T.C.

BARTIN ÜNİVERSİTESİ

LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

AKILLI SİSTEMLER MÜHENDİSLİĞİ ANABİLİM DALI

YÜKSEK LİSANS TEZİ

KONVEYÖR BANT ÜZERİNDE YAPAY ZEKÂ TABANLI BALIK SINIFLANDIRMA

MAHAMAT AHMAT ISSAMADINE

DANIŞMAN

DOÇ. DR. YASEMİN ERKAN

BARTIN-2025

T.C.

BARTIN ÜNİVERSİTESİ

LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

AKILLI SİSTEMLER MÜHENDİSLİĞİ ANABİLİM DALI

KONVEYÖR BANT ÜZERİNDE YAPAY ZEKÂ TABANLI BALIK SINIFLANDIRMA

YÜKSEK LİSANS TEZİ

Mahamat Ahmat ISSAMADINE

JÜRİ ÜYELERİ

Danışman : Doç. Dr. Yasemin ERKAN

Üye : Doç. Dr. Ersin ALAYBEYOĞLU

Üye : Dr. Öğr. Üyesi Tuğba PALABAŞ

BARTIN-2025

KABUL VE ONAY

Mahamat Ahmat ISSAMADINE tarafından hazırlanan “KONVEYÖR BANT ÜZERİNDE YAPAY ZEKÂ TABANLI BALIK SINIFLANDIRMA” başlıklı bu çalışma, 20.11.2025 tarihinde yapılan savunma sınavı sonucunda oy birliği ile başarılı bulunarak jürimiz tarafından Yüksek Lisans Tezi olarak kabul edilmiştir.

Başkan	:	Doç. Dr. Yasemin ERKAN	……
Üye	:	Doç. Dr. Ersin ALAYBEYOĞLU	……
Üye	:	Dr. Öğr. Üyesi Tuğba PALABAŞ	……

Bu tezin kabulü Lisansüstü Eğitim Enstitüsü Yönetim Kurulu’nun ..…/..…/20… tarih ve 20…../…..-….. sayılı kararıyla onaylanmıştır.

Prof. Dr. Zafer CEYLAN
Enstitü Müdürü

BEYANNAME

Bartın Üniversitesi Lisansüstü Eğitim Enstitüsü tez yazım kılavuzuna göre Doç. Dr. Yasemin ERKAN danışmanlığında hazırlamış olduğum “Konveyör Bant Üzerinde Yapay Zekâ Tabanlı Balık Sınıflandırma” başlıklı yüksek lisans tezimin bilimsel etik değerlere ve kurallara uygun, özgün bir çalışma olduğunu, aksinin tespit edilmesi halinde her türlü yasal yaptırımı kabul edeceğimi beyan ederim.

20.11.2025

Mahamat Ahmat ISSAMADINE

ÖN SÖZ

Bu tezin hazırlanma sürecinde bilgi, birikim ve rehberliğini esirgemeyen, akademik anlamda bana yol gösteren ve her zaman destek olan danışman hocam Doç. Dr. Yasemin Erkan'a en içten teşekkürlerimi sunarım.

Tez süresince göstermiş oldukları ilgi, yapıcı eleştiriler ve değerli katkılarla çalışmama önemli katkılarda bulunan tüm hocalarıma ve bu süreçte yanımda olan aileme ve arkadaşlarıma da teşekkür ederim.

Bu çalışma sürecinde edindiğim bilgi ve deneyimlerin, akademik ve mesleki hayatıma ışık tutacağına inanıyor, katkıda bulunan herkese tekrar şükranlarımı sunuyorum.

Mahamat Ahmat ISSAMADINE

ÖZET

Yüksek Lisans Tezi

KONVEYÖR BANT ÜZERİNDE YAPAY ZEKÂ TABANLI BALIK SINIFLANDIRMA

Mahamat Ahmat ISSAMADINE

Bartın Üniversitesi
Lisansüstü Eğitim Enstitüsü
Akıllı Sistemler Mühendisliği Anabilim Dalı

Tez Danışmanı: Doç. Dr. Yasemin ERKAN

Bartın-2025, sayfa: 62

Günümüzde yapay zekâ teknolojilerinin gelişimi, birçok sektörde olduğu gibi balıkçılık alanında da çeşitli sorunlara çözüm sunmaktadır. Balıkçılık sektöründe, aynı anda yakalanan farklı türdeki balıkların manuel olarak sınıflandırılması hem zaman alıcıdır hem de hata payı yüksektir. İnsan eliyle yapılan bu sınıflandırma işlemleri, balık türlerinin doğru tespit edilememesine ve dolayısıyla verimliliğin düşmesine neden olmaktadır. Bu durum, maliyet artışı ve kalite kontrolünde sorunlar yaratmakta, sınıflandırma sürecinin otomatikleştirilmesi gerekliliğini ortaya koymaktadır. Bu çalışmada, hareketli bir konveyör üzerinde ilerleyen balıkların görüntüleri alınarak, bu görüntülerden elde edilen verilerle yapay zekâ tabanlı nesne tespiti ve sınıflandırma işlemleri gerçekleştirilmiştir. Çalışma kapsamında üç farklı balık türüne ait toplam 3915 görüntü toplanarak özel bir veri seti oluşturulmuştur. Görüntü işleme ve nesne tanıma alanında güçlü bir performans sergileyen YOLO algoritmasının son sürümleri olan YOLOv8, YOLOv10 ve YOLOv11 modelleri bu çalışmada kullanılmıştır. Modellerin ortalama mAP50-95 performans değerleri sırasıyla YOLOv8 için %77.7, YOLOv10 için %78.9 ve YOLOv11 için %79.4 olarak elde edilmiştir. Elde edilen sonuçlar, YOLO algoritmalarının balık türlerinin otomatik sınıflandırılmasında yüksek doğruluk ve hız sunduğunu göstermektedir. Bu sistemin uygulanmasıyla birlikte insan müdahalesine olan ihtiyaç azalmakta, sınıflandırma süreci daha tutarlı ve ekonomik hale gelmektedir.

Anahtar Kelimeler: Derin öğrenme, Sınıflandırma, Yapay zekâ, YOLO.

ABSTRACT

M.A. Thesis

ARTIFICIAL INTELLIGENCE-BASED FISH CLASSIFICATION ON CONVEYOR BELTS

Mahamat Ahmat ISSAMADINE

Bartın University
Graduate School
Department of Intelligent Systems Engineering

Thesis Advisor: Doç. Dr. Yasemin ERKAN
Bartın-2025, pp: 62

Today, the development of artificial intelligence technologies offers solutions to various problems in the fishing industry, as in many other sectors. In the fishing industry, manually classifying different types of fish caught at the same time is both time-consuming and prone to error. These manual classification processes result in the incorrect identification of fish species and, consequently, a decrease in efficiency. This situation creates cost increases and quality control issues, highlighting the need to automate the classification process. In this study, images of fish moving on a conveyor belt are captured, and artificial intelligence-based object detection and classification processes are performed using the data obtained from these images. A total of 3,915 images of three different fish species are collected to create a special dataset. The latest versions of the YOLO algorithm, which demonstrates strong performance in image processing and object recognition, namely YOLOv8, YOLOv10, and YOLOv11, are used in this study. After the training, the average mAP50-95 performance values of the models are obtained as 77.7% for YOLOv8, 78.9% for YOLOv10, and 79.4% for YOLOv11, respectively. The results demonstrate that YOLO algorithms offer high accuracy and speed in the automatic classification of fish species. With the implementation of this system, the need for human intervention decreases, and the classification process becomes more consistent and economical.

Keywords: Artificial intelligence, Classification, Deep learning, YOLO.

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

Şekil No **Sayfa No**

TABLOLAR DİZİNİ

Tablo No **Sayfa No**

KISALTMALAR DİZİNİ

AI	: Yapay Zekâ (Artificial Intelligence)
CPU	: Merkezi İşlem Birimi (Central Processing Unit)
CNN	: Konvolüsyonel Sinir Ağı (Convolutional Neural Network)
DL	: Derin Öğrenme (Deep Learning)
ESA	: Evrimsel Sinir Ağları (Evolutionary Synaptic Algorithm)
Fast R-CNN	: Hızlı Bölgesel CNN (Fast Region-based Convolutional Neural Network)
FPS	: Saniyedeki Kare Sayısı (Frames Per Second)
GPU	: Grafik İşlem Birimi (Graphics Processing Unit)
IoU	: Kesişim Bölü Birleşim (Intersection over Union)
K-NN	: En Yakın Komşular Algoritması (K-Nearest Neighbors)
mAP	: Ortalama Doğruluk Oranı (Mean Average Precision)
MR	: Manyetik Rezonans (Magnetic Resonance)
PSO	: Parçacık Sürü Optimizasyonu (Particle Swarm Optimization)
R-CNN	: Bölgesel CNN (Region-based Convolutional Neural Network)
RGB	: Kırmızı Yeşil Mavi Renk Modeli (Red Green Blue)
SSD	: Tek Atışta Çoklu Kutu Dedektörü (Single Shot Multibox Detector)
SVM	: Destek Vektör Makineleri (Support Vector Machines)
YOLO	: Sadece Bir Kez Bak (You Only Look Once)
YZ	: Yapay Zekâ

1. GİRİŞ

Son yıllarda teknolojideki yapay zekânın hayatımıza entegrasyonuyla birlikte, sanayideki üretimden işleme süreçlerine kadar pek çok alanda köklü değişiklikler yaşanmaktadır. Bu yeniliklerin temelini oluşturan bilgisayar görüşü ve makine öğrenmesi uygulamaları, geleneksel üretim yöntemlerini dönüştürerek otomasyonun endüstride daha yaygın kullanılmasını sağlamaktadır. Özellikle ürünlerin kalitesini kontrol etme, sınıflandırma ve tanıma işlemlerinde yapay zekâ destekli sistemler hem doğruluk oranlarını artırmakta hem de insan kaynaklı hataları minimize ederek üretim verimliliğini belirgin şekilde geliştirmektedir. Bu gelişmeler doğrultusunda, pek çok endüstriyel tesiste artık standart hale gelen konveyör bant sistemleri de dönüşüm geçirmektedir. Bu konveyör bantlar üzerinde hareket eden ürünlerin veya parçaların görüntüleri gerçek zamanlı olarak analiz edilebilmekte; böylece nesne tespiti vasıtasıyla bu öğeler tür, kalite veya spesifik özelliklerine göre otomatik olarak sınıflandırılabilmektedir.

Türkiye'nin Karadeniz kıyıları, zengin biyolojik çeşitliliği ve yoğun balıkçılık faaliyetleriyle öne çıkan önemli bir ekosistemi barındırmaktadır. Bu bağlamda, Karadeniz'e kıyısı olan Bartın ili hem ticari hem de geleneksel balıkçılık açısından dikkat çekici bir bölge olarak konumlanmaktadır. Bu süreçte, işleme tesislerine gelen balıkların türlerine göre hızlı ve doğru bir şekilde sınıflandırılması büyük önem taşımaktadır. Nitekim, balık işleme tesislerinde bu ayırma işlemi halen büyük ölçüde insanların elleriyle gerçekleştirilmekte olup (Issamadine vd., 2024), geleneksel olarak insan gücüne dayalı olarak yürütülen bu sınıflandırma süreci, yavaş, maliyetli, yorucu olmasının yanı sıra operatörün deneyimine bağlı olarak tutarlılık sorunları yaşanabilen bir süreçtir. Bu tür zorluklar, sektörde otomasyon sistemlerine olan ihtiyacı belirgin hale getirmiştir.

Endüstriyel otomasyonun temel unsurlarından biri olan konveyör bant sistemleri, üretim hatlarında malzeme akışını sağlamak için yaygın olarak kullanılmaktadır. Son yıllarda bu sistemler, bilgisayarlı görü teknolojileriyle entegre edilerek akıllı platformlara dönüştürülmektedir (Öylek, 2014). Bilgisayarlı görü, makinelerin görsel dünyayı "anlamasını" sağlayan, görüntülerden veya videolardan anlamlı bilgiler çıkaran bir yapay zekâ alanıdır. Bu teknoloji sayesinde, bir konveyör bant üzerinde hareket eden nesnelerin görüntüleri kameralar aracılığıyla anlık olarak yakalanıp analiz edilebilmekte, böylece

otomatik tanıma ve sınıflandırma işlemleri mümkün hale gelmektedir.

Nesne tespiti, bilgisayarlı görünün en önemli alt dallarından biridir ve bir görüntüdeki belirli nesnelerin hem konumlarını hem de sınıflarını belirleme yeteneğine odaklanır. Özellikle derin öğrenme tabanlı modeller, bu alanda devrim yaratmıştır. "You Only Look Once" (YOLO) algoritma ailesi, tek bir sinir ağı geçişiyle nesne tespiti yapabilmesi sayesinde yüksek hız ve doğruluk dengesi sunarak gerçek zamanlı uygulamalar için endüstri standardı haline gelmiştir (Redmon vd., 2016). Algoritmanın zaman içinde geliştirilen YOLOv8, YOLOv10 ve YOLOv11 gibi yeni sürümleri, daha düşük gecikme süresi ve daha verimli mimari sunarak tespit performansını daha da ileriye taşımıştır (Mao ve Hong, 2025).

Balık türlerinin sınıflandırılması önemli bir görev olduğundan, sıklıkla araştırılmaktadır ve birçok farklı yaklaşım araştırmacılar tarafından geliştirilmiştir. Son yıllarda yapılan çalışmalar, balık türlerinin sınıflandırılmasında derin öğrenme yöntemlerinin etkinliğini açıkça ortaya koymaktadır. Su altı görüntülerinde balık türlerinin otomatik tanımlanması için geliştirilmiş derin öğrenme modelleri kullanarak %99'un üzerinde doğruluk oranı elde etmişlerdir (Siri vd., 2024). Katmanlı evrişimsel sinir ağlarını Residual CNN geliştirerek balık ve omurgasız türlerinin sınıflandırılmasında yüksek başarı oranlarına ulaşmıştır (Gong vd., 2023). Bu çalışmaların çoğu, su altı görüntüleri veya geniş kapsamlı veri setleri üzerinde gerçekleştirilmiştir. Buna karşılık, bu tez çalışmasında Türkiye'de ekonomik değere sahip barbun, hamsi ve mezgit balık türleri, konveyör bant üzerinde ve gerçek zamanlı olarak sınıflandırılmakta, böylece literatürdeki mevcut çalışmalardan farklı olarak endüstriyel otomasyon alanına, insan kaynaklı hataların azaltılmasına ve verimliliğin artırılmasına yönelik yenilikçi bir çözüm sunma potansiyeli taşımaktadır.

Bu tez çalışmasının sonraki bölümleri şu şekilde yapılandırılmıştır: İkinci bölümde, nesne tespiti alanında yapay zekâ yöntemlerinin kullanımı ve bu kapsamda gerçekleştirilen güncel akademik çalışmalar ele alınmıştır. Üçüncü bölümde, yapay zekâ temelli nesne tespiti algoritmaları tanıtılmış; özellikle bu tezde kullanılan modelleri detaylı biçimde açıklanmıştır. Dördüncü bölümde, balık sınıflandırmasına yönelik deneysel süreçlere, veri setinin hazırlanması, etiketlenmesi ve modellerin eğitilmesi gibi uygulama adımlarına yer verilmiştir. Beşinci ve son bölümde ise elde edilen bulgular değerlendirilmiş, modellerin performansı karşılaştırılmış ve tez çalışmasının genel sonuçları tartışılmıştır.

2. LİTERATÜR ÖZETİ

Literatür araştırması bölümü iki ana başlık altında ele alınmıştır. İlk başlıkta, derin öğrenmeye dayalı sınıflandırma araştırmalarında bulunan çalışmalar incelenmiş olup, bu çalışmaların derin öğrenme teknikleri kullanılarak nasıl gerçekleştirildiği detaylandırılmıştır.

İkinci başlıkta ise, YOLO nesne tespit modeli ile gerçekleştirilen analizlerine odaklanılmıştır. YOLO algoritması, gerçek zamanlı nesne tespiti için kullanılan güçlü bir modeldir ve sınıflandırma süreçlerinde de yaygın olarak kullanılmaktadır. Literatürde, YOLO'nun farklı versiyonları ile yapılan analizleri ele alınarak, hangi parametrelerin doğruluk oranlarını etkilediği tartışılmıştır.

2.1 Derin Öğrenme Tabanlı Sınıflandırma Üzerine Yapılan Çalışmalar

Derin öğrenme modelleri, son yıllarda görüntü işleme, nesne tanıma gibi birçok alanda etkin bir şekilde kullanılmaktadır. Bu bölümde, derin öğrenmenin temel prensipleri, farklı alanlardaki uygulamaları ve nesne tanıma ile ilgili çalışmaları incelenmiştir.

Su altı balık türlerinin otomatik tanımlanması üzerine yapılan çalışmalar, görüntü işleme ve yapay zekâ tekniklerinin evrimiyle birlikte önemli gelişmeler kaydetmiştir. Öklid, karesel ve üçgen ağ yöntemleri ile balıkların özniteliklerini çıkararak Naive Bayes algoritmasıyla sınıflandırma yapmış ve familya düzeyinde %93.10, tür düzeyinde %75.71 doğruluk elde etmiştir (İşçimen vd. (2014). Aynı araştırma grubu daha sonra iki sırt yüzgeçli balık türlerinin sınıflandırılması için merkez-kenar uzunluğu yöntemini ve En Yakın Komşu algoritmasını kullanmıştır. 7 familyadan 15 türe ait 427 balık görüntüsü üzerinde yapılan çalışmada, balıkların merkez-kenar uzaklıkları öznitelik olarak çıkarılmıştır. Sonuçta %95 genel doğruluk oranı elde edilmiştir (İşçimen vd., 2015). Ogunlana vd. (2015) ise balık türlerini sınıflandırmak için şekil tabanlı özniteliklere dayalı SVM algoritması geliştirmiştir. 150 balıktan elde edilen altı yüzgeç ve gövde uzunluğu kullanılmış, veriler eğitim ve test setlerine ayrılmıştır. Önerilen yöntem %78,59 doğruluk oranı ile diğer yöntemlerden daha iyi sonuç vermiştir. Salman vd. (2016) ise su altı görüntülerinde karmaşık arka planlara ve hareketli hedeflere rağmen CNN mimarisiyle %90 doğruluk oranı elde etmiştir. Aynı yıl Qin

vd. (2016), öncelikle balıkları arka plandan ayırarak özellik çıkarımı gerçekleştirmiş, ardından doğrusal SVM ile %98,64 gibi yüksek bir başarı oranı yakalamıştır. Bu durum, basit yapılı sistemlerle dahi etkili sonuçlar alınabileceğini göstermiştir.

Derin öğrenme yöntemlerinin kullanımıyla birlikte doğruluk oranlarında dikkate değer artışlar gözlemlenmiştir. Villona vd. (2018), CNN tabanlı bir modelle 20 farklı resif balık türünü tanımış ve ardından karar kurallarıyla tanımlama başarımını artırmıştır. Mandal vd. (2018) ise Faster R-CNN mimarisi ile bölge öneri ağı ve üç farklı sınıflayıcıyı entegre ederek %82,4 ortalama kesinlik (mAP) değeri elde etmiş; sistemin zorlu su altı koşullarında bile manuel analizlere kıyasla zaman ve maliyet avantajı sağladığını vurgulamıştır. Nair vd. (2018) tarafından önerilen sistemde ise hızlandırılmış sağlam öznitelikler kullanılmış ve SVM tabanlı sınıflayıcı ile etkili sonuçlar elde edilmiştir. Hridayami vd. (2019), VGG16 tabanlı transfer öğrenme yaklaşımı ile farklı görüntü türlerini (RGB, canny, hibrit) karşılaştırmış ve 50 balık türü üzerinde sınıflandırma denemeleri yapmıştır.

Rekha vd. (2020), veri artırma, nesne tespiti ve sınıflandırma süreçlerini içeren üç aşamalı bir CNN mimarisi önererek %90 tespit ve %92 sınıflandırma doğruluğu sağlamıştır. Kayaalp ve Metlek (2021), CNN ile öznitelik çıkarımı yaparak çok katmanlı yapay sinir ağı ile %73,72 doğruluk oranı yakalamıştır. Pudaruth vd. (2020) ise Mauritius kıyılarındaki 38 balık türünü tanımak için derin öğrenme tabanlı bir mobil uygulama geliştirmiştir. 1520 görüntüyle eğitilen sistem, TensorFlow modeliyle %98 doğruluk oranı elde etmiştir. Uygulama, internet bağlantısına gerek duymadan gerçek zamanlı balık tanıma yapabilmektedir.

Islam vd. (2021), balık görüntülerinin hem yerel hem de global özniteliklerini kullanarak karar ağacı ve K-NN algoritmalarını birleştirmiştir. Saleh vd. (2022) ise 2003–2021 yılları arasındaki literatürü tarayarak, derin öğrenme yöntemlerinin büyük miktarda veriyi hızlı ve doğru şekilde işleyerek deniz ekosistemlerini analiz etmede önemli avantajlar sunduğunu ortaya koymuştur. Gong vd. (2023), transfer öğrenme ve vizyon transformatörlerine dayalı sınıflayıcılarla hesaplama maliyetini azaltmayı hedeflemiştir.

Jayasundara vd. (2023), cep telefonu kameralarıyla elde edilen görüntüler üzerinden Hint Sardinella ve Sarı Yüzgeçli Tuna balıklarının kalitesini değerlendirmek için FishNET-S ve FishNET-T mimarilerini sunmuş ve sırasıyla %84,1 ve %68,3 doğruluk oranları elde

etmiştir. Liu vd. (2023), derin öğrenmeye dayalı nesne tespiti tekniklerini balık sayımı, boy ölçümü ve davranış analizi gibi akuakültür uygulamalarında detaylı şekilde incelemiştir. Burak Gülmez (2023), bitki hastalıklarının sınıflandırılması amacıyla CNN ve PSO (Parçacık Sürü Optimizasyonu) algoritmalarını entegre ederek optimize edilmiş bir model geliştirmiştir. Çankır vd. (2025) ise orman ve göl temalı uydu görüntülerini sınıflandırmak amacıyla MATLAB ortamında geliştirdikleri CNN modeli ile %95 doğruluk oranı elde etmiş ve çevresel izleme uygulamalarında derin öğrenmenin potansiyelini vurgulamıştır.

2.2 YOLO Modeli Kullanılarak Gerçekleştirilen Sınıflandırma Çalışmaları

Nesne tespiti alanında devrim niteliğinde bir yaklaşım sunan You Only Look Once (YOLO) mimarisi öne çıkmaktadır. Bu bölümde, YOLO algoritmasının çalışma prensipleri, çeşitli uygulama alanları ve nesne tanıma üzerine yapılan araştırmalar ele alınmıştır.

İlk çalışmalar, nesne tespiti için YOLO mimarisinin farklı alanlardaki potansiyelini göstermiştir. Tashiev vd. (2017), akıllı ulaşım sistemleri için YOLO tabanlı bir model geliştirerek trafik kamerası görüntülerinden sedan, tır gibi araçları başarıyla sınıflandırmıştır. Aynı yıl, Sung vd. (2017), su altı görüntülerinde balık tespiti için YOLO tabanlı bir evrişimli sinir ağı kullanmış ve gürültülü ve loş görüntülerde %93 sınıflandırma doğruluğu ile saniyede 16.7 kare hız elde etmiştir. Bu erken dönem çalışmaları, YOLO'nun hem karasal hem de su altı ortamlarında etkili olduğunu kanıtlamıştır.

