

T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ



**OTONOM İHA TABANLI DEMİRYOLU BAKIM SİSTEMİ İÇİN
YENİLİKÇİ DERİN ÖĞRENME YÖNTEMLERİNİN
GELİŞTİRİLMESİ**

Mehmet SEVİ

Doktora Tezi

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

OCAK 2025

T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Bilgisayar Mühendisliği Anabilim Dalı

Doktora Tezi

**OTONOM İHA TABANLI DEMİRYOLU BAKIM SİSTEMİ İÇİN
YENİLİKÇİ DERİN ÖĞRENME YÖNTEMLERİNİN GELİŞTİRİLMESİ**

Tez Yazarı
Mehmet SEVİ

Danışman
Prof. Dr. İlhan AYDIN

OCAK 2025
ELAZIĞ

T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Bilgisayar Mühendisliği Anabilim Dalı

Doktora Tezi

Başlığı:	Otonom IHA Tabanlı Demiryolu Bakım Sistemi için Yenilikçi Derin Öğrenme Yöntemlerinin Geliştirilmesi
Yazarı:	Mehmet SEVİ
İlk Teslim Tarihi:	16.12.2024
Savunma Tarihi:	17.01.2025

TEZ ONAYI

Fırat Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına göre hazırlanan bu tez aşağıda imzaları bulunan jüri üyeleri tarafından değerlendirilmiş ve akademik dinleyicilere açık yapılan savunma sonucunda OYBİRLİĞİ ile kabul edilmiştir.

Danışman:	Prof. Dr. İlhan AYDIN Fırat Üniversitesi, Mühendislik Fakültesi	İmza Onayladım
Başkan:	Prof. Dr. Fatih ÖZKAYNAK Fırat Üniversitesi, Teknoloji Fakültesi	Onayladım
Üye:	Doç. Dr. Muhammed YILDIRIM Malatya Turgut Özal Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi	Onayladım
Üye:	Dr. Öğr. Üyesi Gülşah KARADUMAN Fırat Üniversitesi, Mühendislik Fakültesi	Onayladım
Üye:	Dr. Öğr. Üyesi Canan TAŞTİMUR TEMİZ Erzincan Binali Yıldırım Üniversitesi, Mühendislik-Mimarlık Fakültesi	Onayladım

Bu tez, Enstitü Yönetim Kurulunun/...../20..... tarihli toplantısında tescillenmiştir.

İmza

Prof. Dr. Burhan ERGEN
Enstitü Müdürü

BEYAN

Fırat Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırladığım “Otonom IHA Tabanlı Demiryolu Bakım Sistemi için Yenilikçi Derin Öğrenme Yöntemlerinin Geliştirilmesi” Başlıklı Doktora Tezimin içindeki bütün bilgilerin doğru olduğunu, bilgilerin üretilmesi ve sunulmasında bilimsel etik kurallarına uygun davrandığımı, kullandığım bütün kaynakları atıf yaparak belirttiğimi, maddi ve manevi desteği olan tüm kurum/kuruluş ve kişileri belirttiğimi, burada sunduğum veri ve bilgileri unvan almak amacıyla daha önce hiçbir şekilde kullanmadığımı beyan ederim.

17.01.2025

Mehmet SEVİ



ÖNSÖZ

Bu çalışmada, demiryollarının yaygın kullanımı ve hızlı ulaşımın insan yaşamında vazgeçilmez hale gelmesi nedeniyle demiryolu hatlarındaki kusurların tespitinin önemine dikkat çekilmiştir. Bu bağlamda, ilgili kişi ve kuruluşların önerilen önlem politikalarını titizlikle uygulamasını umut etmekteyim. Ayrıca doktora sürecimde bilgi ve tecrübelerini benimle paylaşarak bana destek olan ve cesaretlendiren değerli danışmanım Prof. Dr. İlhan AYDIN'a ve daima yanımda olan aileme gönülden teşekkür ederim.

Bu tez çalışması, Türkiye Bilimsel ve Teknolojik Araştırma Kurumu (TÜBİTAK) tarafından 120E097 numaralı proje ile desteklenmiştir. TÜBİTAK'a desteklerinden dolayı teşekkür ederim. Bu tez çalışması ayrıca, Fırat Üniversitesi Bilimsel Araştırma Projeleri Koordinasyon Birimi (FÜBAP) tarafından ADEB.2022.02. protokol numaralı proje ile desteklenmiştir.

Mehmet SEVİ
ELAZIĞ, 2025

İÇİNDEKİLER

Sayfa

ÖNSÖZ.....	iv
İÇİNDEKİLER	v
ÖZET	viii
ABSTRACT	ix
ŞEKİLLER LİSTESİ	x
TABLolar LİSTESİ	xv
SİMGELER VE KISALTMALAR	xvii
1. GİRİŞ	1
1.1. Demiryolu Bileşenleri.....	3
1.2. Demiryolu Bileşenlerindeki Arızalar ve Sorunlar	4
1.3. Gazebo.....	5
1.4. Parrot Anafı4K İHA	6
1.5. Tezin Amacı	7
1.6. Karşılaşılan Zorluklar	8
1.7. Teze Genel Bakış.....	9
2. LİTERATÜR ÖZETİ	10
3. TEMEL BİLGİLER	16
3.1. Derin Öğrenme	16
3.2. Derin Öğrenme Modelleri	18
3.2.1. Evrişimli Sinir Ağları (CNN)	18
3.2.1.1. Konvolüsyon Katmanı	19
3.2.1.2. Havuzlama Katmanı	19
3.2.1.3. Tam Bağlantılı Katman (Fully Connected Layer)	20
3.2.1.4. Yığın Normalleştirme Katmanı (Batch Normalization Layer)	20
3.2.1.5. Düşüm Seyreltme Katmanı (Dropout Katmanı)	20
3.2.1.6. Aktivasyon Fonksiyonları.....	21
3.2.2. Diğer CNN Modelleri.....	22
3.2.2.1. VGG-16	22
3.2.2.2. ResNet	22
3.2.2.3. InceptionV3	23
3.2.2.4. Hafif Evrişimli Sinir Ağı (Lightweight Convolutional Neural Network)	23
3.2.2.5. YOLO	24
3.2.2.6. Grounding DINO ve Zero-Shot Nesne Tespiti	28
3.2.3. Görüntü Bölütleme	28
3.2.3.1. Unet	29
3.2.3.2. Genişleyen Evrişimsel Sinir Ağları (Dilated Convolutional Neural Networks)	30
3.2.3.3. BiSeNetV2.....	31
3.2.3.4. SAM (Segment Anything Model).....	32
3.3. Görüntü Arttırım Teknikleri	33
3.3.1. Geleneksel Teknikler.....	33
3.3.2. Yapay Zekâ Tabanlı Teknikler	34
3.4. Değerlendirme Ölçütleri	34

4. MATERYAL VE METOT	38
4.1. Simülasyon Ortamında İHA ile Ray Takibi ve Tespiti.....	39
4.1.1. Ufuk Noktası Yöntemi ile Ray Takibi.....	39
4.1.2. Bilgisayar Görmesi Yöntemleri ile Ray Takibi.....	41
4.1.3. BiSeNetV2 ile Demiryolu Hattı Tespiti	43
4.2. Derin Öğrenme Yöntemleri ile Demiryolu Bağlantı Elemanlarının Sınıflandırılması	43
4.3. Bağlantı Elemanlarının Tespiti ve Hata Sınıflandırması için Önerilen Hibrit Yaklaşım	46
4.4. Demiryolu Bağlantı Elemanları Üzerindeki Kusurların Tespit Edilmesi	51
4.5. Anlamsal Bölütleme Kullanarak Ray Üzerindeki Kusurları Tespit Etme	62
4.6. SAM Kullanarak Demiryolu Raylarının Bölütlenmesi.....	64
4.7. Topluluk Öğrenmesine Dayalı Olarak Ray Yüzeyindeki Kusurların Tespiti	66
4.8. Üretken Çekişmeli Ağlar ve Topluluk Yöntemine Dayalı Ray Üzerindeki Kusurların Sınıflandırması	69
4.9. Demiryolu Hatlarını İçeren Görüntülerde Bir Topluluk Modeli Kullanarak Unet Bölütleme Performansını İyileştirme.....	71
4.10. YOLOv8 ile Demiryolu Hattı Çevresindeki Yabancı Nesnelerin Tespiti.....	74
4.11. DNet ve Matematiksel Yöntemler Kullanarak Geliştirilmiş Demiryolu İzleme ve Bölütleme	75
4.12. Hızlı ve Ekonomik Nesne Algılama Yoluyla Demiryolu Güvenliğinin Artırılması: Grounding Dino Yaklaşımı	81
5. BULGULAR VE TARTIŞMA	83
5.1. Simülasyon Ortamında İHA ile Ray Takibi ve Tespitinde Elde Edilen Bulgular	83
5.1.1. Ufuk Noktası Yöntemi ile Ray Takibi Metodunda Elde Edilen Bulgular	83
5.1.2. Bilgisayar Görmesi Yöntemleri ile Ray Takibi Metodunda Elde Edilen Bulgular	84
5.1.3. BiSeNetV2 ile Demiryolu Hattı Tespiti Metodunda Elde Edilen Bulgular.....	86
5.2. Derin Öğrenme Yöntemleri ile Demiryolu Bağlantı Elemanlarının Sınıflandırılması Metodunda Elde Edilen Bulgular.....	87
5.3. Bağlantı Elemanlarının Tespiti ve Hata Sınıflandırması için Önerilen Hibrit Yaklaşım Metodunda Elde Edilen Bulgular.....	92
5.4. Demiryolu Bağlantı Elemanları Üzerindeki Kusurların Tespit Edilmesi Metodunda Elde Edilen Bulgular	96
5.5. Anlamsal Bölütleme Kullanarak Ray Üzerindeki Kusurları Tespit Etme Metodunda Elde Edilen Bulgular	103
5.6. SAM Kullanarak Demiryolu Raylarının Bölütlenmesi Metodunda Elde Edilen Bulgular	107
5.7. Topluluk Öğrenmesine Dayalı Olarak Ray Yüzeyindeki Kusurların Tespiti Metodunda Elde Edilen Bulgular	108
5.8. Üretken Çekişmeli Ağlar ve Topluluk Yöntemine Dayalı Ray Üzerindeki Kusurların Sınıflandırması Metodunda Elde Edilen Bulgular	118
5.9. Demiryolu Hatlarını İçeren Görüntülerde Bir Topluluk Modeli Kullanarak Unet Bölütleme Performansını İyileştirme Metodunda Elde Edilen Bulgular	122
5.10. YOLOv8 ile Demiryolu Hattı Çevresindeki Yabancı Nesnelerin Tespiti Metodunda Elde Edilen Bulgular	127
5.11. DNET ve Matematiksel Yöntemler Kullanarak Geliştirilmiş Demiryolu İzleme ve Bölütleme Metodunda Elde Edilen Bulgular	129
5.12. Hızlı ve Ekonomik Nesne Algılama Yoluyla Demiryolu Güvenliğinin Artırılması: Grounding Dino Yaklaşımı Metodunda Elde Edilen Bulgular	135
6. SONUÇLAR.....	137

ÖNERİLER	139
KAYNAKLAR	140
ÖZGEÇMİŞ	



ÖZET

Otonom İHA Tabanlı Demiryolu Bakım Sistemi için Yenilikçi Derin Öğrenme Yöntemlerinin Geliştirilmesi

Mehmet SEVİ

Doktora Tezi

FIRAT ÜNİVERSİTESİ

Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

Ocak 2025, Sayfa: xvii + 147

Bu tez çalışmasının ana amacı, demiryolu hatlarının ve bileşenlerinin otomatik olarak izlenmesi ve kusurların yüksek doğrulukla tespit edilmesidir. Tezde, geleneksel yöntemlerin yerini alan otonom sistemler ile iş gücü ve zaman kaybını en aza indirerek verimliliği artırmak hedeflenmektedir. Çalışmanın temel motivasyonlarından biri, ray hatlarında ve bileşenlerinde bulunan yabancı nesnelerin tespit edilmesidir. Bu tür nesneler, ray hattı güvenliğini olumsuz etkileyebileceğinden, demiryolu ulaşımının güvenliği açısından önemli bir risk oluşturmaktadır.

Simülasyon ortamında geliştirilen yöntem, gerçek dünya şartlarına uyarlanarak daha etkili ve güvenilir demiryolu bakım sistemleri geliştirilmesine olanak tanımaktadır. Proje kapsamında, Gazebo simülasyon ortamında Parrot Anafi4K İHA kullanılarak demiryolu hattı boyunca uçuş simülasyonları gerçekleştirilmiştir. Gerçek dünya ortamında yapılacak olan çalışmalarda, İHA ile toplanan görüntüler hibrit derin öğrenme tabanlı modellerle işlenerek, demiryolu bileşenlerindeki kusurların tespiti sağlanacaktır.

Çalışmada YOLO, Unet, BiSeNetV2 gibi derin öğrenme algoritmaları ve hibrit yöntemler kullanılarak, demiryolu bileşenlerinde oluşan çatlaklar, deforme bağlantı elemanları gibi kusurların yüksek doğrulukla tespiti amaçlanmaktadır. Bu sayede, demiryolu bakım işlemleri daha hızlı ve maliyetsiz bir şekilde gerçekleştirilebilecektir. Ayrıca, geleneksel Unet modelinden daha iyi performans gösteren, Unet tabanlı geliştirilmiş bir DNet bölütleme modeli de geliştirilmiştir.

Geliştirilen yöntemlerin bulguları, literatürdeki çalışmalarla karşılaştırılarak detaylı bir şekilde sunulmuş ve başarılı ve etkili sonuçlar elde edilmiştir.

Anahtar Kelimeler: Derin Öğrenme, Demiryolu, Bölütleme, Kusur Tespiti, Kusur Sınıflandırması, Veri Oluşturma

ABSTRACT

Development of Innovative Deep Learning Methods for Autonomous UAV-Based Railway Maintenance System

Mehmet SEVİ

Ph.D. Thesis

FIRAT UNIVERSITY
Graduate School of Natural and Applied Sciences
Department of Computer Engineering

January 2025, Pages: xvii + 147

The main purpose of this thesis is to automatically monitor railway lines and components and detect defects with high accuracy. The thesis aims to increase efficiency by minimizing labor and time loss with autonomous systems that replace traditional methods. One of the main motivations of the study is to detect foreign objects on rail lines and components. Such objects pose a significant risk to railway transportation safety, as they can negatively affect rail line safety.

The method developed in the simulation environment enables the developing of more effective and reliable railway maintenance systems by adapting it to real-world conditions. Within the project's scope, flight simulations were carried out along the railway line using the Parrot Anafi4K UAV in the Gazebo simulation environment. In the studies to be carried out in the real-world environment, the images collected by the UAV will be processed with hybrid deep learning-based models to detect defects in railway components.

The study aims to detect defects such as cracks and deformed fasteners in railway components with high accuracy by using deep learning algorithms such as YOLO, Unet, BiSeNetV2, and hybrid methods. In this way, railway maintenance operations can be carried out faster and less costly. In addition, an improved DNet segmentation model based on Unet has been developed, which outperforms the traditional Unet model.

The findings of the developed methods were presented in detail by comparing them with the studies in the literature and successful and effective results were obtained.

Keywords: Deep Learning, Railway, Segmentation, Flaw Detection, Flaw Classification, Data Generation

ŞEKİLLER LİSTESİ

	Sayfa
Şekil 1.1. Ultrasonik ray muayene cihazı	1
Şekil 1.2. Demiryolu bileşenleri	4
Şekil 1.3. Örnek demiryolu arızaları (a) kusurlu bağlantı elemanı (b) ray üzerindeki çatlak	5
Şekil 1.4. Örnek demiryolu hattı çevresindeki yabancı cisimler	5
Şekil 1.5. Gazebo çalışma ortamı	6
Şekil 1.6. (a) Gerçek dünya ortamındaki Anafı4K İHA (b) Simülasyon ortamındaki Anafı4K İHA	6
Şekil 1.7. Tezin yapısı	9
Şekil 2.1. Demiryollarındaki kusurların tespiti ve sınıflandırılması	15
Şekil 3.1. Derin Öğrenme yapay zekâ ilişkisi	16
Şekil 3.2. Yıllara göre ImageNet yarışmasında elde edilen en iyi hata oranları	17
Şekil 3.3. Örnek CNN mimarisi	19
Şekil 3.4. Maksimum havuzlama	20
Şekil 3.5. ResNet artık blok	22
Şekil 3.6. InceptionV3 ağı	23
Şekil 3.7. Hafif evrişimli sinir ağı mimarisi	24
Şekil 3.8. Nesne tespitinden sorumlu YOLO tespit sistemi [60]	25
Şekil 3.9. YOLOv4 mimarisi	25
Şekil 3.10. YOLOv5 mimarisinin alt mimarileri	26
Şekil 3.11. YOLOv5 mimarisinin genel yapısı	27
Şekil 3.12. YOLOv8 mimarisi	27
Şekil 3.13. Anlamsal ve örnek bölütleme arasındaki fark	29
Şekil 3.14. Unet mimarisi	30
Şekil 3.15. D-CNN mimarisi	31
Şekil 3.16. BiSeNetV2 mimarisi	32
Şekil 3.17. SAM mimarisi	33
Şekil 3.18. GAN mimarisi	34
Şekil 3.19. İki sınıflı bir problem için karışıklık matrisi	35
Şekil 4.1. Önerilen yöntemin genel blok gösterimi	38
Şekil 4.2. Gazebo ortamı	39
Şekil 4.3. PID denetleyicisi şeması	41
Şekil 4.4. Uygulamanın genel yapısı	41

Şekil 4.5. Kusurlu bağlantı elemanı üretimi (a) Sağlıklı bağlantı elemanı (b) Sağlıklı referans veri (c) Unet bölütleme sonucu (d) GrabCut ile ön plana çıkarılmış bağlantı elemanı (e) Yeni üretilen kusurlu bağlantı elemanı (f) Şeffaflaştırılmış kusurlu bağlantı elemanı (g) Varsayılan arka plan (h) Varsayılan arka plan eklenmiş kusurlu bağlantı elemanı	44
Şekil 4.6. Oluşturulan CNN mimarisi	46
Şekil 4.7. Bağlantı elemanının kusur tespiti için önerilen hibrit çerçeve	47
Şekil 4.8. YOLOv4'te ve YOLOv4-Tiny'de kullanılan aktivasyon fonksiyonları	48
Şekil 4.9. YOLOv4-Tiny mimarisi	48
Şekil 4.10. YOLOv4 Tiny'de CSP blok kullanımı	48
Şekil 4.11. YOLOv4-Tiny modelini test etmek için tasarlanan arayüz	49
Şekil 4.12. Önerilen hafif CNN modeli	49
Şekil 4.13. Standart ve derinlemesine ayrılabilir evrişimler	50
Şekil 4.14. Ters çevrilmiş artık blok	51
Şekil 4.15. Önerilen sistemin blok şeması	52
Şekil 4.16. Yüksek kontrastlı görüntüyü hesaplaması	54
Şekil 4.17. Ray ve travers algılamasının gösterimi	55
Şekil 4.18. Ray çıkarma algoritması	56
Şekil 4.19. Ray ve travers tespitinin çizimi	57
Şekil 4.20. Yerel ikili örüntü yöntemi	58
Şekil 4.21. LLBP yapısı ve farklı n değerleri için uygulaması	59
Şekil 4.22. Sırasıyla bağlantı elemanı algılama adımları	59
Şekil 4.23. LCNN mimarisi ve sınıflandırması	61
Şekil 4.24. Önerilen anlamsal bölütleme modeli	63
Şekil 4.25. Veri setindeki görüntüler (a) Kusurlu ray görüntüleri (b) Kusurlu raylara ait referans görüntüler	64
Şekil 4.26. SAM çalışmasının akış şeması	64
Şekil 4.27. Railsem19 veri setinden örnekler	65
Şekil 4.28. Veri setinden etiketlenmiş örnekler	67
Şekil 4.29. Önerilen modelin akış diyagramı	68
Şekil 4.30. Önerilen topluluk modelinin akış diyagramı	69
Şekil 4.31. Önerilen topluluk modeli	70
Şekil 4.32. Önerilen bölütleme modelinin mimarisi	72
Şekil 4.33. Veri setinden örnekler	74
Şekil 4.34. Önerilen yöntem	77
Şekil 4.35. Önerilen DNet mimarisi	79

Şekil 4.36. Yapılan saha çalışmasından görüntü	81
Şekil 5.1. Gabor filtresi çıktısı.....	83
Şekil 5.2. (a) Canny algoritması çıktısı (b) Kırpılmış hali.....	83
Şekil 5.3. Hough dönüşümünün çıktısı.....	84
Şekil 5.4. Ufuk noktası tahmini.....	84
Şekil 5.5. (a) Grayscale sonucucu (b) Gaussian sonucu	85
Şekil 5.6. Eşiklenmiş görüntü.....	85
Şekil 5.7. Tespit edilen rota.....	85
Şekil 5.8. BiSeNetV2 modelin simülasyon ortamındaki test sonuçları	86
Şekil 5.9. Unet eğitim sonuçları	87
Şekil 5.10. Unet sonucu elde edilen görüntüler (a) Sağlıklı bağlantı elemanı (b) Sağlıklı referans veri (c) Bölütleme sonucu (d) Ön plana çıkarılmış arızalı bağlantı elemanı (e) Ön plana çıkarılmış eksik bağlantı elemanı	88
Şekil 5.11. Önerilen modellerin öğrenme grafikleri (a) CNN (b) ResNet50 (c) VGG-16.....	89
Şekil 5.12. (a) Sentetik gürültü eklenmiş bağlantı elemanı (b) İyileştirilmiş ve üretilmiş bağlantı elemanı	90
Şekil 5.13. YOLOv4-Tiny'un mAP ve kayıp grafiği. (a) mAP (b) Kayıp grafiği	92
Şekil 5.14. YOLOv4-Tiny için kesinlik-duyarlılık eğrisi.....	92
Şekil 5.15. Farklı koşullar için bağlantı elemanı algılama sonuçları. (a) 4 adet sağlıklı bağlantı elemanı, (b) 1 adet sağlıklı 1 adet kusurlu bağlantı elemanı, (c) 1 adet eksik 1 adet sağlıklı bağlantı elemanı	93
Şekil 5.16. Eğitim ve doğrulama veri seti için doğruluk ve kayıp grafiği	94
Şekil 5.17. Farklı CNN modellerinin karışıklık matrisleri. (a) InceptionV3 (b) VGG16, (c) MobileNetv2, (d) Önerilen CNN	94
Şekil 5.18. Ölçüm sisteminin yapısı	96
Şekil 5.19. Üç farklı bağlantı elemanı durumu (a) Sağlıklı (b) Deforme (c) Kayıp	97
Şekil 5.20. FC1 ve FC2'den çıkarılan özelliklerin t-sne temsili	100
Şekil 5.21. Veri setinden örnek görüntüler; (a) Orijinal kusurlu ray görüntüleri, (b) Referans görüntüler	104
Şekil 5.22. DCNN modelinin sonucu (a) Orijinal gri tonlamalı kusur, (b) Referans kusur (c) DCNN modelinin tahmini.....	104
Şekil 5.23. Unet modelinin sonucu (a) Orijinal gri tonlamalı kusur (b) Referans kusur (c) Unet modelinin tahmini	105
Şekil 5.24. Önerilen modellerin sonuçlarının karşılaştırılması of (a) Orijinal gri tonlamalı kusur, (b) Referans kusur, (c) Unet modelinin tahmini (99.99% doğruluk oranı) ve (d) DCNN modelinin tahmini (96.23% doğruluk oranı).....	106
Şekil 5.25. Unet doğruluk grafiği	106
Şekil 5.26. Unet kayıp grafiği.....	106

Şekil 5.27. Orijinal (sol) ve segmente edilmiş (sağ) görüntü	108
Şekil 5.28. Elde edilen sol ve sağ raya bölütleme görselleri	108
Şekil 5.29. YOLOv4 modelinin kayıp grafiği	109
Şekil 5.30. YOLOv4 modelinin test sonuçları.....	109
Şekil 5.31. YOLOv5s mimarisinin eğitim ve doğrulama sonuçları.....	110
Şekil 5.32. YOLOv5s alt mimarisinin test sonuçları	110
Şekil 5.33. YOLOv5n alt mimarisinin eğitim ve doğrulama sonuçları	111
Şekil 5.34. YOLOv5n alt mimarisinin test sonuçları	111
Şekil 5.35. YOLOv5m mimarisinin eğitim ve doğrulama sonuçları	112
Şekil 5.36. YOLOv5m alt mimarisinin test sonuçları	112
Şekil 5.37. YOLOv5l mimarisinin eğitim ve doğrulama sonuçları	113
Şekil 5.38. YOLOv5l alt mimarisinin test sonuçları	113
Şekil 5.39. 416x416 girdi boyutlu YOLOv6s alt mimarisinin test sonuçları	115
Şekil 5.40. 416x416 girdi boyutlu YOLOv6n alt mimarisinin test sonuçları	115
Şekil 5.41. 416x416 girdi boyutlu YOLOv6t alt mimarisinin test sonuçları	115
Şekil 5.42. 640x640 girdi boyutlu YOLOv6s alt mimarisinin test sonuçları	116
Şekil 5.43. 640x640 girdi boyutlu YOLOv6n alt mimarisinin test sonuçları	116
Şekil 5.44. 640x640 girdi boyutlu YOLOv6t alt mimarisinin test sonuçları	116
Şekil 5.45. GAN ile üretilen kusurlu ray görüntüleri	118
Şekil 5.46. ResNet50 doğruluk ve kayıp grafiği.....	119
Şekil 5.47. InceptionV3 doğruluk ve kayıp grafiği	120
Şekil 5.48. Topluluk modeli doğruluk ve kayıp grafiği.....	120
Şekil 5.49. Topluluk modeli karışıklık matrisi	121
Şekil 5.50. Unet modelinin doğrulama ve eğitim grafikleri	122
Şekil 5.51. Unet modelinin tahmini.....	122
Şekil 5.52. Kullanılan MobileNetV2 omurgasının eğitim ve doğrulama sonuçları.....	123
Şekil 5.53. Kullanılan ResNet34 omurgasının eğitim ve doğrulama sonuçları	123
Şekil 5.54. Kullanılan VGG16 omurgasının eğitim ve doğrulama sonuçları	124
Şekil 5.55. Test sonucunda topluluk kararı ile elde edilen görüntü.....	126
Şekil 5.56. YOLOv8 test sonuçları.....	128
Şekil 5.57. BiSeNetV2 modelinin (a) simülasyon ve (b) gerçek saha ortamındaki test sonuçları.....	129
Şekil 5.58. (a) YOLOv7 ve (b) YOLOv8 modellerinin gerçek saha ortamındaki test sonuçları	130
Şekil 5.59. (a) Tespit edilen sol, sağ ray ve orta noktanın Bezier görüntüsü ve (b) saha sonuçları.....	130

Şekil 5.60. (a) DNet eğitim ve doğrulama kaybı grafiği (b) DNet eğitim ve doğrulama doğruluğu grafiği (c) Klasik Unet eğitim ve doğrulama doğruluğu grafiği (d) Klasik Unet eğitim ve doğrulama kaybı grafiği.....	131
Şekil 5.61. Oluşturulan veri seti üzerinde DNet test sonuçları.....	132
Şekil 5.62. (a) DNet'in Railsem19 veri kümesindeki test sonucu (b) Klasik Unet'in Railsem19 veri kümesindeki test sonucu	134
Şekil 5.63. Demiryolu etrafında tespit edilen insanlar.....	135
Şekil 5.64. Demiryolu etrafında tespit edilen oturan ve yürüyen insanlar.....	136
Şekil 5.65. Demiryolu etrafında tespit edilen insan, ray ve bitki.....	136



TABLÖLAR LİSTESİ

	Sayfa
Tablo 1.1. Hem Gazebo'da hem de gerçek dünyada kullanılan İHA'lar	7
Tablo 4.1. Önerilen modellerinin katman ve parametre sayıları	46
Tablo 4.2. Eğitim parametreleri	46
Tablo 4.3. Önerilen CNN'in yapısı ve parametreleri.....	51
Tablo 4.4. Eğitimde kullanılan parametreler.....	67
Tablo 4.5. Oluşturulan veri setinin detayları.....	79
Tablo 4.6. DNet ve klasik Unet'in karşılaştırılması	80
Tablo 5.1. BiSeNetV2 modeli sonuçları	86
Tablo 5.2. BiSeNetV2'nin bölütleme performansının literatürdeki diğer çalışmalarla karşılaştırılması	86
Tablo 5.3. VGG-16 mimarisinin performans metrikleri	88
Tablo 5.4. ResNet50 mimarisinin performans metrikleri.....	88
Tablo 5.5. CNN mimarisinin performans metrikleri.....	89
Tablo 5.6. Seçilen seyretme değerine göre başarımlar oranları	89
Tablo 5.7. Önerilen modellerin gürültülü ve üretilmiş veri seti üzerindeki performansları	91
Tablo 5.8. Önerilen CNN Modelinin Literatürdeki Modellerle Karşılaştırılması	91
Tablo 5.9. Her bir sınıf için örnek sayıları	93
Tablo 5.10. Hafif evrişimli CNN ağının parametreleri	93
Tablo 5.11. Farklı transfer öğrenme modelleriyle performans karşılaştırması	95
Tablo 5.12. Bağlantı elemanı tanıma yöntemlerinin karşılaştırma sonuçları	95
Tablo 5.13. Görüntü iyileştirme öncesi ve sonrası karşılaştırmalı sonuçlar.....	98
Tablo 5.14. Farklı tam bağlantılı katmanlar için önerilen CNN'nin performansı.....	99
Tablo 5.15 . Önerilen CNN ve temel önceden eğitilmiş CNN modelleri için özellik çıkarma süreci.....	100
Tablo 5.16. Farklı önceden eğitilmiş CNN modelleriyle karşılaştırma sonuçları	101
Tablo 5.17. Önerilen yöntemin literatürdeki temel yöntemler ile karşılaştırılması.....	102
Tablo 5.18. DCNN modelinin performans metrikleri	105
Tablo 5.19. Unet modelinde elde edilen başarımlar oranları	105
Tablo 5.20. Önerilen modellerin literatürdeki modellerle karşılaştırılması	107
Tablo 5.21. Bölütleme sonuçları	108
Tablo 5.22. YOLOv4 ve YOLOv5 mimarilerinin eğitim sonucunda sınıf bazında elde ettiği map.5 değerleri	114
Tablo 5.23. YOLOv5 modelleri ile topluluk modelleme yönteminin doğrulama performansları	114

Tablo 5.24. YOLOv6 alt mimarilerinin eğitim sonuçları.....	114
Tablo 5.25. YOLOv5 modelleri ile topluluk modelleme yönteminin doğrulama performansları.....	117
Tablo 5.26. YOLOv5 topluluk modelleme yöntemi ile literatürdeki yaklaşımların performans karşılaştırması	117
Tablo 5.27. GAN performansı değerlendirme metrikleri.....	119
Tablo 5.28. Topluluk modeli performansı değerlendirme metrikleri	120
Tablo 5.29. Önerilen topluluk modelinin literatürdeki çalışmalarla karşılaştırılması	121
Tablo 5.30. Hibrit kayıp fonksiyonunun eğitim performansına etkisi	124
Tablo 5.31. Topluluk model ve farklı omurgalara sahip Unet modellerinin ortalama IoU değerleri.....	125
Tablo 5.32. Farklı katsayılara sahip topluluk modeli performansları.....	125
Tablo 5.33. Topluluk modeli (eşit katsayı) ve farklı omurgalara sahip Unet modellerinin doğruluk oranı.....	125
Tablo 5.34. Farklı katsayılara sahip topluluk modelinin ve farklı omurgalara sahip Unet modellerinin doğruluk oranı değerleri.....	126
Tablo 5.35. Önerilen modelin literatürdeki çalışmalarla karşılaştırılması	127
Tablo 5.36. YOLOv8 eğitim sonuçları	128
Tablo 5.37. YOLOv8 doğrulama sonuçları.....	128
Tablo 5.38. BiSeNetV2, YOLOv7 ve YOLOv8 sonuçları.....	129
Tablo 5.39. DNet ve Klasik Unet Modellerinin Performans Karşılaştırması.....	131
Tablo 5.40. Oluşturulan Veri Setinin Sınıf Bazında Performans Karşılaştırması	132
Tablo 5.41. Oluşturulan yeni veri setinin sınıf tabanlı performans karşılaştırması.....	133
Tablo 5.42. Yeni Railsem19 veri setinde DNet'in performansı	133
Tablo 5.43. DNet modelinin literatürdeki diğer bölütleme yöntemleriyle karşılaştırılması	134

SİMGELER VE KISALTMALAR

Simgeler

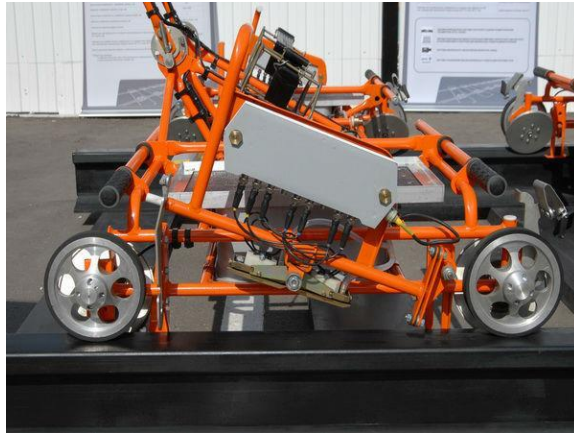
$\sigma_{softmax}$	: Softmax Fonksiyonu
$z^{(i)}$	: Modelin i-inci sınıf için verdiği ham çıkış
G_x	: Görüntüdeki x eksen boyunca yoğunluk değişimi
G_y	: Görüntüdeki y eksen boyunca yoğunluk değişimi
N	: Toplam sınıf sayısı
AP_i	: Ortalama AP değeri

Kısaltmalar

FPS	: Frame Per Second
İHA	: İnsansız Hava Aracı
HDR	: High Dynamic Range
GB	: Gigabayt
ESA	: Evrişimsel Sinir Ağları
mAP	: Mean Average Precision
CNN	: Evrişimli Yapay Sinir Ağı (Convolutional Neural Network)
Fast-SCNN	: Fast Semantic Segmentation Network
YOLO	: You Only Look Once
IoT	: Nesnelerin İnterneti
A	: Doğruluk
P	: Kesinlik
R	: Duyarlılık
F1	: F-Ölçütü
AP	: Ortalama Kesinlik
mAP	: Ortalama Ortalama Kesinlik
IoU	: Kesişim Alanının Birleşim Alanına Oranı
GAN	: Generative Adversarial Networks
ROI	: Region of Interest
DATEM	: Demiryolu Araştırma ve Teknoloji Merkezi
IoT	: Internet of Things
SAM	: Segment Anything Model

1. GİRİŞ

Demiryolu ağlarının hızla genişlemesi ve sunduğu büyük fırsatlar, demiryolu taşımacılığını cazip bir ulaşım seçeneği haline getirmiştir. Bu nedenle, son yıllarda demiryolu taşımacılığı üzerine yapılan araştırmalar artarak devam etmektedir. Demiryolu taşımacılığına olan bu artan ilgi, beraberinde bu alanda karşılaşılan sorunların da ele alınmasını gerektirmiştir. Metal tekerlekli araçlarla mekanik kuvvet uygulayarak insan ve yük taşımacılığını gerçekleştiren tüm sistemler "demiryolu" olarak tanımlanmaktadır [1]. Hızlı trenlerin hızla gelişmesiyle yıllık yolcu sayısına da yansımıştır. 2019 yılı itibarıyla 283 milyon kişi ulaşım için demiryollarını kullanmıştır [2]. Hızlı trenlerin yolcu sayısının artmasıyla demiryollarında oluşan yıpranma miktarı da artmaktadır. Bu yıpranmaya sürekli kullanım ve doğal etmenler neden olabilmektedir. Demiryollarında oluşan yıpranma ve kusurlar ciddi kazalara ve maddi kayıplara neden olmaktadır [3]. Bu sebepten ötürü demiryollarının periyodik bakımı düzenli olarak yapılmalıdır [3]. Geleneksel yöntemlerde insan muayenesi ve temaslı ölçüm teknikleri, raylardaki arızaları tespit etmek için halen kullanılmaktadır. Bu da kusur tespitinde düşük kesinlik ve düşük doğruluk oranına neden olmaktadır. Demiryollarının güvenliği için belli aralıklarla bu ölçümlerin yapılması gerekmektedir. Özel ray ölçüm cihazlarıyla yapılan bu muayene zamandan ve işten kayba neden olmaktadır. Şekil 1.1’de bakım yapmak için kullanılan ultrasonik ray muayene cihazı görülmektedir. Bu cihazları kullanmak için demiryolunun boş olduğu bir zaman seçilmelidir. Günümüzde demiryolu incelemesi için ultrasonik dalgaların yanı sıra görsel kameralar, akustik emisyon ve termografi girdap akımı gibi yöntemler kullanılmaktadır. Bütün bu yöntemler zamandan ve iş gücünden kayba neden olmaktadır. Dezavantajlar düşünüldüğünde yeni teolojilere yelken açmak zorunlu hale gelmiştir.



Şekil 1.1. Ultrasonik ray muayene cihazı

Bazı durumlarda ölçüm trenlerinin belirleyemeyeceği şekilde anlık durumlar kazalara neden olabilmektedir. Özellikle demiryolu hattında insan, hayvan ve hat üzerine gelmiş engellerin erken tespiti kazaların oluşma riskini azaltır. Bu kazaların engellenebilmesi için izinsiz girişlerin ve engellerin otonom olarak önceden tespit edilebilmesi gerekmektedir. Önceden tespit görevinde literatürde son yıllarda özellikle bilgisayarlı görü uygulamaları araştırmacılar tarafından rağbet görmektedir. Tren yollarındaki yetkisiz giriş ve engel tespiti halen tam otonom olmayan ve çoğunlukla insan odaklı denetim ile yapılmaktadır. On binlerce kilometrelik ray hatlarını insanların denetlemesinin zaman maliyetinin yanı sıra personel yetersizliği gibi sorunları ortaya çıkarmaktadır. Hatlar üzerinde çeşitli bölgelere yerleştirilen sabit kameralar ise sadece bulundukları konumları gözlemleyebilmektedir. Son yıllarda bilgisayarlı görü ve derin öğrenme alanındaki gelişmeler özellikle otonom bir şekilde çalışan sistemlerin geliştirilmesi için önemli bir araç haline gelmiştir. Evrişimli sinir ağları ilk olarak görüntü sınıflandırmada kullanılmasına rağmen sonraki çalışmalarda görüntü bölütleme ve nesne tespiti gibi alanlarda başarılı sonuçlar vermiştir.

Bu tez çalışmasında demiryolu araçları için otonom sistem ile demiryolu çevresinin anlamlandırılması sağlanacaktır. Bu amaçla demiryolu hattının belirlenerek yol boyunca hattın genel durumu hakkında bilgiler elde edilebilecektir. Ayrıca hat üzerinde bulunan engel, insan, araba ve hayvan gibi nesnelerin tespiti gerçekleştirilecektir. Bu işlem İHA üzerindeki kamera ile yapılacaktır.

Tezin temel amacı demiryolu görüntülerini zamandan ve iş gücünden tasarruf yaparak incelemek ve bu görüntülerdeki kusurları yüksek doğruluk oranıyla tespit etmek ve sınıflandırmaktır. Tez çalışmasında kullanılan demiryolu görüntüleri Demiryolu Araştırma ve Teknoloji Merkezinden (DATEM), saha çalışmalarından ve görüntü artırım teknikleriyle elde edilmiştir.

- a) Bu sistem demiryolunun belli bileşenlerini inceleyebilmektedir.
- b) Sistem tek bir ray üzerinde hareket ettiği için her bir ray için ayrı ayrı ölçüm yapılmalıdır.

Yukarıdaki dezavantajlar bize var olan sistemin zaman, iş gücü ve verim açısından eksiklerini göstermektedir. İnsansız hava araçları (İHA) sayesinde demiryolu görüntüleri zaman ve iş gücü kaybı olmadan kayıt altına alınabilmektedir. Tezin ikinci kısmında, öncelikle Gazebo robot deney ortamında oluşturulan Anafi4K İHA'sıyla demiryolu görüntüleri kayıt altına alınmıştır. Bu simülasyon sayesinde daha sonra gerçek dünya ortamında yapılacak çalışmalar daha kolay yapılacaktır. Böylelikle daha sonraki aşamalarda gerçek dünya ortamında yapılacak İHA uçuşları için deney ortamı sağlanmıştır. Tez aşamasında demiryolu görüntüleri kusur tespiti ve sınıflandırması için kullanılacaktır. Derin öğrenme tabanlı sistemin geliştirilmesi sırasında, İHA tarafından toplanan veri seti kullanılacak ve bu veri seti görevler için eğitim amacıyla işlenecektir. Veri seti derin öğrenme tabanlı modellerle eğitilerek demiryolu üzerindeki kusurlar anlamsal bölütleme yöntemiyle tespit edilecektir. Daha sonra yine İHA ile oluşturulmuş bir veri seti derin

öğrenme modelleriyle eğitilecektir. Eğitim sonucu oluşan modelle demiryolu bileşenin eksik parça içerip içermediği tespit edilecektir. Sonraki aşamada ise anlamsal bölümlere yardımıyla dengesiz olan veri seti doğru gözlem yapabilmek için dengelenmeye çalışılacaktır. Sayıca eksik olan kusurlu bileşenlerin miktarı arttırılacaktır. Bazı demiryolu veri setlerinde sınıflandırılacak veya tespit edilecek kusur içeren demiryolu bileşeni görüntü tek halde bulunmamaktadır. Bunun için geliştirilen model öncelikle ilgili bileşeni tespit edip, daha sonra ne tür bir kusur içerdiği tespit edilecektir. Tez çalışmasında, YOLO, Unet, BiSeNetV2 ve kendi geliştirdiğimiz hibrit derin öğrenme modelleri kullanılacaktır.

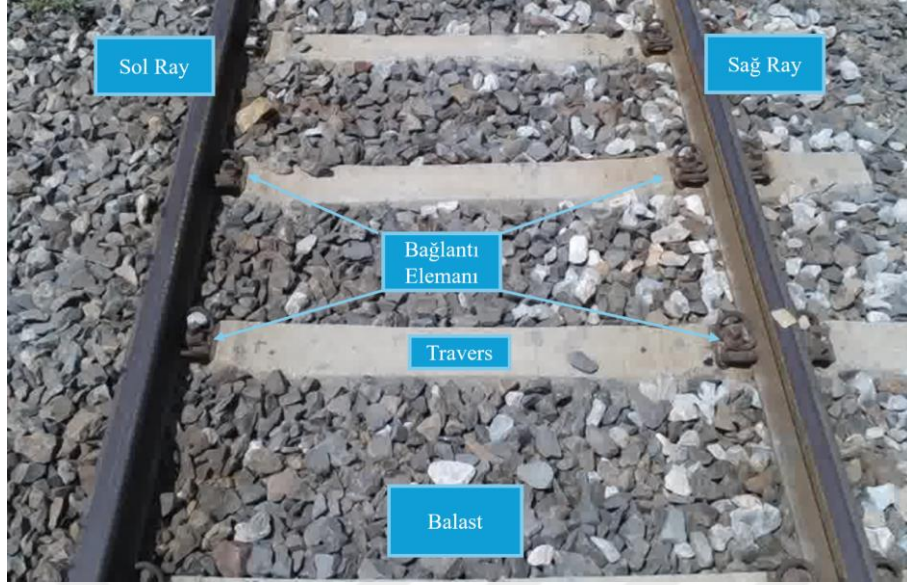
Geleneksel yöntemlerle yapılan demiryolu hattı bakımı maliyetli ve ciddi zaman gerektiren bir işlemdir. Bakım işlemi otonom İHA'larla yapıldığında zaman ve maliyet düşürülürken aynı zamanda İHA'larla elde edilen görüntülere hibrit algoritmalar uygulanarak kusurlu demiryolu bileşenleri yüksek doğruluk oranıyla tespit edilebilmektedir. Bu analizlerden sonra demiryolu alanındaki diğer kuruluşlar gerekli geliştirme ve iyileştirmeleri yaparak kaliteli hizmet sunabileceklerdir.

Daha önce yapılan İHA ile hem gerçek dünya şartlarında hem de simülasyon ortamında otonom demiryolu hattı takibi yapılmamıştır. Bu tez çalışmasında derin öğrenme algoritmaları dahil edilerek demiryolu takibi, bileşen tespiti ve kusur tespiti gerçekleştirilip verimlilik arttırılacaktır. Tez çalışmasının sunduğu avantajlar şunlardır:

- a) Hata tespiti için geleneksel yöntemlerin gerçek zamanlı bir çalışma yöntemi sağlanması.
- b) Simülasyon ortamında test yapılabilmesi.
- c) Derin öğrenme ağları sayesinde birçok farklı kusurun tespit edilebilmesi.
- d) Veri artırım teknikleri sayesinde farklı koşullarda önerilen yöntemin test edilebilmesi.

1.1. Demiryolu Bileşenleri

Demiryolu, metal bir yol üzerinde mekanik bir güçle hareket eden araçlar ile insan ve eşya taşınmasını sağlayan tüm ulaşım tesislerini ifade eder. Elektrikli demiryolları, temiz enerji kullanımının yanı sıra güvenli, konforlu ve ekonomik olmasıyla dikkat çeker; ayrıca yüksek hız desteği sayesinde seyahat sürelerini kısaltması nedeniyle oldukça yaygın tercih edilmektedir. Bugün, demiryolu taşımacılığı, dünyanın en güvenli ulaşım yöntemlerinden biri olarak kabul edilmektedir. Trenlerin güvenli çalışmasını sağlamak için demiryolu hat kalitesi kritik bir unsurdur; bu nedenle hat üzerindeki kusurların incelenmesi günümüzde daha da önem kazanmıştır. Demiryolu araçlarının hareket ettiği paralel metal çubuklara “ray” adı verilir. Raylar birçok bileşenden oluşmakta olup, bu bileşenlerde meydana gelen problemler erken aşamada tespit edilmezse ciddi kazalara neden olabilir. Demiryolu raylarının temel bileşenleri ise Şekil 1.2’de gösterilmiştir.

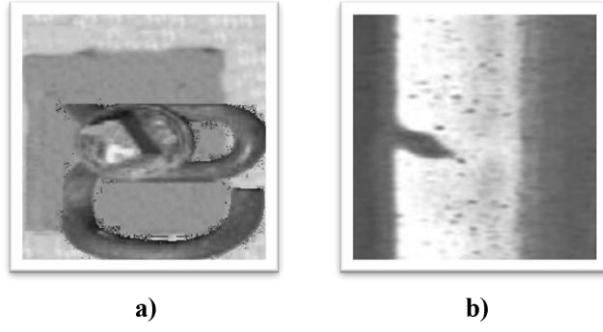


Şekil 1.2. Demiryolu bileşenleri

Şekil 1.2’de, travers, bağlantı elemanları, balast ve rayları içeren bir ray bağlantı sistemi görülmektedir. Özellikle, traversler demiryolu raylarını desteklemek ve sabitlemek, ayrıca yükü raylardan altta yatan balastlara ve alt seviyelere aktarmak için bir taban olarak kullanılır. Bağlantı elemanları iki ray arasındaki travers ile rayları birbirine bağlar ve rayın enlemesine hareketlerini önlemek için çelik çiviler tarafından yerinde tutulur. Bir bağdaki her bağ plakası bir rayı bağladığından, iç ve dış tarafı olmak üzere iki taraf oluşturur. Rayın sabitlenmesi için, ray tabanının alt ve yanlarının etrafını tutacak şekilde bağlantı elemanları yerleştirilir. Bağlantı elemanları rayın iç ve dış tarafında mevcuttur.

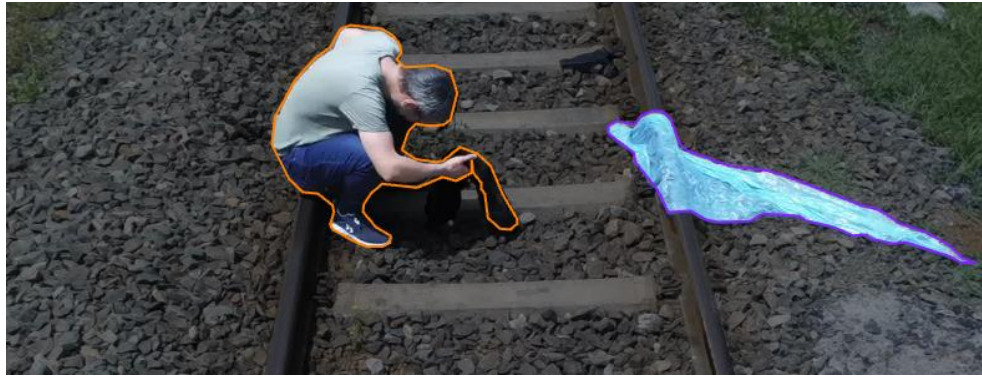
1.2. Demiryolu Bileşenlerindeki Arızalar ve Sorunlar

Demiryolu raylarında oluşan temel arızalar genellikle şunlardır; ray yüzeyindeki çatlaklar, traverslerde oluşan kırılmalar, bağlantı elemanlarının eksikliği veya deforme olması, bağlantı elemanlarının eksikliği veya deforme olması, raylarda oluşan eğilmeler ve ray arası mesafenin değişimi ve traverslerin kaymasıdır. Tez çalışmasında ray yüzeyindeki çatlaklar ve bağlantı elemanlarının eksikliği veya deforme olması ele alınmıştır. Örnek arızalar Şekil 1.3’te görülmektedir.



Şekil 1.3. Örnek demiryolu arızaları (a) kusurlu bağlantı elemanı (b) ray üzerindeki çatlak

Demiryolu ray hattı ve bileşenlerinin tespit edilmesi ve gerçek zamanlı izlenmesi ile literatürde birçok çalışma mevcuttur. Literatürdeki çalışmalarda ray hattı ve bileşenlerinin izlenebilmesi için yabancı nesneler göz ardı edilmektedir. Özellikle ray yüzey ve bileşenlerinin tespit edilmesi ve izlenmesindeki doğruluk oranını yabancı cisimler etkilemektedir. Proje kapsamında yapılan araştırmalar neticesinde ray hattı boyunca çöp ve diğer yabancı nesnelerin bulunduğu gözlemlenmiştir. Özellikle yerleşim yerlerine yakın demiryolu hatlarında yabancı nesneler oldukça fazladır. Bu nesneler ray hattı ve bileşenlerinin tespit edilmesini zorlaştırdığı gibi aynı zamanda demiryolu ulaşım güvenliği içinde tehdit olabilir. Ray hattı üzerinde bulunan yabancı nesneler (insan, ağaç dalları, plastik vb.) trenin raydan çıkmasına neden olabilir. Şekil 1.4'te örnek bir senaryo görülmektedir. Bu problemin giderilmesi, ray hattı üzerindeki yabancı nesnelerin tespit edilebilmesi projenin temel motivasyonunu oluşturmaktadır.

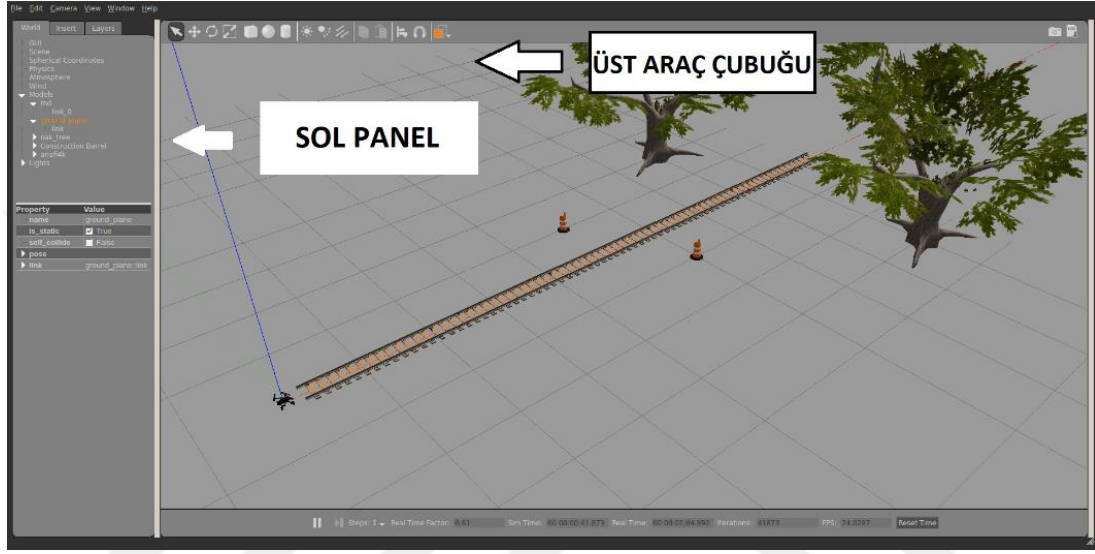


Şekil 1.4. Örnek demiryolu hattı çevresindeki yabancı cisimler

1.3. Gazebo

Gazebo, üç boyutlu iç ve dış ortamlarda robotik ve sensör uygulamalarının simülasyonunu sağlayan bir simülatördür. Bu projede Anafi4K İHA modelini kullanarak 3 boyutlu uçuş simülasyon modelleri oluşturulabilmektedir. Gazebo, ücretsiz ve açık kaynak kodlu, robot

simülasyon ortamıdır. Bir ve birden fazla robotun 3 boyutlu ortamda simüle edilebildiği bir programdır. Gazebo, sensör ekleme ve gerçekleşmesi konusunda daha fazla verimlidir. Gazebo sensörlerinden alınan veriler gerçek dünyadaki verilerle hemen hemen aynıdır. Gazebo çalışma ortamı Şekil 1.5'te görülmektedir. Gazebo ortamında Parrot Anafi İHA modeli kullanılmıştır. Parrot Anafi İHA Olympe yardımıyla otonom bir şekilde oluşturulan ray modelini takip etmiştir. Gazebo simülasyon ortamı Ubuntu işletim sistemi üzerinde çalışmaktadır.



Şekil 1.5. Gazebo çalışma ortamı

1.4. Parrot Anafi4K İHA

Parrot Anafi ekonomik bir İHA'dır, 2018 yılında satışa sunulmuş ve araştırmacılar tarafından mühendislik projelerinde kullanılmaya başlanmıştır. Parrot Anafi İHA 4K HDR kameraya sahiptir. Yüksek menzilli kamera eğim açısı -90 +90 derece arası İHA üzerindeki nesneyi incelemeye izin vermektedir. Parrot Anafi İHA 128 GB'lık bir hafızaya sahiptir. Ayrıca 25 dakika uçuş süresine sahiptir. Şekil 1.6'da projede kullanılan Parrot Anafi İHA görülmektedir.



Şekil 1.6. (a) Gerçek dünya ortamındaki Anafi4K İHA (b) Simülasyon ortamındaki Anafi4K İHA

Olympe, Parrot İHA'lar için oluşturulmuş python denetleyici programlama ara yüzüdür. Simüle edilmiş veya fiziksel İHA'ya bağlanmak, komut göndermek ve mevcut durumu kontrol etmek için kullanılır. Projede hem Gazebo ortamında oluşturulan Parrot Anafi İHA otonom olarak Olympe ortamında hazırlanılan uçuş kodu ile yönetilmiştir. Saha deneylerinde kullanılan fiziksel Parrot Anafi İHA'da aynı şekilde Olympe ortamında hazırlanılan uçuş kodu ile yönetilmiştir. Tablo 1.1'deki İHA'lardan daha yüksek çözünürlüğe sahip olduğu için çalışmada Anafi4K İHA tercih edilmiştir. Anafi4K modelinde bir adet ön kamera bulunmaktadır. Demiryolu hattı bu kamera ile hat takip mantığı ile takip edilmektedir.

Tablo 1.1. Hem Gazebo'da hem de gerçek dünyada kullanılan İHA'lar

İHA	Çözünürlük
Anafi4K	4K
Airborne	480p
Bebop2	1080p
BlueGrass	1080p

1.5. Tezin Amacı

Demiryolu trafiğinin her geçen artmasıyla birlikte demiryolu bakım işlemleri için ayrılan sürelerde azalmaktadır. Günümüz bakım sistemlerinde ray muayene cihazlarının bakım yapabilmesi için demiryolunun trafiğe kapalı olması gerekmektedir. Bu durum bakım hizmetleri için ayrılan zamanın dar olduğunu göstermektedir. Bunun için günümüz İHA teknolojilerinden faydalanabilir. Bunun için bir ön çalışma olarak Gazebo ortamında oluşturulan bir Anafi4K İHA ile gerçek dünya şartlarında demiryolu görüntülerini kaydeden bir İHA simülasyonu tasarlanmıştır. İHA matematiksel metotlar sayesinde demiryolu üzerinde uçuş yaparak ray görüntülerini anlık olarak kaydetmektedir. Demiryollarındaki bir başka sorun ise demiryolu bileşenleri üzerinde oluşan kusurların yüksek performansla tespit edilememesidir. Önerilen derin öğrenme tabanlı modeller sayesinde bu kusurlar yüksek başarımlarında tespit edilmiştir. Günümüzde demiryollarının yaygınlaşması ve hızlı ulaşımın insan hayatında vazgeçilmez olması nedeniyle demiryollarındaki kusurların zamanında tespiti son derece önemli bir konu haline gelmiştir. Bu tez çalışmasında geleneksel yöntemlerle sürdürülen demiryolu bakım ve izleme işlemlerinin otomatik hale getirilmesi amaçlanmıştır. Dinamik olarak İHA ile elde edilecek demiryolu verileri hibrit algoritmalar uygulanarak kusurlu demiryolu bileşenlerinin saptanması, sınıflandırılması hedeflenmektedir. Bu tez çalışması demiryollarında trenlerin güvenli seyredebilmesi için engel ve izinsiz/yetkisiz girişlerin otonom olarak önceden tespit edilmesini amaçlamaktadır. Bu sayede

demiryolu işletmesi kaza potansiyelini önceden fark ederek treni durdurma ya da yavaşlatma gibi eylemleri gerçekleştirecek, can ve mal kayıplarının önüne geçilebilecektir.

Demiryolu ray hatları ve bileşenlerinin tespiti ile gerçek zamanlı izlenmesine yönelik literatürde çok sayıda çalışma mevcuttur. Ancak, bu çalışmalarda ray hatları ve bileşenlerinin izlenmesi sırasında yabancı nesneler genellikle ihmal edilmektedir. Ray yüzeyi ve bileşenlerinin tespit ve izlenmesindeki doğruluk oranı, özellikle yabancı cisimlerin varlığından etkilenmektedir. Tez kapsamında yapılan araştırmalar, ray hattı boyunca çöp ve diğer yabancı nesnelerin bulunduğunu ortaya koymuştur. Bu durum, özellikle yerleşim yerlerine yakın demiryolu hatlarında yaygındır. Yabancı nesneler, ray hatları ve bileşenlerinin tespitini zorlaştırdığı gibi, demiryolu ulaşım güvenliği açısından da ciddi bir tehdit oluşturmaktadır. Ray hattı üzerindeki yabancı cisimler (insan, ağaç dalları vb.), trenin raydan çıkmasına neden olabilecek potansiyele sahiptir. Bu sorunun giderilmesi ve ray hattı üzerindeki yabancı nesnelerin tespit edilmesi, projenin ana motivasyonunu oluşturmaktadır. Projenin temel amaçları;

- a) Demiryolu hattında bulunan yabancı nesnelerin tespit edilmesi.
- b) Veri setinin elde edilip iyileştirilmesi.
- c) Simülasyon ortamında derin öğrenme tabanlı İHA demiryolu takip sisteminin geliştirilmesi.
- d) Simülasyon ortamında geliştirilen yöntemin gerçek dünya şartlarına uygulanması.
- e) Demiryolu bağlantı elemanlarını ayırt eden hibrit derin öğrenme yönteminin gerçekleştirilmesi.
- f) Demiryolu bağlantı elemanlarının içerdiği kusurları tespit eden hibrit yöntemin gerçekleştirilmesi.
- g) Geliştirilen yöntemlerin İHA'ya aktarılması

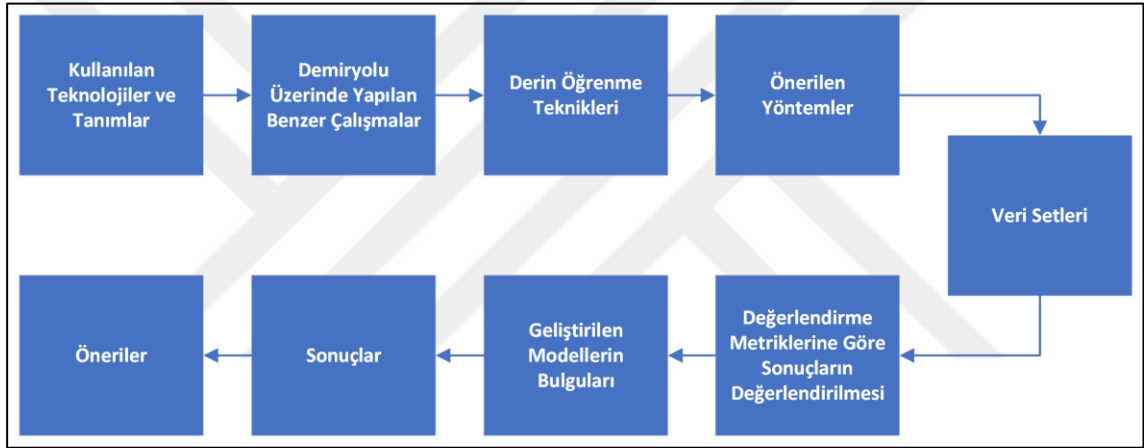
1.6. Karşılaşılan Zorluklar

Demiryolu üzerinde gerçekleştirilen uçuşlar için gerekli izinlerin alınması sürecinde çeşitli zorluklarla karşılaşmıştır. Öncelikle, ilgili kurum ve kuruluşlar arasındaki bürokratik prosedürlerin karmaşıklığı ve süreçlerin uzunluğu, araştırmanın zaman çizelgesini olumsuz etkilemiştir. Ayrıca, güvenlik ve emniyet konularına ilişkin detaylı değerlendirmelerin yapılması gerektiğinden, ek belgeler ve teknik bilgiler talep edilmiş, bu da izin sürecini daha da uzatmıştır. Farklı yetki alanlarına sahip kurumlar arasındaki koordinasyon eksikliği ve iletişim kopuklukları da izin alma sürecini zorlaştıran diğer faktörler arasında yer almıştır. Gerekli izinler temin edilmiş ve araştırma planlandığı şekilde ilerletilmiştir.

1.7. Teze Genel Bakış

Tez çalışmasının ilk bölümünde, çalışmanın konusu ve ele alınan problem özetlenmiştir. İkinci bölümde ise daha önce bu alanda gerçekleştirilen benzer araştırmalara yer verilmiştir.

Üçüncü bölümde, kullanılan derin öğrenme algoritmaları, ilgili teknolojiler ve uygulanan tekniklerin tanımları yapılmıştır. Ayrıca, sınıflandırma sürecinden elde edilen sonuçların nasıl yorumlanacağına dair açıklamalara yer verilmiştir. Dördüncü bölüm, kullanılan veri seti ve uygulanan yöntemleri kapsamaktadır; bu bölümde veri setinin elde edilme süreci detaylandırılmıştır. Beşinci bölümde, derin öğrenme modellerinin elde ettiği sonuçlar üzerinde durulmuştur. Son bölüm ise önerilen yöntemin sonuçlarına ayrılmıştır. Bu tezde izlenen adımlar Şekil 1.7’de gösterilmiştir.



