T.C.
FATİH SULTAN MEHMET VAKIF ÜNİVERSİTESİ
Mühendislik Fakültesi
Bilgisayar Mühendisliği Bölümü

Lisans Bitirme Projesi I

Otonom Araç İçin Derin Öğrenme Tabanlı Çözüm Yaklaşımları

Şerafettin Doruk SEZER - Ali Enes DOĞAN - Dilara ÇELİK
2021221038 - 2121221010 - 2021221042
Doç. Dr. Berna KİRAZ

İstanbul, Ocak 2019

İçindekiler

Şekil Listesi

Tablo Listesi

1 Problem Tanımı ve Motivasyon

Otonom araç teknolojileri, trafikte insan faktöründen kaynaklanan hataları en aza indirgemeyi, güvenliği artırmayı ve ulaşımı daha verimli hale getirmeyi hedefleyen bir yenilik alanıdır. Bu projede, sürücüsüz araçların temel ihtiyaçlarından olan şerit takibi, trafik işareti tespiti, araç, yaya ve sürüşe uygun alanların algılanması gibi görevler, Segmentasyon, nesne tespiti ve derin öğrenme algoritmalarını çoklu görev öğrenme yaklaşımı uygulanarak, CARLA simülasyon platformu üzerinde simülasyonu yapılacaktır. Bu Proje, hem teorik hem de uygulamalı bir bakış açısıyla otonom araç teknolojilerinin gelişimine katkı sağlayacaktır.

1.1 Bilime Katkısı

Şerit takibi ve trafik işareti tespiti gibi görevlerin çoklu görev algoritmaları ile entegre edilmesi, literatüre bu problemlerin ortak çözümüne yönelik yeni bir yaklaşım kazandıracaktır. Çoklu görev öğrenme alanında, birden fazla problemin eşzamanlı çözülmesinin performansa etkisi incelenecek ve analiz edilecektir.

1.2 Uygulamaya Katkısı

Geliştirilen yöntemler, otonom araçlar için hızlı, doğru ve ölçeklenebilir çözümler sağlayarak güvenli sürüşe katkıda bulunacaktır. Araçların daha verimli sürüş yaparak trafik kazalarını ve yakıt tüketimini azaltması mümkün olacaktır.

1.3 Amaç

Bu proje, sürücüsüz araçlar için bilgisayarlı görü ve derin öğrenme tekniklerini kullanarak çevre algılama problemlerini çözmeyi ve çoklu görev algoritmaları ile bu çözümleri entegre etmeyi amaçlamaktadır. Uzun vadede, gerçek dünya senaryolarında uygulanabilecek güvenilir ve verimli otonom araç algoritmaları geliştirilmesi hedeflenmektedir.

1.4 Hedefler

- **Şerit Takibi:** Şerit takibi görevini bir görüntü bölütleme (segmentasyon) problemi olarak ele almak ve çözmek. Şerit algılama için görüntüdeki her pikselin sınıflandırılmasını sağlayacak bir segmentasyon modeli geliştirilecektir. Modelin, özellikle farklı ışık ve hava koşullarında performansını artıracak şekilde optimize edilecektir.
- **Trafik İşareti, Yaya ve Araç Algılama:** Trafik işareti, yaya ve araç algılamayı bir nesne tespiti problemi olarak ele almak ve çözmek. YOLOv8 gibi gerçek zamanlı nesne tespiti algoritmalarını kullanarak yüksek doğrulukta bir model geliştirilmesi. Farklı trafik işareti türlerini, yaya ve araçları doğru bir şekilde algılamak için modelin eğitimi ve test edilmesi.

- **Çoklu Görev Modeli Geliştirilmesi:** Şerit takibi ve nesne tespiti görevlerini bir araya getiren çoklu görev öğrenme algoritmaları tasarlamak ve uygulamak. Ortak bir derin öğrenme mimarisi kullanarak iki görevin aynı modelde eşzamanlı olarak çözülmesi. Çoklu görev kayıp fonksiyonlarının dikkatli bir şekilde tasarlanarak her iki görevin doğruluğunu optimize etmesi.
- **Simülasyon Testleri ve Doğrulama:** Geliştirilen algoritmaları CARLA simülasyonu üzerinde test ederek doğrulamak ve analiz etmek. Farklı simülasyon senaryoları (örneğin, yoğun trafik, değişen hava koşulları) oluşturularak modellerin performansının değerlendirilmesi. Test sonuçlarının istatistiksel olarak analiz edilmesi ve modelin güçlü ve zayıf yönlerinin belirlenmesi.

2 Proje Kapsamı:

Bu bölümde proje kapsamında ele alınan problemler, kullanılan simülasyon araçları ve algoritmalar detaylı bir şekilde açıklanmıştır.

