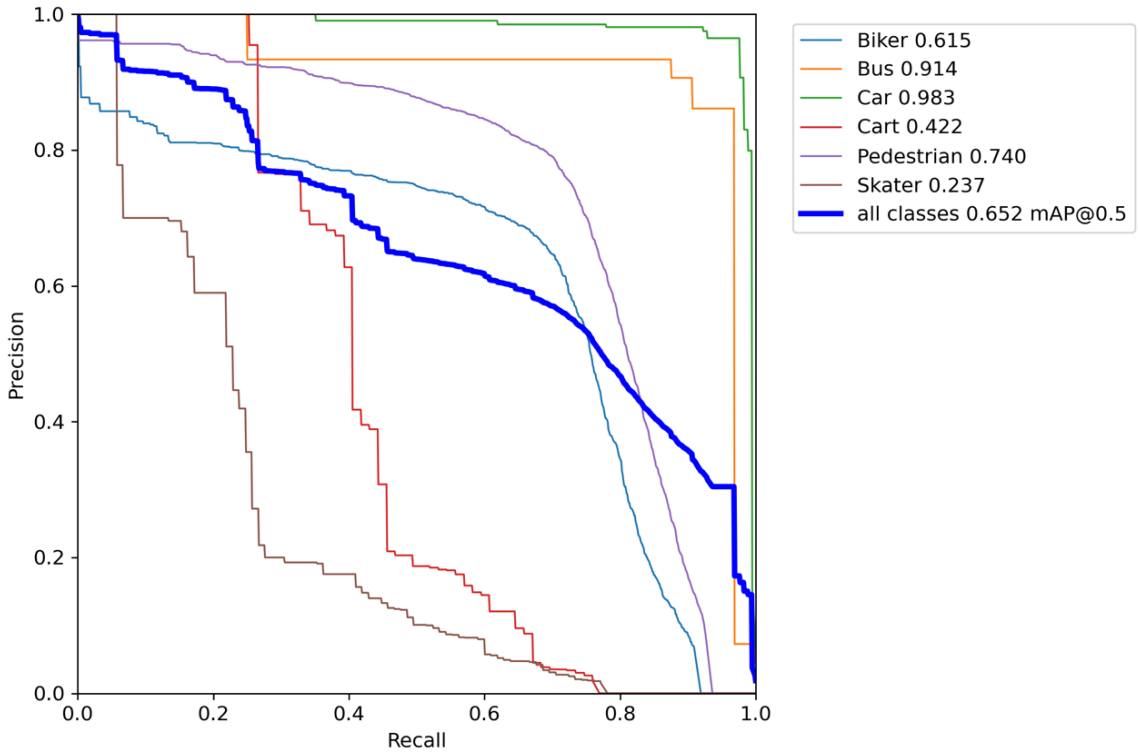
Duyarlılık (recall), gerçek değer olarak pozitif sınıfa atanmış nesnelerin ya da değerlerin ne kadar iyi tahmin edildiğini göstermektedir. Denklem 4.2'de duyarlılık değerinin hesaplanma formülü gösterilmiştir.

$$Duyarlılık\ (Recall)\ = \frac{Doğru\ Pozitif}{(Doğru\ Pozitif\ +\ Yanlış\ Negatif)} \tag{4.2}$$

Şekil 4.5'te YOLOv5 modelinin Stanford drone veri seti kullanılarak yapılan eğitim sürecine ait duyarlılık grafiği gösterilmektedir. Duyarlılık ölçütü de pozitif sonuçları kullanarak doğruluk oranını etkileyen bir değer olduğu için artış göstermesi doğruluk oranının da arttığını göstermektedir.



Şekil 4.5. YOLOv5 modelinin duyarlılık grafiği



Şekil 4.6. YOLOv5 modelinin 1000 epok eğitiminde kesinlik-duyarlılık eğrisi

Şekil 4.6'da ise YOLOv5 modelinin Stanford drone veri seti kullanılarak yapılan 1000 epok eğitim sürecinde kesinlik ve duyarlılık değerlerinin korelasyonunu gösterecek şekilde değişim grafiği gösterilmektedir. Kesinlik-Duyarlılık eğrisinde modelin başarısı değerlendirilirken kesinlik değerinin yüksek olmasına ve duyarlılık değerinin de artış göstermesine dikkat edilir. Buna göre eğitim süreci sonunda en başarılı kesinlik/duyarlılık değerine sahip nesne araba sınıfı olurken en başarısız nesne sınıfının patenli sınıfı olduğu görülmektedir. Tüm sınıfların kesinlik/duyarlılık grafiğine göre orantılı olarak ortalama doğruluk oranları da belirlenmektedir.

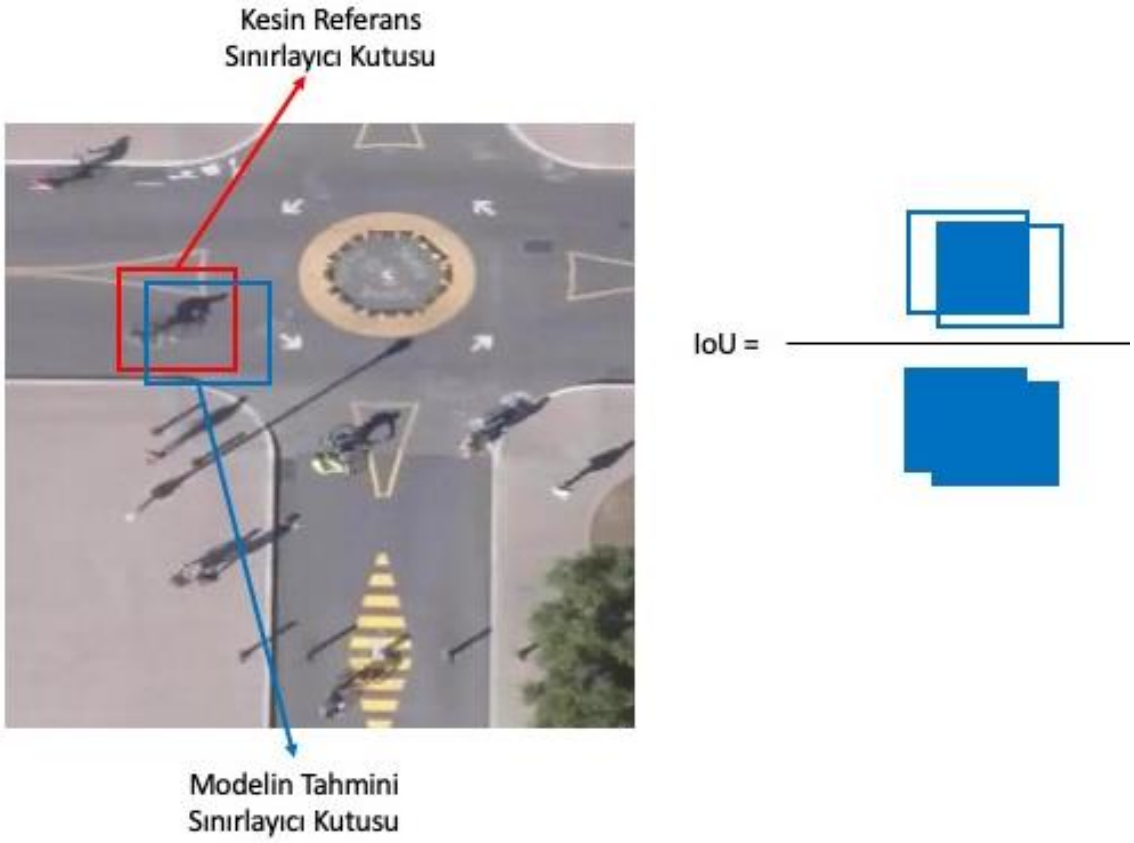
4.2.4. Doğruluk

Doğruluk (accuracy), doğru olarak tahmin edilen değerlerin toplam tahmin değerlerine oranıdır. Denklem 4.3'te kesinlik değerinin hesaplanma formülü gösterilmiştir.

$$Doğruluk\ (Accuracy)\ = \frac{Doğru\ Tahminler}{Toplam\ Tahminler} \tag{4.3}$$

4.2.5. Ortalama Doğruluk Oranı

Nesne tespit ve konum bulma çalışmalarında kesinlik ve duyarlılık gibi ölçütler tam olarak istenilen değerlendirmeyi yapamamaktadır. Model performansı değerlendirilirken iki durum göz önünde bulundurulur. Bunlar doğru tahmin edilen nesne sayısı ve tahmin edilen nesnelerin konumudur. Dolayısıyla bu aşamada test veri kümesinde bulunan ve olması gereken yani kesin referans olarak adlandırılan görüntüler değerlendirmeye alınır. Kesin referans ile modelin tahmini arasındaki benzerlik Jaccard indeksi ile ölçülür. Bu ölçümde kesin referans ile model tahmini arasındaki birleşim üzerinden kesişim (IoU) değeri baz alınır. Skor kesin referans sınırlayıcı kutusu ile model tahmininin sınırlayıcı kutusunun kesişim değerinin birleşim değerine oranı olarak hesaplanmaktadır [66]. Şekil 4.7'de Stanford drone veri setinden alınan bir görüntü kullanılarak birleşim üzerinden keşişim skoru gösterilmektedir.



Şekil 4.7. Birleşim Üzerinden Kesişim Gösterimi

4.3. Veri Setlerinin Aynı Modele Göre Performans Karşılaştırması

Hava fotoğrafları kullanılarak farklı evrişimsel sinir ağı modelleri deneysel çalışmaya tabi tutulmuş ve elde edilen sonuçlar verimlilik, hız ve doğruluk olarak karşılaştırılmıştır. Kullanılan evrişimsel sinir ağı modelleri Bölüm 3.6'da ayrıntılı olarak anlatılmıştır. Evrişimsel sinir ağlarında bir ağın eğitimi esnasında modelin çalışmasını etkileyen farklı parametreler vardır. Hiperparametre olarak adlandırılan bu değişkenlerin doğru bir şekilde ayarlanması gerekir. Hiperparametreler veri setine ve kullanılan modele göre değişiklik gösterir ve ince ayar gerektirir. Öğrenme oranı, momentum, en iyileme gibi hiperparametrelerin eğitim üzerinde büyük etkisi vardır ve farklı modellerde büyük farklılıklar gösterebilir. Tablo 4.4'te uygulama süresince kullanılan modellere ait hiperparametre değerleri gösterilmektedir.

Tablo 4.4. Modellere ait hiperparametrelerin gösterimi

Hiperparametreler	YOLOv5	YOLOv3	RetinaNet	EfficientDet
Öğrenme Oranı	0.01	0.001	0.0001	0.0001
Momentum	0.9	0.9	0.9	0.9
Yığın Boyutu	24	16	8	24
Aktivasyon Fonksiyonu	Leaky ReLU	Leaky ReLU	ReLU	SILU
En İyileme	Adam & SGD	Adam & SGD	SGD	SGD

Oluşturulan iki farklı veri setinin YOLOv5 modeli yapılan eğitim süreci sonunda elde edilen sonuçlar Tablo 4.5'te ayrıntılı olarak gösterilmiştir. Alınan sonuçlara göre aynı model ve hiperparametreler kullanılarak sürdürülen eğitimde veri setindeki görüntü sayısının daha fazla olması eğitim sürecinin en az 2 kat daha uzun sürmesine neden olmuştur. Dolayısıyla modelin eğitim süresini giriş görüntüsü boyutları ve görüntü sayısı orantılı olarak etkilemiştir. Her iki veri setiyle 640x640 piksel boyutlarında eğitim süreci yürütülmüş ve DOTA veri setinin eğitim süreci sonunda ortalama doğruluk oranı 0,36 olarak belirlenirken Stanford drone veri setinin ortalama doğruluk oranı 0,69 olarak tespit edilmiştir. Eğitimin Stanford drone veri setiyle yapılması performans olarak daha iyi sonuç alınmasını sağlamıştır.