Şekil 1.7. Tezin yapısı

2. LİTERATÜR ÖZETİ

Demiryolları, raylar üzerinde hareket eden araçlarla yolcu ve yük taşımacılığını sağlayan yapılardır. Demiryollarının kullanımının hızla yaygınlaşmasıyla beraber demiryollarındaki sorunların çözümü daha acil hale gelmiştir. Son yıllardaki demiryolu trafiğindeki yolcu sayıları giderek artmıştır. Bu nedenle demiryolu alanında yapılan akademik çalışmalar da giderek artmaktadır. Günümüzde demiryollarındaki kusurları tespit etmek için geleneksel yöntemler kullanılmaktadır. Geleneksel yöntemler genellikle insan gücüne ve basit sistemlere dayanmaktadır. Bu yöntemler zaman ve iş gücü kaybına neden olmaktadır. Bundan dolayı demiryolu taşımacılığında güvenlik sekteye uğrayabilir. Rayların ve bağlantı elemanlarının sağlığı, ray üzerindeki trenlerin güvenli seyahati için çok önemlidir. Tren ve vagon tekerlekleri ile ray arasındaki sürtünme ve çevre koşulları, ray bağlantı elemanlarında arızalara neden olmaktadır. Bağlantı elemanlarının aşınması veya bağlantı elemanının eksik olması durumunda ray ile balast arasındaki bağlantıyı gevşetebilmektedir. Böyle bir durumda trenin geçişi sırasında uygulanan basınç nedeniyle bu kısımda çökmeler veya ciddi arızalar meydana gelebilmektedir. Bu nedenle demiryolu denetimi, kazalara yol açabilecek kusurları tespit etmek için önemli konulardan biri haline gelmiştir. Raylarda kullanıma bağlı olarak çeşitli arıza durumları meydana gelmektedir. Bu arızalar, ray yüzeyindeki kusur, bağlantı elemanlarının eksik olması, traverslerin yanlış konumlandırılması gibi durumlardır. Demiryolu taşımacılığındaki bu gelişmeler, ulaşım güvenliği için yeni denetim sistemlerinin geliştirilmesini gerekli kılmıştır. Demiryolu taşımacılığında güvenliği sağlamak için literatürde birtakım çalışmalar yapılmaktadır. Bu çalışmalar genellikle bilgisayarlı görme tabanlıdır. Derin öğrenme ortaya çıktığında donanımsal eksiklikler ve ilgili veri setlerinin yetersiz olması sebebiyle bir süre durgun süreç geçirse de günümüzde bu eksikliklerin giderilmesi ile popüler olmaya başlamış ve akademik topluluğun dikkatlerini üzerine çekmiştir [4]. Derin öğrenme ağları demiryollarındaki kusurları tespit etmek için büyük veri kümelerine ihtiyaç duymaktadır. Flammini ve ark. [5] demiryollarında alt yapılarda oluşan problemleri belirlemek için IHA tabanlı bir sistem önermişlerdir. Bu amaçla farklı IHA yapılarının temel özellikleri verilerek bu alanda kullanılabilmesi ile ilgili teorik bilgiler verilmiştir. Literatürdeki çalışmalarda raydaki kusurları tespit etmek için video görüntüleri kullanılmıştır. Video görüntülerine farklı Evrişimsel Sinir Ağı (ESA) modelleri uygulanarak geleneksel sığ yöntemlere göre daha iyi sonuçlar elde edilmiştir [6]. Liang ve ark. [7] raylardaki çeşitli kusurları tespit etmek için Segnet mimarisini kullanmışlardır. Çalışmalarında 120 görüntü üzerinde yüksek verim ve doğrulama sonuçları elde edilmiştir. Başka bir çalışmada, Zhang ve ark. [8] RVDS sistemini geliştirmişlerdir. RVDS ile elde edilen görüntülerdeki gürültü giderilmiş ve bir eğrilik filtresinden geçirilmiştir. Geliştirilen Gauss modeli ile kusur tespiti yapılmıştır. Ortalama %92 duyarlılık ve %88.8 duyarlılık performansı elde edilmiştir. Başka bir çalışmada ray üzerindeki kusurların tespitinin yanı sıra tespit edilen kusurun

türü de belirlenmiştir. Bu kusurlar çatlak ve dökülme olarak sınıflandırılmıştır [9]. Bu kusurları tespit etmek için bölge genişliğine ve bölge konumuna dayalı iki yöntem etkin bir şekilde uygulanmıştır. Ray kusurları, görüntü işleme işlemleriyle otomatik olarak hesaplanmış ve çıktısı alınmıştır. Başka bir çalışmada, ray yüzeyindeki kusurları tespit etmek için tam evrişimsel bir ağ modeli geliştirilmiştir. Demiryolu görüntüleri uygun bir boyutta kırpılmış, varyansına ve ortalamasına göre normalize edilmiştir. Bu görüntüleri kullanarak, önerilen model kusur bölgelerini segmentlere ayırmak için kullanılmıştır [10]. Bağlantı elemanlarındaki kusurları tespit etmek için benzerlik tabanlı bir derin ağ modeli önerilmiştir [11]. Önerilen yöntem ile hem bağlantı elemanının lokalizasyonu belirlenir hem de bağlantı elemanlarının sınıflandırılması sağlanmıştır. Bağlantı elemanlarındaki kusurların belirlenmesi için iki farklı yöntem karşılaştırılmıştır [12]. Arızalı bağlantı elemanı tespiti için Dense-SIFT ve evrişimli sinir ağına dayalı iki teknik karşılaştırılmıştır. Sonuçlar, Dense-SIFT tabanlı yaklaşımın evrişimli sinir ağına göre daha iyi sonuçlar verdiğini göstermiştir. Bağlantı elemanlarındaki kusurların tespiti için yapılan bir diğer çalışmada, evrişimli sinir ağı tabanlı bir yöntem önerilmiştir [13]. Önerilen yöntemin ana katkısı, bağlantı elemanını bir bölütleme algoritması ile ray sabitleme civatası ve bağlantı klipsi olarak iki parçaya bölmek ve klipleri tespit etmek için yerel bir evrişimli sinir ağı kullanmak olmuştur. Ray bağlantı elemanları üzerindeki kusurlarını tespit etmeye yönelik çalışmalar bir miktar ilerleme kaydetmiştir. Ancak önerilen yöntemlerin performansı ve bağlantı elemanı yerinin belirlenmesi hala iki önemli sorundur. Kusur tespit teknikleri, özellikle bağlantı elemanı kaybolursa veya kısmen kırılırsa, genellikle yanlış sonuçlar vermektedir. Kusurlu bağlantı elemanlarına ait örnekler bulmak son derece zordur. Bu yönde veri setlerin genişletilmesi kusur tespit yöntemlerinin performansı geliştirecektir.

Demiryolu hatlarındaki kusurların tespit edilmesine yönelik literatürde yer alan derin öğrenme tabanlı çözümler, genellikle video ve resimlerden oluşan veri kümelerine ihtiyaç duymaktadır. Yeni nesil demiryolu kusur tespit sistemleri, teknolojik gelişmelerin sağladığı hız, yüksek doğruluk oranı ve maliyet avantajlarından yararlanmaktadır. Feng ve ark. [14], demiryolu hattındaki eksik bağlantı elemanlarını belirlemek amacıyla otomatik bir görsel inceleme sistemi önermiştir. Bu sistemde, tren vagonunun altına yerleştirilen iki kamera ile her iki rayın görüntüsü kaydedilerek veri toplanmış, ardından bağlantı elemanları tespit edilmiştir. Geliştirdikleri STM modeli ile bağlantı elemanlarını %98 doğrulukla ayırt edebilmişlerdir. Guo ve ark. [15] ise YOLOv4-hibrit modelini kullanarak demiryolu bileşenlerinin tespitini gerçekleştirmiş ve çalışmada genel erişime açık bir veri seti kullanmıştır. Özelleştirilmiş YOLOv4-hibrit modeli ile ortalama %94,4 doğruluk (mAP) ve saniyede 78,7 kare (FPS) performansı elde edilmiştir. Başka bir çalışmada, ray ve bağlantı elemanları YOLOv5 modeli kullanılarak tespit edilmiştir. Zheng ve ark. [16], ray yüzeyindeki ve bağlantı elemanlarındaki kusurları incelemek için RSDD veri setini etiketlemiş ve derin öğrenme ağı ile demiryolu bileşenlerinin görüntü üzerindeki konumlarını

belirlemiştir. Mask R-CNN ve ResNet ağlarını kullanarak ray ve bağlantı elemanlarındaki kusurları tespit etmişlerdir. Ayrıca, Adaboost tabanlı bir yöntemle kırık bağlantı elemanlarının tespit edildiği bir diğer çalışmada, öncelikle demiryolu görüntüsünden bağlantı elemanlarının pozisyonları belirlenmiş ve dört parçaya bölünen görüntüler Adaboost algoritması ile tanınmıştır [17]. Fan ve ark. [18], düşük ışık koşullarında bağlantı elemanlarını başarılı bir şekilde tespit eden ve kusur türünü belirleyen bir yöntem sunmuştur. Yapılan çalışmalar incelendiğinde, ray üzerindeki kusurların tespit edilmesi ve sınıflandırılması için yapay zekâ ve görüntü işleme tabanlı yöntemlerin yaygın olarak kullanıldığı görülmektedir [19-20]. Yanan ve ark. [21], bu bağlamda raylardaki kusurları YOLOv3 modeli ile tespit etmişlerdir.

Günümüzde demiryollarının yaygınlaşması ve hızlı ulaşımın insan hayatında vazgeçilmez olması nedeniyle demiryollarındaki kusurların zamanında tespiti son derece önemli bir konu haline gelmiştir. Bölütleme görüntünün piksel bazlı tahmin işleminin gerçekleştirilmesi ile maskesinin çıkarılmasıdır. Anlamsal ve örnek bölütleme olmak üzere iki tür bölütleme vardır. Anlamsal bölütleme aynı nesnede bulunan piksellerin aynı etikete sahip olmasıdır. Örnek bölütleme ise aynı nesnelerin farklı örneklerinin farklı etiketlenmesidir [22]. Anlamsal bölütleme tabanlı demiryolu hattı ve çevresindeki nesne-tabanlı analizi için bir yöntem önerilmiştir [22]. Önerilen yöntem temelde Unet yaklaşımını kullanmasına rağmen doğrulama oranı ve ortalama IoU gibi metriklerde katkı sağlamaktadır.

Giben ve ark. [23] demiryolu hattı üzerinde kullanılan 10 sınıfa ait materyalleri semantik bölütleme tabanlı bir yöntemle tespit etmişlerdir. Geliştirdikleri DCNN yöntemiyle %93'lük doğruluk oranı yakalamışlardır. Demiryolu güvenliğini sağlamak için demiryolu hattının çevresi periyodik olarak izlenmesi çok önemlidir. Geleneksel görüntü işleme yöntemleri, yalnızca basit arka planın demiryolu hattını incelemektedir. He ve ark. [24] demiryolu sahnesini anlamlandırmak için DSSCNN yöntemini tasarlamışlardır. Tasarladıkları yöntem ESA'dan daha başarılı olmuştur. Bir diğer çalışmada demiryolu hattın çevresindeki demiryolu tesislerini tespit etmek için piksel bazında işlem yapan bir anlamsal bölütleme modeli önerilmiştir [25]. Çalışma sonucunda 0.49'luk ortalama IoU değeri yakalanmıştır. Anlamsal bölütleme sadece demiryolu hattı ve çevresini anlamlandırmak için kullanılmamıştır. Demiryolu üzerindeki anomalileri tespit etmek içinde kullanılmıştır [26]. Ray kusurlarını tespit etmek için Mask R- CNN'ye dayalı yeni bir yüzey kusur tespit ağı önermektedir [27]. Deneysel değerlendirme önerilen modelin önerilen veri kümesi üzerinde %98.70 ortalama kesinliğe (MAP) ulaştığını ve kusur yerini daha doğru bir şekilde bulabildiğini göstermektedir. Li ve ark. [28] ray bağlantı elemanlarının tespit etmek için anlamsal bölütleme tabanlı bir algoritma önerilmiştir. Çalışmada PSPNet ve vektör geometri ölçümlerini birleştirerek ray bağlantı elemanı durumlarını tespit edip sınıflandıran, ray bağlantı elemanı tespiti için anlamsal bölütleme tabanlı bir model önermişlerdir. Çalışma sonucunda %93'lük bir oranla farklı durumlardaki bağlantı elemanı tespiti yapılmıştır. Demiryolu güvenliği için mobil haritalama

yöntemleri de literatürde geliştirilmiştir. Grandio ve ark. [29] demiryolu hattını anlamsal bölütlemeyle dayalı bir yöntemle haritalanmıştır. Çalışmanın metodolojisi demiryolu altyapısından hem doğrusal hem de dakik unsurları bölümlere ayırmak ve dört senaryoda test etmektir. I- 90 km uzunluğundaki demiryolu; II- 2 km uzunluğundaki düşük kaliteli nokta bulutları; III)-400 m uzunluğunda yüksek kaliteli nokta bulutları; IV- 1,4 km uzunluğundaki demiryolu, hava haritalama sistemi ile kayıt altına alınmıştır. En uzun olanı, eğitim ve test için kullanılmış ve %90'dan fazla ortalama doğruluk elde edilmiştir. Diğer senaryolar sadece test için kullanılmıştır. Demiryolu hattı ve çevresindeki nesnelerin analizi için önerilen çalışma bilgisayarlı görüşü alanına girmektedir. Gerçek ortamlardan alınan görüntüler çeşitli ön işlemlerden geçirildikten sonra derin öğrenme tabanlı görüntü bölütleme algoritmalarında kullanılacaktır. Görüntülerde bulunan demiryolu rayı ve ray çevresi anlamsal bölütleme yapabilen derin ağlar ile görsel olarak ayrıştırılacaktır. Demiryolu hatlarında insanlar veya otomobiller gibi nesnelerin izinsiz girişleri olabilmektedir. Bu izinsiz girişler hem canlıların hayatlarını tehdit etmekte hem de tren kazalarına neden olabilmektedir. Özellikle yerleşim yerlerine yakın demiryolu hatlarında yabancı nesneler oldukça fazladır. Bu nesneler demiryolu hattı ve bileşenlerinin tespit edilmesini zorlaştırdığı gibi aynı zamanda demiryolu ulaşım güvenliği içinde tehdit olabilir. Bu nedenle bu izinsiz girişlerin zamanında tespit edilmesi son derece önemlidir.

Literatürde demiryollarının otonom izlenmesi ile ilgili çeşitli çalışmalar yapılmıştır. Bütün demiryolu işletmeleri uygun bir bakım planı uygulamak ister. Genel olarak bakım işlemleri geleneksel yöntemlerle yapılmaktadır. Demir yolu sistemlerinde çalışma esnasında oluşabilecek birçok kusur türü vardır. Güvenli yolculuğun sağlanması için bu bilinen kusur durumlarının izlenmesi ve erken teşhisin sağlanması gerekir [30]. Oluşacak kusurların birçoğu başka arızaları da tetiklemektedir. Örneğin bir raylarda oluşacak ciddi bir kusur sadece raylara zarar vermez aynı zamanda demiryolu hattının demiryolu ağı üzerinde geniş hasara yol açabilir [31].

Demiryolu ulaşımı gibi sürdürülebilir ulaşım yöntemlerine yönelik artan talebi karşılamak için demiryolu altyapısının güvenli, güvenilir ve verimli olması gerekmektedir. Demiryolu ulaşımının günümüzde karşılaştığı temel sorunlardan birisi yüksek trafik yoğunluğunun bakım ihtiyacını artırması ve aynı zamanda bakım faaliyetlerini gerçekleştirmek için zaman çizelgelerindeki boşlukların kullanılabilirliğini azaltmasıdır. İHA'lar, iş verimliliğini ve üretkenliği artırma potansiyelleri açısından son yıllarda ticari olarak birçok sektörde kullanılmıştır. Demiryolu endüstrisi son zamanlarda İHA'ları keşfetmiş ve test etmeye başlamıştır. Araştırma, geleneksel bakım faaliyetlerinin büyük ölçüde verimsiz, fiziksel olarak zorlu ve tehlikeli olan pistlerde manuel olarak yürüyerek ya da pist doluluk gerektiren test/ölçüm araçları kullanılarak gerçekleştirildiğini göstermiştir [32]. İHA kullanımının temel olarak; uzaktan denetim ve işletmeye olanak sağlaması, yol doluluk veya yol ihtiyacı olmadan demiryolu denetimini yapabilmesi tercih edilmesindeki başlıca nedenlerden biridir. İHA'lar aynı zamanda bakım faaliyetlerinin çalışma koşullarını,

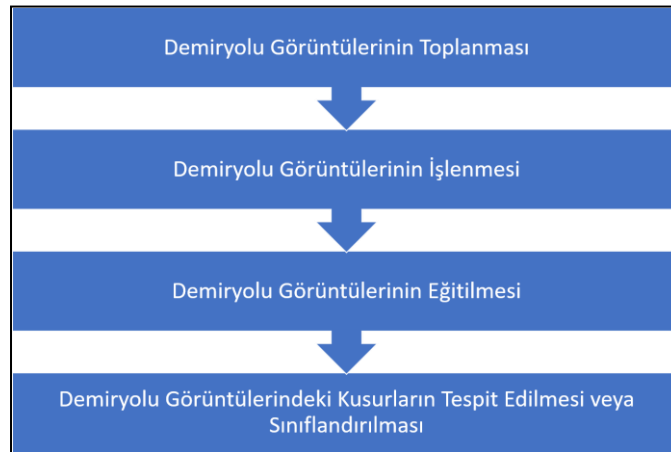
verimliliğini ve kalitesini artırmaktadır. Tez, demiryolu hattını denetlemek için İHA tabanlı potansiyel kullanım senaryosu önermektedir.

Bojarczak ve ark. [33], demiryolu teşhisi için İnsansız Hava Araçlarından (İHA'lar) alınan görüntüleri kullanılmıştır. Ölçüm araçlarıyla yapılan geleneksel çalışmalara kıyasla böyle bir çözümün ana avantajı, demiryolu trafiğini ortadan kaldırılmasıdır. Çalışmada sadece raylardaki tehlikeli bölünmüş kusurların tespiti yapılmıştır. Bu kusurları yaklaşık %81 verimlilik oranıyla tespit etmek için bir algoritma önerilmiştir. Ray başlığını görüntü bölütleme yöntemi ile çıkarmak için Tensorflow ortamında uygulanan FCN-8 derin öğrenme ağı kullanılmıştır. Ray kafasındaki bölünmüş kusurları tespit etmek için bir sezgisel algoritma kullanılmıştır. İHA ile demiryolu analizi için literatürde quadcopters İHA'lar kullanılmıştır. Çalışmada demiryolu denetimi için tamamen otonom bir İHA platformu geliştirilmiştir. Bir PID kontrolörü, bir Parrot Mambo quadrotor üzerindeki demiryoluna bakan kamerayı kullanarak, görüş tabanlı bir yolu etkin bir şekilde takip edecek şekilde İHA'nın durumunu kontrol etmek için tasarlanmıştır. Bilgisayarla görme kullanılarak, İHA konumundan yolun merkez konumuna kadar olan mesafe bulunmuştur. Bu mesafeyi en aza indirmek için sapma kontrolü kullanılması ilgi alanını kameranın görüş alanında tutarak, görüşe dayalı bir yol takibi yöntemini ortaya çıkarmıştır. Demiryoluna bakan kamera aynı zamanda makine öğrenimini kullanarak demiryolundaki kusurları tespit etmiştir [34]. Önerilen bu yöntem yoldaki simüle edilmiş kusurları yaklaşık %90 doğrulukla tespit etmiştir. İHA tabanlı çözümlerin temel avantajı tren trafiğinin engellenmemesi ve optimum kapasitede trenlerin çalışmalarına izin verilmesidir. Fakat yapılan çalışmalardaki temel eksiklik otonom bir şekilde hareket eden bir sistemin olmamasıdır. Bu alanda yapılan bir çalışmada ufuk noktası tespit edilerek otonom bir şekilde ray üzerinde hareket sağlanmıştır [35]. Fakat çalışmada temel görüntü işleme yöntemlerinden olan kenar çıkarım yöntemi ile raylar belirlenmiş ve ray takibi yapılmıştır. Otonom uçuş esnasında ray görüntülerinin alınması ve kusurların belirlenmesi gibi bir yöntem geliştirilmemiştir. İHA'ların özellikle demiryolu hattını meşgul etmemesi ve yüksek kalitede görüntü alabilmeleri bu alanda kullanımlarını arttırmıştır. İHA'lar ile ray yüzey kusurlarının tespiti [36-37], köprü [38] ve tünellerin [39] incelenmesi yapılmıştır.

Literatürde izinsiz/yetkisiz giriş ile ilgili olarak bilgisayar görmesi alanında çeşitli çalışmalar yapılmıştır. Li ve ark. çalışmalarında [40] demiryoluna yapılan izinsiz girişleri tespit etmek için görüntü işleme tekniklerini kullanmışlardır. Dört adımda demiryolundaki anormal durumun göreceli konumunun tespit edildiği çalışmada 35 FPS'lik video görüntüsü üzerinde %96.4 doğruluk oranına ulaşılmıştır. Guo ve ark. çalışmalarında [41] yüksek hızlı demiryollarındaki izinsiz girişlerini tespit etmek için şüpheli kişilerin bölgeler arası takibini gerçekleştirmişlerdir. ResNet-50 ağına kullanıldığı çalışmada bölgeler arası izleme alanındaki doğruluk oranı %88'e ulaşmıştır. Li ve ark. çalışmalarında [42] lokomotif önüne entegre edilmiş bir kamera ile izinsiz girişlerin tespit edilmesini önermişlerdir. Söz konusu çalışmada YOLOv5 algoritması ile izinsiz giriş türü / nesne

tespiti yapılmaya çalışılmıştır. Fast-SCNN algoritması ile demiryolu görüntüsü üzerinde anlamsal bölütleme yapılmıştır. Önerilen model 34 – 90 FPS’lik video görüntülerinde %98.17 doğruluk oranına ulaşmıştır.

Cao ve ark. [43] çalışmalarında karmaşık demiryolu görüntülerindeki izinsiz girişleri tespit etmek için RailDet adını verdikleri bir yöntem önermişlerdir. İki aşamalı yöntemde ilk olarak izinsiz giriş kabul edilecek ray bölgeleri ve izinsiz giriş nesneleri belirlenmektedir. Nesne yasaklı bölgeye girdiğinde yöntemin ikinci aşamasındaki derin öğrenme tabanlı nesne algılama algoritması çalışmaktadır. Önerilen yöntem demiryolu çevresinde belirli noktalara sabitlenmiş kamera görüntüleri üzerinde çalışmaktadır. Yazarlar bu sistemin demiryolları dışındaki sektörlerde de sabit kameralı gözlem noktalarında kullanılabileceğini ifade etmektedirler. Ye ve ark. [44] çalışmalarında demiryollarındaki yabancı nesneleri tespit etmek için, diferansiyel özellik füzyon evrişimli sinir ağı (DFF-Net) kullanan bir nesne algılama yöntemi önerilmiştir. DFF-Net, önceki nesne algılama modülü ve nesne algılama modülü olmak üzere iki modül içerir. Bu çalışmada da özellikle küçük nesneler için algılama performansını artırmak amaçlanmıştır. Yundong ve ark. [45] çalışmalarında İHA gözetiminin demiryolu sahnesine küçük nesne algılamaya dayalı çok bloklu Tek Atışlı Çok Kutulu Dedektör (SSD) yöntemi önerilmiştir. Küçük nesne algılama sınırlamasını ele almak için üç adımdan oluşan çok bloklu bir SSD mekanizması tasarlanmıştır. Mauri ve ark. [46] çalışmalarında hem karayolu hem de demiryolu ortamları için YOLOv5’e dayalı yeni bir tek aşamalı gerçek zamanlı 3D nesne algılama evrişimli sinir ağı (CNN) sunulmuştur. Literatürde yapılan çalışmalar genelde Şekil 2.1’deki adımlardan oluşmaktadır.



Şekil 2.1. Demiryollarındaki kusurların tespiti ve sınıflandırılması

3. TEMEL BİLGİLER

Bu bölümde tez çalışmasında uygulanan yöntemler ve algoritmalar özetlenmiştir. Çalışmanın anlaşılabilirliğini artırmak için, yöntemlerin detaylı matematiksel modelleri ve algoritmaları verilmeden, temel düzeyde bir açıklama yapılmıştır.

3.1. Derin Öğrenme

Derin öğrenme yapay zekâ alanında kullanılan makine öğrenmesi türüdür. Şekil 3.1’de yapay zekâ, makine öğrenmesi ve derin öğrenme arasındaki ilişki görülmektedir



Şekil 3.1. Derin Öğrenme yapay zekâ ilişkisi

Makine öğrenmesi, bilgisayarların matematiksel işlemleri kullanarak verilerden anlamlı sonuçlar çıkarmasını ve bu sonuçlar doğrultusunda tahminlerde bulunmasını sağlayan bir modelleme yöntemidir [47]. İnsanlar, karar alma süreçlerini hızlandırmak ve otomatikleştirmek amacıyla yapay zekâ temelli araçlar ve özellikle makine öğrenmesi algoritmalarından faydalanmıştır. Ancak, yıllar içinde ortaya çıkan yeni gereksinimler, geliştirilen bu algoritmaların bazı durumlarda yeterli yanıt verememesine yol açmıştır. Makine öğrenme algoritmalarının karşılaştıkları zorluklar aşağıdaki gibi verilebilir.

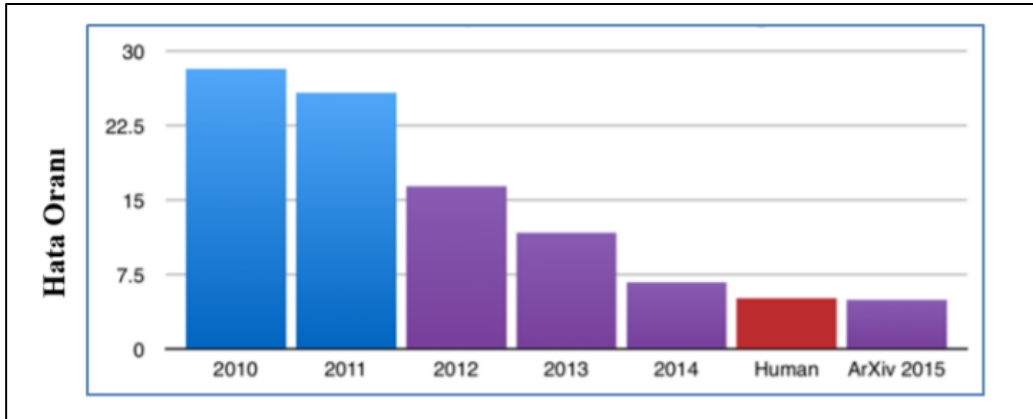
- a) Yetersiz veriler: Makine öğrenme algoritmasının düzgün çalışabilmesi çok fazla miktarda veri gerekmektedir. Söz konusu görüntü ve ses verisi olduğunda ise

milyonlarca veri gerekmektedir. Dengesiz veri setleri de makine öğrenmesinde karşılaşılan zorluluktan birisidir.

- b) Özellik çıkarımı: Bir problemi çözmek için gerekli adımlardan birisi olan özellik çıkarımı makine öğrenmesinde problemde problemde değişmektedir. X problemde kullanılan özellik çıkarımı Y problemde kullanılmaz.
- c) Aşırı uyum ve yetersiz uyum: Aşırı uyum, makine öğrenmesi eğitimi aşamasında verilerden ezberleme durumunu göstermektedir. Yetersiz uyum ise algoritmanın öğrenememesi durumudur.

Derin Öğrenme makine öğrenmesinin bir alt dalı olup makine öğrenmesinin aksine problemi tek katmanda ele almayıp aksine problemi çok katmanlı yapısında çözmeyi hedefler. Günümüzdeki görüntü işleme alanında oldukça popüler olan derin öğrenmenin tarihi 1943 McCulloch ve Pitts'e dayanmaktadır [48]. Derin öğrenme, yapay sinir ağlarına ve insan beynindeki nöronların işlevlerini taklit eden küçük sistemler olarak ortaya çıkmıştır. 2006 yılına kadar geliştirilen YSA'larında kullanılan katmanlar; giriş, gizli ve çıktı katmanlarıyla sınırlı kalmıştır. Buna neden olarak yetersiz donanım teknolojisi gösterilebilir. 2006 yılından sonra çok katmanlı ağların nasıl tasarlanacağı ve nasıl eğitileceği Hinton ve arkadaşları tarafından ispatlanmış [49] ve bu yeni nesil YSA'lara derin ağ (Deep Net) adı verilmiştir. Bu alanda yapılan tüm çalışmalar derin öğrenme (Deep Learning) başlığı altında toplanmıştır. Zamanla gelişen GPU'ların hesaplama gücü derin öğrenme kullanımının artmasına sebep olmuştur. Büyük miktardaki veri (görüntü, ses) böylelikle daha kısa sürelerde eğitilmiş ve test edilmiştir.

2010 yılından beri Google tarafından düzenlenen ImageNet yarışmalarıyla derin öğrenme alanına ciddi katkı sağlanmakla beraber performans oranları geliştirilmiş hata oranları ise azaltılmıştır. Bu yarışmayla birlikte derin öğrenmedeki hata oranları %3,62'ye kadar düşmüştür. Şekil 3.2'de elde edilen en iyi hata oranları görülmektedir.



Şekil 3.2. Yıllara göre ImageNet yarışmasında elde edilen en iyi hata oranları

Derin öğrenmenin hızla popüler olmasında CNN, VGG, ResNet gibi çok katmanlı mimarilerin hızla gelişmesi etken olmuştur. Ayrıca büyük veri setlerine erişim kolaylığı da yadsınamaz bir etkidir. Yeterli veriye erişmenin mümkün olmadığı durumlarda veri arttırma teknikleri ile veri arttırma yaygınlaşmıştır.

3.2. Derin Öğrenme Modelleri

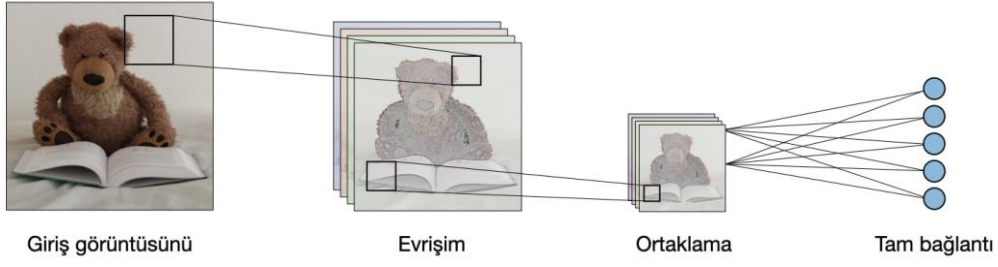
Derin öğrenmede görüntü, ses, metin içerikli problemleri çözmek için farklı mimariler geliştirilmiştir. Bu mimarilere genel olarak derin öğrenme modelleri denmektedir. Akademik çalışmalarda kullanılan bazı modeller şunlardır:

- a) CNN: Genellikle görüntü ve video sınıflandırmada kullanılır.
- b) RNN: Metin bazlı problemlerde kullanılır.
- c) LSTM: Metin bazlı problemlerde kullanılır.
- d) R-CNN: Nesne tespiti için kullanılır.
- e) Fast R-CNN: Nesne tespiti için kullanılır.
- f) Faster R-CNN: Nesne tespiti için kullanılır.

3.2.1. Evrişimli Sinir Ağları (CNN)

Evrişimli sinir ağları, literatürde ESA olarak bilinmektedir ve bilgisayarlı görme alanında yaygın olarak kullanılan bir derin öğrenme modelidir. CNN algoritmasının yapısı, hayvanların görsel işleme mekanizmasından esinlenilerek geliştirilmiştir [50]. Görüntüleri doğrudan girdi olarak alabilen CNN yapıları, bilgisayarla görme alanında etkili bir şekilde kullanılmaktadır ve bir veya daha fazla evrişimli katmandan oluşmaktadır [51]. CNN'in katmanları arasında; giriş katmanı, konvolüsyon katmanı, yığın normalleştirme (batch normalization), düzleştirme (flatten), havuzlama (pooling), tam bağlantılı katman (fully connected veya dense) ve düğüm seyreltme (dropout) katmanı bulunmaktadır. CNN, girdi verilerini aldıktan sonra katmanlar arasında işlemler gerçekleştirilerek eğitim sürecini ilerletir [52]. Eğitim sonucunda elde edilen çıktı ile doğru sonuç karşılaştırılır ve ortaya çıkan hata farkını minimize etmek için geriye yayılım algoritması kullanılır. Her iterasyonda ağırlıklar güncellenerek hatanın azaltılması ve sınıflandırma başarısının artması sağlanır [52].

Özetle CNN modellerinde 3 ana katman bulunmaktadır. Bu katmanlar; konvolüsyon katmanı, havuzlama katmanı ve tam bağlı (fully connected) katmanıdır. Şekil 3.3'te örnek bir CNN modelinin adımları görülmektedir.



Şekil 3.3. Örnek CNN mimarisi

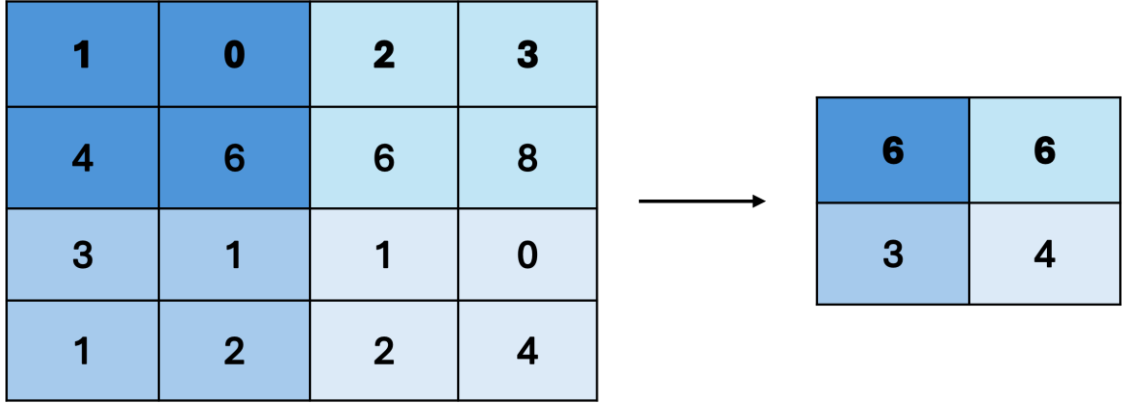
3.2.1.1. Konvolüsyon Katmanı

Konvolüsyon katmanı, bir CNN mimarisinin en temel katmandır ve özellik çıkarma ve doğrusal olmayan işleme için kullanılır. CNN'nin ilk katmanlarında bir konvolüsyon katmanı kullanıldığında, giriş görüntüsünden kenarlar, çizgiler ve köşeler gibi düşük anlamlı özellikleri çıkarılır. Konvolüsyon katmanı ağıın uçlarına doğru kullanıldığında, görüntüdeki bir nesnenin görünürlüğü ve şekli ortaya çıkar. Konvolüsyon, genellikle bir filtreyi görüntü üzerinde hareket ettirerek merkez piksel değerlerinin hesaplanmasıdır. Bir görüntü üzerinde w süzgeci ile evrişim Denklem 3.1'de verilmiştir.

$$C(w * I)_{x,y} = \sum_{s=-a}^a \sum_{t=-b}^b w(s,t) f(x+s, y+t) \quad (3.1)$$

3.2.1.2. Havuzlama Katmanı

Havuzlama katmanları genelde bir konvolüsyonel ağ mimarisinde konvolüsyonel katmanların arasına yerleştirilir. Havuzlama katmanı sayesinde hem parametre sayısı düşürülmekte hem de sistemin aşırı öğrenmesi (overfitting) engellenmektedir. Aynı zamanda boyutu azaltarak hesaplama yükünü hafifletmektedir. Boyutu azaltma işlemini gerçekleştirdiğinde doğal olarak veri kaybı da gerçekleşmiş olur. Bu sebeple bu katmanda filtreler seçilirken en az kayıp için küçük boyutlu filtreler seçilmektedir. Havuzlama işlemi sırasında maksimum, ortalama, toplama ve L2-norm gibi çeşitli yöntemler kullanılabilir. Bu yöntemler arasında en yüksek başarı oranına sahip olan ise maksimum havuzlama yöntemidir. Maksimum havuzlama, belirli bir filtre boyutu ve adım aralığında, seçilen bölgede yer alan piksellerin en yüksek değerini seçerek gerçekleştirilir. Şekil 3.4'te görüldüğü üzere, 2x2 boyutundaki bir filtre uygulanırken, bu filtre içerisindeki en büyük piksel değeri aktarılmaktadır [53].



Şekil 3.4. Maksimum havuzlama

3.2.1.3. Tam Bağlantılı Katman (Fully Connected Layer)

Tam bağlantılı katman, ileri beslemeli sinir ağlarına benzer ve birkaç tam bağlı katman, evrişimli bir sinir ağının uçlarına doğru yerleştirilir. Tam bağlantılı katman, önceki katmanlardan bilgi toplayarak yüksek düzeyde soyutlama sağlamaktadır. Burada çok fazla parametre olduğu için aşırı uyum (overfitting) sorununa yol açmaktadır. Bu nedenle bazı bağlantıları keser. Tam bağlantı katmanında, giriş olarak alınan tüm özellikler doğrusal bir kombinasyon haline getirilir ve ardından aktivasyon fonksiyonu ile işlenmektedir.

3.2.1.4. Yığın Normalleştirme Katmanı (Batch Normalization Layer)

CNN'nin eğitim süresini azaltmak için yığın normalleştirme katmanı kullanılır. Batch normalization, ağıdaki katmanların önceki katmanın öğrenme sürecini beklemeden, paralel olarak öğrenmelerine olanak tanır. Bu durum, modelin eğitim sürecinin daha hızlı gerçekleşmesini sağlamaktadır.

3.2.1.5. Düşüm Seyreltme Katmanı (Dropout Katmanı)

Büyük verilerle yapılan işlemler sonucunda derin öğrenme ağları bazen ezberleme yapabilmektedir. Dropout katmanı, derin öğrenme modellerinde aşırı uyum (overfitting) probleminin önüne geçmek amacıyla kullanılan önemli bir düzenleme tekniğidir. Bu yöntem, modelin eğitim sürecinde bazı nöronların rastgele devre dışı bırakılması yoluyla ağı daha genel bir şekilde öğrenmesini sağlamayı amaçlar. Eğitim sırasında her iterasyonda belirli bir dropout oranına bağlı olarak nöronlar rastgele seçilerek etkinlikleri sıfırlanır ve bu nöronlardan çıkan sinyallerin ileri aktarımı engellenir. Bu işlem, modelin nöronlar arasında gereksiz bağımlılıklar geliştirmesini önler ve her nöronun daha anlamlı ve bağımsız özellikler öğrenmesine katkıda bulunur.

Dropout'un uygulandığı süreçte, eğitim sırasında rastgele devre dışı bırakılan nöronların etkisini dengelemek amacıyla test aşamasında ağırlıklar ölçeklenir. Böylece, eğitim esnasında sağlanan rastgeleliğin modele eklediği esneklik korunurken, test aşamasında tam ağ yapısı kullanılarak tahminler gerçekleştirilebilir. Bu teknik genellikle tam bağlantılı katmanlarda yaygın bir şekilde uygulanmakla birlikte, tekrarlayan sinir ağları (RNN) ve evrişimli sinir ağları (CNN) gibi daha karmaşık yapılar üzerinde de uygun modifikasyonlarla kullanılabilir. Eğitim sürecinde etkin olan dropout işlemi, test aşamasında devre dışı bırakılarak modelin genelleme performansının iyileştirilmesi amaçlanmaktadır.

3.2.1.6. Aktivasyon Fonksiyonları

Aktivasyon fonksiyonları, yapay sinir ağlarında nöronların çıktılarını belirleyen ve modelin doğrusal olmayan ilişkileri öğrenebilmesine imkân tanıyan matematiksel fonksiyonlardır. Sinir ağlarında her nöron, girdi değerlerinin ilgili ağırlıklarla çarpılması ve önyargının eklenmesi sonucunda elde edilen toplamı, aktivasyon fonksiyonuna tabi tutar. Bu süreç, ağın daha karmaşık ilişkileri öğrenebilmesi amacıyla doğrusal olmayan bir dönüşüm gerçekleştirmeyi hedeflemektedir.

Sigmoid Fonksiyonu: Çıkışı 0 ile 1 arasında bir değere dönüştürür, bu nedenle olasılık hesaplamaları için uygundur. Denklem 3.2'deki gibi tanımlanmaktadır.

$$\text{sigmoid}(x) = \frac{1}{1+e^{-x}} \quad (3.2)$$

Relu (Rectified Linear Unit) Fonksiyonu: Girdiyi pozitif olduğu sürece aynı tutar, negatif girdiler için ise sıfıra eşitlemektedir. Relu, gradyan sönmesi problemini azaltır ve hesaplaması kolay olduğu için genellikle gizli katmanlarda kullanılmaktadır. Denklem 3.3'teki gibi tanımlanmaktadır.

$$\text{Relu}(x) = \max(0, x) \quad (3.3)$$

Softmax Fonksiyonu: Genellikle çok sınıflı sınıflandırma problemlerinde, ağın son katmanında kullanılmaktadır. Softmax işlemi Denklem 3.4'te görülmektedir. Her $z^{(i)}$ girdisi için Softmax değeri hesaplanmaktadır.

$$\sigma_{\text{softmax}}(z^{(i)}) = \frac{e^{z_k^{(i)}}}{\sum_{k=1}^c e^{z_k^{(i)}}} \quad (3.4)$$

Tanh Fonksiyonu: Çıkışı -1 ile 1 arasında olan sigmoid benzeri bir fonksiyondur. Denklem 3.5'te hesaplaması görülmektedir.

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (3.5)$$

3.2.2. Diğer CNN Modelleri

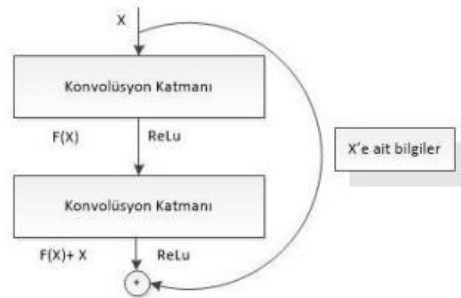
Tez çalışmasında farklı CNN modelleri de kullanılmıştır.

3.2.2.1. VGG-16

Adından da anlaşılacağı üzere, VGG-16 modelinin yapısı genellikle 16 katmandan oluşur. VGG-16, 2014'te düzenlenen ImageNet Challenge'da büyük başarılar elde eden bir derin öğrenme modelidir [54]. Evrişim katmanlarında küçük boyutlu filtreler (3x3) kullanılmıştır [54]. Bu model, 13 evrişim katmanı ve 3 tam bağlantılı katmandan meydana gelir. Ayrıca, 4x4 boyutunda 5 havuzlama katmanı da bulunmaktadır. Son katman, Softmax katmanı olup, bu katman giriş verilerini sınıflandırma amacıyla kullanılır. Aktivasyon fonksiyonu olarak ise genellikle Relu tercih edilmektedir.

3.2.2.2. ResNet

ResNet (Residual Networks), derin öğrenme mimarilerinde bir devrim niteliği taşıyan ve 2015 yılında He ve ark. [55] tarafından önerilen bir modeldir. Bu ağlar, derin ağlarda yaygın olarak karşılaşılan gradyan sönümlenmesi (vanishing gradient) problemini çözmek amacıyla geliştirilmiş olup, "skip connection" adı verilen ve katmanlar arasında atlamalı bağlantılar sağlayan bir mekanizma kullanır. Bu sayede, çok daha derin ağların eğitilmesi mümkün hale gelmiştir. ResNet50, bu mimarinin 50 katmandan oluşan versiyonudur ve daha derin ağlar arasında hesaplama maliyeti açısından iyi bir denge kurar. ResNet50, özellikle görüntü sınıflandırma, nesne tespiti ve bölütleme gibi bilgisayarla görme uygulamalarında yüksek performans göstermekte olup, yaygın olarak kullanılan ImageNet veri kümesinde önemli başarılar elde etmiştir. Model hem akademik hem de endüstriyel uygulamalarda derin öğrenme temelli çözümler için sıklıkla tercih edilen bir yapıdadır. En çok tercih edilen varyasyonlardan biri, 49 evrişimli katman ve tam bağlantılı bir katmandan oluşan ResNet50'dir. Şekil 3.5'te, ResNet'in atlamalı bağlantısı ve bu bağlantının diğer katmana $F(x) + x$ formunda iletimi gösterilmektedir [53].



Şekil 3.5. ResNet artık blok

3.2.2.3. InceptionV3

GoogleNet modeli geliştirilerek InceptionV3 modeli oluşturulmuştur. Bu modelin hesaplama maliyeti VGG16'dan daha düşüktür. InceptionV3, Google tarafından geliştirilmiş derin öğrenme modelidir. 2014 yılında, ImageNet yarışmasında yaklaşık 1,2 milyon görüntü ile eğitilerek başarılı olduğunu kanıtlamıştır. Inceptionv3 modeli basit bir evrişimli sinir ağı modeline göre daha fazla katmana sahiptir [56]. Bu mimaride, ardışık yapıda konvolüsyon ve havuzlama katmanları üst üste gelmektedir. Bu nedenle, hafıza açısından oldukça güçlü bir performans ortaya koymuştur. Inceptionv3 mimarisi, diğer modellerden farklı olarak Inception katmanı adı verilen özel bir katmana sahiptir. Bu katman, önceki katmandan gelen girdiyi farklı boyutlardaki evrişim ve birleştirme işlemlerine tabi tutarak yeniden tek bir vektör halinde birleştirir. Bu yöntem sayesinde iç katmanlar, görseldeki nesnelerin boyutlarına uygun filtreleri kullanarak farklı boyutlardaki şekilleri tanımlayabilir [57]. InceptionV3 katmanının temel yapısı Şekil 3.6'da gösterilmektedir.



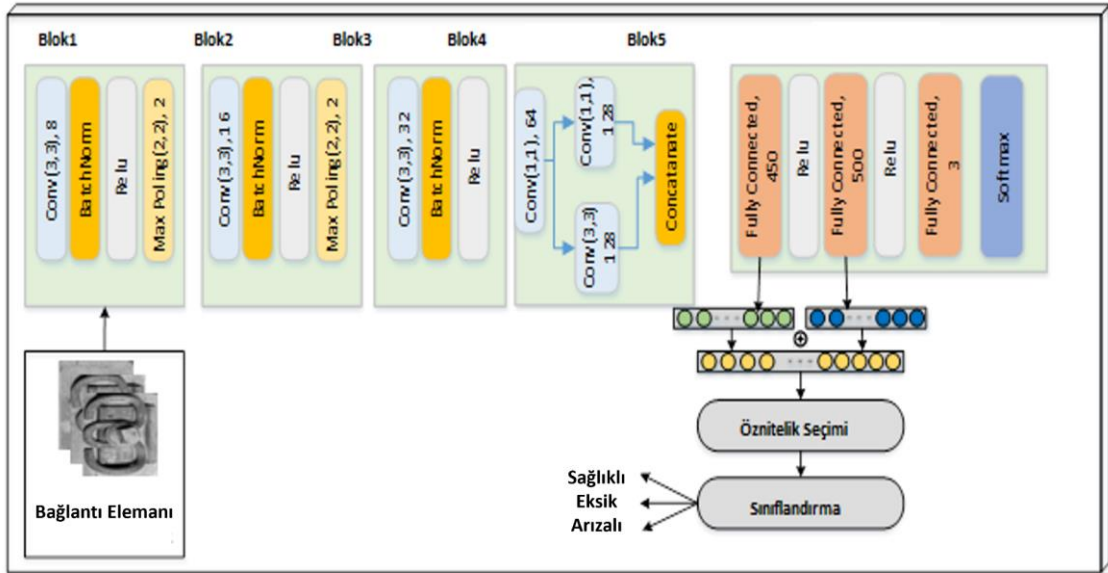
Şekil 3.6. InceptionV3 ağı [57]

5x5 ve 7x7 boyutlarında filtreler yerine, onları karşılamak için iki veya üç adet 3x3 boyutunda filtre kullanılmaktadır.

3.2.2.4. Hafif Evrişimli Sinir Ağı (Lightweight Convolutional Neural Network)

CNN'ler farklı katmanlardan oluşan çok aşamalı bir mimarilerdir. Bu çerçevede, her aşama, özellik haritaları olarak bilinen çok kanallı matrislerde ifade edilen tensörlerden oluşmaktadır. Görüntü sınıflandırma uygulamalarında, evrişimli bir sinir ağının giriş katmanı genellikle 2 boyutlu ve üç kanallı bir görüntüdür. Evrişimsel bir sinir ağı üç katmandan oluşmaktadır. Bu katmanlar

genellikle konvolüsyon, havuzlama ve tam bağlantılı katmanlardır. Evrişimli bir sinir ağının derinliği arttıkça, daha soyut özellikler elde edilir ve sınıflandırma performansı artmasına rağmen ağın eğitimi yavaşlar. Evrişimli bir sinir ağında, tipik olarak birkaç evrişim ve havuzlama katmanı bulunur ve ardından tamamen bağlı katmanlar gelmektedir. Şekil 3.7’de hafif evrişimli sinir ağı mimarisi görülmektedir. Bu LCNN, yüksek sınıflandırma doğruluğunu korurken karmaşıklığı en aza indirmek için tasarlanmıştır. Tipik CNN mimarilerinden daha az katmana sahiptir, bu da onu hesaplama açısından verimli ve gerçek zamanlı uygulamalar için uygun hale getirir. Ağ, evrişimli, toplu normalizasyon, ReLU ve maksimum havuzlama katmanlarını içerir. İki tam bağlı katmandan çıkarılan özellikler, makine öğrenimi sınıflandırıcıları kullanılarak birleştirilir ve işlenir, en iyi sonuçlar Kübik SVM kullanılarak elde edilir [58].



Şekil 3.7. Hafif evrişimli sinir ağı mimarisi

3.2.2.5. YOLO

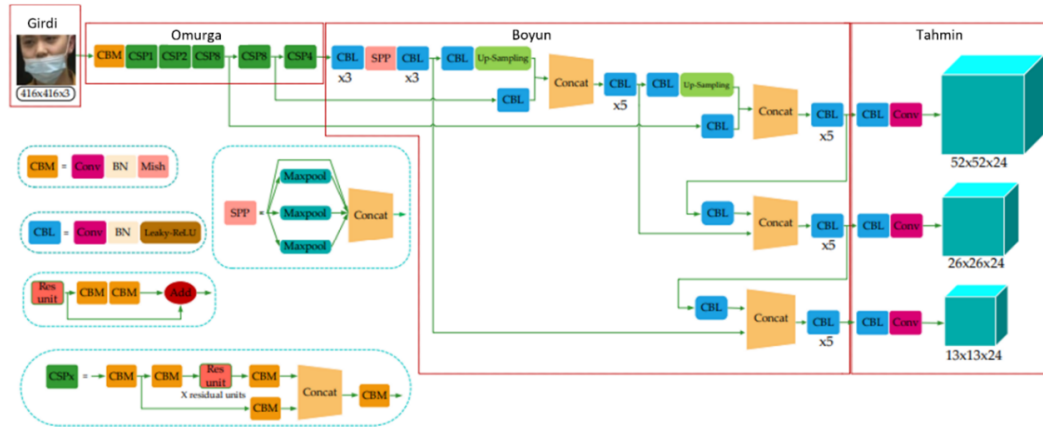
YOLO (You Only Look Once), sınırlayıcı kutular ve sınıf olasılıklarını tahmin etmek için tek bir sinir ağı kullanan ve GoogLeNet’ten esinlenmiş bir derin evrişimsel sinir ağıdır. Bu yaklaşım, görüntünün yalnızca bir kez ağdan geçmesiyle algılama işlemini tamamlamayı mümkün kılar. YOLO, literatürdeki en hızlı genel amaçlı nesne algılama mimarisi olarak kabul edilir ve aynı zamanda bildirilen ilk gerçek zamanlı evrişimsel sinir ağı tabanlı nesne algılama modelidir. Tüm algılama işlemi tek bir ağda gerçekleştirildiği için, doğrudan algılama performansı diğer birçok sinir ağı mimarisine göre daha yüksektir [59]. Şekil 3.8’de, YOLO’nun tespit sisteminin yapısı gösterilmektedir.



Şekil 3.8. Nesne tespitten sorumlu YOLO tespit sistemi [60]

YOLO mimarisi, görüntü sınıflandırma için kullanılan GoogLeNet modelinden ilham almıştır. Bu sinir ağı modeli, 24 evrişim katmanına ek olarak 2 tamamen bağlı katmana sahiptir. Evrişim katmanları, görüntüden özellik çıkarma görevini üstlenirken, tamamen bağlantılı katmanlar ise sınıf olasılıklarını ve koordinatları tahmin eder. YOLO, C ve CUDA ile yazılmış açık kaynaklı bir sinir ağı çerçevesi olan 'Darknet' üzerinde çalıştırılmaktadır. Bu yaklaşım sayesinde, gerçek zamanlı video akışının işlenmesi için GPU'nun hesaplama gücünden faydalanılmaktadır.

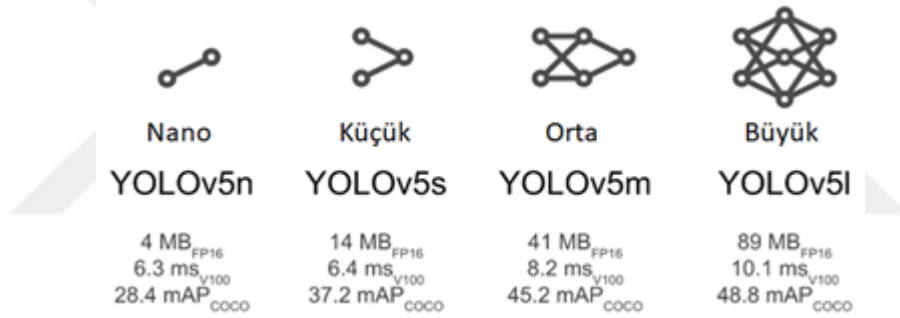
YOLOv4, Nisan 2020'de yayınlanmış ve COCO veri setinde üstün performans gösteren bir gerçek zamanlı nesne algılama modelidir. YOLOv3 temel alınarak optimize edilen bu model, Şekil 3.9'da gösterildiği gibi üç ana bölümden oluşur: omurga, boyun ve tahmin. Omurga kısmında CSPDarknet53 ağı yer almakta olup, veri setini zenginleştirmek için BoF (Bag of Freebies) ve BoS (Bag of Specials) adı verilen iki ek bölüm kullanılır [61]. Boyun kısmı ise farklı ölçeklerde nesneleri algılamaya yönelik olarak tasarlanmış olup, aşağıdan-yukarı ve yukarıdan-aşağı akıştan gelen özellik haritaları, kafa bölümünü beslemeden önce eleman bazında veya birleştirilerek eklenir [61]. Boyun yapısı, Uzamsal Piramit Havuzu (SPP) ve Yol Toplama Ağı (PANet) tekniklerini içerir. Tahmin katmanı ise kafa bölümünde yer almaktadır.



Şekil 3.9. YOLOv4 mimarisi

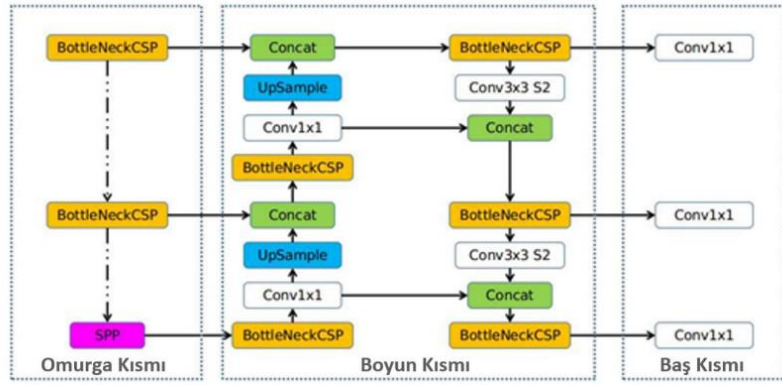
YOLOv4'ün yayınlanmasının ardından kısa bir süre sonra, Glenn ve ekibi mimarinin yeni sürümü olan YOLOv5'i tanıtmıştır [62]. Bu yeni sürüm, önceki versiyonlara göre işlem süresini önemli ölçüde azaltarak dikkat çekmiş ve Pytorch tabanlı yeni bir eğitim ortamında derlenerek eğitim sürecini Darknet'e kıyasla daha kullanıcı dostu hale getirmiştir. YOLOv5'in algılama doğruluğu ve hızı, önceki sürümlere göre belirgin bir şekilde artış göstermiştir. Ayrıca, ağ modelinin ağırlık dosyası boyutları YOLOv4'e kıyasla yaklaşık %90 daha küçüktür.

YOLOv5 modeli, gömülü cihazlarda gerçek zamanlı tespit uygulamaları için ideal bir çözüm olduğunu kanıtlayan pek çok avantaja sahiptir. Modelin yapısı, boyut ve model parametrelerinin sayısına göre ölçeklenen dört farklı alt mimariden oluşur: YOLOv5s, YOLOv5n, YOLOv5m ve YOLOv5l [63]. Bu alt mimariler, sinir ağındaki belirli bir noktada yer alan özellik çıkarma modüllerinin ve kullanılan evrişim çekirdeklerinin sayısındaki farklılıklar ile birbirlerinden ayrılır. Ayrıca, modelin boyutu ve parametre sayısı bu alt mimarilere göre artış göstermektedir. Bu artış, Şekil 3.10'da gösterilmektedir.



Şekil 3.10. YOLOv5 mimarisinin alt mimarileri

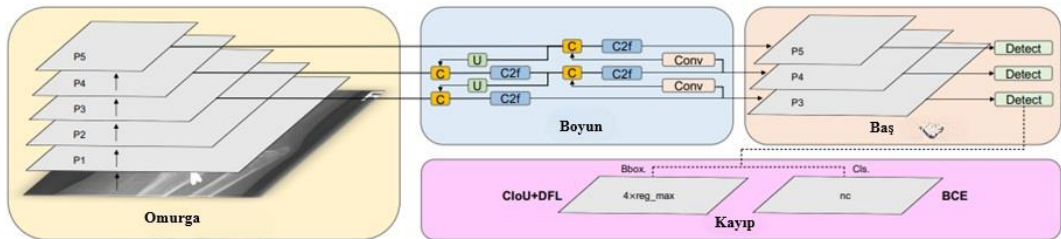
Şekil 3.11'de gösterilen YOLOv5 mimarisi üç ana bileşenden oluşur: omurga, boyun ve baş. Omurga bölümünde CSPDarknet [64], boyun bölümünde PANet [64], baş bölümünde ise YOLO katmanı yer almaktadır. Bu süreçte, veriler önce CSPDarknet üzerinden geçerek özellik çıkarımı yapılır, ardından PANet ile bu özellikler birleştirilir. Son olarak, YOLO katmanı sınıf, skor, konum ve boyut gibi çıktıları sağlamaktadır.



Şekil 3.11. YOLOv5 mimarisinin genel yapısı [64]

YOLOv6 modeli, Haziran 2022’de Meituan tarafından piyasaya sunulmuştur ve COCO veri kümesi üzerinde yapılan karşılaştırmalarda başarılı sonuçlar elde etmiştir. YOLOv6, YOLO mimarisi üzerine inşa edilmiş olup, önceki YOLO modellerine kıyasla çeşitli iyileştirmeler ve yenilikler sunmaktadır. PyTorch kullanılarak geliştirilen bu model, üç önemli güncellemeye sahiptir: donanım dostu omurga/boyun yapısı, verimliliğe yönelik özel baş tasarımı ve etkili eğitim stratejileri. Yapılan testlerde, YOLOv6’nın nesne algılama performansının hem hız hem de doğruluk açısından aşamalı sürümlerle birlikte diğer CNN tabanlı algoritmalarla kıyaslanabilir olduğu kanıtlanmıştır [65].

YOLOv8, 2021 yılında Darknet tarafından oluşturulan bir nesne algılama algoritmasıdır. YOLOv4 algoritmasının bir güncellemesidir ve daha hızlı, daha doğru ve daha az kaynak gerektirir. YOLOv8 model mimarimiz, Şekil 3.12’de gösterildiği gibi omurga, Boyun ve Baştan oluşmaktadır [66]. YOLOv8, nesne algılama için güçlü bir algoritmadır. Daha hızlı, daha doğru ve daha az kaynak ihtiyaç duyduğu için gerçek zamanlı nesne algılama için ve daha düşük güçlü cihazlarda nesne algılama için uygun bir seçimdir. Çalışmada YOLOv8’in alt mimarileri olan YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l ve YOLOv8x kullanılmıştır. Çalışmada YOLOv8’in mAP50 başarımlarında rakiplerine göre zaman ve doğruluk parametrelerinde göre değerlendirmede çok başarılı olduğu gözlemlenmiştir.



Şekil 3.12. YOLOv8 mimarisi

3.2.2.6. Grounding DINO ve Zero-Shot Nesne Tespiti

Grounding DINO, transformer tabanlı bir nesne algılama modelidir ve hem görsel hem de metinsel verileri kullanarak nesne tespiti yapar. Model, görsellerdeki nesneleri bulmak için kullanıcıdan aldığı metinsel komutları (örneğin, "ayaktaki insanlar?") kullanır [67]. Bu yaklaşım, görsel ve dil modellerinin birleştirilmesine dayalıdır ve oldukça esnek bir yapıya sahiptir. Grounding DINO, Detection Transformer (DINO) mimarisini temel alır ve buna metin desteği ekleyerek görsellerle metinler arasında güçlü bir bağ kurar. Özellikle çoklu nesne içeren sahnelerde ve karmaşık görüntülerde etkili bir çözüm sunar.

Zero-shot detection, bir modelin daha önce hiç eğitim verisinde karşılaşmadığı nesneleri veya kategorileri, sadece metinsel tanımlara ya da genel bilgiye dayanarak tanıması anlamına gelir [68]. Geleneksel nesne algılama sistemleri, eğitim verileriyle sınırlıyken, zero-shot modeller, öğrenme aşamasında görmedikleri nesneleri dahi tespit edebilirler. Bu yetenek, dil ve görsel bilgiyi aynı uzayda (shared embedding space) temsil eden modellerin gücünden gelir. Özellikle görsel-dil modelleri, ortak temsiller sayesinde yeni nesneleri tanımlama becerisi kazanırlar.

Grounding DINO'nun en önemli yeteneklerinden biri, zero-shot detection gerçekleştirebilmesidir. Model, görsellerdeki nesneleri tanımlamak için metin desteği aldığından, daha önce eğitim sırasında görmediği nesneleri metinsel açıklamalara dayalı olarak bulabilir. Bu durum, Grounding DINO'nun zero-shot öğrenme yapısını doğrudan desteklediğini gösterir. Özetle, Grounding DINO, dil ve görsel bilgiyi birleştirerek metinsel sorgularla yeni nesneleri algılayabilen ve zero-shot detection gerçekleştirebilen güçlü bir sistemdir.