YOLO'nun çok yönlülüğü, sonraki çalışmalarda farklı alanlara uygulanarak daha da belirginleşmiştir. Algur vd. (2018), dış ortamdaki insan hareketlerini sınıflandırmak için YOLO ile kişileri tespit etmiş, ardından özel bir CNN modeli ile hareketlerini (%96,86 doğruluk) sınıflandırarak hibrit bir sistem geliştirmiştir. Benzer şekilde, Xu ve Matzner (2018), su altı temiz enerji projelerinde balık tespiti için YOLO modelini kullanmış, ancak modelin yalnızca eğitimde kullanılan veri kümelerinde başarılı olduğunu, yeni veri kümelerinde yetersiz kaldığını gözlemlemiştir. Bu sonuç, modellerin genellenebilirliğinin önemini vurgulamıştır. Yusup vd. (2020), YOLO algoritmasıyla mercan resifi balık türlerini %82,82 genel doğrulukla sınıflandırmayı başarmıştır.

Bu alandaki araştırmalar, COVID-19 önlemlerinin takibi gibi güncel konulara da odaklanmıştır. Gurkan vd. (2021), pandemi döneminde maske tespiti ve sosyal mesafe ihlali

takibi için DenseNet-121 modelinin başarısı (%96,86 doğruluk) ve YOLOv3'ü entegre eden bütünleşik bir sistem geliştirmiştir. Sağlık alanında ise Karacı (2021), röntgen görüntülerinde omuz implantı üreticilerini tespit etmek için YOLOv3 ile DenseNet201'i birleştirerek %84,76 gibi yüksek bir doğruluk değeri elde etmiştir. Aynı yıl, Wageeh vd. (2021), su altı çiftliklerinde YOLO tabanlı nesne tespitiyle balıkları izlemiş ve saymıştır; bulanık su koşullarında görüntü netliğini artırmak için MSR algoritmasını kullanmış, böylece YOLO'nun zorlu koşullara adaptasyonunu göstermiştir.

YOLO'nun farklı versiyonları, performans ve verimlilik açısından sürekli olarak karşılaştırılmıştır. Altay ve Yılmaz (2023), T-hücre görüntülerini sınıflandırmak için YOLOv4 ve YOLOv5 modellerini uygulamış, her iki modelin de %95 doğruluk gösterdiğini, ancak YOLOv5'in daha kısa eğitim süresiyle öne çıktığını belirlemiştir. İnşaat güvenliği alanında Türkdamar vd. (2023), transfer öğrenmeli YOLOv5 modelinin işçi kaskı tespitinde %98 F1 skoru ile en yüksek performansı sergilediğini bulmuştur. Tarım sektöründe ise Dulkadir ve Gültekin (2023), otonom robotlar için muz olgunluk seviyelerini YOLOv5s, YOLOv8n ve YOLOv8m modelleriyle sınıflandırmış, her birinin %90'ın üzerinde ortalama kesinlik değeri elde ettiğini, özellikle YOLOv8n'in daha küçük boyutuyla yüksek doğruluk sağladığını göstermiştir. Kuswantori vd. (2023) ise, konveyör üzerindeki balık türlerini YOLOv4 ile optimize ederek %98,15 doğruluk oranına ulaşmıştır. Tıbbi görüntüleme alanında da YOLOv7 ve YOLOv7-tiny algoritmalarıyla beyin tümörlerinin tespit edildiği bir model geliştiren Yılmaz (2023), %97 doğruluk oranına ulaşarak YOLO'nun tanı süreçlerindeki güvenilirliğini kanıtlamıştır.

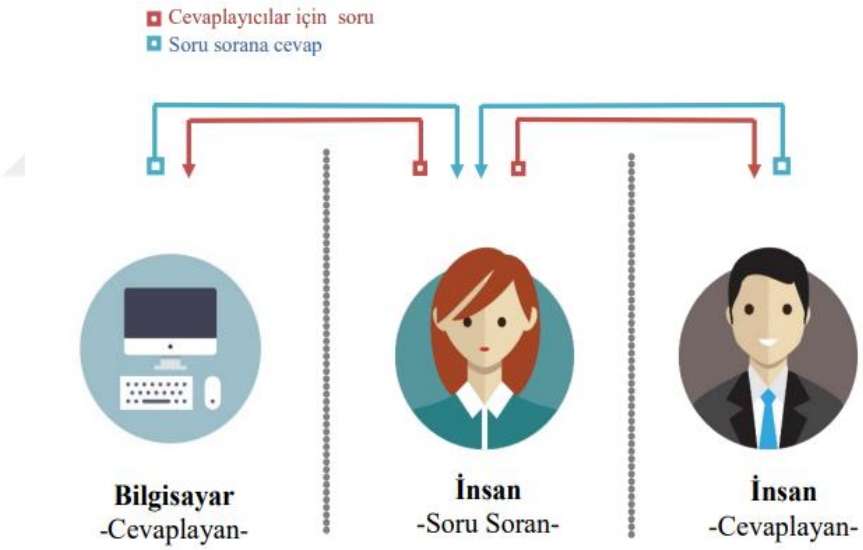
Rahman vd. (2024), su altı robotları için altı farklı balık türünü içeren bir veri setiyle YOLOv5m tabanlı bir sistem geliştirmiştir. Görüntü iyileştirme ve dikkat modülleri (CBAM) ile modelin doğruluğu artırılmış, hafifletilmiş ağ yapısı sayesinde gerçek zamanlı uygulamalar için uygun hale getirilmiştir. Bu çalışma, daha karmaşık sistemlerin gerçek dünya uygulamaları için nasıl optimize edilebileceğini göstermektedir.

3. MATERYAL VE METOT

Bu bölümde, yapılan çalışmanın daha iyi anlaşılabilmesi amacıyla çalışmada kullanılan temel kavramlar açıklanmıştır.

3.1 Yapay Zekâ Yaklaşımı

Yapay zekâ (AI) fikri, 1950'lerde Alan Turing'in "Makineler düşünebilir mi?" sorusuyla gündeme gelmiştir (Turing, 1950). Turing, bu soruya yanıt ararken, makinelerin zekâ seviyesini ölçmeyi amaçlayan bir test önermiştir. Bu test, daha sonra "Turing Testi" olarak adlandırılmıştır. Yapay zekâ terimi ise 1956 yılında Dartmouth Konferansında John McCarthy tarafından ortaya atılmıştır. Bu konferans, yapay zekâ araştırmalarının resmi başlangıcı olarak kabul edilir.



Şekil 3.1: Turing testi

Yukarıdaki şekil 3.1'de temel olarak bir makinenin insan benzeri düşünme yeteneğini değerlendirmek amacıyla geliştirilmiş yapay zekâ temelli bir sorgulama sistemidir. Orijinal formatında, bir sorgulayıcı insan, bir insan denek ve bir bilgisayar yer alır. Sorgulayıcı, belirli bir bağlamda sorular yöneltir ve amacına ulaşmak için hem insan denek hem de bilgisayarla etkileşim kurar. Belirli bir süre sonunda, sorgulayıcı, cevap verenlerden hangisinin insan ve hangisinin makine olduğuna karar vermeye çalışır. Eğer bilgisayar,

tekrarlanan testlerin en az yarısında sorgulayıcıyı ikna etmeyi başarırsa, "zeki bir makine" olarak kabul edilir (Turing, 1950; Arslan, 2020).

Miraç (2024)'a göre yapay zekâ, bilgisayarların veya bilgisayar kontrollü makinelerin, genellikle insan zekâsı gerektiren görevleri yerine getirebilme yeteneğidir. Bu, bilgisayar bilimi ve veri analitiğinin birleşimiyle, algoritmalar aracılığıyla tahminler yapma veya problemleri çözme yeteneğini içerir. Yapay zekâ, sağlık hizmetlerinden iklim değişikliğiyle mücadeleye kadar birçok alanda olumlu ilerlemeleri mümkün kılmaktadır.

YZ, genel bir üst kavram olup, makine öğrenimi ve derin öğrenme gibi alt alanları kapsamaktadır. Makine öğrenimi, yapay zekânın verilerden otomatik olarak öğrenebilen alt dalıdır. Derin öğrenme ise insan beynindeki sinir ağlarını model alarak çok katmanlı yapılar oluşturmayı hedeflemektedir. Özellikle büyük veri kümeleri üzerinde yüksek doğrulukla çalışabilen bu yöntemler, görüntü işleme ve doğal dil işleme gibi alanlarda çığır açan sonuçlar elde etmiştir (LeCun, 2015)

3.2 Makine Öğrenmesi Temelleri

Samuel (1959) tarafından 1950'lerde temelleri atılan makine öğrenmesi, Aksaç (2022)'a göre, bilgisayarların istatistiksel metotlar vasıtasıyla belirlenen hedeflere erişmesini sağlayan bir yapay zekâ alt dalıdır. Makine öğrenmesi, verilerden modeller oluşturmak için çeşitli yöntemleri içeren bir alandır ve bu süreçte kullanılan algoritmalar, veriyi anlamlı modellere dönüştüren temel araçlardır. En uygun algoritma seçimi, incelenen verinin özelliklerine, mevcut kaynaklara ve ulaşılmak istenen hedeflere göre değişiklik gösterir. Makine öğrenimi sistemleri, kendilerine sunulan girdi ve çıktı verilerini analiz ederek, bu veriler arasındaki ilişkileri ve kuralları öğrenir, bu nedenle geleneksel programlama yerine eğitim esastır. Örneğin, etiketlenmiş görselleri inceleyerek otomatik etiketleme yapmayı öğrenirler.

Makine öğrenmesi algoritmaları, farklı alanlarda yaygın olarak kullanılmaktadır. (Aylak, 2021):

Tablo 3.1: Makine öğreniminin kullanım alanları

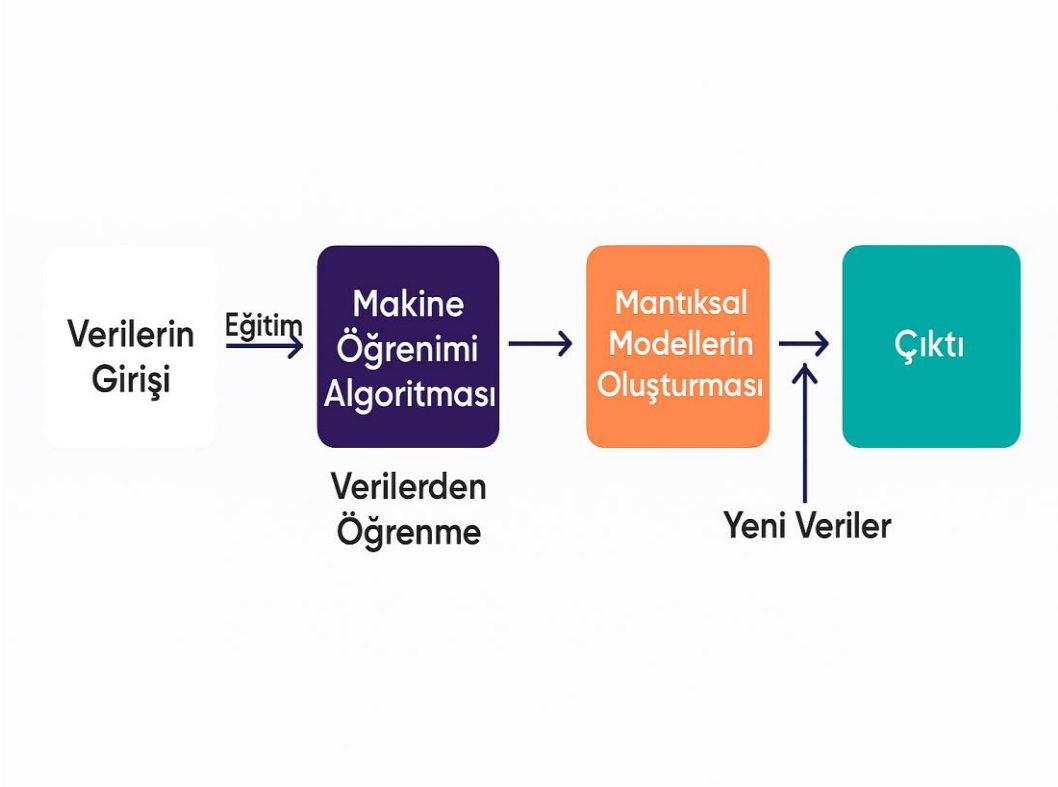
Alan	Uygulamalar
Bilgisayarlı görme	Yüz tanıma, nesne tanıma, robotik, otonom araçlar, vb.
İşaret	Dil İşleme vb.
Tıp	İlaç geliştirme, biyomedikal veri analizi, vb.
Coğrafya	Deprem Algılama, Hava Durumu Tahmin Etme, vb.
Finans ve E-ticaret	Analitik, sigorta, dolandırıcılık tespiti, finansal tahmin, spam e-posta, vb.

- **Bilgisayarlı Görme:** Bilgisayarla görme, dijital görüntülerin veya videoların otomatik olarak işlenmesi ve anlamlandırılması sürecidir. Yüz tanıma, nesne algılama, robotik sistemler ve otonom araçlar gibi uygulamalarla görüntü verisinden anlam çıkarılması hedeflenmektedir.
- **İşaret**: Doğal dil işleme ve konuşma işleme alanları, yazılı ve sözlü verilerin analizini içermekte olup, insan-makine etkileşiminin temelini oluşturmaktadır. Bu alanda kullanılan derin öğrenme modelleri, sesli komut sistemleri ve dil çeviri gibi uygulamalarda yaygın olarak yer almaktadır.
- **Tıp**: Derin öğrenme algoritmaları, ilaç geliştirme süreçlerinde moleküler düzeyde modellemelerden başlayarak biyomedikal veri analizi ve sinirbilim gibi alanlarda klinik karar destek sistemlerine kadar birçok aşamada etkin şekilde kullanılmaktadır.
- **Coğrafya**: Coğrafi bilgi sistemlerinde, özellikle deprem ve hava durumu gibi doğa olaylarının tespit ve öngörülmesinde derin öğrenme teknikleri kullanılarak büyük veri analizi yapılmakta, bu sayede hızlı müdahale ve önlem alınması sağlanmaktadır.
- **Finans ve E-ticaret:** Finansal verilerin analizi, sigorta sistemlerinde risk tahmini, dolandırıcılık tespiti, spam e-posta filtreleme ve finansal öngörü gibi alanlarda derin öğrenme yöntemleri sayesinde yüksek doğrulukta otomasyon sağlanmaktadır.

3.2.1 Makine Öğrenmesinin Çalışması

Makine öğrenmesi sistemleri, geçmiş verilere dayanarak öğrenir, bu verilerden tahmin modelleri geliştirir ve sonrasında sunulan yeni veriler için tahminlerde bulunur. Elde edilen tahminlerin doğruluğu, kullanılan veri miktarına bağlı olarak değişiklik gösterir; çünkü daha fazla veri, daha doğru ve güçlü bir modelin oluşturulmasını sağlar. Bu süreci daha iyi kavrayabilmek için, karmaşık bir problem karşısında belirli tahminlerde bulunmamız gerektiğini varsayalım. Bu tür durumlarda, doğrudan kod yazarak çözüm üretmek yerine, veriler genel makine öğrenmesi algoritmalarına sunulur. Algoritmalar bu verilerden yola çıkarak belirli mantık ilişkileri kurar ve buna uygun çıktılar üretir.

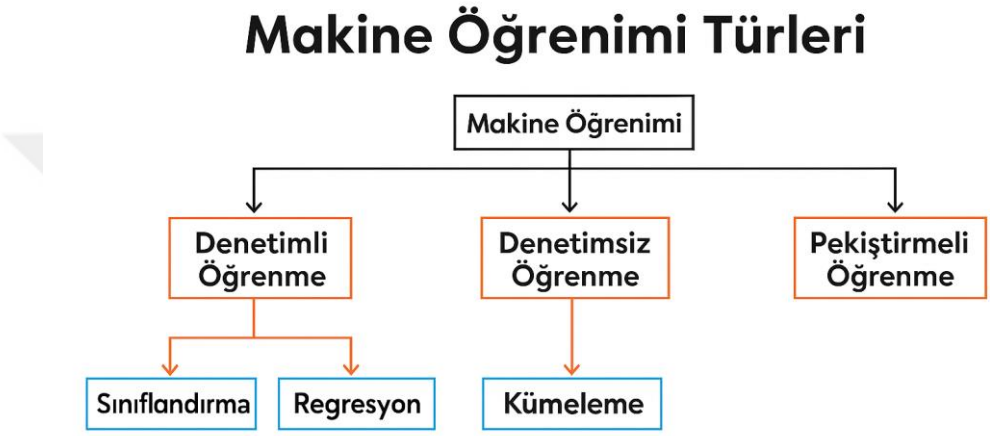
Makine öğrenmesi, problemlere yaklaşım biçimimizi daha sistematik, etkili ve verimli hale getirmiştir. Aşağıda Şekil 3.2'te sunulan blok diyagramı, bu algoritmaların çalışma prensibini görsel olarak ortaya koymaktadır.



Şekil 3.2: Makine öğreniminin işleyişi

3.2.2 Makine Öğrenmesinin Türleri

Makine öğrenmesi yaklaşımları ile makineler, karmaşık problemleri çözebilme yeteneği elde eder ve edindiği deneyimler doğrultusunda hem anlık hem de ileriye yönelik durumlarda karar verme kapasitesi geliştirir (Yılmaz S., 2023). Makine öğrenimi algoritmaları denetimli öğrenme, denetimsiz öğrenme ve pekiştirmeli öğrenme olmak üzere üç temel kategoride sınıflandırılmaktadır. Şekil 3.3'te, makine öğreniminin bu alt alanlarını göstermektedir.



Şekil 3.3: Makine öğreniminin bu alt kategorileri

Denetimli öğrenme yaklaşımı, etiketlenmiş örnek veri setleri kullanılarak makine öğrenmesi modellerinin eğitildiği bir metodolojiye dayanmaktadır. Bu yaklaşımda sistem, kendisine sunulan etiketli veriler aracılığıyla örnekler üzerinde analiz gerçekleştirerek bir model geliştirir. Eğitim süreci tamamlandıktan sonra, oluşturulan modelin güvenilirliği ve doğruluğu test verileri kullanılarak değerlendirilir (Veranyurt vd., 2020). Denetimli öğrenme yöntemleri iki ana kategoriye ayrılmaktadır:

- Sınıflandırma Algoritmaları: Bu kategori, çıktı değişkeninin önceden tanımlanmış kategorilerden birine ait olduğu problemleri kapsamaktadır. Örneğin, sonuçların evet-hayır şeklinde ikili kategoriye veya problem kapsamında mevcut olan diğer sınıflara ayrılabildiği durumlar bu gruba dahildir. Bu alanda kullanılan başlıca

algoritmalar arasında Rastgele Orman, Karar Ağacı, Lojistik Regresyon ve Destek Vektör Makinesi bulunmaktadır.

- Regresyon Algoritmaları: Bu yöntemler, girdi değerleri ile çıktı arasında sayısal bir ilişki kurulabildiği durumlarda uygulanır. Diğer bir ifadeyle, geçmiş veriler temelinde gelecekteki değerlerin tahmin edilebildiği problemler bu kategori altında incelenir. Hava durumu tahmini ve piyasa trend analizleri bu tür uygulamalara örnek teşkil eder. Bu kategoride kullanılan temel algoritmalar arasında Doğrusal Regresyon, Regresyon Ağacı, Doğrusal Olmayan Regresyon, Bayes Doğrusal Regresyon ve Polinom Regresyonu yer almaktadır.

Denetimsiz öğrenme, sistemin herhangi bir dış müdahale veya rehberlik olmaksızın veri setlerinden öğrenme sağladığı bir yaklaşımdır. Bu metodolojide, etiketlenmemiş ve sınıflandırılmamış veri kümeleri sisteme sunulur ve algoritmanın bu veriler üzerinde bağımsız olarak işlem yapması beklenir. Temel amaç, girdi verilerini yeni özellikler çerçevesinde yeniden yapılandırmak veya benzer karakteristiklere sahip nesne grupları oluşturmaktır. Denetimsiz öğrenme, kümeleme kategorisini içermektedir.

- Kümeleme Yöntemleri**:** Bu yaklaşım, ortak özelliklere sahip veri nesnelerini aynı grup içerisinde toplayan bir tekniktir. Bu süreçte, birbirleriyle uyumlu olmayan özellikler gösteren nesneler farklı gruplara ayrılır. Kümeleme analizi, veri setindeki nesneler arasındaki ortak noktaları tespit ederek bunları bu benzerlikler temelinde gruplandırma prensibine dayanır.

Pekiştirmeli öğrenme, geri bildirim mekanizması temelli bir makine öğrenmesi yaklaşımıdır. Bu sistemde, öğrenme ajanı doğru eylemler için ödüllendirilirken, hatalı davranışlar için cezalandırılır. Ajan, aldığı bu geri bildirimlerden yararlanarak otomatik öğrenme sürecini gerçekleştirir ve performansını sürekli olarak geliştirmeye çalışır. Pekiştirmeli öğrenme sürecinde, ajan çevresi ile aktif bir etkileşim kurarak keşif sürecini verimli bir şekilde yürütür. Ajanın temel hedefi, maksimum ödül puanlarına ulaşarak genel performansını optimize etmektir.

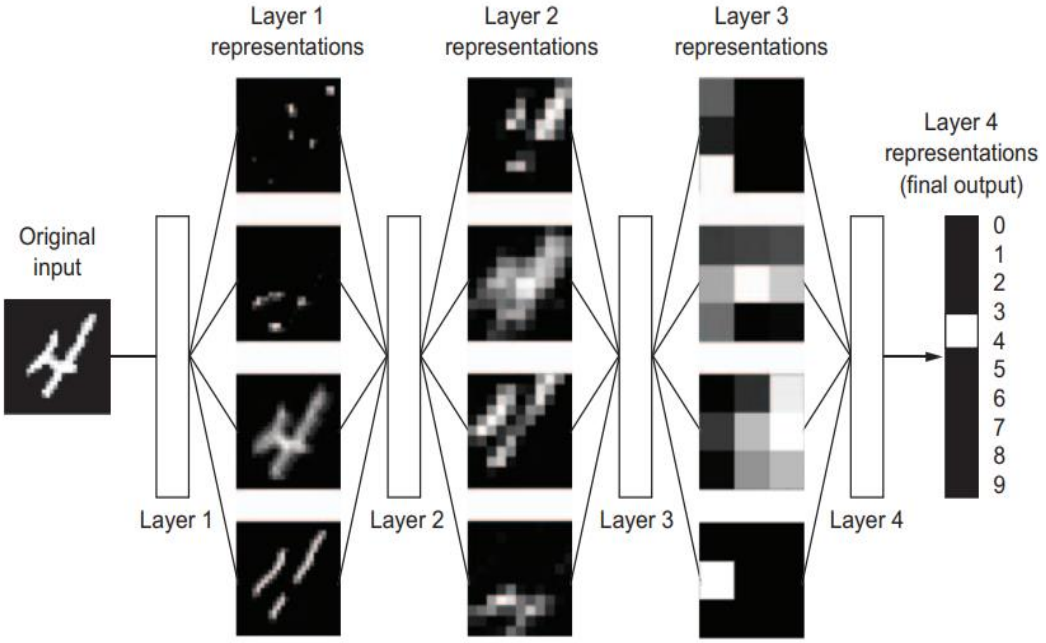
3.3 Derin Öğrenme Yöntemleri

Derin öğrenme, yapay zekâ temellerinden yola çıkarak makine öğrenmesi teknikleriyle geliştirilmiş bir yaklaşımdır; bu yöntemde veriler çok katmanlı yapılar arasında işlenerek sistemin hızlı öğrenme ve hedeflenen sonuçlara ulaşma kapasitesi sağlanır. (Tan vd., 2021). Şekil 3.4'de, yapay zekâ, makine öğrenmesi ve derin öğrenme arasındaki bağlantıyı görsel olarak sunmaktadır.