Tablo 4.5. İki farklı veri setinin YOLOv5 modeli ile eğitim sonuçları

Model Adı	Veri Seti	Yığın Sayısı (Batch Size)	Devir Sayısı (Epok)	Eğitim Süresi	Ortalama Doğruluk Oranı (mAP)
YOLOv5 Pytorch	DOTA	24	500	1 Saat 49 Dakika	0.360
YOLOv5 Pytorch	Stanford Drone	24	500	3 Saat 56 Dakika	0.699

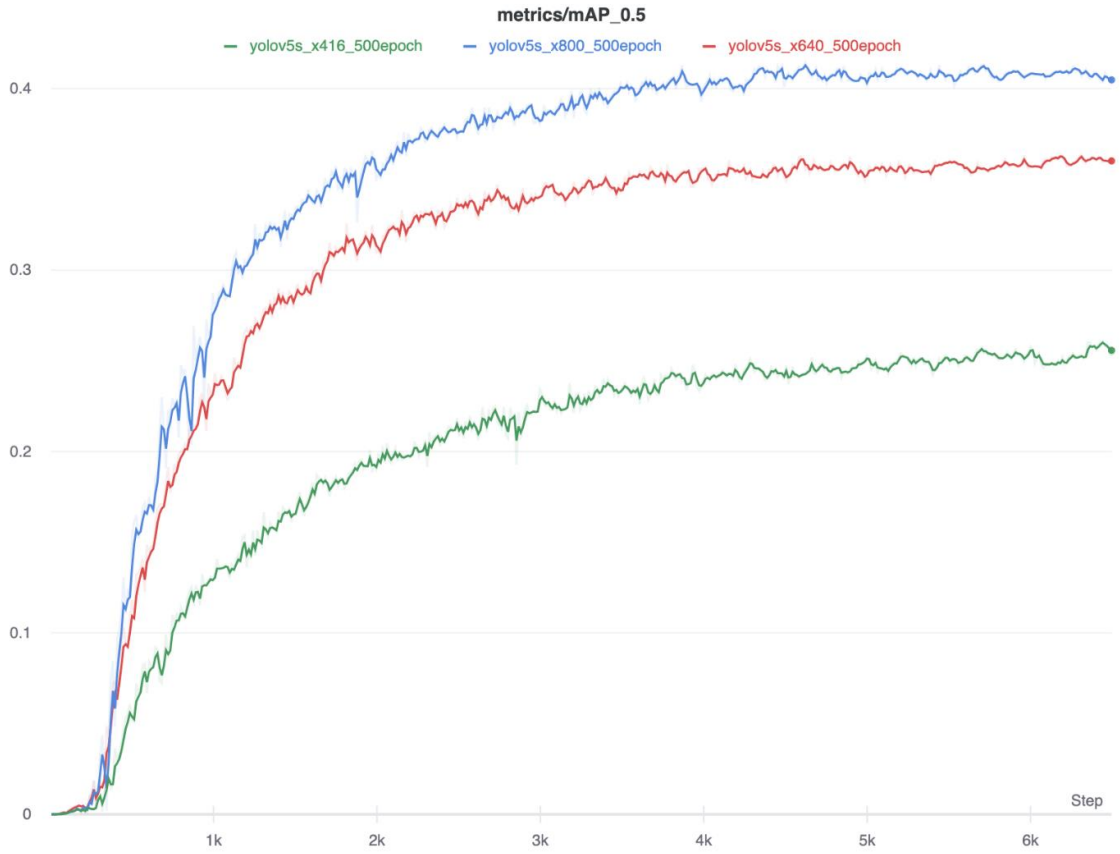
YOLOv5 modeli ile yapılan eğitim çalışmalarında görüntü boyutunun doğruluk, hız ve verimlilik konusunda oluşturduğu etkiyi anlayabilmek amacıyla önce görüntü boyutu yükseklik ve genişlik olarak 640 piksel olarak ayarlanarak eğitime tabi tutulmuştur. Daha sonra diğer tüm bileşenler aynı kalmak şartıyla sadece görüntü boyutu yükseklik ve genişlik olarak 416 ve 800 piksel ayarlanarak eğitime tabi tutulmuştur. Giriş görüntüsü boyutları küçüldükçe modelin eğitimi hızlanmıştır. Ancak genişlik ve yükseklik değeri 800 piksel olarak alınan giriş görüntüsünde eğitim süresi hem oldukça uzun sürmüş hem de aşırı ekran kartı kullanımı hatası vermiştir. Elde edilen sonuçlara göre eğitim süresi ve ortalama doğruluk oranı görüntü boyutu arttığı için artış göstermiştir. Değerlendirmelerin daha net ve anlaşılır olabilmesi için veri setleri oluşturulurken herhangi bir veri artırma (data augmentation) yöntemi kullanılmamıştır. DOTA veri setinin farklı çözünürlük ve epok değerleri ile YOLOv5 modelinde yapılan eğitim sürecinin sonuçları Tablo 4.6'da ayrıntılı olarak gösterilmiştir.

Tablo 4.6. YOLOv5 modeli farklı çözünürlük ve epok değerleri ile çalışma sonuçları

Model Adı	Görüntü Boyutu (Image Size)	Yığın Sayısı (Batch Size)	Devir Sayısı (Epok)	Eğitim Süresi	Ortalama Doğruluk Oranı (mAP)
YOLOv5 Pytorch	416 x 416px	24	500	1 Saat 48 Dakika	0.254
YOLOv5 Pytorch	640 x 640px	24	500	1 Saat 49 Dakika	0.360
YOLOv5 Pytorch	800 x 800px	24	500	4 Saat 4 Dakika	0.404
YOLOv5 Pytorch	416 x 416px	24	1000	3 Saat 14 Dakika	0.277
YOLOv5 Pytorch	640 x 640px	24	1000	6 Saat 10 Dakika	0.380

İlk geliştirilen YOLO modeli 448x448 piksel, bu modelin geliştirilerek iyileştirildiği YOLOv2 modeli ise varsayılan giriş görüntüsü boyutu olarak 416x416 piksel kabul etmektedir. Diğer evrişimsel sinir ağı modellerine göre eğitim süresinin oldukça hızlı sürdüğü YOLOv5 modelinde diğer bütün durumların aynı olması kaydıyla farklı çözünürlüklerin giriş görüntüsü olarak verilmesiyle aynı devir (epok) sayısında eğitime tabi tutulması sonucu görüntü boyutunun yüksek olması eğitim süresini artırmasına rağmen ortalama doğruluk oranının da artmasıyla daha başarılı hale gelmiştir.

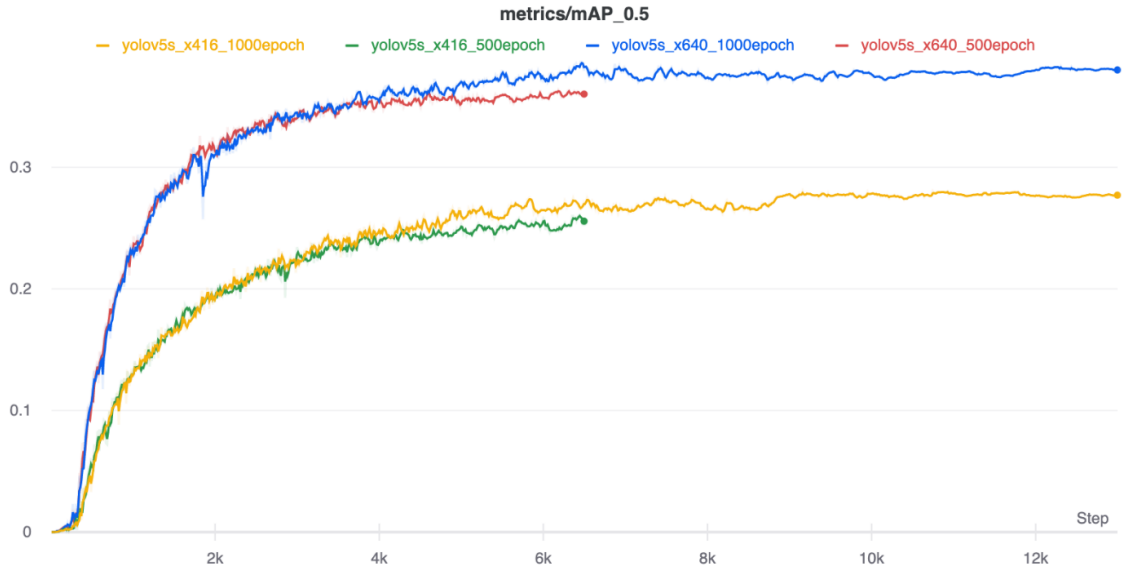
Her üç denemede de 500 epok sonunda grafik eğrisi yatay bir hal almış ve doğruluk oranının artışı oldukça düşük kalmıştır. En küçük çözünürlük değerlerinde modelin eğitim sonucunda ortalama doğruluk oranı (mAP) %25,4 olarak gerçekleşirken en yüksek çözünürlük değerinde ortalama doğruluk oranı (mAP) yaklaşık %41 olarak gerçekleşmiştir. DOTA veri seti kullanılarak farklı giriş görüntüsü çözünürlükleri ile eğitim sonuçlarına ait ortalama doğruluk oranları Şekil 4.8'de ayrıntılı olarak gösterilmiştir.



Şekil 4.8. YOLOv5 modelinin farklı çözünürlüklerle eğitim sonuçları

YOLOv5 modelinin Google Colaboratory ortamında Tesla P100-PCIE-16GB ekran kartı ile farklı çözünürlük ve farklı devir (epok) sayıları ile eğitim sonuçları Şekil 4.9'da gösterilmiştir. Sonuçlara göre hem 416x416 piksel hem de 640x640 piksel boyutlarında giriş görüntüsü modele sunulduğunda ilk 500 epok boyunca ortalama doğruluk oranı artışının gözlemlendiği, 500 epok sonrasında ise her iki grafiğin yatay

seyir aldığı ve artışın çok fazla olmadığı tespit edilmiştir. Giriş görüntüsü boyutları 416x416 piksel ile eğitime tabi tutulan model de epok sayısının 2 katına çıkarılması eğitim süresini yaklaşık %80 oranında artırmasına karşılık ortalama doğruluk oranını yaklaşık %2,3 artırmıştır. Aynı şekilde giriş görüntüsü boyutları 640x640 piksel ile eğitime tabi tutulan modelin epok sayısının 2 katına çıkarılması eğitim süresini yaklaşık olarak 2,5 kat artırırken ortalama doğruluk oranını yaklaşık %2 oranında artırmıştır. Bu sonuçlara göre modelin 500 ile 1000 epok arasında eğitilmesi çalışma süresi olarak dikkat çekici şekilde fark ortaya çıkarmamıştır. Ancak bilgisayarlı görüde nesne algılama çalışmalarında ortalama doğruluk oranının %2 civarında artış göstermesi de önemli bir başarı yükselmesi olarak görülebilir.



Şekil 4.9. YOLOv5 modelinin farklı çözünürlük ve epok sayıları ile eğitim sonuçları

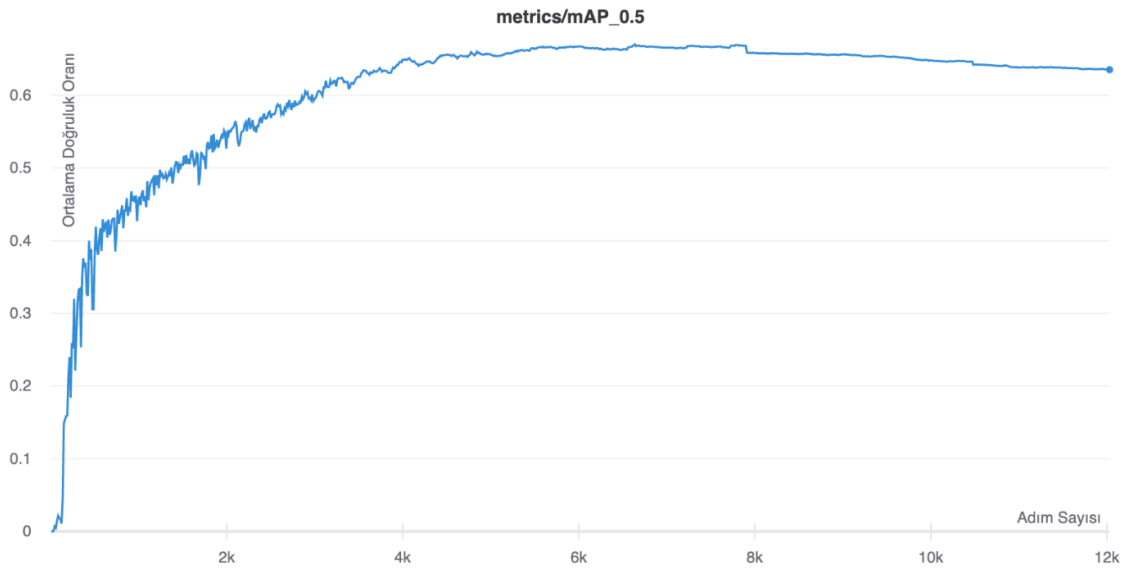
Önce iki veri setinin karşılaştırması daha sonra ise farkı anlayabilmek için DOTA veri setinin farklı çözünürlük ve epok değerleri ile tekraren eğitime tabi tutulması süreci YOLOv5 modeli ile yapılmıştır. Bu süreçte model seçiminde hız ve değerlendirme ölçütlerinin ayırt ediciliği dikkate alınmıştır.