3.2.3. Görüntü Bölütleme

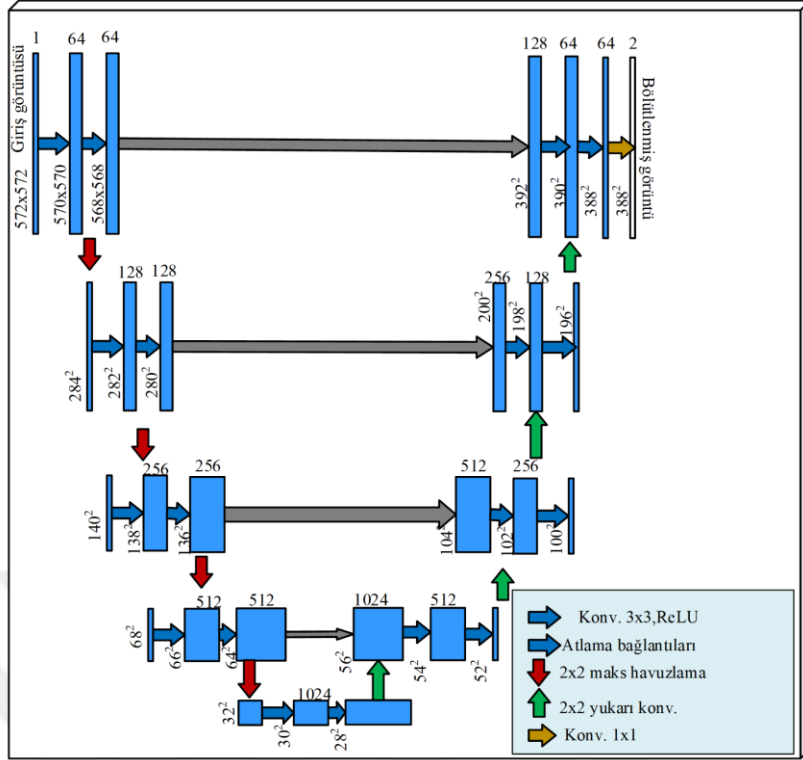
Bölütleme, genellikle görüntü analizinde ilk adım olarak gerçekleştirilir. Görüntü bölütleme, bir görüntünün anlamlı özellikler içeren çeşitli bölgelere ayrılması sürecidir. Örneğin, görüntüde benzer parlaklık düzeylerine sahip alanlar olabilir ve bu alanlar, görüntünün farklı kısımlarındaki nesneleri temsil edebilir. Görüntü bölütleme, semantik bölütleme (anlamsal bölütleme) ve instance bölütleme (örnek bazlı bölütleme) olarak iki ana kategoriye ayrılmaktadır [69]. Seminer çalışmasında anlamsal bölütleme uygulamaları geliştirilmiştir. Anlamsal bölütleme, görüntüdeki her bir pikseli anlama tanımlama işlemini yapar ve nesne sınıfına göre sınıflandırır [70]. Örnek bölütleme ise görüntüdeki her bir pikseli anlama tanımlama işlemini yapar ve her bir nesneyi aynı sınıftan da olsa farklı olduğunu belirtir [71]. Anlamsal ve örnek bölütleme arasındaki fark Şekil 3.13'te görülmektedir.



Şekil 3.13. Anlamsal ve örnek bölütleme arasındaki fark

3.2.3.1. Unet

Unet, görüntü bölütleme işlemlerinde yaygın olarak kullanılan otomatik bir yöntemdir ve ilk olarak Ronneberger ve çalışma arkadaşları tarafından biyomedikal görüntü bölütleme amacıyla geliştirilmiştir [72]. Unet, piksel bazlı bir bölütleme yaklaşımı sunar ve farklı katmanlardan oluşan bir evrişimsel sinir ağı mimarisine sahiptir. Geleneksel yöntemlere kıyasla, daha yüksek performans elde edilmiştir ve az sayıda görüntüden oluşan veri setlerinde de başarı gösterdiği kanıtlanmıştır. Unet, adını mimarisinin U şekline benzemesinden alır ve Şekil 3.14'te bu yapı gösterilmiştir. Mimari, bir kodlayıcı ve bir çözücü olmak üzere iki ana bölümden oluşur. Kodlayıcı, görüntünün içeriğini analiz eder ve maksimum havuzlama katmanlarını içerir. Çözücü kısmı ise simetrik bir genişleme yolunu izleyerek hassas yerleştirme sağlar ve bu süreçte devredilen konvolüsyonlar kullanılır (aynı zamanda dekode olarak da adlandırılmaktadır) [73].



Şekil 3.14. Unet mimarisi

Şekil 3.14'te gösterildiği gibi, her çözücü bloğu, kodlayıcı yolundaki karşılık gelen özellik haritalarıyla yukarı örneklenmiş özellik haritalarını birleştirir. Bu atlama bağlantıları, ağıın hem yerel hem de global bağlamsal bilgiyi farklı ölçeklerde yakalamasına olanak tanır. Atlama bağlantıları, kodlayıcı yolundan gelen ayrıntılı bilgilerin çözümleme sırasında kaybolmasını engelleyerek korunmasını sağlar. Bu bağlantılar sayesinde düşük seviyeli özellikler ve mekânsal bilgiler çözücüye aktarılarak, bölütleme sınırlarının daha hassas çizilmesine yardımcı olunur ve daha doğru sonuçlar elde edilir. Unet; tıbbi görüntü bölütleme, hücre çekirdeği bölütleme ve otonom sürüşte yol bölütleme gibi birçok alanda başarıyla kullanılmaktadır. Bu başarı, modelin sınırlı eğitim verisi ile çalışabilme kapasitesi ve hem yerel hem de global bağlamsal bilgiyi etkili bir şekilde yakalama yeteneği ile ilişkilendirilmektedir [72, 73].

3.2.3.2. Genişleyen Evrişimsel Sinir Ağları (Dilated Convolutional Neural Networks)

Genişletilmiş evrişim, her adımda parametre sayısını artırmaksızın alıcı alanı büyüterek geniş bir vektör gösterimini tarar. Standart evrişim (genişleme oranı 1) ile karşılaştırıldığında, bu yöntemde kullanılan parametre sayısının yaklaşık üç kat azaldığı gözlemlenmiştir [74]. İlk katmanda, genişleme oranı 1 olan ve 3x3 boyutundaki bir filtre uygulanır, bu da standart bir evrişim katmanı anlamına gelir. Daha büyük genişleme oranlarına sahip evrişim filtreleri sonraki katmanlarda kullanılır. D-CNN mimarisinde genişleme oranları sırasıyla 1, 2, 4 ve 8 olarak

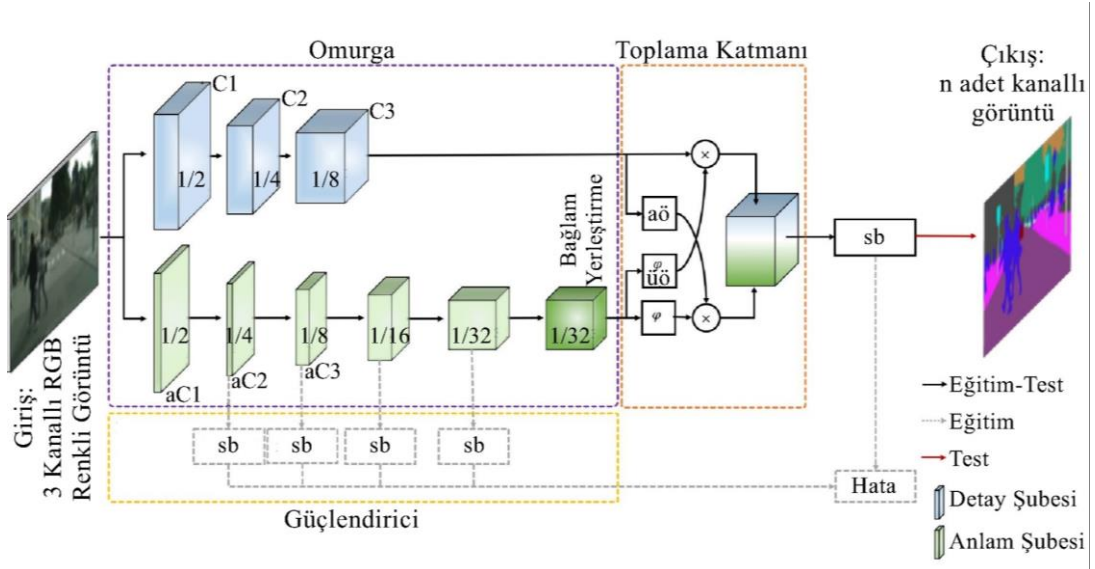
belirlenmiştir. Bu oranlar, başlangıçta 3x3 olan filtrenin sırasıyla 5x5, 9x9 ve 17x17 boyutlarına genişlemesini sağlayarak daha geniş bir alanı kapsar. Her bir katmanın çıktısı, bir önceki katmanın daha küçük boyutlu bir özetidir. D-CNN, çözünürlük veya kapsama alanında kayıp olmaksızın alıcı alanın üssel genişlemesini mümkün kılmaktadır [75]. Şekil 3.15'te bir D-CNN mimarisi örneği gösterilmektedir.



Şekil 3.15. D-CNN mimarisi

3.2.3.3. BiSeNetV2

BiSeNetV2 algoritması kullanılarak demiryolu görüntüleri için hızlı anlamsal bölütleme ray tespiti gerçekleştirilmiştir. Semantik bölütleme, bilgisayarlı görmede önemli bir alandır ve piksel düzeyinde bir görüntüyü anlamayı amaçlar. Semantik bölütlemenin temel amacı, görüntüdeki her pikseli belirli bir sınıfa etiketlemektir. Bu çalışmada, gerçek zamanlı anlamsal bölütleme için iki yönlü bölütleme ağı BiSeNetV2 olarak adlandırılan iki yönlü bir mimari önerilmiştir. BiSeNetV2, yerleştirme problemini anlamsal çıkarım probleminden ayırmanın ve ardından iki bilgiyi uygun şekilde birleştirmenin bir yolunu sunmaktadır. BiSeNetV2, gerçek zamanlı anlamsal bölütleme için oluşturulmuş iki yönlü bir mimaridir [76]. İlk yol, Ayrıntı Dalı adı verilen geniş kanallar ve sık katmanlarla uzamsal ayrıntıları yakalamak için tasarlanmıştır. Buna karşılık, dar kanallar ve derin katmanlarla kategorik anlambilimi çıkarmak için Semantik Dal adı verilen diğer yol tanıtıldı. Semantik Dal, anlamsal bağlamı yakalamak için geniş bir alıcı alana ihtiyaç duyarken, ayrıntılı bilgi Ayrıntı Dalı tarafından sağlanabilir [77]. Bu nedenle, Semantik Dal, daha az kanal ve daha hızlı alt örnekleme stratejisi ile çok hafif hale getirilebilir. Her iki özellik temsili türü, daha güçlü ve daha kapsamlı bir özellik temsili oluşturmak için birleştirilir. BiSeNetV2'nin yapısı Şekil 3.16'da gösterilmektedir. Detay Dal ve Semantik Dal, özellik temsilinde birbirini tamamlayıcıdır.



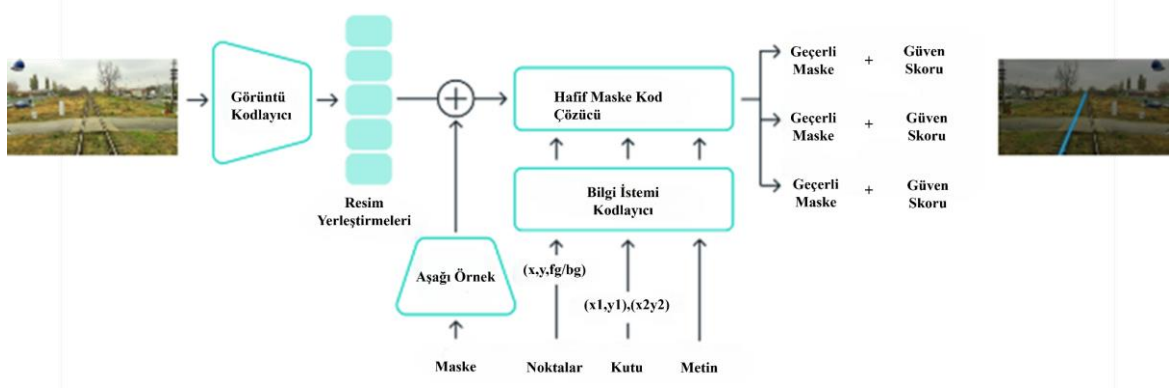
Şekil 3.16. BiSeNetV2 mimarisi

3.2.3.4. SAM (Segment Anything Model)

SAM, Meta AI tarafından geliştirilen bir bölütleme yöntemidir. SAM, eğitim verilerinde mevcut olmasa bile, bir görüntüdeki herhangi bir nesneyi bölümlendirmek için kullanılabilen, görüntü bölümlendirmeye yönelik yeni bir mimaridir [78]. Model üç ana bileşenden oluşmaktadır.

- Görüntü kodlayıcı:** Bu bileşen, girdi olarak bir görüntüyü alır ve görüntünün sabit uzunlukta gömülmesini sağlar. Görüntü kodlayıcı, görüntü sınıflandırma ve nesne algılama konusunda oldukça etkili olan bir tür derin öğrenme modeli olan Vision Transformer (ViT) modelidir [79].
- Bilgi istemi kodlayıcı:** Bu bileşen, girdi olarak bir bilgi istemini alır ve istemin sabit uzunlukta bir yerleştirmesini üretir. Bilgi istemi, görüntüdeki bir nokta, sınırlayıcı kutu veya bölümlendirilecek nesnenin metin açıklaması gibi bölümleme görevine rehberlik etmeye yardımcı olan herhangi bir şey olabilir [79].
- Maske kod çözücü:** Bu bileşen, görüntü gömme ve istemli gömme işlemlerini girdi olarak alır ve görüntüdeki nesne için bir bölütleme maskesi üretir. Maske kod çözücü, piksel düzeyinde bölümleme maskelerini tahmin etmek için eğitilmiş hafif bir transformatör modelidir [79].

SAM modeli, manuel olarak bölümlere ayrılmış bir görüntü veri kümesi üzerinde eğitilir. Eğitim süreci görüntü kodlayıcının, istem kodlayıcının ve maske kod çözücünün ortaklaşa optimize edilmesini içermektedir [80]. SAM modelini kullanmak için bir görüntü ve istem sağlamak yeterlidir. Model daha sonra görüntüdeki nesne için bir bölütleme maskesi oluşturacaktır. Maske, nesne algılama, görüntü düzenleme ve sahneyi anlama gibi çeşitli görevler için kullanılabilir. Şekil 3.17, SAM mimarisinin bir diyagramını göstermektedir.



Şekil 3.17. SAM mimarisi

3.3. Görüntü Arttırım Teknikleri

Geleneksel veri artırma yöntemleri, veri setini genişletmek amacıyla çeşitli geometrik ve istatistiksel dönüşümler uygulanarak gerçekleştirilir. Bu teknikler, görüntü yansıması, döndürme, ölçeklendirme, gürültü ekleme ve parlaklık ayarı gibi basit işlemler içermektedir. Bu yöntemler, verinin yapısını doğrudan değiştirmeden varyasyonlar oluşturarak modelin daha çeşitli örneklerle eğitilmesini sağlamaktadır. Geleneksel artırma yöntemleri, hesaplama açısından verimlidir ve yapay zekâ modellerinin aşırı uyumunu (overfitting) önlemeye yardımcı olmaktadır.

Öte yandan, yapay zekâ tabanlı veri artırma teknikleri, özellikle Generative Adversarial Networks (GAN) gibi modeller kullanılarak gerçekleştirilir. GAN'lar, bir "üretici" ve "ayırıcı" ağdan oluşmaktadır [81]. Üretici ağ, orijinal verilere benzer yeni veriler üretirken, ayırıcı ağ bu verilerin orijinal olup olmadığını ayırt etmeye çalışır. GAN tabanlı artırma, özellikle sınırlı veri setlerinde yüksek kaliteli ve gerçekçi sentetik veriler üretebilir. Bu yöntem, geleneksel yöntemlerden daha karmaşık olup, verinin özelliklerine daha derinlemesine bir müdahale sağlar ve daha sofistike veri artırma işlemleri sunar.

3.3.1. Geleneksel Teknikler

Geleneksel görüntü artırma teknikleri, görüntü yansıması, parlaklık ayarı ve gürültü ekleme yöntemlerinden oluşmaktadır [82]. Görüntü yansıması, görüntünün x eksenini ya da y eksenini boyunca çevrilmesiyle elde edilen yeni bir görüntüdür. Gürültü ekleme, orijinal görüntünün her bir pikseline rastgele ek bilgiler ekleyerek, güçlendirilmiş bir görüntü oluşturmayı hedefler. Parlaklık ayarı ise, görüntüdeki her piksele uygulanan doğrusal bir dönüşüm sürecidir. Parlaklık dönüşümü, $I = \lambda I$ formülü ile ifade edilir; burada λ , parlaklık dönüşüm katsayısını temsil eder. $0 < \lambda < 1$ aralığı görüntüyü karartırken, $\lambda > 1$ değeri görüntünün aydınlanmasını sağlamaktadır.

3.3.2. Yapay Zekâ Tabanlı Teknikler

Tez çalışmasında yapay zekâ tabanlı görüntü arttırma tekniği olarak GAN kullanılmıştır. GAN algoritmik mimarisi iki farklı sinir ağından oluşur. Bunlar; üretici ve ayırmacıdır. Bu ağlar çekişmeli bir şekilde çalışır. Üretken ağ, girdi olarak aldığı gürültü vektörünün eğitim setindeki gerçek görüntüleri simüle etmeyi amaçlarken, ayırmacı ağ, sentetik olarak üretilmiş görüntüler (sahte) ile gerçek eğitim görüntüleri arasında ayırım yapmaya çalışır [83]. Şekil 3.18, geleneksel GAN mimarisini göstermektedir. GAN mimarisinde amaç, gürültü alanını gerçek veri dağılımına eşlemektir. Eğitim, ayırt edicinin tahminleri neyin gerçek neyin sahte olduğunu ayırt edemeyene kadar devam eder. GAN mimarisi tarafından üretilen görüntülerin kalitesi ve çeşitliliği, önerilen modelin performansını doğrudan etkilemektedir.



Şekil 3.18. GAN mimarisi

3.4. Değerlendirme Ölçütleri

Kullanılan derin öğrenme modellerinin ortaya koyduğu sonuçlar büyük önem taşır. Başka bir ifadeyle, bu modellerin iyi performans göstermesi beklenir. Modelin performansını ölçmek amacıyla çeşitli değerlendirme kriterleri bulunmaktadır.

Karışıklık matrisi, sınıflandırma algoritmalarının başarımını değerlendirmede kullanılır ve hedeflenen tahminlerin gerçek değerlerle karşılaştırılmasını sağlar. İki sınıflı bir sınıflandırma modeli için örnek bir karışıklık matrisi Şekil 3.19’da gösterilmiştir. Bu matris içerisinde GP (gerçek pozitif), YP (yanlış pozitif), YN (yanlış negatif) ve GN (gerçek negatif) terimleri şu anlamlara gelmektedir.

	Gerçek Etiket	
	GP	YP
Tahmin Edilen Etiket	YN	GN

Şekil 3.19. İki sınıflı bir problem için karışıklık matrisi

Sınıflandırıcının performansı aşağıdaki beş standart metrik kullanılarak değerlendirilmektedir. Doğruluk oranı (accuracy rate) ise genel performansın ana değerlendirme kriterlerinden biridir ve doğru sınıflandırılmış örneklerin toplam örnekler içindeki oranı olarak tanımlanır. Bu formül ise Denklem 3.6’da gösterilmektedir [84]. Duyarlılık (recall), bir modelin pozitif sınıfları ne kadar doğru tahmin ettiğini gösteren bir ölçüttür. Duyarlılık hesaplaması Denklem 3.7’de verilmiştir [84]. Kesinlik (precision) ise, doğru bir şekilde pozitif olarak sınıflandırılan örneklerin, toplam pozitif sınıflandırmalar içindeki oranını ifade eder. Bu hesaplama ise Denklem 3.8’de yer almaktadır [84]. F-ölçütü (F1 skoru), duyarlılık ve kesinliğin harmonik ortalamasını temsil eder ve formülü Denklem 3.9’da bulunabilir [84].

- a) GP: Tahmin edilen sınıf etiketinin pozitif ve gerçek sınıf etiketinin doğru olduğu örneklerin sayısı.
- b) YP: Tahmin edilen sınıf etiketinin pozitif ve gerçek sınıf etiketinin yanlış olduğu örneklerin sayısı.
- c) YN: Tahmin edilen sınıf etiketinin negatif ve gerçek sınıf etiketinin yanlış olduğu örneklerin sayısı.
- d) GN: Tahmin edilen sınıf etiketinin negatif ve gerçek sınıf etiketinin doğru olduğu örneklerin sayısı.

$$\text{Doğruluk}(A) = \frac{GP+GN}{TP+TN+YP+YN} \quad (3.6)$$

$$\text{Duyarlılık}(\text{Recall}, R) = \frac{GP}{GP+YN} \quad (3.7)$$

$$\text{Kesinlik}(\text{Precision}, P) = \frac{GP}{GP+YP} \quad (3.8)$$

$$F1 = 2 * P * R / (P + R) \quad (3.9)$$

Denklem 3.10’daki AP (ortalama kesinlik), bir modelin belirli bir sınıfta ne kadar doğru tespit yaptığını, doğruluk ve duyarlılık grafiği altındaki alanı ölçerek değerlendirmektedir [85]. Denklem 3.11’deki mAP (ortalama ortalama kesinlik), birden fazla sınıf için hesaplanan AP değerlerinin ortalamasını alarak modelin genel performansını göstermektedir [85]. Yüksek mAP,

modelin farklı sınıflarda tutarlı bir şekilde doğru sonuç verdiğini ifade etmektedir. AP ve mAP, özellikle bilgisayar görüşü ve derin öğrenme tabanlı uygulamalarda kullanılır. Nesne tespiti ve anlamsal bölütmede değerlendirme ölçütü olarak kullanılmaktadırlar. Denklem 3.11'deki N toplam sınıf sayısını, AP_i her bir sınıf için hesaplanan AP değerini temsil etmektedir.

$$AP = \int_0^1 P(R) dR \quad (3.10)$$

$$mAP = \frac{\sum_{i=1}^N AP_i}{N} \quad (3.11)$$

Önerilen modellerin performansın metriklerini değerlendirmek için isabet oranının yanı sıra Jaccard indeksi (IoU) ve Dice katsayısı (F1) metrikleri de kullanılmaktadır. IoU olarak da bilinen Jaccard İndeksi, anlamsal bölütleme de en önemli değerlendirme kriterlerinden biridir. IoU, doğru sınıflandırılmış piksellerin (tahmin edilen bölütleme pikselleri ile referans arasındaki örtüşme) referans ve tahmin edilen bölütleme piksellerinin birleşimine oranıdır [86]. F1 puanı olarak da bilinen Dice katsayısı, IoU'ya benzer özellikler taşımaktadır. Dice katsayısı, tahmin edilen bölütleme pikselleri ile referans arasındaki örtüşmenin, tahmin edilen ve referans görüntüsündeki toplam piksel sayısına oranının iki katıdır [86].

Literatürde GAN mimarisinin performansını ölçen metrikler vardır. Bu performans analizi, Görsel Turing Testi (VTT), Frechet Başlangıç Mesafesi (FID) ve İstatistiksel Olarak Farklı Kutuların Sayısını (NDB) içerir. VTT, yapay veri ile gerçek verinin uzmanlar tarafından oylanmasıdır. Bu zaman alıcı bir süreçtir. FID ve NDB, otomatik performans ölçümleridir. FID metriği Denklem 3.12'deki gibidir.

$$FID(x, g) = \|\mu_x - \mu_g\|_2^2 + Tr(C_x + C_g - 2(C_x C_g)^{1/2}) \quad (3.12)$$

Denklem 3.12, (μ_x, C_x) ve (μ_g, C_g) sırasıyla gerçek ve sahte görüntülerin ortalama ve kovaryans değerlerini gösterir [83]. Sahte görüntü gerçek görüntüye ne kadar yakınsa, FID değeri 0'a o kadar yakındır. NDB, Richardson ve Weiss tarafından üretilen örnek çeşitliliğini ve mod çökmesini ölçmek için önerilmiştir [87]. Aynı dağılımdan iki grup örnek verildiğinde, belirli bir kutudaki örnek sayısı, örnekleme gürültüsüne kadar aynı olmalıdır. NDB değeri 0'a ne kadar yakınsa o kadar başarılıdır.

Anlamsal bölütlemelerde yaygın olarak kullanılan bir diğer performans ölçütü, ortalama IoU'dur. IoU, tahmin edilen görüntü ile referans görüntünün kesişim alanının, bu iki görüntünün birleşim alanına oranlanmasıyla elde edilir. Her sınıf için ayrı ayrı IoU değerleri hesaplanarak performans değerlendirmesi yapılmaktadır. Bu değerlerin ortalaması alınarak ortalama IoU değeri elde edilmektedir [88]. Doğruluk oranı ise, doğru piksel tahminlerinin sayısını ölçmektedir. Denklem 3.13'te hesaplaması görülmektedir.

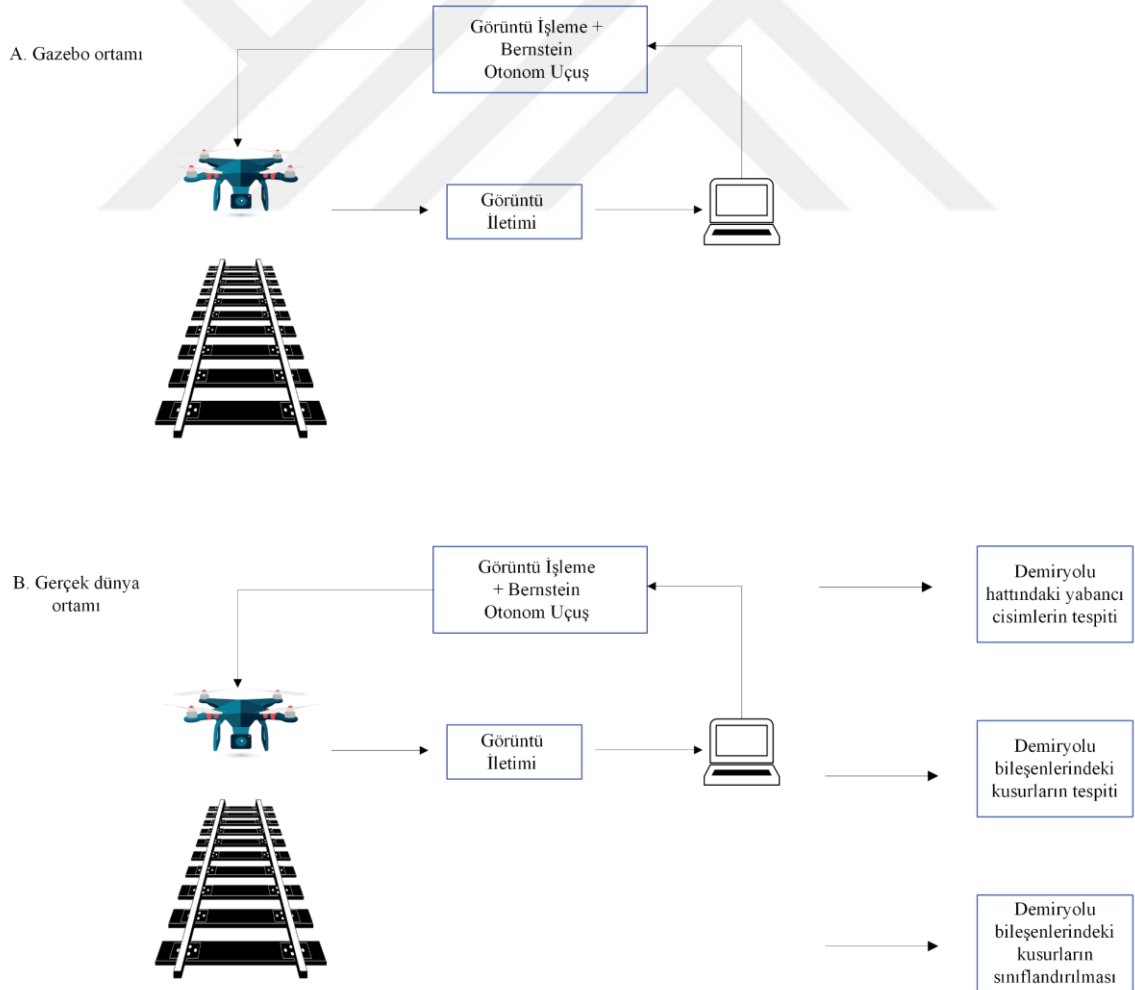
$$IoU = \frac{Gerçek \cap Tahmin}{Gerçek \cup Tahmin} \quad (3.13)$$



4. MATERYAL VE METOT

Tez çalışmasında kullanılacak İHA öncelikle Gazebo simülasyon ortamında otonom bir uçuş için hazırlanacaktır. Otonom bir şekilde demiryolu hattını takip edecektir. Bunun için görüntü işleme ve matematiksel algoritma tabanlı bir model geliştirilmiştir. Bu model için demiryolu hattını içeren görüntüler eğitim veri seti için kullanılacaktır. Simülasyon sonucunda elde edilen model gerçek dünya ortamındaki bire bir aynı olan Anafı4K İHA'sı üzerinde kullanılacaktır.

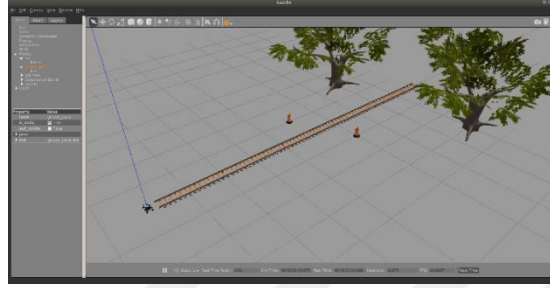
Otonom İHA uçuşu ile elde edilen demiryolu görüntüleri hazır veri setlerine ek olarak veri seti olarak kullanılmıştır. Elde edilen bu veri seti üzerinde geliştirilen hibrit derin öğrenme modelleri ve görüntü işleme teknikleri sayesinde demiryolu bileşenleri tespit edilmiştir. Daha sonra veri artırım teknikleri ile kusurlu bileşenler ayrılacak ve kusur tespit yöntemleri gerçekleştirilmiştir. Aynı zamanda demiryolu çevresinde bulunan yabancı cisimlerde tespit edilmiştir. Önerilen yöntemin genel gösterimi Şekil 4.1'de görülmektedir.



Şekil 4.1. Önerilen yöntemin genel blok gösterimi

4.1. Simülasyon Ortamında İHA ile Ray Takibi ve Tespiti

Gazebo, üç boyutlu robot simülasyonları için kullanılan bir platformdur [89]. Bu simülatör, çeşitli sensör modellerini ve fizik motorlarını kullanarak üç boyutlu sanal bir ortam oluşturur [89]. Gazebo'nun önemli avantajlarından biri, İHA, ağaç ve ev gibi popüler modelleri simülasyon ortamında kullanıma sunmasıdır. Şekil 4.2'de Gazebo'nun simülasyon ortamı gösterilmektedir.



Şekil 4.2. Gazebo ortamı

Parrot Anafi4k, Gazebo ortamıyla uyumlu çalışan bir insansız hava aracıdır. Ön kısmında bir kamera bulunmaktadır ve demiryolu ray görüntüleri bu kamerayla kaydedilmiştir. Geniş menzilli kamera, -90 ila +90 derece arasında eğim açısı sunarak İHA üzerindeki nesneleri inceleme olanağı sağlar. Gazebo'da demiryolu rayı modeli oluşturmak için 3D Warehouse'dan Collada (.dae) formatında bir üç boyutlu model dosyası indirilmiştir [90]. Sonrasında bu dosya Gazebo'ya uygun bir SDF formatına dönüştürülmüştür.

4.1.1. Ufuk Noktası Yöntemi ile Ray Takibi

Gerçek dünyada demiryolu rayları perspektiften bakıldığında ufuk çizgisinde birleşiyor gibi görünür. Paralel hatların bu kesişim noktalarından hareketle demiryolunun ufuk noktası belirlenebilir. Rayların görüntülerden çıkarılmasına ilişkin işlem adımları aşağıda anlatılmıştır. Görüntüdeki çizgileri tespit edebilmek için keskinleştirme, kenar algılama ve yumuşatma gibi düşük seviyeli görüntü işleme teknikleri kullanılmıştır. İlk olarak Gabor filtresi uygulanarak Anafi4k İHA'sından alınan canlı görüntülerde belirli yönlerdeki kenarlar tespit edilmiştir. Gabor filtresi, belirli bir yön ve frekansta uzanan ayrıtları yakalamak için ideal bir filtredir. Bu filtre, rayların paralel olarak uzandığı yönleri başarılı bir şekilde ortaya çıkarmıştır. Anafi4k İHA'sından alınan canlı görüntüler üzerinde Gabor filtresi uygulanarak, belirli bir yöne uzanan çizgiler tespit edilmiştir. Gabor filtresi ile, rayların yönelimi ve kenar yapısı net bir şekilde görüntüye yansıtılmıştır.

Daha sonra, görüntüdeki kenarları belirlemek amacıyla Canny kenar algılama yöntemi kullanılmıştır. John Canny tarafından geliştirilen bu yöntem, kenar tespitinde yaygın olarak

kullanılan dört aşamalı bir algoritmadır: Giriş görüntüsü Gauss filtresi ile yumuşatılır, her pikselin kenar olma olasılığı hesaplanır, büyük olmayan komşu kenar eğilimleri sıfırlanır ve çift eşik değeri uygulanır [91]. Denklem 4.1, standart sapma değerini kullanarak Gauss filtresi yardımıyla kenar algılama sürecini ifade eder [92].

$$\text{Kenar Olma Eğilimi}(G) = \sqrt{G_x + G_y} \quad (4.1)$$

Denklem 4.1’de G_x ve G_y giriş görüntüsünün x eksenindeki türevini, y eksenindeki türevlerini vermektedir.

Son olarak, görüntüdeki raylar Hough dönüşümü yöntemiyle çıkarılmıştır. Hough dönüşümü, görüntü üzerinde düz çizgileri ve parametrik eğrileri tanımlamak için kullanılan bir tekniktir [93]. Denklem 4.2’de Hough dönüşümünün matematiksel formülü gösterilmiştir.

$$\rho = x \cos \theta + y \sin \theta \quad (4.2)$$

Denklem 4.2’de kullanılan θ parametresi, doğru parçasının açısı değerini temsil etmektedir [94]. Bu gösterimle, xy düzlemindeki her nokta, $\rho\theta$ parametre uzayında bir sinüzoidde eşlenir. Parametre uzayında xy düzleminde iki nokta eş doğrusal ise, bunların sinüzoidleri Denklem 4.2’ye göre doğruyu tanımlayan belirli bir noktada (ρ, θ) kesişecektir [95]. Algoritmanın genel fikri, birçok sinüzoidin kesiştiği uzay parametrelerindeki noktaları (ρ, θ) bulmaktır. Bu noktalar orijinal görüntüdeki çizgilere karşılık gelecektir [95].

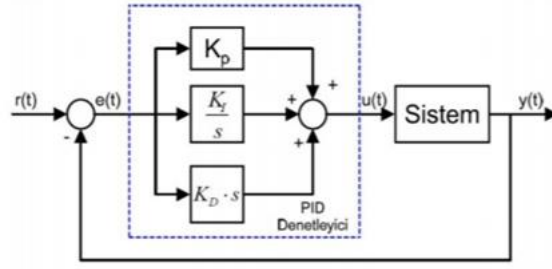
Hough dönüşümünden elde edilen başlangıç ve bitiş noktaları, iki ayrı diziye kaydedilmiş ve en küçük kareler yöntemi kullanılarak ufuk noktası tahmini gerçekleştirilmiştir. Intersect() fonksiyonu, bu iki diziyi parametre olarak alarak ufuk noktasının koordinatlarını hesaplamaktadır.

Belirlenen ufuk noktası, takipEt() fonksiyonuna parametre olarak iletilmiş ve takip işlemi PID (Proportional, Integral, Derivative) kontrolörü ile gerçekleştirilmiştir. PID, endüstride yaygın olarak kullanılan bir kapalı döngü geri besleme mekanizmasıdır. Bir PID denetleyicisi sürekli olarak hatayı hesaplar ve bu hatayı oransal, integral ve türev terimleriyle düzeltmeye çalışır. Matematiksel olarak PID denetleyicisi Denklem 4.3 ve 4.4 şeklinde gösterilmektedir.

$$u(t) = K_p e(t) + K_i \int_0^t e(t) dt + K_d \frac{de(t)}{dt} \quad (4.3)$$

$$e(t) = r(t) - y(t) \quad (4.4)$$

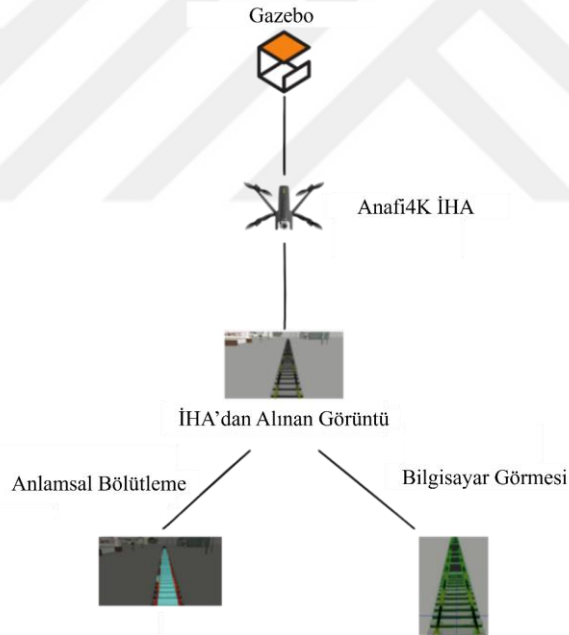
Denklem 4.3 ve 4.4’te $r(t)$ giriş, $u(t)$ kontrol, $e(t)$ hata ve $y(t)$ sistemin çıktı sinyalidir. K_p , K_i , K_d katsayıları ise sırası ile oransal, integral, türev katsayılarıdır [96]. Şekil 4.3’te PID denetleyicisine ait şema görülmektedir. PID denetleyicisi sayesinde Anafı4k İHA yaw ekseninde anlık olarak hatayı hesaplayarak ufuk noktasını tahmin etmektedir.



Şekil 4.3. PID denetleyicisi şeması

4.1.2. Bilgisayar Görmesi Yöntemleri ile Ray Takibi

Bu çalışmada, İHA modelinin otonom olarak demiryolu hattını takip etmesi hedeflenmiştir. Uygulamanın genel yapısı Şekil 4.4'te sunulmaktadır. İHA'nın kamera görüntülerinden yararlanarak ray hattı üzerinde otonom bir şekilde ilerlemesi sağlanmıştır. Geliştirilen yöntem, aşağıdaki adımlardan oluşmaktadır.



Şekil 4.4. Uygulamanın genel yapısı

Görüntü Ön İşleme: Python ortamında, İHA kameradan alınan gerçek zamanlı görüntüler üzerinde işlem yapılmıştır. İlk olarak, görüntünün ray bölgesine odaklanabilmesi için görüntü belirli bir boyuta kırpılmıştır. Bu adım, İHA'nın yalnızca ray hattına odaklanarak gereksiz bilgi yığınının arınmasını sağlar. Görüntü, $I(x, y)$ olmak üzere bir $M \times N$ boyutunda piksel matrisidir. Burada x ve y , görüntüdeki yatay ve dikey piksel konumlarını temsil eder. Kırpma işlemi, görüntünün yalnızca belirli bir bölgesini koruyarak geri kalan kısmını yok saymayı içerir. Örneğin,

kırpılan bölge $[x_1: x_2, y_1: y_2]$ aralığındaki piksellerden oluşuyorsa, kırpılan görüntü $I'(x', y')$ Denklem 4.5'teki gibi ifade edilmektedir.

$$I'(x', y') = I(x_1 + x', y_1 + y') \quad (4.5)$$

Bu işlem, görüntünün sadece ilgilenilen kısmının işlenmesini sağlar ve İHA'nın ray hattına odaklanmasına yardımcı olmaktadır.

Gri Tonlama ve Gauss Bulanıklığı: Kırpılmış görüntü $I'(x, y)$ renkli bir görüntü ise her piksel kırmızı (R), yeşil (G) ve mavi (B) bileşenleri içerir. Gri tonlamaya çevrilmiş görüntü $I_{gray}(x, y)$ Denklem 4.6'ya göre hesaplanmaktadır.

$$I_{gray}(x, y) = 0.299 * R(x, y) + 0.587 * G(x, y) + 0.114 * B(x, y) \quad (4.6)$$

Denklem 4.6, her pikselin renk bileşenlerinden bir ağırlıklı ortalama olarak gri seviyesini elde etmektedir. Daha sonra, Gauss bulanıklığı uygulanmıştır. Gauss bulanıklığı, bir $G(x, y)$ Gauss çekirdeği ile görüntünün konvolüsyonu ile elde edilir. Denklem 4.7'de görülmektedir.

$$I_{blur}(x, y) = I_{gray}(x, y) * G(x, y) \quad (4.7)$$

Gauss fonksiyonu, iki boyutta Denklem 4.8'deki gibi tanımlanmaktadır.

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \quad (4.8)$$

Bu işlem, görüntüdeki yüksek frekanslı bileşenleri (gürültüyü) azaltarak daha pürüzsüz bir görüntü elde edilmesini sağlamaktadır.

İkili Görüntüleme (Thresholding): Eşikleme işlemi, gri tonlamalı görüntüye bir eşik değeri (T) uygulanarak gerçekleştirilir. Piksel yoğunluğu ($I_{blur}(x, y)$) eşik değerinden büyükse piksel beyaz (1), küçükse siyah (0) olarak atanır. Matematiksel olarak bu Denklem 4.9'daki gibi ifade edilmektedir. Bu işlem, görüntüyü yalnızca iki seviyeye indirgeyerek, rayların belirginleştirilmesine yardımcı olmaktadır.

$$I_{blur}(x, y) = \begin{cases} 1 & \text{eğer } I_{blur}(x, y) \geq T \\ 0 & \text{eğer } I_{blur}(x, y) < T \end{cases} \quad (4.9)$$

Çizgi Algılama ve Rota Belirleme: İkili görüntüde çizgi algılama işlemi için Hough Dönüşümü gibi bir teknik kullanılır. Hough Dönüşümü, görüntüdeki doğrusal özellikleri tanımlamak için kullanılan bir yöntemdir. Hough dönüşümü ile bir çizgi $y = mx + b$ formunda tanımlanmaktadır. Burada ρ , çizginin orijine olan dik mesafesi ve θ ise çizginin eğim açısıdır. Bu

parametrik form kullanılarak, görüntüdeki çizgiler tespit edilir ve çizgi koordinatları belirlenir. Bu çizgi koordinatlarına göre İHA'ya uçuş talimatları verilmiştir.

Hat takibi ile İHA modelinin otonom olarak demiryolu hattını takip etmesi amaçlanıyor. Öncelikle Python ortamında İHA kameradan alınan gerçek zamanlı görüntü kırpılır. Burada amaç İHA'nın sadece ray kısmına odaklanmasını sağlamaktır. Daha sonra gerçek zamanlı görüntü gri tonlamaya dönüştürülür ve Gauss bulanıklığı uygulanır. Görüntüdeki bulanıklığı gidermek için Gauss kullanılmıştır.

Ortaya çıkan görüntüye bir renk eşiği uygulanarak görüntü ikili değere dönüştürülür. Eşiklenmiş görüntüde çizgiler algılandı. Tespit edilen hattın koordinatlarına göre İHA'ya uçuş talimatı verilmiştir

4.1.3. BiSeNetV2 ile Demiryolu Hattı Tespiti

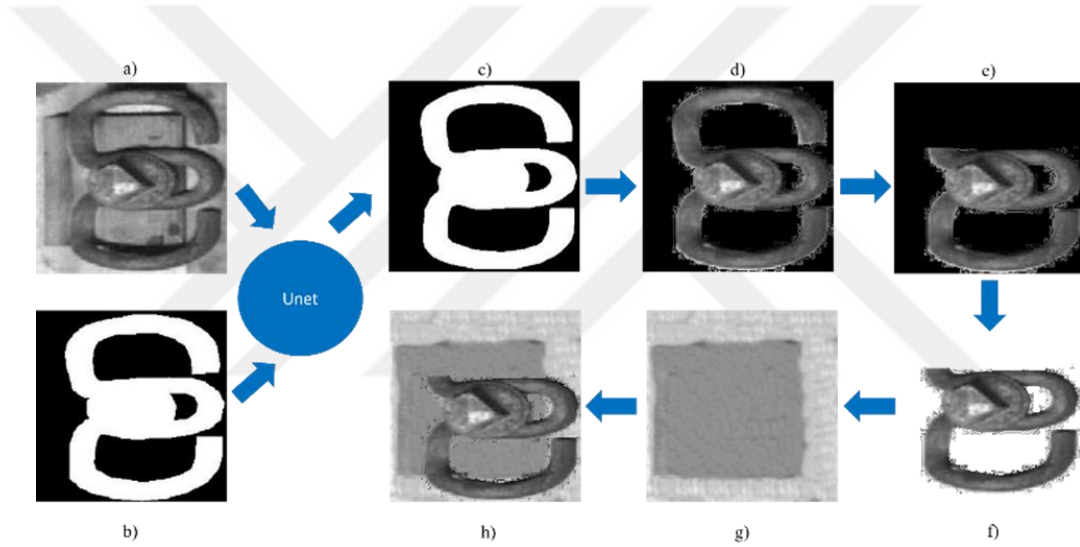
Çalışmada, bölütleme için önceden eğitilmiş BiSeNetV2 modeli kullanılmış ve RailSem19 veri setinin uyarlanmış bir versiyonu üzerinde eğitim yapılmıştır. Çalışma, BiSeNetV2'yi RailSem19 veri setinin yalnızca üç etiketli ("ray-yükseltilmiş", "ray-ray", "arka plan") değiştirilmiş bir versiyonu üzerinde eğitmiştir. BiSeNetV2'nin mimarisi, hızlı bir alt örnekleme stratejisi kullanarak semantik yolun hafif ve verimli olmasını sağlamaktadır. Detay yolunun geniş kanalları, uzamsal bilgiyi etkili bir şekilde işleyerek ayrıntılı yerelleştirme yapılmasını mümkün kılarken, semantik yol daha derin katmanlarla görüntünün genel bağlamını anlamaktadır. Bu iki tamamlayıcı özellik temsili, daha güçlü bir bölütleme performansı elde etmek için bir araya getirilmektedir.

Model performansı, temel gerçek ile modelin tahminleri arasındaki benzerliği ölçmek amacıyla IoU (Intersection over Union) metriği ile değerlendirilmiştir. IoU, iki şeklin kesişim alanının bu şekillerin birleşim alanına oranı olarak hesaplanır. IoU değeri 0 ile 1 arasında değişmekte olup, 1 mükemmel örtüşmeyi ifade eder.

4.2. Derin Öğrenme Yöntemleri ile Demiryolu Bağlantı Elemanlarının Sınıflandırılması

Bu çalışmada, dengesiz bir veri seti üzerinde ray bağlantı elemanlarının kusurlarını tespit etmek için yeni bir yöntem geliştirilmiştir. Yöntem iki ana aşamadan oluşmaktadır. İlk aşamada, kusurlu veri üretme yöntemi kullanılarak arızalı ve eksik bağlantı elemanı örnekleri oluşturulmuştur. İkinci aşamada ise önerilen CNN, VGG-16 ve ResNet50 modelleri yardımıyla kusurlu bağlantı elemanlarının sınıflandırılması hedeflenmiştir. Önerilen kusurlu veri üretim sürecine ilişkin diyagram Şekil 4.5'te sunulmaktadır. Dış mekân ortamlarında çekilen görüntüler, yetersiz aydınlatma nedeniyle genellikle bozuk olmaktadır ve yüksek gürültü seviyeleri bilgisayarlı görme algoritmalarının genel performansını olumsuz etkileyebilir. Bu durumda, görüntülerin

netliğini artırmak için düşük ışık koşullarında görüntü iyileştirme yöntemlerine başvurulmuştur. Düşük ışıkta çekilen görüntülerin görünürlüğünü artırmak amacıyla pus giderme teknikleri kullanılmış ve böylece veri setindeki demiryolu bağlantı elemanlarının sınıflandırmaya uygun hale getirilmesi sağlanmıştır. Demiryolu veri setlerinde, kusurlu bağlantı elemanlarının sayısı sağlıklı olanlara göre oldukça azdır ve kusurlu veriye ulaşmak zordur. Bu sebeple, sağlıklı ve kusurlu bağlantı elemanlarının sayısını dengelemek amacıyla Unet tabanlı bir model önerilmiştir. Unet modelinin ürettiği görüntü ile orijinal sağlıklı bağlantı elemanının görüntüsünü kullanarak, GrabCut algoritması bağlantı elemanının ön plana çıkarılmasını sağlamıştır. GrabCut, görüntüdeki nesneyi ön plana çıkarmak ve bölütlemeyi iyileştirmek için grafik kesimleri yinelemeli olarak uygular. Böylece, sağlıklı ve kusurlu bağlantı elemanlarının sayıları dengelenmiş ve daha etkili bir sınıflandırma modeli oluşturulmuştur.



Şekil 4.5. Kusurlu bağlantı elamanı üretimi (a) Sağlıklı bağlantı elemanı (b) Sağlıklı referans veri (c) Unet bölütleme sonucu (d) GrabCut ile ön plana çıkarılmış bağlantı elemanı (e) Yeni üretilen kusurlu bağlantı elamanı (f) Şeffaflaştırılmış kusurlu bağlantı elamanı (g) Varsayılan arka plan (h) Varsayılan arka plan eklenmiş kusurlu bağlantı elamanı

Unet, otomatik görüntü bölütleme amacıyla kullanılan bir yöntemdir ve Ronneberger ve ekibi tarafından biyomedikal görüntülerin bölütlenmesi için geliştirilmiştir. Farklı mimari yapılar ve evrişimli sinir ağı katmanları içeren piksel tabanlı bir bölütleme tekniği olan Unet, klasik modellere kıyasla daha iyi sonuçlar elde etmiştir. Az sayıda görüntü bulunan veri setlerinde de etkili performans göstermiştir. U şeklindeki mimarisi nedeniyle bu adı alan Unet'in yapısı Şekil 3.14'te gösterilmektedir. Model, kodlama ve kod çözme aşamalarından oluşmaktadır. Kodlama aşamasında görüntünün içeriği yakalanır ve maksimum havuzlama katmanları kullanılır. Kod çözme aşaması ise devredilen konvolüsyonları kullanarak hassas lokalizasyon sağlayan simetrik bir genişleme yoludur ve dekode olarak da bilinmektedir.

Unet modeli kullanılarak dengelenen veri seti, sınıflandırma için uygun hale getirilmiştir. Bu veri setinde bağlantı elemanları, sağlıklı ve kusurlu (arızalı veya eksik) olarak sınıflandırılmıştır. Sınıflandırma sürecinde VGG16, ResNet50 ve kullanıcı tarafından tasarlanan CNN modelleri uygulanmıştır.

Çalışmada yer verilen VGG-16 modeli, toplamda 24 katmandan oluşur. VGG-16, ImageNet Challenge 2014'te başarı gösteren bir derin öğrenme modelidir ve evrişim katmanlarında küçük boyutlu filtreler (3x3) kullanır. Model, 13 evrişim katmanı, 3 tam bağlantı katmanı ve 4x4 boyutunda 5 havuzlama katmanı içerir. Son katman, giriş verilerini sınıflandırmak için kullanılan bir softmax katmanıdır ve aktivasyon fonksiyonu olarak ReLU kullanılmıştır.

ResNet mimarisi, 2015 yılında He Kaiming tarafından önerilmiştir [97]. Derin öğrenme mimarilerinde katman sayısı arttığında bir noktaya kadar performans artarken bir noktadan sonra hızlıca düşüş yaşanmaktadır. ResNet mimarisi derinliği artırırken modelin performansını da korumuştur.

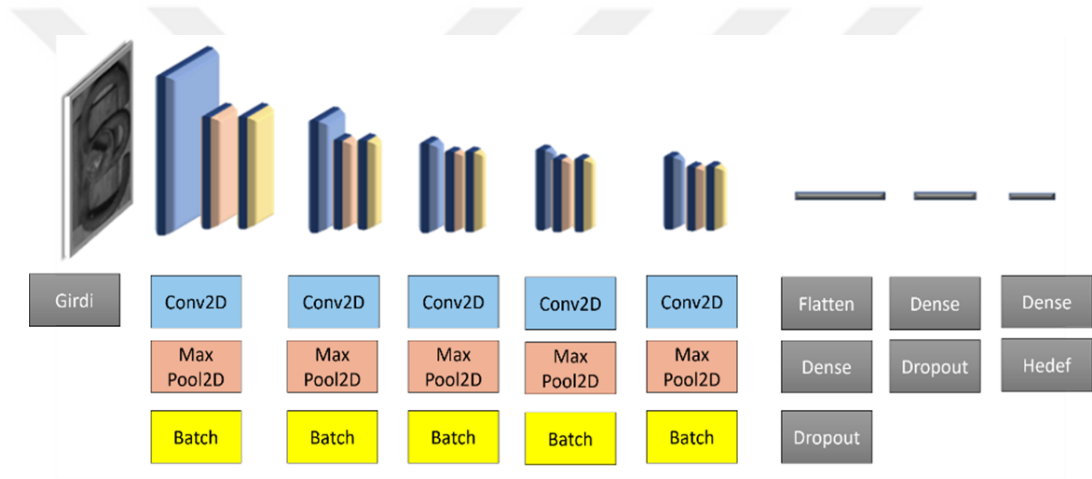
Bu modelde, hesaplama karmaşıklığı açısından önemli bir unsur olan parametre sayısı, diğer ResNet modellerine göre azaltılmıştır. ResNet50 modeli toplamda 180 katmandan oluşur ve aktivasyon fonksiyonu olarak hem Softmax hem de ReLU kullanılmaktadır.

Evrişimli sinir ağı (CNN), bilgisayarlı görme alanında kullanılan yaygın derin öğrenme yapılarından biridir. Hayvanların görsel sistemi örnek alınarak geliştirilen bu algoritma, görüntüleri girdi olarak almak üzere tasarlanmıştır ve bilgisayarlı görmeye etkin bir şekilde kullanılmaktadır. CNN yapısı; giriş katmanı, evrişim (convolution) katmanı, yığın normalleştirme (batch normalization) katmanı, düzleştirme (flatten) katmanı, havuzlama (pooling) katmanı, tam bağlantılı (fully connected veya dense) katman ve düğüm seyreltme (dropout) katmanından oluşur. CNN, giriş verisini alarak her bir katmanda işlem yapar ve eğitim sürecini yürütür. Sonuç olarak, doğru sonuç ile karşılaştırılmak üzere bir hedef çıktı üretir ve oluşan hata farkını azaltmak için geriye yayılım algoritmasını kullanır. Her iterasyonda ağırlıkların güncellenmesiyle hata miktarı azaltılır ve sınıflandırma doğruluğu artırılır. VGG-16 ve ResNet50 modellerinde olduğu gibi, CNN modelinde de Softmax ve ReLU aktivasyon fonksiyonları uygulanmıştır. Tablo 4.1'de gösterildiği üzere, CNN modeli ResNet50 ve VGG-16 modellerine kıyasla daha az sayıda parametre ve katman içermektedir. Geleneksel katman yapısına ek olarak yığın normalleştirme ve düğüm seyreltme katmanları dahil edilmiştir. Düğüm seyreltme (dropout) yöntemi, aşırı öğrenmeyi önleyerek modelin genelleme kabiliyetini artırır; tam bağlantılı katmandaki bazı düğümler arasındaki bağlantıları kopararak zayıf bilgilerin unutulmasını sağlar ve böylece eğitim performansını yükseltir. Yığın normalleştirme ise, evrişim katmanları arasında uygulanarak modelin düzenli hale gelmesine ve eğitim süresinin kısalmasına katkıda bulunur. Eğitim süresi, performans kaybı yaşanmaksızın %39 oranında azalmıştır. Önerilen CNN mimarisi Şekil 4.6'da gösterilmektedir.

Tablo 4.1. Önerilen modellerinin katman ve parametre sayıları

Ağ	Yıl	Katman Sayısı	Parametre Sayısı
VGG-16	2014	24	14M
ResNet50	2015	180	24M
Oluşturulan CNN	-	21	0.6M

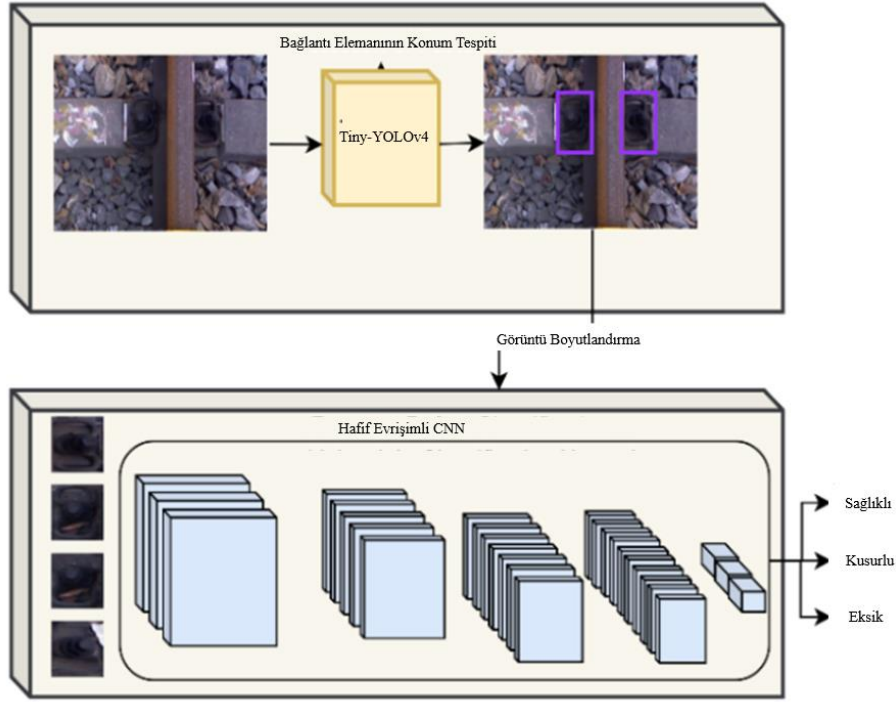
Tablo 4.2’de, kullanılan derin öğrenme tabanlı modellerin eğitim parametreleri yer almaktadır. Tabloya göre, seyreltme değeri başlangıçta 0.3 olarak belirlenmiş, ardından 0.5’e yükseltilmiştir. Bu artış, modellerin ezberleme riskini azaltmak amacıyla yapılmıştır. Seyreltme değerinin artırılmasına rağmen, model performansında herhangi bir düşüş yaşanmamıştır.

**Şekil 4.6.** Oluşturulan CNN mimarisi**Tablo 4.2.** Eğitim parametreleri

Model	Döngü Sayısı	Mini Batch Boyutu	Öğrenme Oranı	Seyreltme Değeri
VGG-16	20	8	1e-3	0.3, 0.5
ResNet50	20	8	1e-3	0.3, 0.5
Oluşturulan CNN	20	8	1e-3	0.3, 0.5

4.3. Bağlantı Elemanlarının Tespiti ve Hata Sınıflandırması için Önerilen Hibrit Yaklaşım

Önerilen hibrit yaklaşım, bağlantı elemanı kusur tespitini iki aşamada gerçekleştirir. İlk aşama bağlantı elemanının konumunun belirlenmesi, ikinci aşama ise bağlantı elemanı hata tespitidir. Önerilen yaklaşımın blok diyagramı Şekil 4.7’de verilmiştir.

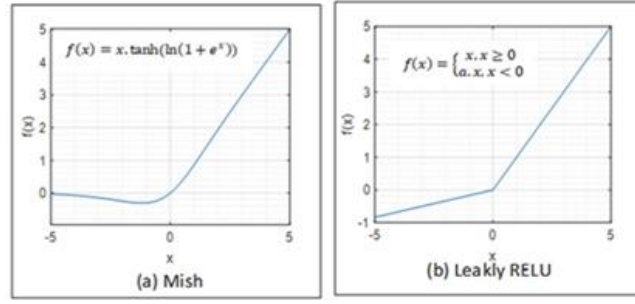


Şekil 4.7. Bağlantı elemanının kusur tespiti için önerilen hibrit çerçeve

Şekil 4.7’de önerilen yöntemin ilk adımında bağlantı elemanlarının konumları belirlenir. Demiryolu görüntüsündeki ray, travers ve balast gibi bileşenlerin karmaşık arka planı nedeniyle bağlantı elemanının tespiti zordur. Bu nedenle, bağlantı elemanının gerçek zamanlı ve yüksek doğrulukta tespiti için YOLOv4-Tiny kullanılacaktır [98]. Bağlantı elemanı durumunun tanınması için hafif bir evrişimli sinir ağı önerilmiştir. Ters çevrilmiş artık bloklar ve dikkat modülü sayesinde bu model hem boyut olarak daha küçüktür hem de daha hızlı tanıma gerçekleştirebilmektedir.

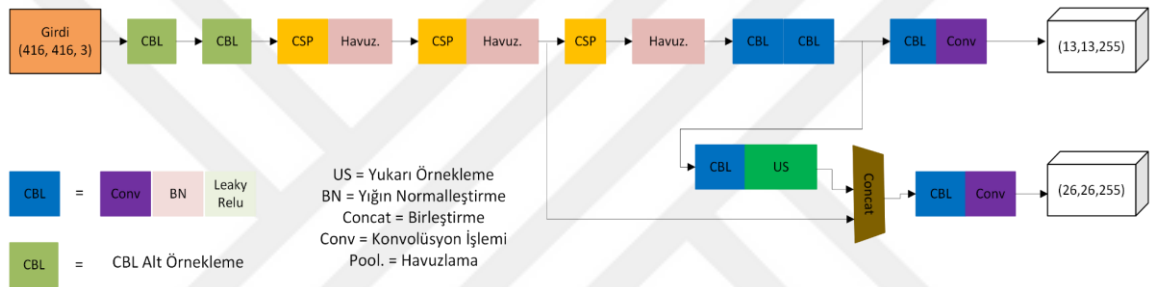
Bu kısımda bağlantı elemanının konumunu belirlemek için YOLOv4-Tiny modeli kullanılmaktadır. Bağlantı elemanı konumlandırma yöntemi, karmaşık arka planların varlığında hem hız hem de doğruluk açısından iyi sonuçlar vermelidir. YOLOv4-Tiny, YOLOv4 modeli üzerine tasarlanmış bir modeldir ve nesne tespiti için daha hızlı çalışır. Bu özelliği sayesinde özellikle gömülü sistemlerde ve mobil cihazlarda gerçek zamanlı olarak kullanılabilir.

YOLOv4-Tiny, CSPDarknet53 modülünü omurga olarak değiştirerek modifiye edilmiştir. Bu modül, Artık ağda ReSBlock yerine CSPBlock’u kullanır. CSPBlock’un hesaplama maliyetini azaltmak için darboğaz blokları kullanılmayarak bu maliyet azaltılır ve ağın doğruluğu ve öğrenme yeteneği artırılır. YOLOv4-Tiny’nin standart YOLOv4’e göre bir diğer farkı da CSPDarknet53’teki Mish aktivasyon fonksiyonu yerine Leaky Relu aktivasyon fonksiyonunu kullanmasıdır. Şekil 4.8’de iki aktivasyon fonksiyonu verilmiştir.



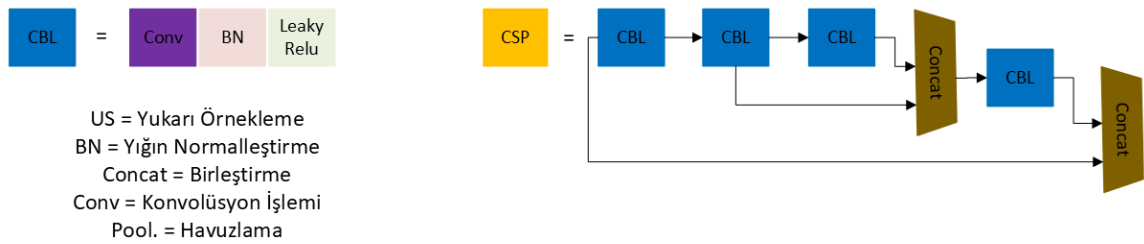
Şekil 4.8. YOLOv4'te ve YOLOv4-Tiny'de kullanılan aktivasyon fonksiyonları

Şekil 4.9'da gösterilen mimari, farklı ölçeklerde özellik haritaları elde etmek için piramit ağ yapısını kullanır.