Şekil 3.4: Yapay zekâ, makine öğrenmesi, derin öğrenme bağlantısı

Derin öğrenme, makine öğrenmesinin alt dalı olup, verilerin temsillerini öğrenmeye yönelik çok katmanlı yapılar aracılığıyla çalışan bir yöntemdir. Bu yaklaşımda "derinlik" kavramı, modelde yer alan katman sayısını ifade eder. Geleneksel makine öğrenmesi teknikleri genellikle yalnızca bir veya iki katmanlı temsillere dayanırken, derin öğrenme sistemleri onlarca hatta yüzlerce ardışık temsil katmanı kullanarak daha karmaşık örüntüleri öğrenebilmektedir. Bu katmanlı yapılar genellikle yapay sinir ağları ile gerçekleştirilmekte ve tüm parametreler eğitim verileri aracılığıyla otomatik biçimde öğrenilmektedir. Her ne

kadar "sinir ağı" terimi nörobiyolojik bir çağrışım taşısa da derin öğrenme yöntemlerinin biyolojik sinir sistemleriyle doğrudan bir ilişkisi bulunmamaktadır. Derin öğrenme, bu yönüyle özellikle görüntü işleme, doğal dil işleme, nesne tanıma ve sağlık bilişimi gibi birçok alanda etkin biçimde kullanılmaktadır (Chollet, 2021).



Şekil 3.5: Bir rakam sınıflandırma modeli tarafından öğrenilen derin temsiller (Chollet, 2021).

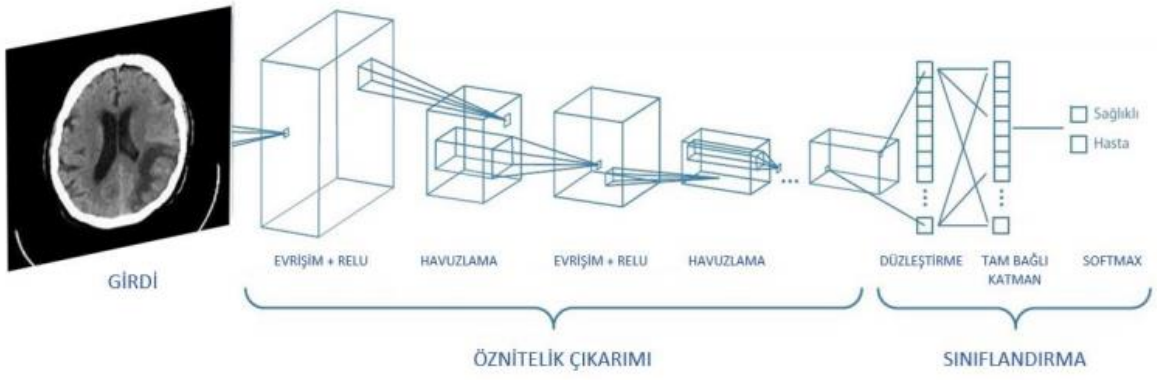
Şekil 3.5'te görüldüğü gibi, yapay sinir ağı, giriş olarak verilen rakam görselini katmanlar aracılığıyla kademeli olarak işler. Her bir katmanda, giriş verisinin temsili dönüşüme uğrayarak daha soyut ve görevle ilişkili bir yapıya ulaşır. Başlangıçta yalnızca basit kenar ve şekil gibi düşük seviyeli özellikler çıkarılırken, sonraki katmanlarda bu özellikler birleştirilerek daha yüksek seviyeli örüntüler ve anlamlı yapılar oluşturulur. Bu süreçte her katman, bir önceki katmandan gelen bilgileri filtreleyip dönüştürerek daha karmaşık ve hedefe yönelik temsiller üretir. Son katmanda ise, tüm bu işlem basamaklarının sonucunda ağ, görselin hangi rakama ait olduğunu belirleyebilecek düzeyde anlamlı ve özetlenmiş bilgiye ulaşır. Bu işleme süreci, bilginin katmanlar boyunca damıtılarak sadeleştirildiği ve nihayetinde sınıflandırma gibi görevlerde kullanılabilecek bir temsil düzeyine ulaştırıldığı bir bilgi işleme zinciri olarak değerlendirilebilir (Chollet, 2021).

Derin öğrenme, temel olarak verilerin temsilini çok katmanlı yapılar aracılığıyla öğrenmeyi amaçlayan bir yöntemdir. Bu yaklaşım kavramsal olarak basit görünmekle birlikte, uygun

şekilde ölçeklendirildiğinde ve yeterli veri ile desteklendiğinde, oldukça karmaşık ve etkileyici sonuçlar üretebilmektedir.

3.4 Konvolüsyonel Sinir Ağları (CNN)

Konvolüsyonel Sinir Ağları (CNN), derin öğrenmenin önemli bir alt dalı olup, temelleri 1988 yılında Yann LeCun tarafından atılan ve 1998'e kadar geliştirilen LeNet mimarisine dayanır. LeNet ağında, alt katmanlar sıralı konvolüsyonel ve maksimum havuzlama katmanlarından oluşurken, üst katmanlar geleneksel çok katmanlı yapay sinir ağlarına karşılık gelmekteydi. Günümüzde CNN'ler, büyük veri kümelerinde sıkça kullanılan çok katmanlı yapay sinir ağları olarak derin öğrenme alanında öne çıkmaktadır. Özellikle tıp ve görüntü işleme gibi alanlarda yoğun olarak kullanılan bu algoritmalar, görüntü işleme görevleri için tercih edilen başlıca yöntemdir. Bir CNN, özellikleri saptayan "konvolüsyonel katman", sisteme doğrusal olmayanlığı kazandıran "doğrusal olmama katmanı", ağırlık sayısını azaltıp uygunluğu kontrol eden "havuzlama (örnekleme) katmanı", özellik sayısını tek boyuta dönüştüren "düzleştirme katmanı" ve sınıflandırma yapan "tam bağlı katman" gibi çeşitli bileşenlerden oluşur. Temelde sınıflandırma sorunlarını standart bir sinir ağıyla çözen CNN'ler, ayrıştırıcı bilgileri ve spesifik özellikleri belirlemek için bu farklı katmanları bütünsel bir şekilde kullanır (Ghrairi, 2019). Konvolüsyonel sinir ağları modelinin genel mimarisi Şekil 3.6'da gösterilmiştir.



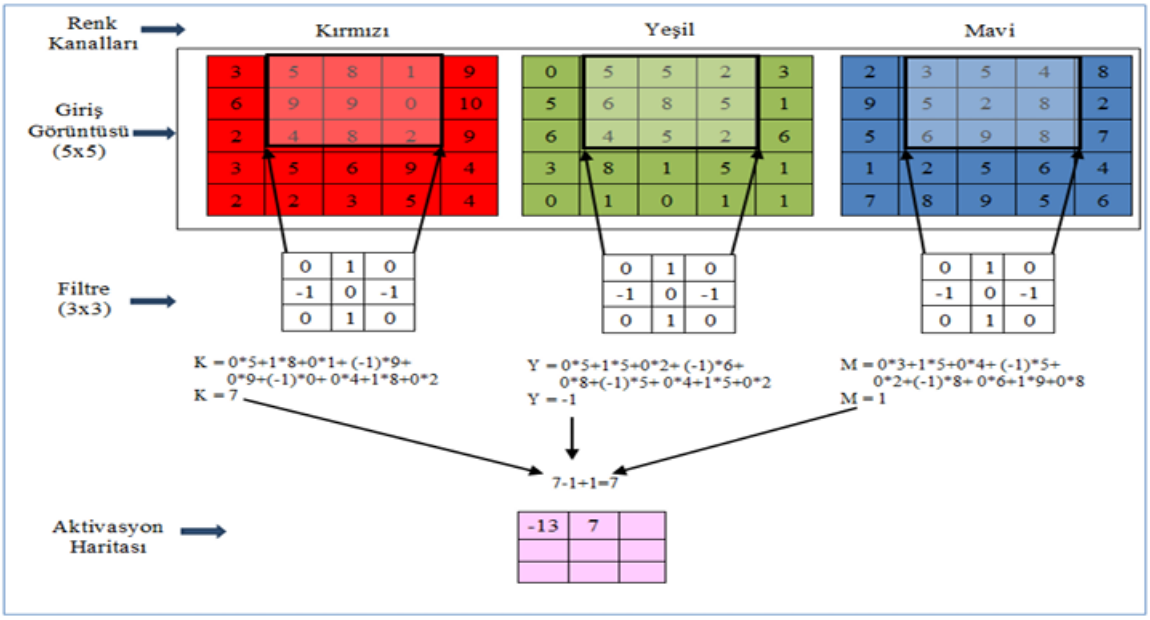
Şekil 3.6: Konvolüsyonel sinir ağları modelinin genel mimarisi

Giriş katmanı, konvolüsyonel sinir ağının eğitim sürecinde kullanılacak veri matrislerinin sisteme dahil edildiği temel bileşen olarak işlev görmektedir. Bu katmanda sunulan veri setlerinin boyutsal büyüklüğü ile ağın başarı performansı arasında pozitif bir korelasyon

bulunmakla birlikte, bu durum eğitim ve test süreçlerinin uzamasına da neden olmaktadır. Giriş katmanına aktarılan verilerin hedeflenen çıktıya yakınlığı, veri seçim sürecinin doğruluğu ile doğru orantılı bir ilişki sergilemektedir. Buna karşın, düşük çözünürlüklü ve sınırlı sayıda veri ile gerçekleştirilen eğitim süreçleri, sistemin çıktı başarı oranında önemli ölçüde azalmaya yol açmaktadır. Bu durum, veri kalitesi ve miktarının model performansı üzerindeki kritik etkisini ortaya koymakta ve optimal sonuçlar için dengeli bir veri seçim stratejisinin gerekliliğini vurgulamaktadır (İnik ve Ülker, 2017).

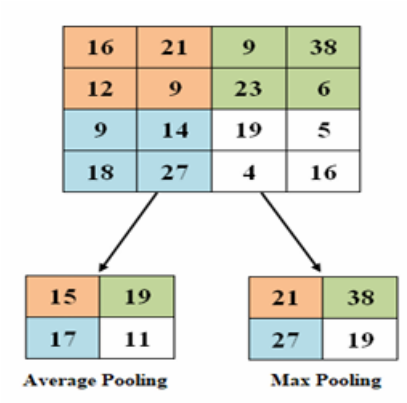
Bir konvolüsyon katmanının temel görevi, giriş verisinden anlamlı özellikleri çıkarmaktır. Konvolüsyon, iki fonksiyon arasında uygulanan matematiksel bir işlemdir. CNN yapısında bu işlem, çekirdek ya da filtre adı verilen küçük bir matrisin, giriş verisi üzerinde kaydırılarak her pozisyonda karşılık gelen elemanlarla çarpılması ve elde edilen değerlerin toplanması şeklinde gerçekleştirilir. Bu kaydırma işlemi sırasında her pencere için hesaplanan toplam değer, ilgili bölgenin çıktısını oluşturur. Filtrenin tüm veri boyunca kaydırılmasıyla birlikte, özellik haritası olarak adlandırılan konvolüsyon katmanının çıktısı meydana gelir (Ghrairi, 2019).

Konvolüsyonel Sinir Ağlarının (CNN) temel bileşenlerinden biri olan konvolüsyon katmanı, dönüşüm katmanı olarak da adlandırılmakta ve görüntü üzerinde belirli bir filtrenin sistematik olarak kaydırılması esasına dayanmaktadır. Bu filtreler, mimarinin temel unsurlarından biri olup genellikle 2×2, 3×3 veya 5×5 boyutlarında tanımlanır. Konvolüsyon işlemi sırasında, filtre bir önceki katmandan gelen giriş görüntüsü üzerine belirli bir adım aralığıyla (stride) uygulanarak, her bir konumda eleman bazlı çarpımların toplamı hesaplanır ve bu işlem tüm görüntü matrisi boyunca tekrarlanır. Bu işlem sonucunda ortaya çıkan aktivasyon haritası (özellik haritası), filtrelerin algıladığı belirli örüntüleri temsil eder. CNN'lerin eğitimi sürecinde filtre katsayıları, öğrenme algoritması doğrultusunda güncellenerek modelin verideki anlamlı bölgeleri tanımasını sağlar. Örneğin, 5×5 boyutundaki renkli (RGB) bir görüntü için uygulanan 3×3 boyutundaki bir filtre, her bir renk kanalı üzerinde ayrı ayrı işlenmekte ve sonuçların toplamı ile aktivasyon haritası oluşturulmaktadır. Elde edilen aktivasyon değerleri, genellikle filtre katsayılarının toplamına bölünerek normalize edilir; ancak filtre ağırlıkları toplamının sıfır olduğu durumlarda bu işlem gerçekleştirilmemektedir. Bu yapı sayesinde, CNN mimarileri düşük seviyeli özellikleri yakalayarak üst düzey temsillerin kademeli olarak öğrenilmesine katkı sağlamaktadır. (İnik ve Ülker, 2017).



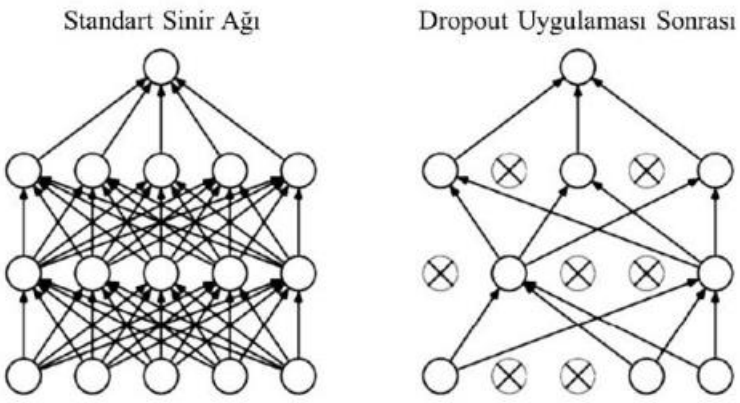
Şekil 3.7: Üç kanallı 5x5 boyutundaki görüntü matrisine 3x3'lük filtrenin uygulanmasıyla gerçekleştirilen konvolüsyonel dönüşüm süreci (İnik ve Ülker, 2017).

Havuzlama (pooling) katmanı, konvolüsyonel sinir ağlarında kullanılan önemli bir bileşendir ve temel amacı, modeldeki parametre sayısını azaltarak işlem süresini kısaltmak ve hesaplama yükünü hafifletmektir. Bu katman, giriş görüntüsünü daha küçük parçalara ayırır ve her parçadan temsil edici bir değer alarak boyut indirgeme işlemi yapar. Genellikle iki tür havuzlama yöntemi kullanılır: maksimum havuzlama (max pooling) ve ortalama havuzlama (average pooling). Maksimum havuzlama, seçilen alan içindeki en büyük değeri çıkışa geçirirken; ortalama havuzlama, o alandaki değerlerin ortalamasını alarak çıkışa verir. Örneğin, 4x4 boyutundaki bir görüntüye 2x2 boyutunda pencere uygulanarak dört küçük alana bölünür. Bu alanlardan max pooling, her birinin en yüksek değerini alırken; average pooling, sayıların ortalamasını alır ve çıkışa aktarır. Böylece ağ, daha küçük boyutlu ama önemli bilgileri koruyan bir temsil elde eder. Bu yöntem hem modelin öğrenme sürecini hızlandırır hem de aşırı öğrenmeyi (overfitting) azaltmaya katkı sağlar (Taşhan, 2017).



Şekil 3.8: Maksimum ve ortalama havuzlama yöntemlerinin görsel gösterimi (Taşhan, 2017).

Evrişimsel Sinir Ağlarında (ESA) büyük veri kümeleriyle yapılan eğitim süreçlerinde modelin veriyi ezberlemesi, yani aşırı öğrenme (overfitting) problemi ortaya çıkabilir. Bu durumun önüne geçmek amacıyla Dropout katmanı kullanılır. Dropout katmanında uygulanan temel prensip, eğitim sırasında ağın bazı düğümlerinin (nöronlarının) rastgele devre dışı bırakılmasıdır. Böylece ağ, tüm düğümlere bağımlı kalmaksızın daha genelleyici ve dayanıklı özellikler öğrenebilir. Bu yöntem, modelin yalnızca eğitildiği verilere özel kalıpları ezberlemesini önleyerek, farklı veri örneklerine karşı da başarılı sonuçlar üretmesini sağlar (İnik ve Ülker, 2017).



Şekil 3.9: ESA ağına dropout katmanın uygulaması

Sınıflandırma katmanı, genellikle tam bağlantılı katmandan sonra gelen ve derin öğrenme mimarilerinde sınıflandırma işleminin gerçekleştirildiği son katmandır. Bu katmanın çıkış boyutu, modelin sınıflandıracağı sınıf (nesne) sayısına eşittir. Örneğin, eğer sistem 15 farklı nesneyi sınıflandıracak şekilde tasarlanmışsa, bu katmanın çıkış birimi sayısı da 15

olmalıdır. Tam bağlantılı katmandan gelen örneğin 4096 boyutundaki çıkış vektörü ile sınıflandırma katmanı arasında 4096x15 boyutunda bir ağırlık matrisi oluşturulur. Bu katmanda genellikle softmax gibi çok sınıflı sınıflandırmalarda başarıyla kullanılan aktivasyon fonksiyonları tercih edilir. Softmax fonksiyonu her bir sınıfa ait olasılığı 0 ile 1 arasında bir değerde ifade eder ve en yüksek olasılığa sahip sınıf, modelin tahmin ettiği sınıf olarak kabul edilir (İnik ve Ülker, 2017).

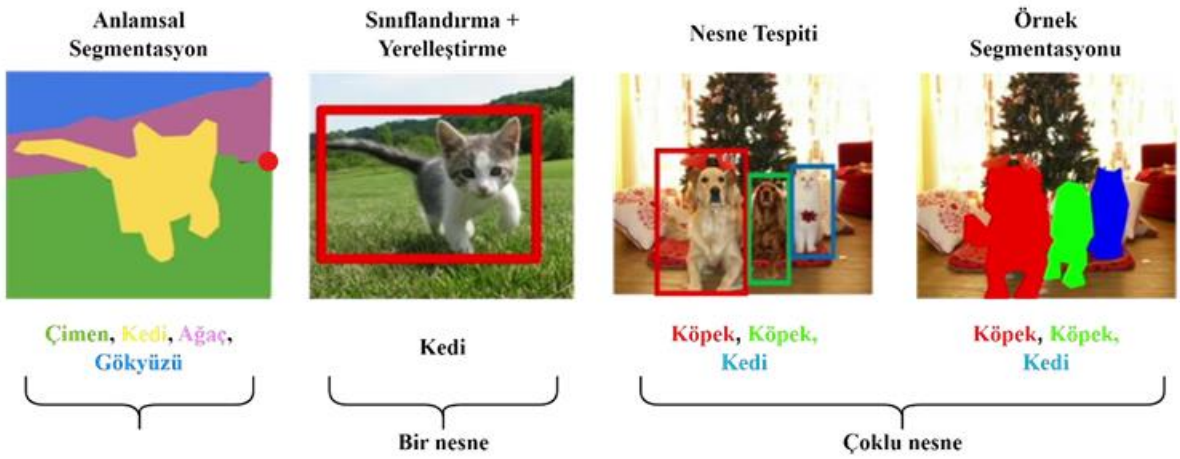
3.5 Nesne Tespiti Yöntemleri

Yapay zekâ teknolojileri, hava durumu tahmini, ekonomik analizler, haber metinlerinin otomatik oluşturulması, sanat üretimi, bireysel sağlık izleme, mühendislik hesaplamaları ve görüntü işleme gibi çeşitli alanlarda yaygın bir şekilde kullanılmaktadır. Bu uygulamalardan biri de nesne tespiti olup, bir görüntüdeki nesnelerin arka plandan ayrıştırılarak belirgin hâle getirilmesi sürecini ifade etmektedir. Başka bir ifadeyle nesne tespiti, görüntü içerisinde yer alan belirli nesnelerin konumlarının ve hangi sınıfa ait olduklarının belirlenmesidir. Bu süreçte nesneler genellikle dikdörtgen çerçeveler ile gösterilmektedir. Nesne tespiti, bilgisayarlı görü alanının temel problemlerinden biri olarak değerlendirilmekte olup; yüz tanıma, yaya algılama, insan uzuvlarının belirlenmesi, nesne takibi ve yüzey alanı ölçümü gibi çok sayıda gerçek dünya probleminin çözümünde temel rol oynamaktadır. Bu tür uygulamaların başarısı ise doğrudan etkili bir nesne algılama algoritmasının kullanılmasına bağlıdır (Aktürk ve Serbest, 2022).

Nesne tespiti, bilgisayar görüşü alanında görüntü veya videolardaki belirli nesnelerin hem konumunu hem de kategorisini belirlemeyi amaçlayan temel bir görevdir. Bu süreçte, tespit edilen nesneler genellikle sınırlayıcı kutular (bounding box) içine alınarak konumları işaretlenir ve ardından bu nesnelerin hangi sınıfa (örneğin, kedi, araba, sandalye) ait olduğu sınıflandırma ile belirlenir. Ancak, sadece sınıflandırma yapmak, bir görüntüde çok sayıda ve farklı sınıflara ait nesneler bulunduğunda yetersiz kalabilir; bu durum, nesnelerin konumlarının doğru bir şekilde tespit edilmesinin önemini ortaya koyar (Hanbay ve Üzen, 2017).

Nesne tespiti, Toplama veya nesneye yönelik işlemlerde, hedef nesnenin konumunun doğru bir şekilde belirlenmesi oldukça kritiktir. Bu amaçla geliştirilen nesne algılama algoritmaları, özellikle çok sayıda nesne barındıran görüntülerde, sensörlerden elde edilen

verileri analiz ederek hedefin yerini saptamaya çalışır. Bununla birlikte, bilgisayarla görme alanında yalnızca nesne tespiti değil; anlamsal segmentasyon, sınıflandırma, yerelleştirme ve örnek segmentasyonu gibi başka görevler de yer alır. Bu görevlerin her biri görüntüler üzerinde farklı düzeylerde bilgi çıkarmayı amaçlar ve aralarındaki temel ayrımlar Şekil 3.10'da görsel olarak özetlenmektedir (Rieder ve Verbeet, 2019).



Şekil 3.10: Görüntü işlemede nesne algılama yaklaşımlarının karşılaştırılması (Rieder ve Verbeet, 2019).