4.4. Hava Fotoğrafları için Modellerin Karşılaştırılması

Tez çalışmasında kullanılan DOTA ve Stanford Üniversitesi drone veri setinin başarı ve verimlilik olarak karşılaştırılması yapıldıktan sonra performans ve verim açısından daha faydalı olduğu değerlendirilen Stanford drone veri seti modellerin çalışması sırasında kullanılmıştır. Yapılan çalışmalara ait sonuçlar ayrıntılı olarak analiz edilmiştir.

4.4.1. YOLOv5 Modelinin Çalışma Sonuçları

Stanford drone veri setinin daha iyi performans gösterdiği anlaşıldığı için YOLOv5 modeli ile en iyi doğruluk oranına ulaşması amacıyla yığın boyutu 24 ve epok sayısı 1000 olacak şekilde eğitim süreci başlatılmıştır. Eğitim süreci yaklaşık olarak 6 saat 21 dakika sürmüş ve başarı olarak 0,69 ortalama doğruluk oranı elde edilmiştir. Bu sonuçlara göre model eğitim ve doğrulama veri setinde bulunan görüntülerdeki nesneleri yaklaşık %69 oranında sınıflandırma ve yerelleştirme yapabilmiştir. Şekil 4.10'da veri setinin eğitim sonucunda aldığı ortalama doğruluk oranı grafiği gösterilmektedir.

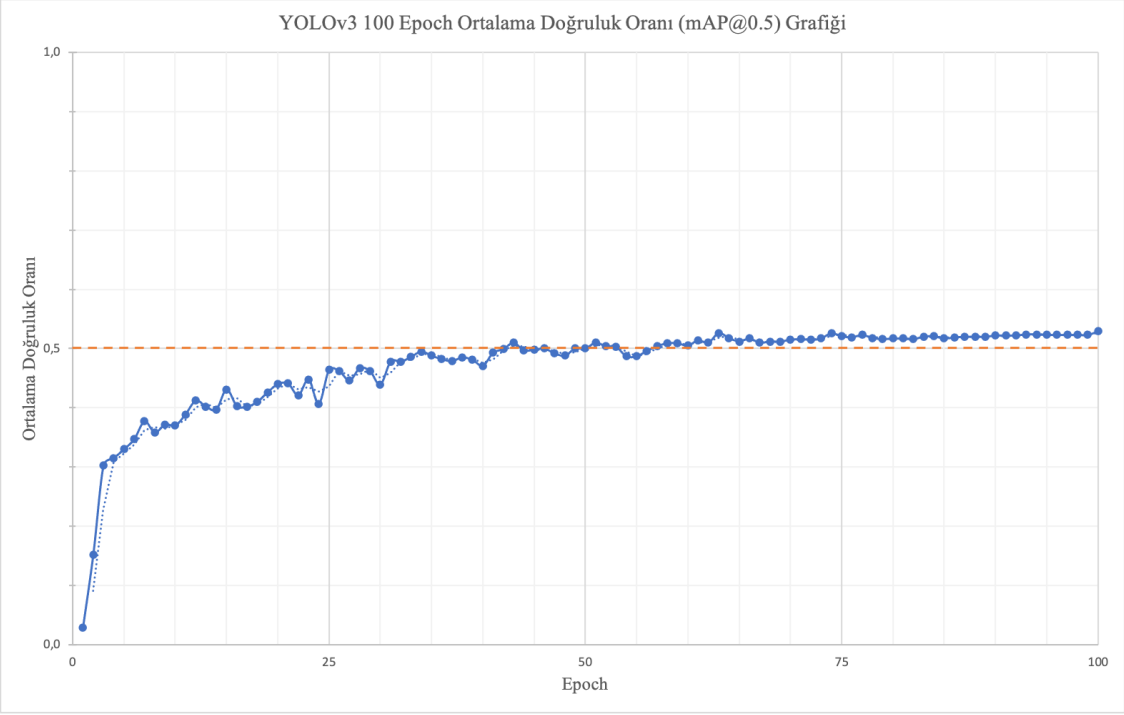


Şekil 4.10. YOLOv5 modelinin 1000 epok eğitim sonucu

YOLOv5 modelinin tasarımı gereği eğitim sürecinde her epok sonunda doğrulama veri seti kullanılarak ortalama doğruluk oranı belirlenmiştir. Bu durum diğer modellerde değişiklik göstermektedir.

4.4.2. YOLOv3 Modelinin Çalışma Sonuçları

Stanford drone veri seti ile YOLOv3 modelinin 100 epok olarak ayarlanan eğitimi yaklaşık 2 saat sürmüş ve eğitim esnasında modelin iyileştirilmesi için her epok sonunda değerlendirme veri seti ile o döngüye ait eğitim durumu gözden geçirilmiştir. Elde edilen eğitim çıktılarına göre ortalama doğruluk oranı 0,52 olarak gerçekleşmiştir. Modelin YOLOv5 modeli ile çalıştırılan epok sayısı karşılaştırıldığında 1/10'u kadar epok değeri ile çalışmasına karşın ortalama doğruluk oranı olarak iyi bir başarı elde ettiği değerlendirilmektedir. Daha fazla epok değeri kullanılarak yapılan çalışmalarda ortalama doğruluk oranında belirgin bir artış gözlenmemiştir. Şekil 4.11'de YOLOv3 modelinin veri seti ile eğitimi sonunda elde edilen ortalama doğruluk oranı grafiğini göstermektedir.



Şekil 4.11. YOLOv3 modelinin 100 epok eğitim sonucu

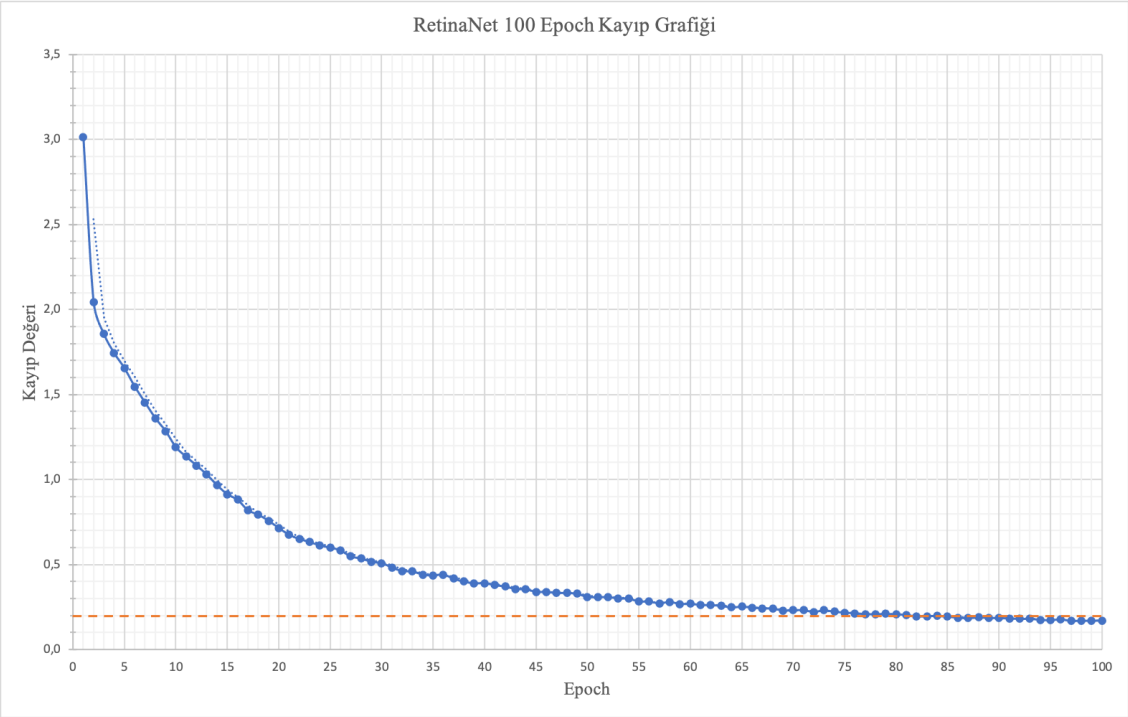
Modelin eğitimi tamamlandıktan sonra test görüntülerinin kullanıldığı değerlendirme çalışması yapılmış ve her nesne sınıfına ait ortalama doğruluk oranları ayrı ayrı elde edilmiştir. Modelin eğitimi sırasında ortalama doğruluk oranı 0,52 değerine kadar çıkmış olmasına rağmen test görüntülerinden elde edilen doğruluk oranı 0,36 olarak belirlenmiştir. Her nesne sınıfına ait ortalama doğruluk oranı ve sınıfın etiket sayısı Tablo 4.7'de gösterilmiştir. Tablo verileri incelendiğinde ortalama doğruluk oranına artırıcı etki yapan sınıfların yaya, bisikletli, araba ve otobüs sınıfları olduğu ve kampüs arası ile patenli sınıflarının düşük seçicilik nedeniyle ortalama doğruluk oranı üzerinde olumsuz etki yaptığı görülmüştür.

Tablo 4.7. YOLOv3 modelinin test veri seti ait ortalama doğruluk oranı tablosu

Etiket Adı	Test Veri Setindeki Etiket Sayısı	Sınıfa Ait Ortalama Doğruluk Oranı
Pedestrian (Yaya)	2773	0,36
Biker (Bisikletli)	1322	0,33
Car (Araba)	109	0,81
Skater (Patenli)	52	0,00
Cart (Kampüs Arabası)	46	0,19
Bus (Otobüs)	10	0,48
TOPLAM	**4312**	**0,364**

4.4.3. RetinaNet Modelinin Çalışma Sonuçları

RetinaNet modelinin Keras kütüphanesi tasarımı kullanılarak Stanford drone veri seti ile yapılan eğitim çalışması yaklaşık olarak 5 saat 40 dakika sürmüştür. Eğitim toplamda 100 epok olarak sürmüş ve elde edilen kayıp fonksiyonu grafiği Şekil 4.12'de gösterilmiştir. Eğitim tamamlandıktan sonra elde edilen öğrenilmiş değerleri barındıran .h5 uzantılı ağırlık dosyası değerlendirme veri setinde bulunan 642 adet görselde değerlendirmeye tabi tutulmuştur.