Şekil 4.9. YOLOv4-Tiny mimarisi

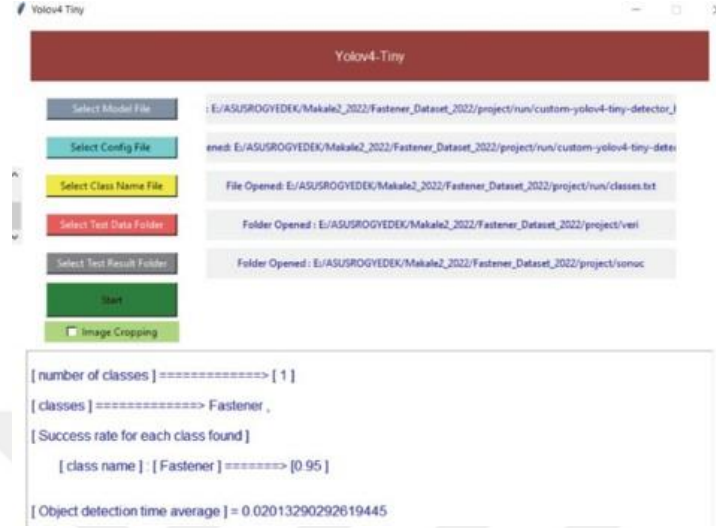
Şekil 4.9'daki yapı, tespit hızını artırır ve YOLOv4'te kullanılan uzaysal piramitten farklıdır. Ayrıca tahmin için 26x26 ve 13x13 olmak üzere 2 farklı ölçekte özellik haritaları kullanır. YOLOv4-Tiny modelinde kullanılan CSP bloğu Şekil 4.10'da verilmiştir.



Şekil 4.10. YOLOv4 Tiny'de CSP blok kullanımı

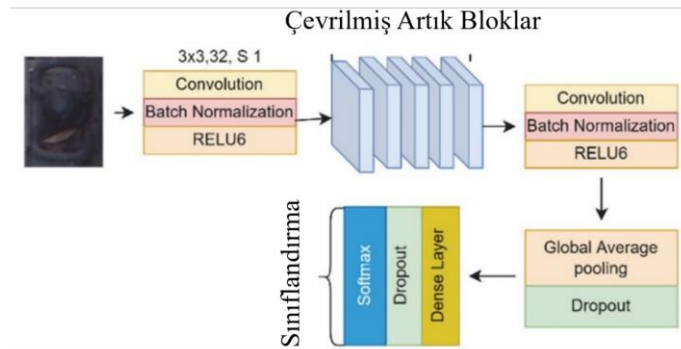
Şekil 4.10'daki CSP bloğu taban katmanını iki parçaya böler. İkinci parçaya belli bir konvolüsyon işlemi uygulandıktan sonra birinci parça ile birleştirilir. Yapının tabanında kalıntı bloklar bulunmaktadır. Bu bloklar algoritmanın hızını artırır. Bu çalışmada ray bağlantı

elemanlarının konum tespiti için YOLOv4-Tiny modeli kullanılmıştır. Bu amaçla bir arayüz hazırlanmıştır. Oluşturulan arayüz Şekil 4.11'de verilmiştir.



Şekil 4.11. YOLOv4-Tiny modelini test etmek için tasarlanan arayüz

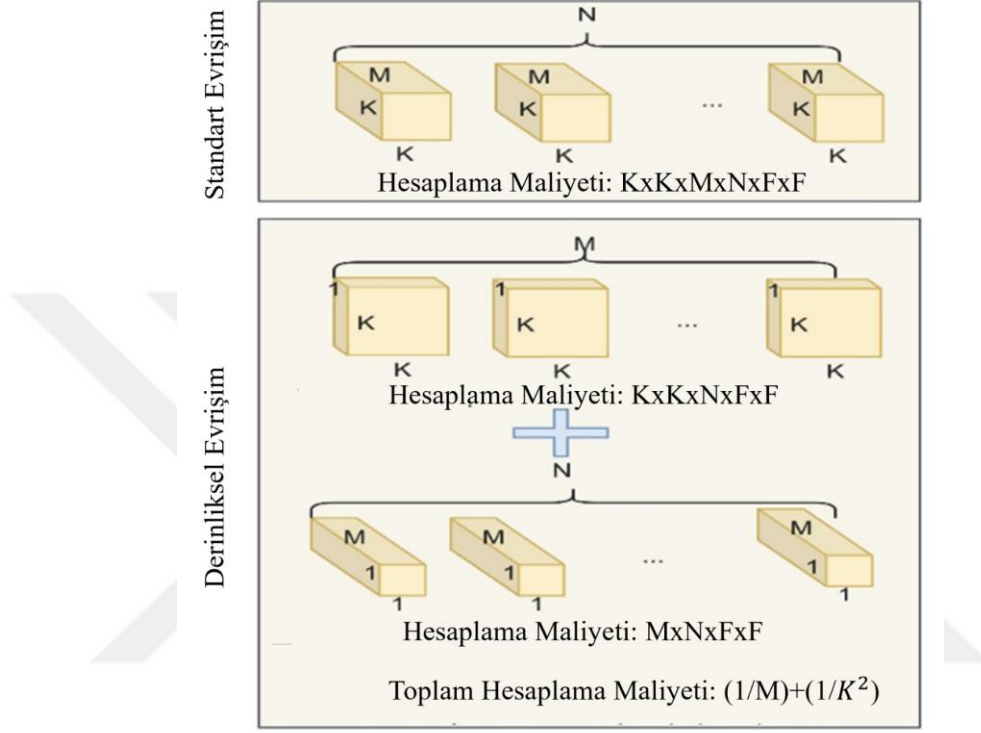
Şekil 4.11'de eğitilen model seçilir ve seçilen klasörün görüntülerinde bağlantı elemanları tespit edilir. Daha sonra tespit edilen bağlantı elemanları kırılır ve bir sonuç klasörüne yazılır. YOLOv4-Tiny modeli ile bağlantı elemanı tespit edildikten sonra bağlantı elemanındaki kusur tipinin tespiti için hafif bir ağ modeli önerilmiştir. Gerçek zamanlı olarak çalışacak, yüksek sınıflandırma doğruluğuna sahip bir ağ oluşturmak için ters çevrilmiş artık blokları kullanan düşük ağırlıklı bir ağ modeli önerilmiştir. Önerilen ağ modeli Şekil 4.12'de verilmiştir.



Şekil 4.12. Önerilen hafif CNN modeli

Şekil 4.12'de önerilen yöntem hafif bir ağ modelinden oluşmaktadır. Önerilen model, derinlemesine evrişim katmanları kullanarak ters çevrilmiş artık bloklara sahiptir. Derinlemesine

evrişim katmanları ilk olarak Xception modelinde tanıtıldı ve hafif ağlarda parametre sayısını azaltmak için kullanıldı [99]. Derinlik açısından ayrılabilir evrişim işlemi, her kanal için 3x3 tek kanallı bir evrişim işlemi gerçekleştirir ve bunu 1x1 M-kanallı noktasal evrişim işlemi izler. Şekil 4.13 yapıyı göstermektedir.

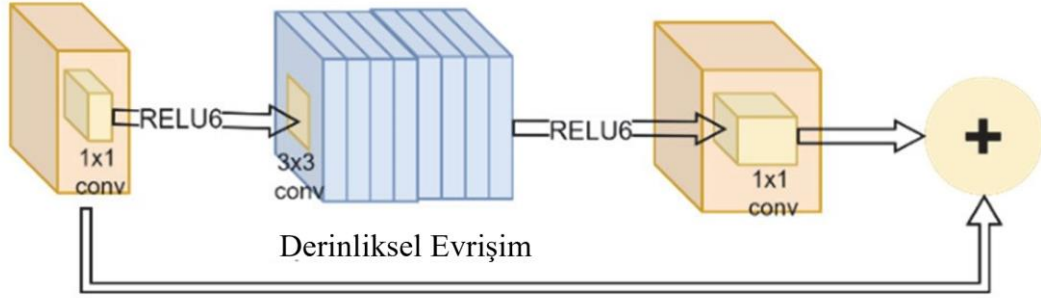


Şekil 4.13. Standart ve derinlemesine ayrılabilir evrişimler

Şekil 4.13'te, $F \times F$ boyutunda bir görüntü üzerinde derinlemesine ayrılabilir evrişim kullanıldığında, hesaplama maliyeti standart bir evrişim katmanının kullanılmasından 10 kat daha azdır. Birçok evrişimli sinir ağında, her evrişim işleminden sonra RELU aktivasyonu uygulanarak ağın gücü artırılır. Bu işlem karmaşık ağlarda iyi çalışsa da daha küçük boyutlu ağlarda veri kaybına neden olur. Bu nedenle konvolüsyon katmanından sonra verilerin doğrudan diğer katmana aktarılmasını sağlayan doğrusal darboğaz işlemi, kaçırılan özelliklerin RELU ile kullanılmasına olanak sağlar. MobileNetv2 ağından sonra ters çevrilmiş artık bloklar kullanıldı [100]. Bu bloklar üç evrişim işleminden oluşur. Bu katmanlar, özellik boyutunu artıran darboğaz bloğu, düşük ağırlıklı özellik çıkarımı sağlayan derinlikli evrişim katmanı ve noktasal evrişim katmanından oluşur. Bu yapı Şekil 4.14'te gösterilmektedir.

Şekil 4.14'te verilen model, geleneksel artık bloklara göre daha yüksek doğrulukta sonuçlar sağlar. Geleneksel artık blokların boyut küçültmeden sonra özellikleri çıkarması nedeniyle birçok özellik kaybolur. Tersine, ters çevrilmiş artık bloklar, özellik çıkarmadan önceki boyutu artırır ve

derinlik açısından evrişim katmanları nedeniyle daha az sayıda parametreye sahiptir. Ayrıca global havuzlama katmanından sonra modele yoğun bir katman da dahil edilmektedir. Önerilen modelin yapısı ve parametreleri Tablo 4.3'te verilmiştir.



Şekil 4.14. Ters çevrilmiş artık blok

Tablo 4.3'teki t değeri, giriş tensörüne uygulanan kanal genişletme faktörüdür. Örneğin t değeri üç alınırsa ve giriş tensörü 16 kanaldan oluşuyorsa çıkış $3 \times 16 = 48$ olur. n değeri blokların kaç kez tekrarlanacağını gösterir. s değeri evrişim işlemindeki adım parametresidir.

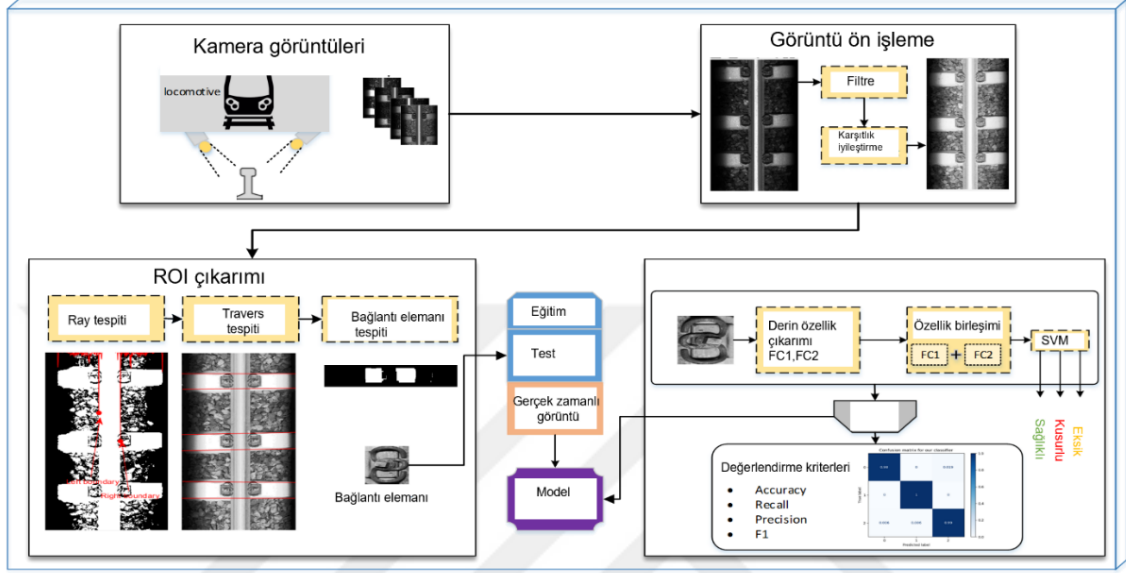
Tablo 4.3. Önerilen CNN'in yapısı ve parametreleri

Filtre Sayısı	Katman	t	n	s
32	Convolution2d 3	-	1	1
32	Ters Artık	1	1	1
8	Ters Artık	6	1	2
16	Ters Artık	6	2	1
32	Ters Artık	6	3	2
64	Ters Artık	6	2	1
256	Global Avaraj	-	-	-
512	Dense	-	-	-

4.4. Demiryolu Bağlantı Elemanları Üzerindeki Kusurların Tespit Edilmesi

Bu uygulamada, ray bağlantı elemanlarındaki kusurların sınıflandırılması için yeni bir yaklaşım önerilmiştir. Bağlantı elemanı kusurlarını belirlemek için önerilen yöntem dört önemli adımdan oluşmaktadır. İlk adımda, bir kamera yardımıyla demiryollarından görüntüler toplanmıştır. Ray görüntüleri genellikle trenin altına monte edilmiş bir kameradan alındığı için aydınlatma koşulları iyi değildir. Bu nedenle, demiryolu görüntüleri çok fazla gürültü içermektedir. İkinci adımda, yüksek kontrastlı bir ray görüntüsü elde etmek için görüntü iyileştirme yöntemi uygulanmıştır. Elde edilen görüntüde gürültüler ve ışık efektleri ortadan kaldırılmıştır. Üçüncü adımda, bağlantı elemanı bileşeni, ray görüntülerinde kesin olarak algılanmıştır. Ray ve travers

konumu belirlendikten sonra travers kısmına LLBP uygulanarak bağlantı elemanının konumu elde edilir. Son adımda, önerilen hafif evrişimli sinir ağlarından elde edilen öznitelikler kullanılarak sağlıklı, deforme olmuş ve eksik bağlantı elemanları belirlenmiştir. Önerilen yöntemin blok diyagramı Şekil 4.15'te verilmiştir.



Şekil 4.15. Önerilen sistemin blok şeması

Şekil 4.15'te öncelikle ray, travers ve bağlantı elemanından oluşan ray bileşenlerinin görüntüleri tren altına monte edilmiş yüksek hızlı bir kamera ile toplanmıştır. Daha sonra ray görüntüsü üzerinde görüntü iyileştirme işlemi yapılmış ve gürültüler giderilmiştir. Üçüncü aşamada öncelikle ray ve travers bloklarının konumları belirlenmiştir. Daha sonra belirlenen traversleri içeren görüntülere LLBP algoritması uygulanmıştır. LLBP algoritması bağlantı elemanı bileşenlerini ortaya çıkarmasına rağmen, delikleri kapatmak için morfolojik operatör uygulanmaktadır. Son aşamada, elde edilen bağlantı elemanı bileşenleri geliştirilen evrişimsel sinir ağına verilmiş ve elde edilen öznitelikler destek vektör makineleri ile sınıflandırılmıştır.

Renkli bir görüntü, Kırmızı (R), Yeşil (G) ve Mavi (B) olarak adlandırılan üç bileşenden oluşur. Bu matrislerin her biri, görüntüyle ilişkili yoğunluk değerlerini tutar. Görüntüdeki herhangi bir (x, y) konumundaki pikselin değeri, R (x, y), G (x, y) ve B (x, y) içindeki değerlerle temsil edilir. Renkli bir görüntüyü gri bir görüntüye dönüştürürken genellikle üç katsayı kullanılır. Bu katsayıların k_r , k_b ve k_g değerleri genellikle $k_r = 0.2989$, $k_g = 0.5870$ ve $k_b = 0.1140$ olarak seçilir. Genel olarak, gri görüntüye dönüştürmek için Denklem 4.10 kullanılır.

$$I(x, y) = k_r R(x, y) + k_b B(x, y) + k_g G(x, y) \quad (4.10)$$

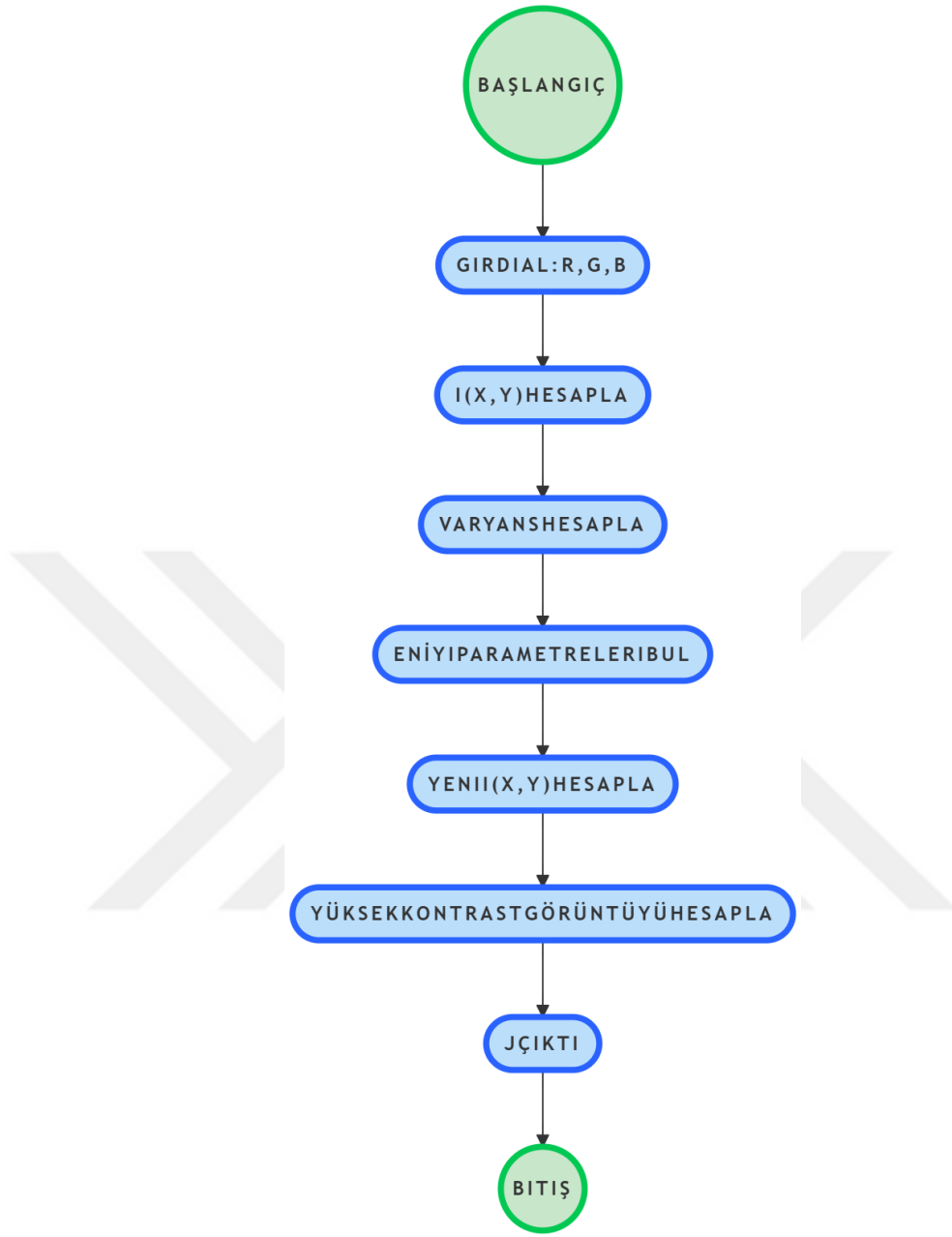
Trenin alt kısmından elde edilen görüntülerin kontrast değeri düşük olduğu için ilk adımda görüntü iyileştirme işlemi uygulanacaktır. Yüksek kontrastlı ve düşük homojenliğe sahip bir görüntü elde etmek için uygun katsayılar belirlenmelidir [101]. Denklem 4.10'da bu genellemeyi kaybetmeden k_b değeri 1 olarak seçilebilir. Homojenliği düşük bir görüntü elde etmek için yüksek varyansa sahip monokrom bir görüntü oluşturulmalıdır. İlk olarak, görüntünün piksel değerleri Denklem 4.11'e göre 0 ile 1 arasında olacak şekilde normalize edilir

$$J(x, y) = \frac{I(x, y) - I_{min}}{I_{max} - I_{min}} \quad (4.11)$$

Denklem 4.11'deki, I_{max} ve I_{min} değerleri, sırasıyla monokrom görüntünün maksimum ve minimum değerlerini temsil eder. Görüntünün kontrast değerini artırmak için homojenliğinin düşük olması gerekir. Bu nedenle görüntünün standart sapmasını maksimize edecek k_r ve k_g değerlerinin bulunması gerekmektedir [102]. Bu amaç fonksiyonu Denklem 4.12'de verilmiştir.

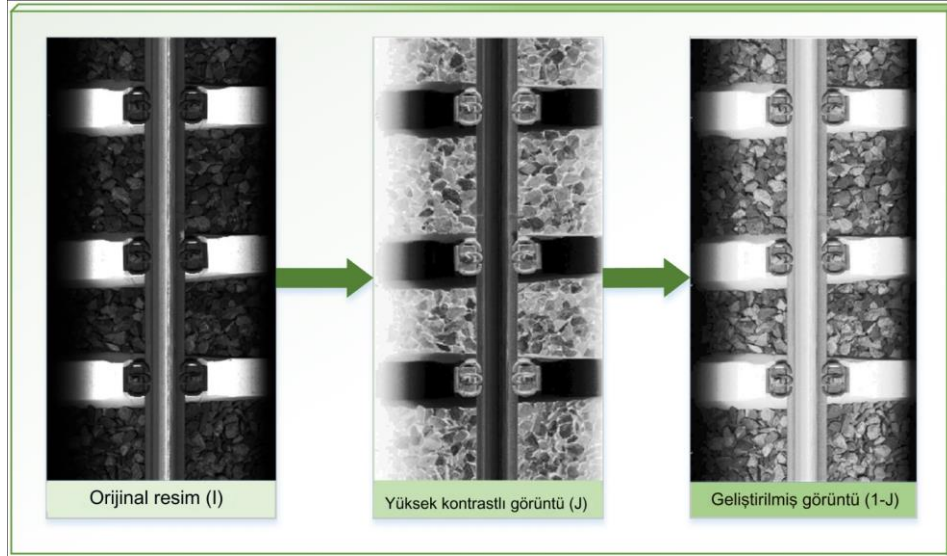
$$(\hat{k}_r, \hat{k}_g) = \arg \max_{k_r, k_g} (\sigma_f^2) \quad (4.12)$$

Denklem 4.12'de verilen standart sapmayı maksimize etmek için Nelder-Mead Simplex yöntemi kullanılmıştır. Bunun için σ_f^2 değerini minimum yapan k_r ve k_g değerleri bulunmalıdır. Nelder-Mead yöntemi doğrusal olmayan sınırsız bir optimizasyon yöntemidir ve Matlab'ın bu işlemi gerçekleştiren `fminsearch` yöntemi vardır. Yüksek kontrastlı bir görüntü elde etmek için önerilen yöntem Şekil 4.16'da verilmiştir.



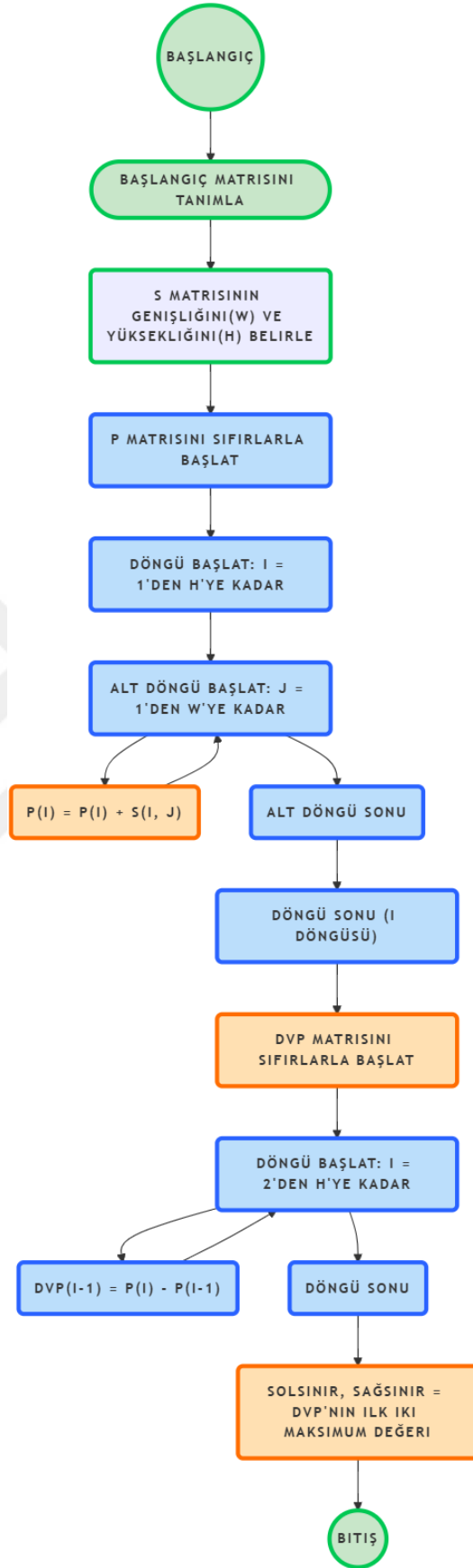
Şekil 4.16. Yüksek kontrastlı görüntüyü hesaplaması

Şekil 4.16'da renkli görüntü gri görüntüye dönüştürülürken, yüksek kontrastlı görüntü elde etmek için uygun katsayılar hesaplanmıştır. Bu katsayılar standart sapmayı maksimize edecek şekilde seçilir. Yüksek kontrastlı görüntü ve geliştirilmiş görüntü, bir ray görüntüsü için Şekil 4.17'de verilmiştir.



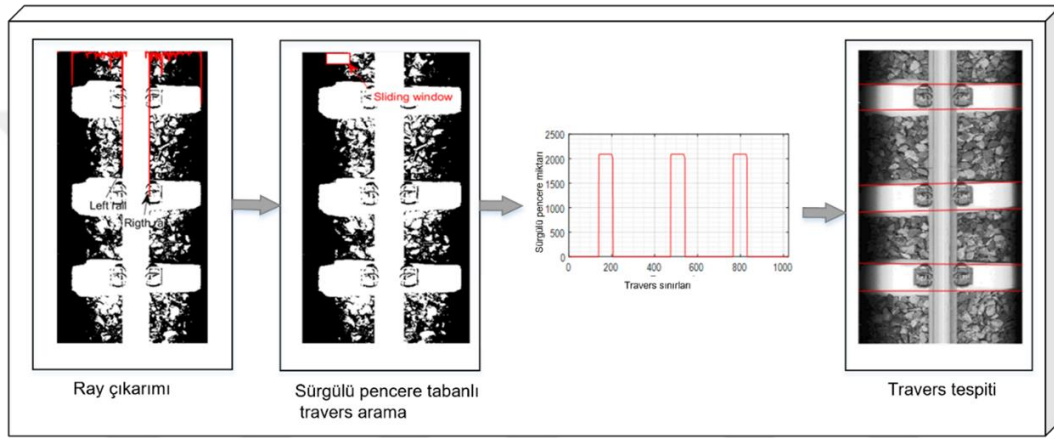
Şekil 4.17. Ray ve travers algılamasının gösterimi

Şekil 4.17’de ilk olarak orijinal görüntü yüksek kontrastlı bir görüntüye dönüştürülmüştür. Bu görüntünün piksel değerleri 0 ile 1 arasında değerlerden oluşmaktadır. Görüntünün tersini elde etmek için tüm piksel değerlerini tek bir değerden çıkarmak gerekir. Bu hesaplama sonucunda kontrast iyileştirilmiş ve ışıklandırma problemi olmayan bir görüntü elde edilmiştir. Bu işlem, bir sonraki adımda bölütleme için ray bileşenlerini daha kolay belirlememizi sağlamaktadır. Kontrastlı görüntüler elde edildikten sonra ray ve travers bileşenlerinin algılanması gerekmektedir. Bu amaçla elde edilen görüntü ikili görüntüye dönüştürülmüştür. Otsu yöntemi ile bir eşik noktası belirlenerek monokrom görüntü ikili görüntüye dönüştürülür [103]. Daha sonra yatay iz düşüm yöntemi ile ray konumu belirlenir. Sol ve sağ rayı elde etmek için Şekil 4.18’de gösterilen bir ray çıkarma algoritması uygulanmıştır.



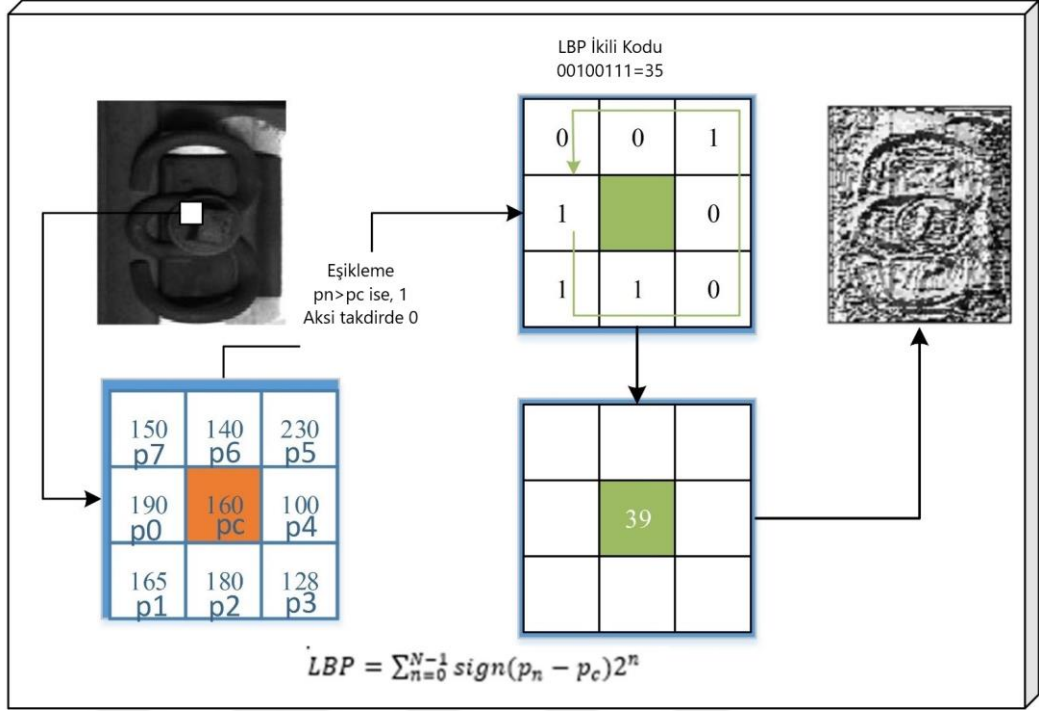
Şekil 4.18. Ray çıkarma algoritması

Şekil 4.18’de $S(i, j)$, (x, y) konumundaki parçalı görüntünün piksel değerini temsil etmektedir. Parçalı görüntü, 0 veya 1’lerden oluşan ikili bir görüntüdür. Daha sonra elde edilen $P(j)$ sinyalinin ardışık farkları hesaplanır. DVP olarak elde edilen ardışık farklar sinyalinde en büyük iki değer rayın kenarlarını gösterir. Travers bileşeninin tespiti, ray bileşeninden biraz farklıdır. Bunun nedeni, görüntüde birden fazla travers olmasıdır. Her traversin başlangıç ve bitiş noktası, bağlantı elemanının bu bölgede olabileceği ROI’yi ifade eder. Her travers için başlangıç ve bitiş konumları, rayın sol tarafına yakın bir noktadan yukarıdan aşağıya kayan bir pencere ile belirlenir. Ray ve travers yerleşimlerinin çizimleri Şekil 4.19’da verilmiştir.



Şekil 4.19. Ray ve travers tespitinin çizimi

Şekil 4.19’da görüldüğü üzere ray ve travers parçaları belirlendikten sonra, görselde yer alan bağlantı elemanlarının konumları da tespit edilmelidir. Bu işlem için yerel ikili örüntü yöntemi kullanılmıştır. Klasik yerel ikili örüntü yöntemi, merkezdeki pikseli komşu piksellerle karşılaştırarak bir değer oluşturur. Merkez pikselin değeri " p_c " ve komşu pikselin değeri " p_n " olarak ele alınırsa, eğer $p_n > p_c$ ise, ilgili piksele 1 değeri verilir; aksi halde 0 değeri atanır [104]. Daha sonra, merkez pikselin etrafındaki her bir bitin ağırlıklı toplamı alınarak, merkez pikselin değeri hesaplanır. Yerel ikili örüntü yönteminin detayları Şekil 4.20’de gösterilmiştir.

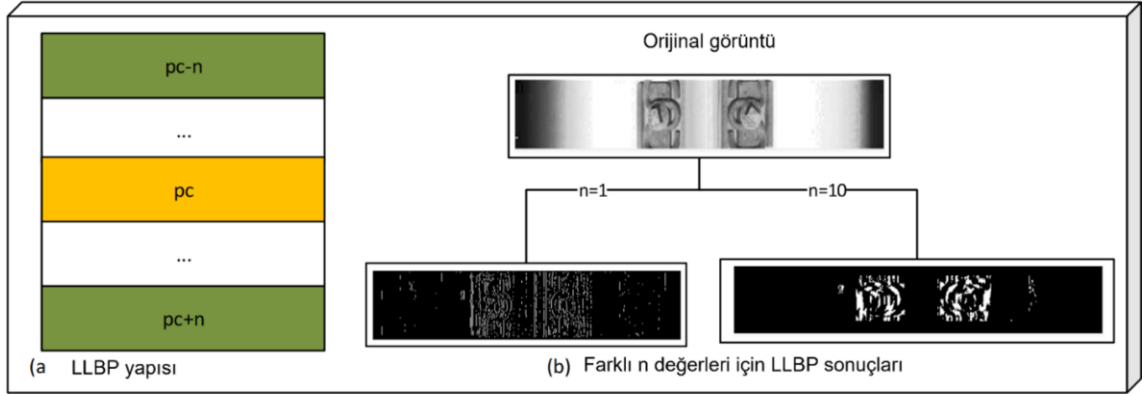


Şekil 4.20. Yerel ikili örüntü yöntemi

Şekil 4.20'deki yerel ikili örüntü ile elde edilen görüntüde bağlantı elemanını tespit etmek oldukça zordur. Bu nedenle bu çalışmada LLBP yöntemi kullanılmıştır. Bu yöntemin farkı, çizgi yönünde karşılaştırma yapmasıdır [105]. Fan ve ark. [106], Bağlantı elemanı tespiti için LLBP'yi kullanmışlardır. Ancak onların önerdiği LLBP yöntemi görüntünün tamamına uygulanmakta ve balast ve ray gibi bileşenleri içerdiğinden bağlantı elemanını doğrudan tespit etmede zorlamıştır. Bu nedenle, bağlantı elemanı bileşenini aramak için ayrı bir sürgülü pencere filtresi kullanılmıştır. Bu filtrenin boyutunun ayarlanması gerekmektedir. Bizim yöntemimizde LLBP sadece traversin bulunduğu bölgeye uygulanır ve bağlantı elemanlarının tespiti için herhangi bir arama işlemi gerektirmez. Bu yöntemin matematiksel Denklem 4.13'te verilmiştir.

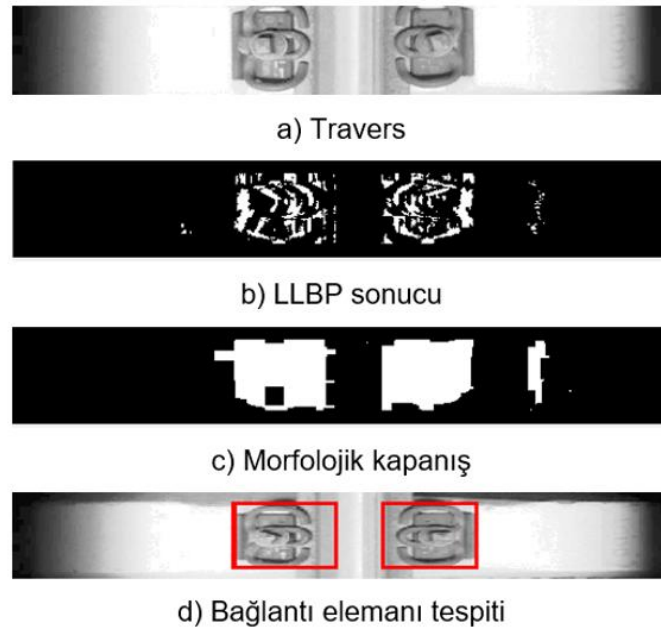
$$LLBP = sign(p_{(c-n)} - p_c) \times sign(p_{(c+n)} - p_c) \quad (4.13)$$

Denklem 4.13 p_c ve $p_{c \pm n}$ merkezi pikseli ve iki komşusunu gösterir. LLBP'nin sadece n parametresi vardır ve bu parametre ne kadar büyük seçilirse, bağlantı elemanı o kadar geniş görünür. Küçük n değerleri çok fazla gürültü içerdiğinden bağlantı elemanını belirlemek zordur. Şekil 4.21, LLBP yöntemini ve bunun farklı n parametreleri için uygulamasını göstermektedir.



Şekil 4.21. LLBP yapısı ve farklı n değerleri için uygulaması

Şekil 4.21’de görüldüğü gibi n değeri küçük seçildiğinde tamamen gürültülü bir görüntü elde edilmekte ve gürültülü piksel değerleri bağlantı elemanının tespit edilmesini zorlaştırmaktadır. Öte yandan, n değeri yüksek seçildiğinde gürültü azalır ve bağlantı elemanı tespiti kolaylaşır. Travers üzerindeki bağlantı elemanı belirlendikten sonra görüntü konturu çizilerek kenar noktaları bulunur. Şekil 4.22, travers algılamasından sonra bağlantı elemanlarının LLBP yöntemiyle nasıl algılandığını göstermektedir.



Şekil 4.22. Sırasıyla bağlantı elemanı algılama adımları

Şekil 4.22’de görüldüğü gibi travers bölgesi elde edildikten sonra bağlantı elemanı bileşenlerini elde etmek için bu bölgeye LLBP yöntemi uygulanmaktadır. Bölütleme tabanlı teknikler kullanıldığında, özellikle travers taraflarındaki gölgeler nedeniyle gürültülü bölgeler

nesne olarak alınır. LLBP yönteminin n parametresi uygun şekilde seçildiğinde sadece bağlantı elemanı bileşeni elde edilir. Bu çalışmada n 'nin en uygun değeri deneme yanılma yoluyla 10 olarak belirlenmiştir.

Evrişim katmanı, bir CNN mimarisinin en temel ögesidir ve özellik çıkarma ve doğrusal olmayan işleme için kullanılır. CNN'nin ilk katmanlarında bir evrişim katmanı kullanıldığında, giriş görüntüsünden kenarlar, çizgiler ve köşeler gibi düşük anlamlı özellikleri çıkarılır. Evrişim katmanı ağırlı uçlarına doğru kullanıldığında, görüntüdeki nesnenin görünürlüğü ve şekli ortaya çıkar [107]. Evrişim, genellikle bir filtreyi görüntü üzerinde hareket ettirerek merkez piksel değerlerinin hesaplanmasıdır. Bir I görüntüsü üzerinde w süzgeci ile evrişim işlemi Denklem 4.14'te verilmiştir.

$$C(w * I)_{x,y} = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x + s, y + t) \quad (4.14)$$

Giriş görüntüsüne evrişim işlemi uygulanırken birçok filtre kullanılarak bir eşleme yapılır ve elde edilen eşlemelerin birleştirilmesiyle ağırlıklı toplam hesaplanır. Evrişimden kaynaklanan negatif değerler, ağırlıklı toplamın elde edilen görüntüsüne doğrusal olmayan bir işlem uygulayarak elimine edilir. Doğrusal olmayan operatörler, sigmoid, tanjant hiperbolik ($\tanh$) ve doğrultulmuş doğrusal birimler (ReLU) gibi farklı şekillerde uygulanır. En çok kullanılan doğrusal olmayan işlem genellikle ReLU'dur ve bir x değeri üzerindeki matematiksel yorumu $y = \max(0, x)$ ile ifade edilir. Havuzlama katmanı, ReLU işleminden sonra uygulanır ve görüntü boyutunu aşağı örneklemek için kullanılır. Maksimum havuzlama bu amaç için en çok kullanıldır ve özellik eşlemeli görüntüler farklı maksimum değerlere alt örneklenir. Böylece hem hesaplama karmaşıklığı hem de parametre sayısı azaltılmış olur.

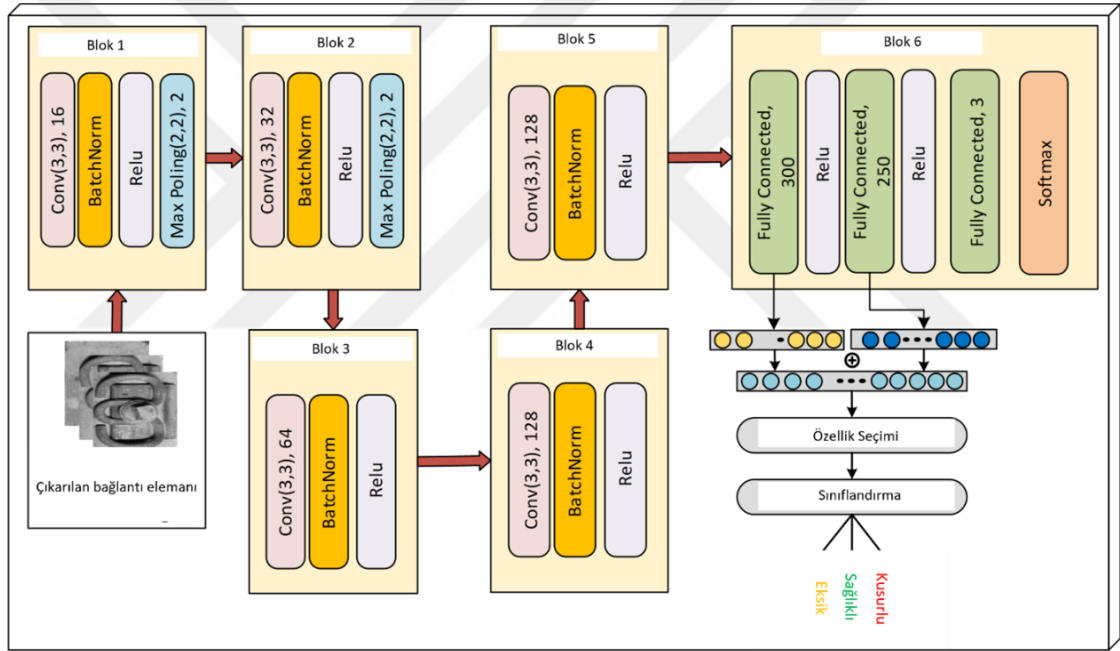
Tam bağlantılı katman, ileri beslemeli sinir ağlarına benzer ve birkaç tam bağlı katman, evrişimli bir sinir ağının uçlarına doğru yerleştirilir. Tam bağlantılı katman, önceki katmanlardan bilgi toplayarak yüksek düzeyde soyutlama sağlar. Burada çok fazla parametre olduğu için aşırı uyum sorununa yol açmaktadır. Bu nedenle bazı bağlantıları kesilir. Tam bağlantılı katmandan alınan çıktılardan sınıf tahmin puanları üretmek için Softmax fonksiyonu kullanılır. Bu süreç Denklem 4.15'te görülmektedir.

$$\sigma_{softmax}(z^{(i)}) = \frac{e^{z^{(i)}}}{\sum_{k=1}^c e^{z_k^{(i)}}} \quad (4.15)$$

Denklem 4.15'te, her $z^{(i)}$ girişi için $\sigma_{softmax}$ hesaplanır. CNN'nin eğitim süresini azaltmak için toplu normalizasyon katmanı kullanılır. Bu katman, ağ başlatma kesinliği için de gereklidir. Normalizasyon işlemi Denklem 4.16'daki gibi yapılmaktadır.

$$x_{normalized} = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} \quad (4.16)$$

Denklem 4.16'da mini parti ortalaması ve varyans sırasıyla μ ve σ ile gösterilmiştir. ε parametresi, sıfıra bölmeyi önlemek için seçilen küçük bir eşik değeridir. Bu aşamada, ROI çıkarılan bağlantı elemanlarının detaylı sınıfının tanınması amaçlanmaktadır. Yüksek sınıflandırma doğruluğu ve daha az hesaplama süresi elde etmek için derin bir sinir ağı modeli önerilmiştir. Küçük boyutlu veri kümelerini eğitmek için genellikle önceden eğitilmiş ağlara bir aktarım öğrenme algoritması uygulanır. Ancak bu ağlar birçok katmana ve karmaşık bir yapıya sahiptir. Bu nedenle, bu çalışmada daha küçük boyutlu bir ağ modeli önerilmiştir. Giriş katmanından alınan görüntülere sırasıyla evrişim, toplu normalleştirme, Relu ve havuzlama işlemleri uygulanmıştır. Bu işlem beş kez tekrarlandıktan sonra ağın sonunda tam bağlantılı 3 katman kullanılmıştır. Tam bağlantılı iki katmandan elde edilen öznitelikler birleştirilir ve girdi olarak makine öğrenme tabanlı sınıflandırıcılara verilmiştir. Önerilen yöntemin blok diyagramı Şekil 4.23'te verilmiştir.



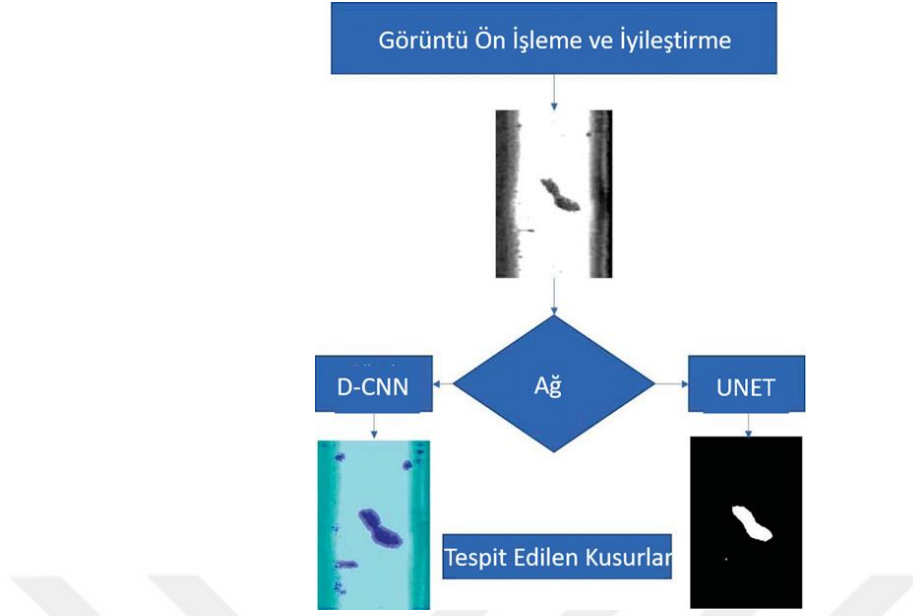
Şekil 4.23. LCNN mimarisi ve sınıflandırması

Şekil 4.23'te gösterilen evrişimli sinir ağı 26 katmana sahiptir. Öznitelik sayısı ilk tam bağlı katmanda 300, sonraki tam bağlı katmanda 250 olarak belirlenmiştir. Bu özellik numaraları deneylerle bulunmuştur. Son tam bağlı katmandaki öznitelik sayısı, sınıf sayısını gösteren 3'tür. İlk iki katmanda elde edilen özellikler birleştirilmiştir. Daha sonra sınıflama modeline verilerek bağlantı elemanı durumu belirlenmiştir. Sınıflandırma amacıyla farklı modeller test edilecek ve en iyi sınıflandırıcı seçilecektir. Önerilen CNN tabanlı hata tespit sisteminin performansı, bir ölçüm dizisinden alınan bağlantı elemanı görüntüleri kullanılarak değerlendirilmektedir.

4.5. Anlamsal Bölütleme Kullanarak Ray Üzerindeki Kusurları Tespit Etme

Bu uygulamada ray üzerindeki kusurları tespit etmek için görüntü işleme tabanlı bir model önerilmiştir. Yapay zekâ alanının alt dallarından biri olan derin öğrenme yöntemleri ile ray görüntülerinden otonom bir şekilde kusurları tespit etmek oldukça mümkündür. Derin öğrenme teknikleri, geleneksel tekniklere kıyasla probleme özgü özelliklerin otomatik olarak öğrenilmesine yardımcı olur. Yani derin öğrenme, makine öğrenmesinde tanımlanması gereken parametreleri keşfeder ve makine öğrenmesine göre daha doğru sonuçlar vermektedir. Evrişimli sinir ağları, derin öğrenme mantığına dayanmaktadır. Normalde, evrişimli sinir ağları gibi geniş özellikli öğrenme ağlarının eğitim için büyük veri kümelerine sahip olması gerekmektedir. Bir diğer önemli nokta ise klasik evrişimli sinir ağlarında genel görüntü eğitiminin sınıf etiketleri üzerinden yapılmasıdır. Ancak, bazı problemler için piksel tabanlı yaklaşımlar gerekmektedir. Sağlık veya güvenlik alanları gibi hassas yaklaşım gerektiren alanlarda her pikselin sınıf bilgisine ihtiyaç duyulmaktadır. Görüntü bölütleme burada devreye girmektedir. Görüntü bölütleme, bir görüntüyü her biri içerisinde farklı özelliklerin tutulduğu anlamlı bölgelere ayırmak olarak tarif edilebilir [108]. Görüntüyü anlamsal olarak bölümlere ayırır. Temel olarak, iki farklı bölütleme tekniği vardır. Bunlar anlamsal bölümlendirme ve örnek bölümlendirme. Bu çalışmada anlamsal bölütleme yöntemleri kullanılmıştır. Anlamsal bölütleme, bir görüntüdeki her pikselin hangi sınıfa ait olduğunu belirler [108].

Bu bölümde, ray üzerindeki kusurları tespit için oluşturulan anlamsal bölütleme ağımları anlatılmaktadır. Bölütleme modeli, Unet ve D-CNN olmak üzere iki kısımdan oluşmaktadır. Veri setindeki görüntüler, bölütlemeye önce görüntü ön işleme ve iyileştirme adımlarından geçmiştir. Bu aşamada veri setindeki ray görüntüleri kırılmış ve boyutlandırılmıştır. Önerilen model Şekil 4.24'te verilmiştir.



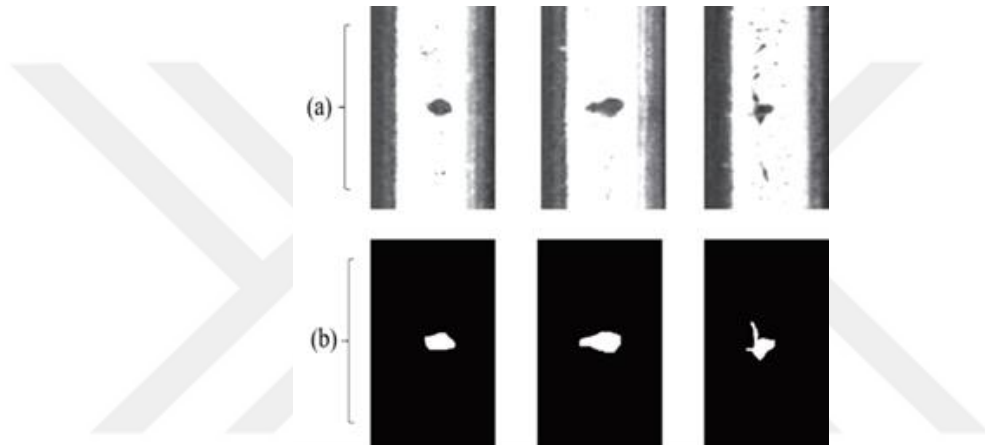
Şekil 4.24. Önerilen anlamsal bölütleme modeli

Çalışma, genişletilmiş konvolüsyonlara dayalı basit bir anlamsal bölütleme ağı kullanmaktadır. İlk olarak, veri seti eğitilir ve her bir etiket için piksel değerleri alınır. Görüntüler arka plan ve kusur etiketlerinden oluşmaktadır. Piksel etiketlerinin çoğu arka plan olarak etiketlenmiştir. Bu durum sınıflar arasında dengesizliğe yol açabilmektedir. Bunu düzeltmek ve sınıfları dengelemek için ters frekans sınıf ağırlıklandırması kullanılmıştır. İlk olarak, girdi görüntülerinin boyutuna karşılık gelen girdi katmanı kullanılarak piksel sınıflandırması için bir ağ oluşturulmuştur. İkinci katmanda genişleme faktörü 1 olan 3x3 boyutunda 32 filtre içeren Relu katmanı kullanılmaktadır. Genişleme faktörü 2 ve 4 olarak 2 Relu katmanının daha kullanılmıştır. Genişleme faktörü çözünürlük kaybı yaşamadan 3x3 boyutlu filtreyi sırasıyla 7x7 ve 15x15 şeklinde güncelleştirir. Daha sonra Softmax katmanı ve ters sınıf ağırlıklarına sahip bir pixelClassification katmanı kullanılmıştır. Daha sonra, eğitilmiş ağın orijinal ve referans görüntüler anlamsal bölütleme kullanılarak test edilmiştir.

Unet, otomatik bir anlamsal bölütleme yöntemidir. Unet, evrişimli sinir ağı katmanlarından oluşturulan farklı mimari ve piksel tabanlı görüntü bölütleme açısından klasik modellere göre daha başarılı sonuçlar vermektedir. Ayrıca, az sayıda eğitim görüntüsü olsa bile iyi sonuçlar vermektedir. Unet, adını U harfine benzer mimarisinden almaktadır. Unet 24 katmandan oluşmaktadır. İki kısımdan oluşmaktadır. Bunlar kodlayıcı ve kod çözücü kısımlarıdır. Giriş görüntüleri, çıkışta çıkış haritası olarak elde edilir. Kodlayıcı kısmı, tipik evrişimli ağ karakterini göstermektedir. Kodlayıcı kısmı, birden çok iki 3x3 konvolüsyon katmanı içermektedir. Her konvolüsyonu, rektifiye edilmiş bir Relu [109] ile 2x2 havuzlama katmanı takip etmiştir. Kod çözücü kısım, 2x2 yukarı konvolüsyon ile tekrarlanan üst örneklemeden oluşmaktadır. Her yukarı örnekleme iki 3x3

konvolüsyon katmanı ve bir Relu katmanı takip etmektedir. Son katman ise, 1x1 konvolüsyon katmanıdır. Ağ için ezberle durumu ortadan kaldırılmış ve değerlendirme ve tahmin için uygun hale gelmiştir.

Veri setindeki ham görüntüler DATEM'den alınmıştır. Ham görüntüler bölütleme işleminden önce görüntü ön işleme ve iyileştirme süreçlerinden geçmiştir. Kullanılan veri seti, 603 tane kusurlu ray görüntüsünden oluşmaktadır. Bu görüntülerin 453'ü eğitim için, 150'si ise test için kullanılmıştır. Eğitim ve test veri setlerinde orijinal gri tonlamalı görüntülerin yanı sıra referans (ground truth) görüntüleri de mevcuttur. Kullanılan veri setindeki örnek kusurlu ray görüntüleri Şekil 4.25'te gösterilmiştir.



Şekil 4.25. Veri setindeki görüntüler (a) Kusurlu ray görüntüleri (b) Kusurlu raylara ait referans görüntüler

4.6. SAM Kullanarak Demiryolu Raylarının Bölütlenmesi

Bu çalışmada demiryolu raylarının bölütlemesi için Meta AI tarafından geliştirilen Segment Anything Model (SAM) kullanılmıştır. SAM, belirli bir eğitim verisine ihtiyaç duymadan görüntülerdeki nesneleri ayırabilen yenilikçi bir bölütleme modelidir. Çalışmanın akış şeması Şekil 4.26'da görülmektedir.



Şekil 4.26. SAM çalışmasının akış şeması

SAM modeli üç temel bileşenden oluşmaktadır: Görüntü Kodlayıcı, İstem Kodlayıcı ve Maske Çözücü. Görüntü Kodlayıcı, SAM'in ana bileşenlerinden biri olup, girdi olarak aldığı

görüntüyü işleyip sabit uzunlukta bir gömme (embedding) oluşturan Vision Transformer (ViT) modeline dayanmaktadır. İstem Kodlayıcı, bölütleme görevinde yol gösterici olan noktalar, sınırlayıcı kutular veya nesneye ilişkin metin açıklamaları gibi istemleri alarak bunları gömmelere dönüştürür. Maske Çözücü ise görüntü ve istem gömmelerini birleştirerek, hedeflenen nesnenin piksel düzeyinde bölütleme maskesini üreten hafif bir transformatör modelidir. Çalışmada, Railsem19 veri seti kullanılarak ray bölütlemesi yapılmıştır. Şekil 4.27’de veri setinden örnek görüntüler görülmektedir. Deneyler, Python 3 ve GPU desteği sunan çevrimiçi platform Google Colab üzerinde gerçekleştirilmiştir. SAM modeli, demiryolu raylarını etkili bir şekilde segmentlere ayırmış ve sol ray için 0.9760, sağ ray için ise 0.9595 IoU değerleri elde edilmiştir.



Şekil 4.27. Railsem19 veri setinden örnekler

SAM, görüntü bölütlemesi için güçlü ve çok yönlü bir araçtır. Eğitim verilerinde mevcut olmasa bile görüntüdeki herhangi bir nesneyi segmentlere ayırmak için kullanılabilir. Model halen geliştirilme aşamasındadır ancak görüntü bölütlemesi alanında devrim yaratma potansiyeline sahiptir. SAM'in bazı avantajları:

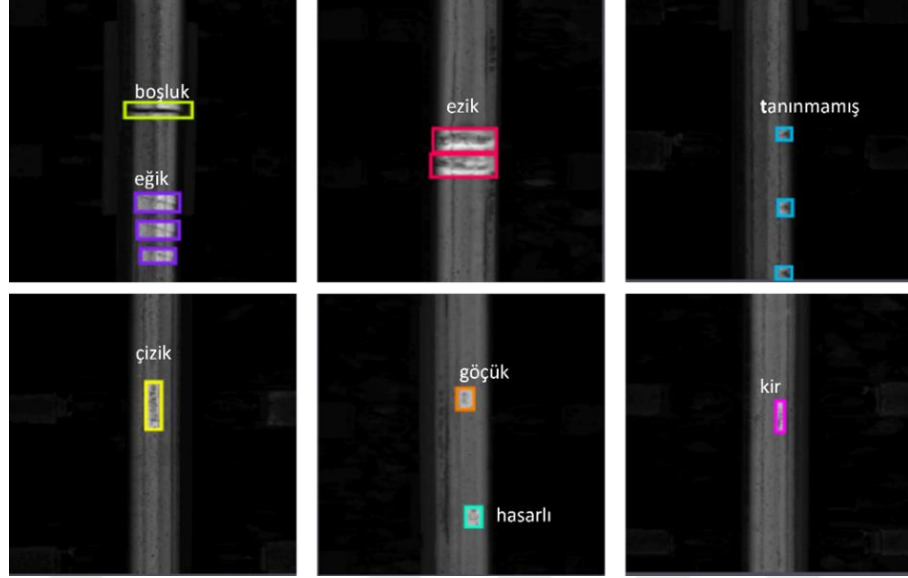
- a) Eğitim verilerinde mevcut olmasa bile görüntüdeki herhangi bir nesneyi segmentlere ayırabilir.
- b) Kullanımı nispeten kolaydır.
- c) Hızlı ve etkilidir.
- d) Nesne algılama, görüntü düzenleme ve sahne anlama gibi çeşitli görevlerde kullanılabilir.

4.7. Topluluk Öğrenmesine Dayalı Olarak Ray Yüzeyindeki Kusurların Tespiti

Önerilen yöntem, raylardaki kusurların tespiti amacıyla YOLOv4, YOLOv5 ve YOLOv6 modellerinin kullanımını içermektedir. Şekil 4.28’de yer alan görüntüde görüldüğü gibi, ray üzerinde çeşitli kusur türleri yer almakta ve tek bir görüntüde birden fazla kusur tespit edilebilmektedir. YOLO tabanlı modellerin tercih edilmesinin nedeni hem hızlı tahminler yapabilmeleri hem de yüksek doğruluk oranları sunabilmeleridir. Buna karşılık, R-CNN gibi bölge bazlı nesne tespit algoritmaları, önce olası alanları belirleyip daha sonra bu alanlar üzerinde ayrı ayrı evrişimsel sinir ağı (CNN) sınıflandırıcıları çalıştırmaktadır. Bu yöntem iyi sonuçlar verebilmesine rağmen, görüntünün iki aşamalı işleminden geçmesi işlem sayısını artırır ve bu da düşük FPS (saniye başına kare) oranına yol açmaktadır.

Bu çalışmada, toplamda 399 görüntüden oluşan bir veri seti kullanılmıştır [110]. Kusurlu demiryolu raylarına ait görüntülerin sayısı, görüntü artırım yöntemleri uygulanarak 939’a çıkarılmıştır. Kullanılan görüntü artırım yöntemleri arasında görüntüleri çevirme, parlaklık ayarı ve gürültü ekleme gibi teknikler yer almaktadır [111]. Veri setinde 810 görüntü eğitim, 90 görüntü değerlendirme ve 39 görüntüde test için ayrılmıştır. Veri setinde, 8 farklı türde ray kusuru bulunmaktadır. Şekil 4.28’de bu veri setindeki örnek kusurların bazıları gösterilmiştir. Bu kusurlar şunlardır:

- a) Çizik, ray kafasının yan düzlemlerinde küçük/hafif çizik.
- b) Ezik, demiryolunun yanal düzlemlerinin büyük/şiddetli aşınması.
- c) Kir, rayın yüzeyini kaplayan boya veya çamur.
- d) Boşluk, bir demiryolu hattında birbirini takip eden raylar arasında kalan boşluklardır.
- e) Tanınmamış, tanımlanamayan kusurlar.
- f) Eğik, yatay pozisyonda aşağıya doğru eğimli kusurlar.
- g) Göçük, ray yüzeyindeki oyuklar.

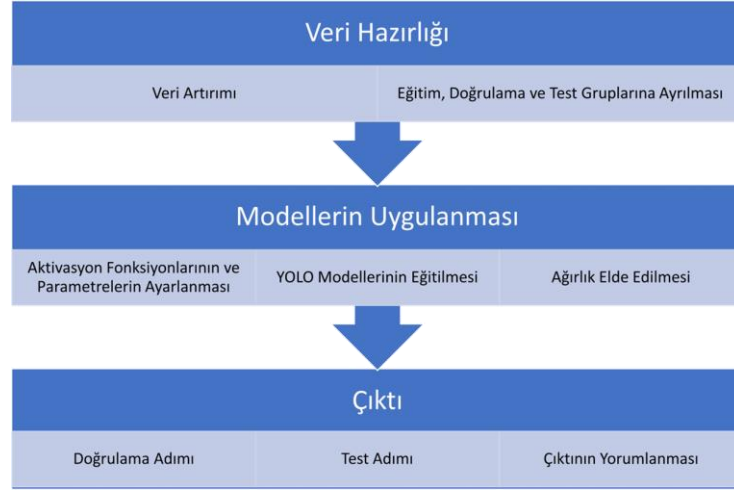


Şekil 4.28. Veri setinden etiketlenmiş örnekler

Eğitim süreci, tüm modellerde Google Colab platformu kullanılarak gerçekleştirilmiştir. Deneyler, Tesla P100-PCIE-16GB grafik işlem birimi (GPU) ve 2.20GHz hızında çalışan Intel(R) Xeon(R) merkezi işlem birimi (CPU) içeren bir sistem üzerinde yürütülmüştür. Etiketlenmiş veri seti Roboflow aracına yüklenmiş ve mevcut 399 görüntüye görüntü çevirme, parlaklık ayarlama ve gürültü ekleme gibi işlemler uygulanarak görüntü sayısı 939'a çıkarılmıştır. Bu genişletilmiş veri seti Google Colab ortamına aktarılmış, ardından adım adım YOLO algoritmalarının kodları çalıştırılarak eğitim, doğrulama ve test işlemleri tamamlanmıştır. Tüm modellerde kullanılan eğitim parametreleri Tablo 4.4'te sunulmuştur. Önerilen modelin akış diyagramı Şekil 4.29'da sunulmuştur.