Bilgisayarın görsel dünyayı anlamasına yönelik farklı seviyelerde yaklaşımlar bulunuyor. Bunların başında gelen Anlamsal Segmentasyon, bir görüntüdeki her pikselin ait olduğu sınıfı belirleyerek (örneğin gökyüzü, ağaç, kedi) genel bir harita çıkarır; ancak aynı sınıftan farklı nesneleri birbirinden ayırmaz, yani birden fazla kedi olsa da hepsini tek bir "kedi" alanı olarak görür. Buna karşılık, Sınıflandırma + Yerelleştirme tek bir ana nesneye odaklanır; hem onun ne olduğunu ("kedi") söyler hem de görüntüdeki yerini bir kutuyla işaretler, böylece o nesnenin tam konumunu da belirlemiş oluruz. Daha karmaşık sahneler için geliştirilen Nesne Tespiti ise, bir görüntüdeki birden fazla nesneyi aynı anda tanıyıp her birini ayrı ayrı kutular içine alarak sınıflandırır (örneğin hem kedileri hem köpekleri tek tek işaretler). En detaylı yöntem olan Örnek Segmentasyonu ise anlamsal segmentasyon ve nesne tespitinin birleşimidir; yalnızca her pikselin sınıfını belirtmekle kalmaz, aynı zamanda aynı sınıfa ait olsalar bile her bir bireysel nesneyi piksel düzeyinde birbirinden ayırır (örneğin iki kediyi farklı renklerle belirginleştirir). Bu yöntemlerin her biri, farklı uygulama ihtiyaçlarına göre görsel veriden bilgi çıkarımında özel görevler üstlenir (Rieder ve Verbeet, 2019).

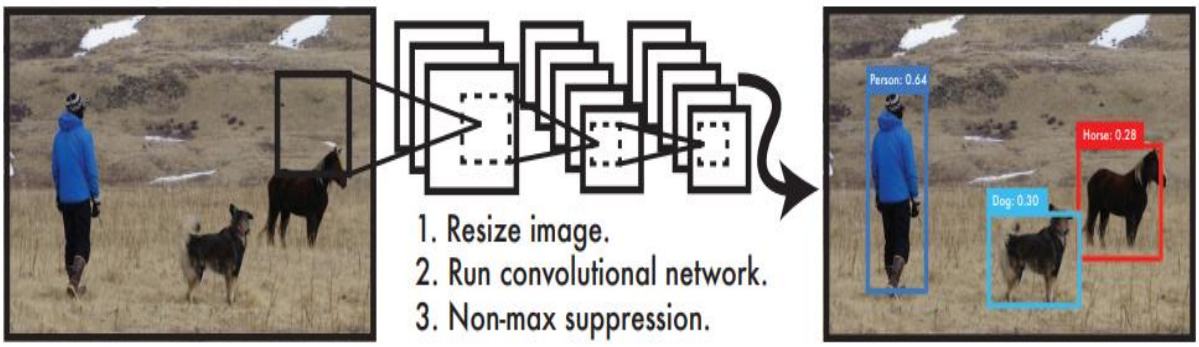
Modern derin öğrenme tabanlı nesne tespit sistemleri, işleyiş biçimlerine göre tek aşamalı ve iki aşamalı olmak üzere iki ana gruba ayrılır. İki aşamalı dedektörler, tespit doğrulukları açısından oldukça başarılı olsalar da, hızları tek aşamalı rakiplerine göre daha düşüktür. Bu grubun önde gelen örnekleri arasında R-CNN, Fast R-CNN ve Faster R-CNN yer almaktadır. Bu serinin ilk halkası olan R-CNN, önceden eğitilmiş Evrişimsel Sinir Ağlarını (CNN) kullanarak özellik çıkarımı yapar; ancak yaklaşık 2000 adet bölge önerisi üretmesi ve her bir bölge üzerinde ayrı ayrı hesaplamalar yapması nedeniyle oldukça yavaştır, bu da onu gerçek zamanlı uygulamalar için pek uygun kılmaz. R-CNN'in bu yavaşlığını gidermek amacıyla geliştirilen Fast R-CNN daha hızlı çalışsa da, bölge önerilerini bulmak için Destek Vektör Makineleri (SVM) kullanması nedeniyle hala zaman alıcıdır. Bu zincirin en gelişmiş halkası olan Faster R-CNN ise önceki iki yönteme göre daha iyi sonuçlar verir. Ancak onun da dezavantajı, tüm görüntüyü tek seferde işlemeye odaklanmak yerine, görüntüyü parçalara bölerek analiz etmesi ve bu durumun nesne tespiti sürecini yine de yavaşlatmasıdır (Miraç, 2024).

Tek aşamalı nesne tespit yöntemleri arasında dikkat çekenler SSD, RetinaNet ve YOLO'nun farklı versiyonlarıdır. YOLO'nun başlangıçtaki sürümleri, nesne tespitini oldukça hızlı gerçekleştirmelerine rağmen, doğruluk oranları diğer tek aşamalı dedektörlere kıyasla daha düşüktü. Ancak, algoritmanın sonraki ve geliştirilmiş versiyonlarında bu doğruluk açığı başarıyla kapatılmıştır. YOLO'yu diğer yaklaşımlardan ayıran en önemli özelliklerden biri, gerçek zamanlı çalışabilme kapasitesidir; bu, görüntü akışlarında nesneleri anında ve minimum gecikmeyle tespit edebilme anlamına gelir. Bu evrimle birlikte YOLO, artık sadece hızlı değil, aynı zamanda yüksek doğrulukta tespit yapabilmekte, çok çeşitli nesne sınıflarını ayırt edebilmekte ve geniş veri setleri ile yüksek çözünürlüklü görüntülere kolayca uyum sağlayabilmektedir. Bu sayede YOLO, nesne tespiti uygulamaları için önemli avantajlar sunan bir yöntem haline gelmiştir (Tan vd., 2021).

Önceki açıklamada bahsedilen tüm bu üstün özellikler (hız, yüksek doğruluk, çoklu sınıf tanıma ve farklı veri veya görüntü boyutlarına kolay uyum) göz önünde bulundurularak, bu çalışmada nesne tespiti görevi için YOLO algoritması tercih edilmiştir.

3.6 YOLO Nesne Tespit Algoritması

Nesne tespiti alanında derin öğrenme tabanlı bir yaklaşım olan YOLO (You Only Look Once), bir görüntüyü tek bir geçişte analiz ederek, nesnelerin hem uzamsal konumlarını hem de ilgili sınıflarını eş zamanlı olarak tahmin etmektedir. Bu tek aşamalı mimari, yöntemi hızlı ve gerçek zamanlı nesne tespiti uygulamaları için oldukça etkili hale getirmektedir. Ağır sinir ağı modellerini kullanarak karmaşık nesne örüntülerini öğrenme kabiliyetine sahip olan YOLO, aynı zamanda birden fazla nesne sınıfını yüksek bir hassasiyetle algılayarak geniş bir uygulama alanında kullanıma olanak tanımaktadır (Redmon vd., 2016).



Şekil 3.11: YOLO algılama sistemi (Redmon vd., 2016).

Şekil 3.11'de de görüldüğü gibi, YOLO'nun tek bir evrişimli ağı, görüntüdeki hem nesnelerin yerini belirleyen kutuları hem de bu nesnelerin hangi sınıfa ait olduğunu aynı anda tahmin etmektedir. Böylece model, yalnızca bireysel nesneler üzerine değil, aynı zamanda görüntünün tamamı ve içerisindeki tüm nesneler üzerinde bütünsel bir değerlendirme yapma yeteneği kazanmaktır. YOLO'nun bu mimarisi, uçtan uca (end-to-end) eğitim ve gerçek zamanlı tespit yeteneği ile yüksek ortalama doğruluk (average precision) sağlamaktadır (Redmon vd., 2016).

Şekil 3.12 sistem, giriş görüntüsünü S × S boyutunda bir ızgaraya böler ve bir nesnenin merkezi bir hücreye denk geliyorsa, o hücre bu nesnenin tespitiyle sorumlu hale gelir. Her bir ızgara hücresi, B adet sınır kutusu (bounding box) ve bu kutulara ait güven skoru (confidence score) tahmin eder. Güven skoru, hem kutunun gerçekten bir nesne içerip içermediğine olan güveni hem de bu kutunun doğru olup olmadığına ilişkin modelin inancını yansıtır.

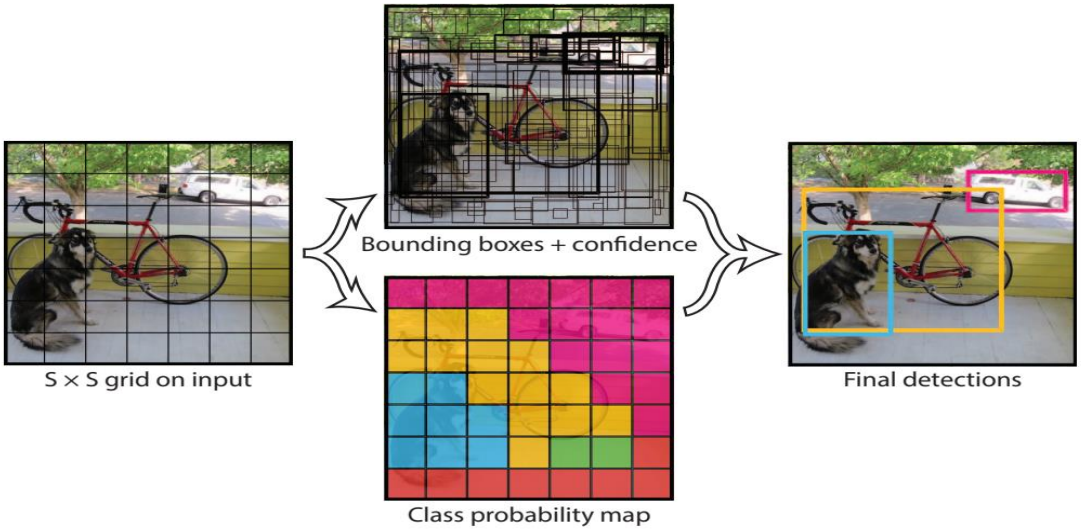
Bu güven değeri şu şekilde ifade edilir:

$$\mathbf{P(Nesne)} \times \mathbf{IoU}_{\boldsymbol{tahmin}}^{\boldsymbol{gerçek}}$$

Eğer hücrede nesne yoksa, güven skoru sıfır olmalıdır. Ancak bir nesne mevcutsa, bu skor ile tahmin edilen kutu ile gerçek kutu arasındaki örtüşme oranı olan **IoU** (Intersection Over Union) eşit olmalıdır. Her sınır kutusu 5 parametreden oluşur: (x, y, w, h, güven skoru). Burada (x, y) koordinatları kutunun hücreye göre merkezini, w ve h kutunun genişlik ve yüksekliğini, güven skoru ise tahmin edilen kutunun herhangi bir gerçek kutu ile olan **IoU** değerini belirtir. Buna ek olarak her hücre, C adet koşullu sınıf olasılığı $\boldsymbol{P_r}(\mathbf{Sınıf_i} \mid \mathbf{Nesne})$ tahmin eder. Bu olasılıklar, yalnızca hücre bir nesne içerdiğinde hesaplanır. Test aşamasında, sınıfa özgü güven skoru elde etmek için koşullu sınıf olasılığı ile güven skorunun çarpımı alınır:

$$\mathbf{P(Sınıf_i} \mid \mathbf{Nesne)} \times \mathbf{P(Nesne)} \times \mathbf{IoU}_{\boldsymbol{tahmin}}^{\boldsymbol{gerçek}} = \mathbf{P(Sınıf_i)} \times \mathbf{IoU}_{\boldsymbol{tahmin}}^{\boldsymbol{gerçek}}$$

Bu çarpım sonucu elde edilen değer, hem belirli bir sınıfa ait nesnenin kutu içerisinde bulunma olasılığını hem de kutunun tahmin edilen nesneye ne ölçüde uyum sağladığını ifade etmektedir (Redmon vd., 2016).

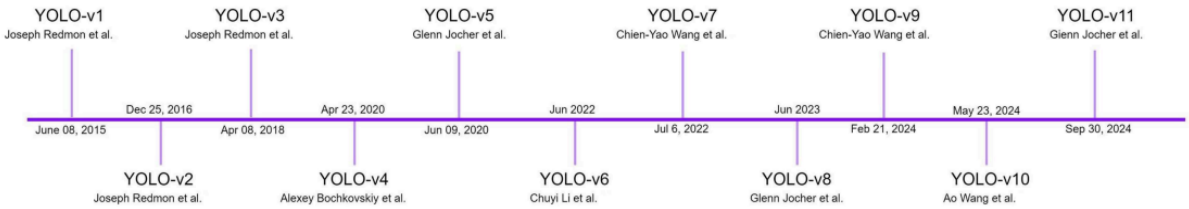


Şekil 3.12: YOLO'nun nesne tespiti adımları (Redmon vd., 2016).

Görüntüyü S × S ızgarasına böler ve her ızgara hücresi için B sınırlayıcı kutu, bu kutular için

güvenilirlik ve C sınıf olasılıkları tahmin eder. Bu tahminler, S × S × (B ∗ 5 + C) tensörü olarak kodlanmıştır (Redmon vd., 2016).

YOLO, görüntülerdeki nesneleri verimli bir şekilde konumlandırmak ve sınıflandırmak için tasarlanmış gerçek zamanlı bir nesne algılama çerçevesidir. Bilgisayar görüşünde önemli bir görev olan nesne algılama, bu hedefleri gerçekleştirmek için sinir ağlarından yararlanır. Bu bölüm, Şekil 3.13'te gösterildiği gibi YOLO-v1'den YOLO-v11'e kadar YOLO varyantlarının evrimine odaklanmaktadır (Mao ve Hong, 2025).



Şekil 3.13: YOLO sürümlerinin evrimi

YOLO algoritmaları, 2016 yılından itibaren nesne tespiti alanında gerçek zamanlı çözümler sunarak derin öğrenmenin en etkili uygulamalarından biri haline gelmiştir. YOLOv1 ile temelleri atılan bu yaklaşım, her yeni sürümle birlikte hem hız hem doğruluk açısından önemli iyileştirmeler geçirmiştir. Aşağıda, YOLO algoritmasının v1'den v11'e kadar olan tüm sürümleri, teknik gelişmeleri ve kullanım alanları açısından özetlenmiştir (Mao ve Hong, 2025).

YOLOv1 (2016), nesne tespiti görevini tek bir evrişimli sinir ağı ile gerçekleştiren ilk model olarak devrim niteliğindedir. Görüntüyü 7x7'lik bir ızgaraya bölerek, her hücrede sınırlayıcı kutu ve sınıf olasılıklarını eş zamanlı tahmin etmesi, onu oldukça hızlı hale getirmiş ve yaklaşık 45 FPS ile gerçek zamanlı uygulamalara uygunluğunu göstermiştir. Bu birleşik öğrenme yapısı, uygulamanın sadeliği ve sınırlı donanımda dahi çalışabilme yeteneği gibi avantajlar sunmuştur. Ancak, küçük ve yakın nesnelerin tespitinde zayıflıklar, sınırlayıcı kutu konumlandırmasında hassasiyet eksiklikleri ve karmaşık sahnelerde performans düşüşleri gibi dezavantajları da mevcuttu. Buna rağmen, YOLOv1'in mAP değeri dönemi için tatmin edici bulunmuş ve sonraki YOLO sürümleri için temel teşkil etmiştir.

YOLOv2 (2017), YOLOv1'in eksiklerini gidermek amacıyla önemli teknik geliştirmelerle

ortaya konulmuştur. "Anchor box" mekanizmasının entegrasyonuyla nesne kutularının daha doğru tanımlanması sağlanmış, "Batch Normalization" ile eğitim kararlılığı artırılmıştır. Model, daha yüksek çözünürlükteki girişlerle daha iyi sonuçlar elde ederken, "Darknet-19" mimarisiyle ağ daha verimli hale getirilmiştir. Bu iyileştirmeler sayesinde özellikle küçük nesnelerin tespitinde belirgin bir ilerleme kaydedilmiş, veri artırma teknikleriyle genel başarım yükselmiş ve sınıf tahminlerindeki doğruluk artırılmıştır. YOLOv2, hız ve doğruluk arasında başarılı bir denge sunarak pratik uygulamalar için daha kullanılabilir bir model haline gelmiştir.

YOLOv3 (2018), daha derin ve güçlü bir mimari olan "Darknet-53" ile donatılmıştır. Bu mimari, artık bağlantılar (residual connections) kullanarak eğitimin kolaylaştırılmasını sağlamıştır. Modelin en önemli yeniliklerinden biri, üç farklı ölçekte tahmin yaparak özellikle küçük nesne tespitindeki yeteneğini ciddi anlamda geliştirmesidir. Sınıflandırmada Sigmoid aktivasyon fonksiyonunun kullanılması ve Softmax yerine bağımsız sınıf skorlarının verilmesi, çoklu etiketli nesnelerin tespiti için avantaj sunmuştur. YOLOv3, COCO veri kümesinde yüksek başarı göstererek hem doğruluk hem de hız açısından dengeli bir performans sergilemiş, büyük ve karmaşık veri setlerine uygulanabilirliği artırılmıştır.

YOLOv4 (2020), hız ve doğruluğu daha da optimize eden kapsamlı bir güncelleme olarak öne çıkmıştır. Model, daha verimli bir yapı için CSPDarknet-53 omurgasını kullanmış, Spatial Pyramid Pooling (SPP) ve PANet ile çoklu ölçekten bilgi toplama kapasitesini artırmıştır. Eğitim sürecinde Mosaic veri artırma ve DropBlock düzenleme gibi teknikler ile Mish aktivasyon fonksiyonu ve CIoU kayıp fonksiyonu gibi yenilikler entegre edilmiştir. Bu sürüm, sınırlı donanımda dahi yüksek performans hedeflemiş ve açık kaynak olarak sunularak toplulukta geniş kabul görmüştür. YOLOv4, COCO test setinde yüksek mAP değerleri elde ederek eğitim stabilitesini artırmış ve endüstriyel standartları belirlemede önemli bir rol oynamıştır.

YOLOv5 (2020) Ultralytics tarafından PyTorch tabanlı olarak geliştirilen YOLOv5, nesne tespiti alanındaki en yaygın kullanılan sürümlerden biridir. Farklı boyutlarda (s, m, l, x) dört temel versiyon sunarak kullanıcıya esneklik sağlamış, eğitim, test ve dağıtım süreçlerini oldukça kullanıcı dostu hale getirmiştir. CSPNet mimarisi ve PANet yapısıyla optimize edilmiş parametreler sunarken, veri artırma teknikleri, auto-anchor ve Mosaic/MixUp gibi yaklaşımlar genellenebilirliği artırmıştır. Modelin küçük boyutlarda dahi yüksek performans

göstermesi ve düşük donanımlarda çalışabilmesi, gerçek zamanlı uygulamalar için ideal bir tercih olmasını sağlamıştır. Sürekli güncellenmesi ve temiz kod yapısıyla, YOLOv5 günümüzde de en çok tercih edilen modeller arasında yer almaktadır.

YOLOv6 (2022), özellikle endüstriyel uygulamaların ihtiyaçlarına yönelik olarak Meituan tarafından geliştirilmiştir. EfficientRep ve Rep-PAN bloklarıyla ağ yapısı yeniden tasarlanmış, anchor-free ve anchor-based versiyonlarla esneklik sağlanmıştır. SIoU kayıp fonksiyonunun kullanımı, daha dengeli kutu yerleşimlerine olanak tanımıştır. Model, endüstriyel sahnelerde daha doğru ve hızlı tespit yapabilme yeteneğiyle öne çıkmış, gerçek zamanlı takip sistemleriyle uyumlu hale getirilmiştir. Düşük güç tüketimi için optimize edilmiş yapısı, üretim hatları ve güvenlik sistemleri gibi alanlarda ve gömülü sistemlerde yaygın olarak tercih edilmesini sağlamıştır.

YOLOv7 (2022), YOLO ailesi içerisinde performans ve çıkarım hızı dengesini en iyi sağlayan modellerden biri olarak öne çıkmaktadır. E-ELAN mimarisi sayesinde bilgi akışı artırılmış, model yapısı daha kompakt ancak daha güçlü hale getirilmiştir. "Bag-of-Freebies" teknikleri ile eğitimde doğruluk artırılmış ve inferans sırasında optimize edilmiş modüller sayesinde yüksek FPS değerlerine ulaşılmıştır. Çeşitli boyutlarda versiyonları ve çok sayıda benchmark testindeki liderliği, onu özellikle drone görüntüleri ve video akışları gibi zorlu senaryolarda başarılı kılmıştır. Edge cihazlar için uygunluğu ve eğitimdeki hız avantajı ile YOLOv7, nesne tespitinde "state-of-the-art" performansa yaklaşarak uygulamalı yapay zekâ projelerinde sıkça kullanılmaktadır.

YOLOv8 (2023), mimarisinde anchor-free bir yapıya geçiş yaparak önemli bir sadeleşme sağlamıştır. "Decoupled head" yapısı sayesinde sınıflandırma ve lokalizasyon süreçleri ayrı ayrı optimize edilmiş ve (PyTorch framework)'ü üzerinden tamamen yeniden yazılmıştır. Modelin ölçeklenebilirliği artırılmış, segmentasyon, poz tahmini ve sınıf tespiti gibi farklı görevler aynı çatı altında toplanmıştır. "Dynamic NMS" ile yanlış pozitifler azaltılırken, mAP değerlerinde YOLOv5'e göre belirgin iyileşmeler kaydedilmiştir. Edge ve mobil cihazlarda verimli çalışabilmesi, yüksek FPS değerleriyle gerçek zamanlı uygulamalara uygunluğu ve geliştirici dostu kod yapısıyla YOLOv8 hem araştırma hem de endüstride yaygın olarak benimsenmiştir.

YOLOv9 (2023), YOLO mimarisine ilk kez Transformer tabanlı dikkat mekanizmalarını

entegre etmesiyle öne çıkmaktadır. Bu entegrasyon, görüntü anlayışını daha bağlamsal hale getirmiş ve "Unified head" mimarisi ile çok görevli tespit yeteneği sağlanmıştır. Özellikle kalabalık sahnelerde üstün başarı gösteren model, hem anchor-free hem de anchor-based yapılarla uyumludur. Görüntü üzerinde hem sınıf hem de poz tahmini aynı yapıda gerçekleştirilirken, eğitim sürecinin esnekliği ve gelişmiş augmentasyon teknikleri ile performans artırılmıştır. Hafif sürümleri mobil cihazlarda sorunsuz çalışabilen YOLOv9, yüksek doğruluk ve hız dengesini koruyarak video ve canlı akışlarda stabil performans sağlamıştır.

YOLOv10 (2024), hibrit dikkat mekanizması (CNN + Transformer) ile geliştirilmiş, sahneye göre dinamik çözünürlük ayarlama özelliğini entegre etmiştir. "Dual-head" yapısı sayesinde çok görevli eşleştirme yapılabilirken, eğitim süreci daha stabil ve verimli hale getirilmiştir. Model, edge cihazlara özel olarak optimize edilmiş, CIoU loss gibi yeni kayıp fonksiyonları ve gelişmiş örnek eşleme algoritmaları ile isabet oranı artırılmıştır. Yüksek kalitede tespitlerde daha düşük yanlış pozitif üretimi ve büyük veri kümelerinde hızlı yakınsama sağlayan YOLOv10, özellikle üretim ve savunma sektöründe kullanıma uygun olarak açık kaynaklı sunulmuştur.