Şekil 4.12. RetinaNet modelinin 100 epok eğitim sonucu

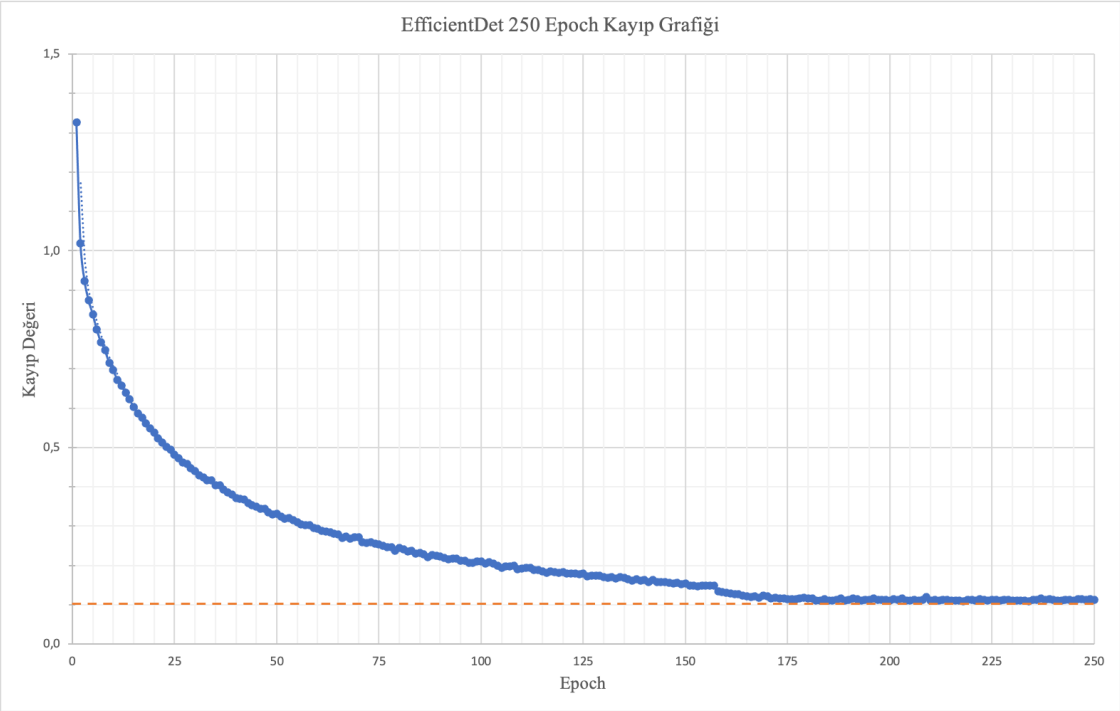
Her modelin tasarımı gereği farklı değerlendirme çıktıları oluşmaktadır. RetinaNet modelinde eğitim aşaması sonrasında kayıp fonksiyonuna uygun olarak yapılan değerlendirmede modelin ortalama doğruluk oranı 0,518 olarak çıkmıştır. Model değerlendirme aşamasında 642 adet görselden oluşan değerlendirme veri setini kullanmış ve her nesne sınıfına ait ortalama doğruluk oranlarını ayrı ayrı belirlemiştir. Tablo 4.8'de her nesne sınıfının ortalama doğruluk oranı ve nesne sınıfına ait toplam etiket sayısı ayrıntılı olarak gösterilmiştir. Ortalama doğruluk oranları ve etiket sayıları incelendiğinde patenli nesne sınıfının hem düşük temsil hem de yayalardan ayırt edilme zorlukları nedeniyle doğruluk oranları oldukça düşük çıkmıştır. Bu durum genel doğruluk oranının da düşmesine neden olmaktadır. Bunun dışında otobüs gibi ayırt ediciliği yüksek nesnelerin temsil sayısı az olmasına rağmen doğruluk oranı oldukça yüksek çıkmıştır. En fazla temsil oranına sahip yayaların ortalama doğruluk oranının genel doğruluk oranına yakın çıkması bu sınıfın belirleyici olduğunu göstermektedir.

Tablo 4.8. RetinaNet modeli değerlendirme veri seti ortalama doğruluk oranı tablosu

Etiket Adı	Değerlendirme Veri Setindeki Etiket Sayısı	Sınıfa Ait Ortalama Doğruluk Oranı
Pedestrian (Yaya)	5404	0,520
Biker (Bisikletli)	2546	0,423
Car (Araba)	180	0,976
Skater (Patenli)	110	0,065
Cart (Araba)	79	0,286
Bus (Otobüs)	32	0,836
TOPLAM	**8351**	**0,518**

4.4.4. EfficientDet Modelinin Çalışma Sonuçları

Stanford drone veri seti ile EfficientDet-D0 modelinin 250 epok eğitimi yaklaşık 4 saat 14 dakika sürmüştür. Eğitim süresince elde edilen kayıp fonksiyonuna ait grafik Şekil 4.13'te gösterilmektedir. Eğitimde model tasarımı gereği ilk 150 devir öğrenme hızı $1x10^{-4}$ olarak başlamış, kayıp değerinin durumuna bağlı olarak kademeli bir şekilde $1x10^{-8}$'e kadar düşürülmüştür.



Şekil 4.13. EfficientDet modeli 250 epok eğitim sonucu

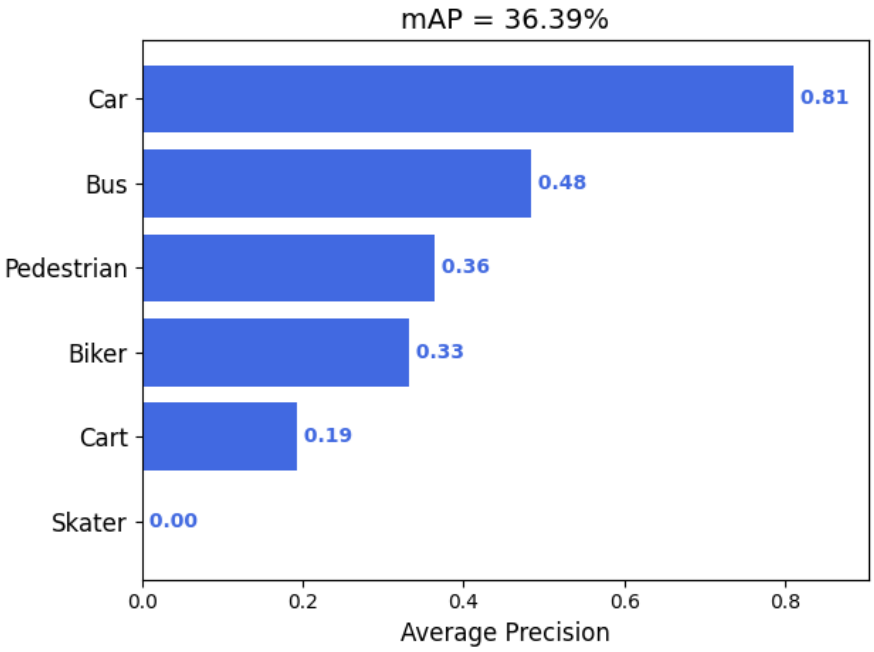
EfficientDet modelinin epok sayısının düşürülerek çalıştırılmasının ortalama başarı oranı üzerindeki etkisini tespit edebilmek için 100 epok olarak yapılan diğer eğitimi tamamlandıktan sonra test veri seti kullanılarak performansı değerlendirilmiş ve ortalama doğruluk oranı 0.223 olmuştur. Her nesne sınıfına ait doğruluk oranları ayrı ayrı çıkarılmış ve Tablo 4.9'da veri setindeki etiket sayısı ile birlikte gösterilmiştir. Bu model ile yapılan eğitimde en çok etikete sahip nesne sınıfı olan yayalarda oldukça düşük başarı elde edilmiştir. En dikkat çekici nokta ise çok fazla temsil oranı olmamasına rağmen araba sınıfının oldukça yüksek başarı elde etmesidir. Diğer sınıflarda nesne tespit başarı oranları oldukça düşük çıkmıştır.

Tablo 4.9. EfficientDet modelinin test veri seti ortalama doğruluk oranı tablosu

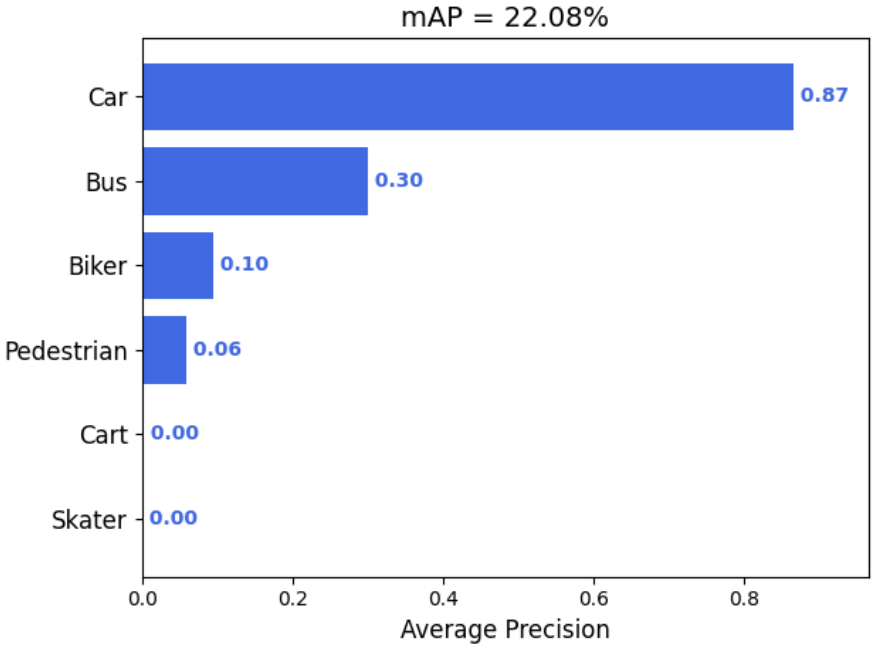
Etiket Adı	Test Veri Setindeki Etiket Sayısı	Sınıfa Ait Ortalama Doğruluk Oranı
Pedestrian (Yaya)	2773	0.059
Biker (Bisikletli)	1322	0.095
Car (Araba)	109	0,867
Skater (Patenli)	52	0,00
Cart (Araba)	46	0.003
Bus (Otobüs)	10	0.30
TOPLAM	**4312**	**0.223**

4.4.5. Tüm Modellerin Çalışma Sonuçlarının Karşılaştırılması

Bu aşamada YOLOv3 ve EfficientDet-D0 modellerinin Stanford drone veri seti ile eğitim sonuçları değerlendirme veri setleri kullanılarak karşılaştırılmış ve ortalama doğruluk oranı olarak YOLOv3 daha başarılı bulunmuştur. Modellerin başarı kıyaslaması Şekil 4.14'te gösterilmiştir.



(a) YOLOv3 modelinin kesinlik değeri sonuçları



(b) EfficientDet modelinin kesinlik değeri sonuçları

Şekil 4.14. YOLOv3 ile EfficientDet modellerinin karşılaştırılması

Şekil 4.14.(a) incelendiğinde YOLOv3 modelinin başarısını temsil oranı sadece %1-2 aralığında olan otobüs ve araba sınıflarının artırdığı görülmektedir. Değerlendirme ver setinde temsil oranı %65 olan yaya sınıfının kesinlik değeri 0,36 ve %31 olan bisikletli sınıfının kesinlik değeri 0,33 olmuştur. Bu durum yaya ve bisikletli sınıflarının genel başarı oranı üzerinde belirleyici etkisinin olduğunu göstermektedir.

Şekil 4.14.(b) incelendiğinde EfficientDet modelinin başarı oranı üzerinde değerlendirme veri setinde %1-2 aralığında temsil edilen otobüs ve araba sınıflarının artırıcı etkisi olmuştur. Değerlendirme veri setinin yaklaşık olarak %96'lık kesimini oluşturan yaya ve bisikletli sınıflarının kesinlik değerlerinin oldukça düşük çıkması genel başarı oranını da düşürmüştür.

Her iki modele ait sonuçlar incelendiğinde modellerin büyük nesnelere karşı tespit hassasiyeti aynı ölçüde yüksek olmasına karşın YOLOv3 modelinin küçük nesnelere karşı tespit oranı EfficientDet modeline kıyasla oldukça yüksek çıkmıştır. Bu durum YOLOv3 modelinin daha başarılı tespit sonuçları üretmesini sağlamıştır.

Tablo 4.10'da kullanılan 4 modelin de belirtilen hiperparametre değerleriyle çalıştırıldıktan sonra elde edilen eğitim süresi ve başarı oranları gösterilmiştir. Eğitim süresince sadece YOLOv5 modelinin epok değeri diğerlerinden yüksek tutulmuş ve elde edilen sonuçlar karşılaştırılmıştır.