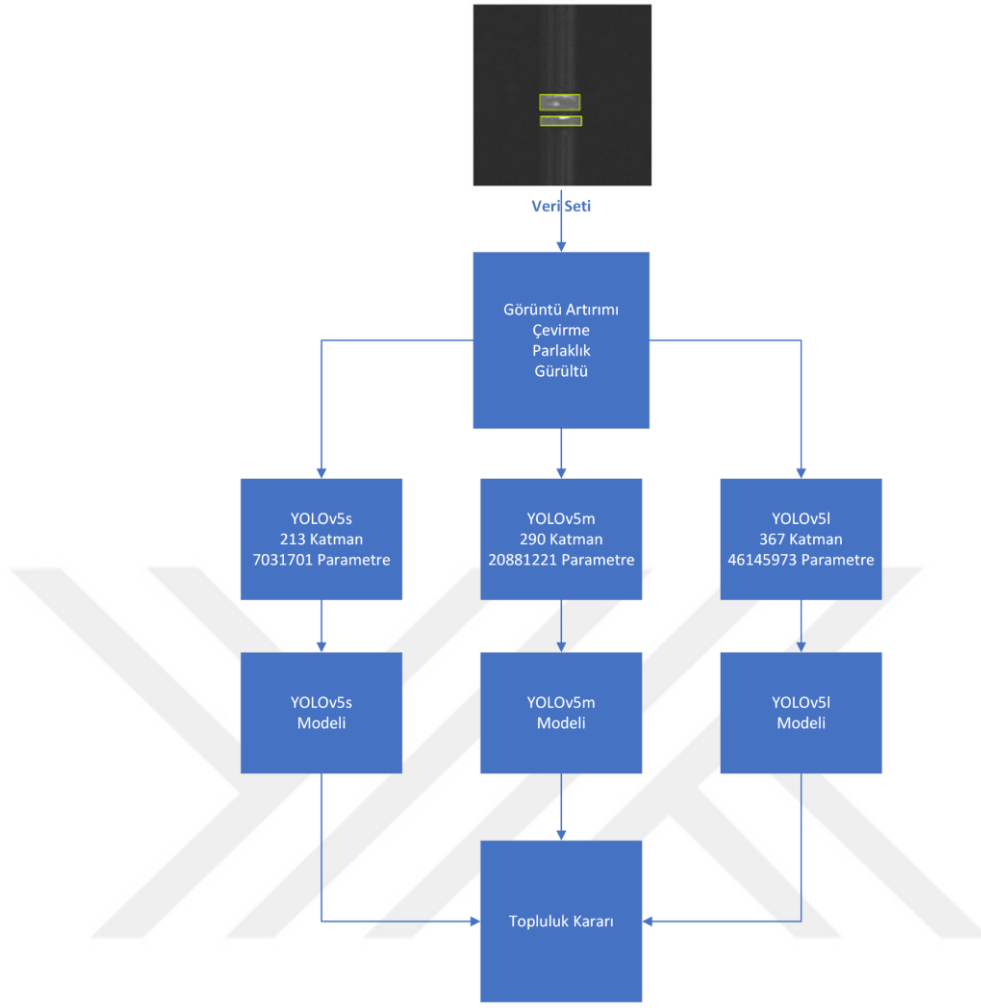
Tablo 4.4. Eğitimde kullanılan parametreler

Parametre	Değer
Girdi Boyutu	416
Döngü Sayısı	200
Bir iterasyon içerisinde her aşamada incelenen görüntü sayısı	32



Şekil 4.29. Önerilen modelin akış diyagramı

Bu çalışmada ayrıca YOLOv5 ile eğitilmiş ağırlıklar kullanılarak topluluk modellemesi yöntemi uygulanmıştır. Bu yöntemin mAP.5 ve duyarlılık değerleri üzerinde olumlu bir etkisi olduğu belirlenmiştir. Topluluk modelleme, bir sonuca ulaşmak için farklı modelleme algoritmalarını veya eğitim veri setlerini kullanarak çeşitli modellerin oluşturulduğu bir süreçtir. Bu yöntem, her bir temel modelin tahminlerini bir araya getirerek, görünmeyen veriler için nihai bir tahmin üretir. Topluluk modellemesinin temel amacı, tahminlerin genelleme hatasını azaltmaktır. Temel modeller çeşitli ve bağımsız olduğu sürece, topluluk modeli kullanıldığında tahmin hatası azalmaktadır. Şekil 4.30'da önerilen topluluk modeli gösterilmektedir. Bu yapıda, YOLOv5 alt mimarileri bireysel olarak eğitilmiştir. Topluluk öğrenimi, farklı modellerin sonuçlarını birleştirerek tüm tahminleri entegre eder. Bu modelde, ray kusurlarını algılamak için görüntüler, çeşitli parametrelerle eğitilmiş birden fazla modele gönderilir ve bir modelin sınırlamalarını önlemek için farklı modellerin çıktıları birleştirilir. Ardından, tahmin edilen sınırlayıcı kutuların ortalaması alınır ve tüm modellerin kategori tahminleri üzerinden bir oylama yapılır. Topluluk modeli, birçok temel model içerse de tek bir model gibi davranır [112]. Çalışmada YOLOv5'in alt mimarileri olan YOLOv5s, YOLOv5n, YOLOv5m ve YOLOv5l kullanılarak veri seti eğitilmiş ve bu ağırlıklar topluluk modellemesinde kullanılmıştır.



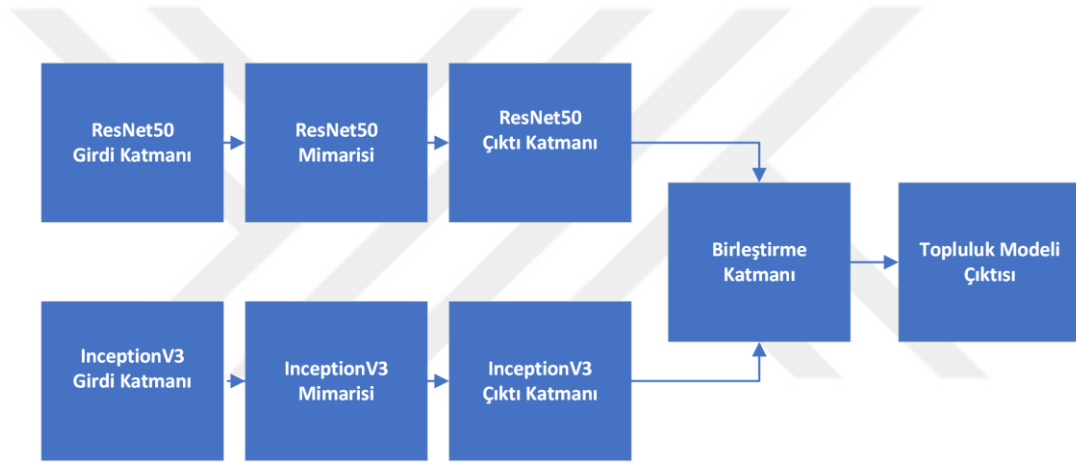
Şekil 4.30. Önerilen topluluk modelinin akış diyagramı

4.8. Üretken Çekişmeli Ağlar ve Topluluk Yöntemine Dayalı Ray Üzerindeki Kusurların Sınıflandırması

Bu çalışmada kullanılan veri seti, 1838 ray görüntüsünden oluşmaktadır. Bu görüntülerden 416'sı çatlak ray yüzeylerini, 492'si sağlıklı ray yüzeylerini ve 930'u ise delik ray yüzeylerini içermektedir. Aktif olarak kullanılan demiryolu hatlarında arızalı ray görüntülerine ulaşmak oldukça zordur. Demiryolu bakımına yönelik geliştirilen otomatik bakım sistemlerinde, görüntülerde hem arızalı hem de sağlıklı bileşenlerin bulunması gerekmektedir. Bu çalışmada, bu duruma çözüm olarak GAN (Üretici Çekişmeli Ağlar) algoritması kullanılmıştır. GAN algoritmik mimarisi, iki farklı sinir ağından oluşmaktadır: üretici ve ayrıştırıcı. Bu ağlar birbirleriyle çekişmeli bir şekilde çalışmaktadır. Üretici ağ, girdi olarak aldığı gürültü vektörünü kullanarak eğitim setindeki gerçek görüntüleri simüle etmeyi amaçlarken, ayrıştırıcı ağ, sentetik olarak üretilmiş (sahte) görüntüler ile gerçek eğitim görüntüleri arasındaki farkı ayırt etmeye çalışır [83]. GAN mimarisinde amaç, gürültü uzayını gerçek veri dağılımına haritalamaktır. Eğitim süreci,

ayırıştırıcının tahminlerinin, gerçek ile sahteyi ayırt edemeyeceği ana kadar devam etmektedir. GAN mimarisi tarafından üretilen görüntülerin kalitesi ve çeşitliliği, önerilen performansı doğrudan etkilemektedir. Literatürde GAN mimarisinin performansını ölçen bazı metrikler bulunmaktadır. Bu performans analizine Görsel Turing Testi (VTT), Frechet Inception Distance (FID) ve İstatistiksel Olarak Farklı Kutuların Sayısı (NDB) dahildir. VTT, yapay veri ile gerçek verinin uzmanlar tarafından oylanmasıdır. Oldukça zaman alıcı bir süreçtir. FID ve NDB ise otomatik performans metrikleridir.

Çalışmada önerilen model, transfer öğrenme tabanlı derin sinir ağları (ResNet50 ve InceptionV3) kullanılarak, topluluk kararına dayalı olarak ray görüntülerini içerdiği arızaya göre otomatik olarak sınıflandırmaktadır. Önerilen yöntemin şematik gösterimi Şekil 4.31'de yer almaktadır.



Şekil 4.31. Önerilen topluluk modeli

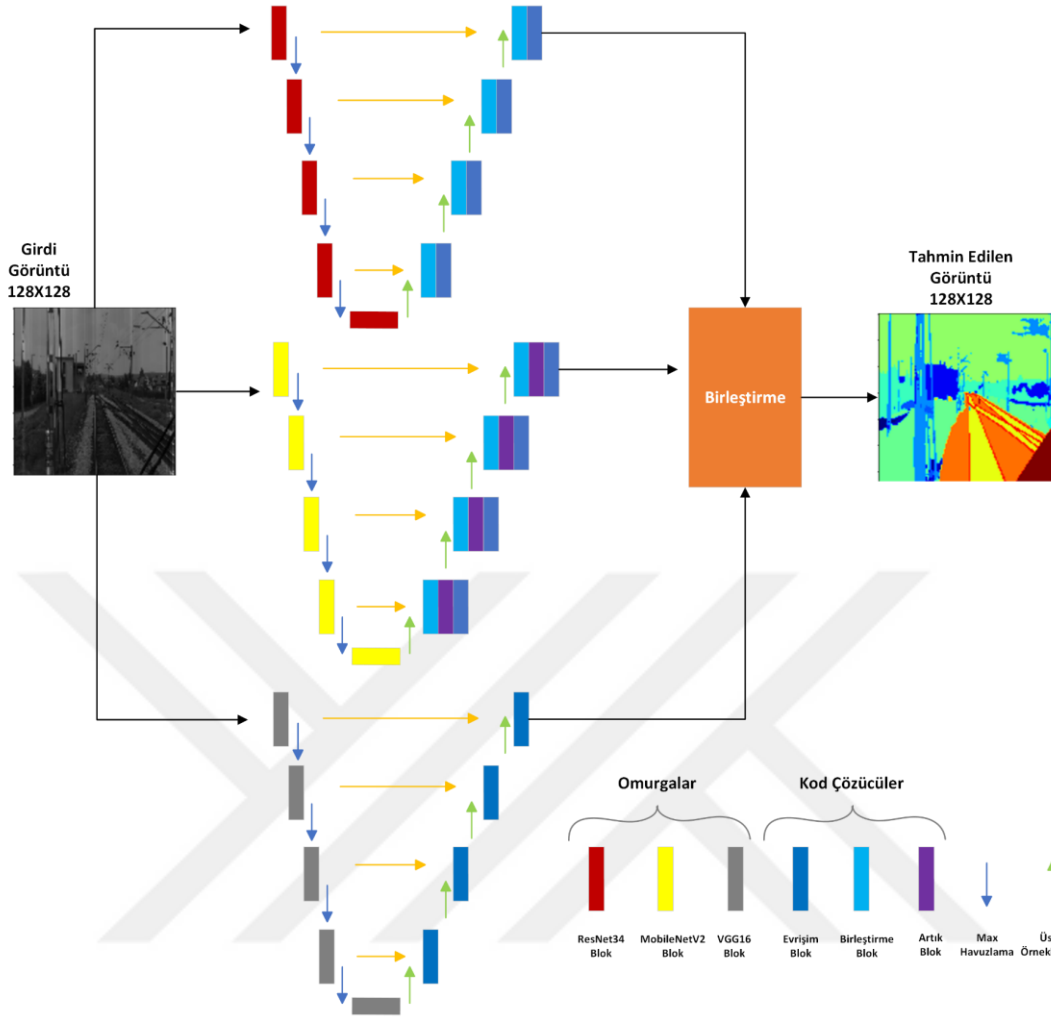
ResNet50 ve InceptionV3 modelleri, öncelikle ayrı ayrı eğitim işlemlerine tabi tutulmuştur. Her iki model için yapılan deneysel çalışmalarda, konvolüsyon katmanlarında ReLU ve çıktı katmanında softmax fonksiyonu kullanılmıştır. Derin sinir ağı mimarilerinde, tamamen bağlı katmanda çok sınıflı sınıflandırmaya ihtiyaç duyulduğundan, her iki modelde de softmax fonksiyonu kullanılmaktadır. Eğitilmiş ResNet50 ve InceptionV3 modelleri, model ağırlıklarının birleştirilmesi yoluyla birleştirilmiş ve nihai tahminler için topluluk oylama tekniği kullanılmıştır.

Önerilen yöntem, geleneksel ve derin sinir ağı tabanlı sınıflandırma yaklaşımlarına üstünlük sağlamıştır. Geliştirilen topluluk modeli, ray görüntülerini üç sınıfta (sağlıklı, çatlak ve delik) etkili bir şekilde sınıflandırmaktadır. Derin öğrenmeye dayalı önerilen arıza tespit sisteminin etkinliği, bir ölçüm dizisinden elde edilen ray bileşeni görüntüleri analiz edilerek değerlendirilmiştir.

4.9. Demiryolu Hatlarını İçeren Görüntülerde Bir Topluluk Modeli Kullanarak Unet Bölütleme Performansını İyileştirme

Demiryolu hattı çevresinde homojen nesneler elde etmek için Unet derin öğrenme tabanlı bir model önerilmiştir. Yapay zekanın alt dallarından biri olan derin öğrenme yöntemleri ile görüntülerden otonom olarak kusurları tespit etmek oldukça mümkündür. Derin öğrenme teknikleri, geleneksel tekniklere kıyasla probleme özgü farklı özelliklerin otomatik olarak öğrenilmesine yardımcı olur [58, 113]. Yani derin öğrenme, makine öğreniminde tanımlanması gereken parametreleri keşfeder ve makine öğreniminden daha doğru sonuçlar verebilir. Evrişimli sinir ağları, derin bir öğrenme stratejisine dayanır. Normalde, evrişimli sinir ağları gibi geniş özellikli öğrenme ağlarının, eğitim için büyük veri kümelerine sahip olması gerekir. Bir diğer önemli nokta da klasik evrişimli sinir ağlarında genel görüntü eğitiminin sınıf etiketleri üzerinden yapılmasıdır. Ancak, bazı problemler piksel tabanlı yaklaşımlar gerektirir. Sağlık veya güvenlik alanları gibi hassas yaklaşım gerektiren alanlarda her bir pikselin sınıf bilgisine ihtiyaç duyulur. Görüntü bölütlemenin devreye girdiği yer burasıdır. Bu çalışmada semantik bölütleme yöntemleri kullanılmıştır. Semantik bölütleme, bir görüntüdeki her bir pikselin hangi sınıfa ait olduğunu belirler [114].

Unet, kodlayıcı ve kod çözücü parçalarından oluşur. Kodlayıcı kısmında görüntüdeki içerik yakalanır. Maksimum havuzlama katmanları da bu bölümdedir. Öte yandan, kod çözme kısmı, yetkilendirilmiş konvolüsyonları kullanarak hassas yerelleştirme elde etmek için kullanılan simetrik genişleme yoludur. Buna kod çözücü de denir. Bu çalışma, önceden eğitilmiş omurga ağları ile 3 Unet mimarisini birleştiren demiryolu görüntüleri için toplu karar vermeye dayalı semantik bir bölütleme modeli sunmaktadır. Önerilen bölütleme modellerinden biri, omurga olarak [115] üzerinde önceden eğitilmiş ResNet34 mimarisine sahiptir. Bu omurga 34 katmandan oluşmaktadır. Diğer omurgalar MobileNetV2 ve VGG16'dır. Bu omurgalar, ImageNet üzerinde önceden eğitilmiştir. Bahsedilen tüm omurga ağırlıkları, ImageNet veri setinde [116] önceden eğitilmiştir. Bunun avantajı, ağıın son birkaç katmanının yaptığı öğrenme prosedürünü kısaltması, yakınsamayı hızlandırması ve veri setinden yüksek performans almasıdır.



Şekil 4.32. Önerilen bölütleme modelinin mimarisi

Şekil 4.32’de görüldüğü gibi, Unet mimarisini kullanan önceden eğitilmiş 3 omurga, tahmin edilen görüntü için ortak karar verir. 3 omurgada Unet mimarisinin kodlayıcı kısmı olarak kullanılmaktadır. Kod çözümlü yolu otomatik olarak tanımlanır. Bu omurgalar, Unet’in ilk yarısında kodlayıcı olarak kullanıldı. Ardından, kod çözümlü katmanları veri seti ile eğitilir. Ardından, kod çözme katmanları artırılmış veri kümesiyle eğitilir. Eğitim prosedürünün kaydedilmesine yardımcı olur ve Unet’in küçük verilerden öğrenme yeteneğini geliştirir. Her şeyden önce, üç omurga da birbirinden bağımsız olarak eğitilir. Şekil 4.32’de görüldüğü gibi, ResNet34, MobileNetV2 ve VGG16’yı kod çözümlü olarak işleyebilen bir mimari öneriyoruz. Unet’in kodlayıcı kısmında her omurga kendi bloğunu kullanıyordu. Kodlayıcı yolunda, geleneksel evrişimli yığın ve 2x2 maksimum havuzlama katmanları kullanıldı. Maksimum havuzlama, yalnızca verilen adımda maksimum giriş değerini alır. Kod çözümlü kısmında ise her omurganın kendine özgü katman yapısı vardır. Kod çözümlü, üst örnekleme ve birleştirme ve ardından düzenli evrişim işlemlerinden oluşur. Sağdaki genişletme yolunda üst örneklemeden sonra özellik kanallarının sayısını yarıya indiren 2

x 2 evriřim katmanı, birleřtirme veya dikkat bloęu kullanılmıřtır. Son olarak, önerilen model, her bir Unet modelinin sonucunu birleřtirmektedir.

Unet'in verilen verileri ne kadar iyi modelledięini anlamak için ařaęıdaki kayıp fonksiyonları kullanılmıřtır. alıřmada iki tip kayıp fonksiyonu birleřtirilmiřtir, Dice kayıp fonksiyonu ve Focal kayıp fonksiyonu. Önerilen modelde literatürde en ok kullanılan kayıp fonksiyonları birleřtirilerek hibrit bir kayıp modeli elde edilmektedir. Önerilen hibrit modelde kullanılan kayıp fonksiyonları ařaęıda açıklanmıřtır. Dice kaybı, Denklem 4.17'deki kesinlik ve Denklem 4.18'deki duyarlılık deęerleri kullanılarak yazılır. Dice kaybının hesaplaması Denklem 4.19'da gösterilmiřtir. Dice kaybı fonksiyonu, en yaygın olarak kullanılan bölütleme deęerlendirme metrięidir [117], doęrudan optimize eder. Literatürdeki alıřmalarda β 1 olarak alındıęında Unet modelinin doęruluk oranının maksimum olduęu gözlemlenmiřtir [118].

$$\text{Kesinlik} = \text{Doęru Pozitif} / (\text{Doęru Pozitif} + \text{Yanlıř Negatif}) \quad (4.17)$$

$$\text{Duyarlılık} = \text{Doęru Pozitif} / (\text{Doęru Pozitif} + \text{Yanlıř Pozitif}) \quad (4.18)$$

$$\text{Dice}(\text{Kesinlik}, \text{Duyarlılık}) = 1 - 1 + \beta^2 \frac{\text{Kesinlik} * \text{Duyarlılık}}{\beta^2 * \text{Kesinlik} * \text{Duyarlılık}} \quad (4.19)$$

Denklem 4.20'deki, temel gerek gt ile tahmin edilen pr [119] arasındaki kategorik Focal kaybını ölen bir kriter oluřturur. α deęeri 0,25 ve γ deęeri 2 olarak alınmıřtır.

$$\text{Focal}(gt, pr) = -gt * \alpha * (1 - pr) * \log(pr) \quad (4.20)$$

Hem Dice kaybından hem de Focal kaybından gelen katkılardan oluřan karma bir kayıp kullanıyoruz. Daha sonra bu iki kayıp fonksiyonu Denklem 4.21'deki gibi iřlenir ve nihai kayıp deęeri model eęitimine gönderilir.

$$\text{Toplam Kayıp} = (\text{Dice Kaybı} + \text{Focal Kaybı}) \quad (4.21)$$

Anlamsal bölütlemelerde sıklıkla kullanılan bir dięer performans deęerlendirme metrięi de ortalama IoU metrięidir. IoU metrięi, tahmin edilen görüntü ve referans görüntü kesiřiminin tahmin edilen görüntü ve referans görüntü kombinasyonuna bölünmesiyle hesaplanır. Denklem 4.22'de görölmektedir. IoU deęeri her sınıf için ayrı ayrı hesaplanır. Bu deęerlerin ortalaması alınarak ortalama IoU deęeri elde edilir [120]. Doęruluk oranı, doęru piksel tahminlerinin toplam tahminlere oranını ölçmektedir.

$$\text{IoU} = \frac{\text{Tahmin} \cap \text{Referans}}{\text{Tahmin} \cup \text{Referans}} \quad (4.22)$$

Uygulamada RailSem19 veri seti kullanılmıştır. RailSem19, demiryolu alanındaki semantik bölütleme için özel olarak tasarlanmış bir veri kümesidir. Bir demiryolu aracına monte edilmiş bir mobil haritalama sisteminden alınan yüksek çözünürlüklü görüntüler içerir. Veri kümesi, raylar, katenerler, platformlar ve çevredeki altyapı dahil olmak üzere farklı perspektiflerden yakalanan çeşitli demiryolu sahnelerini içerir. Veri setinde 19 sınıf bulunmaktadır. Ek olarak arka plan sınıfını 20. sınıf olarak ekledik. Bazı önemli sınıflar; demiryolu, tramvay, insan, trafik işareti, araba, raylı taşıtlar, metal raylar, sürülebilir raylar, yol, yaya geçidi, bitki örtüsü ve arazi sınıflarıdır. RailSem19, nesne algılama, sahne anlama ve durum izleme gibi demiryolu altyapı analizi ile ilgili algoritmalar ve modeller geliştirmek ve değerlendirmek için değerli bir veri setidir. Demiryolu bakımı, güvenliği ve otomasyon uygulamalarında araştırma ve geliştirmeyi kolaylaştırabilir. Veri seti, demiryolu ve çevresindeki nesneleri içeren 8500 görüntüden ve bu görüntülerin referans görüntülerinden oluşmaktadır [77]. Veri setini oluşturan görseller, karmaşık ray senaryoları ve zorlu hava koşullarını içeren görsellerden oluşmaktadır. Bu durum önerilen modelin zor ve farklı koşullardaki performansını desteklemektedir. Şekil 4.33, veri kümesinden örnekleri göstermektedir. Veri setinin %80'i eğitim, %10'u doğrulama ve %10'u test için kullanılmıştır.



Şekil 4.33. Veri setinden örnekler

4.10. YOLOv8 ile Demiryolu Hattı Çevresindeki Yabancı Nesnelerin Tespiti

Bu çalışmada, demiryolu hattı çevresinde bulunan yabancı nesnelerin tespit edilmesi amacıyla derin öğrenme tabanlı bir yaklaşım önerilmiştir. Özellikle, YOLOv8 nesne algılama modeli kullanılarak bu nesneler tespit edilmektedir. Demiryolu güvenliği açısından büyük önem taşıyan bu tespit, geleneksel yöntemlerin zaman ve maliyet dezavantajlarını gidermeye yönelik olarak geliştirilmiştir. Çalışmada, demiryolu hattına ait görüntülerden oluşan RailSem19 veri seti kullanılmış ve bu veri seti üzerindeki insan ve araç sınıfları elle etiketlenerek eğitim, doğrulama ve test süreçlerine tabi tutulmuştur. RailSem19 veri seti, 2019 yılında Avusturya Teknoloji Enstitüsü tarafından araştırmacıların hizmetine sunulan, demiryolu araçlarının sürüş güvenliğini artırmaya yönelik bir veri setidir [77].

YOLOv8, derin öğrenme yöntemleri arasında, hız ve doğruluk açısından en verimli modellerden biri olarak öne çıkmaktadır. Görüntü işleme süreci, görüntünün tek bir sinir ağı üzerinden geçirilmesiyle gerçekleştirilir. Bu ağ, sınırlayıcı kutular (bounding boxes) ve nesne sınıflarını aynı anda tahmin ederek, tespit işlemini hızlı ve verimli bir şekilde tamamlar [121]. YOLOv8, omurga (spine), boyun (neck) ve baş (head) olmak üzere üç ana bileşenden oluşur. Bu mimari, hem hesaplama kaynaklarını etkin kullanarak gerçek zamanlı nesne tespiti sağlamakta hem de daha düşük güçlü cihazlarda bile etkili sonuçlar sunmaktadır [122]. Omurga kısmı, görüntüdeki önemli özelliklerin çıkarılmasında kullanılır. Bu aşamada farklı katmanlardan geçen görüntü, düşük ve yüksek seviyeli özelliklerin yakalanmasını sağlar. Boyun, farklı çözünürlüklerdeki özellik haritalarını birleştirerek daha kapsamlı bir tespit sağlar. Bu aşama, özellikle küçük nesnelerin tespitinde önemlidir. Baş kısmı ise modelin çıktısını oluşturan katmandır; burada nesnelerin sınırlayıcı kutuları ve sınıf etiketleri tahmin edilir [122].

Veri işleme aşamasında, veri seti üzerinde birkaç veri artırma yöntemi uygulanmıştır. Bu yöntemler arasında görüntü çevirme, gürültü ekleme ve parlaklık ayarı gibi işlemler yer almaktadır. Görüntü çevirme, bir görüntünün yatay veya dikey ekseninde döndürülmesiyle yeni veriler oluştururken, gürültü ekleme işlemi görüntünün her pikseline rastgele ek bilgi katarak görüntü kalitesini değiştirmektedir. Parlaklık ayarı ise görüntüde doğrusal bir dönüşüm yaparak piksel parlaklığını artırma ya da azaltma işlevi görmektedir [59].

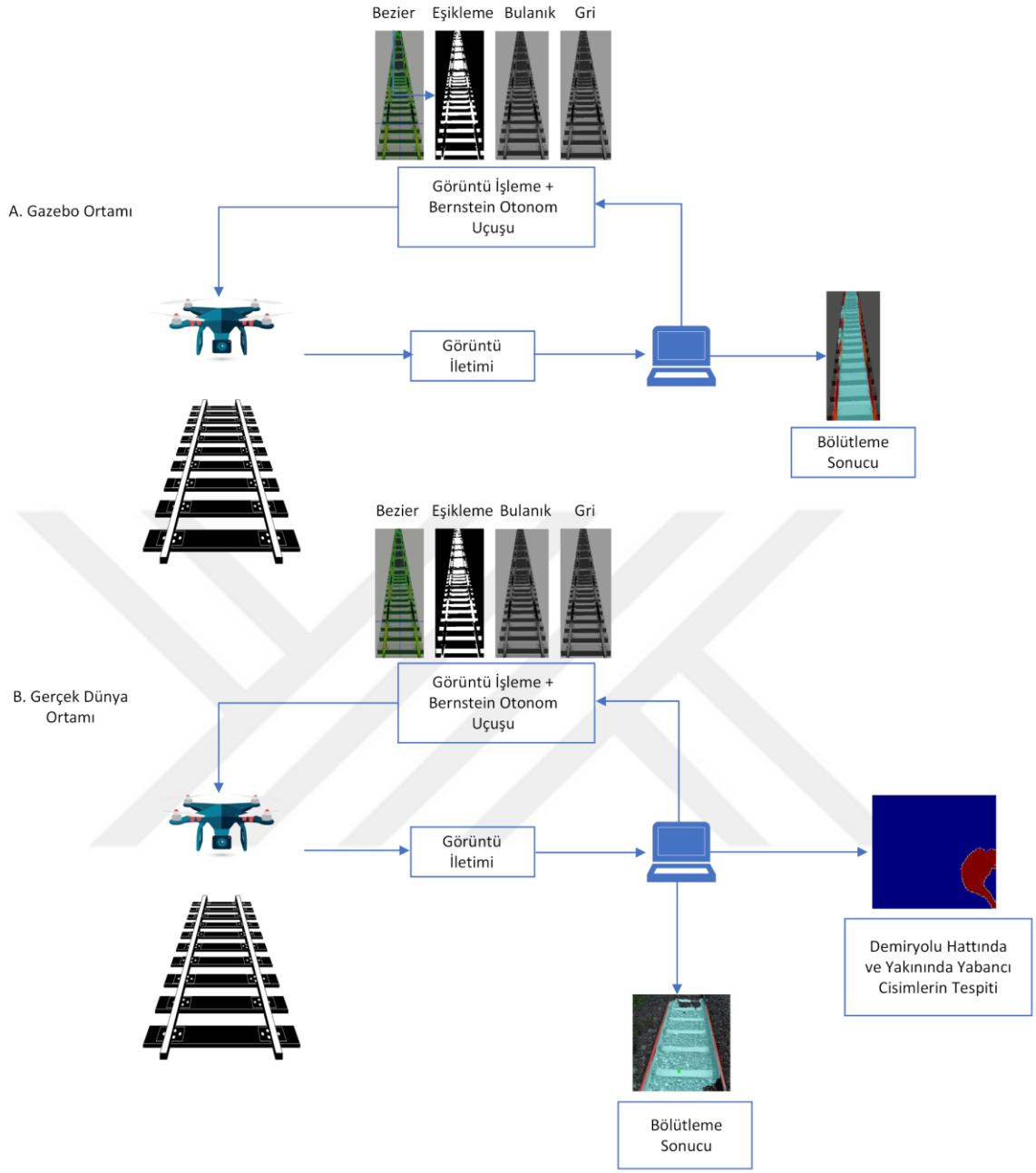
Sonuç olarak, YOLOv8'in nesne algılama kapasitesi hem hız hem de doğruluk açısından değerlendirilmiş ve özellikle mAP50 başarımleri metriklerinde güçlü performans sergilediği gözlemlenmiştir [123]. Bu modelin, rakip yöntemlerle karşılaştırıldığında daha hızlı ve verimli olduğu belirtilmiştir. Çalışmada kullanılan YOLOv8'in farklı alt mimarileri (YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, YOLOv8x) ile çeşitli denemeler yapılarak modelin farklı zorluk seviyelerinde nasıl performans gösterdiği incelenmiştir.

4.11. DNet ve Matematiksel Yöntemler Kullanarak Geliştirilmiş Demiryolu İzleme ve Bölütleme

Bu çalışmada demiryolu hattını segmente etmesi için BiSeNetV2 modeli kullanılmıştır. Bu çalışmada, BiSeNetV2 algoritması kullanılarak demiryolu görüntüleri için hızlı anlamsal bölütleme ray tespiti gerçekleştirilmiştir. Anlamsal bölütleme, bilgisayarlı görmede önemli bir alandır ve piksel düzeyinde bir görüntüyü anlamayı amaçlar. Anlamsal bölütlemenin temel amacı, görüntüdeki her pikseli belirli bir sınıfa etiketlemektir. Bu çalışmada, gerçek zamanlı anlamsal bölütleme için iki yönlü bölütleme ağı BiSeNetV2 olarak adlandırılan iki yönlü bir mimari önerilmiştir.

YOLO, nesne tanıma görevlerinde kullanılan bir nesne tespit modelidir. Çalışmada BiSeNetV2 bölütleme işlemine ek olarak YOLO modelleri ile de demiryolu hattını segmente etmek amaçlanmıştır. YOLO modellerinin eğitimi sırasında da Railsem19 veri seti kullanılmıştır.

Simülasyon ve gerçek saha testleri sırasında Anafi4K'nın kamerası ile demiryolu hattı bölütlenmeye etmeye çalışılmıştır. İHA'nın uçuşu sırasında otonom bir şekilde demiryolu hattını takip edebilmesi için görüntü işleme tabanlı bir yöntem önerilmiştir. İHA kamerasından alınan anlık görüntüye gri tonlama ve eşik değeri uygulanarak kenar tespiti yapılmıştır. Görüntü işleme süreci içinde, özellikle belirli bir bölgede (ROI - Region of Interest) rayların konumları tespit edilmeye çalışılır. Burada, konvolüsyon ve kayan pencere gibi teknikler kullanılarak sol ve sağ rayların yaklaşık konumları belirlenir. Sol ve sağ rayların belirlenmesi, daha sonra kullanılacak olan Bezier eğrisi oluşturma sürecine giden adımdır. Sol ve sağ rayların konumlarından, Bezier eğrileri oluşturulur. Bu eğriler, rayların hareketini daha düzenli ve takip edilebilir hale getirmek için kullanılır. Bezier eğrileri, rayların yola uyum sağlamasına yardımcı olmaktadır. Bezier eğrileri, Bernstein polinomları kullanılarak hesaplanır. Bu polinomlar, kontrol noktalarının ağırlıklarını belirleyerek eğrinin şeklini oluşturmaktadır. Her kontrol noktası için bir Bernstein polinomu kullanılır. PID kontrol algoritması, önceki adımlardan elde edilen bilgileri kullanarak İHA'nın konumunu düzenler. PID kontrolü, hedef konum ile mevcut konumu karşılaştırarak İHA'nın yatay ve dikey pozisyonunu ayarlar. Özellikle, Bezier eğrilerinden alınan hedef konumlar ve mevcut konumlar kullanılarak PID kontrol algoritması devreye sokulur. PID kontrolü, İHA'nın sol veya sağa dönmesi, ileri gitmesi veya durması gibi hareketleri düzenler. Bu adımlar, İHA'nın demiryolu hattını takip edebilmesi ve güvenli bir şekilde ilerleyebilmesi için gereklidir. Bu adımlar arasında sağlam bir ilişki, görüntü işleme ile başlayıp, ray tespiti ve onun üzerinden Bezier eğrileri oluşturarak PID kontrolüne kadar devam eder. İşlenen görüntülerden elde edilen bilgiler, İHA'nın güvenli bir şekilde hareket etmesini sağlamak üzere PID kontrol algoritması kullanılarak değerlendirilir ve İHA'nın hareketlerini yönlendirir. Önerilen yöntem Şekil 4.34'te görülmektedir.



Şekil 4.34. Önerilen yöntem

Bezier eğrisi, matematiksel bir ifade ile tanımlanan düzgün bir eğridir. Bezier eğrisi, iki veya daha fazla kontrol noktasını birleştirerek bir eğri oluşturur. İki kontrol noktası arasında geçen eğri, kontrol noktalarının konumlarına ve ağırlıklarına bağlı olarak belirlenir. Noktalar P_0 , P_1 , P_2 ve P_3 arasında bir Bezier eğrisi Denklem 4.23'teki gibi ifade edilmektedir.

$$B(t) = (1 - t)^3 P_0 + 3(1 - t)^2 t P_1 + 3(1 - t) t^2 P_2 + t^3 P_3 \quad (4.23)$$

Burada t , eğrinin parametre değeridir ve genellikle 0 ile 1 arasında değişir. P_0 , P_1 , P_2 ve P_3 kontrol noktalarını temsil eder. Bu formül, Bezier eğrisinin her bir noktasının x ve y koordinatlarını ifade eder. Eğri, t değeri 0'dan 1'e doğru değiştikçe kontrol noktaları arasında akıcı bir şekilde ilerlemektedir.

Bernstein polinomları, Bezier eğrisinin matematiksel temelini oluşturan polinomlardır. N kontrol noktası için $B_i^n(t)$ Bernstein polinomunu Denklem 4.24'teki gibi ifade edilmektedir.

$$B_i^n(t) = \binom{n}{i} (1-t)^{n-i} t^i \quad (4.24)$$

Burada n , kontrol noktalarının sayısını temsil eder. i , her bir Bernstein polinomunun derecesini ifade eder. Bernstein polinomları, Bezier eğrisinin kontrol noktalarının ağırlıklarını belirler. Bezier eğrisinin her bir noktası, bu polinomlar kullanılarak hesaplanır. Bu polinomlar, kontrol noktalarının fonksiyon içindeki ağırlıklarını belirler ve bu ağırlıklar eğrinin şeklini kontrol etmektedir. Önerilen demiryolu tespiti ve takibi algoritmasının sözde kodu Algoritma 1'de verilmiştir.

Algoritma 1. Demiryolu algılama ve izleme algoritmasının sözde kodu	
1	Başlangıç
2	GİRİŞ: Video Akışı (video), Kamera, PID Parametreleri
3	ÇIKTI: İHA Kontrolleri
4	Kamera ve İHA Ayarlarını Başlat
5	<i>Kamera ayarlarını kontrol edin ve yapılandırın</i>
6	<i>PID kontrol parametrelerini ayarlayın</i>
7	<i>Başlat ve İHA'ya bağlan</i>
8	WHILE
9	<i>Video akışından bir kare yakalayın</i>
10	<i>Çerçevedeki rayları algıla</i>
11	<i>Bezier eğrisini hesaplayın</i>
12	IF raylar tespit edilmişse
13	<i>PID denetimlerini uygula</i>
14	<i>İHA'yı kontrol et</i>
15	<i>Çerçeveye Bezier eğrisini çizin</i>
16	<i>Bezier eğrisi için orta nokta hesapla</i>
17	IF raylar tespit edilmemişse
18	<i>Uyarı mesajını ekrana bas</i>
19	<i>İHA'nın pozisyonunu koru</i>
20	<i>Ekranda bilgi ve sonuçları görüntüle</i>
21	<i>İHA kontrol bilgilerini güncelle</i>
22	<i>Çerçeveyi çıktı video dosyasına yaz</i>
23	END WHILE
24	Bağlantıları Kapatın
25	END

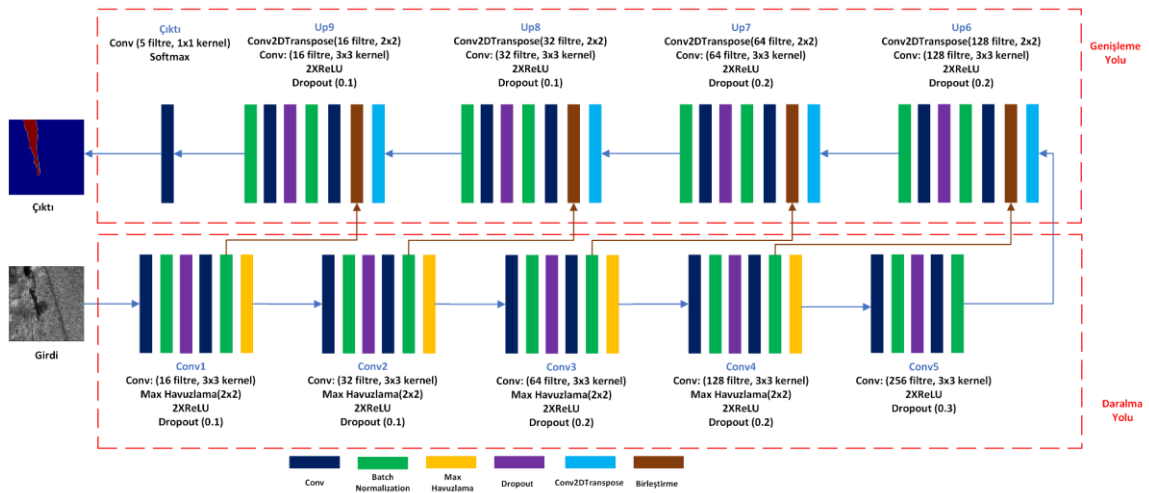
Geliştirilen otonom uçuş modeli sayesinde İHA ile yapılan uçuşlar sırasında bir veri seti oluşturulmuştur. Veri setinde demiryolu etrafında ve üzerinde bulunan yabancı cisimler etiketlenmiştir. Bu yabancı cisimler; insan, plastik poşet, bitki ve çubuktan oluşmaktadır. Veri setinde toplam 1100 adet görüntü mevcuttur. Tablo 4.5'te görülmektedir.

Tablo 4.5. Oluşturulan veri setinin detayları

Sınıf Adı	Adet
Bitki	1039
İnsan	843
Poşet	341
Çubuk	31

100 görüntü çeşitli görüntü artırma yöntemleri ile 2646'ya arttırılmıştır. Bu görüntü artırım teknikleri görüntü çevirme, parlaklık ve gürültüdür.

Yabancı nesneleri tespit etmek için Unet benzeri, Unet'in performansını yükselten DNet adında bir bölütleme sinir ağı modeli önerilmiştir. DNet iki bölümden oluşmaktadır: Daralma yolu (kodlayıcı) ve genişleme yolu (kod çözücü). Giriş katmanı 256x256 boyutlarında ve 3 kanallı (RGB) görüntüleri kabul etmektedir. Daraltma yolu, giriş verilerinden daha düşük seviyeli özellikleri çıkarmak ve özellik haritasının boyutunu azaltmak için kullanılmaktadır. Conv2D katmanı ilk adımda ReLU aktivasyon fonksiyonu, 'he_normal' ağırlık başlatma ve 'same' dolgusu ile gelmektedir. Batch Normalization ve Dropout (0.1) katmanları ağı stabilize eder ve aşırı öğrenmeyi azaltmak için kullanılmıştır. Daha sonra Conv2D ve Batch Normalization'ın bir katmanı daha eklenerek özellik haritası daha da zenginleştirilmiştir. DNet modelinde her Conv2D katmanının hemen ardından BatchNormalization katmanı gelmektedir. Bu, modelin eğitimi stabilize etmeye ve daha hızlı öğrenmesine yardımcı olabilir. Klasik Unet modellerde, bu sıralama farklıdır ve BatchNormalization kullanılmaz [72]. Şekil 4.35'te görüldüğü gibi genişleme yolunda blok sonlarında MaxPooling2D katmanı kullanılmıştır.

**Şekil 4.35.** Önerilen DNet mimarisi

Genişleme yolu, daralma yolundan özellik haritasını genişleterek daha yüksek çözünürlüklü bir çıktı elde etmeyi amaçlamaktadır. Conv2DTranspose katmanı, önceki adımda boyutu küçültülen özellik haritasını genişletmektedir. Birleştirme işlemi, bu genişletilmiş özellik haritasını, daraltma yolundaki özellik haritasıyla birleştirmektedir. Sonraki Conv2D katmanları ve BatchNormalization özellik haritasını daha da iyileştirmiştir. Dropout (0.2) uygulanarak aşırı öğrenme kontrol altında tutulmuştur. Bu adımlar, önceki daralma yolundan elde edilen özellik haritasını genişleterek daha yüksek çözünürlüklü bir çıktı elde etmeyi amaçlamaktadır. Benzer adımlar, Şekil 4.35'te görüldüğü gibi, genişleme yolunun diğer katmanlarında tekrarlanmıştır.

Çıktı katmanı, sınıfların olasılık dağılımlarını sağlayacak şekilde tasarlanmıştır. Conv2D katmanını ve Softmax aktivasyon fonksiyonunu içermektedir. Softmax, piksel değerlerini sınıflar arası olasılıklara dönüştürerek son sınıflandırma çıktısını elde etmektedir.

DNet modelinde, BatchNormalization katmanı Conv2D katmanından hemen sonra gelmiştir. Bu, modelin eğitimi stabilize etmeye ve daha hızlı öğrenmesine yardımcı olmaktadır. Bu sıralama klasik Unet modelinde farklıdır ve BatchNormalization'a yer verilmemiştir. Önerilen DNet modelindeki filtre sayısı, özellikle daralma ve genişleme yolunda klasik Unet modelinden farklılık göstermektedir. Şekil 4.35'te görüldüğü üzere önerilen modelde daralma ve genişleme yollarının her bir adımında filtre sayısı azalmakta ve artmaktadır. Bu yaklaşım, daha az filtreyle daha düşük ölçekli özellikleri çıkarır ve genişleme yolunda daha yüksek ölçekli özellik haritaları elde eder. Daralma yolundaki Dropout katmanı, genellikle aşırı uyum (aşırı öğrenme) riskini azaltmak için kullanılan bir düzenleme tekniğidir [124]. Dropout, eğitim sırasında rastgele seçilen nöronları (veya düğümleri) devre dışı bırakarak çalışır. Bu nöronlar ağ çıktısını oluştururken ve güncellerken kullanılmaz. Dropout katmanları klasik Unet modelinde bulunmaz. DNet modelini genişletme yolunda, Conv2DTranspose katmanları özellik haritalarını genişletmeye yardımcı olmuştur. Bu, daha büyük bir özellik haritası sağlamış ve daha iyi yerleştirme yetenekleri sunmuştur. Önerilen DNet ve klasik Unet, Tablo 4.6'da karşılaştırılmıştır.

Tablo 4.6. DNet ve klasik Unet'in karşılaştırılması

Model	Parametre Sayısı	Ortalama Eğitim Süresi (Döngü Başına)
Klasik Unet	31,030,853	270 ms
DNet	1,940,885	60 ms

Tablo 4.6'da görüldüğü gibi, önerilen DNet modeli klasik Unet'ten daha az toplam parametreye sahiptir. Daha az parametre genellikle ağın daha az karmaşık olduğu ve daha hızlı eğitildiği anlamına gelmektedir. Bu, eğitim süresini azaltıp ve modelin daha hızlı bir şekilde yakınsamasına yardımcı olmaktadır. Ek olarak, daha az parametre genellikle daha düşük bellek gereksinimleri ve daha hızlı tahmin süreleri ile sonuçlanmaktadır. Daha büyük model yapıları

genellikle daha fazla veri gerektirir ve aşırı uyum sağlama eğilimindedir. Daha küçük, daha basit modeller genellikle daha sınırlı veri koşulları altında daha iyi performans gösterir ve aşırı uyumu azaltabilir. Daha az parametre, modelin eğitim ve tahmin süreçlerinde daha az hesaplama gücü ve enerji tüketimine yol açmaktadır. Özellikle sınırlı hesaplama kaynaklarına veya enerji kısıtlamalarına sahip uygulamalarda daha az parametrelili modeller kullanmak avantajlıdır.

4.12. Hızlı ve Ekonomik Nesne Algılama Yoluyla Demiryolu Güvenliğinin Artırılması: Grounding Dino Yaklaşımı

Bu çalışmada, demiryolu güvenliğini artırmak amacıyla hızlı ve ekonomik bir nesne tespit yöntemi geliştirilmiştir. Veriler, Anafı4K İHA kullanılarak demiryolu çevresinde çekilen görüntülerden elde edilmiştir. Çalışmada kullanılan veri seti, demiryolu çevresinde bulunan insanları içeren 843 görüntüden oluşmaktadır. Modelin genel performansını artırmak amacıyla veri seti, görüntü döndürme, kırpma ve gürültü ekleme gibi çeşitli görüntü iyileştirme teknikleri kullanılarak genişletilmiştir. Bu yöntemler, modelin farklı koşullarda daha başarılı bir şekilde çalışmasını sağlamıştır. Şekil 4.36'da yapılan saha çalışmasından örnek bir görüntü görülmektedir.



Şekil 4.36. Yapılan saha çalışmasından görüntü

Nesne tespit sürecinde Grounding DINO ve Zero-Shot Object Detection (Sıfır Atışlı Nesne Tespiti) yöntemleri kullanılmıştır. Grounding DINO, modelin öğrenme sürecini optimize ederek nesne tespit doğruluğunu artırırken, Zero-Shot Object Detection, önceden eğitim verilmemiş

nesneleri tanıma yeteneği ile dikkat çekmektedir. Bu özellik, modelin demiryolu çevresindeki insanları yabancı nesneler olarak tanımasını sağlamaktadır.

Grounding DINO, dil ve görüntü modalitelerini etkili bir şekilde birleştirerek metin tabanlı nesne tespiti yapma yeteneğine sahiptir. Bu model, insan girdileri (örneğin, kategori isimleri veya referans ifadeler) aracılığıyla rastgele nesneleri tespit etme kapasitesine sahiptir. Sıfır Atışlı Nesne Tespiti ile kullanıldığında, modelin yeni nesneleri önceden eğitim verilmemiş sınıflar arasında tespit etme kabiliyeti geliştirilmiştir. Bu yaklaşım, demiryolu güvenliğini artırmaya yönelik önemli bir katkı sunmaktadır.

Çalışmanın önemli bir gücü hem hızlı hem de ekonomik bir çözüm sunmasıdır. Bu özellik, demiryolu işletmecileri ve güvenlik birimleri için pratik bir çözüm sağlayarak kaynakların daha etkili kullanılmasına olanak tanımaktadır. Ayrıca, demiryolu çevresindeki kişilerin erken tespit edilmesi, potansiyel tehlikelerin önceden belirlenmesine katkıda bulunmaktadır.

Bu yöntem, literatürdeki eksiklikleri gidermeyi amaçlayan yeni bir yaklaşım sunmakta ve demiryolu güvenliğini artırmaya yönelik önemli adımlar atmaktadır.

5. BULGULAR VE TARTIŞMA

5.1. Simülasyon Ortamında İHA ile Ray Takibi ve Tespitinde Elde Edilen Bulgular

5.1.1. Ufuk Noktası Yöntemi ile Ray Takibi Metodunda Elde Edilen Bulgular

Gazebo simülasyon ortamında Anafi4k İHA'sından elde edilen canlı görüntülere Gabor filtresi uygulanarak, belirli yönlere uzanan çizgiler başarıyla tespit edilmiştir. Gabor filtresi sayesinde, rayların yönelimi ve kenar yapısı net bir biçimde görüntülenmiştir. Şekil 5.1'de Gabor filtresi uygulamasının çıktısı gösterilmektedir.



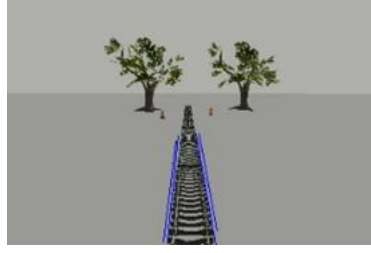
Şekil 5.1. Gabor filtresi çıktısı

Gabor işlemi sonucu elde edilen görüntüdeki kenarları tespit edebilmek için Canny edge detection yöntemi kullanılmıştır. *Kenar Olma Eğilimi(G)*, Canny algoritmasının sonuç görüntüsünü vermektedir. Şekil 5.2'de Canny algoritmasının çıktısı görülmektedir.



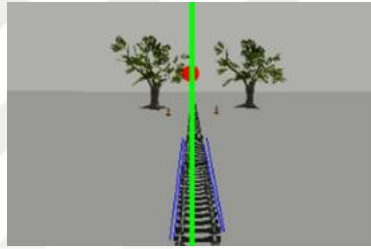
Şekil 5.2. (a) Canny algoritması çıktısı (b) Kırpılmış hali

Canny edge tespiti sonrasında, rayların görüntüdeki doğrusal yapıları tespit edilebilmesi için Hough dönüşümü kullanılmıştır. Şekil 5.3'te Hough dönüşümünün uygulandığı görüntüde rayların tespiti gösterilmiştir.



Şekil 5.3. Hough dönüşümünün çıktısı

Hough dönüşümü ile elde edilen rayların başlangıç ve bitiş noktaları iki ayrı dizi halinde saklanmış ve bu diziler yardımıyla en küçük kareler yöntemi ile ufuk noktası tahmini yapılmıştır. Intersect() fonksiyonu, rayların kesişim noktalarını belirleyerek ufuk noktasının koordinatlarını döndürmektedir. Şekil 5.4'te Anafı4k İHA'sının kamerasından alınan görüntü üzerinde tahmin edilen ufuk noktası gösterilmiştir.

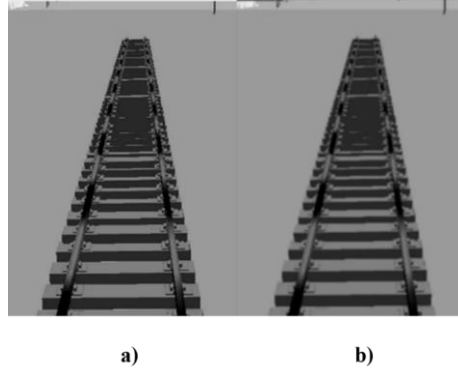


Şekil 5.4. Ufuk noktası tahmini

Tespit edilen ufuk noktası, İHA'nın otomatik olarak yönlendirilmesi için PID kontrol algoritmasına entegre edilmiştir. PID kontrol, İHA'nın yaw ekseninde ufuk noktasını takip ederek hatayı minimize eder. PID denetleyicisi, oransal K_p , integral K_i ve türev K_d katsayılarını kullanarak, İHA'nın yönünü sürekli olarak düzeltir.

5.1.2. Bilgisayar Görmesi Yöntemleri ile Ray Takibi Metodunda Elde Edilen Bulgular

Hat takibi ile İHA modelinin otonom olarak demiryolu hattını takip etmesi amaçlanıyor. Öncelikle Python ortamında İHA kameradan alınan gerçek zamanlı görüntü kırpılmıştır. Burada amaç İHA'nın sadece ray kısmına odaklanmasını sağlamaktır. Daha sonra gerçek zamanlı görüntü gri tonlamaya dönüştürülür ve Gauss bulanıklığı uygulanır. Görüntüdeki bulanıklığı gidermek için Gauss kullanılmıştır. Bu adımların sonuçları Şekil 5.5'te gösterilmektedir.



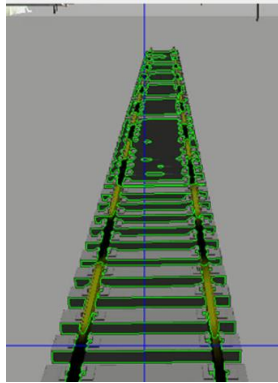
Şekil 5.5. (a) Grayscale sonucucu (b) Gaussian sonucu

Ortaya çıkan görüntüye bir renk eşiği uygulanarak görüntü ikili değere dönüştürülür. Elde edilen eşiklenmiş görüntü, Şekil 5.6'da gösterilmiştir.



Şekil 5.6. Eşiklenmiş görüntü

Eşiklenmiş görüntüde çizgiler algılanmıştır. Tespit edilen hattın koordinatlarına göre İHA'ya uçuş talimatı verilmiştir. Nihai rota Şekil 5.7'de gösterilmiştir.



Şekil 5.7. Tespit edilen rota

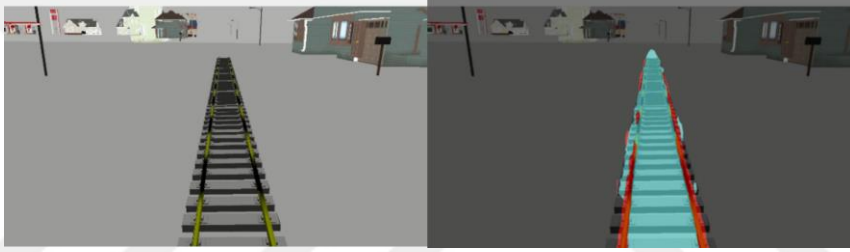
5.1.3. BiSeNetV2 ile Demiryolu Hattı Tespiti Metodunda Elde Edilen Bulgular

Önerilen yöntem kısmında belirtildiği gibi demiryolu hattını bölütlemek etmek için BiSeNetV2 bölütleme modeli kullanılmıştır. BiSeNetV2 modelinin eğitimi sonucunda Tablo 5.1'deki performans metriklerini elde etmiştir.

Tablo 5.1. BiSeNetV2 modeli sonuçları

Model	Metrik	Değer
BiSeNetV2	Doğruluk	0.9746

Şekil 5.8'de BiSeNetV2 modelinin Gazebo ortamındaki test sonuç sonuçları görsel olarak verilmiştir. BiSeNetV2 modeli test sonucu olarak 0.99'luk bir doğrulama oranı elde etmiştir.



Şekil 5.8. BiSeNetV2 modelin simülasyon ortamındaki test sonuçları

BiSeNetV2 modeli, Tablo 5.2'deki gibi literatürdeki diğer çalışmalardan daha iyi performans göstermiştir.

Tablo 5.2. BiSeNetV2'nin bölütleme performansının literatürdeki diğer çalışmalarla karşılaştırılması

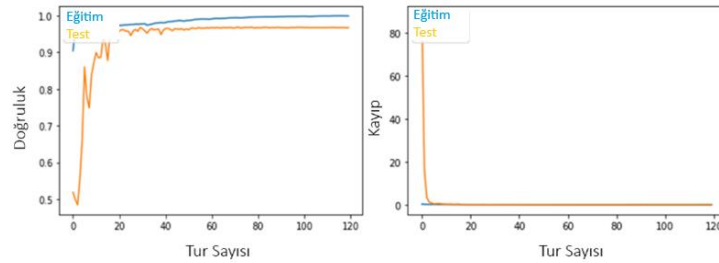
Model	IoU
RailNet [125]	0.89
RailCNN [126]	0.94
BiSeNetV2	0.99

Gazebo ortamında simüle edilen Parrot Anafi modelinin kamerası ile demiryolu hattının ortasındaki konumunu otonom olarak koruyabiliyor. Bu süre zarfında aynı kamera ile gerçek zamanlı olarak semantik bölütleme ile demiryolu hattı tespit edilebilir. Gazebo simülasyon ortamında, bir İHA, ray takibine dayalı otonom bir uçuşu başarıyla gerçekleştirdi. Semantik bölütleme, hata tespiti, malzeme sınıflandırması ve demiryolu hattı tespiti gibi problemlerde de kullanılmıştır. Çalışmada IoU değeri 0.99 olarak elde edilmiştir. BiSeNetV2 mimarisi, çoğu

evrişimli model tarafından uygulanabilen genel bir mimaridir. BiSeNetV2 modeli, iyi bir bölütleme doğruluğu ve çıkarım hızı elde etmiştir. Bu simülasyon, gelecekteki saha testleri için bir rehberdir.

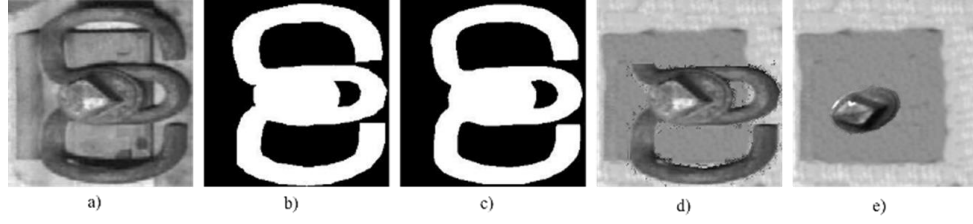
5.2. Derin Öğrenme Yöntemleri ile Demiryolu Bağlantı Elemanlarının Sınıflandırılması Metodunda Elde Edilen Bulgular

Bu çalışmada, demiryolu hattından toplanan sağlıklı bağlantı elemanlarını içeren veri seti, veri artırma teknikleri kullanılarak dengelenmiştir. Ardından, derin öğrenme yöntemleri olan CNN, VGG-16 ve ResNet50 modelleri dengelenmiş bu veri setine uygulanmıştır. Çalışmada kullanılan deney ortamı, Windows 10 işletim sistemi, Intel(R) Core (TM) i5-1135G7 @ 2.40GHz işlemci, 16 GB bellek ve NVIDIA GeForce MX450 2GB ekran kartından oluşmaktadır. DATEM'den sağlanan veri seti, 534 adet sağlıklı demiryolu bağlantı elemanı görüntüsünü içermektedir. Bu görüntülere ek olarak, bölütleme işlemiyle 534 adet arızalı ve eksik bağlantı elemanı görüntüsü elde edilmiştir. Yeni bağlantı elemanlarını oluşturmak amacıyla Unet bölütleme modeli kullanılarak eğitim gerçekleştirilmiştir. Eğitim ve doğrulama veri setleri, sırasıyla %80 ve %20 oranlarında ayrılmıştır. Modelin optimizasyon sürecinde Adam algoritması tercih edilmiş, eğitim döngü sayısı ise 120 olarak belirlenmiştir. Unet modelinin doğruluk ve kayıp oranlarını gösteren grafikler Şekil 5.9'da sunulmaktadır.



Şekil 5.9. Unet eğitim sonuçları

Şekil 5.9'daki grafiklere göre, 10. turdan sonra kayıp değeri bir değişiklik olmamasına rağmen, döngü sayısının 120 olarak seçilmesi modelin aşırı öğrenmeden kaçındığını göstermektedir. Şekil 5.9'daki kayıp grafiğinin incelenmesi, kayıp değerinin 0'a yakın olduğunu ortaya koymakta; bu değer 0'a yakın olması eğitimin başarıyla tamamlandığını ifade etmektedir. Şekil 5.10'da ise Unet ile elde edilen görüntüler ve bu görüntülerden üretilen arızalı ve eksik demiryolu bağlantı elemanları görülmektedir. Unet modeli %99 doğruluk oranına ulaşmıştır. Kusurlu veri bulunamadığında, sağlıklı demiryolu bağlantı elemanlarını kullanarak eksik ya da arızalı bileşenlerin üretilebileceği ispat edilmiştir; bu sayede veri seti dengeli hale getirilmiştir.



Şekil 5.10. Unet sonucu elde edilen görüntüler (a) Sağlıklı bağlantı elemanı (b) Sağlıklı referans veri (c) Bölütleme sonucu (d) Ön plana çıkarılmış arızalı bağlantı elemanı (e) Ön plana çıkarılmış eksik bağlantı elemanı

Bir sınıflandırıcının başarımını değerlendirmek için tek bir metrik yeterli değildir. Tüm sınıflar için doğruluğun ölçülmesi önemlidir, ancak yalnızca doğruluk metriği modelin performansını eksik gösterebilir. Bu nedenle, modelin başarı oranını değerlendirmek için doğruluğa ek metriklerin kullanılması gerekmektedir. Her bir sınıfa özgü olan bu metrikler, sınıflandırma algoritmasının ilgili sınıfta duyarlılığını hesaplamaktadır. VGG-16 modelinin performans değerlendirme ölçütleri Tablo 5.3'te sunulmaktadır.

Tablo 5.3. VGG-16 mimarisinin performans metrikleri

Sınıf/Metrik	Kesinlik	Duyarlılık	F1
Arızalı	1.00	1.00	1.00
Sağlıklı	1.00	1.00	1.00
Eksik	1.00	1.00	1.00
Doğruluk Oranı	100%		

Tablo 5.4'te ResNet50 mimarisinin performans değerlendirme ölçütleri görülmektedir.

Tablo 5.4. ResNet50 mimarisinin performans metrikleri

Sınıf/Metrik	Kesinlik	Duyarlılık	F1
Arızalı	0.94	1.00	0.97
Sağlıklı	1.00	0.94	0.97
Eksik	1.00	1.00	1.00
Doğruluk Oranı	98%		

Tablo 5.5'te oluşturulan CNN mimarisinin performans değerlendirme ölçütleri görülmektedir.