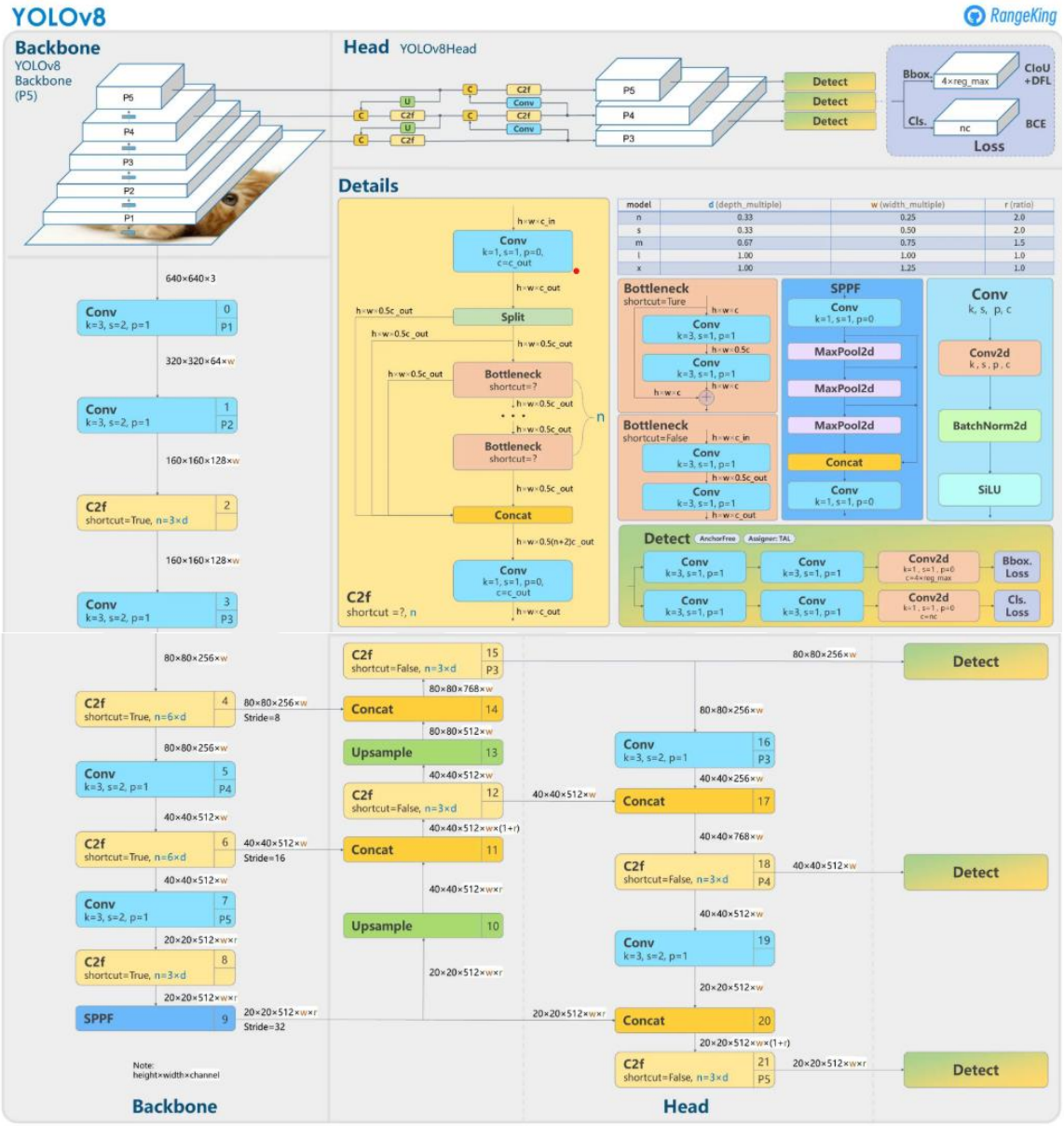
YOLOv11 (2025), çok ölçekli dikkat (multi-scale attention) mekanizmalarını içeren en yeni nesil YOLO modelidir. Yarı denetimli öğrenme yapıları sayesinde az etiketli verilerle dahi etkili çalışabilme yeteneğine sahiptir. "Dynamic model scaling" ile donanıma göre otomatik yapılandırılabilirken, yeni nesil CIoU kaybı ile daha keskin sınırlamalar yapılmaktadır. Görsel zekâ alanında detaylı sahne analizi imkânı sunan bu model, düşük ışık ve zorlu koşullarda da başarılı sonuçlar vermektedir. Segmentasyon, sınıflandırma ve poz tahmini gibi görevleri entegre bir şekilde yapabilmesi, büyük modellerde daha az parametre ile yüksek doğruluk sağlaması, onu sağlık, savunma ve otomotiv gibi alanlarda ileri düzey bir çözüm haline getirmektedir. YOLOv11, hızdan ödün vermeden doğruluğu artırarak otonom sistemlerle kolayca entegre olabilir ve derin öğrenmede yeni bir tespit standardı getirmektedir.

3.7 YOLOv8, YOLOv10 ve YOLOv11 Modeli

Bu çalışmada, nesne tespitinde yaygın olarak kullanılan ve farklı mimari özellikler sunan YOLOv8, YOLOv10 ve YOLOv11 modelleri temel alınmıştır.

3.7.1 YOLOv8 Modeli

YOLOv8, You Only Look Once algoritma serisinin en güncel versiyonu olup, gelişmiş model bileşenleri ve yeni işleme teknikleri entegre edilerek gerçek zamanlı nesne tespitinde önemli ilerlemeler sağlamıştır. Bu model, nesne tespiti, örnek segmentasyonu, sınıflandırma, yönlendirilmiş tespit ve poz tespiti gibi çoklu görüntü işleme görevlerini desteklemektedir. YOLOv8'in temel mimarisi modüler bir yapıya sahiptir ve çapasız tespit metodolojisi, ayrıştırılmış yerelleştirme ve sınıflandırma başlıkları, uyarlanabilir çapasız yaklaşım ve geliştirilmiş maksimum olmayan bastırma (NMS) tekniklerini içermektedir. Bu teknik yenilikler, farklı görüntü boyutlarında doğruluğu artırmakta, esnek ölçekleme imkânı sunmakta ve gerçek zamanlı uygulamalarda yüksek adaptabilite sağlamaktadır. Modelin uyarlanabilir tasarımı, kullanıcıların donanım kısıtlamaları ve uygulama gereksinimleri doğrultusunda model derinliği ve genişliğini optimize etmelerine olanak tanımaktadır. YOLOv8'de yerelleştirme ve sınıflandırma işlemlerinin bağımsız yönetimi hem doğruluğu hem de işlem hızını artıran gelişmiş özellik çıkarma teknikleri kullanılmaktadır. Uyarlanabilir çapasız algılama ve geliştirilmiş NMS tekniği ile yanlış pozitif oranları minimize edilmekte ve çıktı tahminlerinin netliği artırılmaktadır. Model mimarisi, hesaplama açısından kısıtlı ortamlarda bile yüksek doğruluğu koruyarak çeşitli görüntü boyutlarını desteklemekte ve ölçeklenebilirlik özelliği sunmaktadır. YOLOv8'in beş farklı varyantı (n, s, m, l, x) model boyutu, doğruluk ve hesaplama verimliliği arasında farklı dengeler sunarak çeşitli uygulama ihtiyaçlarına cevap vermektedir (Miraç, 2024). Şekil 3.16’da YOLOv8’in mimarisi yer almaktadır (Sohan, 2024).



Şekil 3.14 : YOLOv8 mimarisi

YOLOv8 mimarisi, birbirini takip eden üç temel yapısal bileşenden oluşmaktadır: Backbone (Omurga), Neck (Boyun) ve Head (Baş) modülleri.

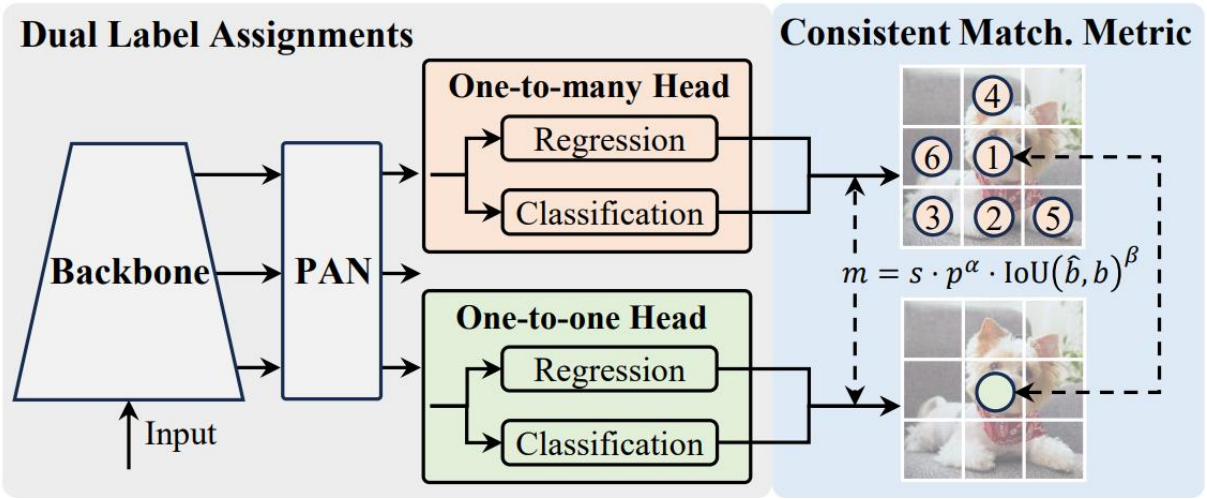
Omurga (Backbone) bölümü, görüntüye bakıp temel özellikleri bulan kısımdır. Bu bölüm görüntüdeki çizgileri, renkleri, şekilleri ve nesnelerin parçalarını tanır. Basit özelliklerden başlayarak giderek daha karmaşık özellikleri öğrenir.

Boyun (Neck) bölümü, omurga bölümünden gelen farklı bilgileri birleştiren ara katmandır. Görüntüdeki hem küçük hem de büyük nesneleri fark edebilmek için farklı boyutlardaki bilgileri bir araya getirir.

Baş (Head) bölümü, son kararı veren kısımdır. Bu bölüm iki önemli işi yapar: nesnelerin ne olduğunu söyler (araba, insan, köpek gibi) ve bu nesnelerin görüntüde nerede olduğunu belirten dikdörtgen kutular çizer. Bu üç parça sırayla çalışarak YOLOv8'in görüntülerdeki nesneleri bulmasını sağlar.

3.7.2 YOLOv10 Modeli

YOLOv10 modeli, Wang ve arkadaşları (2024) tarafından geliştirilmiş ve önceki YOLO sürümlerinin getirdiği yenilikler üzerine kurulmuştur. Bu modelin en önemli özelliği, yapay sinir ağı içinde hem Transformer tabanlı dikkat mekanizması hem de evrişimsel (convolutional) dikkat mekanizması kullanmasıdır. Şekil 3.14'te gösterildiği üzere, bu iki dikkat türünün bir arada kullanılması, özellikle görüntüde çok sayıda nesne veya karmaşa olduğunda, modelin detayları daha doğru algılamasına olanak tanımaktadır. YOLOv9 yalnızca Transformer dikkat yapısını kullanırken, YOLOv10'da daha erken katmanlarda evrişimsel dikkat kullanılarak yerel (küçük alanlardaki) özelliklerin daha iyi anlaşılması sağlanmıştır. İlerleyen katmanlarda ise Transformer katmanları devreye girerek genel (tüm görüntü düzeyinde) bağlamı daha iyi analiz etmektedir. Bu yapı sayesinde YOLOv10 hem görüntüdeki küçük ayrıntıları hem de genel yapıyı dikkate alarak daha başarılı nesne tespiti yapabilmektedir (Wang vd., 2024).



Şekil 3.14: YOLO-v10'un mimarisi (Wang vd., 2024).

YOLOv10 modeli, önceki sürümlere göre daha gelişmiş bir nesne tespiti performansı sunmak amacıyla geliştirilmiştir. Bu model, evrişimsel dikkat (convolutional attention) ve

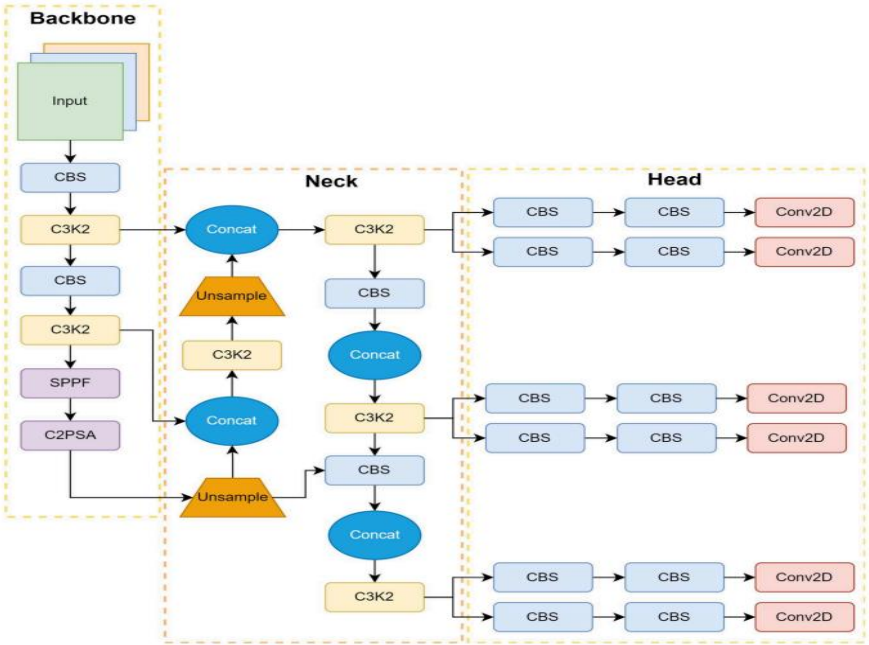
Transformer tabanlı dikkat mekanizmalarını bir araya getirerek hem küçük detayları (örneğin ince kenarlar veya doku farkları) hem de görüntünün genel yapısını aynı anda değerlendirebilmektedir. Bu yapı, özellikle üst üste binen kusurların tespiti gibi zor görevlerde oldukça başarılı sonuçlar vermektedir. YOLOv10 mimarisi, "çift etiketleme stratejisi" olarak adlandırılan bir yöntem kullanır. Bu strateji; biri çoklu eşleşme (one-to-many), diğeri tekli eşleşme (one-to-one) yapan iki ayrı başlıktan oluşur. Çoklu eşleşme başlığı, modelin duyarlılığını artırmak için birden fazla eşleşme üzerinden sınıflandırma ve konum tahmini yapar. Tekli eşleşme başlığı ise, daha kesin sonuçlar elde etmek için her hedefle yalnızca bir eşleşme üzerinden çalışır. Bu yaklaşım, aynı nesnenin birden fazla kez algılanmasını önler. Modelin işleyişi, giriş verisini alan bir modül ve ardından çok katmanlı özellik çıkarımı yapan bir omurga (backbone) ile başlar. Daha sonra, Path Aggregation Network (PAN) adı verilen yapı, bu özellikleri işleyerek çok ölçekli bilgileri iyileştirir. Son aşamada ise "Consistent Match Metric" adlı özel bir ölçüm yöntemi ile eşleştirmeler daha kararlı hale getirilir. Bu da hem sınıflandırma hem de takip (tracking) süreçlerinde daha tutarlı sonuçlar elde edilmesini sağlar. YOLOv10 ayrıca, düşük donanımlı sistemlerde kullanılmak üzere özel olarak tasarlanmıştır. Bu nedenle, modelde yer alan hafif (lightweight) modüller sayesinde, kenar cihazlar (edge devices) gibi düşük güç tüketen donanımlarda da yüksek hız ve doğrulukla çalışabilir. Uyarlanabilir çözünürlük ölçekleme (adaptive resolution scaling) özelliği ise sahnedeki nesne yoğunluğuna göre çözünürlüğü otomatik olarak ayarlayarak hem hızdan ödün vermeden çalışmayı hem de yüksek tespit başarımını garanti eder. YOLOv10 modelinde geliştirilmiş bir izleme (tracking) modülü de bulunmaktadır. Bu modül, özellikle otonom robotlar, akıllı güvenlik sistemleri ve endüstriyel otomasyon gibi alanlarda kullanılmak üzere geliştirilmiştir ve nesne takibi görevlerinde yüksek performans sağlamaktadır (Mao ve Hong, 2025).

3.7.3 YOLOv11 Modeli

YOLOv11 modeli, Ultralytics ekibi tarafından geliştirilmiş olup, nesne tespiti alanında önemli yenilikler sunan gelişmiş bir mimaridir (Jocher ve Qiu, 2024). Bu sürüm, YOLOv10'un çift dikkat (dual-attention) sistemini temel alarak, çok ölçekli dikkat (multi-scale attention) mimarisini entegre eder. Böylece, ağın farklı katmanlarında odak noktası dinamik olarak uyarlanarak hem küçük hem de büyük nesnelerin daha hassas bir şekilde tespit edilmesi sağlanır. Bu yetenek, özellikle mikro yırtıklar gibi küçük hataların yanı sıra kumaş dokusundaki büyük yapısal bozulmaların algılanmasında modeli son derece etkili

kılmaktadır. YOLOv11, uzamsal dikkat mekanizması sayesinde farklı çözünürlüklerdeki özellik haritalarını (feature maps) daha verimli hâle getirerek değişken ışık koşullarında bile yüksek tespit doğruluğu sağlamaktadır (Mao ve Hong, 2025).

Model ayrıca dinamik model ölçeklendirme özelliğine sahiptir; bu sayede mimari, mevcut donanım kaynaklarına göre derinlik ve genişliğini otomatik olarak ayarlayabilir. Böylece hem yüksek performanslı GPU'larda hem de sınırlı kaynaklara sahip uç cihazlarda verimli çalışabilir. YOLOv11'in bir diğer önemli katkısı, yarı denetimli öğrenme (semi-supervised learning) stratejisidir. Bu yöntem sayesinde hem etiketli hem de etiketsiz verilerden öğrenme gerçekleştirilir ve böylece manuel etiketleme ihtiyacı azaltılır. Ayrıca, modelin bağlama duyarlı ve kendi kendine denetimli ince ayar mekanizması (context-aware, self-supervised fine-tuning) sayesinde gerçek dünya koşullarına kolayca uyum sağlaması mümkün olur. Modelin yapısal bileşenleri arasında yer alan optimize edilmiş hafif bir tespit başlığı (lightweight detection head) ile birlikte kullanılan uyarlanabilir non-maximum suppression (NMS) mekanizması hem işlem verimliliğini artırır hem de yanlış pozitif oranlarını düşürür. Şekil 3.15'te sunulan mimaride görüldüğü üzere, YOLOv11'in omurga (backbone), boyun (neck) ve başlık (head) bileşenleri; verilerin etkili şekilde çıkarımı, uzamsal olarak iyileştirilmesi ve doğru tespit sonuçlarının elde edilmesini sağlamaktadır (Mao ve Hong, 2025).



Şekil 3.15: YOLOv11'in mimarisi (Mao ve Hong, 2025).

YOLO modelleri, farklı gereksinimlere yönelik olarak optimize edilmiş çeşitli alt modellerden oluşmaktadır. Bu modeller, nesne tespit doğruluğunu artırmak adına daha karmaşık yapılar ve derin ağlar kullanmaktadır. Ancak bununla birlikte hesaplama kaynakları üzerindeki yük de artmaktadır. Özellikle gerçek zamanlı çoklu nesne tespiti hedeflendiğinde, hız ve doğruluk arasında optimum dengeyi kurmak kritik hâle gelmektedir. COCO veri kümesi üzerinde yapılan deneyler, YOLO modellerinin bu dengeyi başarılı bir şekilde sağladığını ortaya koymaktadır. Yüksek hız ve doğruluk oranları sayesinde YOLO mimarisi, gerçek dünya uygulamalarında yüksek hassasiyet ve bağlamsal farkındalık gerektiren senaryolarda etkin bir şekilde kullanılabilmektedir. Bu doğrultuda geliştirilen YOLOv serisi, "n, s, m, b, l ve x" varyantlarıyla farklı büyüklükte ve kapasitede modeller sunarak kullanıcıya seçim esnekliği sağlar. Her bir model varyantı, işlem gücü, parametre sayısı ve doğruluk oranı gibi kriterler açısından farklılık göstermektedir (Bahadır, 2015).

3.8 Model Kıyaslaması

Tablo 3.1 ve Tablo 3.2 bu varyantlar arasında yapılan karşılaştırmayı sunmakta olup, şu başlıklar altında değerlendirmeler yapılmıştır:

- **Model:** Kullanılan YOLO model varyantı
- **Boyut:** Giriş görüntüsünün çözünürlüğü
- **mAPval 0.5:0.95:** %50 ile %95 arasındaki IoU (Intersection over Union) değerleri üzerinden hesaplanan ortalama hassasiyet (mean Average Precision).
- **mAPval 0.5:** IoU eşiği %50 kabul edilerek elde edilen ortalama hassasiyet değeri.
- **Hız CPU b1:** CPU üzerinde, yığın (batch) boyutu 1 ile ölçülen ortalama çıkarım süresi.
- **Hız V100 b1:** NVIDIA V100 GPU üzerinde, yığın boyutu 1 ile elde edilen çıkarım süresi.
- **Hız V100 b32:** NVIDIA V100 GPU üzerinde, yığın boyutu 32 ile hesaplanan çıkarım süresi.
- **Parametre Sayısı:** Modelin içerdiği toplam öğrenilebilir parametre sayısı.

- **FLOPs:** Saniyede gerçekleştirilen kayan noktalı işlem miktarı, modelin hesaplama karmaşıklığını temsil eder.

Aşağıda, YOLO-v10 ve YOLO-v11 modellerine ait varyantların karşılaştırmalı başarı ölçütleri sunulmuştur:

Tablo 3.2: YOLO-v8 modellerinin MS COCO veri seti üzerindeki performans karşılaştırması (Mao ve Hong, 2025).

Model	**Giriş Boyutu (Piksel)**	**mAP50-95 (%)**	**CPU Hızı (ONNX, ms)**	**GPU Hızı (A100 TensorRT, ms)**	**Parametre Sayısı (M)**	**FLOPs (B)**
YOLOv8-n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8-s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8-m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8-l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8-x	640	53.9	479.1	3.53	68.2	257.8

Tablo 3.2'de**,** YOLOv8 modelinin farklı varyantlarının (n, s, m, l, x) performans karşılaştırması verilmiştir. Deney sonuçları, model boyutu arttıkça doğruluk oranının (mAP50-95) yükseldiğini ancak işlem süresinin de buna paralel olarak arttığını göstermektedir. YOLOv8-n modeli, yalnızca 3.2 milyon parametre ile %37.3 mAP elde ederken**,** CPU üzerinde 80.4 ms **ve** GPU üzerinde 0.99 ms gecikme süresiyle en hafif ve en hızlı model olmuştur. Buna karşılık, en büyük model olan YOLOv8-x**,** %53.9 mAP değeriyle en yüksek doğruluğu sağlamış; ancak 479.1 ms CPU hızı ve 3.53 ms GPU hızı ile daha yüksek işlem maliyeti ortaya koymuştur. (Miraç, 2024).