Tablo 4.10. Tüm modellerin karşılaştırılması

Model Adı	Görüntü Boyutu	Yığın Sayısı	Epok Sayısı	Eğitim Süresi	Ortalama Doğruluk Oranı
YOLOv5 Pytorch	640x640	24	1000	6 saat 21 dakika	0,69
YOLOv3 Darknet	640x640	16	100	2 saat	0,52
RetinaNet Keras	640x640	8	100	5 saat 40 dakika	0,51
EfficientDet Pytorch	640x640	8	100	4 saat 14 dakika	0,22

Modellerin eğitim, değerlendirme test veri kümesi üzerinde çalışması tamamlandıktan sonra daha önce aynı veri setinden alınan ve modele gösterilmemiş olan toplam 20 adet görüntü etiketsiz bir şekilde modellerin nesne tespitine tabi tutulmuştur.

Görüntülerde yaya ve bisikletli sınıfları ağırlıklı olmak üzere toplamda 303 adet nesne etiketi bulunmaktadır. Seçilen görüntülerde dikkat çekici bir diğer husus araba sınıfına ait hiç nesne yoktur. Ancak kampüs arabası sınıfına ait 24 nesne bulunmaktadır. Tablo 4.11'de teste tabi tutulan 20 adet görüntü ile modellerin test sonucunda tahminlerinin sınıflara göre dağılımı gösterilmiştir. Tahmin görüntüleri tek tek incelendiğinde 4 modelden en düşük başarı sağlayan EfficientDet-D0 modelinin test görüntüleri ile yapılan tespit çalışmasında da başarısı oldukça düşük çıkmıştır. Bunun yanı sıra YOLO modellerinin nesne tespitinde oldukça başarılı olduğu ve insan gözüyle dahi kolayca ayırt edilemeyecek yaya ve bisikletli sınıflarını başarıyla tespit ettikleri anlaşılmıştır.

Modellerin eğitim süreci sonunda elde ettiği ortalama başarı oranları ile kıyaslandığında bütün modellerde pozitif yönde korelasyon olduğu söylenebilir. Eğitim süresi, model verimliliği, tahmin başarısı ve kod okunabilirliği olarak YOLOv5 modelinin diğer modeller karşısında üstün olduğu tespit edilmiştir. YOLOv5 modeli eğitim sürecinde elde ettiği başarıyı test görüntülerinde de aynı şekilde göstermiştir. Değerlendirmenin hata matrisi kullanılarak yapılması sonuçların anlaşılmasını kolaylaştırmaktadır. Buna göre hata matrisinde doğru pozitif olarak belirtilen ve modellerden beklenen gerçek etiket ile tahmin değerinin aynı olması durumu her sınıf için ayrı ayrı belirtilmiştir. Hata matrisinde yanlış negatif olarak belirtilen değerler, referans görüntüsünde etiketlenmemiş nesnelerin tahmin görüntülerinde sınırlayıcı kutu çizilerek tespit edilmesi durumudur. Yani orjinal görüntü içerisinde herhangi bir sınıfa ait etiketlenmemiş nesne tahmin görüntüsünde sınırlayıcı kutu ile belirtilmiş ise bu durum yanlış negatif olarak işaretlenmiştir. Yanlış pozitif durumu ise orjinal görüntüde ve tahmin görüntüsünde aynı yerde nesne tespit edilmiş ancak başka bir sınıfa ait tahminde bulunulmuştur. Yani nesne tespit edilmiş ancak yanlış algılanmıştır. Tabloda bu sonuçlar ayrıntılı olarak gösterilmiştir.

Tablo 4.11. Tüm modellerin test görüntüleri ile karşılaştırılması

Model Adı	Yaya	Bisikletli	Patenli	Araba	Kampüs Arabası	Otobüs	Yanlış Negatif	Yanlış Pozitif
Orijinal Etiket Sayısı	191	79	5	0	24	4	-	-
YOLOv5	148	61	0	0	7	0	14	77
YOLOv3	78	31	0	0	0	2	13	29
RetinaNet	111	69	0	0	0	1	74	48
EfficientDet	5	7	0	0	0	1	12	11

Test görüntülerinden her modelin nesne sınıflarına ait tahmin yüzdeleri Tablo 4.12'de ayrıntılı olarak gösterilmiştir. Çalışmalarda en başarılı sonuçları üreten YOLOv5 modelinin tahmin sonuçları da diğer modellere göre üstün çıkmıştır. Nesne sınıfları bazında detaylı olarak incelendiğinde YOLOv5 modelinin yaya sınıflarını tahmin etme konusunda diğer modellere göre üstün olduğu gözlemlenirken RetinaNet modelinin ise özellikle bisikletli sınıfının tahmininde en başarılı sonuçları ürettiği görülmüştür. Ancak araba, kampüs arabası, otobüs ve patenli sınıflarının düşük temsil edilmesi nedeniyle tahmin sonuçları da neredeyse tamamında sıfır olarak çıkmıştır. Bu durum da genel başarı oranı üzerinde olumsuz etki oluşturmuştur.

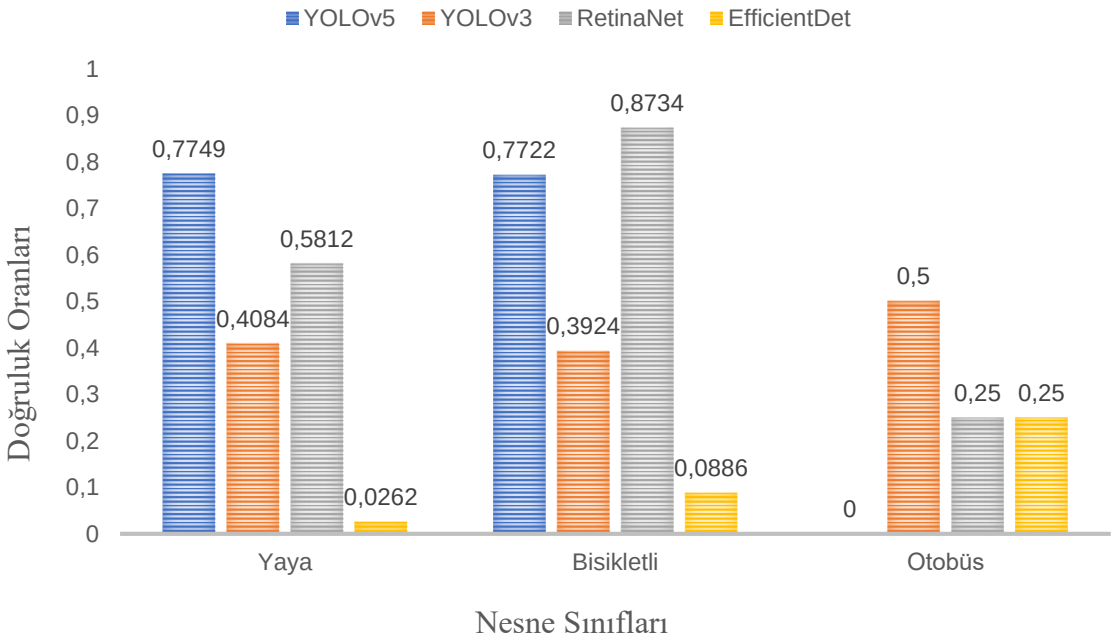
Tablo 4.12. Modellerin tahmin yüzdesi

Model Adı	Yaya	Bisikletli	Patenli	Araba	Kampüs Arabası	Otobüs	Ortalama Doğruluk Oranı
YOLOv5	%77	%77	0	0	%29	0	%70
YOLOv3	%41	%39	0	0	0	%50	%73
RetinaNet	%58	%87	0	0	0	%25	%59
EfficientDet	%0,2	%0,8	0	0	0	%25	%36

YOLOv5 modeli 20 adet görüntüde bulunan toplam 303 adet nesne için toplam 307 adet tahminde bulunmuş ve yaklaşık %70 oranında doğru olarak tespit etmiştir. YOLOv3 modeli toplam 153 adet tahminde bulunmuş ve yaklaşık %73 oranında doğru

olarak tespit etmiştir. RetinaNet modeli toplam 303 adet tahminde bulunmuş ve yaklaşık %59 oranında doğru olarak tespit etmiştir.

Şekil 4.15'te tahmin için modellere sunulan 20 adet görüntüye ait her modelin elde ettiği doğruluk oranları gösterilmiştir. Modellerin en başarılı olduğu yaya, bisikletli ve otobüs sınıfları gösterime tabi tutulmuştur. Diğer nesne sınıflarının başarı oranı çok düşük olduğu için gösterilmemiştir.

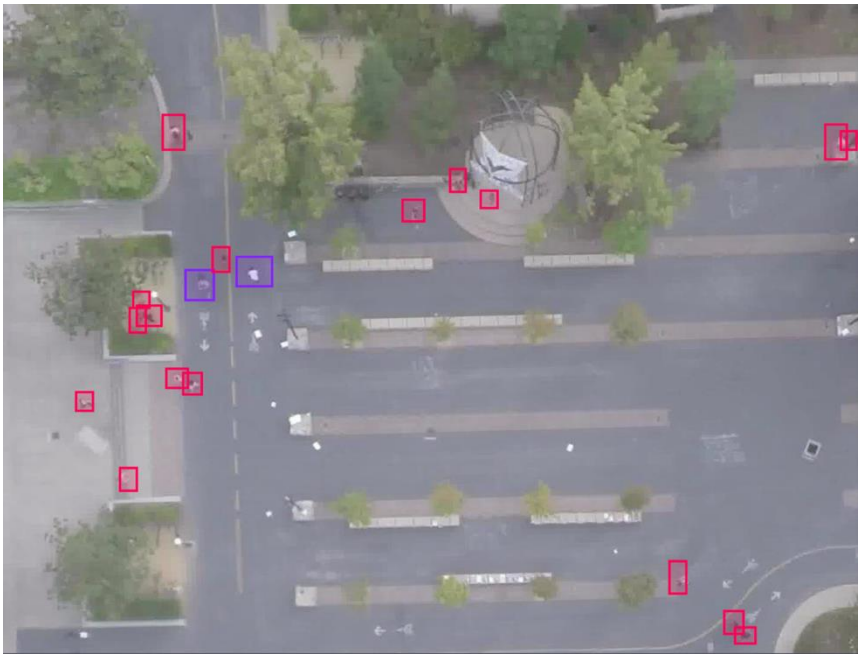


Şekil 4.15. Modellerin Başarı Oranları

Modellerin doğruluk oranları belirlenirken doğru olarak yapmış oldukları tahmin sayısının toplam tahmin sayısına oranı hesaplanmıştır. Modellerin başarılı tahminlerinin yanı sıra yanlış pozitif ve yanlış negatif tahminleri de hesaplamaya dahil edilerek doğruluk oranları belirlenmiştir. Grafik incelendiğinde yaya ve bisikletli nesne sınıflarında istikrarlı bir şekilde YOLOv5 modelinin oldukça başarılı olduğu görülmektedir. YOLOv3 modelinin YOLOv5 modeline göre daha yüksek doğruluk oranına ulaşmasının nedeni yarı yarıya daha az nesne tespitinde bulunmasına rağmen doğru tespit sayısının toplam tespit sayısına göre oranının yüksek olmasıdır. Yani YOLOv3 modelinin yanlış pozitif ve yanlış negatif tespit sayıları YOLOv5 ve RetinaNet modeline göre oldukça düşük çıkmıştır. Bu durum da modelin doğruluk oranını olumlu

yönde etkilemiştir. Bununla birlikte bisikletli nesne sınıfını tespit etmede en başarılı sonucu RetinaNet modelinin ürettiği anlaşılmaktadır. Otobüs nesne sınıfına ait 20 adet görüntüde sadece 4 adet etiketli nesne olması tahmin doğruluğunun belirleyici düzeyde yüksek çıkmasına neden olmaktadır. YOLOv3 modelinin 2 defa otobüs nesnesini doğru tahmin etmesi doğruluk oranının 0,50 çıkmasını sağlamıştır. EfficientDet modelinin genel başarı oranında görüldüğü gibi doğruluk oranında da diğer modeller karşısında oldukça başarısız sonuçlar ürettiği tespit edilmiştir.