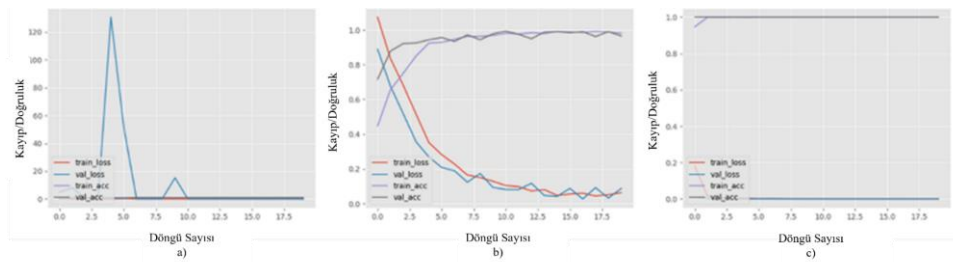
Tablo 5.5. CNN mimarisinin performans metrikleri

Sınıf/Metrik	Kesinlik	Duyarlılık	F1
Arızalı	1.00	1.00	1.00
Sağlıklı	1.00	1.00	1.00
Eksik	1.00	1.00	1.00
Doğruluk Oranı	100%		

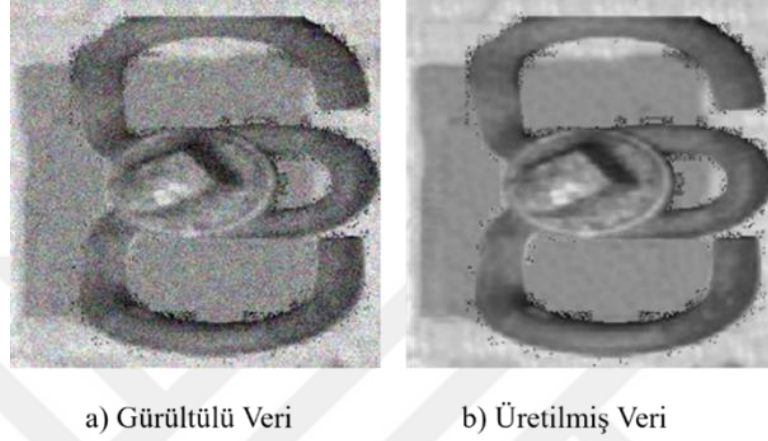
Tablo 5.3 ve Tablo 5.5'te de gösterildiği gibi, CNN ve VGG-16 mimarilerinin performansları, RestNet50 mimarisine göre daha yüksek olmuştur. Çalışmada kullanılan tüm modeller yüksek doğruluk oranlarına ulaşmıştır. CNN ve VGG-16 modelleri, sağlıklı, arızalı ve eksik demiryolu bağlantı elemanlarını %100 doğruluk oranıyla sınıflandırmayı başarmıştır. Bu modellerin öğrenme grafikleri Şekil 5.11'de gösterilmiştir. Performans ölçümleri, seyretme değeri 0.3 olarak belirlendiğinde elde edilmiştir ve seyretme arttırıldığında performansta herhangi bir kayıp yaşanmamıştır. Sonuçlar Tablo 5.6'da özetlenmiştir. Modellerin doğruluk oranlarındaki istikrar, ezberleme yapmadan öğrenme gerçekleştirdiklerini göstermektedir.

Tablo 5.6. Seçilen seyretme değerine göre başarımları

Model	Metrik	Seyretme Değeri (0.3)	Seyretme Değeri (0.5)
CNN	Doğruluk Oranı	100%	100%
CNN	Duyarlılık	1.00	1.00
RestNet50	Doğruluk Oranı	98%	98%
RestNet50	Duyarlılık	1.00	1.00
VGG-16	Doğruluk Oranı	100%	100%
VGG-16	Duyarlılık	1.00	1.00

**Şekil 5.11.** Önerilen modellerin öğrenme grafikleri (a) CNN (b) ResNet50 (c) VGG-16

DATEM'den sađlanan veri seti iyileřtirme ve veri üretim tekniklerine tabi tutulmuş, ardından derin öğrenme modelleriyle eğitilerek test edilmiştir. Bölütleme sonucu elde edilen verilere sentetik gürültü eklendiğinde, önerilen modellerin başarımında kayda değer bir düşüş gözlenmemiştir. Şekil 5.12'de, gürültü eklenmiş ve bölütleme sonucu oluşturulmuş bağlantı elemanları görsel olarak sunulmaktadır.



Şekil 5.12. (a) Sentetik gürültü eklenmiş bağlantı elemanı (b) İyileştirilmiş ve üretilmiş bağlantı elemanı

Tablo 5.7'ye göre, gürültü eklenmesi başarımlarını olumsuz etkilememiştir. Farklı çeşitlilikte kusurlu görüntülerin üretilmesi, önerilen modellerin başarımında temel bir faktör olarak öne çıkmıştır; bu sayede gürültülü veri setleri üzerinde performans sorunlarının çözülebileceği kanıtlanmıştır. Önerilen CNN modelinin başarımları, literatürdeki diğer temel yöntemlerle karşılaştırılmıştır ve bu karşılaştırmanın sonuçları Tablo 5.8'de gösterilmektedir.

Demiryollarındaki arızaların erken tespiti hem ekonomik hem de güvenlik açısından önemli bir konudur. Tren raylarındaki arızalı ve eksik bağlantı elemanlarının tespiti, önemli sorunlardan biridir. Bu çalışmada, bu sorunun çözümüne yönelik olarak CNN tabanlı bir derin ağ modeli geliştirilmiştir. Önerilen model iki ana aşamadan oluşmaktadır: İlk olarak, demiryolu veri setindeki dengesizlikler Unet yardımıyla dengelenmiştir. Daha sonra, çeşitli derin öğrenme modelleri ile ray kusurları sınıflandırılmıştır. Bu CNN mimarisinin iki temel avantajı bulunmaktadır: Basit bir yapıya sahip olup, daha az işlem süresi gerektirir ve literatürdeki diğer sınıflandırma modellerine göre daha yüksek sınıflandırma başarımları sunar. Çalışma ayrıca demiryolu bağlantı elemanlarını içeren yeni bir veri seti de sağlamaktadır. Modelin performansını doğrulamak için çeşitli deneyler gerçekleştirilmiştir. Aynı zamanda, önerilen modeller gürültülü veri seti ile de test edilmiş olup, performanslarında herhangi bir düşüş gözlenmemiştir.

Tablo 5.7. Önerilen modellerin gürültülü ve üretilmiş veri seti üzerindeki performansları

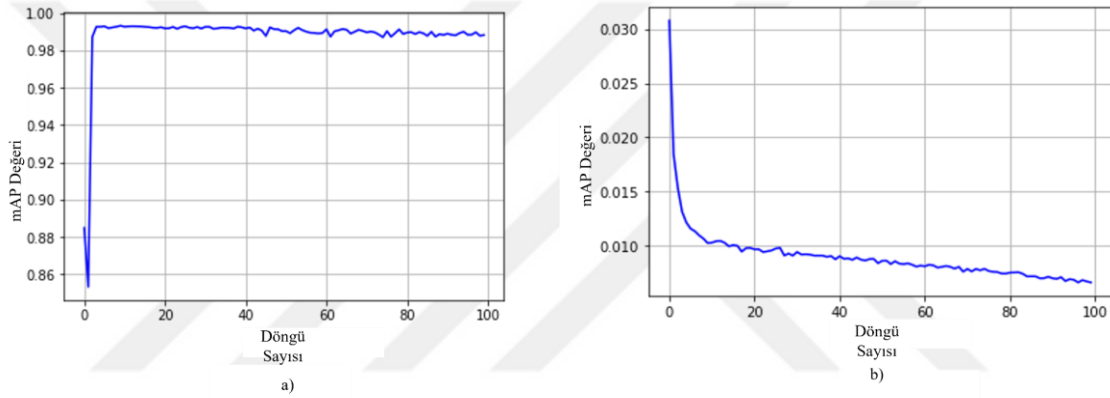
Veri Seti	Mimari	Etiket	P	Etiket	R	Etiket	F1	A	Döngü Sayısı
Gürültülü Veri Seti	VGG-16	Sağlıklı	1.00	Sağlıklı	1.00	Sağlıklı	1.00	1.00	20
		Arızalı	1.00	Arızalı	1.00	Arızalı	1.00		
		Eksik	1.00	Eksik	1.00	Eksik	1.00		
Gürültülü Veri Seti	ResNet50	Sağlıklı	1.00	Sağlıklı	0.94	Sağlıklı	0.97	0.98	20
		Arızalı	0.95	Arızalı	1.00	Arızalı	0.97		
		Eksik	1.00	Eksik	1.00	Eksik	1.00		
Gürültülü Veri Seti	Oluşturulmuş CNN	Sağlıklı	1.00	Sağlıklı	1.00	Sağlıklı	1.00	1.00	20
		Arızalı	1.00	Arızalı	1.00	Arızalı	1.00		
		Eksik	1.00	Eksik	1.00	Eksik	1.00		
Üretilmiş Veri Seti	VGG-16	Sağlıklı	1.00	Sağlıklı	1.00	Sağlıklı	1.00	1.00	20
		Arızalı	1.00	Arızalı	1.00	Arızalı	1.00		
		Eksik	1.00	Eksik	1.00	Eksik	1.00		
Üretilmiş Veri Seti	ResNet50	Sağlıklı	1.00	Sağlıklı	0.93	Sağlıklı	0.97	0.98	20
		Arızalı	0.94	Arızalı	1.00	Arızalı	0.97		
		Eksik	1.00	Eksik	1.00	Eksik	1.00		
Üretilmiş Veri Seti	Oluşturulmuş CNN	Sağlıklı	1.00	Sağlıklı	1.00	Sağlıklı	1.00	1.00	20
		Arızalı	1.00	Arızalı	1.00	Arızalı	1.00		
		Eksik	1.00	Eksik	1.00	Eksik	1.00		

Tablo 5.8. Önerilen CNN Modelinin Literatürdeki Modellerle Karşılaştırılması

Referans	Model	Etiket	P	Etiket	R	Etiket	F1	A
[14]	Olasılıksal yapı modellemesi	Sağlıklı	0.97	Sağlıklı	0.97	Sağlıklı	0.97	0.96
		Arızalı	0.97	Arızalı	0.80	Arızalı	0.81	
		Eksik	0.97	Eksik	0.88	Eksik	0.85	
[127]	En yakın komşuya dayalı şablon eşleştirme	Sağlıklı	0.92	Sağlıklı	0.97	Sağlıklı	0.94	0.96
		Arızalı	0.67	Arızalı	0.87	Arızalı	0.75	
		Eksik	0.83	Eksik	0.87	Eksik	0.85	
[132]	Geliştirilmiş Alexnet	Sağlıklı	0.97	Sağlıklı	0.92	Sağlıklı	0.95	0.95
		Arızalı	0.96	Arızalı	1.00	Arızalı	0.98	
		Eksik	1.00	Eksik	0.99	Eksik	0.99	
Önerilen Yöntem	Önerilen CNN	Sağlıklı	1.00	Sağlıklı	1.00	Sağlıklı	1.00	1.00
		Arızalı	1.00	Arızalı	1.00	Arızalı	1.00	
		Eksik	1.00	Eksik	1.00	Eksik	1.00	

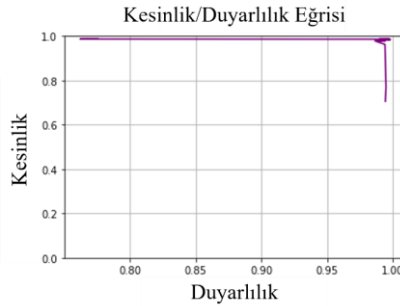
5.3. Bağlantı Elemanlarının Tespiti ve Hata Sınıflandırması için Önerilen Hibrit Yaklaşım Metodunda Elde Edilen Bulgular

Önerilen derin öğrenme tabanlı yaklaşımla, bağlantı elemanlarını tespit etmek ve kusurları belirlemek için 8461 bağlantı elemanı görüntüsü içeren bir veri seti kullanıldı [128]. Veri seti, bir demiryolu aracına yerleştirilen kamera ile alınmıştır. Algılama ve tanıma ağırları, 8 GB belleğe sahip GTX2070 Max Q GPU kartına sahip bir mobil iş istasyonu bilgisayarında eğitilmiştir. YOLOv4-Tiny modelinin ImageNet üzerinde transfer öğrenimi ile önceden eğitilmiş bir modeli kullanılmıştır. Model eğitiminde 64 batch büyüklüğü alınmış ve 100 epokluk eğitim gerçekleştirilmiştir. Öğrenme oranı $1e-4$ olarak alınmış ve optimize edici olarak SGD seçilmiştir. Şekil 5.13, YOLOv4-Tiny eğitimindeki mAP ve kayıp grafiğini göstermektedir.



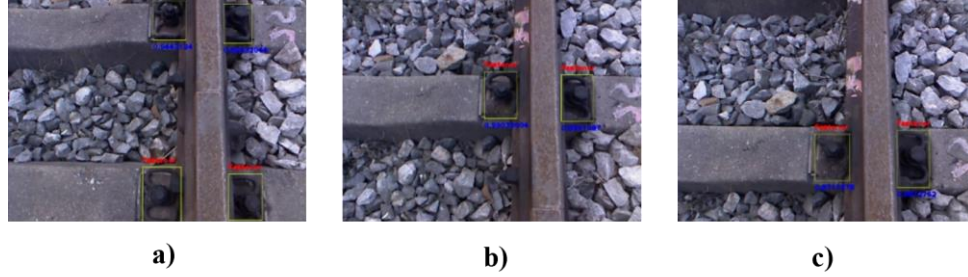
Şekil 5.13. YOLOv4-Tiny'un mAP ve kayıp grafiği. (a) mAP (b) Kayıp grafiği

Şekil 5.13'te elde edilen kayıp ve mAP grafiğine göre bağlantı elemanlarının tespit oranı oldukça yüksektir. Ayrıca, kesinlik-hatırlama eğrisi, performans ölçümü için kullanılabilecek başka bir ölçüdür. Bu eğri 1'e ne kadar yakınsa, modelin performansı o kadar yüksek olur. YOLOv4-Tiny modeli için kesinlik-duyarlılık eğrisi Şekil 5.14'te verilmiştir.



Şekil 5.14. YOLOv4-Tiny için kesinlik-duyarlılık eğrisi

Bu eğri 1'e yakın olduğu için hem kesinlik hem de hatırlama açısından iyi performans gösterdiği söylenebilir. Şekil 5.15'te farklı numuneler için elde edilen tespit sonuçları verilmiştir.



Şekil 5.15. Farklı koşullar için bağlantı elemanı algılama sonuçları. (a) 4 adet sağlıklı bağlantı elemanı, (b) 1 adet sağlıklı 1 adet kusurlu bağlantı elemanı, (c) 1 adet eksik 1 adet sağlıklı bağlantı elemanı

Şekil 5.15'te YOLOv4-Tiny modelinin sağlamlığını test etmek için arızalı ve eksik bağlantı elemanları da verilmiştir. Bağlantı elemanının eksik veya kısmen kırıldığı durumlarda algoritma algılama işlemini gerçekleştirir. Bağlantı elemanları tespit edildikten sonra, kusur tespiti için CNN modeli çalıştırılır. Bağlantı elemanı için sağlıklı, kusurlu ve eksik bağlantı elemanları dikkate alınır. Her bir durum için veri setindeki örnek sayıları Tablo 5.9'da verilmiştir.

Tablo 5.9. Her bir sınıf için örnek sayıları

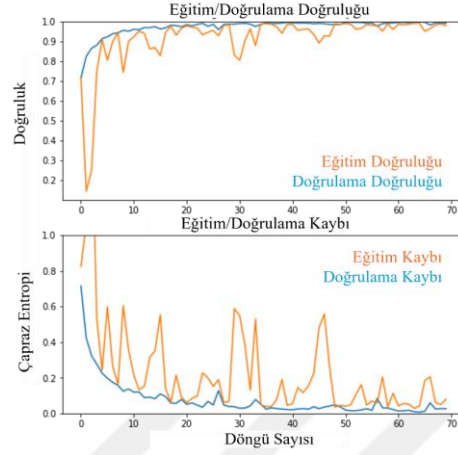
Sınıf	Örnek Sayısı
Sağlıklı	5000
Kusurlu	1000
Eksik	1000

Tablo 5.9'da kusurlu numune sayısı sağlıklı numune sayısından daha azdır. Özellikle performans ölçütlerindeki sınıf dengesizliğini hesaba katmak için veri seti bu şekilde seçilmiştir. Önerilen derin sinir ağının eğitim parametreleri Tablo 5.10'da verilmiştir.

Tablo 5.10. Hafif evrişimli CNN ağının parametreleri

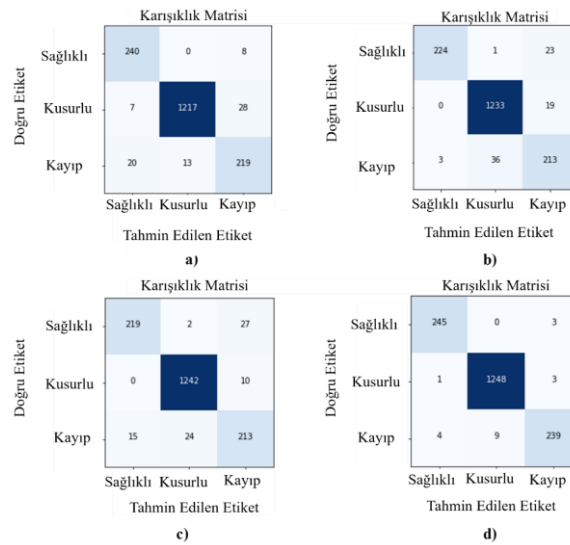
Parametre	Değer
Öğrenim oranı	1e-3
Optimizasyon fonksiyonu	Adam
Batch boyutu	32
Döngü Sayısı	70

Derin öğrenme ağı Tablo 5.10'daki parametrelere göre çalıştırılmıştır. Buna göre, eğitim ve doğrulama için sınıflandırma performansı ve kayıp fonksiyonu Şekil 5.16'da verilmiştir.



Şekil 5.16. Eğitim ve doğrulama veri seti için doğruluk ve kayıp grafiği

Şekil 5.16'da verilen doğruluk ve kayıp değerlerinde, belirli bir iterasyondan sonra hem eğitim hem de doğrulama veri setlerinde doğruluğun sabitlendiği görülmektedir. Önerilen CNN tabanlı hata sınıflandırma ağının performansı da literatürde bilinen transfer öğrenme yöntemleri ile karşılaştırılmıştır. Bu amaçla önerilen yöntem MobilenetV2 [100], InceptionV3 [129] ve VGG16 [130] gibi yöntemlerle karşılaştırılmıştır. Her bir yöntem için karışıklık matrisleri Şekil 5.17'de verilmiştir.



Şekil 5.17. Farklı CNN modellerinin karışıklık matrisleri. (a) InceptionV3 (b) VGG16, (c) MobileNetv2, (d) Önerilen CNN

Şekil 5.17'de elde edilen karışıklık matrisleri üzerinde sınıflandırma için performans metrikleri karşılaştırılmıştır. Bu amaçla, karışıklık matrislerinin doğruluk, kesinlik, hatırlama ve F1 kriterleri karşılaştırılmıştır. Ayrıca her bir yöntem için parametre sayısı ve işlem süreleri değerlendirilmiştir. Bu karşılaştırma sonuçları Tablo 5.11'de verilmiştir.

Tablo 5.11. Farklı transfer öğrenme modelleriyle performans karşılaştırması

Model	Kesinlik	Duyarlılık	F1	Doğruluk	İşlem Zamanı (ms)	Eğitilebilen Parametre (Milyon)	Model Boyutu (MB)
InceptionV3	91.66	93.66	92.33	96.00	6.0	23.8	97
VGG16	93.33	91.00	92.00	95.00	6.4	138.4	204
MobileNetV2	92.33	90.66	91.66	96.00	2.17	3.4	16.6
Önerilen CNN	98.33	98.00	97.66	99.00	1.53	0.3	1.71

Bağlantı elemanı hata tespiti için başarı oranı yüksek modeller seçilmiştir. Bu modellerden bazıları çok sayıda parametreye sahiptir. Ancak önerilen model, darboğaz bloklarının yanı sıra kullandığı derinlik ve noktasal evrişim katmanları sayesinde küçük bir ağ yapısına sahiptir. Ayrıca son katmanda kullanılan tam bağlantılı katman sayesinde ağın genelleme özelliği artırılmıştır. Önerilen yöntemin etkinliğini göstermek için, bağlantı elemanı hata tanıma için literatürde önerilen yaklaşımlarla karşılaştırmalar yapılmıştır. Karşılaştırma sonuçları Tablo 5.12'de verilmiştir.

Tablo 5.12. Bağlantı elemanı tanıma yöntemlerinin karşılaştırma sonuçları

Referans	Metot	Sınıf Sayısı	Doğruluk Oranı (%)	FPS
[131]	Derin öğrenme tabanlı bölütleme ve şablon eşleştirme	3	93.30	21.10
[13]	Faster RCNN tabanlı bağlantı elemanı denetimi	3	97.90	260.00
[58]	Kontrast geliştirme ve geliştirilmiş CNN	3	96.80	221.00
[11]	İki CNN kullanarak görsel benzerlik	4	92,69	196.00
[128]	RGBD-görüntü tabanlı SVM	3	95.83	71.37
[132]	Unet tabanlı bölütleme ve geliştirilmiş AlexNet	3	97.52	188.00
Önerilen	YOLOv4_Tiny+Geliştirilmiş CNN Modeli	3	98.57	279.06

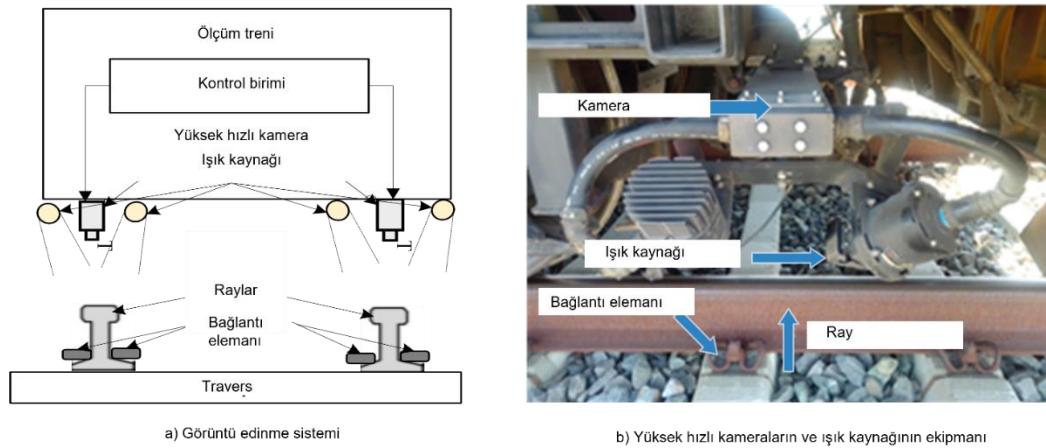
Tablo 5.12'de verilen karşılaştırma sonuçlarında, [131]'te önerilen yaklaşım, önce YOLACT tabanlı bölütleme yöntemiyle bağlantı elemanlarını böler ve daha sonra desen eşleştirme ile hatalı

bağlantı elemanları bulur. Yöntem, ayrıntılı etiketleme gerektirir ve rayın sağındaki ve solundaki elemanlar kusurluysa yanlış sonuçlar verir. [13]'de sunulan çalışmada, ray bağlantı elemanı tespiti için Faster RCNN yöntemi kullanılmış ve hem hata sınıflandırması hem de tespiti gerçekleştirilmiştir. [58]'deki çalışmada, bağlantı elemanının konum belirlemesini daha doğru hale getirmek için görüntü iyileştirme uygulanmıştır. Ancak kullanılan veri setinin boyutu küçük ve dengelidir. [11]'te önerilen yaklaşım, iki aşamalı bir evrişimli sinir ağı önermektedir. Son aşamada ise ayrılmış ağ yapısında hem sınıflandırma hem de benzerlik karşılaştırmaları yapılmıştır. [99]'daki çalışmada, sınıflandırma için RGBD görüntüleri elde edilmiştir ve bu görüntülerin alınması ekstra bir yük getirmektedir. Tablo 5.12'deki karşılaştırma sonuçları incelendiğinde, önerilen yöntemin hem doğruluk hem de hız açısından daha iyi olduğu görülmektedir.

5.4. Demiryolu Bağlantı Elemanları Üzerindeki Kusurların Tespit Edilmesi Metodunda Elde Edilen Bulgular

Bir bağlantı elemanı kusur tespit sistemi, yüksek hızlı bir görüntü elde etme sistemi, bir görüntü işleme ve tanıma sistemi, bir veri yönetim sistemi ve bir merkezi kontrol sisteminden oluşmaktadır.

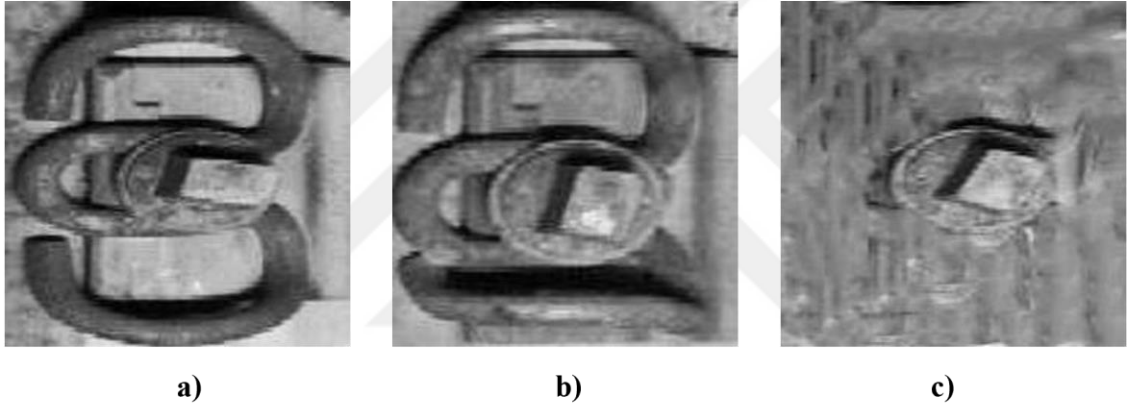
Görüntü toplama sistemi, sürekli görüntü toplamak için bir ölçüm dizisinin altına yerleştirilmiş yüksek hızlı kameralar ve aydınlatma cihazlarından oluşur. Toplanan görüntüler merkezi kontrol sistemine iletilir ve GPS bilgileri ile kayıt altına alınır. Bu çalışmada, bir ölçüm trenine monte edilen Türkiye Cumhuriyeti Devlet Demiryolları Araştırma Merkezi (DATEM) deney sistemi kullanılarak farklı zamanlarda gerekli verileri elde ettik. Şekil 5.18, verilerin elde edildiği ölçüm sistemini temsil etmektedir.



Şekil 5.18. Ölçüm sisteminin yapısı

Şekil 5.18'de gösterildiği gibi, bağlantı elemanı görüntüleri, ölçüm dizisinin altına dikey bir açıyla yerleştirilmiş iki hatlı sahne kameraları ile toplanır. Sağ ve sol ışın görüntüleri ayrı kameralarla toplanır. Dış ortamdaki ışık farklılıklarını ortadan kaldırmak için her kamera iki LED ışıkla donatılmıştır. Kamera çözünürlüğü 2096 piksel, kameranın hat hızı 68.000 satır/saniye ve görüntü çözünürlüğü 1 mm x 0.15 mm'dir. Veri setinde sağlıklı bağlantı elemanlarına ait 525 görüntü, hasarlı bağlantı elemanlarına ait 594 görüntü ve eksik bağlantı elemanlarına ait 479 görüntü bulunmaktadır. Tüm görüntüler Türkiye'deki Ankara-Konya ve Ankara-Eskişehir demiryolu hatları boyunca toplanır ve el ile etiketlenmiştir.

Veri setinde iki tip bağlantı elemanı hatası olduğu bulunmuştur. Bir tip kısmen deforme olmuş bir bağlantı elemanıdır ve diğer tip tamamen kayıp bir bağlantı elemanıdır. Her durum için örnekler Şekil 5.19'da verilmiştir.



Şekil 5.19. Üç farklı bağlantı elemanı durumu (a) Sağlıklı (b) Deforme (c) Kayıp

Şekil 5.19'da üç farklı durum için gösterilen bağlantı elemanı görsellerinde kısmen aşınmış veya tamamen eksik olan bağlantı elemanları tren geçişi sırasında daha ciddi sorunlara neden olarak tren çalışma güvenliğini olumsuz etkilemektedir. Görüntü alma sistemi ile çekilen görüntüler raylar, traversler, bağlantı elemanları ve balastlardan oluşmaktadır. Bu çalışmanın amacı bağlantı elemanı hatalarını belirlemek olduğundan, bağlantı elemanı kısmi görüntüden çıkarılmalıdır. Ancak elde edilen görüntüler, trenin altındaki aydınlatma koşullarından ve günün farklı saatlerinde çekilen görüntülerin neden olduğu kontrast farklılıklarından dolayı gürültü içermektedir. Bu nedenle orijinal görüntüye kontrast iyileştirme yöntemi uygulanmıştır. Görüntü iyileştirme aşaması için kullanılan fminsearch fonksiyonunun başlangıç parametreleri $kr=1$ ve $kg=1$ olarak seçilmiştir.

Bağlantı elemanı tespiti için önce hem ray hem de travers tespit edilmelidir. Ray ve travers konumlarını elde etmek için Otsu yöntemi kullanılarak yüksek kontrastlı bir görüntü segmentlere ayrılır; ray konumu belirlenir ve rayın soluna yakın bir noktadan bir filtre kullanılarak traverslerin konumları belirlenir; yüksek kontrastlı görüntüden elde edilen traverslere LLBP algoritması

uygulanır ve son olarak kırılmış travers görüntülerinden morfolojik işlemler uygulanarak bağlantı elemanı konumları elde edilir. LLBP sonucunda elde edilen ikili görüntüde bağlantı elemanlarının konumunu belirlemek için morfolojik yakınlaştırma uygulanmıştır. Bu işlem için yapı elemanı kare olarak seçilmiş ve komşuluk değeri 15 olarak alınmıştır.

Ray görüntüsünden bağlantı elemanı bölgesi elde edildikten sonra veri seti oluşturulmuştur. Eğitilmiş ağ modeli, aşağıdaki eğitim parametresi değerleriyle oluşturulmuş veri seti LCNN modeline verilerek elde edilir: ilk öğrenme oranı = 10-4, maksimum döngü = 30, mini batch boyutu = 32, doğrulama frekansı = 15 ve optimizasyon = momentumlu stokastik gradyan inişi.

Eğitim sırasında hem geliştirilmiş görüntüler hem de herhangi bir ön işleme tabi tutulmadan doğrudan trenin altından alınan görüntüler kullanılır. Ayrıca yansıma ve ölçekleme gibi veri büyütme her iki yönde de uygulanmaktadır. Veri seti, %70 eğitim ve %30 doğrulama olarak bölümlere ayrılmıştır. Doğrulama performansı her koşul için karşılaştırılır ve sonuçlar Tablo 5.13'te verilmiştir.

Tablo 5.13. Görüntü iyileştirme öncesi ve sonrası karşılaştırmalı sonuçlar

Veri seti	Doğrulama Oranı (%)
Orjinal görüntüler	91.37
Orjinal görüntülerde veri artırma	92.58
İyileştirilmiş görüntüler	93.15
Arttırılmış ve iyileştirilmiş görüntüler	96.82

Tablo 5.13'te geliştirilmiş görüntü üzerinde model eğitiminin doğrulama sonuçlarının orijinal görüntüye göre daha iyi olduğu görülmektedir. Bu sonuçlar şaşırtıcı değildir, çünkü orijinal görüntülerde, aydınlatma koşullarının etkileri nedeniyle görüntüyü temsil eden özelliklerin çıkarılması zordur. Ayrıca görüntü büyütme adımından sonra performansta artış olur. Tam bağlantılı, softmax ve sınıflandırma katmanları ile model eğitimi yeterli performansı sağlamamaktadır. Bu durum, kullanılan veri setinin çeşitliliği ve zorluğundan da kaynaklanabilir. Modelin genel başarısını artırmak için, veri ön işleme adımlarının optimize edilmesi önem taşır. Amacımız için, en iyi sınıflandırma yöntemini seçmek için sınıflandırma katmanından önce tam bağlantılı iki katmandan elde edilen öznitelikler birleştirilecektir. Tam bağlantılı katmanların çıktı boyutu da ağ performansını etkiler. Doğru yapılandırılmış bir ağ mimarisi, sınıflandırma doğruluğunun önemli ölçüde artmasına olanak sağlar. Son tam bağlı katman, sınıf sayısını temsil eder ve çalışmamızda üç sınıf olduğu için üç olarak seçilmiştir. Ancak tam bağlı katman-1 (FC1) ve tam bağlı katman-2 (FC2) doğru seçilmelidir. Tablo 5.14'te FC1 ve FC2 için seçilen boyutlara göre elde edilen özniteliklerin sınıflandırma performansı verilmiştir.

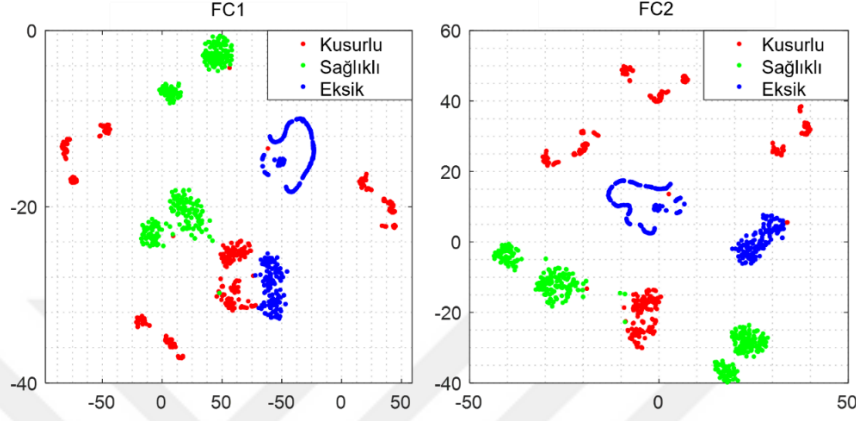
Tablo 5.14. Farklı tam bağlantılı katmanlar için önerilen CNN'nin performansı

Tamamen bağı katmanların boyutu		Birleşik özelliklerin boyutu	Birleştirilmiş özelliklerin doğruluk oranı
FC1	FC2		
100	50	150	93.11
150	100	250	94.15
200	150	350	95.00
250	200	450	96.61
300	250	550	99.70
350	300	650	97.2
400	350	750	97.9
450	400	850	97.9
500	450	950	97.1

Tablo 5.14'te, sınıflandırma için Matlab Classification Learner aracına seçilen öznitelikler verilmiştir. Burada, en yüksek performansı veren sınıflandırıcıya göre birleştirilmiş özniteliklerin sınıflandırma performansı seçilir. Tablo 5.14'te en iyi performans FC1 ve FC2 için sırasıyla 300 ve 250 öznitelik seçildiğinde elde edilir. CNN'leri eğitmek için yeterli veri olmadığında, sınıflandırma aşamasından önce önceden eğitilmiş bir CNN'nin katmanından elde edilen öznitelikler bir makine öğrenmesi yöntemine verilir ve sınıflandırma yapılır. Önceden eğitilmiş bir ağdan özellik çıkarmı, daha fazla eğitim gerektirmediğinden, düşük özellikli bir bilgisayarda bile yapılabilir. Ancak bu önceden eğitilmiş ağlarda elde edilen özniteliklerin boyutsallığı oldukça yüksektir. Bu çalışmada daha küçük bir boyut özelliği çıkarılarak bağlantı elemanı hata sınıflandırması yapılmıştır. Tablo 5.15'te, milyonlarca (M) cinsinden eğitilebilir parametre sayısı ve önerilen CNN mimarisinin özellik çıkarma performansı, son teknoloji ürünü önceden eğitilmiş ağlarla karşılaştırılmıştır.

Tablo 5.15'te verilen önceden eğitilmiş ağlar, özellikle küçük boyutlu ve daha az parametrelili olanlardan seçilmiştir. Örneğin, SqueezeNet Alexnet'e yakın performans göstermesine rağmen 50 kat daha hızlı çalışır. GoogleNet'in mimarisi 22 katmana sahiptir ve model boyutu diğer CNN'lere göre düşüktür. Aynı şekilde VGG19 modeli de 16 katmanlı bir ağ modelidir ve Imagenet veri setinde biraz daha yüksek doğruluk oranına sahiptir. 737 örnek üzerinden öznitelik çıkarımları karşılaştırılmış ve bir görüntü için öznitelik çıkarma süresi verilmiştir. Buradan da görüleceği üzere önerilen ağ modelinin öznitelik çıkarma performansı diğer modellere göre daha iyidir ve gerçek zamanlı uygulamalarda kullanılması mümkün görünmektedir. İstatistiksel bir yöntem olan T-dağıtılmış stokastik komşu gömme (t-SNE), önerilen CNN modelinin tam bağlantılı iki katmanından elde edilen özniteliklerin dağılımını göstermek için kullanılır. Teknik, bir ağ aktivasyon katmanından elde edilen yüksek boyutlu verileri iki boyutlu olarak gösterir. t-SNE

yöntemi, doğrusal olmayan eşlemeyi gerçekleştirirken mesafeleri korur. Böylece belirli bir katmandan elde edilen özelliklerin dağılımı öğrenilerek hatalı dağılım gösteren örnekler belirlenebilir. Şekil 5.20’de, önerilen CNN mimarisinin FC1 ve FC2 katmanının bir t-SNE temsili verilmektedir.



Şekil 5.20. FC1 ve FC2’den çıkarılan özelliklerin t-sne temsili

Tablo 5.15 . Önerilen CNN ve temel önceden eğitilmiş CNN modelleri için özellik çıkarma süreci

CNN Modeli	Öğrenilebilir Katman Sayısı	Öğrenilebilir Parametre Sayısı	Özellik Katmanı	Özellik Sayısı	Zaman (ms)
SqueezeNet	18	1.24 M	pool10	1000	8.46
GoogleNet	22	6.7 M	pool5-drop_7x7_s1	1024	18.00
VGG19	16	143 M	avg_pool	2048	38.47
LCNN	8	0.9 M	FC1-FC2	550	6.58

Şekil 5.20’de görüldüğü gibi, FC1 katmanının t-SNE temsiliinde sağlıklı ve kusurlu verileri ayırt etmek daha zordur. Buna karşın, FC2 katmanında bazı bozulmamış ve deforme olmuş örnekler daha net ayırt edilebilmektedir. Önerilen yöntem, FC1 ve FC2’den elde edilen öz nitelikleri birleştirip makine öğrenmesi yöntemleriyle sınıflandırmaktadır. Bu amaçla, farklı sınıflandırma yöntemleri kullanılmış ve en yüksek doğruluğu veren sekiz yöntemin sonuçları sunulmuştur. Yöntem, makine öğrenmesi ile çıkarılan öz niteliklerin yanı sıra, CNN’deki tam bağlantılı ve softmax katmanları ile de karşılaştırılmıştır. Tüm önceden eğitilmiş ağlar için parametreler aynıdır ve sınıflandırma aşamasında 5 kat çapraz doğrulama uygulanmıştır. Test performanslarının ortalaması alınarak doğruluk değeri hesaplanmıştır. Önerilen CNN modelinin önceden eğitilmiş ağlarla karşılaştırma sonuçları Tablo 5.16’da sunulmuştur.

Tablo 5.16. Farklı önceden eğitilmiş CNN modelleriyle karşılaştırma sonuçları

Sınıflandırıcı	CNN modellerini doğruluk oranları (%)			
	SqueezeNet	GoogLeNet	VGG19	Bizim
Fine Tree	89.2	92.3	90.5	93.9
Linear Discriminant	86.3	89.6	88.8	92.6
Gaussian Naïve Bayes	91.4	95.4	96.7	96.8
Cubic SVM	96.9	97.8	98.4	99.7
Weighted KNN	93.3	94.4	95.6	96.2
Ensemble Subspace	96.2	97.9	97.2	96.3
KNN				
Ensemble Boosted Tree	50.6	53.3	60.4	51.4
Ensemble RusBoosted	90.7	93.4	94.7	94.8
Trees				
Softmax	92.4	93.7	94.8	96.8

Tablo 5.16'da seçilen önceden eğitilmiş ağların ilgili katmanlarından elde edilen sonuçlar, önerilen yöntemin sonuçları ile karşılaştırıldığında, dokuz sınıflandırma yönteminden yedisinde önerilen yöntemlerin daha iyi sonuçlar verdiği görülmektedir. Önerilen CNN mimarisi, daha az özellik kullanır ve performansı, son teknoloji ürünü önceden eğitilmiş ağlardan daha yüksektir. Cubic SVM, makine öğrenme yöntemleri arasında en iyi performansı verdi. Cubic SVM için doğruluk oranı %99,2'dir.

Önerilen yöntemin etkinliğini göstermek için literatürdeki ilgili yöntemleri de karşılaştırdık. Bu yöntemler geleneksel bilgisayarlı görme tekniklerinden ve derin öğrenmeye dayalı tekniklerden oluşmaktadır. Farklı yöntemlerle bağlantı elemanı hata sınıflandırma sonuçları Tablo 5.17'de verilmiştir. Tablo 5.17'deki sınıflandırma sonuçlarında önerilen modelin aynı probleme uygulanan mevcut modellerden daha iyi performans sağladığı görülmektedir. Ayrıca derin öğrenme yöntemlerinin sınıflandırma aşamasında daha iyi performans sağladığı görülmektedir. Olasılıksal yapı modeli, bağlantı elemanı için bir model oluşturarak hata tespitini gerçekleştirir. Ancak bu model birçok parametrenin ayarlanmasını gerektirir. Desen eşleştirme tabanlı en yakın komşu algoritması, Yönlendirilmiş Gradyanların Histogramından elde edilen özelliklerin büyük boyutu nedeniyle yüksek hesaplama gücü gerektirir. Bu nedenle, bu yöntemi kullanan kare hızı düşüktür. Hibrit YOLOv4 kullanan uygulamada, bağlantı elemanı bileşeni geometrik bir nesne olarak etiketlenmiştir. Bu, hassas etiketleme gerektirdiğinden etiketlemenin yavaş olmasına neden olur. Ek olarak, YOLOv4, aydınlatma koşullarına duyarlıdır ve örnek sayısı sınıflar arasında eşit olmayan bir şekilde dağıtıldığında performansı düşüktür. Benzerliğe dayalı derin öğrenme yöntemi, kusur verileri üretirken örnek bağlantı elemanı çiftlerini kullanır. Ancak oluşturulan hata türleri

birbirine çok benzediği için numune sayısı artırılrsa bile sadece sınırlı kusurlu bağlantı elemanlarının özellikleri öğrenilir. Yöntemimiz için örnek sayıları eşit olarak dağıtılır. Ayrıca aydınlatma ile ilgili problemler ön işleme aşamasında çözüldüğü için önerilen derin öğrenme yönteminin performansı arttı. Önerilen yöntemin literatürdeki yöntemlerle karşılaştırılması sırasında kare/saniye (FPS) değerleri dikkate alınmıştır. Önerilen derin öğrenme ağı, önceden eğitilmiş ağ yapılarından daha az katmana sahip olduğundan, nispeten yüksek bir kare hızına sahiptir (burada bildirilen uygulamada 221 FPS).

Tablo 5.17. Önerilen yöntemin literatürdeki temel yöntemler ile karşılaştırılması

Referans	Metot	Sınıf	P	R	F1	Doğruluk (%)	FPS
[14]	Olasılıksal yapı modellemesi	Sağlıklı	0.97	0.97	0.97	96.50	14
		Deforme	0.83	0.80	0.81		
		Kayıp	0.82	0.88	0.85		
[11]	Benzerlik tabanlı derin öğrenme yöntemi	Sağlıklı	0.85	0.85	0.85	92.91	196
		Deforme	0.98	0.96	0.97		
		Kayıp	0.98	0.99	0.99		
[15]	Hibrit YOLOv4	Sağlıklı	0.78	1.00	0.87	94.40	78
		Deforme	0.91	0.98	0.95		
		Kayıp	0.82	0.89	0.85		
[127]	En yakın komşuya dayalı şablon eşleştirme	Sağlıklı	0.92	0.97	0.94	96.34	6
		Deforme	0.67	0.87	0.75		
		Kayıp	0.83	0.87	0.85		
[132]	Geliştirilmiş Alexnet	Sağlıklı	0.97	0.92	0.95	97.52	188
		Deforme	0.96	1.00	0.98		
		Kayıp	1.00	0.99	0.99		
Önerilen Yöntem	Kontrast geliştirme ve hafif evrişimli CNN	Sağlıklı	1.00	1.00	1.00	99.7	221
		Deforme	0.99	0.99	0.99		
		Kayıp	0.99	0.99	0.99		

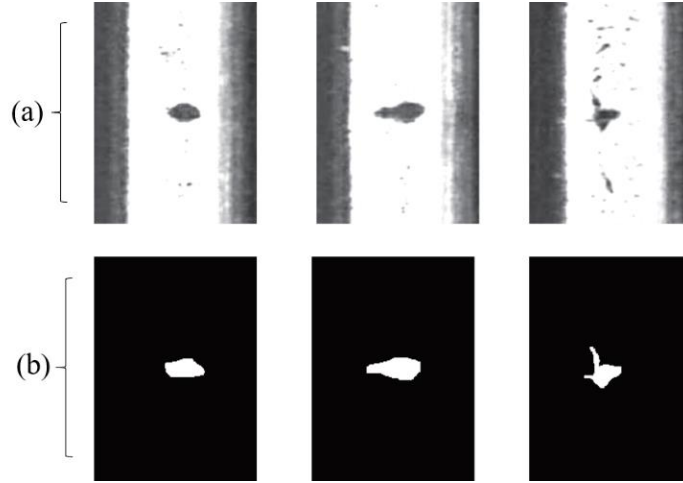
Tren raylarındaki kusurların önceden tespiti hem insan güvenliği hem de ekonomik performans için çok önemli bir konudur. Tren raylarındaki kusurlu bağlantı elemanlarını tespit etmek en büyük zorluklardan biridir. Bu çalışmada, bu sorunu çözmek için CNN tabanlı yeni bir derin ağ modeli önerilmiştir. Önerdiğimiz modelin dört ana aşaması vardır: (1) demiryollarından görüntüler toplanır, (2) yüksek kontrastlı ray görüntüleri elde etmek için görüntü iyileştirme

uygulanır, (3) ray görüntülerinde bağlantı elemanı bileşenleri tam olarak algılanır ve (4) bağlantı elemanı görüntüleri, hafif bir evrişimli sinir ağı kullanılarak sınıflandırılır. Önerilen LCNN mimarisinin iki ana avantajı vardır. Birincisi, mimari olarak basittir ve daha az hesaplama süresi gerektirir. İkincisi, literatürdeki ana sınıflandırma modellerine kıyasla gelişmiş sınıflandırma performansı gösterir. Önerilen derin modelden elde edilen öznetelikleri sınıflandırmak için en bilinen makine öğrenmesi algoritmaları kullanılmaktadır.

Önerilen modelin performansını doğrulamak için bir dizi deney yapıldı. Bu deneylerde, mevcut ana derin mimarilerin performansları, önerilen modelin performansı ile karşılaştırıldı. Deneysel çalışmalar sonucunda önerilen modelin performansının literatürdeki son teknoloji modellere göre daha iyi olduğu gözlemlenmiştir. Önerilen yöntem ayrıca derin öğrenme tabanlı bağlantı elemanı tespit yöntemleri ile karşılaştırılmış ve daha iyi sonuçlar elde edilmiştir. Derin öğrenmeye dayalı tespit yöntemleri, sağlıklı bağlantı elemanlarının tespitine daha fazla odaklanır ve kusurlu bağlantı elemanlarının tespitine daha az uygundur. Bunun temel nedeni, sahada toplanan verilerin çoğunun sağlıklı bağlantı elemanları içermesi ve hatalı veya eksik bağlantı elemanlarının sayısının çok sınırlı olmasıdır. Bu nedenle, gelecekteki çalışmalar, farklı aydınlatma koşulları altında kusurlu ve eksik bağlantı elemanları üretmek için üretken derin öğrenme tekniklerini kullanarak eğitim verilerinin dengesini iyileştirmeyi amaçlayacaktır.

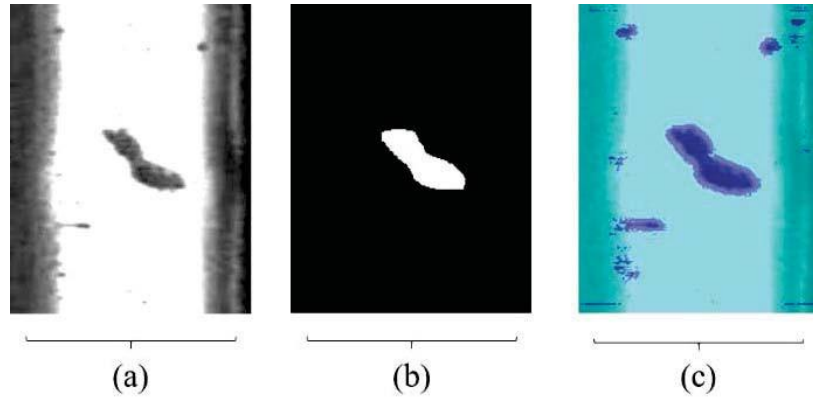
5.5. Anlamsal Bölütleme Kullanarak Ray Üzerindeki Kusurları Tespit Etme Metodunda Elde Edilen Bulgular

Veri setindeki ham görüntüler DATEM'den alınmıştır. Ham görüntüler tarafımızca bölütlenmeden önce görüntü ön işleme ve iyileştirme sürecinden geçirilmiştir. Kullanılan veri seti, 603 ray kusuru içeren görüntülerden oluşmaktadır. Bu görüntülerin 453'ü eğitim için, 150'si ise test için kullanmıştır. Eğitim ve test veri setlerinde orijinal gri tonlamalı görüntülerin yanı sıra referans görüntüleri de mevcuttur. Kullanılan veri setindeki kusurları içeren ray görüntüleri Şekil 5.21'de gösterilmiştir.



Şekil 5.21. Veri setinden örnek görüntüler; (a) Orijinal kusurlu ray görüntüleri, (b) Referans görüntüler

Unet eğitim sürecinde 24 katman kullanılmıştır. 192×256 boyutundaki girdi görüntüler döngü boyutu 100 ayarlanarak eğrtilmiştir. DCNN modeli 5 katmana sahiptir ve Unet modeline benzer aşamalardan geçmiştir. Önerilen modeller, AMD FX-8350 4.00 GHZ ve 32 GB belleğe sahip bir bilgisayarda gerçekleştirilmiştir. Unet modelinin eğitim ve test aşamaları Spyder ile Python 3.6 üzerinde yapılmıştır. DCNN modeli ise Matlab R2018b kullanılarak gerçekleştirilmiştir. DCNN modelinin tahmin edilen görüntüleri, orijinal gri tonlamalı görüntü ve referans görüntü ile karşılaştırılmıştır. Sonuçlar kabul edilebilir seviyededir. Şekil 5.22, raydaki kusurların tespit edildiğini göstermektedir.



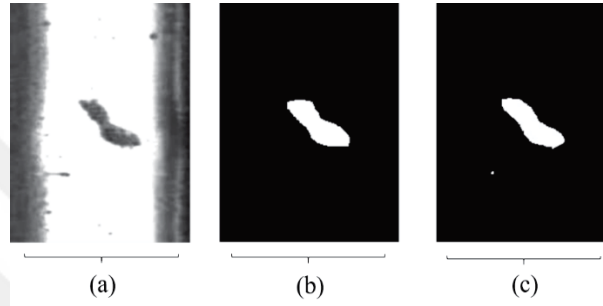
Şekil 5.22. DCNN modelinin sonucu (a) Orijinal gri tonlamalı kusur, (b) Referans kusur (c) DCNN modelinin tahmini

Tablo 5.18, DCNN modelinin performans metriklerini göstermektedir. DCNN tabanlı kusur tespit deneyinde doğruluk oranı %96,23, IoU %64,29 ve F1 skoru %36,11 olarak bulunmuştur.

Tablo 5.18. DCNN modelinin performans metrikleri

Doğruluk Oranı	IoU (Jaccard)	F1 Score (Dice)
0.9623	0.6429	0.3611

Şekil 5.23'te orijinal gri tonlamalı görüntü ile, referans görüntüler ve Unet modelinin tahmin edilen görüntüsü karşılaştırılmıştır. Şekil 5.23'te demiryolu üzerindeki kusurlar DCNN modeline göre daha iyi tespit edildiği görülmektedir. Şekil 5.23, ray üzerinde tespit edilen kusurları göstermektedir.



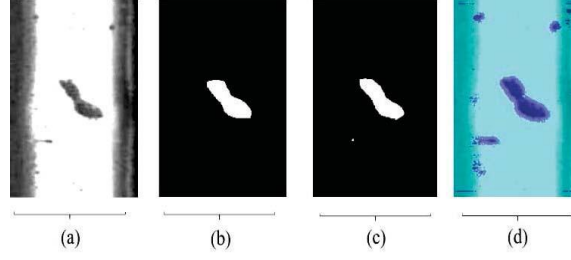
Şekil 5.23. Unet modelinin sonucu (a) Orijinal gri tonlamalı kusur (b) Referans kusur (c) Unet modelinin tahmini

Tablo 5.19 Unet modelinin performans metriklerini göstermektedir. Unet tabanlı kusur tespit deneyinde doğruluk oranı %99,99, IoU %74,93 ve F1 skoru %42,83 olarak bulunmuştur.

Tablo 5.19. Unet modelinde elde edilen başarımlar

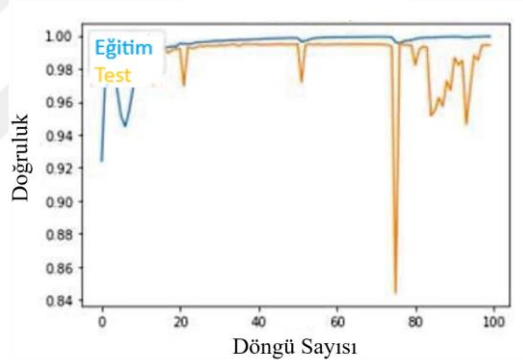
Doğruluk Oranı	IoU	F1 Skoru
0.9999	0.7493	0.4283

Önerilen modeller aynı veri seti üzerinde test edilmiştir. Önerilen modellerin, deneylerde ray üzerindeki kusurları tespit edebildiği ispatlanmıştır. Unet modeli, %99,99 doğruluk oranıyla DCNN modelinden daha iyi bir doğruluk oranı yakalamıştır. İki model arasındaki katman sayısı farkı bu sonuca neden olmuş olabilir. Şekil 5.24'te göre, her iki model tarafından yapılan tahminler, referans görüntüsü ile karşılaştırıldığında, Unet modeli DCNN modeline göre daha iyi bir bölütleme sonucu elde etmiştir.

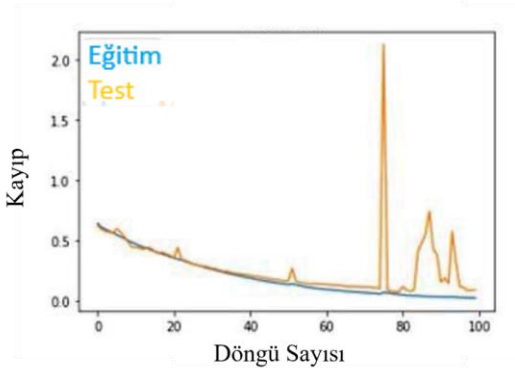


Şekil 5.24. Önerilen modellerin sonuçlarının karşılaştırılması of (a) Orijinal gri tonlamalı kusur, (b) Referans kusur, (c) Unet modelinin tahmini (99.99% doğruluk oranı) ve (d) DCNN modelinin tahmini (96.23% doğruluk oranı)

Eğitim doğruluğu ve kayıp grafikleri Şekil 5.25 ve Şekil 5.26'da verilmiştir. Şekil 5.25 ve Şekil 5.26'da olduğu gibi, düşük kayıplı yüksek doğruluk, düşük hata anlamına gelmektedir [52]. Doğruluk ve kayıp eğrilerini gösteren Unet eğitim grafikleri, kullanılan sınırlı demiryolu eğitim verisine rağmen modelin gereğinden fazla ezber yapmadığını göstermektedir. Şekil 5.25 ve Şekil 5.26, Unet modelinin %99,99 doğruluk oranını doğrulamaktadır.



Şekil 5.25. Unet doğruluk grafiği



Şekil 5.26. Unet kayıp grafiği

Önerilen modellerin performansı literatürde önerilen yaklaşımlarla karşılaştırılmıştır. Karşılaştırma sonuçları Tablo 5.20'de verilmiştir. Tablo 5.20'de görüldüğü gibi Unet modeli, düşük örneklem büyüklüğüne rağmen literatürdeki diğer çalışmalardan daha iyi sonuçlar vermiştir. Unet modeli %99,99 doğruluk oranına sahiptir.

Tablo 5.20. Önerilen modellerin literatürdeki modellerle karşılaştırılması

Referans	Görüntü Sayısı	Model	Döngü Sayısı	Doğruluk Oranı
[7]	22408	DCNN	40	92%
[10]	17412	CNN	-	91.19%
[133]	4101	Unet	50	80.53%
[133]	4101	Deeplabv3+	50	70.51%
[133]	4101	MCnet	50	85.28%
Önerilen	603	Unet	100	99.99%

Önerilen modellerde toplam 603 ray görüntüsü incelendi. Unet modelinin doğruluk oranı, DCNN'ye göre daha iyidir. Unet ile neredeyse mükemmel bir bölütleme sonucu elde edildi ve Tablo 5.20'de görülmektedir. Önerilen model, ray kusurlarını geleneksel yöntemlerden daha verimli hale getirerek demiryolu güvenliğinin sağlanmasına katkıda bulunmuştur. Ayrıca Unet ve DCNN modellerinin raydaki kusurları tespit etmede başarılı olduğu çalışmada kanıtlanmıştır. Bu makalede demiryollarındaki aksaklıkların tespit edilmesinin demiryolu güvenliği açısından önemi vurgulanmıştır. Demiryolunun kullanımı her geçen gün artacağından kusur sorunu da her geçen gün artacaktır. Gelecekteki çalışmalarda, anlamsal bölütlemenin yanına örnek bölütlemeye eklenerek kusur türleri belirlenebilmektedir.

5.6. SAM Kullanarak Demiryolu Raylarının Bölütlenmesi Metodunda Elde Edilen Bulgular

Görüntü bölütlemesi zorlu bir iştir ancak SAM bu alanda devrim yaratma potansiyeline sahiptir. Görüntülerdeki nesneleri segmentlere ayırmayı kolaylaştıran umut verici yeni bir yaklaşımdır. Bu çalışmada veri seti olarak Railsem19 veri seti kullanılmıştır. Tüm deneyler çevrimiçi Google Colab platformunda gerçekleştirildi. Google Colab çevrimiçi bir platformdur. Google Colab, geliştiriciler için Python 3 destekli dizüstü bilgisayar, grafik işlem birimi (GPU), rastgele erişim belleği (RAM) ve depolama alanı sunmaktadır.

SAM'in en büyük avantajlarından biri bu çalışmada kullanılan eğitim verisi olmadan nesneleri segmentlere ayırmasıdır. SAM modeli otomatik maske oluşturmayı çalıştıracak şekilde çalıştırılmıştır. Şekil 5.27'de görüldüğü gibi SAM modeli, herhangi bir eğitim gerektirmeden veri setindeki bir demiryolu hattı görüntüsüne ait nesneleri başarılı bir şekilde bölütlemeyi başarmıştır.



Şekil 5.27. Orijinal (sol) ve segmente edilmiş (sağ) görüntü

Şekil 5.27 ayrıca sol ve sağ rayların başarılı bir şekilde bölümlere ayrıldığını da göstermektedir. Görsel değerlendirmenin yanı sıra, SAM performansı da tahmin edilen IoU metriği ile değerlendirildi. Tablo 5.21’de elde edilen tüm ray bölütlemelerinin IoU sonuçları ayrı ayrı gösterilmektedir.

Tablo 5.21. Bölütleme sonuçları

Demiryolu Bileşeni	IoU
Sol ray	0.9760
Sağ ray	0.9595

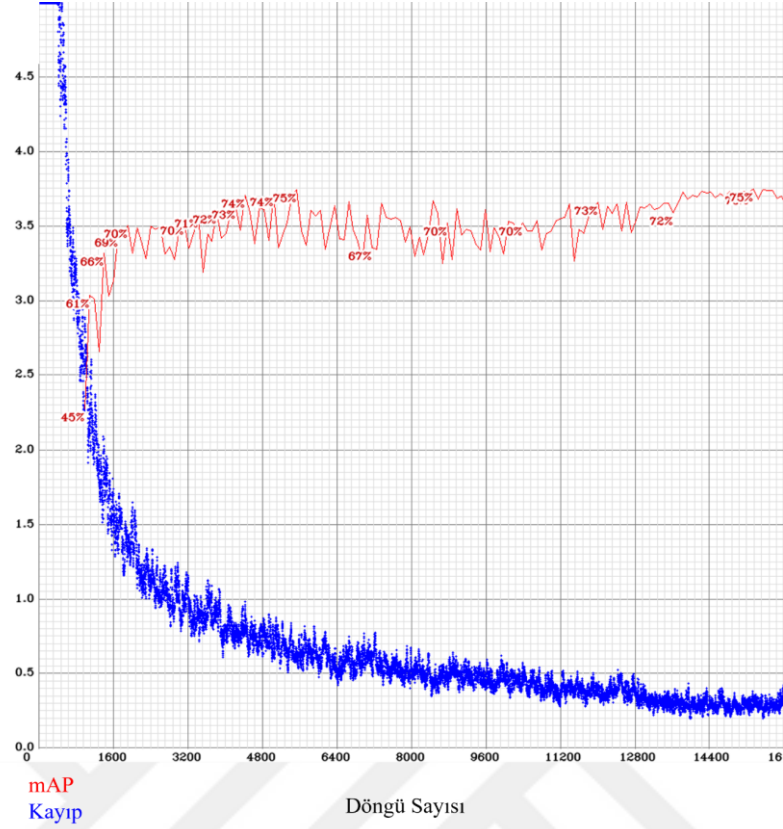
Tablo 5.21’de görüldüğü gibi sol ve sağ raylar başarıyla bölümlendi. Şekil 5.28, elde edilen tüm ray bölümlendirmelerini ayrı ayrı göstermektedir.



Şekil 5.28. Elde edilen sol ve sağ raya bölütleme görselleri

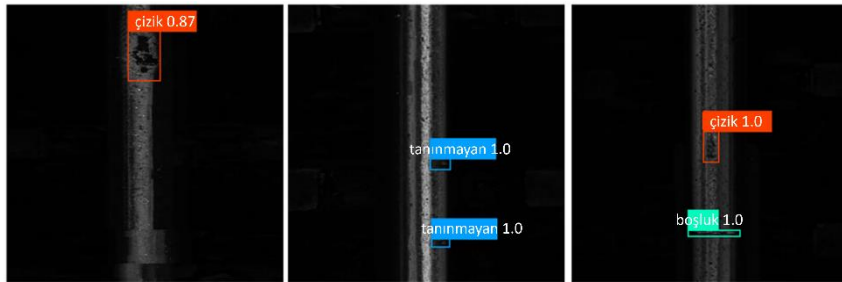
5.7. Topluluk Öğrenmesine Dayalı Olarak Ray Yüzeyindeki Kusurların Tespiti Metodunda Elde Edilen Bulgular

Bu çalışmada eğitilen YOLOv4 modeli, 0.7338 mAP.5 değerine ulaşmıştır. Ayrıca, model 0.73 kesinlik ve 0.69 duyarlılık oranları elde etmiştir. YOLOv4 modeline ait kayıp grafiği ise Şekil 5.29’da sunulmuştur.