Tablo 3.3: YOLO-v10 modellerinin MS COCO veri seti üzerindeki performans karşılaştırması (Mao ve Hong, 2025).

Model	Girdi Boyutu (piksel)	mAP50-95 (%)	CPU Hızı (ONNX, ms)	GPU Hızı (TensorRT10, ms)	Parametre Sayısı (M)	FLOPs (B)
YOLO-v10-n	640	39.5	-	1.56	2.3	6.7
YOLO-v10-s	640	46.7	-	2.66	7.2	21.6
YOLO-v10-m	640	51.3	-	5.48	15.4	59.1
YOLO-v10-b	640	52.7	-	6.54	24.4	92.0
YOLO-v10-l	640	53.3	-	8.33	29.5	120.3
YOLO-v10-x	640	54.4	-	12.2	56.9	160.4

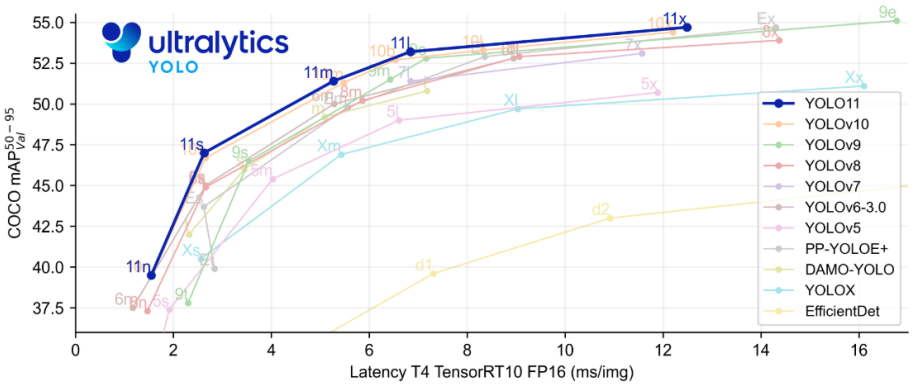
Tablo 3.3'de, COCO veri seti üzerinde yapılan deneyler, YOLO-v10 mimarisinin, gerçek zamanlı çoklu nesne tespitinde yüksek hız ve doğruluğu koruyarak önceki modellere kıyasla önemli gelişmeler sunduğunu göstermektedir. Bu model, yüksek hassasiyet ve bağlamsal farkındalık gerektiren uygulamalarda etkili sonuçlar üretmektedir. Yapılan karşılaştırmalarda, YOLO-v10'un altı farklı varyantı (n, s, m, b, l, x) test edilmiş; örneğin YOLO-v10-m modeli, YOLO-v10-s'ye göre yaklaşık 1,8 kat daha hızlı çalışırken benzer ortalama doğruluk (mAP) değerlerini korumuştur. Benzer şekilde, YOLO-v10-b modeli, YOLO-v10-l'ye göre daha düşük gecikme süresi ve daha az parametreyle benzer performans sunmuştur (Mao ve Hong, 2025).

Tablo 3.4: YOLO-v11 modellerinin MS COCO veri seti üzerindeki performans karşılaştırması (Mao ve Hong, 2025).

Model	Girdi Boyutu (piksel)	mAP50–95 (%)	Hız (CPU, ONNX, ms)	Hız (T4, TensorRT10, ms)	Parametre Sayısı (M)	FLOPs (B)
YOLO-v11-n	640	39.5	56.1 ± 0.8	1.5 ± 0.0	2.6	6.5
YOLO-v11-s	640	47.0	90.0 ± 1.2	2.5 ± 0.0	9.4	21.5
YOLO-v11-m	640	51.5	183.2 ± 2.0	4.7 ± 0.1	20.1	68.0
YOLO-v11-l	640	53.4	238.6 ± 1.4	6.2 ± 0.1	25.3	86.9
YOLO-v11-x	640	54.7	462.8 ± 6.7	11.3 ± 0.2	56.9	194.9

Tablo 3.4'te, YOLOv11'in beş farklı varyantını (n, s, m, l ve x) karşılaştırmalı olarak sunmaktadır. Bu çizelge, model boyutu, doğruluk ve hesaplama verimliliği arasındaki dengeyi göstermektedir. YOLOv5'ten YOLOv11'e kadar olan tüm modeller, giriş olarak 640 × 640 piksel boyutunda görüntülerle çalışmaktadır. Buna ek olarak, YOLOv7 modeli 1280 × 1280 piksel çözünürlüğü de desteklemektedir. Modellerin başarımı; farklı eşiklerdeki ortalama doğruluk (mAP 50 ve mAP 50-95), CPU ONNX çıkarım süresi (ms), A100 GPU üzerinde TensorRT hızları (ms), parametre sayısı (M), saniyede yapılan kayan noktalı işlem sayısı (FLOPs) ve FPS gibi temel ölçütlere göre değişiklik göstermektedir. Bu başarımlara ilişkin veriler, COCO veri kümesi üzerinde önceden eğitilmiş tespit modelleri kullanılarak yapılan deneyler sonucunda elde edilmiştir (Mao ve Hong, 2025).

Tez çalışmasının deneysel uygulama sürecinde, başlangıçta en güncel modellerden biri olan YOLOv8 kullanılmıştır. Bu model ile yüksek doğrulukta ve hızlı nesne tespiti yapılması amaçlanmıştır. Ancak çalışmalar devam ederken, daha gelişmiş özellikler sunan YOLOv10 ve YOLOv11 sürümü yayınlanmıştır (Jocher ve Qiu, 2024). YOLOv11, özellikle doğruluk oranı, hız ve kaynak verimliliği açısından önceki sürümlere göre daha başarılı sonuçlar vermektedir. Bu nedenle deneylerde YOLOv11'in değerlendirilmiş ve YOLOv8 ve YOLOv10 ile karşılaştırmalı olarak analiz edilmiştir. Her modelin COCO veri setiyle eğitilen farklı varyantlarının performans karşılaştırmaları Şekil 3.16'te sunulmuştur.



Şekil 3.16: YOLOv11'in önceki sürümlerle karşılaştırılması (Jocher ve Qiu, 2024).

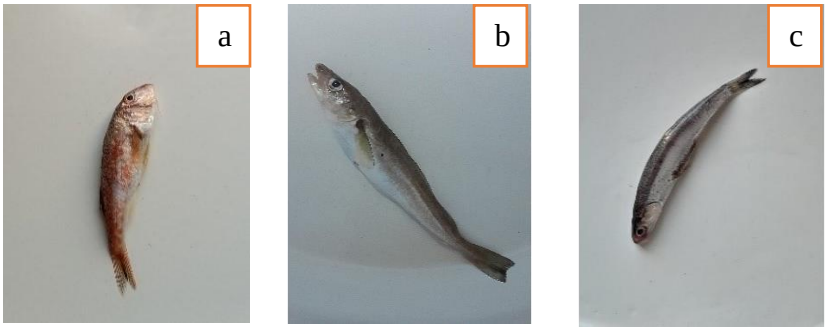
Sonuçlara göre, YOLOv11 genel olarak daha yüksek başarı sağlamış ve deneysel çalışmalarda daha iyi sonuçlar üretilmiştir (Jocher ve Qiu, 2024). Elde edilen bu sonuçlar, Bulgular ve Tartışma bölümünde ayrıntılı olarak açıklanmıştır.

3.9 Veri Toplama ve Ön İşleme Süreci

Bu tez çalışmasında kullanılan veri seti, Türkiye'nin Bartın ilçesinden toplanan ve her biri 1305 adet olmak üzere barbun, hamsi ve mezgit türlerini içeren toplam 3915 balık görüntüsünden oluşmaktadır. Barbun, hamsi ve mezgit türlerine ait örnek görüntüler sırasıyla Şekil 3.16 A, B ve C'de sunulmuştur.

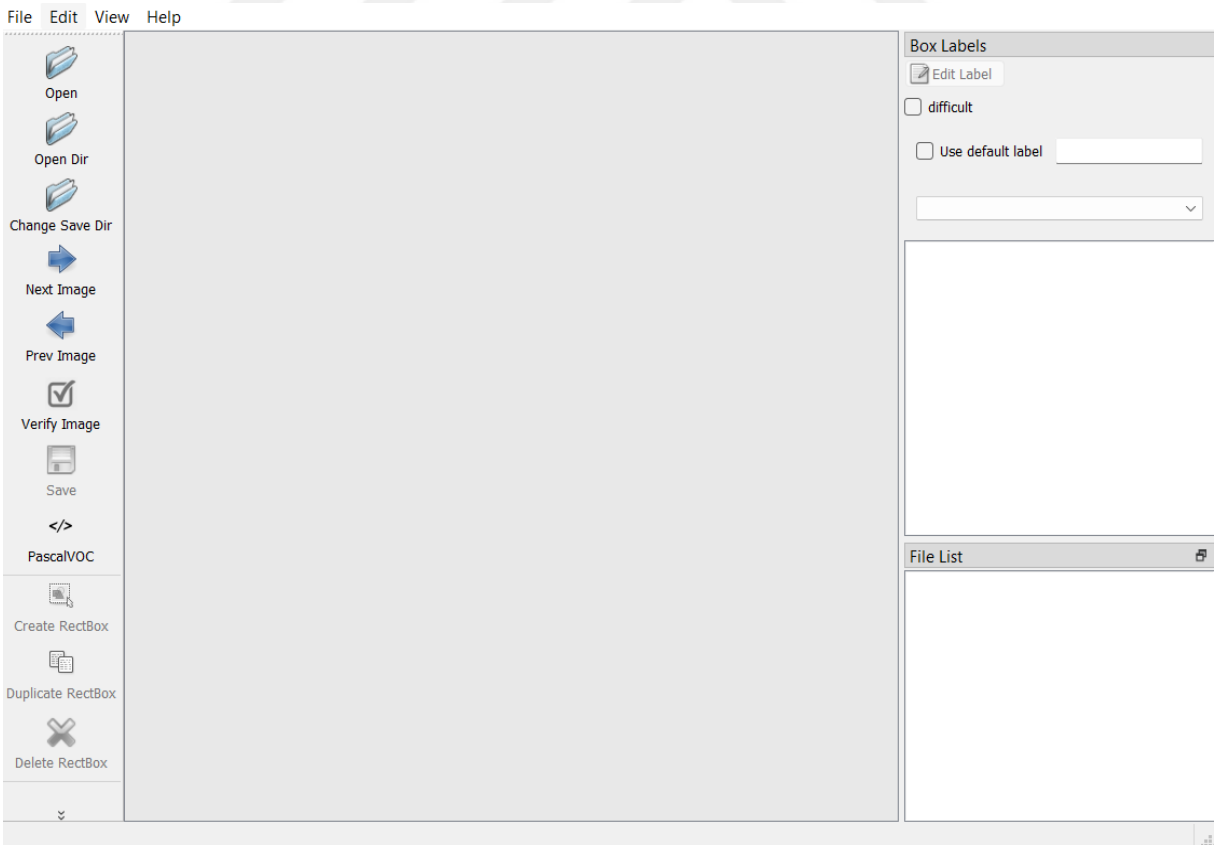
Veri seti, modelin eğitim sürecini hızlandırmak amacıyla Roboflow platformuna yüklenmiş ve tüm görüntüler 640 × 640 piksel boyutuna yeniden boyutlandırılarak ön işleme tabi tutulmuştur. Roboflow arayüzü aracılığıyla veri seti; eğitim ve test olmak üzere iki alt gruba ayrılmıştır. Bu alt gruplandırmada eğitim-veri kümesi %90, doğrulama kümesi ise %10 oranında tutulmuştur.

Elde edilen veri kümesi, YOLO modelleri için uygun olacak şekilde PyTorch formatında dışa aktarılmış ve yerel ortama indirilmiştir. Roboflow, veri toplama, etiketleme, ön işleme ve model eğitimi gibi süreçlerde etkin çözümler sunan, bilgisayarlı görü projelerine yönelik kapsamlı bir geliştirme platformudur.



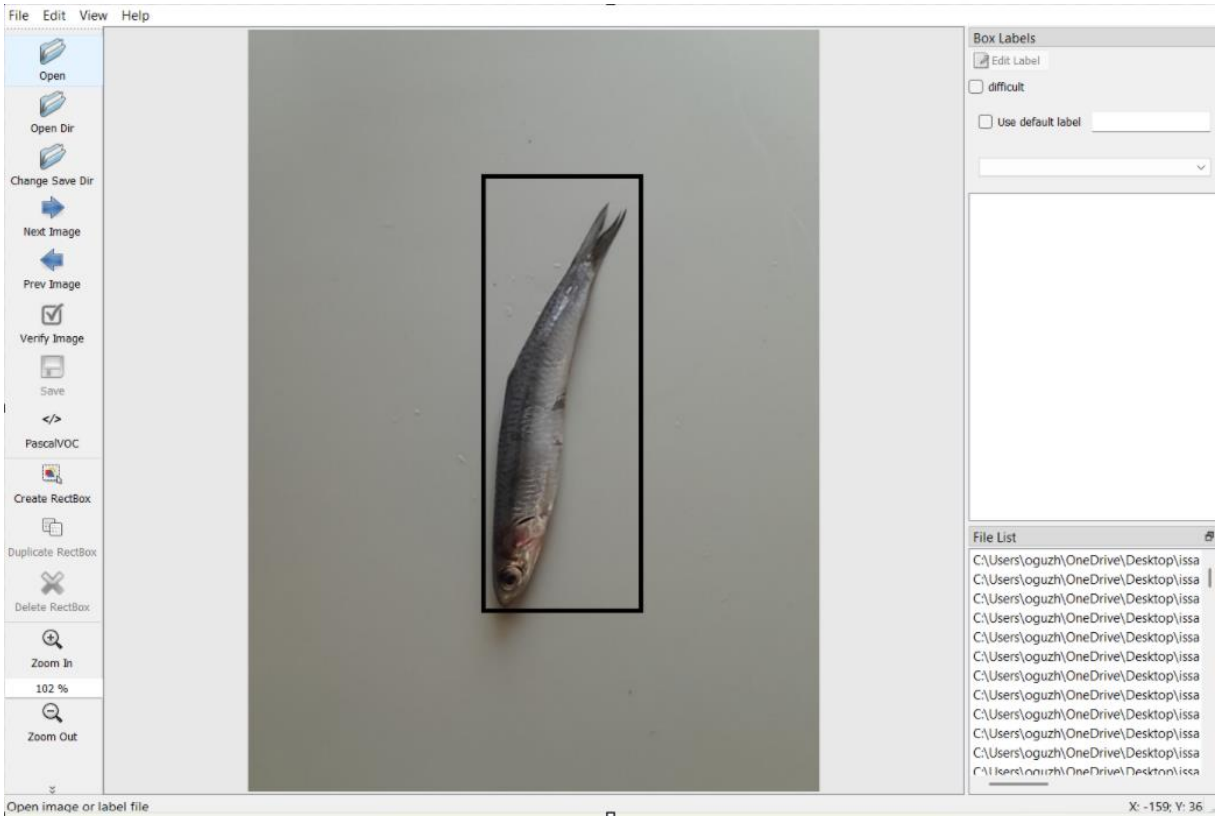
Şekil 3.16: Barbun, Hamsi ve Mezgit balıkları için örnekler. Barbun balığı (a), Hamsi balığı (b), Mezgit balığı (c)

LabelImg, Python diliyle geliştirilmiş ve Qt tabanlı bir grafiksel kullanıcı arayüzüne sahip, açık kaynak kodlu ve ücretsiz bir görüntü açıklama aracıdır. Bu yazılım, görüntüler üzerinde sınıflandırma ve etiketleme işlemlerini gerçekleştirmeye olanak tanımakta ve çıktıları YOLO algoritması ile uyumlu metin dosyası formatında kaydetmektedir. YOLO formatına uygun olarak nesne etiketleme işlemlerinin gerçekleştirildiği grafiksel kullanıcı arayüzüne ait ekran görüntüsü Şekil 3.17'de sunulmuştur.



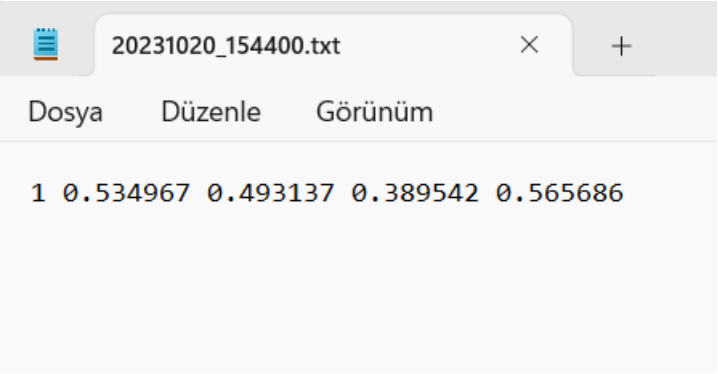
Şekil 3.17: LabelImag arayüzü

Etiketleme işlemine başlamadan önce, arayüzde yer alan “Pascal/VOC” butonuna tıklanarak etiketleme formatı YOLO olarak değiştirilir. Ardından, etiketleme yapılacak görüntü “Open” butonu aracılığıyla programa yüklenir. Görüntü üzerinde nesnenin bulunduğu alanı belirtmek amacıyla “Create RectBox” komutu kullanılarak sınırlayıcı bir dikdörtgen çizilir. Bu işlem tamamlandığında, açılan etiket penceresinden ilgili sınıf etiketi seçilir ve “OK” butonuna basılarak işlem onaylanır. Etiketleme tamamlandıktan sonra “Save” butonuna tıklanarak veri kaydedilir. Aynı adımlar, görüntüde yer alan diğer nesneler için tekrarlanarak tüm veri seti etiketlenir. Bu sürece ilişkin etiketleme arayüzü Şekil 3.18’de gösterilmiştir. Bu arayüz, YOLO formatında etiketlenmiş veri kümelerinin oluşturulmasında kullanıcıya görsel ve işlevsel kolaylık sağlamaktadır.



Şekil 3.18: LabelImg’da etiketleme işlemi

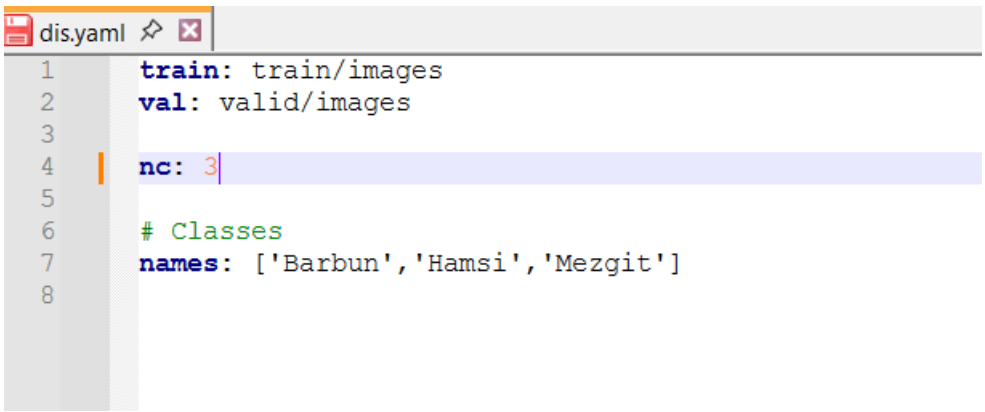
Etiketleme işlemi tamamlandığında, her görüntü için bir etiket dosyası (.txt) oluşturulur. Bu dosyada Şekil 3.19’da gibi, her satırda bir nesnenin sınıfı ve konum bilgileri yer alır. YOLO formatında bu bilgiler: sınıf_no x_merkez y_merkez genişlik yükseklik şeklindedir. Tüm değerler, görüntü boyutuna göre normalize edilmiş olur.



Şekil 3.19: Txt. dosyası

YOLO modellerinde eğitim için gerekli olan (.yaml) uzantılı dosya, veri kümesinin konumunu ve sınıf bilgilerini tanımlayan bir yapılandırma konfigürasyon dosyasıdır. Bu dosya, modelin eğitim sürecinde veri kümesine doğru şekilde erişebilmesini ve sınıfların doğru bir şekilde tanımlanmasını sağlar. YOLO formatına uygun olarak hazırlanan bu .yaml dosyasında eğitim ve test veri yolları ile sınıf sayısı ve sınıf adları belirtilir. Şekil 3.20'de örnek bir dataset.yaml dosyasının yapısı gösterilmiştir. Aşağıda bu dosyanın hazırlanma adımları sıralanmıştır:

1. Train: Eğitim veri kümesinin yolu belirtilir.
2. Val: Doğrulama veri kümesinin yolu belirtilir.
3. Nc: Veri kümesindeki toplam sınıf sayısı girilir
4. Names: Her sınıfa karşılık gelen isimlerin listesi girilir.



Şekil 3.20: Yaml dosyası

3.10 Performans Değerlendirme Metrikleri

Modelin performansı, ortalama kesinlik değeri (mean Average Precision – mAP) ölçütüyle değerlendirilmiştir. mAP, nesne tespit algoritmalarının doğruluk düzeyini belirlemek için yaygın olarak kullanılan bir ölçümdür. Bu ölçüt, karışıklık matrisi, duyarlılık, kesinlik ve F1 skoru gibi değerlendirme kriterlerine dayanarak hesaplanır. Elde edilen mAP değeri 0 ile 1 arasında bir skordur; bu değerin 1'e yaklaşması, modelin nesne tespiti görevinde daha yüksek başarı sağladığını gösterir (Miraç, 2024).

Karışıklık matrisi: bir sınıflandırma modelinin tahmin performansını değerlendirmek amacıyla kullanılan temel araçlardan biridir. Bu matris, modelin gerçek sınıf etiketleriyle yaptığı tahminlerin karşılaştırılması sonucunda elde edilen dört ana değeri içerir: Doğru Pozitif (TP), Yanlış Pozitif (FP), Doğru Negatif (TN) ve Yanlış Negatif (FN). Şekil 3.21'de iki sınıf için ürettiği karışıklık matrisleri sunulmuştur.