Şekil 4.16'da en son aşamada kullanılan test görüntülerinden örnek alınan bir görüntünün YOLOv5 modeli ile yapılan tahmin sonuçlarına ait görüntü karşılaştırılmıştır.



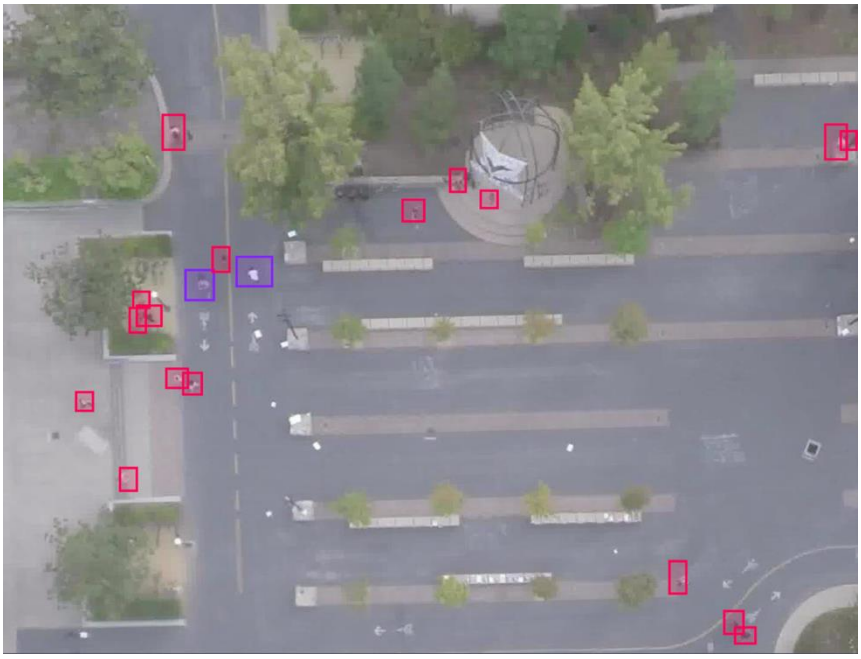
(a) Test Görüntüsü



(b) Tahmin Görüntüsü

Şekil 4.16. Test görüntüsünün YOLOv5 çıkarım görüntüsü ile karşılaştırılması

Şekil 4.17'de en son aşamada kullanılan test görüntülerinden örnek alınan bir görüntünün YOLOv3 modeli ile yapılan tahmin sonuçlarına ait görüntü karşılaştırılmıştır.



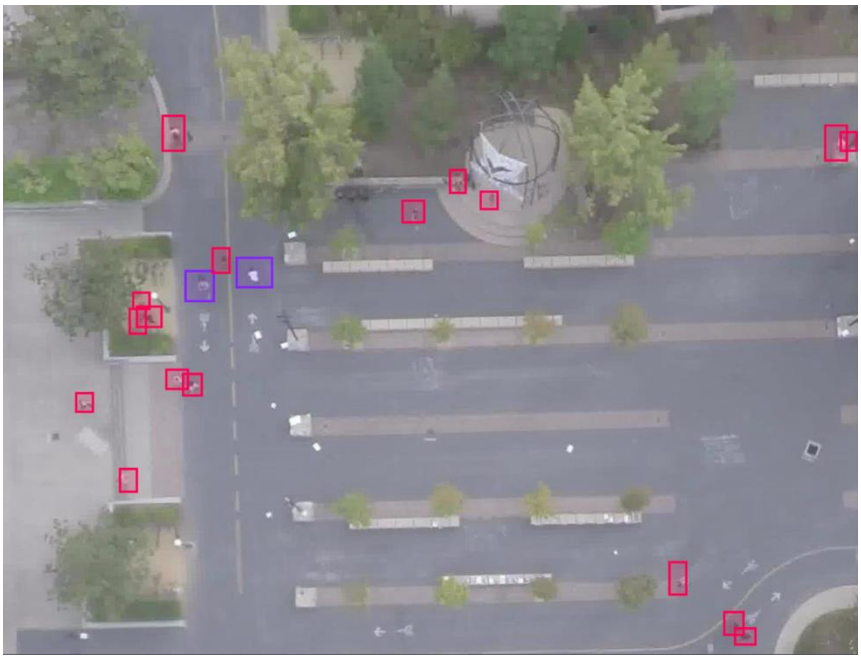
(a) Test Görüntüsü



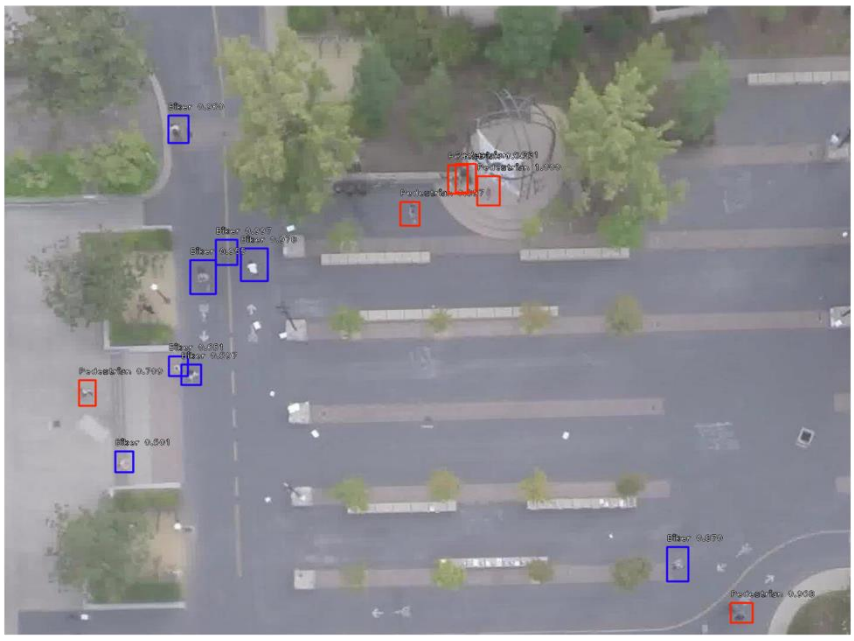
(b) Tahmin Görüntüsü

Şekil 4.17. Test görüntüsünün YOLOv3 çıkarım görüntüsü ile karşılaştırılması

Şekil 4.18'de en son aşamada kullanılan test görüntülerinden örnek alınan bir görüntünün RetinaNet modeli ile yapılan tahmin sonuçlarına ait görüntü karşılaştırılmıştır.



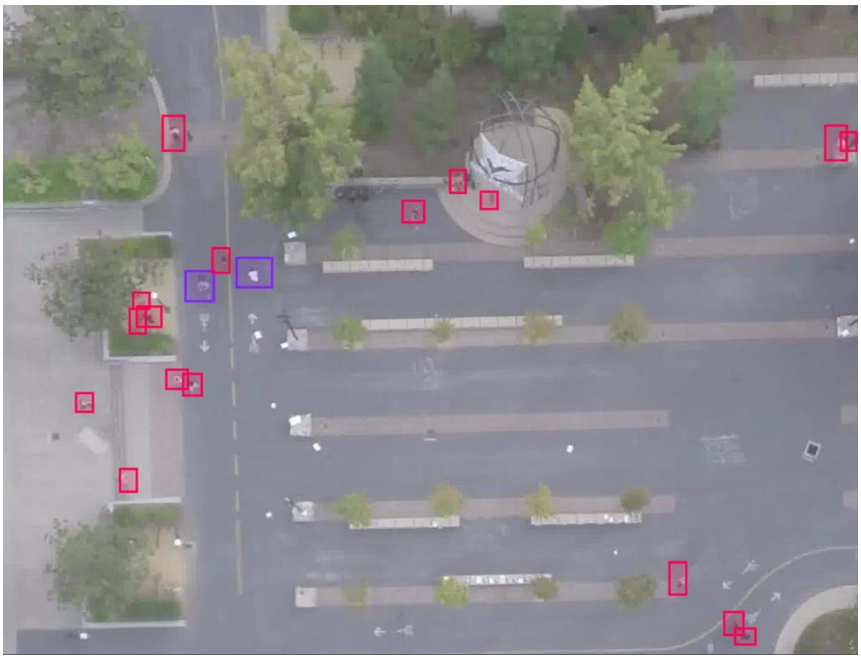
(a) Test Görüntüsü



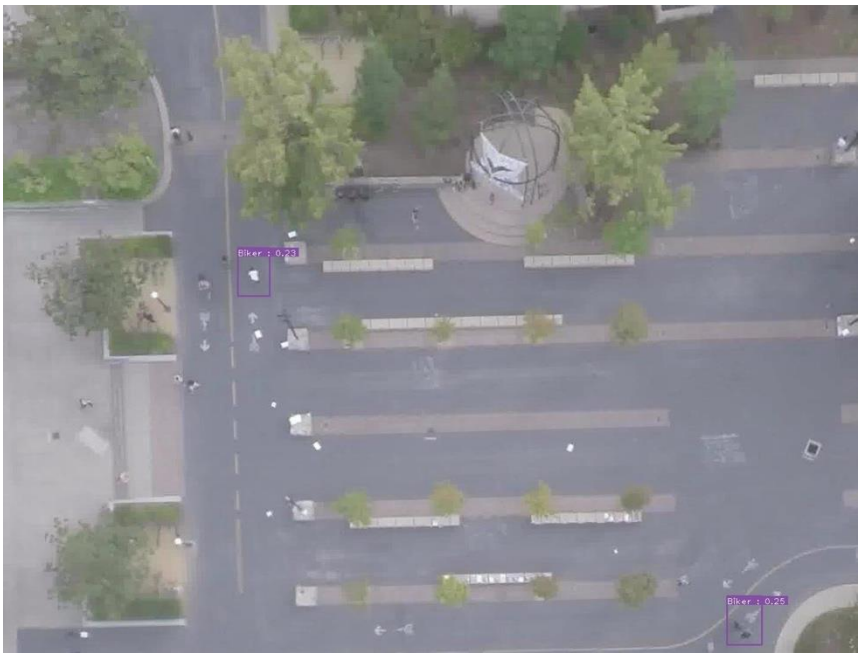
(b) Tahmin görüntüsü

Şekil 4.18. Test görüntüsünün RetinaNet çıkarım görüntüsü ile karşılaştırılması

Şekil 4.19'da en son aşamada kullanılan test görüntülerinden örnek alınan bir görüntünün EfficientDet modeli ile yapılan tahmin sonuçlarına ait görüntü karşılaştırılmıştır.



(a) Test Görüntüsü



(b) Tahmin görüntüsü

Şekil 4.19. Test görüntüsünün EfficientDet çıkarım görüntüsü ile karşılaştırılması

Şekil 4.20 test görüntülerinden örnek bir görüntünün YOLOv5 modeli ile yapılan çıkarım sonucunu göstermektedir.