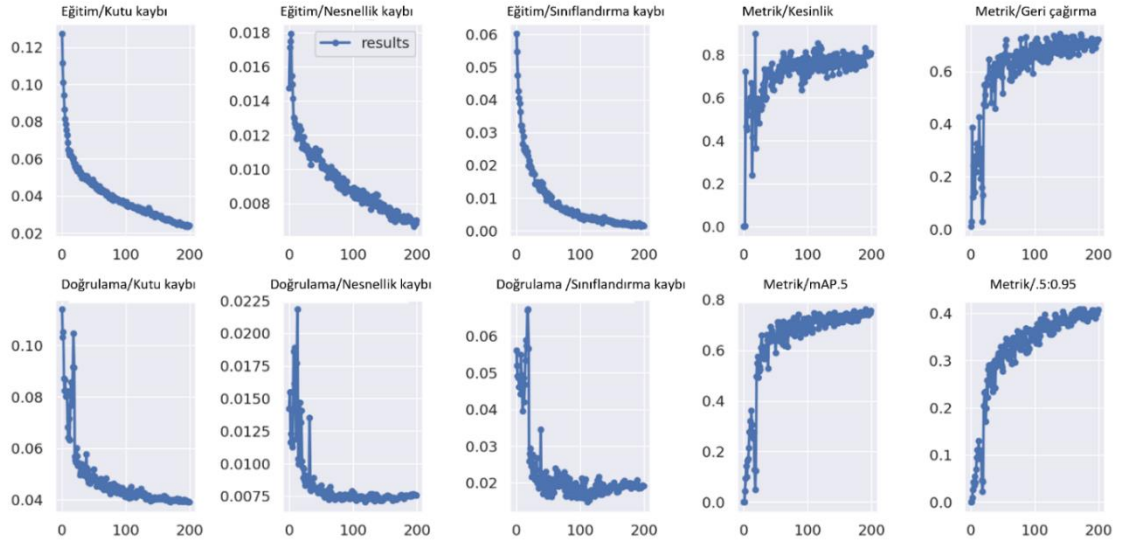
Şekil 5.29. YOLOv4 modelinin kayıp grafiği

Şekil 5.29’da gösterildiği üzere, maksimum batch sayısının artmasıyla birlikte kaybın kademeli olarak azaldığı gözlemlenmiştir. Toplam iterasyon sayısı, maksimum batch değeri ile belirlenir. Yeterli sayıda iterasyona ulaşıldığında, bu durum kayıp oranı ve mAP üzerinde belirgin bir etki yaratmıştır. Ancak, toplam iterasyon sayısı düşük olduğunda, eğitim kaybı ve doğrulama mAP oranları istenen seviyelerde görünmemektedir. Eğitim sonucunda elde edilen ağırlıklar kullanılarak ray kusuru içeren görüntüler test edilip görselleştirildiğinde, Şekil 5.30’da sunulan sonuçlar ve mAP.5 oranları elde edilmiştir.



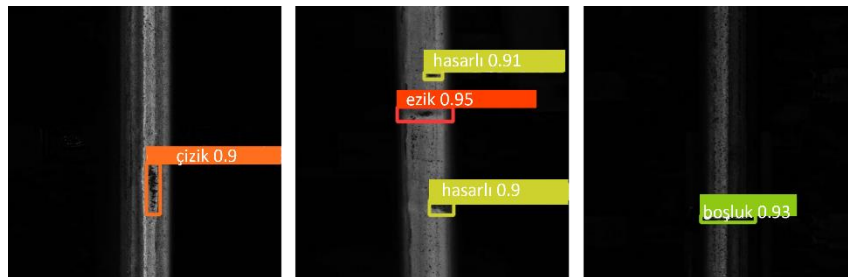
Şekil 5.30. YOLOv4 modelinin test sonuçları

Bu çalışmada, nesne tespiti amacıyla YOLOv5 modelinin farklı alt mimarileri kullanılmıştır. Bu alt mimariler; YOLOv5s, YOLOv5n, YOLOv5m ve YOLOv5l olarak sıralanmaktadır. Şekil 5.31’de, YOLOv5s alt mimarisine ait eğitim ve doğrulama grafiklerine yer verilmiştir.

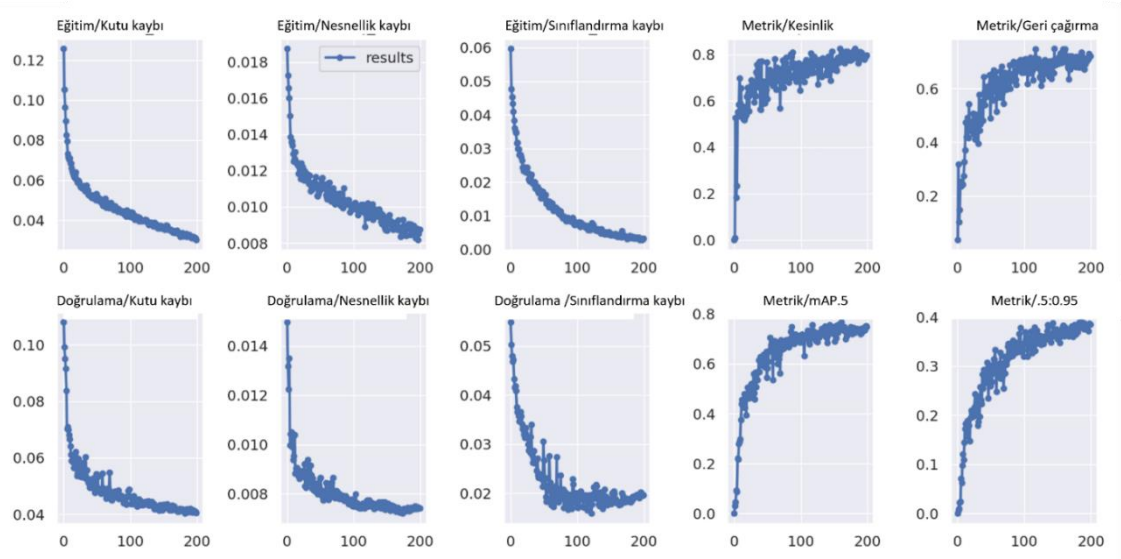


Şekil 5.31. YOLOv5s mimarisinin eğitim ve doğrulama sonuçları

YOLOv5s alt mimarisi ile gerçekleştirilen eğitim sonucunda 0,7555 mAP.5 değeri elde edilmiştir. Buna ek olarak, model 0,744 kesinlik ve 0,742 duyarlılık oranlarına ulaşmıştır. Eğitimle elde edilen ağırlıklar kullanılarak ray kusuru içeren görüntüler test edilmiş ve bu testlerin sonuçları Şekil 5.32’de sunulmuştur. Şekil 5.33’te YOLOv5n alt mimarisine ait eğitim ve doğrulama grafikleri görülmektedir.

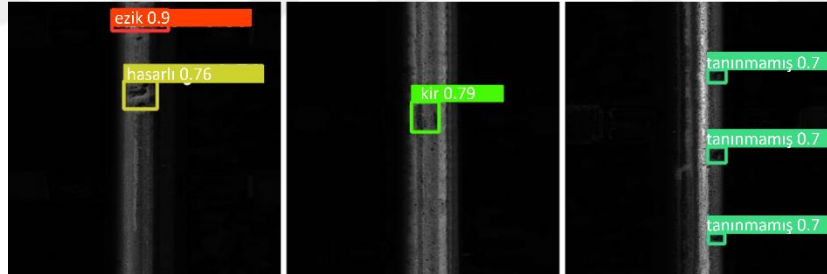


Şekil 5.32. YOLOv5s alt mimarisinin test sonuçları



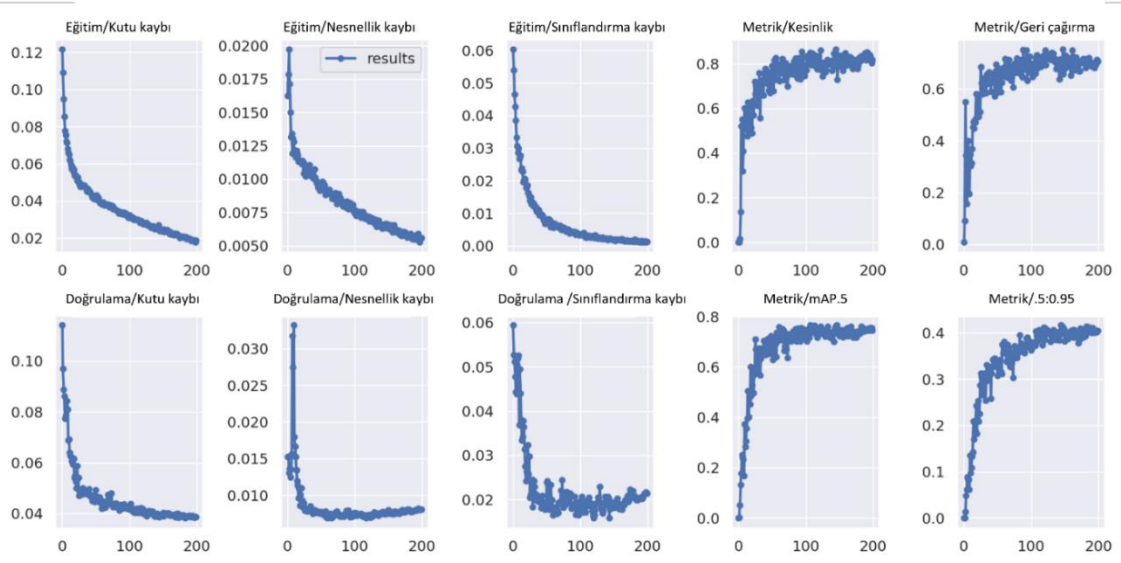
Şekil 5.33. YOLOv5n alt mimarisinin eğitim ve doğrulama sonuçları

YOLOv5n alt mimarisi ile yapılan eğitim sonucunda elde edilen ağırlıklar kullanılarak ray kusuru içeren görüntüler test edilmiş ve bu testler sonucunda Şekil 5.34'teki görseller elde edilmiştir.



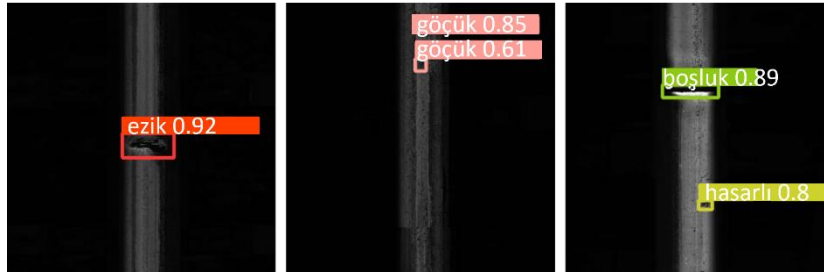
Şekil 5.34. YOLOv5n alt mimarisinin test sonuçları

YOLOv5n alt mimarisi ile gerçekleştirilen eğitim sonucunda 0,755 mAP.5 değeri elde edilmiştir. Bunun yanında, model 0,794 kesinlik ve 0,704 duyarlılık oranlarına ulaşmıştır. Şekil 5.35'te ise YOLOv5m alt mimarisine ait eğitim ve doğrulama grafiklerine yer verilmiştir.



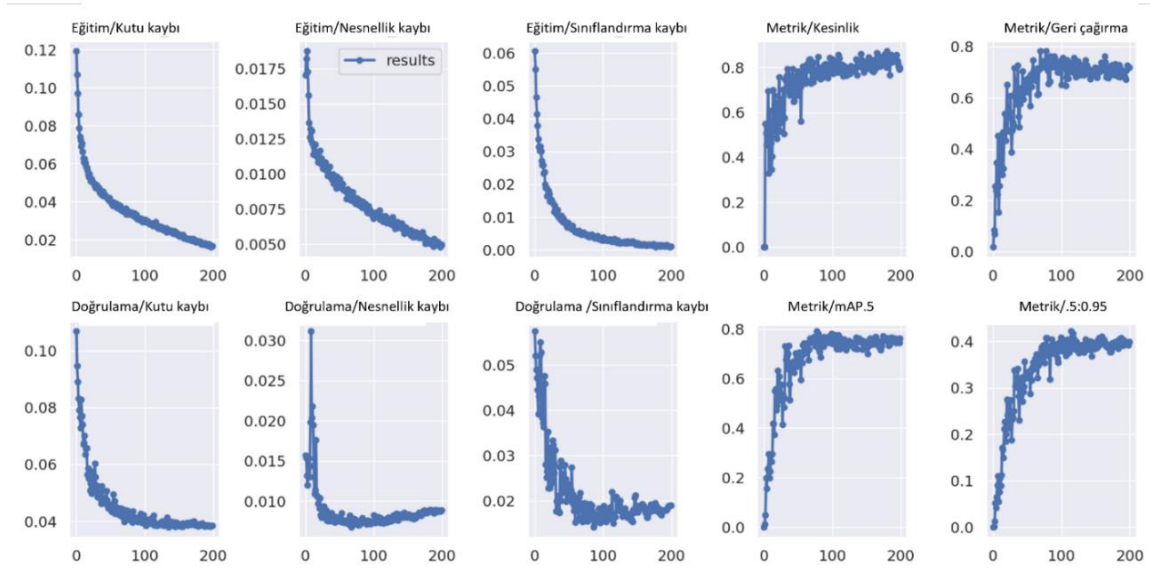
Şekil 5.35. YOLOv5m mimarisinin eğitim ve doğrulama sonuçları

YOLOv5m modelinin farklı bir alt mimarisi üzerinde gerçekleştirilen eğitim sonucunda 0,752 mAP.5 değerine ulaşılmıştır. Ayrıca, modelin performans ölçümlerinde kesinlik oranı 0,812 ve duyarlılık oranı 0,68 olarak kaydedilmiştir. YOLOv5m mimarisine dayalı bu eğitimin sonucunda elde edilen ağırlıklar, ray kusurlarını içeren görüntüler üzerinde test edilmiş ve sonuçlar Şekil 5.36’da görsel olarak sunulmuştur.



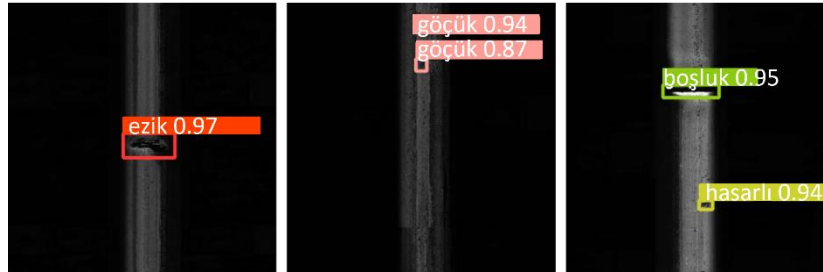
Şekil 5.36. YOLOv5m alt mimarisinin test sonuçları

Şekil 5.37’de YOLOv5l alt mimarisine ait eğitim ve doğrulama grafikleri görülmektedir.



Şekil 5.37. YOLOv5l mimarisinin eğitim ve doğrulama sonuçları

YOLOv5l modelinin alt mimarisiyle yapılan eğitim sonucunda, 0,759 mAP.5 değeri elde edilmiştir. Modelin doğrulama performansı, 0,813 kesinlik ve 0,723 duyarlılık oranlarıyla ölçülmüştür. Eğitim sürecinde elde edilen ağırlıklar, ray kusurlarını içeren test görüntüleri üzerinde uygulandığında, Şekil 5.38’deki sonuçlar görsel olarak elde edilmiştir.



Şekil 5.38. YOLOv5l alt mimarisinin test sonuçları

Tablo 5.22’de YOLOv4 ve YOLOv5 modellerinin eğitimi sonucu sınıf bazında elde edilen mAP.5 değerleri görülmektedir.

Tablo 5.23, farklı YOLOv5 alt mimarilerinin doğrulama performanslarını ve topluluk modelleme yönteminin bu performans üzerindeki etkilerini göstermektedir. Topluluk modellemeyen gelen ağırlıklar, mAP.5 değerlerinde bir artış sağlayarak doğrulama performansını iyileştirmiştir.

Tablo 5.22. YOLOv4 ve YOLOv5 mimarilerinin eğitim sonucunda sınıf bazında elde ettiği map.5 değerleri

Kusur Çeşidi	YOLOv4 mAP.5 Değeri	YOLOv5s mAP.5 Değeri	YOLOv5n mAP.5 Değeri	YOLOv5m mAP.5 Değeri	YOLOv5l mAP.5 Değeri
Ezik	0,930	0,912	0.869	0.969	0.922
Göçük	0,650	0,688	0.797	0.722	0.794
Çizik	0,770	0,832	0.786	0.829	0.800
Eğik	0,680	0,728	0.842	0.681	0.730
Hasarlı	0,380	0,349	0.253	0.325	0.327
Kir	0,600	0,712	0.718	0.712	0.719
Boşluk	1,000	0,995	0.977	0.990	0.979
Tanınmamış	0,830	0,821	0.799	0.786	0.805

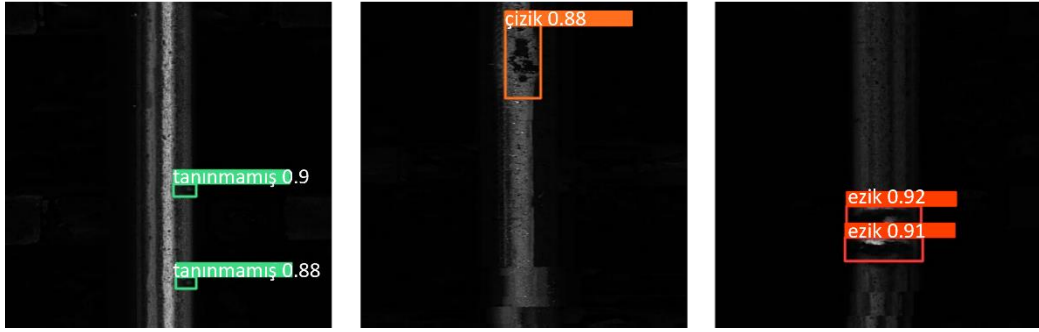
Tablo 5.23. YOLOv5 modelleri ile topluluk modelleme yönteminin doğrulama performansları

Model	P	R	mAP.5
YOLOv5n	0.794	0.704	0.756
YOLOv5s	0.744	0.742	0.753
YOLOv5m	0.812	0.68	0.752
YOLOv5l	0.875	0.716	0.783
YOLOv5n+ YOLOv5s+ YOLOv5m+ YOLOv5l	0.843	0.763	0.809
YOLOv5s+ YOLOv5m+ YOLOv5l	0.849	0.751	0.811

YOLOv6'nın YOLOv6s, YOLOv6n ve YOLOv6t modelleri, veri seti üzerinde eğitilmiş ve artan girdi boyutlarının mAP.5 oranındaki artışla ray kusurlarını daha yüksek doğrulukla tespit ettiği Tablo 5.24'te gösterilmiştir. Şekil 5.39-Şekil 5.44, bu eğitimlerden elde edilen ağırlıkların test sonuçlarına yansıyan görselleri sunmaktadır.

Tablo 5.24. YOLOv6 alt mimarilerinin eğitim sonuçları

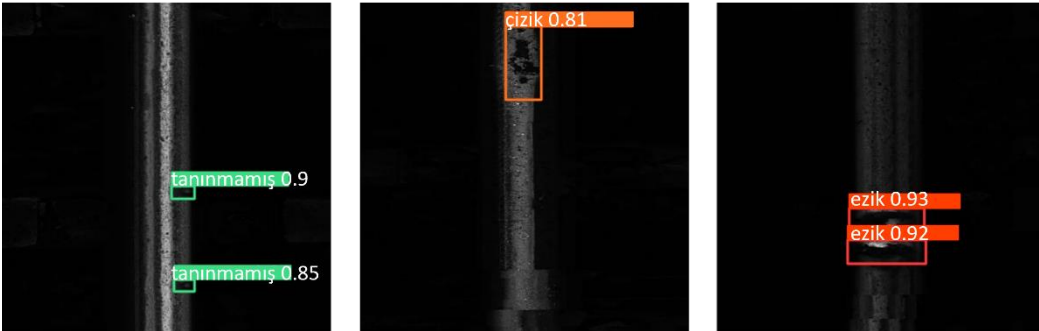
Model	Girdi Boyutu	mAP.5 Değeri
YOLOv6s	416	0.7074
YOLOv6n	416	0.7063
YOLOv6t	416	0.6858
YOLOv6s	640	0.7802
YOLOv6n	640	0.7473
YOLOv6t	640	0.7792



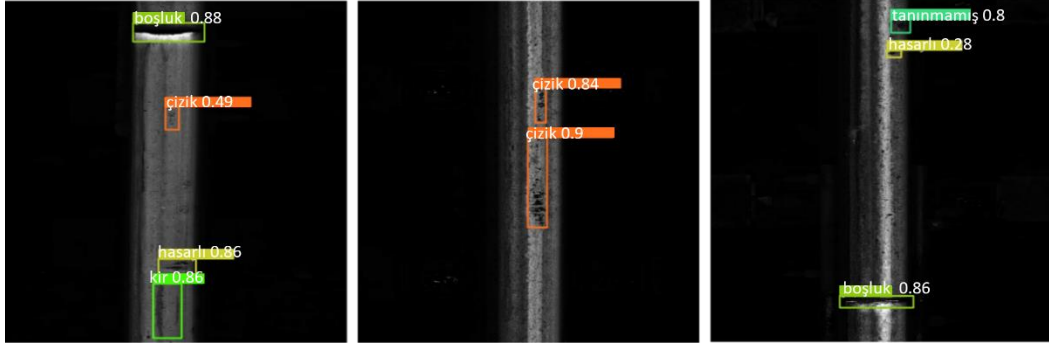
Şekil 5.39. 416x416 girdi boyutlu YOLOv6s alt mimarisinin test sonuçları



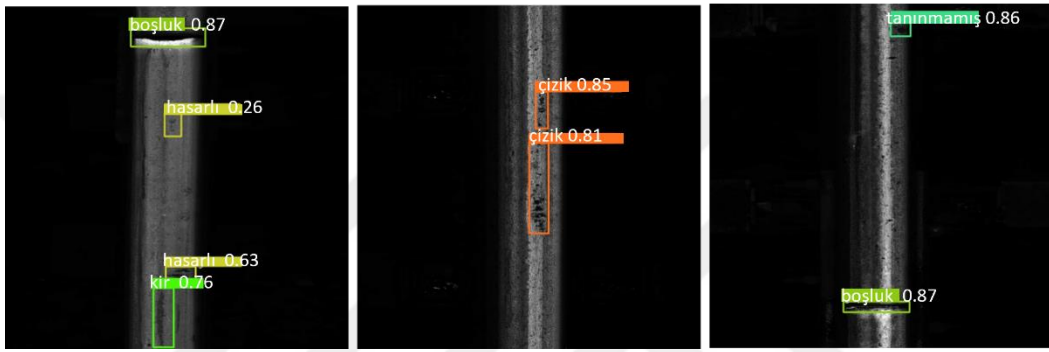
Şekil 5.40. 416x416 girdi boyutlu YOLOv6n alt mimarisinin test sonuçları



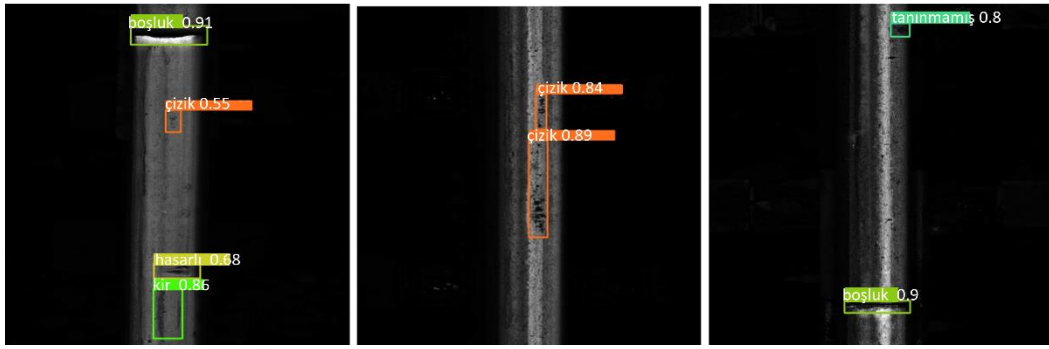
Şekil 5.41. 416x416 girdi boyutlu YOLOv6t alt mimarisinin test sonuçları



Şekil 5.42. 640x640 girdi boyutlu YOLOv6s alt mimarisinin test sonuçları



Şekil 5.43. 640x640 girdi boyutlu YOLOv6n alt mimarisinin test sonuçları



Şekil 5.44. 640x640 girdi boyutlu YOLOv6t alt mimarisinin test sonuçları

YOLOv4 modelinin eğitimi 0,7338 mAP.5 oranına ulaşırken, YOLOv5'in YOLOv5n, YOLOv5s, YOLOv5m ve YOLOv5l alt modelleri sırasıyla 0,7555, 0,755, 0,752 ve 0,752 mAP.5 değerlerine ulaşmıştır. Öte yandan, YOLOv6'nın YOLOv6s modeli, girdi veri boyutunun artmasıyla Tablo 5.24'te de gösterildiği üzere 0,7802 mAP.5 oranına ulaşmıştır. Çalışma sonuçlarına göre, YOLOv5 ve YOLOv6 modelleri, YOLOv11 ile kıyaslandığında daha yüksek doğrulama başarılarına ulaşmıştır. Modellerin doğrulama performansları, 90 test görüntüsü kullanılarak Tablo 5.25'te sunulmuştur.

Tablo 5.25. YOLOv5 modelleri ile topluluk modelleme yönteminin doğrulama performansları

Model	Görüntü Sayısı	Girdi Boyutu	mAP.5
YOLOv5n	939	416	0.756
YOLOv5s	939	416	0.753
YOLOv5m	939	416	0.752
YOLOv5l	939	416	0.783
YOLOv5n+ YOLOv5s+ YOLOv5m+ YOLOv5l	939	416	0.809
YOLOv5s+ YOLOv5m+ YOLOv5l	939	416	0.811
YOLOv6s	939	416	0.734
YOLOv6n	939	416	0.702
YOLOv6t	939	416	0.704
YOLOv6s	939	640	0.770
YOLOv6n	939	640	0.738
YOLOv6t	939	640	0.785

Tablo 5.25'teki verilere göre, topluluk modelleme YOLOv5 alt mimarilerinin doğrulama performansını artırırken, YOLOv6 modelinde girdi veri boyutunun artırılması doğrulama performansını olumlu etkilemiştir. Bununla birlikte, bu değişiklik eğitim süresini 1,8 saatten 3,2 saate çıkarmıştır. Ayrıca, demiryolu alanında yapılan çalışmalar arasında, YOLOv5 topluluk modelleme ve YOLOv6 yöntemlerinin nadiren kullanıldığı gözlenmiştir. Tablo 5.26'da görüldüğü üzere, önerilen topluluk modelleme yöntemi, literatürdeki bazı yöntemlerle karşılaştırıldığında daha yüksek kesinlik ve duyarlılık oranları sunmaktadır.

Tablo 5.26. YOLOv5 topluluk modelleme yöntemi ile literatürdeki yaklaşımların performans karşılaştırması

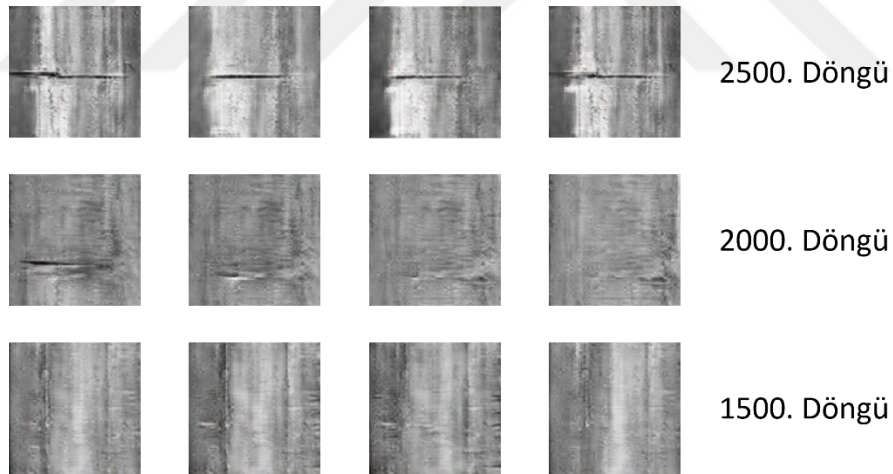
Referans	Yaklaşım	Görüntü Sayısı	P	R
[20]	BSM	195	0.278	0.726
[20]	CTFM	195	0.841	0.732
[20]	REM	195	0.039	0.472
[134]	YOLOv4	278	0.880	0.666
[134]	ASFF-YOLOv4	278	0.725	0.439
Önerilen Metot-I	YOLOv5n+ YOLOv5s+ YOLOv5m+ YOLOv5l	939	0.843	0.763
Önerilen Metot-II	YOLOv5s+ YOLOv5m+ YOLOv5l	939	0.849	0.751

Bu çalışmada, demiryolu rayları üzerindeki kusurların tespitine yönelik olarak YOLOv4, YOLOv5 ve YOLOv6 tabanlı bir model geliştirilmiştir. Demiryolu hattından toplanan ve 8 farklı

kusur türü içeren, toplamda 399 görüntüden oluşan veri seti üzerinde görüntü artırma teknikleri uygulanmıştır. Bu teknikler, görüntülerin çevirme, parlaklık ayarlama ve gürültü ekleme işlemlerini kapsamaktadır. Özellikle gürültü ekleme işlemi, modelin performansını daha zorlu koşullarda değerlendirme imkânı sağlamıştır. Eğitim aşamasında 810 adet kusur içeren görüntü kullanılarak model eğitimi tamamlanmış; doğrulama için 90, test için ise 39 görüntü ayrılmıştır. Veri seti dağılımı oranları ise sırasıyla, eğitim için %86, doğrulama için %10 ve test için %4 olarak belirlenmiştir. Topluluk modelleme yöntemi ile 8 farklı ray kusurunu tespit eden YOLOv5 modeli, %81 mAP.5, %84 kesinlik ve %75 duyarlılık oranlarına ulaşmıştır. Bu ölçüm sonuçları, topluluk modelleme yaklaşımı ile girdi verisi boyutunun artışının, raylardaki kusur tespit başarımını olumlu yönde etkilediğini ortaya koymaktadır.

5.8. Üretken Çekişmeli Ağlar ve Topluluk Yöntemine Dayalı Ray Üzerindeki Kusurların Sınıflandırması Metodunda Elde Edilen Bulgular

Uygulamanın eğitim aşaması tüm modellerde Google Colab aracıyla yapılmıştır. Deney ortamı Tesla P100-PCIE-16GB GPU ve Intel(R) Xeon(R) CPU @ 2.20GHz CPU sisteminden oluşmaktadır. GAN sonucu üretilen bazı ray görüntüleri Şekil 5.45'te görülmektedir.



Şekil 5.45. GAN ile üretilen kusurlu ray görüntüleri

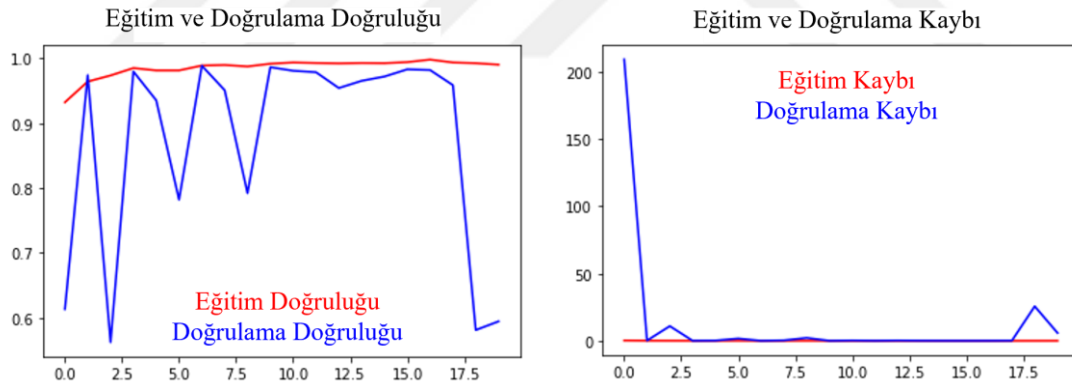
GAN ile üretilen sağlıklı, kırık ve delik ray görüntüleri eğitim için kullanılmıştır. Buradaki amaç, eğitim için daha fazla kusurlu ray örneği kullanarak doğruluğu artırmaktır. GAN ile üretilen görüntülerin kalitesini değerlendirmek için VTT, NBD ve FID metriklerine başvurulmuştur. İnsan gözüyle (VTT) üretilen örnekleri değerlendirdiğimizde çeşitlilik ve kalite bakımından üretilen görüntüler tatmin edici seviyede olduğu görülmektedir.

Tablo 5.27. GAN performansı değerlendirme metrikleri

Metrik Sınıf	NBD	FID
Sağlıklı	8	103.750
Çatlak	9	120.504
Delik	2	74.359

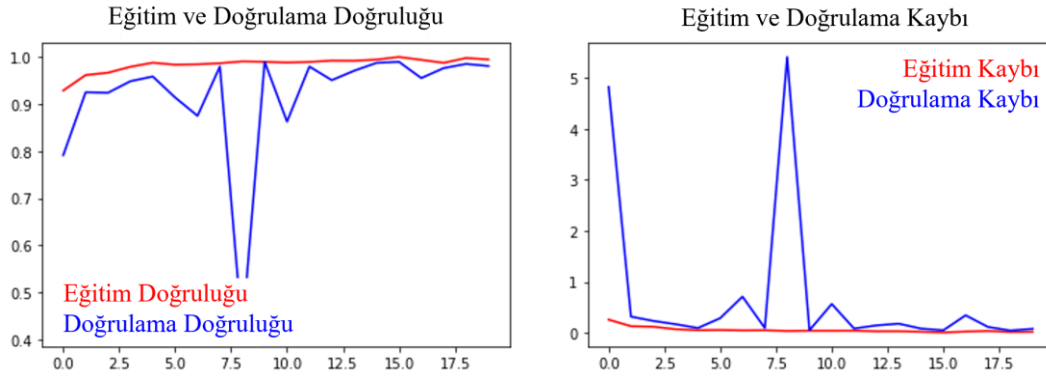
Tablo 5.27’de GAN mimarisinin sınıf bazında performansı görülmektedir. NBD ve FID metrikleri 0’a ne kadar yakınsa GAN modeli o kadar başarılı demektir. Yüzeyinde delik içeren ray görüntüsü oluşturma konusunda GAN modeli diğer sınıflara göre çeşitlilik ve kalite açısından daha başarılı olmuştur.

Şekil 5.46’da ResNet50 modelinin doğruluk ve kayıp grafikleri görülmektedir. ResNet50 modeli doğrulama sırasında 0.9889 doğruluk oranı yakalamıştır. Eğitim sırasında her doğrulama aşamasından sonra doğruluk oranını takip ettiğimiz bir kontrol noktası parametresi tanımlanmıştır. Bunu kullanarak, eğitim sürecinde en iyi performansı gösteren modelin bir kopyası topluluk tahmini için kayıt altına alınmıştır.



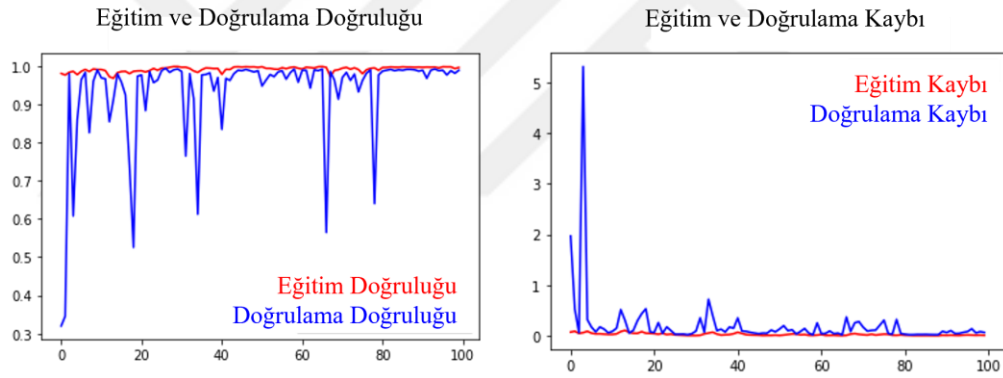
Şekil 5.46. ResNet50 doğruluk ve kayıp grafiği

Şekil 5.47’de InceptionV3 modelinin doğruluk ve kayıp grafikleri görülmektedir. InceptionV3 modeli de doğrulama sırasında 0.9889 doğruluk oranı yakalamıştır. Eğitim sürecinde modelin gösterdiği en iyi performans topluluk tahmini için kayıt altına alınmıştır.



Şekil 5.47. InceptionV3 doğruluk ve kayıp grafiği

Şekil 5.48’de topluluk modelinin doğruluk ve kayıp grafiği görülmektedir. InceptionV3 modeli de doğrulama adımında 0.9944’lük doğruluk oranı yakalamıştır. Topluluk modeli bireysel olarak tahmin yapan modellerin performanslarını geçmiştir.



Şekil 5.48. Topluluk modeli doğruluk ve kayıp grafiği

Tablo 5.28’de topluluk modelinin performans metrikleri görülmektedir.

Tablo 5.28. Topluluk modeli performansı değerlendirme metrikleri

Sınıf	Kesinlik	Duyarlılık	F1-Skoru
Çatlak	1.00	0.97	0.98
Sağlıklı	0.98	1.00	0.99
Delik	0.99	1.00	0.99

Şekil 5.49’da üretilen topluluk modelinin performansının değerlendirilmesi için karışıklık matrisi görselleştirilmiştir.

Tahmin Edilen Etiket	Çatlak	278	5	4
	Sağlıklı	0	300	0
	Delik	1	0	312
		Çatlak	Sağlıklı	Delik
		Doğru Etiket		

Şekil 5.49. Topluluk modeli karışıklık matrisi

Tren raylarındaki kusurların önceden tespiti hem insan güvenliği hem de ekonomik performans için çok önemli bir konudur. Tren raylarındaki kusurlu tespit etmek en büyük zorluklardan biridir. Bu çalışmada, bu sorunu çözmek için CNN tabanlı yeni bir derin ağ modeli önerilmiştir. Bu çalışma, ray yüzeyi kusurlarının tespiti ve sınıflandırılması ile ilgilidir. Önerilen yöntem, görüntü çoğaltma, derin öğrenme modellerinden ve topluluk kararına dayalı sınıflandırmadan oluşmaktadır. Tablo 5.29’da görüldüğü gibi önerilen yaklaşım, literatürde mevcut bildirilen yöntemlerden daha iyi olan %99,44'lük bir sınıflandırma başarısı elde etmiştir.

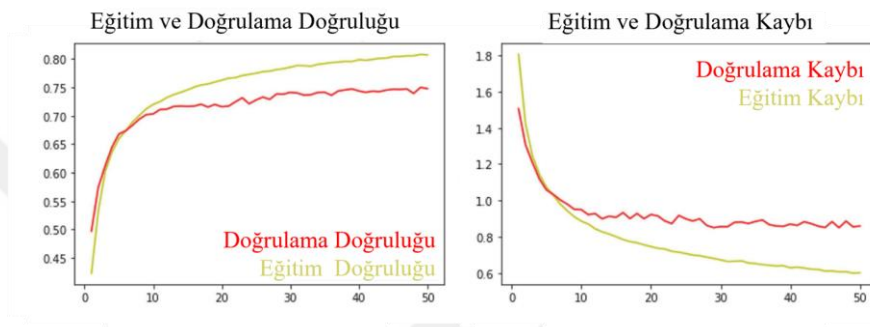
Tablo 5.29. Önerilen topluluk modelinin literatürdeki çalışmalarla karşılaştırılması

Referans	Model	Doğruluk Oranı
[19]	CNN	%91.19
[135]	ResNet	%95.94
[135]	VGGNet	%92.91
[6]	DCNN-Large	%92.47
[136]	CNN	%97.00
Önerilen Model	Topluluk Modeli	%99.44

Derin öğrenmeye dayalı kusur tespit yöntemleri, sağlıklı rayların tespitine daha fazla odaklanır ve kusurlu rayların tespitine daha az uygundur. Bunun temel nedeni, sahada toplanan verilerin çoğunun sağlıklı ray görüntülerini içermesi ve kusurlu ray görüntüsü sayısının çok sınırlı olmasıdır. Bu nedenle, çeşitli ve kaliteli kusur içeren ray görüntülerini üretmek için GAN tekniği kullanarak eğitim verilerinin dengesi iyileştirilmiştir.

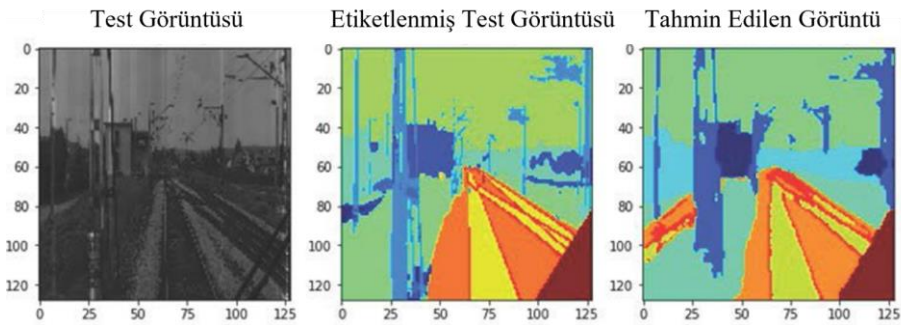
5.9. Demiryolu Hatlarını İçeren Görüntülerde Bir Topluluk Modeli Kullanarak Unet Bölütleme Performansını İyileştirme Metodunda Elde Edilen Bulgular

Çalışmada önerilen modeller, 11. Nesil Intel(R) Core(TM) i5-1135G7 @ 2.40GHz 2.42 GHz ve NVIDIA GeForce MX450 grafik kartı ile deneysel bir ortamda gerçekleştirilmiştir. Unet bölütleme işlemi eğitim parametreleri: döngü sayısı 120, batch boyutu 16 olarak ayarlanmıştır. 50. döngüden sonra performans metriklerinde herhangi bir ilerleme görülmediği için döngü sayısı 50 olarak sabitlenmiştir. Performans ve kayıp grafikleri Unet modelinin eğitim ve doğrulama aşamaları Şekil 5.50'de gösterilmiştir.



Şekil 5.50. Unet modelinin doğrulama ve eğitim grafikleri

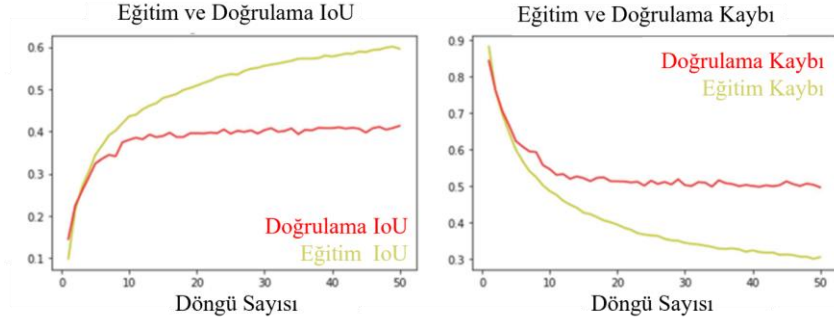
Şekil 5.50'de görüldüğü gibi, doğrulama işlemi sonucunda 0,7478 doğruluk oranı ve ortalama 0,3625 IoU değeri elde edildi. Eğitilmiş Unet modeli kullanılarak veri seti üzerinde tahmin yapılmıştır. İşlemin sonucu Şekil 5.51'de gösterilmektedir. Şekil 5.51 sırasıyla test görüntüsünü, görüntünün referans görüntüsünü ve tahmin edilen görüntüyü göstermektedir.



Şekil 5.51. Unet modelinin tahmini

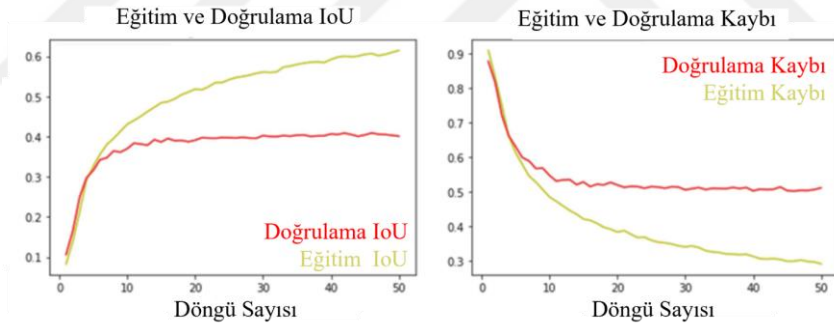
Elde edilen ortalama IoU ve doğrulama değerlerini artırmak için Unet mimarisi kullanılarak omurgaların eğitilmesi sonucunda topluluk karar verme sürecini temel alan bir model önerilmiştir.

ResNet34, MobileNetV2 ve VGG16 omurga olarak kullanıldı. MobileNetV2 omurgası, eğitim sonucunda Şekil 5.52'deki değerleri elde etmiştir.



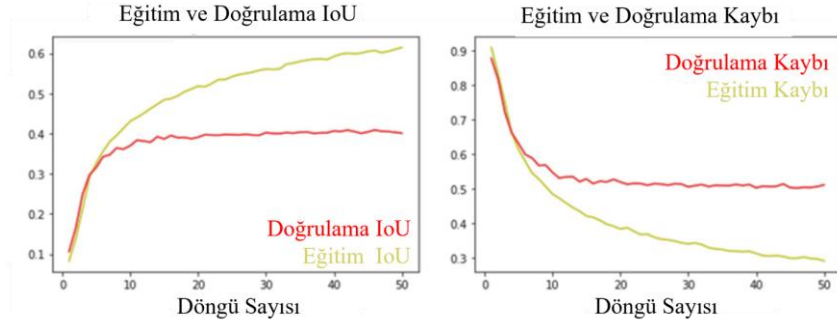
Şekil 5.52. Kullanılan MobileNetV2 omurgasının eğitim ve doğrulama sonuçları

Doğrulama sonucunda MobileNetV2 omurgası ortalama 0,4131 IoU değerine ulaştı. MobileNetV2 omurgası, eğitim süreci sonucunda ortalama 0,5957 IoU değeri elde etti. ResNet34 omurgası, eğitim sonucunda Şekil 5.53'teki eğitim ve doğrulama değerlerini elde etmiştir.



Şekil 5.53. Kullanılan ResNet34 omurgasının eğitim ve doğrulama sonuçları

Doğrulama sonucunda ResNet34 omurgası ortalama 0,4009 IoU değerine ulaştı. ResNet34 omurgası, eğitim süreci sonucunda ortalama 0,6145 IoU değeri elde etti. Son olarak, Unet modeli VGG16 omurga veri seti üzerinde eğitilmiştir. Şekil 5.54'teki eğitim ve doğrulama değerleri elde edilmiştir.



Şekil 5.54. Kullanılan VGG16 omurgasının eğitim ve doğrulama sonuçları

Doğrulama sonucunda VGG16 omurgası ortalama 0,6117 IoU değeri elde etti. VGG16 omurgası, eğitim süreci sonucunda ortalama 0,4056 IoU değeri elde etti. Eğitim IoU ve kayıp grafikleri Şekil 5.52, 5.53 ve 5.54'te gösterilmektedir. Şekil 5.52, Şekil 5.53 ve Şekil 5.54'te olduğu gibi, düşük kayıplı yüksek IoU, veri setinde düşük hata anlamına gelmektedir. Kayıp azalmaya başladığında, Unet omurgalarının kayıp ve IoU grafiğinde IoU artmaya başlamıştır. Bu, modelin öğrendiği ve iyi çalıştığı anlamına gelmektedir. Tablo 5.30'da görüldüğü gibi, önerilen hibrit toplam kayıp fonksiyonu, Unet omurgalarının doğrulama performansına olumlu katkıda bulunmuştur. Test süreci, eğitilen veri setinin performansını değerlendirmek için 3 Unet omurgası ile gerçekleştirilmiştir. Eğitim sonucunda elde edilen ağırlıklar kullanılarak Tablo 5.31'deki test sonuçları elde edilmiştir.

Tablo 5.31'de görüldüğü gibi anlamsal bölütleme ağının omurgası olarak MobileNetV2, ResNet34 ve VGG16 kullanıldığında klasik Unet mimarisine göre daha iyi sonuçlar elde edilmektedir. Omurga olarak seçilen modeller, toplu karar verme sürecine girdiklerinde tek başlarına elde ettikleri ortalama IoU değerlerini aşıtlar. Tablo 5.31'de görülebilir.

Tablo 5.31'de topluluk kararı olarak elde edilen IoU değeri, aynı ağırlığa sahip 3 modelin karar verme sürecini etkileyerek elde edilmiştir. Topluluk kararının performansını en üst düzeye çıkarmak için antrenman sonucunda elde edilen ağırlıklar bir katsayı ile çarpılarak topluluk kararına etki edilmesi amaçlanmaktadır. Eğitilmiş omurgalar ve bunların test performansına ilişkin katsayıları Tablo 5.31'de gösterilmektedir.

Tablo 5.30. Hibrit kayıp fonksiyonunun eğitim performansına etkisi

Unet Omurga	Focal Kayıp / Ortalama IoU	Dice Kayıp / Ortalama IoU	Toplam Kayıp / Ortalama IoU
ResNet34	0.5483	0.5988	0.6145
MobileNetV2	0.5231	0.5875	0.5957
VGG16	0.5713	0.6049	0.6117

Tablo 5.31. Topluluk model ve farklı omurgalara sahip Unet modellerinin ortalama IoU değerleri

Model	Ortalama IoU (Test)
Unet	0.3625
Unet (ResNet34)	0.5289
Unet (MobileNetV2)	0.4685
Unet (VGG16)	0.4106
Topluluk Modeli (Eşit Katsayı)	0.5308

Tablo 5.32, farklı katsayılara sahip Unet omurgalarının topluluk kararı üzerindeki etkisini göstermektedir. ResNet34 omurgası da MobileNetV2'den sonra en iyi performansı gösterdi. Bu performanslar, topluluk karar verme sürecini doğrudan etkiledi. Topluluk kararının en iyi performansı $0,3x$ MobileNetV2 + $0,2x$ ResNet34 + $0,0x$ VGG16 katsayıları ile elde edilmiştir. Bu katsayılara sahip topluluk modeli, 0,5430 ortalama IoU değeri elde eder. 0,5'ten büyük ortalama IoU değeri elde eden bir modelin iyi bir performansa sahip olduğu kabul edilir [137]. Unet omurgaları için belirlenen en iyi katsayılar iç içe döngülerle yapılan deneylerde bulunmuştur.

Tablo 5.32. Farklı katsayılara sahip topluluk modeli performansları

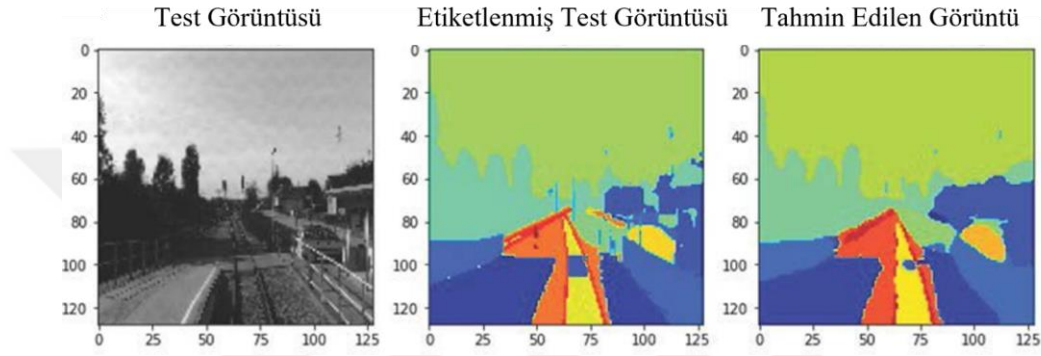
Omurga ve Katsayı	Ortalama IoU (Test)
$0.3x$ MobileNetV2 + $0.2x$ ResNet34 + $0.1x$ VGG16	0.5348
$0.3x$ MobileNetV2 + $0.2x$ ResNet34 + $0.2x$ VGG16	0.5331
$0.3x$ MobileNetV2 + $0.2x$ ResNet34 + $0.1x$ VGG16	0.5354
$0.3x$ MobileNetV2 + $0.2x$ ResNet34 + $0.0x$ VGG16	0.5430

Şekil 5.55'te görüldüğü gibi topluluk karar modeli, tahmin sonucu olarak referans görüntüye benzer bir görüntü üretmiştir. Toplu karar alma, ortalama IoU metriğini ve ayrıca doğruluğu iyileştirmiştir. Tablo 5.33'te görüldüğü gibi topluluk karar modeli, tek başına tahmin yapan geleneksel Unet modelinden daha iyi bir doğruluk oranına ulaşmıştır.

Tablo 5.33. Topluluk modeli (eşit katsayı) ve farklı omurgalara sahip Unet modellerinin doğruluk oranı

Model	Doğruluk Oranı (Test)
Topluluk Modeli (Eşit Katsayı)	0.8575
Unet (MobileNetV2)	0.8423
Unet (ResNet34)	0.8466
Unet (VGG16)	0.8269

Tablo 5.34, karar verme sürecine dahil edilen farklı omurgalara sahip Unet modellerinin katsayı oranları ile performanslarını göstermektedir. Test oranını maksimize eden katsayılar belirlenmiştir. Tablo 5.32 ve Tablo 5.34’te görülebileceği gibi, önerilen topluluk modeli ortalama 0,5430 IoU ve 0,8589 doğruluk oranına ulaşmıştır. Ortalama IoU ve doğruluk, anlamsal bölütleme için standart ölçümlerdir. Doğruluk oranı metriği, görüntünün büyük bölümünü kaplayan arka plan gibi ayırt edici sınıflarda çok başarılı değildir. Bu nedenle, anlamsal bölütleme performansını değerlendirmek için ortalama IoU metriği daha önemlidir.



Şekil 5.55. Test sonucunda topluluk kararı ile elde edilen görüntü

Tablo 5.34. Farklı katsayılara sahip topluluk modelinin ve farklı omurgalara sahip Unet modellerinin doğruluk oranı değerleri

Omurga ve Katsayı	Doğruluk Oranı (Test)
0.3x MobileNetV2 + 0.3xResNet34 + 0.2xVGG16	0.8589
0.3x MobileNetV2 + 0.3xResNet34 + 0.1xVGG16	0.8586
0.3x MobileNetV2 + 0.2xResNet34 + 0.2xVGG16	0.8572
0.3x MobileNetV2 + 0.1xResNet34 + 0.0xVGG16	0.8459

Bu uygulama, RailSem19 veri setini etkin bir şekilde bölümlere ayırmak için bir metodoloji sunmuştur. Çalışmada 8500 görsel incelendi. Literatürdeki diğer çalışmalarda RailSem19, veri kümesinin görüntülerini segmentlere ayırmak için derin öğrenme metodolojileri uygulamıştır, ancak bunların hiçbiri Unet omurgaları MobileNetV2, ResNet34 ve VGG16'nın bölütleme modellerini birlikte kullanmaya odaklanmamıştır. Bu da yapılan çalışmanın demiryolu sektörüne katkısını göstermektedir. Tablo 5.35, çalışmada önerilen modelin performansının literatürdeki diğer çalışmalarla karşılaştırıldığını göstermektedir.

Tablo 5.35. Önerilen modelin literatürdeki çalışmalarla karşılaştırılması

Referans	Model	Ortalama IoU (Test)	Sınıf Sayısı
Önerilen Model	0.3x MobileNetV2 + 0.2xResNet34 + 0.0xVGG16	0.5430	29
[138]	Unet	0.2960	19
[139]	SegNet3	0.5364	4
[140]	UNet16	0.4258	2

Tablo 5.35’te görüldüğü gibi, önerilen topluluk karar modeli, ortalama IoU metriğine dayalı olarak literatürdeki diğer çalışmalarla karşılaştırıldı. Önerilen model, literatürdeki diğer çalışmalara kıyasla daha fazla sınıfı daha başarılı bölütlemiştir. Bahsedilen literatür çalışmaları, demiryolu güvenliğinin çeşitli yönlerine ve semantik bölütleme ve derin öğrenme tekniklerini kullanan analizlere odaklanmaktadır. Bu çalışmalar, malzeme tanımlama, anormallik tespiti, kusur tespiti ve demiryolu altyapısının haritalanması gibi alanları kapsayan demiryolu güvenliği ve analizinde anlamsal bölütleme önemi toplu olarak vurgulamaktadır. Literatür çalışmaları ile karşılaştırıldığında, çalışmamız Unet omurgaları MobileNetV2, ResNet34 ve VGG16’nın bölütleme modellerini birlikte kullanarak RailSem19 veri setini etkin bir şekilde segmentlere ayırmaya odaklanmaktadır. Literatürdeki önceki çalışmaların hiçbiri bu kombinasyonu kullanmadığından, bu yaklaşım demiryolu sektörüne yeni bir katkıdır. Önerilen topluluk modelimiz ortalama %54,30 IoU değerine ulaştı ve diğer çalışmalara kıyasla daha fazla sınıfı segmentlere ayırdı. Genel olarak çalışmamız derin öğrenme tabanlı görüntü bölütleme algoritmaları kullanarak demiryollarındaki arızaların ve izinsiz girişlerin tespit edilmesinin önemini vurgulayarak demiryolu güvenliği alanına katkıda bulunur. Birincisi, çalışma Türkçe ve İngilizce makalelerle sınırlıdır. Bu, ilgili bazı çalışmaların hariç tutulmasına yol açmıştır. Son olarak, güçlü bir test ortamı bulunmadığından geliştirme süreci daha uzun sürmüştür. Bu çalışmada demiryollarında arıza tespitinin demiryolu güvenliği açısından önemi vurgulanmıştır. Demiryolu ulaşımının kullanımı her geçen gün artacağı için ortaya çıkan sorunlar da gün geçtikçe artacaktır.

5.10. YOLOv8 ile Demiryolu Hattı Çevresindeki Yabancı Nesnelerin Tespiti Metodunda Elde Edilen Bulgular

Eğitim aşaması tüm modellerde Google Colab aracıyla yapılmıştır. Deney ortamı Tesla P100- PCIE-16GB grafik işlemci birimi (GPU) ve Intel(R) Xeon(R) merkezi işlem birimi (CPU) @ 2.20GHz CPU sisteminden oluşmaktadır. Veri boyutu olarak 640x640 çözünürlüğü tercih edilmiş döngü sayısı olarak 25 ayarlanmıştır. Optimize edici olarak SGD, öğrenme oranı 0.01 olarak ayarlanmıştır. Eğitim sonuçları Tablo 5.36’da görülmektedir.

Tablo 5.36. YOLOv8 eğitim sonuçları

Alt Mimari	mAP50	mAP50 (Araç)	mAP50 (İnsan)	Eğitim Süresi (Saat)
YOLOv8n	0.837	0.890	0.784	0.248
YOLOv8s	0.848	0.908	0.788	0.251
YOLOv8m	0.888	0.939	0.837	0.341
YOLOv8l	0.859	0.922	0.797	0.441
YOLOv8x	0.862	0.919	0.805	0.671

Tablo 5.36'daki eğitim sonuçlarına göre YOLOv8m ortalama ve sınıf bazında mAP50 değerlerinde diğer YOLOv8 alt mimarilerine göre daha başarılı olmuştur. YOLOv8m eğitim süresi bakımından YOLOv8n ve YOLOv8s alt mimarilerine göre daha yavaş kalmıştır. Ama başarı oranı olarak daha iyi sonuçlar elde etmiştir. Tablo 5.37'de doğrulama sonuçları görülmektedir.

Tablo 5.37. YOLOv8 doğrulama sonuçları

Alt Mimari	mAP50	mAP50 (Araç)	mAP50 (İnsan)
YOLOv8n	0.843	0.895	0.791
YOLOv8s	0.865	0.919	0.811
YOLOv8m	0.888	0.943	0.833
YOLOv8l	0.859	0.921	0.797
YOLOv8x	0.871	0.915	0.827

Tablo 5.37'deki doğrulama bakıldığında eğitim aşamasındaki gibi YOLOv8m alt mimarisi diğer alt mimarilere göre ortalama ve sınıf bazında mAP50 değerlerinde daha başarılı olmuştur. YOLOv8m alt mimarisine ait test sonuçları Şekil 5.56'da görülmektedir. Şekil 5.56'da görüldüğü gibi insan ve araç gibi nesneler görüntülerde net bir şekilde ayırt edilmiştir.

**Şekil 5.56. YOLOv8 test sonuçları**

5.11. DNET ve Matematiksel Yöntemler Kullanarak Geliştirilmiş Demiryolu İzleme ve Bölütleme Metodunda Elde Edilen Bulgular

Bu bölümde, BiSeNetV2, YOLOv7 ve YOLOv8 modellerinin demiryolu hattı bölütlemesindeki performansını değerlendirilmiştir. Gazebo simülasyon ortamında YOLO tabanlı modelin performansının FPS (saniye başına kare) sorunları nedeniyle başarısız olduğu kayıt altına alınmıştır. Üç modelinin karşılaştırmaları Tablo 5.38'de verilmiştir.

Tablo 5.38. BiSeNetV2, YOLOv7 ve YOLOv8 sonuçları

Model	Metrik	Değer
BiSeNetV2	Accuracy	0.9746
YOLOv7	mAP 0.5	0.8972
YOLOv8	mAP 0.5	0.9119

Tablo 5.38'de görüldüğü gibi, BiSeNetV2 modelinin eğitimi sonucunda doğruluk metriği açısından 0.97'lik yüksek bir performans elde edildi. YOLOv7 ve YOLOv8 modelleri, eğitim sonrası doğrulama oranlarına dayalı performans değerlendirmelerini içeren Tablo 5.38'de sunulmaktadır. YOLOv7 0.8972'lik mAP 0,5 değerine ve YOLOv8 0.9119'luk değere ulaşmıştır.

Şekil 5.57, Gazebo simülasyonunda ve gerçek saha testlerinde BiSeNetV2 modelinin sonuçlarını görsel olarak sunmaktadır. Model, simülasyon ve gerçek saha ortamlarında 0.99'luk yüksek doğruluk oranlarına ulaşmıştır.



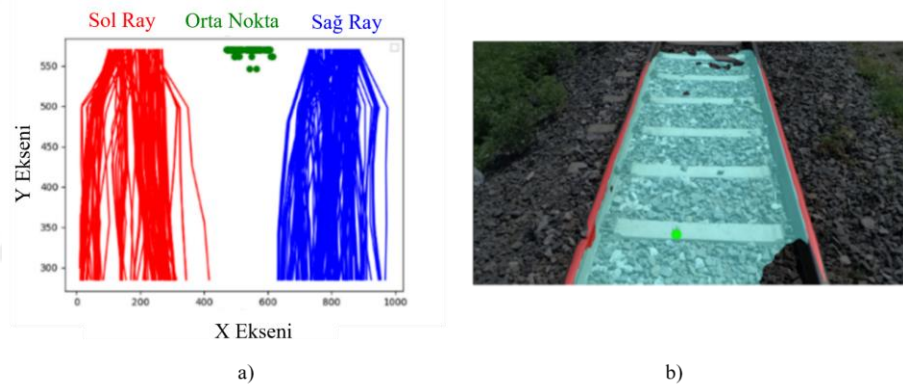
Şekil 5.57. BiSeNetV2 modelinin (a) simülasyon ve (b) gerçek saha ortamlarındaki test sonuçları

Şekil 5.58, her iki modelin gerçek saha ortamlarındaki test sonuçlarını göstermektedir.



Şekil 5.58. (a) YOLOv7 ve (b) YOLOv8 modellerinin gerçek saha ortamındaki test sonuçları

Şekil 5.59, Bezier eğrisi kullanılarak belirlenen sol, sağ ray ve orta nokta grafiklerini göstermektedir. Bu görsel analiz, demiryolu hattındaki önemli konumları temsil ederek modelin başarılı bir şekilde bölütlendirdiğini göstermektedir.