	Gerçek Pozitif (Pozitif Sınıf)	**Gerçek Negatif (Negatif Sınıf)**
Tahmin: Pozitif	**Doğru Pozitif (TP)** Doğru olarak pozitif tahmin edilen örnekler	**Yanlış Pozitif (FP),** Yanlış olarak pozitif tahmin edilen örnekler
Tahmin: Negatif	**Yanlış Negatif (FN)** Yanlış olarak negatif tahmin edilen örnekler	**Doğru Negatif (TN)** Doğru olarak negatif tahmin edilen örnekler

Şekil 3.21: Karışıklık matrisi

Kesinlik, modelin bulduğunu düşündüğü nesnelerin ne kadarının gerçekten doğru olduğunu ifade eder. Yani modelin verdiği pozitif tahminlerin güvenilirliği ölçülür. Kesinliği yüksek bir model, yanlış pozitiflerden kaçınır.

$$Precision = \frac{TP}{TP + FP}$$

Duyarlılık: modelin gerçek pozitif örnekleri ne kadar doğru tespit ettiğini gösteren bir ölçüttür. Nesne tespitinde, modelin var olan nesneleri ne oranda kaçırmadan algıladığını ölçer. Duyarlılığı yüksek bir model, gerçek nesneleri kaçırmadan yakalayabilir.

$$Recall = \frac{TP}{TP + FN}$$

F1 skoru, modelin performansını değerlendirmede yaygın olarak kullanılan ölçütlerden biridir ve Kesinlik ile Duyarlılık değerlerinin dengeli bir ortalaması olarak hesaplanır.

$$\mathrm{F1} = 2\left(\frac{\mathrm{Pr} \times \mathrm{Re}}{\mathrm{Pr} \times \mathrm{Re}}\right)$$

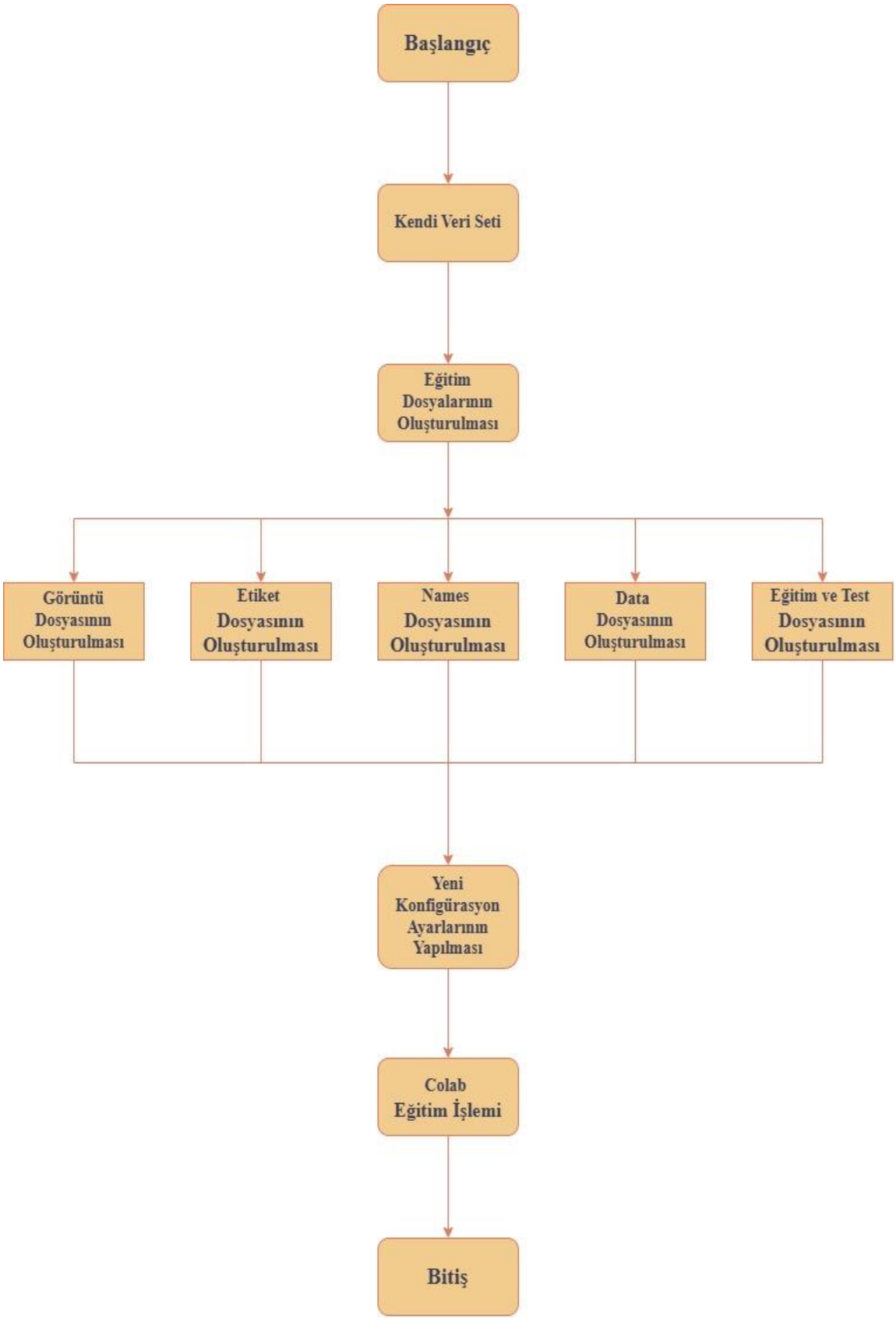
4. BULGULAR VE TARTIŞMA

Balık tespiti ve sınıflandırılması, bilgisayarla görme alanında oldukça önemli ve geniş bir konudur. Araştırmacılar, özellikle balık türlerini ayırt etmek için farklı yöntemler ve algoritmalar geliştirmiştir. Teknolojinin gelişmesiyle birlikte, bu alanda her geçen gün yeni ve daha etkili çözümler ortaya çıkmaktadır. YOLOv8, YOLOv10 ve YOLOv11 algoritmaları kullanılarak balıkların nasıl tespit edilip sınıflandırıldığı anlatılmıştır.

Tez çalışmasında amaç, görüntülerdeki balıkları otomatik olarak tanımak ve doğru tür ile eşleştirmektir. Bu çalışma özel olarak barbun, mezgit ve hamsi türleri üzerinde yapılmıştır. Bu balıklara ait görüntülerden oluşan veri seti hazırlanmış ve bu veri ile YOLOv8, YOLOv10 ve YOLOv11 modelleri eğitilmiştir.

Önceki bölümde, balık tespiti ve sınıflandırma uygulamasında izlenen adımlar sade bir şekilde açıklanmıştır. İlk olarak, veri setinin nasıl oluşturulacağı anlatılmıştır. Ardından, eğitim için gerekli dosyaların hazırlanma sürecine geçilmiş ve görüntü, etiket, names ve data dosyalarının nasıl oluşturulacağına değinilmiştir. Bu dosyaların hazırlanmasının ardından, sistemin doğru çalışması için gerekli olan konfigürasyon ayarlarının nasıl yapılacağı gösterilmiştir.

Bu bölümde, hazırlanan verilerle birlikte Colab ortamında model eğitimi gerçekleştirilmiş ve tüm bu işlemlerin sonunda balıkların türlerine göre başarıyla sınıflandırılması sağlanmıştır. Bu süreçte izlenen tüm adımlar, Şekil 4.1'de gösterilen akış şeması ile görsel olarak sunulmuştur.



Şekil 4.1: Akış diyagramı

4.1 YOLOv8, YOLOv10 ve YOLOv11 Modelinin Eğitim Aşaması

Bu çalışma, model eğitim ve çıkarım işlemlerini hızlandırmak amacıyla Google Colab üzerinde yapılandırılmış bir deneysel ortamda gerçekleştirilmiştir. Eğitim ortamında Python 3.12, PyTorch 2.2.2 ve CUDA 12.1 destekli NVIDIA Tesla T4 GPU kullanılmıştır. Bu yapılandırma, özellikle derin öğrenme modellerinin daha kısa sürede eğitilebilmesi için GPU hızlandırmalı işlem kapasitesinden faydalanmayı mümkün kılmaktadır. Eğitim süreci boyunca kullanılan donanım ve yazılım bileşenlerine ilişkin detaylar aşağıdaki Tablo 4.1'de sunulmuştur:

Tablo 4.1: Eğitim ortamı

Bileşen	**Açıklama**
İşletim Sistemi	Linux tabanlı (Ubuntu 22.04)
Derin Öğrenme Çatısı	PyTorch 2.2.2 (CUDA 12.1)
Programlama Dili	Python 3.12
İşlemci (CPU)	Intel® Xeon® (2.20 GHz)
Grafik İşlem Birimi (GPU)	NVIDIA Tesla T4 (CUDA destekli)
GPU Bellek	15 GB
Sistem Belleği (RAM)	25 GB

Bu tez çalışmasında, kullanılan YOLOv8, YOLOv10 ve YOLOv11 modelleri Google Colab ortamında eğitilmiştir. Eğitim işlemlerinde Ultralytics tarafından sunulan ultralytics kütüphanesi kullanılmış olup, Python ortamı üzerine kurulmuştur. Eğitim sürecine başlanmadan önce, Colab çalışma zamanı GPU moduna geçirilmiş ve ardından ultralytics kütüphanesini yüklenir. Eğitim sırasında kullanılan parametrelerin aşağıda açıklanmıştır:

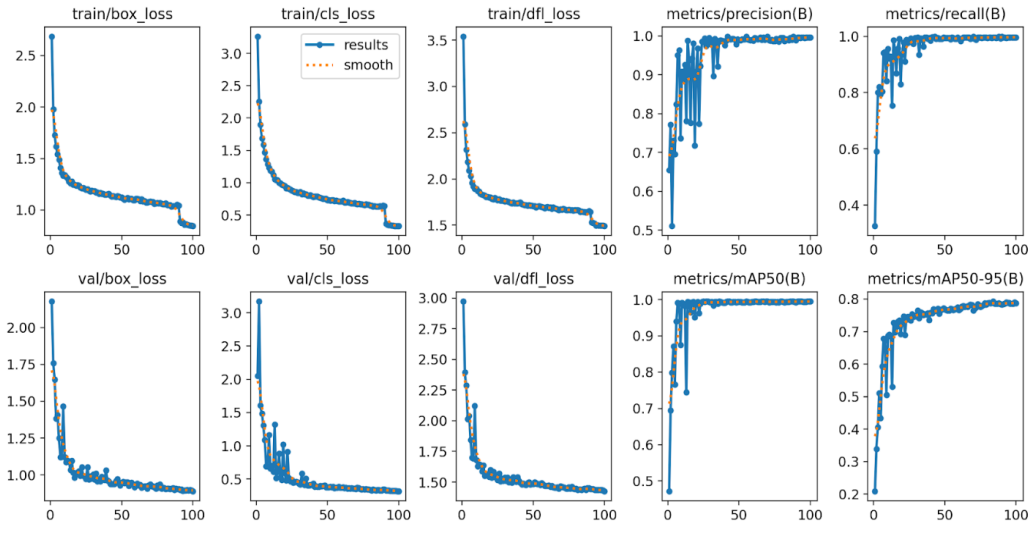
1. **data**: Veri kümesine ait data.yaml dosyasının konumunu belirtir. Bu dosyada eğitim ve doğrulama (validation) verilerinin yolları ile sınıf isimleri yer alır.
2. **epochs**: Modelin eğitim süresi boyunca tüm veri seti üzerinde kaç kez eğitim yapılacağını belirtir. Bu çalışmada 100 epoch tercih edilmiştir.

3. **batch**: Her eğitim adımında ağa verilecek görüntü sayısını belirtir. 16 batch ile eğitim gerçekleştirilmiştir.
4. **imgsz**: Modelin eğitileceği görüntü boyutu. Tüm görseller, eğitim öncesinde 640×640 boyutuna getirilmiştir.
5. **weights**: Eğitime başlanmadan önce kullanılacak olan başlangıç ağırlıklarıdır. yolov10m.pt ya da yolov11m.pt gibi önceden eğitilmiş ağırlıklar kullanılabilir.
6. **name**: Eğitim sonuçlarının kayıt altına alınacağı klasörün adını tanımlar. Bu sayede her model için sonuçlar ayrı dizinlerde saklanabilir.

4.2 Test Sonuçları ve Performans Analizi

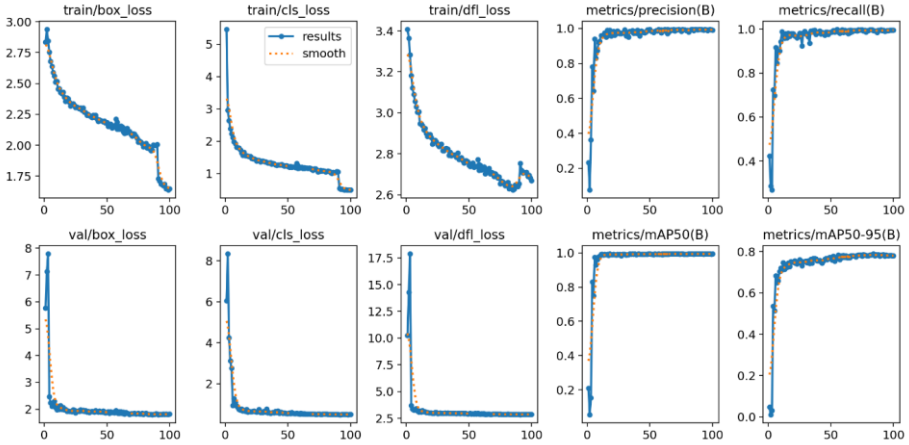
Bu tez çalışmasında, barbun, hamsi ve mezgit balık türlerinin tespiti amacıyla en güncel nesne algılama modelleri olan YOLOv8, YOLOv10 ve YOLOv11 kullanılmıştır. Bu modeller, önceki sürümlere kıyasla daha yüksek doğruluk ve daha hızlı çıkarım süresi sunmaktadır. Bu bölümde sunulan deneysel çalışma sonuçları; doğruluk (precision), duyarlılık (recall), ortalama kesinlik (mAP), F1 skoru, karışıklık matrisi, modelin başarıyla tespit ettiği performans sonuçlarını kapsamaktadır.

Modellerin performans metriklerinin grafikleri:



Şekil 4.2: YOLOv8 performans metriklerinin grafiği

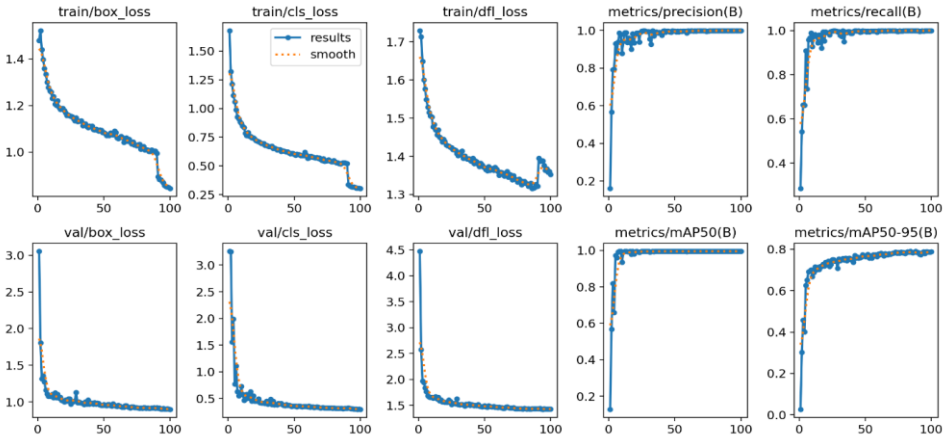
Şekil 4.2’de, YOLOv8 modeline ait eğitim ve doğrulama süreci analiz edildiğinde, kutu kaybı (box_loss), sınıflandırma kaybı (cls_loss) ve dağılım kaybı (dfl_loss) değerlerinin eğitim süresi boyunca düzenli bir azalma eğilimi gösterdiği tespit edilmiştir. Bu eğilim, modelin nesnelerin konumlarını, ait oldukları sınıfları ve dağılım parametrelerini giderek daha isabetli biçimde tahmin edebildiğini göstermektedir. Doğrulama aşamasındaki kayıp değerlerinin de benzer biçimde zamanla düşmesi, modelin yalnızca eğitim verisine uyum sağlamakla kalmayıp doğrulama verileri üzerinde de yüksek düzeyde genelleme başarısı sergilediğini ortaya koymaktadır.



Şekil 4.3: YOLOv10 performans metriklerinin grafiği

Şekil 4.3’de, YOLOv10 modelinin eğitim ve doğrulama süreci boyunca elde ettiği performans metrikleri grafiksel olarak sunulmuştur. İlk üç grafik, eğitim aşamasında kullanılan kutu kaybı (box_loss), sınıflandırma kaybı (cls_loss) ve dağılım kaybı (dfl_loss) değerlerinin zamanla azaldığını göstermektedir. Bu düşüş, modelin öğrenme sürecinde doğru şekilde iyileştiğini ve stabil bir eğitime ulaştığını göstermektedir. Alt sırada yer alan doğrulama kayıpları da benzer şekilde zamanla azalarak modelin yalnızca eğitim verisine değil, aynı zamanda doğrulama verisine karşı da başarılı bir şekilde genelleme yaptığını göstermektedir.

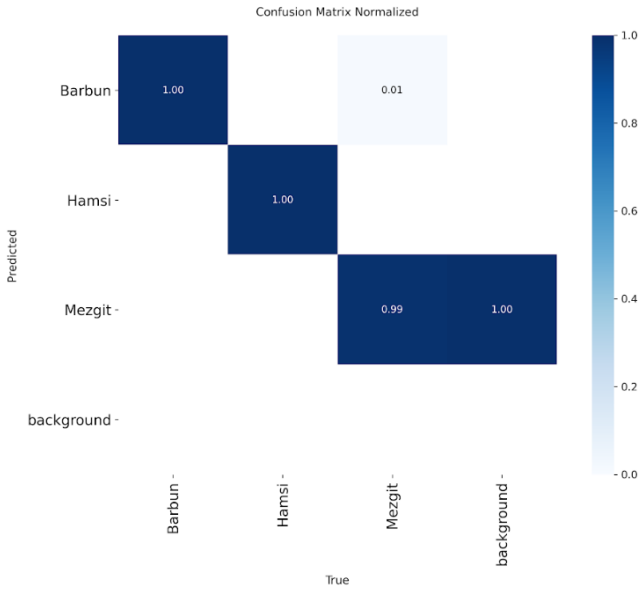
Grafiklerin sağ tarafında ise modelin doğruluk ve duyarlılık metrikleri ile mAP@50 ve mAP@50-95 değerleri gösterilmiştir. Bu metrikler zamanla artarak, modelin hem doğru tahmin yapma oranının yüksek olduğunu hem de hedef nesneleri başarıyla tespit edebildiğini göstermektedir. Özellikle mAP@50-95 metriğinin %80’e yaklaşması, modelin farklı eşik değerlerinde dahi güçlü bir performans sergilediğini ortaya koymaktadır.



Şekil 4.4: YOLOv11 performans metriklerinin grafiği

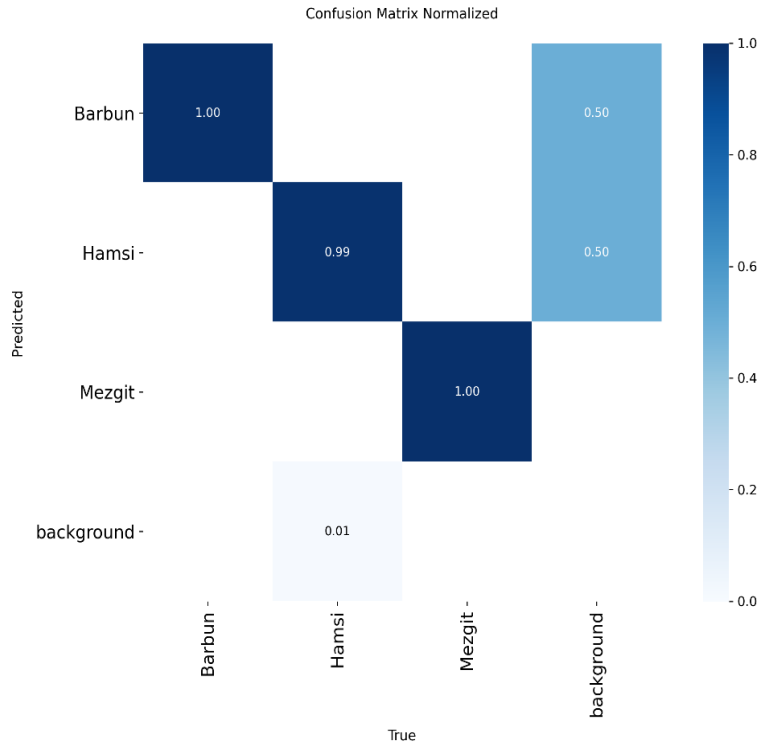
Şekil 4.4'te sunulan eğitim sonuçları, YOLOv11 modelinin hem kayıp fonksiyonlarında istikrarlı bir düşüş hem de başarım metriklerinde tutarlı bir yükseliş sergilediğini açıkça ortaya koymaktadır. Özellikle kesinlik, duyarlılık, mAP@50 ve mAP@50-95 değerlerinin eğitim sonunda 1'e yakın seviyelere ulaşması, modelin belirlenen nesne tespiti görevinde oldukça başarılı bir öğrenme süreci geçirdiğini ve yüksek genelleme yeteneğine sahip olduğunu düşündürmektedir. Bu bulgular, YOLOv11'in pratik uygulamalarda yüksek performanslı bir çözüm olarak kullanılabileceği potansiyelini desteklemektedir.

YOLOv8, YOLOv10 ve YOLOv11 modellerin karışıklık matrisi:



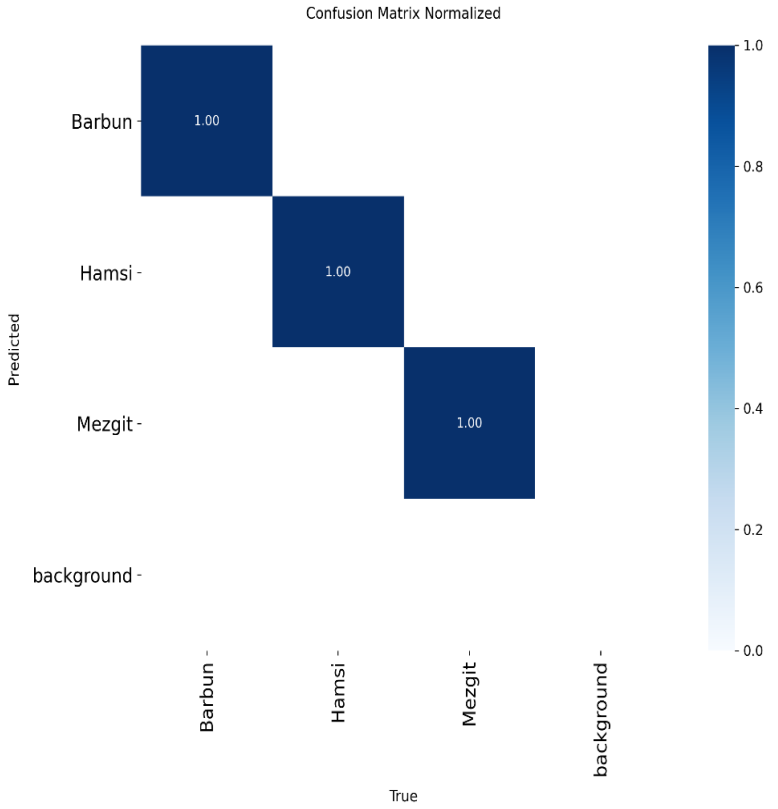
Şekil 4.5: YOLOv8 karışıklık matrisi.