(a) Test Görüntüsü (b) Tahmin Görüntüsü

Şekil 4.20. Test görüntüsünün YOLOv5 çıkarım görüntüsü ile karşılaştırılması

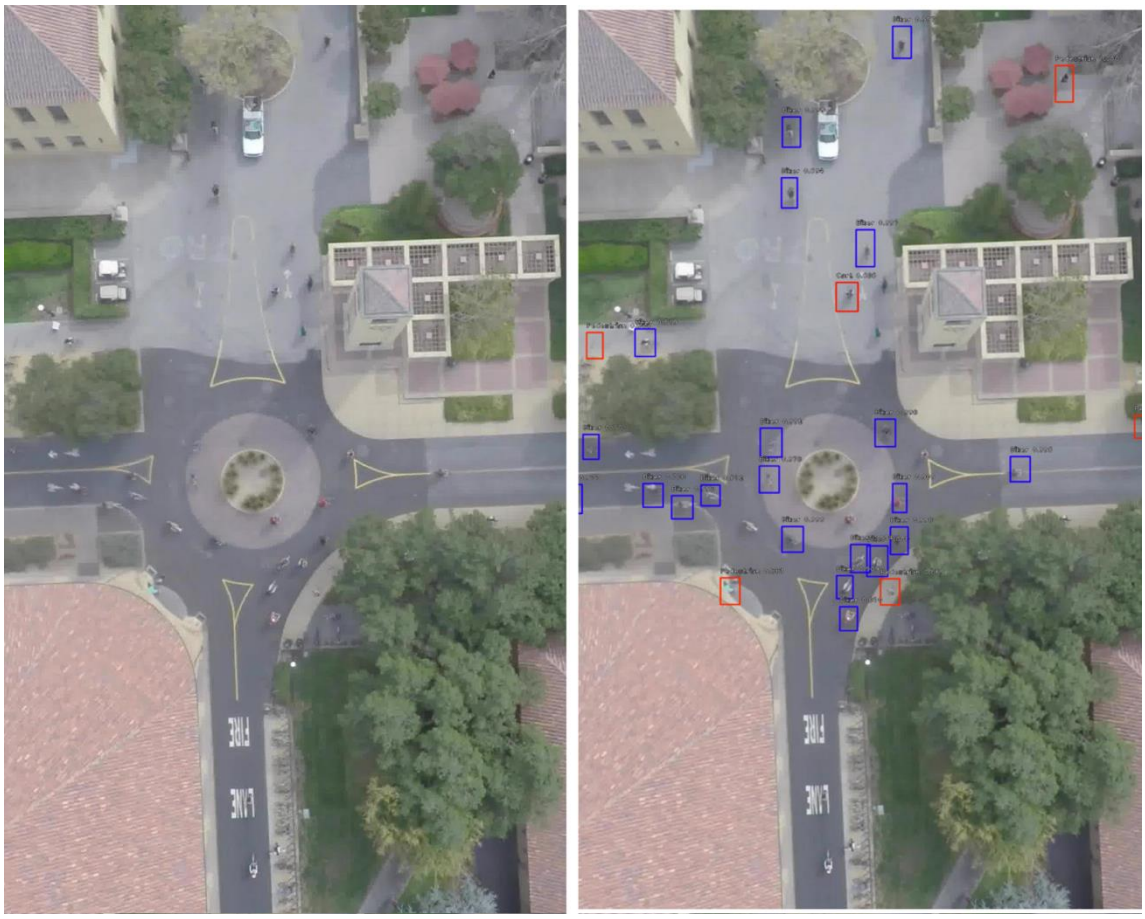
Şekil 4.21 test görüntülerinden örnek bir görüntünün YOLOv3 modeli ile yapılan çıkarım sonucunu göstermektedir.



(a) Test Görüntüsü (b) Tahmin Görüntüsü

Şekil 4.21. Test görüntüsünün YOLOv3 çıkarım görüntüsü ile karşılaştırılması

Şekil 4.22 test görüntülerinden örnek bir görüntünün RetinaNet modeli ile yapılan çıkarım sonucunu göstermektedir.



(a) Test Görüntüsü (b) Tahmin Görüntüsü

Şekil 4.22. Test görüntüsünün RetinaNet çıkarım görüntüsü ile karşılaştırılması

Şekil 4.23 test görüntülerinden örnek bir görüntünün EfficientDet modeli ile yapılan çıkarım sonucunu göstermektedir.



(a) Test Görüntüsü (b) Tahmin Görüntüsü

Şekil 4.23. Test görüntüsünün EfficientDet çıkarım görüntüsü ile karşılaştırılması

BÖLÜM 5. SONUÇLAR

İnsansız hava araçlarından elde edilen görüntüler kullanılarak farklı evrişimsel sinir ağı modelleri üzerinde performans ve verimlilik çalışmaları yapılmıştır. Tez çalışması süresince her modele ait farklı senaryolar defalarca denenmiş ve her bir hiperparametre değerinin etkisinin en optimum düzeye ulaşılması sağlanmıştır.

Tez çalışmasına başlandığı andan itibaren görülmüştür ki literatürde tek aşamalı nesne tespit modellerinden YOLOv5 modeline ait kaynak olabilecek çalışma ya da makale eksikliği bulunmaktadır. Önceki YOLO modellerinin aksine YOLOv5 modelinin geliştiricileri tarafından resmi olarak makale yayımlanmamıştır. Çalışma boyunca hem YOLO modellerine hem de çalışılan diğer modellere ait kapsamlı bir literatür araştırması yapılmıştır. Araştırmada kaynak noktası modellerin insansız hava aracı görüntüleri kullanılarak denenmesi olmuştur.

Yapılan defalarca deneme ve iyileştirme neticesinde 4 farklı evrişimsel sinir ağı modeli ile hava fotoğraflarından nesne tespiti ve yerelleştirme çalışmalarında başarı ölçütü olarak en yüksek ortalama doğruluk oranı 0,69 şeklinde tespit edilmiştir. YOLOv5 modeline ait olan bu sonuç ile çalışmanın başında belirtmiş olduğumuz hipotez büyük ölçüde gerçekleşmiştir. Tek aşamalı nesne tespit modellerinden YOLOv5 modeli yeni nesil diğer tek aşamalı nesne tespit modellerine göre daha başarılı sonuçlar üretmiştir.

Her ne kadar üzerinde daha çok deneme yaptığımız YOLOv5 modeli başarı olarak beklenen sonuca ulaşmış olsa da çalışmaları bir bütün olarak düşündüğümüzde genel başarı oranı düşük çıkmıştır. Çünkü üzerinde çalışan 4 modelin ImageNet, MS COCO gibi diğer veri setleri ile yapılan performans ölçümlerinde başarı oranları çok daha yüksek çıkmaktadır. Bu tez çalışmasında en yüksek ortalama doğruluk oranının 0,69 olarak elde edilmesinin başlıca nedeni veri kaynağı olarak kullandığımız hava fotoğraflarıdır.

Hava fotoğraflarından nesne tespitinde başarının diğer veri setlerine göre düşük kabul edilebilecek oranda çıkmasının sebeplerini anlayabilmek amacıyla tez çalışmasında iki veri seti birden kullanılmıştır. Yapılan çalışmalar şu şekilde değerlendirilmektedir:

- İlk olarak kullanılan DOTA veri seti modelin eğitimine sunulmadan incelendiğinde görüntü kalitesi oldukça yüksek ve nesne belirginliği fazla olan görüntülerden oluşmaktadır. Her bir görüntü 4000x4000 piksel boyutlarında olan

bu veri setinin bu kadar yüksek çözünürlük ile modellerin eğitimine sunulması mümkün olmamıştır. Görüntü boyutları yaklaşık yaklaşık 1/6 oranında küçültüldüğü için nesne seçiciliği de düşmüştür. Aynı şekilde Stanford drone veri seti de çözünürlükleri düşürülerek modellerin eğitimine sunulmuştur. Bu durum nesnelerin algılanabilirliğini düşürmüştür. Zira en son aşamada kullanılan test görüntülerinde yaya ya da bisikletli sınıfına ait nesneleri insan gözü ile dahi ayırt etmek mümkün olmamıştır.

- Evrişimsel sinir ağlarında modellerin başarısını değerlendirmek için kullanılan diğer veri setleri oldukça basit, net ve küçük boyutludur. Bu veri setlerinde günlük hayata dair nesneler çok yönlü olarak görülebilmektedir. Ancak insansız hava aracından alınan görüntüler nesnelerin uzay açıdan sadece tek perspektiften görünmeleri nedeniyle tanınabilme ya da algılanabilme olasılığı da azalmaktadır. Bu durum da başarıyı olumsuz yönde etkilemektedir.
- İnsansız hava aracından alınan görüntülerde başarıyı etkileyen en önemli kriterlerden biri de hava aracının yerden yüksekliğidir. Zira yerden yükseklik ne kadar fazla olursa görüntü içerisinde nesne ve özellik sayısı artacak ancak nesnelerin anlaşılabilirliği o oranda azalacaktır. Kullanılan DOTA veri setinde 15 nesne sınıfına ait binlerce etiketli nesne bulunmasına rağmen özellikle küçük araç, büyük araç, depolama tankı gibi nesne sınıflarının algılanma ihtimali düşüktür. Bu durum da modellerin performansını düşürmüştür.

Bu tez çalışması ile YOLOv5 tek aşamalı nesne tespit dedektörü kullanılarak insansız hava aracı fotoğraflarından diğer modellere göre hızlı ve doğru nesne tespiti sağlayacak yazılım geliştirilmiştir. Çalışma kapsamında hedeflenen amaç doğrultusunda geliştirilen uygulamanın başarısı diğer tek aşamalı nesne tespit modelleri ile karşılaştırılarak sonuçlar ayrıntılı bir şekilde analiz edilmiştir. En son aşamada kullanılan tüm modeller ile etiketsiz veri kullanılarak geliştirilen nesne tespit uygulaması değerlendirilmiştir.

Hızı ve doğruluğu her yeni geliştirme sonunda artan tek aşamalı nesne tespit dedektörlerinin öncelikli kullanım amacı otonom ve sürücüsüz araçlar olarak kabul edilmektedir. Dolayısıyla bu modellerin gerçek zamanlı olarak başarılarının artırılmasına önem verilmektedir. Önümüzdeki dönemlerde hava araçlarından alınan görüntüler kullanılarak özellikle YOLOv5 modeli ile gerçek zamanlı daha gelişmiş çalışmalar yapılacaktır.

KAYNAKÇA

[1] T. Kashiyama, H. Sobue ve Y. Sekimoto, «Sky Monitoring System for Flying Object Detection Using 4K Resolution Camera,» *Sensors,* cilt 24, no. 7071, 2020.

[2] H. Yu, G. Li, W. Zhang, Q. Huang, D. Du, Q. Tian ve N. Sebe, «The Unmanned Aerial Vehicle Benchmark: Object Detection, Tracking and Baseline,» *International Journal of Computer Vision,* no. 128, pp. 1141-1159, 2020.

[3] M. Kahveci ve N. Can, «İnsansız Hava Araçları: Tarihçesi, Tanımı, Dünyada ve Türkiye'deki Yasal Durumu,» *Selçuk Üniv. Müh. Bil. ve Tek. Dergisi,* cilt 5, no. 4, pp. 511-537, 2017.

[4] *https://www.twenty20.com/photos/71767c68-9d8b-4f56-a89d-83e57aa80fb9.* [Art].

[5] Milli Eğitim Bakanlığı, «Hava Fotoğrafı,» %1 içinde *Grafik ve Fotoğraf,* Ankara, Milli Eğitim Bakanlığı, 2013, pp. 3-6.

[6] H. Sarıbaş, H. Çevikalp ve S. Kahvecioğlu, «İnsansız Hava Araçlarından Alınan Görüntülerdeki Araçların Konumlarının Bulunması,» %1 içinde *26. IEEE Sinyal İşleme ve İletişim Uygulamaları Kurultayı*, İzmir, 2018.

[7] S. Toraman, «Derin Öğrenme ile İnsansız Hava Aracı Görüntülerinden Yaya Tespiti,» *Journal of Aviation,* cilt 2, no. (2), pp. 64-69, 2018.

[8] X. Wang, P. Cheng, X. Liu ve B. Uzochukwu, «Fast and Accurate, Convolutional Neural Network Based Approach for Object Detection from UAV,» arXiv.org, 2018.

[9] B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar ve K. Ouni, «Car Detection using Unmanned Aerial Vehicles: Comparison between Faster R-CNN and YOLOv3,» arXiv, 2018.

[10] N. Ammour, H. Alhichri, Y. Bazi, B. Benjdira, N. Alajlan ve M. Zuair, «Deep Learning Approach for Car Detection in UAV Imagery,» *Remote Sensing,* cilt 9, p. 312, 2017.

[11] M. Lan, Y. Zhang, L. Zhang ve B. Du, «Defect Detection from UAV Images based on Region-Based CNNs,» *School of Computer Science,* 2018.

[12] Y. Xu, M. Fu, Q. Wang, Y. Wang, K. Chen ve G.-S. Xia, «Gliding Vertex on the Horizontal Bounding Box For Multi-Oriented Object Detection,» *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 2020.

[13] J. Lee, J. Wang, D. Crandall, S. Sabanovic ve G. Fox, «Real-Time, Cloud-based Object Detection for Unmanned Aerial Vehicles,» %1 içinde *2017 First IEEE International Conference on Robotic Computing*, Indiana, 2017.