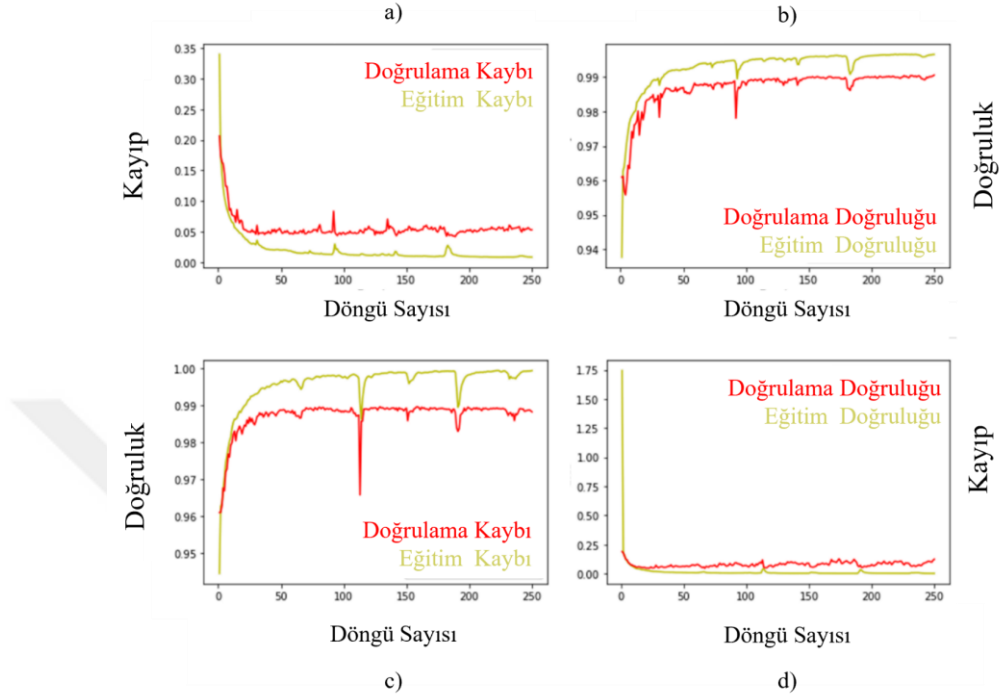


Şekil 5.59. (a) Tespit edilen sol, sağ ray ve orta noktanın Bezier görüntüsü ve (b) saha sonuçları

DNet ve klasik Unet modellerinin eğitim ve doğrulama performansı, kayıp ve doğruluk grafikleriyle Şekil 5.60’da sunulmaktadır. Şekil 5.60’da, modelin eğitim verilerindeki performansı, eğitim sırasında 250 döngü boyunca izlenmiştir. Eğitim kaybı grafiği, modelin eğitim verilerini öğrenme sürecini yansıtır. Grafikte gözlenen aşağı yönlü eğilim, modelin kaybının zamanla azaldığını ve dolayısıyla eğitim verilerini daha iyi öğrenmeye başladığını göstermektedir. Eğitim kaybının 0,05 civarında sabitlenmesi, modelin eğitim verileri üzerinde mükemmel bir performans elde ettiğini göstermektedir.

Benzer şekilde, eğitim doğruluğu grafiği, modelin eğitim verilerini sınıflandırmadaki başarısını göstermektedir. 250 döngü boyunca eğitim doğruluğundaki artış, modelin zaman içinde daha fazla veri örneğini sınıflandırabildiğini göstermektedir. Eğitim doğruluğunun %99 civarında sabitlenmesi, modelin eğitim verileri üzerinde son derece doğru bir performans gösterdiğini işaret etmektedir. Bulgular ışığında, DNet modelinin eğitim verilerini etkili bir şekilde öğrendiği ve

yüksek doğrulukla sınıflandırdığı sonucuna varılabilir. Bu durum, modelin ilgili problem alanında kullanılabilirliğinin ümit verici bir göstergesidir.



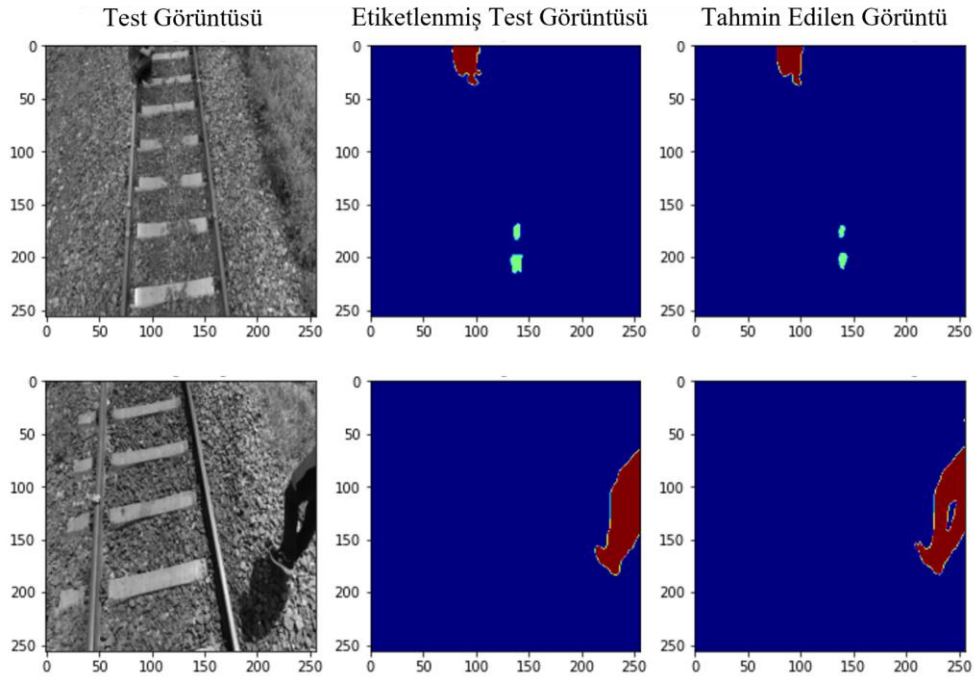
Şekil 5.60. (a) DNet eğitim ve doğrulama kaybı grafiği (b) DNet eğitim ve doğrulama doğruluğu grafiği (c) Klasik Unet eğitim ve doğrulama doğruluğu grafiği (d) Klasik Unet eğitim ve doğrulama kaybı grafiği

Tablo 5.39, klasik Unet modeli ve DNet modelinin eğitim performans ölçümlerini ve değerlerini vermektedir. Tablo 5.39'da görüldüğü gibi, klasik Unet modelinde yapılan iyileştirmeler performansını iyileştirmiştir.

Tablo 5.39. DNet ve Klasik Unet Modellerinin Performans Karşılaştırması

Model	Metrik	Değer
Klasik Unet	Doğruluk oranı	0.9881
DNet	Doğruluk oranı	0.9905
Klasik Unet	Ortalama IoU	0.7219
DNet	Ortalama IoU	0.7432

Şekil 5.61'de DNet modelinin eğitim sonuçları ve oluşturulan veri setinden test edilen görüntüler sunulmaktadır.



Şekil 5.61. Oluşturulan veri seti üzerinde DNet test sonuçları

Şekil 5.61, DNet modelinin demiryolu hattındaki yabancı cisimleri başarılı bir şekilde bölütlediğine ilişkin görsel örnekleri içermektedir. Tablo 5.40, oluşturulan veri setindeki sınıfları bölütlemeye DNet modelinin performansını ayrıntılı olarak göstermektedir.

Tablo 5.40. Oluşturulan Veri Setinin Sınıf Bazında Performans Karşılaştırması

Sınıf	Ortalama IoU
Bitki	0.6707
İnsan	0.8423
Poşet	0.5192
Çubuk	0.5192

Tablo 5.40, DNet modelinin veri setindeki farklı sınıfları başarılı bir şekilde ayırt etme yeteneğini nicel olarak değerlendirilmiştir. Modelin sınıflandırma performansı, bölütleme görevindeki etkinliğine odaklanır ve belirli sınıfların veya diğer performans ölçümlerinin doğruluk oranları aracılığıyla sunulmuştur. Bu analitik yaklaşım, modelin veri setindeki sınıfları başarılı bir şekilde bölütleme yeteneğini nesnel olarak değerlendirmiştir. Veri setinde daha az sayıya sahip sınıfların performans oranı da aynı şekilde düşüktür. Bu düşüklük, ortalama IoU ölçümünü etkilemiştir. Bu nedenle, geliştirilen DNet bölütleme modeli, oluşturduğumuz veri setinin daha

basit bir versiyonunda yeniden test edildi. Yeni veri seti yalnızca insan ve bitki sınıflarından oluşmaktadır.

Tablo 5.41. Oluşturulan yeni veri setinin sınıf tabanlı performans karşılaştırması

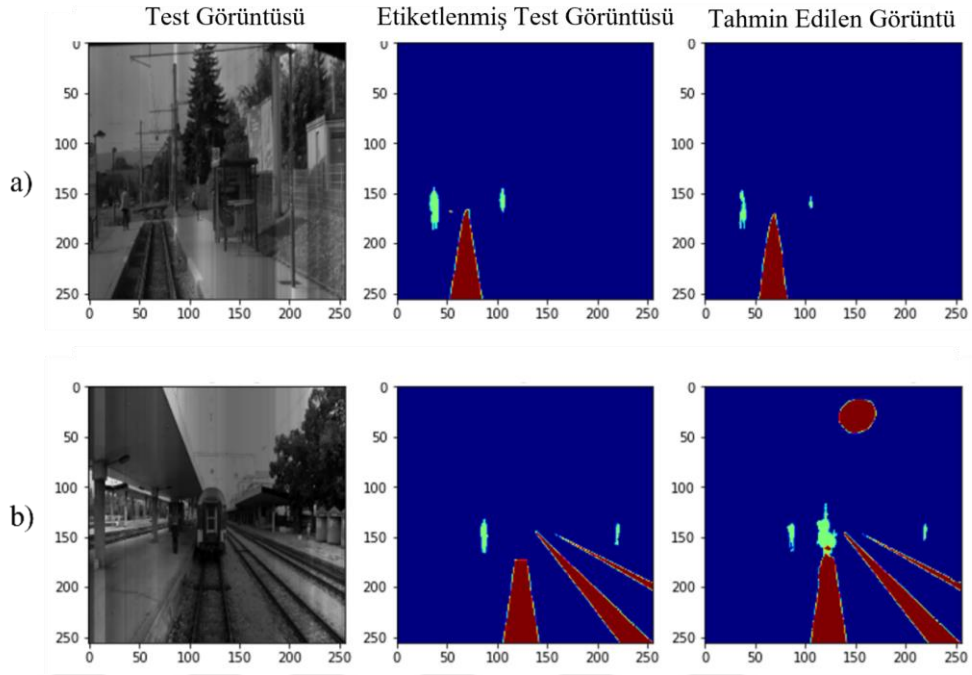
Sınıf	Metrik	Değer
Hepsi	Ortalama IoU	0.8598
Bitki	IoU	0.6818
İnsan	IoU	0.9368

Tablo 5.41'de sunulan DNet modelinin ortalama IoU metriği, veri setindeki basitleştirmeden sonra %74'ten %85'e yükselmiştir. Aynı seviyede, doğrulama oranı da %99,05'ten %99,28'e yükselmiştir. Çalışmanın başında, demiryolu hattını bölütlemek için BiSeNetV2, YOLOv7 ve YOLOv8 modelleri önerilmiştir. Geliştirilen DNet modeli ayrıca demiryolu hattının etrafındaki yabancı nesneleri başarıyla tespit etmiştir. DNet modelinin demiryolu hattını etkili bir şekilde bölütleme yeteneğini doğrulamak için, çalışmanın ilk aşamasında kullanılan özel Railsem19 veri kümesi üzerinde de test edilmiştir. Railsem19 veri setinin görüntülerinde, yayalar da insan olarak etiketlenmiştir. Bu nedenle, 635 görüntüden ve iki sınıftan (kişi ve demiryolu hattı) oluşan Railsem19 veri seti özel bir sürümü kullanılmıştır. Tablo 5.42, özelleştirilmiş Railsem19 veri kümesinde DNet modeli tarafından elde edilen performans ölçümlerini içermektedir.

Tablo 5.42. Yeni Railsem19 veri setinde DNet'in performansı

Model	Metrik	Değer
DNet	Ortalama IoU	0.7528
DNet	Doğrulama Oranı	0.9921
Klasik Unet	Ortalama IoU	0.6108
Klasik Unet	Doğrulama Oranı	0.9692

Şekil 5.62, DNet ve klasik Unet modellerinin test sonuçlarını göstermektedir. Şekil 5.62'de görüldüğü gibi, demiryolu hattı ve çevresindeki yayalar başarıyla bölütlendirilmiştir. Tablo 5.41 ve 5.42'de görüldüğü gibi, önerilen DNet modeli her iki veri kümesinde de klasik Unet modelinden daha başarılı sonuçlar elde etmiştir.



Şekil 5.62. (a) DNet'in Railsem19 veri kümesindeki test sonucu (b) Klasik Unet'in Railsem19 veri kümesindeki test sonucu

Demiryolu hattı etrafındaki yabancı cisimlerin bölütlemesinde kullanılan derin öğrenme modeli olarak geliştirilen DNet modeli de bu çalışma kapsamında değerlendirilmiştir. Gazebo simülasyon ortamında ve gerçek saha testleri sonucunda oluşturulan veri seti üzerinde incelenmiş ve elde edilen sonuçlar klasik Unet modeliyle karşılaştırıldığında dikkate değer bir etkinlik göstermiştir.

DNet modeli, demiryollarının karmaşık yapısını başarılı bir şekilde analiz ederek bölütleme görevinde %99 doğruluk oranına ve %85 ortalama IoU değerine ulaşmıştır.

Geliştirilen DNet modeli literatürdeki diğer bölütleme yöntemleriyle karşılaştırılmıştır. Tablo 5.43'te görüldüğü gibi geliştirilen DNet modeli literatürdeki diğer çalışmalardan daha başarılı olmuştur.

Tablo 5.43. DNet modelinin literatürdeki diğer bölütleme yöntemleriyle karşılaştırılması

Model	Ortalama IoU
Unet [141]	0.5278
Topluluk Modeli [141]	0.5430
RFODLab [142]	0.8378
DeepLabv3+ [142]	0.8009
DNet	0.8598

5.12. Hızlı ve Ekonomik Nesne Algılama Yoluyla Demiryolu Güvenliğinin Artırılması: Grounding Dino Yaklaşımı Metodunda Elde Edilen Bulgular

Grounding DINO, metin istemleriyle nesne algılamaya yönelik son teknoloji bir modeldir [143]. Giriş olarak, algılamak istediğiniz nesneleri (insanlar ve raylar) açıklayan bir resim ve metin istemi sağlanmıştır. Grounding DINO hem görüntüyü hem de metin istemini analiz etmek için ayrı metin ve görüntü işleme omurgalarını kullanmaktadır. Çıkarılan özellikler, nesne tespiti için ilgili bilgilerin vurgulanması amacıyla daha da geliştirilmiştir. Model, metin istemine göre görüntüden nesne açıklamasına uygun en alakalı özellikleri seçmektedir. Grounding DINO, görüntüde algılanan nesnelerin etrafındaki sınırlayıcı kutuların yanı sıra her algılamamanın ne kadar kesin olduğunu gösteren güven puanları da üretmiştir. Şekil 5.63'te görüldüğü gibi Grounding Dino, demiryolu etrafındaki insanları 0.79 ve 0.88 mAP oranlarıyla tespit etmiştir. Şekil 5.63'te, giriş metni istemi insan olarak ayarlanmıştır.



Şekil 5.63. Demiryolu etrafında tespit edilen insanlar

Şekil 5.64'te görüldüğü gibi Grounding Dino yöntemi, demiryolu üzerinde oturan bir kişiyi ve demiryolu üzerinde yürüyen bir kişiyi 0,67 ve 0,63 mAP oranlarıyla tespit etti. Şekil 5.64'teki örnekte giriş metinleri, istemin demiryolu üzerinde oturan kişiye ve demiryolu üzerinde yürüyen kişi olarak ayarlanmıştır.



Şekil 5.64. Demiryolu etrafında tespit edilen oturan ve yürüyen insanlar

Şekil 5.65'te görüldüğü gibi Grounding Dino yöntemi demiryolu üzerinde insan, ray ve bitki olmak üzere birden fazla nesne tespit etmiştir. mAP oranları sırasıyla 0,72, 0,43 ve 0,37'dir. Şekil 5.65'teki görselde, bilgi istemi metinleri insan, ray ve bitki olarak belirlenmiştir.



Şekil 5.65. Demiryolu etrafında tespit edilen insan, ray ve bitki

6. SONUÇLAR

Bu tez çalışmasında, demiryolu bakım hizmetlerinin önemi, demiryolu kusurlarının tespiti ve bu kusurların tespitine yönelik kullanılan teknikler ayrıntılı olarak ele alınmıştır. Günümüzde demiryolu taşımacılığının artan kullanımı ile, güvenlik ihtiyacı da paralel olarak büyümektedir. Artan kullanım nedeniyle raylarda ortaya çıkan kusurların tespiti ve sınıflandırılması, güvenli bir ulaşımın sağlanması için kritik bir gereklilik haline gelmiştir.

Tez çalışmasında, demiryolu üzerindeki kusurların tespiti ve sınıflandırılmasının demiryolu güvenliği açısından taşıdığı önem vurgulanmış, bu sorunların ilerleyen dönemlerde de artacağı göz önünde bulundurulmuştur. Bu kapsamda, topluluk öğrenmesine dayalı yöntem kullanılarak, raylardaki farklı kusurlar başarıyla tespit edilmiştir. Ayrıca, geliştirilen CNN tabanlı model ile demiryolu görüntülerindeki bağlantı elemanlarının konumu belirlenmiş ve kusurlar, türlerine göre doğru bir şekilde sınıflandırılmıştır.

Bunun yanı sıra, DNet modeli ile demiryolu etrafındaki yabancı cisimlerin tespiti etkin bir şekilde gerçekleştirilmiştir. Gazebo simülasyon ortamında, Anafi4k İHA'nın geliştirilen otonom uçuş yöntemi sayesinde saha testleri gerçekleştirilmiş ve bu testler sonucunda özgün bir veri seti oluşturulmuştur. Bu veri seti, demiryolu kusurlarının tespitine yönelik çalışmalara katkı sunacak niteliktedir.

Sonuçlar, geliştirilen modellerin demiryolu kusurlarını ve potansiyel tehlikeleri tespit etme ve sınıflandırma açısından başarılı olduğunu ortaya koymuştur. Çalışmada elde edilen bulgular, literatürde yer alan benzer çalışmalarla karşılaştırılarak önerilen yöntemlerin başarıları değerlendirilmiştir. Ayrıca, alternatif veri artırma teknikleri ile yeni kusurlu veri setleri oluşturulmuş ve bu tekniklerin derin öğrenme modellerinin performansını iyileştirmede önemli bir rol oynadığı gözlemlenmiştir.

Tez çalışması boyunca elde edilen bulgular, demiryolu güvenliğinin artırılması açısından önemli bir zemin oluşturmuştur. Özellikle, topluluk öğrenmesine dayalı yaklaşımın yanı sıra geliştirilen derin öğrenme modellerinin entegre edilmesi, kusurların tespitinde doğruluğu önemli ölçüde artırmıştır. Bu bağlamda, özellikle ray üzerindeki yüzey kusurlarının belirlenmesi ve sınıflandırılması, demiryolu işletmecileri için kritik bir öneme sahiptir. Geliştirilen DNet modelinin yanı sıra CNN tabanlı yaklaşımlar, zamanında yapılan müdahaleler sayesinde demiryolu güvenliğini sağlamada etkin bir rol oynamaktadır. Çalışmada kullanılan simülasyon ortamları ve otonom uçuş testleri, gerçek dünya koşullarında uygulanabilirliğini ve pratikliğini destekleyen önemli sonuçlar sunmaktadır.

Ayrıca, bu tez, demiryolu altyapısının izlenmesi ve bakım süreçlerinin optimize edilmesi açısından da katkı sağlamaktadır. Geliştirilen uygulamalar, demiryolu ağlarının izlenmesinde kullanılabilecek yenilikçi çözümler sunarak, bakım hizmetlerinin etkinliğini artırma potansiyeli

taşımaktadır. Uzun vadede, bu tür uygulamaların yaygınlaşması, demiryolu işletmelerinin sürdürülebilirliğine ve verimliliğine önemli katkılarda bulunacaktır. Elde edilen verilerin analizi, demiryolu işletmecilerine, altyapılarında meydana gelen sorunları önceden tahmin etme ve bu sorunlara yönelik stratejik çözümler geliştirme fırsatı sunmaktadır. Böylece, demiryolu güvenliğinin artırılması ve işletme verimliliğinin sağlanması açısından kritik bir adım atılmış olmaktadır.

Genel olarak, bu tez çalışması kapsamında geliştirilen yöntemler, demiryolu kusur tespiti alanında yenilikçi ve etkili çözümler sunmakta olup, demiryolu güvenliğinin sağlanmasında alternatif yaklaşımlar geliştirilmesine olanak tanımaktadır.



ÖNERİLER

Bu tez çalışmasında elde edilen bulgular ışığında, demiryolu kusurlarının tespitine yönelik uygulamaların daha da geliştirilmesi ve genişletilmesi için aşağıdaki öneriler sunulmaktadır:

- a) Gerçek Zamanlı İzleme Sistemleri: Geliştirilen yöntemlerin, gerçek zamanlı izleme sistemleri ile entegre edilmesi, anlık kusur tespitini ve müdahale sürelerini azaltacaktır. IoT tabanlı çözümler, veri akışının hızlanmasını sağlayabilir.
- b) Eğitim Programları: Demiryolu personelinin, yeni teknolojilerin kullanımı ve model sonuçlarının yorumlanması konusunda eğitim alması, elde edilen verilerin etkin bir şekilde kullanılmasını sağlayacaktır. Eğitim programlarının oluşturulması, operasyonel verimliliği artırabilir.
- c) Araştırma ve Geliştirme: Bu alanda daha fazla akademik araştırma yapılması, yeni teknolojilerin ve yöntemlerin keşfedilmesine katkıda bulunabilir. Özellikle, makine öğrenimi ve derin öğrenme tekniklerinin demiryolu güvenliği üzerindeki etkileri üzerine odaklanılması önerilmektedir.
- d) Çok Disiplinli Çalışmalar: Demiryolu güvenliği ile ilgili farklı disiplinlerin bir araya gelerek ortak projeler yürütmesi, sistemin genel verimliliğini artırabilir. Mühendislik, veri bilimi ve işletme alanlarından uzmanların iş birliği yapması faydalı olacaktır.

KAYNAKLAR

- [1] H. Aydemir, "Türkiye'nin Ulaştırma Politikaları Çerçevesinde Demiryolu Ulaştırma Sisteminin Genel Durumunun İrdelenmesi ve Geleceğine Bakış", Demiryolu Mühendisliği, sy. 3, ss. 41–46, Haziran 2016.
- [2] T. C. D. D., "Türkiye Cumhuriyeti Devlet Demiryolları," TCDD. <https://www.tcdd.gov.tr/files/istatistik/20152019yillik.pdf>, Erişim: 27 Şubat 2024.
- [3] O. Yaman, Mehmet Karakose, E. Akin, ve I. Aydın, "Image processing based fault detection approach for rail surface," May 2015, doi: <https://doi.org/10.1109/siu.2015.7130031>.
- [4] D. Çetintaş, "Derin Öğrenme ve Tıbbi Uygulamaları," Doktora Semineri, Bilgisayar Mühendisliği Bölümü, Fırat Üniversitesi, Elazığ, Türkiye, 2019.
- [5] F. Flammini, C. Pragliola ve G. Smarra, "Railway infrastructure monitoring by drones," 2016 International Conference on Electrical Systems for Aircraft, Railway, Ship Propulsion and Road Vehicles & International Transportation Electrification Conference (ESARS-ITEC), 2016, pp. 1-6, doi: [10.1109/ESARS-ITEC.2016.7841398](https://doi.org/10.1109/ESARS-ITEC.2016.7841398).
- [6] S. Faghih-Roohi, S. Hajizadeh, A. Núñez, R. Babuska, B. De Schutter, "Deep convolutional neural networks for detection of rail surface defects," 2016 International Joint Conference on Neural Networks (IJCNN), pp. 2584-2589, 2016.
- [7] Z. Liang, H. Zhang, L. Liu, Z. He, K. Zheng, "Defect Detection of Rail Surface with Deep Convolutional Neural Networks," 2018 13th World Congress on Intelligent Control and Automation (WCICA), pp. 1317- 1322, 2018.
- [8] H. Zhang, X. Jin, Q. M. J. Wu, Y. Wang, Z. He, Y. Yang, "Automatic Visual Detection System of Railway Surface Defects With Curvature Filter and Improved Gaussian Mixture Model" IEEE Transactions on Instrumentation and Measurement, vol. 67, no. 7, pp. 1593-1608, 2018.
- [9] Z. Liu, W. Wang, X. Zhang, ve W. Jia, "Inspection of rail surface defects based on image processing," Mar. 2010, doi: <https://doi.org/10.1109/car.2010.5456793>.
- [10] H. Kim, S. Lee ve S. Han, "Railroad Surface Defect Segmentation Using a Modified Fully Convolutional Network," KSII Transactions on Internet and Information Systems, vol. 14, no. 12, pp. 4763-4775, 2020.
- [11] J. Liu, Y. Huang, Q. Zou, M. Tian, S. Wang, X. Zhao, and S. Ren, "Learning visual similarity for inspecting defective railway fasteners," IEEE Sensors Journal, vol. 19, no. 16, pp. 6844–6857, 2019.
- [12] A. Ma, Z. Lv, X. Chen, L. Li, Y. Qiu, S. Zheng, and X. Chai, "Pandrol track fastener defect detection based on local convolutional neural networks," Proc. Inst. Mech. Eng., Part I: J. Syst. Control Eng., vol. 235, no. 10, pp. 1906–1915, 2021.
- [13] X. Wei, Z. Yang, Y. Liu, D. Wei, L. Jia, ve Y. Li, "Railway track fastener defect detection based on image processing and deep learning techniques: A comparative study," Engineering Applications of Artificial Intelligence, vol. 80, pp. 66–81, Feb. 2019, doi: [10.1016/j.engappai.2019.01.008](https://doi.org/10.1016/j.engappai.2019.01.008).
- [14] H. Feng, Z. Jiang, F. Xie, P. Yang, J. Shi ve L. Chen, "Automatic Fastener Classification and Defect Detection in Vision- Based Railway Inspection Systems," in IEEE Transactions on Instrumentation and Measurement, vol. 63, no. 4, pp. 877- 888, 2014.
- [15] F. Guo, Y. Qian, ve Y. Shi, "Real-time railroad track components inspection based on the improved YOLOv4 framework," Automation in Construction, vol. 125, p. 103596, Feb. 2021, doi: [10.1016/j.autcon.2021.103596](https://doi.org/10.1016/j.autcon.2021.103596).
- [16] D. Zheng, L. Li, S. Zheng, X. Chai, S. Zhao, Q. Tong, and L. Guo, "A defect detection method for rail surface and fasteners based on deep convolutional neural network," Comput. Intell. Neurosci., vol. 2021, no. 1, Art. no. 2565500, 2021.
- [17] J. Y. Xia, F. Xie ve Z. Jiang, "Broken Railway Fastener Detection Based on Adaboost Algorithm," 2010 International Conference on Optoelectronics and Image Processing, 2010, pp. 313-316, doi: [10.1109/ICOIP.2010.303](https://doi.org/10.1109/ICOIP.2010.303).

- [18] H. Fan, P. C. Cosman, Y. Hou ve B. Li, "High-Speed Railway Fastener Detection Based on a Line Local Binary Pattern," in *IEEE Signal Processing Letters*, vol. 25, no. 6, pp. 788-792, June 2018, doi: 10.1109/LSP.2018.2825947.
- [19] L. Shang, Q. Yang, J. Wang, S. Li ve W. Lei, "Detection of rail surface defects based on CNN image recognition and classification," 2018 20th International Conference on Advanced Communication Technology (ICACT), pp. 45-51, 2018.
- [20] H. Yu, Q. Li, Y. Tan, J. Gan, J. Wang, Y. A. Geng, and L. Jia, "A coarse-to-fine model for rail surface defect detection," *IEEE Trans. Instrum. Meas.*, vol. 68, no. 3, pp. 656–666, 2018.
- [21] S. Yanan, Z. Hui, L. Li and Z. Hang, "Rail Surface Defect Detection Method Based on YOLOv3 Deep Learning Networks," 2018 Chinese Automation Congress (CAC), 2018, pp. 1563-1568, doi: 10.1109/CAC.2018.8623082.
- [22] A. M. Hafiz and G. M. Bhat, "A survey on instance segmentation: state of the art," *International Journal of Multimedia Information Retrieval*, vol. 9, no. 3, pp. 171–189, Jul. 2020, doi: <https://doi.org/10.1007/s13735-020-00195-x>.
- [23] X. Giben, V. M. Patel, and R. Chellappa, "Material classification and semantic segmentation of railway track images with deep convolutional neural networks," *IEEE Xplore*, Sep. 01, 2015.
- [24] Z. He, P. Tang, W. Jin, C. Hu, and W. Li, "Deep Semantic Segmentation Neural Networks of Railway Scene," *IEEE Xplore*, Jul. 01, 2018.
- [25] Y. Furitsu, D. Deguchi, Y. Kawanishi, I. Ide, H. Murase, H. Mukojima, and N. Nagamine, "Semantic segmentation of railway images considering temporal continuity," in *Pattern Recognition: 5th Asian Conference, ACPR 2019, Auckland, New Zealand, November 26–29, 2019, Revised Selected Papers, Part I 5*, Springer International Publishing, 2020, pp. 639–652.
- [26] K. Jahan, J. P. Umesh, and M. Roth, "Anomaly detection on the rail lines using semantic segmentation and self-supervised learning," 2021 IEEE Symposium Series on Computational Intelligence (SSCI), Dec. 2021, doi: 10.1109/ssci50451.2021.9659920..
- [27] H. Wang, M. Li, and Z. Wan, "Rail surface defect detection based on improved Mask R-CNN," *Computers & Electrical Engineering*, vol. 102, p. 108269, Aug. 2022, doi: 10.1016/j.compeleceng.2022.108269.
- [28] L. Li, R. Sun, S. Zhao, X. Chai, S. Zheng, and R. Shen, "Semantic-Segmentation-Based Rail Fastener State Recognition Algorithm," *Mathematical Problems in Engineering*, vol. 2021, pp. 1–15, Mar. 2021, doi: 10.1155/2021/8956164.
- [29] J. Grandio, B. Riveiro, M. Soilán, and P. Arias, "Point cloud semantic segmentation of complex railway environments using deep learning," *Automation in Construction*, vol. 141, p. 104425, Jun. 2022, doi: 10.1016/j.autcon.2022.104425..
- [30] A. Schöbel and T. Maly, "Operational fault states in railways," *European Transport Research Review*, vol. 4, no. 2, pp. 107–113, Jan. 2012, doi: 10.1007/s12544-011-0068-z.
- [31] T. Koyama, "Detection of Pantograph Failures Using Sensor Fixed to Catenary System", *Railway Technology Avalanche*, vol. 41, 246.
- [32] A. A. Boguslavskii and S. M. Sokolov, "Detecting objects in images in real-time computer vision systems using structured geometric models," *Programming and Computer Software*, vol. 32, no. 3, pp. 177–187, May 2006, doi: 10.1134/s036176880603008x..
- [33] Bojarczak, P., & Lesiak, P. (2021). UAVs in rail damage image diagnostics supported by deep-learning networks. *Open Engineering*, 11(1), 339-348.
- [34] Debevec, R. (2019). A Smart UAV Platform for Railroad Inspection, *Electronic Theses and Dissertations*, 6475.
- [35] Páli, E., Mathe, K., Tamas, L., & Buşoni, L. (2014, May). Railway track following with the AR. Drone using vanishing point detection. In 2014 IEEE International Conference on Automation, Quality and Testing, Robotics (pp. 1-6). IEEE.
- [36] P. V. Verschraegen, 'A Model of the Pantograph Arc Impedance for 50 Hz Catenary Voltage', Dissertation, 2010.

- [37] I. Aydin, Mehmet Karakose, and E. Akin, "A New Contactless Fault Diagnosis Approach for Pantograph-Catenary System Using Pattern Recognition and Image Processing Methods," *Advances in Electrical and Computer Engineering*, vol. 14, no. 3, pp. 79–88, Aug. 2014, doi: <https://doi.org/10.4316/aece.2014.03010>.
- [38] M. Surajit, "Conducted and Radiated Electromagnetic Interference in Modern Electrified Railways with Emphasis on Pantograph Arcing," *Doktora Tezi, Kungliga Tekniska Högskolan, Stockholm, İsveç*, 2009.
- [39] Y. Yi, X. Ye, Z. Li, and K. Yue, "Non-contact measurement of contact wire," *Proceedings of SPIE, the International Society for Optical Engineering/Proceedings of SPIE*, vol. 7160, p. 71600O, Nov. 2008, doi: 10.1117/12.806919.
- [40] C. Li, Z. Xie, Y. Qin, L. Jia, L. Guan, and X. Ma, "Abnormal perception of railway perimeter based on trackside surveillance video," in *Lecture notes in electrical engineering*, 2022, pp. 149–157. doi: 10.1007/978-981-16-9909-2_16.
- [41] T. Guo, L. Jia, Z. Xie, Y. Qin, and F. Lin, "Research on the algorithm of cross-region target tracking in the perimeter of high speed railway based on deep learning," in *Lecture notes in electrical engineering*, 2022, pp. 176–183. doi: 10.1007/978-981-16-9909-2_19.
- [42] Y. Li, B. Fu, Y. Qin, R. Yuan, Z. Cao, Z. Xie, and Y. Gao, "Railway pedestrian intrusion detection using onboard forward-viewing camera," in *Proc. Int. Conf. Electr. Inf. Technol. Rail Transp.*, Singapore: Springer Singapore, Oct. 2021, pp. 388–396.
- [43] Z. Cao, Y. Qin, Z. Xie, Q. Liu, E. Zhang, Z. Wu, and Z. Yu, "An effective railway intrusion detection method using dynamic intrusion region and lightweight neural network," *Measurement*, vol. 191, Art. no. 110564, 2022.
- [44] T. Ye, X. Zhang, Y. Zhang, and J. Liu, "Railway traffic object detection using differential feature Fusion convolution Neural network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1375–1387, Feb. 2020, doi: 10.1109/tits.2020.2969993.
- [45] Y. Li, H. Dong, H. Li, X. Zhang, B. Zhang, and Z. Xiao, "Multi-block SSD based on small object detection for UAV railway scene surveillance," *Chinese Journal of Aeronautics*, vol. 33, no. 6, pp. 1747–1755, Mar. 2020, doi: 10.1016/j.cja.2020.02.024.
- [46] A. Mauri, R. Khemmar, B. Decoux, M. Haddad, and R. Bouteau, "Lightweight convolutional neural network for real-time 3D object detection in road and railway environments," *Journal of Real-Time Image Processing*, vol. 19, no. 3, pp. 499–516, Feb. 2022, doi: 10.1007/s11554-022-01202-6.
- [47] M. Sevi, "Sosyal Medyada Sahte Hesap ve Hesap Gruplarının Tespiti İçin Özellik Çıkarımı ve Yapay Zeka Temelli Tekniklerin Geliştirilmesi", *Yüksek Lisans Tezi, Fırat Üniversitesi, Elazığ, Türkiye*, 2019.
- [48] R. Daş , B. Polat ve G. Tuna , "Derin Öğrenme ile Resim ve Videolarda Nesnelerin Tanınması ve Takibi", *Fırat Üniversitesi Mühendislik Bilimleri Dergisi*, c. 31, sayı. 2, ss. 571-581, Eyl. 2019, doi:10.35234/fumbd.608778.
- [49] G. E. Hinton, "Learning multiple layers of representation," *Trends Cogn. Sci.*, vol. 11, no. 10, pp. 428–434, Oct. 2007.
- [50] G. Gündüz, İ. H. Cedimoğlu, "Derin Öğrenme Algoritmalarını Kullanarak Görüntüden Cinsiyet Tahmini," *Sakarya University Journal Of Computer And Information Sciences*, 2, 9-17, 2019.
- [51] Ö. Inik and E. Ülker, "Derin Öğrenme ve Görüntü Analizinde Kullanılan Derin Öğrenme Modelleri," *Gaziosmanpaşa Bilimsel Araştırma Dergisi*, vol. 6, no. 3, pp. 85–104, Dec. 2017, [Online]. Available: <http://dergipark.gov.tr/download/article-file/380999>.
- [52] M. Sevi and İ. Aydın, "COVID-19 Detection Using Deep Learning Methods," *2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI)*, Sakheer, Bahrain, 2020, pp. 1-6, doi: 10.1109/ICDABI51230.2020.9325626.
- [53] D. Çetintaş, "Göz Hareketlerinin Takip Edilmesiyle İnsan Davranış ve Sağlığının Tespiti", *Doktora Tezi, Fırat Üniversitesi, Elâzığ, Türkiye*, 2023.

- [54] S. Toraman and B. Daş, "Evrişimsel sinir ağları kullanılarak normal ve göğüs kanseri hücreleri içeren genomların sınıflandırılması," DÜMF Mühendislik Dergisi, vol. 11, no. 1, pp. 81–90, Mar. 2020, doi: 10.24012/dumf.610879.
- [55] E. Erguvan Etgin and E. Güvenoğlu, "Derin Öğrenme Yöntemleri Yardımıyla Görüntülerin Otomatik Konumlu Betimlenmesi ve Türkçe Alt Yazı Oluşturulması için Yeni Veri Kümesi ile bir Yöntem Önerisi", bbmd, vol. 17, no. 1, pp. 48–55, 2024, doi: 10.54525/bbmd.1454524.
- [56] E. Erdem and T. Aydın, "Göğüs kanseri Histopatolojik görüntü sınıflandırması," Bilişim Teknolojileri Dergisi, vol. 14, no. 1, pp. 87–94, Jan. 2021, doi: 10.17671/gazibtd.746673.
- [57] O. Bilginer, B. Tunga, ve R. M. Demirer, "Deri lezyonlarının evrişimsel yapay sinir ağları ile sınıflandırılması", Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi, c. 28, sy. 2, ss. 208–214, 2022.
- [58] İ. Aydın, M. Sevi, M. U. Salur, and E. Akin, "Defect classification of railway fasteners using image preprocessing and alightweight convolutional neural network," Turkish Journal of Electrical Engineering and Computer Sciences, vol. 30, no. 3, pp. 891–907, Mar. 2022, doi: 10.55730/1300-0632.3817.
- [59] A. Akdağ, "Derin öğrenme algoritmaları kullanılarak gerçek zamanlı silah tanıma uygulaması," 2017. [Online]. Available: <http://acikerisim.erbakan.edu.tr/xmlui/handle/20.500.12452/6612>.
- [60] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, Real-Time Object Detection," In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Jun. 2016, doi: 10.1109/cvpr.2016.91.
- [61] E. Güçlü, İ. Aydın, K. Şahbaz, E. Akin, ve M. Karaköse, "Demiryolu Bağlantı Elemanlarında Bulunan Kusurların YOLOv4 ve Bulanık Mantık Kullanarak Tespiti", Demiryolu Mühendisliği, sy. 14, ss. 249–262, Temmuz 2021, doi: 10.47072/demiryolu.939830.
- [62] S. Chourasia, R. Bhojane, and L. Heda, "Safety Helmet Detection: A Comparative Analysis Using YOLOv4, YOLOv5, and YOLOv7," IEEE Xplore, Jan. 01, 2023.
- [63] C. Dewi, R.-C. Chen, Y.-C. Zhuang, and H. J. Christanto, "Yolov5 Series Algorithm for Road Marking Sign Identification," Big Data and Cognitive Computing, vol. 6, no. 4, p. 149, Dec. 2022, doi: 10.3390/bdcc6040149.
- [64] E. Güney, "Sürücü asistan sistemleri için mobil gpu tabanlı gerçek zamanlı durum analizi ve tespit uygulamaları", Yüksek Lisans Tezi, Sakarya Üniversitesi, 2021.
- [65] G. Conley, S. C. Zinn, T. Hanson, K. McDonald, N. Beck, and H. Wen, "Using a deep learning model to quantify trash accumulation for cleaner urban stormwater," Computers Environment and Urban Systems, vol. 93, p. 101752, Jan. 2022, doi: 10.1016/j.compenvurbsys.2021.101752.
- [66] H. Li, J. Huang, Z. Gu, D. He, J. Huang, and C. Wang, "Positioning of mango picking point using an improved YOLOv8 architecture with object detection and instance segmentation," Biosyst. Eng., vol. 247, pp. 202–220, 2024..
- [67] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, and L. Zhang, "Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection," in Proc. Eur. Conf. Comput. Vis., Cham: Springer, 2025, pp. 38–55.
- [68] M. Sevi and İ. Aydın, "Enhancing railway safety through fast and economical object detection: a grounding DINO approach," in Lecture notes in networks and systems, 2024, pp. 198–205. doi: 10.1007/978-3-031-67192-0_25.
- [69] R. Terzi, M.U. Demirezen, "Açık Kaynak Veri Seti ile Eğitilen Yapay Zekâ Modellerinin Klinik Ortamdaki Performans Analizi", J Ankara Üniversitesi Fac Med. 2022 Dec;75(1):25-34.
- [70] E. Arıcı, "Derin öğrenme ile anlamsal bölütleme ve piksel görüntülerinden gerçek görüntü üretimi", Yüksek Lisans Tezi, Batman Üniversitesi Lisansüstü Eğitim Enstitüsü, Batman, 2023.
- [71] C. Yin, J. Tang, T. Yuan, Z. Xu and Y. Wang, "Bridging the Gap Between Semantic Segmentation and Instance Segmentation," in IEEE Transactions on Multimedia, vol. 24, pp. 4183-4196, 2022, doi: 10.1109/TMM.2021.3114541.

- [72] O. Ronneberger, P. Fischer, and T. Brox, "U-NET: Convolutional Networks for Biomedical Image Segmentation," in *Lecture notes in computer science*, 2015, pp. 234–241. doi: 10.1007/978-3-319-24574-4_28.
- [73] F. Türk, M. Lüy ve N. Barışçı, "Böbrek Tümör Segmentasyonu İçin Unet ve Unet-ResNet Modellerinin Karşılaştırılması," 2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, Turkey, 2019, pp. 1-5, doi: 10.1109/ISMSIT.2019.8932725.
- [74] N. Pervan, "Semantic Inference From Turkish Texts Using Deep Learning Approaches", Yüksek Lisans Tezi, Bilgisayar Mühendisliği, Ankara Üniversitesi, Ankara, Türkiye, 2019.
- [75] F. Yu, & V. Koltun, "Multi-Scale Context Aggregation by Dilated Convolutions" CoRR, 2015, abs/1511.07122.
- [76] E. Courdier and F. Fleuret, "Real-Time Segmentation Networks Should be Latency Aware," in *Lecture notes in computer science*, 2021, pp. 603–619. doi: 10.1007/978-3-030-69525-5_36.
- [77] O. Zendel, M. Murschitz, M. Zeilinger, D. Steininger, S. Abbasi and C. Beleznaï, "RailSem19: A Dataset for Semantic Rail Scene Understanding," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Long Beach, CA, USA, 2019, pp. 1221-1229, doi: 10.1109/CVPRW.2019.00161.
- [78] M. A. Mazurowski, H. Dong, H. Gu, J. Yang, N. Konz, and Y. Zhang, "Segment anything model for medical image analysis: An experimental study," *Medical Image Analysis*, vol. 89, p. 102918, Aug. 2023, doi: 10.1016/j.media.2023.102918.
- [79] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, and R. Girshick, "Segment anything," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 4015–4026.
- [80] M. Sevi, İ. Aydın and T. K. Şener, "Segmentation of Railway Rails Using SAM," 2023 4th International Conference on Data Analytics for Business and Industry (ICDABI), Bahrain, 2023, pp. 211-213, doi: 10.1109/ICDABI60145.2023.10629571.
- [81] E. I. Ünlü and A. Çinar, *Generating Ground Truth Images Using GAN (Generative Adversarial Network)*. EasyChair, 2022.
- [82] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, no. 1, 2019.
- [83] G. Çelik and M. F. Talu, "Çekişmeli üretken ağ modellerinin görüntü üretme performanslarının incelenmesi," *Balıkesir Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, vol. 22, no. 1, pp. 181–192, 2019.
- [84] I. Aydın, M. Sevi, and M. U. Salur, "Detection of fake twitter accounts with machine learning algorithms," in *2018 International Conference on Artificial Intelligence and Data Processing (IDAP)*, 2018.
- [85] Z. Yücel ve D. Çetintaş, "Yolov9 ile Kan Hücrelerinin Otomatik Tanımlanması: Optimizasyon ve Öğrenme Oranı Etkileri", *Adıyaman Üniversitesi Mühendislik Bilimleri Dergisi*, c. 11, sy. 22, ss. 125–135, 2024, doi: 10.54365/adyumbd.1388891.
- [86] Y. Majeed, J. Zhang, X. Zhang, L. Fu, M. Karkee, Q. Zhang, and M. D. Whiting, "Apple tree trunk and branch segmentation for automatic trellis training using convolutional neural network based semantic segmentation," *IFAC-PapersOnLine*, vol. 51, no. 17, pp. 75–80, 2018.
- [87] A. Borji, "Pros and cons of GAN evaluation measures," *Comput. Vis. Image Underst.*, vol. 179, pp. 41–65, 2019.
- [88] S. Alasu and M. F. Talu, "Mısır Tohumu Embriyolarının Bölütlenmesinde Tam Evrimsel Ağ Tabanlı Mimarilerin Tam Bağlı Şartlı Rastgele Alanlar ile Entegrasyonu," *Afyon Kocatepe Univ. J. Sci. Eng.*, vol. 22, no. 1, pp. 100–111, 2022.
- [89] K. Takaya, T. Asai, V. Kroumov, and F. Smarandache, "Simulation environment for mobile robots testing using ROS and Gazebo," in *Proc. 2016 20th Int. Conf. Syst. Theory, Control Comput. (ICSTCC)*, Oct. 2016, pp. 96–101.

- [90] M. Sevi, İ. Aydm, E. Güçlü and E. Akin, "Developing an Image Processing Based Method for Rail Tracking with an Unmanned Aerial Vehicle in a Simulation Environment," 2021 Innovations in Intelligent Systems and Applications Conference (ASYU), Elazig, Turkey, 2021, pp. 1-5, doi: 10.1109/ASYU52992.2021.9599085.
- [91] J. Canny, "A computational approach to edge detection," IEEE Trans. Pattern Anal. Mach. Intell., no. 6, pp. 679–698, 1986.
- [92] J. M. Geusebroek, A. W. M. Smeulders, and J. van de Weijer, "Fast anisotropic Gauss filtering," IEEE Trans. Image Process., vol. 12, no. 8, pp. 938–943, Aug. 2003, doi: 10.1109/TIP.2003.812429.
- [93] I. Aydin, M. Karakose, and E. Akin, "A new contactless fault diagnosis approach for pantograph-catenary system using pattern recognition and image processing methods," Adv. Electr. Comput. Eng., vol. 14, no. 3, pp. 79–88, 2014.
- [94] J. Illingworth and J. Kittler, "A survey of the Hough transform," Comput. Vis. Graph. Image Process., vol. 44, no. 1, pp. 87–116, 1988.
- [95] L. Chandrasekar and G. Durga, "Implementation of Hough transform for image processing applications," in Proc. 2014 Int. Conf. Commun. Signal Process., Apr. 2014, pp. 843–847.
- [96] T. Huang, D. Huang, and D. Luo, "Attitude tracking for a quadrotor UAV based on fuzzy PID controller," in Proc. 2018 5th Int. Conf. Inf., Cybernetics, Comput. Social Syst. (ICCSS), Aug. 2018, pp. 1–6..
- [97] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in Computer Vision – ECCV 2016, Cham: Springer International Publishing, 2016, pp. 630–645.
- [98] Z. Jiang, L. Zhao, S. Li, and Y. Jia, "Real-time object detection method based on improved YOLOv4-tiny," arXiv [cs.CV], 2020.
- [99] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," arXiv [cs.CV], 2016.
- [100] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," arXiv [cs.CV], 2018.
- [101] L. Ang, B. Zhang, J. Wu, H. Xu, X. Chen, and W. Na, "Computer vision system for detecting the loss of rail fastening nuts based on kernel two-dimensional principal component-two-dimensional principal component analysis and a support vector machine," Proceedings of the Institution of Mechanical Engineers, vol. 230, pp. 1842–1850, 2016.
- [102] D. Mery and F. Pedreschi, "Segmentation of colour food images using a robust algorithm," J. Food Eng., vol. 66, no. 3, pp. 353–360, 2005.
- [103] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," IEEE Transactions on Systems Man and Cybernetics, vol. 9, no. 1, pp. 62–66, Jan. 1979, doi: 10.1109/tsmc.1979.4310076.
- [104] J. N. Anita and S. Kumaran, "An efficient meningioma tumor segmentation system in WMSN using UNET-RCNN classification approaches," J. Pharm. Negative Results, pp. 1735–1743, 2023.
- [105] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. "," IEEE Transactions on pattern analysis and machine intelligence, vol. 24, no. 7, pp. 971–987, 2002.
- [106] H. Fan, P. C. Cosman, Y. Hou, and B. Li, "High-speed railway fastener detection based on a line local binary pattern. "," IEEE Signal Processing Letters, vol. 25, no. 6, pp. 788–792, 2018.
- [107] P. Mahalakshmi and N. S. Fatima, "Ensembling of text and images using Deep Convolutional Neural Networks for Intelligent Information Retrieval"," Wireless Personal Communications, pp. 1–19, 2021.
- [108] O. Budak, "SegNet Mimarisi ile Bilgisayarlı Tomografi Görüntülerinden Karaciger Bolgesinin Bölütlenmesi," Fırat Üniversitesi Mühendislik Bilimleri Dergisi, c, pp. 215–222, 2019.
- [109] L. Zhuang, L. Wang, Z. Zhang, and K. Tsui, "Automated Vision inspection of Rail Surface Cracks: A Double-Layer Data-Driven Framework"," Transportation Research Part C: Emerging Technologies, vol. 92, pp. 258–277, 2018.

- [110] H. Li, F. Wang, J. Liu, H. Song, Z. Hou, and P. Dai, "Ensemble model for rail surface defects detection," *PLoS One*, vol. 17, no. 5, p. e0268518, 2022.
- [111] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, no. 1, 2019.
- [112] V. Kotu and B. Deshpande, "Data Science process," in Elsevier eBooks, 2019, pp. 19–37. doi: 10.1016/b978-0-12-814761-0.00002-2.
- [113] D. Çetintaş and T. Tuncer, "Determining the Type of Document Read Using Eye Movement Properties by Hybrid CNN Method," *Traitement du Signal*, vol. 39, no. 4, 2022.
- [114] E. Menteşe and E. Hançer, "Histopatoloji Görüntülerde Derin Öğrenme Yöntemleri ile Çekirdek Segmentasyonu," *European Journal of Science and Technology*, 2020.
- [115] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- [116] M. H. Bendiabdallah and N. Settouti, "A comparison of U-net backbone architectures for the automatic white blood cells segmentation," *WAS Science Nature (WASSN)*, vol. 4, no. 1, 2021.
- [117] S. Jadon, "A survey of loss functions for semantic segmentation," in 2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), 2020.
- [118] R. Zhao, B. Qian, X. Zhang, Y. Li, R. Wei, Y. Liu, and Y. Pan, "Rethinking dice loss for medical image segmentation," in *Proc. 2020 IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2020, pp. 851–860.
- [119] J. Chang, X. Zhang, M. Ye, D. Huang, P. Wang, and C. Yao, "Brain tumor segmentation based on 3D Unet with multi-class focal loss," in *Proc. 2018 11th Int. Congr. Image Signal Process., BioMed. Eng. Informatics (CISP-BMEI)*, Oct. 2018, pp. 1–5.
- [120] D. Kang, S. S. Benipal, D. L. Gopal, and Y.-J. Cha, "Hybrid pixel-level concrete crack segmentation and quantification across complex backgrounds using deep learning," *Autom. Constr.*, vol. 118, no. 103291, p. 103291, 2020.
- [121] R.Y. Ju and W. Cai, "Fracture detection in pediatric wrist trauma X-ray images using YOLOv8 algorithm," *Sci. Rep.*, vol. 13, no. 1, p. 20077, 2023.
- [122] C. Fang and X. Yang, "Lightweight YOLOv8 for Wheat Head Detection," in *IEEE Access*, vol. 12, pp. 66214–66222, 2024, doi: 10.1109/ACCESS.2024.3397556.
- [123] N. Chitraningrum, L. Banowati, D. Herdiana, B. Mulyati, I. Sakti, A. Fudholi, and A. Andria, "Comparison study of corn leaf disease detection based on deep learning YOLO-v5 and YOLO-v8," *J. Eng. Technol. Sci.*, vol. 56, no. 1, pp. 61–70, 2024.
- [124] C. Guo, M. Szemenyei, Y. Pei, Y. Yi, and W. Zhou, "SD-unet: A structured dropout U-net for retinal vessel segmentation," in 2019 IEEE 19th International Conference on Bioinformatics and Bioengineering (BIBE), 2019.
- [125] Y. Wang, L. Wang, Y. H. Hu, and J. Qiu, "RailNet: A segmentation network for railroad detection," *IEEE Access*, vol. 7, pp. 143772–143779, 2019.
- [126] S. Belyaev, I. Popov, V. Shubnikov, P. Popov, E. Boltenkova, and D. Savchuk, "Railroad semantic segmentation on high-resolution images," in 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), 2020.
- [127] Y. Dou, Y. Huang, Q. Li, and S. Luo, "A fast template matching-based algorithm for railway bolts detection," *International Journal of Machine Learning and Cybernetics*, vol. 5, no. 6, pp. 835–844, 2014.
- [128] H. Zhong, L. Liu, J. Wang, Q. Fu, and B. Yi, "A real-time railway fastener inspection method using the lightweight depth estimation network," *Measurement (Lond.)*, vol. 189, no. 110613, p. 110613, 2022.
- [129] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

- [130] S. Mascarenhas and M. Agarwal, "A comparison between VGG16, VGG19 and ResNet50 architecture frameworks for image classification," in Proc. 2021 Int. Conf. Disruptive Technol. Multi-Disciplinary Res. Appl. (CENTCON), vol. 1, Nov. 2021, pp. 96–99.
- [131] Z. Tu, S. Wu, G. Kang, and J. Lin, "Real-time defect detection of track components: Considering class imbalance and subtle difference between classes," IEEE Trans. Instrum. Meas., vol. 70, pp. 1–12, 2021.
- [132] J. Liu, Y. Teng, X. Ni, and H. Liu, "A fastener inspection method based on defective sample generation and deep convolutional neural network," IEEE Sens. J., vol. 21, no. 10, pp. 12179–12188, 2021.
- [133] D. Zhang, K. Song, J. Xu, Y. He, M. Niu, and Y. Yan, "MCnet: Multiple context information segmentation network of no-service rail surface defects," IEEE Trans. Instrum. Meas., vol. 70, pp. 1–9, 2021.
- [134] C. Zhang, X. Hu, J. He, and N. Hou, "Yolov4 high-speed train wheelset tread defect detection system based on multiscale feature fusion," J. Adv. Transp., vol. 2022, pp. 1–13, 2022.
- [135] X. Song, K. Chen, and Z. Cao, "ResNet-based image classification of railway shelling defect," in 2020 39th Chinese Control Conference (CCC), 2020.
- [136] R. Gasparini, A. D'Eusania, G. Borghi, S. Pini, G. Scaglione, S. Calderara, and R. Cucchiara, "Anomaly detection, localization and classification for railway inspection," in Proc. 2020 25th Int. Conf. Pattern Recognit. (ICPR), Jan. 2021, pp. 3419–3426.
- [137] C.I. Cira, M.Á. Manso-Callejo, R. Alcarria, T. Fernández Pareja, B. Bordel Sánchez, and F. Serradilla, "Generative learning for postprocessing semantic segmentation predictions: A lightweight conditional Generative Adversarial Network based on Pix2pix to improve the extraction of road surface areas," Land (Basel), vol. 10, no. 1, p. 79, 2021.
- [138] O. Katar and E. Duman, "Automated Semantic Segmentation for Autonomous Railway Vehicles," Tehnički glasnik, vol. 16, no. 4, pp. 484–490, 2022.
- [139] D. Daimary, M. B. Bora, K. Amitab, and D. Kandar, "Brain tumor segmentation from MRI images using hybrid convolutional neural networks," Procedia Comput. Sci., vol. 167, pp. 2419–2428, 2020.
- [140] X. Fu, K. Li, J. Liu, K. Li, Z. Zeng, and C. Chen, "A two-stage attention aware method for train bearing shed oil inspection based on convolutional neural networks," Neurocomputing, vol. 380, pp. 212–224, 2020.
- [141] M. Sevi and İ. Aydın, "Improving Unet segmentation performance using an ensemble model in images containing railway lines," Turkish Journal of Electrical Engineering and Computer Sciences, vol. 31, no. 4, pp. 739–750, 2023.
- [142] H. Song, S. Wang, Z. Gu, P. Dai, X. Du, and Y. Cheng, "Modeling and optimization of semantic segmentation for track bed foreign object based on attention mechanism," IEEE Access, vol. 9, pp. 86646–86656, 2021.
- [143] Y. Cheng, L. Li, Y. Xu, X. Li, Z. Yang, W. Wang, and Y. Yang, "Segment and track anything," arXiv preprint arXiv:2305.06558, 2023.

ÖZGEÇMİŞ

Mehmet SEVİ

Category	Value
Category 1	Value 1
Category 2	Value 2
Category 3	Value 3
Category 4	Value 4
Category 5	Value 5
Category 6	Value 6
Category 7	Value 7

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

- [REDACTED] [REDACTED]
- [REDACTED] [REDACTED]

[REDACTED]
[REDACTED]

AKADEMİK FAALİYETLER

Makaleler:

1. Sevi, M., & Aydin, I. (2023). Improving Unet segmentation performance using an ensemble model in images containing railway lines. *Turkish Journal of Electrical Engineering and Computer Sciences*, 31(4), 739-750.
2. Aydin, I., Sevi, M., Akin, E., Güçlü, E., Karaköse, M., & Aldarwich, H. (2023). A deep learning-based hybrid approach to detect fastener defects in real-time. *Tehnički vjesnik*, 30(5), 1461-1468.
3. Aydin, I., Sevi, M., Salur, M. U., & Akin, E. (2022). Defect classification of railway fasteners using image preprocessing and a lightweight convolutional neural network. *Turkish Journal of Electrical Engineering and Computer Sciences*, 30(3), 891-907.
4. Sevi, M., & Aydın, İ. (2025). Enhanced railway monitoring and segmentation using DNet and mathematical methods. *Signal, Image and Video Processing*, 19(1), 1-14.

5. Sevi, M., Aydın, İ., & Akın, E. (2023). YOLOv5 ile Topluluk Öğrenmesine Dayalı Olarak Ray Yüzeyindeki Kusurların Tespiti. *Demiryolu Mühendisliği*, (17), 115-132.
6. Aydın, İ., Şener, T. K., & Sevi, M. (2024). Demiryolu Ray ve Çevresinin Anlamlandırılması için Derin Öğrenme Tabanlı Yöntemlerin Karşılaştırmalı Analizi. *Demiryolu Mühendisliği*, (19), 1-16.
7. Sevi, M., Aydın, İ., & Karaköse, M. (2022). Derin Öğrenme Yöntemleri ile Demiryolu Bağlantı Elemanlarının Sınıflandırılması. *Avrupa Bilim ve Teknoloji Dergisi*, (35), 268-274.
8. Sevi, M., & Aydın, İ. (2023). Detection of foreign objects around the railway line with yolov8. *Computer Science*, (IDAP-2023), 19-23.
9. Tuğal, İ., Almaz, C., & Sevi, M. (2021). Üniversitelerdeki Siber güvenlik sorunları ve farkındalık eğitimleri. *Bilişim Teknolojileri Dergisi*, 14(3), 229-238

Bildiriler:

1. Sevi, M., & Aydın, İ. (2020, October). COVID-19 detection using deep learning methods. In 2020 International conference on data analytics for business and industry: way towards a sustainable economy (ICDABI) (pp. 1-6). IEEE.
2. Aydın, I., Mehmet, Sevi & Salur, M. U. (2018, September). Detection of fake twitter accounts with machine learning algorithms. In 2018 International Conference on Artificial Intelligence and Data Processing (IDAP) (pp. 1-4). IEEE.
3. Aydın, I., Sevi, M., Sahbaz, K., & Karaköse, M. (2021, October). Detection of Rail Defects with Deep Learning Controlled Autonomous UAV. In 2021 International Conference on Data Analytics for Business and Industry (ICDABI) (pp. 500-504). IEEE.
4. Sevi, M., & Aydın, İ. (2021, July). Detecting flaws on railways using semantic segmentation. In 2021 International Conference on Information Technology (ICIT) (pp. 179-183). IEEE.
5. Sevi, M., & Aydın, İ. (2019, September). Detection of Fake Twitter Accounts with Multiple Classifier and Data Augmentation Technique. In 2019 International Artificial Intelligence and Data Processing Symposium (IDAP) (pp. 1-6). IEEE.
6. Sevi, M., & Aydın, İ. (2022, March). Rail Tracking and Detection with Drone in Gazebo Environment. In 2022 International Conference on Decision Aid Sciences and Applications (DASA) (pp. 1450-1454). IEEE.
7. Sevi, M., Aydın, İ., Güçlü, E., & Akın, E. (2021, October). Developing an Image Processing Based Method for Rail Tracking with an Unmanned Aerial Vehicle in a Simulation Environment. In 2021 Innovations in Intelligent Systems and Applications Conference (ASYU) (pp. 1-5). IEEE.
8. Sevi, M., Aydın, İ., & Şener, T. K. (2023, October). Segmentation of Railway Rails Using SAM. In 2023 4th International Conference on Data Analytics for Business and Industry (ICDABI) (pp. 211-213). IEEE.
9. Aydın, İ., & Sevi, M. (2023, August). Intelligent Unsupervised Defect Detection of Rail Surface via Generative Adversarial Networks. In International Conference on Intelligent and Fuzzy Systems (pp. 222-229). Cham: Springer Nature Switzerland.
10. Sevi, M., & Aydın, İ. (2023, August). Intelligent Classification of Defective Rails Through GAN and Ensemble Predictive Model. In International Conference on Intelligent and Fuzzy Systems (pp. 230-237). Cham: Springer Nature Switzerland.
11. Aydın, İ., Sevi, M., Güngören, G., & İrez, H. C. (2022, October). Signal Synchronization of Traffic Lights Using Reinforcement Learning. In 2022 International Conference on Data Analytics for Business and Industry (ICDABI) (pp. 103-108). IEEE.

12. Sevi, M., & Aydın, İ. (2019, November). Determination of Bot Groups in Social Network Accounts by Multiple Sequence Alignment Method. In 2019 1st International Informatics and Software Engineering Conference (UBMYK) (pp. 1-4). IEEE.
13. Sevi, M., & Aydın, İ. Enhancing Railway Safety Through Fast and Economical Object Detection: A Grounding DINO Approach. In Intelligent and Fuzzy Systems: Intelligent Industrial Informatics and Efficient Networks Proceedings of the INFUS 2024 Conference, Volume 3 (p. 198). Springer Nature.

Projeler:

1. TUBİTAK 1001, Demiryollarında Otonom İHA Kullanan Görme Tabanlı Akıllı Hata Tespit Sisteminin Geliştirilmesi, 120E097, 2020-2022, Doktora Bursiyeri
2. Fırat Üniversitesi Araştırma Destekleme Projesi, Demiryolu Araçları İçin Sentetik Görme Tabanlı Akıllı Otonom Sistem Geliştirme, ADEP.22.02, 2022-2024, Araştırmacı