Şekil 4.5'te sunulan YOLOv8 modelinde barbun ve hamsi sınıflarında yüksek doğruluk elde edilmesine rağmen, mezgit sınıfında yanlış pozitif tespitlerin görece daha fazla olduğu, özellikle hamsi-mezgit sınıfları arasında belirgin bir sınıf karışıklığı yaşandığı gözlemlenmektedir. Ayrıca background sınıfında düşük oranlı da olsa hatalı sınıflandırmalar mevcuttur. Bu durum, YOLOv8'in genel olarak başarılı bir performans sergilese de sınıflar arası ayrım kabiliyetinde iyileştirme potansiyeli bulunduğunu göstermektedir.



Şekil 4.6: YOLOv10 karışıklık matrisi.

Şekil 4.6'te gösterilen YOLOv10 modelinde ise barbun ve mezgit sınıfları %100, hamsi sınıfı %99 doğruluk oranına ulaşmıştır. Model, hem eğitim süreci boyunca stabil bir performans gelişimi göstermiş hem de nihai değerlendirme metriklerinde oldukça yüksek başarılara ulaşmıştır. Model, balık türleri arasındaki sınıflandırmada neredeyse hatasızken, bazı zorlu örneklerde nesneleri arka plandan ayırmakta küçük zorluklar yaşamıştır. Buna rağmen, genel olarak yüksek hassasiyet ve doğruluk değerleri ile balık tespiti ve sınıflandırması görevi için oldukça güçlü ve güvenilir bir çözüm sunduğu sonucuna varılmaktadır.



Şekil 4.7: YOLOv11 karışıklık matrisi.

Şekil 4.7'te sunulan YOLOv11 modelinde ise tüm balık sınıfları barbun, hamsi ve mezgit'in %100 doğruluk oranına ulaşmış, yanlış pozitif tespitler tamamen elimine edilmiş ve background hatası ortadan kaldırılmıştır. YOLOv11'in mükemmel karışıklık matrisi, sınıflar arası ayrım performansının önemli ölçüde iyileştirildiğini ve modelin balık türlerini ayırt etmedeki başarısının artırıldığını göstermektedir. Bu gelişim, YOLOv11'in endüstriyel uygulamalarda daha yüksek güvenilirlik ve doğruluk sağlayacağını işaret etmektedir.

Bu üç modele ait karışıklık matrisleri incelendikten sonra, modellerin sayısal performans farklarını daha net ortaya koymak amacıyla performans tablolarda doğrulu ve hız temel ölçütler üzerinden ayrıntılı bir performans karşılaştırması sunulmuştur.

YOLOv8, YOLOv10 ve YOLOv11 Performans Tablosu:

Tablo 4.2: YOLOv8 performans tablosu

Sınıf	Görüntü	Örnek	Kesinlik	Duyarlılık	mAP50	mAP50-95
Tümü	315	315	0.999	0.922	0.995	0.777
Barbun	105	105	0.999	0.870	0.995	0.740
Hamsi	105	105	0.999	0.973	0.995	0.782
Mezgit	105	105	0.998	0.973	0.995	0.809

Tablo 4.3: YOLOv10 performans tablosu

Sınıf	Görüntü	Örnek	Kesinlik	Duyarlılık	mAP50	mAP50-95
Tümü	315	315	0.995	0.994	0.994	0.789
Barbun	105	105	0.999	1.000	0.995	0.771
Hamsi	105	105	0.985	0.990	0.993	0.773
Mezgit	105	105	1.000	0.991	0.995	0.823

Tablo 4.4: YOLOv11 performans tablosu

Sınıf	Görüntü	Örnek	Kesinlik	Duyarlılık	mAP50	mAP50-95
Tümü	315	315	0.999	1.000	0.995	0.794
Barbun	105	105	0.999	1.000	0.995	0.784
Hamsi	105	105	1.000	1.000	0.995	0.763
Mezgit	105	105	0.999	1.000	0.995	0.823

Karşılaştırmalı analiz, YOLOv8, YOLOv10 ve YOLOv11 modellerinin balık tespiti görevindeki performans farklılıklarını göstermektedir. YOLOv8 modeli yüksek kesinliğe sahip olsa da (0.999), duyarlılık değeri (0.922) diğer modellere göre daha düşüktür. Bu durum, bazı nesneleri tespit edemediğini ortaya koymaktadır. YOLOv10, kesinlik ve

duyarlılıkta daha iyi sonuçlar elde etmektedir. Bu modeller arasında en üstün performans YOLOv11'e aittir. Modelin mAP50-95 değeri (0.794) ile en yüksek doğruluk ve konumlandırma hassasiyetine sahip olduğu, ayrıca karmaşıklık matrisinin test verisinde hiçbir hata yapmadığı görülmektedir. Bu bulgular, her üç modelin de güçlü çözümler olmasına rağmen, YOLOv11'in bu görev için en güvenilir ve en doğru seçenek olduğunu kanıtlamaktadır.

Üç modelde de mezgit sınıfının mAP50–95 değerlerinin diğer sınıflara kıyasla en yüksek olduğu görülmektedir (YOLOv8: 0.809, YOLOv10: 0.823, YOLOv11: 0.823). Bu sonuç, modelin mezgit türünü diğer sınıflardan ayırt etme becerisinin daha güçlü olduğunu göstermektedir. Mezgit sınıfına ait görüntülerin aydınlatma koşulları, pozlama açıları ve arka plan çeşitliliğinin diğer türlere göre daha tutarlı olması, modelin eğitim sürecinde daha kararlı örüntüler öğrenmesine katkı sağlamıştır. Böylece model, bu türün sınırlarını daha doğru tespit etmiş ve konum hassasiyetinde yüksek performans elde etmiştir. Bu faktörler, mezgit sınıfında gözlemlenen yüksek doğruluk ve mAP50–95 değerlerinin temel nedenleri olarak değerlendirilmektedir.

Modellerin sınıflandırma sonuçlarına ilişkin Şekillerde sunulmuştur. Bu şekil, yapılan tahminleri ve her kategori için hesaplanan güven skorlarını göstermektedir.

Şekil 4.8: Yolov8 test sonuçları

Şekil 4.9: Yolov10 test sonuçları

Şekil 4.10: Yolov11 test sonuçları

Yukardaki Şekil 4.8, Şekil 4.9 ve Şekil 4.11'da balık algılama modelinin sınıflandırma ve yerelleştirme görevlerini nasıl yerine getirdiğini gösteren kritik görsel çıktılardır. Her balık görüntüsünün üzerinde bir sınıf etiketi, güven oranı (confidence score) ve sınırlayıcı kutu (bounding box) bulunmaktadır. Görsel çıktılari ilişkin detaylar aşağıdaki Tablo 4.5'te sunulmuştur:

Tablo 4.5: Çıktı bileşenlerinin anlamı

Bileşen	Tanım	Anlamı
Sınıf Etiketi	Kutu üzerinde yazan balık türü (Örn: Hamsi, Barbun, Mezgit).	Modelin, çerçevelediği nesneyi hangi kategoriye atadığını gösterir.
Güven Oranı	Etiketin yanındaki ondalık sayı (Örn: 0.83, 0.96).	Modelin bu sınıflandırmanın doğru olduğuna dair olan güvenilirlik seviyesidir (1.00'e ne kadar yakınsa o kadar iyidir).
Sınırlayıcı Kutu	Nesneyi çevreleyen renkli dikdörtgen.	Modelin, algılanan nesnenin resimdeki konumunu ve boyutunu belirlediği yerelleştirme çıktısıdır.

YOLOv11 modeli, en yüksek güven skorlarını elde ederek genel olarak en güçlü performansı sergilemiştir. Model, 0.97'ye varan tekil skorlar kaydetmiştir. YOLOv8 ve YOLOv10 modeli de rekabetçi sonuçlar ortaya koymuşlardır. YOLOv10 modeli, test edilen örnekler arasında en düşük skorları (0.71) kaydederek genel güvenilirlik açısından geride kalmıştır. Sonuç olarak, bu üç modelden YOLOv11, hem en yüksek skorları vermesi hem de genel performansı ile öne çıkmaktadır.

5. SONUÇ VE ÖNERİLER

Bu tez çalışmasında, balık sınıflandırılması amacıyla derin öğrenme tabanlı nesne tespit algoritmalarından YOLOv8, YOLOv10 ve YOLOv11 modelleri kullanılarak barbun, hamsi ve mezgit türü balıkların tespiti ve sınıflandırması gerçekleştirilmiştir. Bu amaçla Bartın ilinden toplanan 1305'er adet barbun, hamsi ve mezgit görüntüsü olmak üzere toplam 3915 örnekten oluşan bir veri kümesi hazırlanmıştır. Bu üç türün seçilme nedeni, Karadeniz bölgesinde ekonomik değeri yüksek ve avcılığı en yaygın balık türleri arasında yer almalarıdır. Böylece çalışma, hem bölgeye ait balık türlerinin tanınmasına katkı sağlamakta hem de yerel balıkçılık endüstrisinde otomatik tür tanımlama süreçlerine bilimsel bir temel sunmaktadır.

Model eğitimi süreci, Google Colab üzerinde Ultralytics destekli PyTorch ortamında gerçekleştirilmiştir. Eğitim sürecinde 100 epoch boyunca model performansı Tablo 4.2 Tablo 4.3 ve Tablo 4.4'te göstermiştir. Doğruluk, duyarlılık, ortalama kesinlik değeri, mAP@50 ve mAP@50-95 gibi metriklerle değerlendirme yapılmıştır. Elde edilen sonuçlara göre, modeller iyi sonuçlar üretmiştir. mAP@0.5 değeri her üç modelde %90'ın üzerinde çıkmış, modellerin mAP@50-95 değeri ise YOLOv8 %77,7, YOLOv10 %78,9 ve YOLOv11 0.794'a kadar genel bir performans göstermiştir. Bu modelin balık işleme sektöründe otomasyon sistemlerine entegre edilebileceğine işaret etmektedir.

Gelecek çalışmalarda, balık türü çeşitliliği ve veri seti kapsamı artırılarak modelin sınıflandırma kabiliyeti geliştirilebilecektir. Ayrıca, eğitim verilerinin çevresel çeşitliliği artırılarak modelin farklı koşullara karşı daha dayanıklı olması sağlanabilecektir. Bununla birlikte, YOLOv11'in ilerleyen sürümleri ile karşılaştırmalı olarak incelenerek performans analizleri yapılabilecektir.

KAYNAKLAR

Aksaç, K. (2022). Prediction of the future success of candidates before recruitment with machine learning: A case study in the banking sector. Yüksek Lisans tezi, Galatasaray Üniversitesi Fen Bilimleri Enstitüsü, Endüstri Mühendisliği Anabilim Dalı, İstanbul, 72 s.

Aktürk, S. ve Serbest, K. (2022). Nesne tespiti için derin öğrenme kütüphanelerinin incelenmesi. *Journal of Smart Systems Research*, 3(2): 97–119.

Algur, Ö., Tümen, V. ve Yıldırım, Ö. (2018). Dış ortam görüntülerindeki insan hareketlerinin hibrit derin öğrenme yöntemleri kullanarak sınıflandırılması. *Fırat Üniversitesi Mühendislik Bilimleri Dergisi*, 30(3): 121–129.

Altay, A. ve Yılmaz, S. (2023). YOLO algoritması kullanılarak T hücrelerinin sınıflandırılması. *İleri Mühendislik Çalışmaları ve Teknolojileri Dergisi*, 3(2): 66–81.

Arslan, K. (2020). Eğitimde yapay zekâ ve uygulamaları. *Batı Anadolu Eğitim Bilimleri Dergisi*, 11(1): 71–88.

Bahadır, Y. (2015). YOLO ile engelli bireylerin kullandığı yardımcı gereçlerin tespiti. Yüksek Lisans Tezi, Eskişehir Osmangazi Üniversitesi Fen Bilimleri Enstitüsü, Elektrik-Elektronik Mühendisliği Anabilim Dalı, Eskişehir, 102 s.

Chollet, F. (2021). Deep learning with Python. Simon and Schuster.

Çankır, K., Noğay, H. S. ve Yiğit, E. (2025) Derin öğrenme tabanlı model ile orman ve göl görüntülerinin tanınması. *ISARC International Science and Art Research Center*, 149–158.

Dulkadir, S. ve Gültekin, G. K. (2023). Tarımsal otomasyon sistemleri için muz olgunluk seviyelerinin derin öğrenme yöntemleri ile sınıflandırılması. *EMO Bilimsel Dergi*, 13(3): 27–34.

Ghrairi, E. S. (2019). Konvolüsyonel sinir ağları kullanılarak çiçek türlerinin sınıflandırılması. Yüksek Lisans Tezi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, Konya, 58 s.

Gong, B., Dai, K., Shao, J., Jing, L. ve Chen, Y. (2023). Fish-TViT: A novel fish species classification method in multi water areas based on transfer learning and vision transformer. Heliyon, 9(6).

Gurkan, C., Kozalıoğlu, S. ve Palandöken, M. (2021). Real time mask detection, social distance and crowd analysis using convolutional neural networks and YOLO architecture designs. *Academic Perspective Procedia*, 4(1): 195–204.

Gülmez, B. (2023). Parçacık sürü optimizasyonu destekli derin öğrenme ile gül yaprağı hastalık tespiti. *1st International Conference on Modern and Advanced Research*, 348–352.

Hanbay, K. ve Üzen, H. (2017). Nesne tespit ve takip metotları: Kapsamlı bir derleme. *Türk Doğa ve Fen Dergisi*, 6(2): 40–49.

Hridayami, P., Putra, I. K. G. D. ve Wibawa, K. S. (2019). Fish species recognition using VGG16 deep convolutional neural network. *Journal of Computing Science and Engineering*, 13(3): 124–130.

Islam, S. M., Bani, S. I. ve Ghosh, R. (2021). Content-based fish classification using combination of machine learning methods. *International Journal of Information Technology and Computer Science (IJITCS)*, 13(8): 62-68.

Issamadine, M. A., Erkan, Y. ve Alaybeyoğlu, E. (2024). Industrial fish classifier with deep artificial neural network. *Orclever Proceedings of Research and Development*, 5(1): 99–109.

İnik, Ö. ve Ülker, E. (2017). Derin öğrenme ve görüntü analizinde kullanılan derin öğrenme modelleri. *Gaziosmanpaşa Bilimsel Araştırma Dergisi*, 6(3), 85–104.

İşçimen, B., Kutlu, Y., Uyan, A. ve Turan, C. (2015). Merkez-kenar uzunluğu yöntemi kullanılarak iki sırt yüzgeçli balık türlerinin sınıflandırılması. *Sinyal İşleme ve İletişim Uygulamaları Kurultayı.*

İşçimen, B., Kutlu, Y., Reyhaniye, A. N. ve Turan, C. (2014). Image analysis methods on fish recognition. *2014 22nd Signal Processing and Communications Applications Conference (SIU)*, 1411–1414. IEEE.

Jayasundara, J. M. V. D. B., Ramanayake, R. M. L. S., Senarath, H. M. N. B., Herath, H. M. S. L., Godaliyadda, G. M. R. I., Ekanayake, M. P. B., ... Ariyawansa, S. (2023). Deep learning for automated fish grading. *Journal of Agriculture and Food Research*, 14: 100711.

Jocher, G. ve Qiu, J. (2024). *Ultralytics yolo11*. GitHub: San Francisco, CA, USA.

Karacı, A. (2021). X-ışını görüntülerinden omuz implantlarının tespiti ve sınıflandırılması: YOLO ve önceden eğitilmiş evrişimsel sinir ağı tabanlı bir yaklaşım. *Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, 37(1): 283–294.

Kayaalp, K. ve Metlek, S. (2021). Derin öğrenme ile balık türlerinin tespiti. *International Journal of 3D Printing Technologies and Digital Industry*, 5(3): 569–576.

LeCun, Y., Bengio, Y. ve Hinton, G. (2015). *Deep learning. Nature*, 521(7553): 436–444.

Liu, H., Ma, X., Yu, Y., Wang, L. ve Hao, L. (2023). Application of deep learning-based object detection techniques in fish aquaculture: a review. *Journal of Marine Science and Engineering*, 11(4): 867.

Mandal, R., Connolly, R. M., Schlacher, T. A. ve Stantic, B. (2018). Assessing fish abundance from underwater video using deep neural networks. *2018 International Joint Conference on Neural Networks (IJCNN)*, 1–6. IEEE.

Mao, M. ve Hong, M. (2025). YOLO object detection for real-time fabric defect inspection in the textile industry: A review of YOLOv1 to YOLOv11. *Sensors (Basel, Switzerland)*, 25(7): 2270.

Miraç T. Ç., (2024), YOLOv8 nesne tespit modeli ile plastik parça yüzey kusurlarının gerçek zamanlı tespit edilmesi, Yüksek Lisans Tezi, Bursa Teknik Üniversitesi Lisansüstü Eğitim Enstitüsü, Endüstri Mühendisliği Anabilim Dalı, Bursa, 84 s.

Nair, S. S., Mon, F. A. ve Suthendran, K. (2018). Under water fish species recognition. *International Journal of Pure and Applied Mathematics*, 118(7): 357-361.

Villon, S., Mouillot, D., Chaumont, M., Darling, E. S., Subsol, G., Claverie, T. ve Villéger, S. (2018). A deep learning method for accurate and fast identification of coral reef fishes in underwater images. Ecological informatics, 48: 238-244.

Ogunlana, S. O., Olabode, O., Oluwadare, S. A. A. ve Iwasokun, G. B. (2015). Fish classification using support vector machine. *African Journal of Computing ve ICT*, 8(2): 75-82.

Pudaruth, S., Nazurally, N., Appadoo, C., Kishnah, S., Vinayaganidhi, M., Mohammoodally, I., ... Chady, F. (2020). SuperFish: A mobile application for fish species recognition using image processing techniques and deep learning. *International Journal of Computing and Digital Systems*, 10(1): 1-14.

Qin, H., Li, X., Liang, J., Peng, Y. ve Zhang, C. (2016). DeepFish: Accurate underwater live fish recognition with a deep architecture. Neurocomputing, 187: 49–58.

Rahman, W., Rahman, M. M., Mozumder, M. A. I., Sumon, R. I., Chelloug, S. A., Alnashwan, R. O. ve Muthanna, M. S. A. (2024). A Deep CNN-based salinity and freshwater fish identification and classification using deep learning and machine learning. Sustainability, 16(18): 7933.

Redmon, J., Divvala, S., Girshick, R. ve Farhadi, A. (2016). You only look once: Unified, real-time object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 779–788.

Saleh, A., Sheaves, M. ve Rahimi Azghadi, M. (2022). Computer vision and deep learning for fish classification in underwater habitats: A survey. Fish and Fisheries, 23(4): 977–999.

Salman, A., Jalal, A., Shafait, F., Mian, A., Shortis, M., Seager, J. ve Harvey, E. (2016). Fish species classification in unconstrained underwater environments based on deep learning. *Limnology and Oceanography: Methods*, 14(9): 570–585.

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3): 210–229.

Siri, D., Vellaturi, G., Ibrahim, S. H. S., Molugu, S., Desanamukula, V. S., Kocherla, R. ve Vatambeti, R. (2024). Enhanced deep learning models for automatic fish species identification in underwater imagery. *Heliyon*, *10*(15).

Sohan, M., Sai Ram, T. ve Rami Reddy, C. V. (2024). A review on yolov8 and its advancements. *In International Conference on Data Intelligence and Cognitive Informatics*, Springer, Singapore, 529-545.

Sung, M., Yu, S. C. ve Girdhar, Y. (2017). Vision based real-time fish detection using convolutional neural network. *OCEANS 2017- Aberdeen*, 1–6. IEEE.

Tan, F. G., Yüksel, A. S., Aydemir, E. ve Ersoy, M. (2021). Derin öğrenme teknikleri ile nesne tespiti ve takibi üzerine bir inceleme. *Avrupa Bilim ve Teknoloji Dergisi*, (25): 159–171.

Tan, L., Huangfu, T., Wu, L. ve Chen, W. (2021). Comparison of RetinaNet, SSD, and YOLO v3 for real-time pill identification. *BMC Medical Informatics and Decision Making*, 21: 1–11.

Tashiev, İ., Kul, S., Şentaş, A., Küçükayvaz, F., Eken, S., Sayar, A. ve Becerikli, Y. (2017). Konvolüsyonel sinir ağı kullanarak gerçek zamanlı araç tipi sınıflandırması. Real-Time Vehicle Type Classification Using Convolutional Neural Network. 1. *Ulusal Bulut Bilisim ve Büyük Veri Sempozyumu B3S'17At:* Antalya.

Turing, A. M. (1950*).* Computing machinery and intelligence. Mind.

Türkdamar, M. U., Taşyürek, M. ve Öztürk, C. (2023). Transfer öğrenmeli ve transfer öğrenmesiz derin ağlar ile inşaat alanında kask tespiti. *Niğde Ömer Halisdemir Üniversitesi Mühendislik Bilimleri Dergisi*, 12(1): 39-51.

Veranyurt, Ü., Deveci, A., Esen, M. F. ve Veranyurt, O. (2020). Makine öğrenmesi teknikleriyle hastalık sınıflandırması: random forest, k-nearest neighbour ve adaboost algoritmaları uygulaması. *Uluslararası Sağlık Yönetimi ve Stratejileri Araştırma Dergisi*, 6(2): 275-286.

Villon, S., Mouillot, D., Chaumont, M., Darling, E. S., Subsol, G., Claverie, T. ve Villéger, S. (2018). A deep learning method for accurate and fast identification of coral reef fishes in underwater images. *Ecological Informatics*, 48: 238–244.

Wageeh, Y., Mohamed, H. E. D., Fadl, A., Anas, O., ElMasry, N., Nabil, A. ve Atia, A. (2021). YOLO fish detection with Euclidean tracking in fish farms. *Journal of Ambient Intelligence and Humanized Computing*, 12: 5–12.

Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z. ve Han, J. (2024). Yolov10: Real-time end-to-end object detection. *Advances in Neural Information Processing Systems*, 37: 107984-108011.

Xu, W. ve Matzner, S. (2018). Underwater fish detection using deep learning for water power applications. *2018 International Conference on Computational Science and Computational Intelligence (CSCI)*, 313–318. IEEE.

Yılmaz, S. (2023). Beyin tümörü tanıları için YOLOv7 algoritması tabanlı karar destek sistemi tasarımı. *Kocaeli Üniversitesi Fen Bilimleri Dergisi*, 6(1): 47–56.

Yusup, I. M., Iqbal, M. ve Jaya, I. (2020). Real-time reef fishes identification using deep learning. IOP Conference Series: *Earth and Environmental Science*, 429(1): 012046.

ÖZGEÇMİŞ

Kişisel Bilgiler

Adı Soyadı	:	MAHAMAT AHMAT ISSAMADINE
Doğum Yeri ve Tarihi	:	

Eğitim Durumu

Lisans Öğrenimi	:	Bilgisayar Mühendisliği
Yüksek Lisans Öğrenimi	:	Akıllı Sistemler Mühendisliği
Bildiği Yabancı Diller	:	Arapaça, Fransızca, Türkçe
Bilimsel Faaliyet/Yayınlar	:	Industrial Fish Classifier with Deep Artificial Neural Network
Aldığı Ödüller	:	

İş Deneyimi

Stajlar	:	
Projeler ve Kurs Belgeleri	:	
Çalıştığı Kurumlar	:	

İletişim

E-Posta Adresi	:	

Tarih : 20/11/2025 (Tez Savunma Tarihi)