[14] T. Ophoff, S. Puttemans, V. Kalogirou, J. P. Robin ve T. Goedemé, «Vehicle and Vessel Detection on Satellite Imagery: A Comparative Study on Single-Shot Detectors,» *Remote Sensing,* cilt 12, no. 1217, 2020.

[15] M. Mandal, L. K. Kumar ve S. K. Vipparthi, «MOR-UAV: A Benchmark Dataset and Baselines for Moving Object Recognition in UAV Videos,» %1 içinde *Computer Vision and Pattern Recognition,* Seattle, 2020.

[16] U. Algancı, M. Soydaş ve E. Sertel, «Comparative Research on Deep Learning Approaches for Airplane Detection from Very High-Resolution Satellite Images,» *Remote Sensing,* cilt 12, no. 3, p. 458, 2020.

[17] T. Hinzmann, T. Stegemann, C. Cadena ve R. Siegwart, «Deep Learning-based Human Detection for UAVs with Optical and Infrared Cameras: System and Experiments,» Zurich, 2020.

[18] J. Carlet ve B. Abayowa, «Fast Vehicle Detection in Aerial Imagery,» Computer Science Bibliograph, 2017.

[19] S. H. Onim, M. I. Akash, M. Haque ve R. I. Hafiz, «Traffic Surveillance using Vehicle License Plate Detection and Recognition in Bangladesh,» %1 içinde *11th International Conference on Electrical and Computer Engineering (ICECE),* Dhaka, 2020.

[20] N. J. Nilsson, %1 içinde *Yapay Zeka Geçmişi ve Geleceği*, Boğaziçi Üniversitesi Yayınevi, 2010, p. 13.

[21] A. Yılmaz, «Bilgisayarlı Öğrenme ve Makine Öğrenmesi,» %1 içinde *Yapay Zeka,* İstanbul, Kodlab Yayın, 2018, p. 44.

[22] I. Goodfellow, Y. Bengio ve A. Courville, «Derin Öğrenme,» %1 içinde *Derin Öğrenme,* Ankara, Buzdağı Yayınevi, 2018, pp. 1-1.

[23] «The Perceptron,» 29 July 2018. [Çevrimiçi]. Available: https://towardsdatascience.com/the-perceptron-3af34c84838c. [Erişildi: 23 Haziran 2020].

[24] F. Tüminçin, A. Aytekin ve A. Ayaz, «İnsan Beyni Modellemesi ve Yapay Sinir Ağları,» %1 içinde *SADAB 5th International Social Research and Behavioral Sciences Symposium*, Tbilisi, 2019.

[25] K. Krzyk, «Coding Deep Learning for Beginners — Linear Regression (Part 2): Cost Function,» Medium, 8 8 2018. [Çevrimiçi]. Available: https://towardsdatascience.com/coding-deep-learning-for-beginners-linear-regression-part-2-cost-function-49545303d29f#:~:text=comes%20to%20help.-,Cost%20Function,formed%20in%20many%20different%20ways.. [Erişildi: 29 12 2020].

[26] A. Ng, «Neural Networks and Deep Learning,» Coursera, 2020.

[27] N. Donges, «Gradient Descent,» 11 7 2019. [Çevrimiçi]. Available: https://www.experfy.com/blog/ai-ml/gradient-descent/. [Erişildi: 31 12 2020].

[28] A. Al-Masri, «How Does Back-Propagation in Artificial Neural Networks Work?,» Medium, 30 1 2019. [Çevrimiçi]. Available: https://towardsdatascience.com/how-does-back-propagation-in-artificial-neural-networks-work-c7cad873ea7#:~:text=Back%2Dpropagation%20is%20just%20a,lower%20weights%20and%20vice%20versa.. [Erişildi: 30 12 2020].

[29] I. Goodfellow, Y. Bengio ve A. Courville, «Derin İleri Besleme Ağları,» %1 içinde *Derin Öğrenme*, MIT, 2016, pp. 201-217.

[30] N. Çarkacı, «Derin Öğrenme Uygulamalarında En Sık kullanılan Hiper-parametreler,» 22 01 2018. [Çevrimiçi]. Available: https://medium.com/deep-learning-turkiye/derin-ogrenme-uygulamalarinda-en-sik-kullanilan-hiper-parametreler-ece8e9125c4. [Erişildi: 01 01 2021].

[31] H. Zulkifli , «Understanding Learning Rates and How It Improves Performance in Deep Learning,» 21 01 2018. [Çevrimiçi]. Available: https://towardsdatascience.com/understanding-learning-rates-and-how-it-improves-performance-in-deep-learning-d0d4059c1c10. [Erişildi: 01 01 2021].

[32] A. Kızrak, «Comparison of Activation Functions for Deep Neural Networks,» 9 5 2019. [Çevrimiçi]. Available: https://towardsdatascience.com/comparison-of-activation-functions-for-deep-neural-networks-706ac4284c8a. [Erişildi: 1 1 2021].

[33] E. Seyyarer, F. Ayata, T. Uçkan ve A. Karcı, «Derin Öğrenmede Kullanilan Optimizasyon Algoritmalarinin Uygulanmasi Ve Kiyaslanmasi,» *Anatolian Journal of Computer Sciences,* cilt 5, no. 2, pp. 90-98, 2020.

[34] «Reducing Loss: Gradient Descent,» Google, 02 10 2020. [Çevrimiçi]. Available: https://developers.google.com/machine-learning/crash-course/reducing-loss/gradient-descent. [Erişildi: 03 01 2021].

[35] S. Raschka, «Gradient Descent and Stochastic Gradient Descent,» [Çevrimiçi]. Available: http://rasbt.github.io/mlxtend/user_guide/general_concepts/gradient-optimization/. [Erişildi: 03 01 2021].

[36] G. Orr, N. Schraudolph ve F. Cummins, «Momentum and Learning Rate Adaptation,» Willamette University, 1999. [Çevrimiçi]. Available: https://www.willamette.edu/~gorr/classes/cs449/momrate.html. [Erişildi: 01 01 2021].

[37] J. Du, «The Frontier of SGD and Its Variants in Machine Learning,» %1 içinde *Journal of Physics: Conference Series*, Qingdao, 2019.

[38] V. Bushaev, «Adam — latest trends in deep learning optimization,» towards data science, 22 10 2018. [Çevrimiçi]. Available: https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c. [Erişildi: 03 01 2021].

[39] J. Brownlee, «Difference Between a Batch and an Epoch in a Neural Network,» 26 10 2019. [Çevrimiçi]. Available: https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/. [Erişildi: 04 01 2021].

[40] S. Albawi, T. A. Mohammed ve S. Al-Zawi, «Understanding of a Convolutional Neural Network,» %1 içinde *ICET,* Antalya, 2017.

[41] K. O'Shea ve R. Nash, «An Introduction to Convolutional Neural Networks,» 13 8 2015.

[42] A. Amidi ve S. Amidi, «CS 230 - Derin Öğrenme,» Stanford University, 2019.

[43] A. Amidi ve S. Amidi, «Deep Learning cheatsheets for Stanford's CS 230,» Stanford University, 20 05 2020. [Çevrimiçi]. Available: https://github.com/afshinea/stanford-cs-230-deep-learning. [Erişildi: 08 01 2021].

[44] «CS231n Convolutional Neural Networks for Visual Recognition,» Stanford University, 2020. [Çevrimiçi]. Available: https://cs231n.github.io/convolutional-networks/#pool. [Erişildi: 04 01 2021].

[45] K. Rana, «Pooling Layer — Short and Simple,» 3 4 2020. [Çevrimiçi]. Available: https://medium.com/ai-in-plain-english/pooling-layer-beginner-to-intermediate-fa0dbdce80eb. [Erişildi: 04 01 2021].

[46] Arunava, «Convolutional Neural Network,» 25 12 2018. [Çevrimiçi]. Available: https://towardsdatascience.com/convolutional-neural-network-17fb77e76c05. [Erişildi: 04 01 2021].

[47] A. Amidi ve S. Amidi, «Evrişimli Sinir Ağları,» Stanford University, 01 05 2019. [Çevrimiçi]. Available: https://stanford.edu/~shervine/l/tr/teaching/cs-230/cheatsheet-convolutional-neural-networks. [Erişildi: 07 01 2021].

[48] V. Dumoulin ve F. Visin, «A guide to convolution arithmetic for deep learning,» *ArXiv e-prints,* no. 1603.07285, 2016.

[49] «ImageNet,» Stanford Vision Lab, 2016. [Çevrimiçi]. Available: http://image-net.org/about-overview. [Erişildi: 16 12 2020].

[50] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li ve L. Fei-Fei, «ImageNet: A Large-Scale Hierarchical Image Database,» %1 içinde *IEEE Bilgisayarla Görme ve Örüntü Tanıma (CVPR)*, 2009.

[51] M. Everingham, L. v. Gool, C. Williams, J. Winn ve A. Zisserman, «The PASCAL Visual Object Classes Homepage,» 11 2013. [Çevrimiçi]. Available: http://host.robots.ox.ac.uk/pascal/VOC/. [Erişildi: 16 12 2020].

[52] T. Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick ve P. Dollar, «Microsoft COCO: Common Objects in Context,» 2015.

[53] «MNIST database,» National Institute of Standards and Technology, [Çevrimiçi]. Available: https://en.wikipedia.org/wiki/MNIST_database. [Erişildi: 05 01 2021].

[54] A. Krizhevsky, V. Nair ve G. Hinton, «CIFAR Dataset,» Toronto University, 2009. [Çevrimiçi]. Available: https://www.cs.toronto.edu/~kriz/cifar.html. [Erişildi: 04 01 2021].

[55] J. Redmon, S. Divvala, R. Girshick ve A. Farhadi, «You Only Look Once: Unified, Real-Time Object Detection,» 2016.

[56] J. Redmon ve A. Farhadi, «YOLO9000: Better, Faster, Stronger,» 2016.

[57] J. Redmon ve A. Farhadi, «YOLOv3: An Incremental Improvement,» Washington, 2018.

[58] R. Geng, Y. Ma ve W. Huang, «An improved helmet detection method for YOLOv3 on an unbalanced dataset,» Dalian, 2020.

[59] A. Bochkovskiy, C. Y. Wang ve H.-Y. M. Liao, «YOLOv4: Optimal Speed and Accuracy of Object Detection,» 2020.

[60] J. Hui, «YOLOv4,» 4 5 2020. [Çevrimiçi]. Available: https://jonathan-hui.medium.com/yolov4-c9901eaa8e61. [Erişildi: 8 1 2021].

[61] T.-Y. Lin, P. Goyal, R. Girshick, K. He ve P. Dollar, «Focal Loss for Dense Object Detection,» Facebook AI Research (FAIR), 2017.

[62] M. Tan, R. Pang ve . Q. V. Le, «EfficientDet: Scalable and Efficient Object Detection,» Google Research, Brain Team, 2020.

[63] M. Tan ve A. Yu, «EfficientDet: Towards Scalable and Efficient Object Detection,» 15 4 2020. [Çevrimiçi]. Available: https://ai.googleblog.com/2020/04/efficientdet-towards-scalable-and.html. [Erişildi: 31 12 2020].

[64] G.-S. Xia, X. Bai, J. Ding, Z. Zhu, S. Belongie, J. Luo, M. Datcu, M. Pelillo ve L. Zhang, «DOTA: A Large-scale Dataset for Object Detection in Aerial Images,» %1 içinde *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[65] A. Robicquet, A. Sadeghian, A. Alahi ve S. Savarese, «Learning Social Etiquette: Human Trajectory Prediction In Crowded Scenes,» %1 içinde *European Conference on Computer Vision (ECCV)*, 2016.

[66] S. Açıl, «Intersection over Union (IoU) ve Mean Average Precision (MAP) Nedir?,» Deep Learning Türkiye, 17 9 2018. [Çevrimiçi]. Available: https://medium.com/deep-learning-turkiye/intersection-over-union-iou-ve-mean-average-precision-map-nedir-532241ab74a9. [Erişildi: 04 01 2021].