2.1 Problemin Tanımı:

Otonom araçların güvenli ve etkin bir şekilde çalışabilmesi için çevre algılaması ve nesne tanıma yeteneklerinin geliştirilmesi gerekmektedir. Projede ele alınan temel problemler şunlardır:

- **Şerit Tespiti:** Araçların yoldaki şeritleri algılayarak doğru şeritte ilerlemesi için bölütleme teknikleri kullanılarak şerit tespiti yapılacaktır.
- **Yaya Tespiti:** Yayaların güvenliğinin sağlanması için derin öğrenme tabanlı modellerle yaya tespiti gerçekleştirilecektir.
- **Trafik İşareti Tespiti:** Trafik işaretlerinin tanınması ve yorumlanması, otonom araçların karar mekanizmasında kritik bir rol oynamaktadır.
- **Araç Tespiti:** Diğer araçların tespiti ve pozisyonlarının belirlenmesi, çarpışmaları önlemek ve güvenli bir sürüş sağlamak için gereklidir.
- **Sürüşe Uygun Alanların Belirlenmesi:** Araçların sürüş sırasında engellerden kaçınarak güvenli bölgelerde seyahat etmesi için bu bölgelerin algılanması gereklidir.

2.2 CARLA Simülasyonu

CARLA (Car Learning to Act), otonom araç araştırmaları için geliştirilmiş açık kaynaklı bir simülasyon platformudur. Projede CARLA simülasyonu kullanılarak gerçek dünyaya yakın senaryolar oluşturulacak ve algoritmalar test edilecektir. CARLA'nın sunduğu özellikler:

- Gerçekçi şehir ortamları
- Farklı trafik durumlarını ve hava koşullarını simüle edebilme
- Geniş kapsamlı veri kümeleri ve sensör simülasyonları

Şekil 1: CARLA Simülasyon ortamı

2.2.1 CARLA Simülasyonunun Kısıtları

- **Hesaplama Kaynakları:** Yüksek kaliteli simülasyonlar, yüksek hesaplama gücü gerektirir.
- **Gerçek Dünya Verisi ile Tutarlılık:** Simülasyondan elde edilen veriler, gerçek dünya verilerinden farklılık gösterebilir.
- **Kısıtlı Veri Kümesi:** CARLA'nın sunduğu veri kümeleri belirli senaryolarla sınırlıdır ve daha fazla çeşitlilik için ek veri kümelerine ihtiyaç duyulabilir.

3 Kullanılan Teknolojiler

Proje kapsamında kullanılacak teknolojiler ve yöntemler şu şekilde özetlenebilir:

3.1 YOLOv8 – Nesne Tespiti

Derin Öğrenmede Nesne Tespiti: YOLOv8 (You Only Look Once, Versiyon 8), nesne tespiti ve görüntü işleme görevlerinde kullanılan, hız ve doğruluk açısından optimize edilmiş bir derin öğrenme algoritmasıdır. YOLOv8, önceki YOLO modellerinin (YOLOv3, YOLOv4, YOLOv5 gibi) en güncel versiyonudur ve nesne tespiti, segmentasyon ve sınıflandırma görevlerinde etkili çözümler sunar.

3.1.1 YOLOv8'in Genel Özellikleri:

- **Tek Aşamalı Algoritma:** YOLO, nesne tespiti sürecini tek bir adımda gerçekleştiren bir algoritmadır. Bu, işlem hızını artırır ve gerçek zamanlı uygulamalara uygun hale getirir.
- **Esnek Mimari:** YOLOv8, modüler bir yapıya sahiptir ve çeşitli görevler için özelleştirilebilir. Sadece nesne tespiti değil, aynı zamanda görüntü segmentasyonu ve sınıflandırma için de kullanılabilir.
- **Gelişmiş Performans:** YOLOv8, daha hafif ve optimize edilmiş bir mimari sunarak daha az kaynakla daha yüksek doğruluk sağlamayı hedefler.

3.1.2 YOLOv8'in Otonom Araçlardaki Kullanımı:

Proje kapsamında YOLOv8, sürücüsüz araç görevlerine yönelik olarak aşağıdaki alanlarda kullanılacaktır:

- **Trafik İşareti Tespiti:** Trafik işaretlerinin hızlı ve doğru bir şekilde tanınması, araçların çevreye uyum sağlamasında kritik öneme sahiptir. YOLOv8, trafik işaretlerini yüksek doğrulukla tespit etmek için eğitilecektir.
- **Yaya Tespiti:** Yayaların tespiti, özellikle kaza risklerini azaltmak ve güvenli sürüş sağlamak için önemlidir. YOLOv8, yayaların farklı pozisyonlarda ve açılardaki görüntülerini tanıyabilir.
- **Araç Tespiti:** Diğer araçların pozisyonunun belirlenmesi, çarpışmaları önlemek için gereklidir. YOLOv8, çeşitli boyut ve açılardaki araçları tanımlamak için kullanılacaktır.

3.2 OpenCV – Görüntü İşleme

OpenCV, görüntü işleme görevleri için kullanılacak temel kütüphanedir. Şerit tespiti ve sürüşe uygun alanların belirlenmesinde OpenCV'nin segmentasyon ve görüntü analizi yöntemleri kullanılacaktır.

3.3 U-Net – Segmentasyon Modelleri

U-Net, özellikle biyomedikal görüntü segmentasyonu için geliştirilmiş bir model olmasına rağmen, şerit tespiti gibi görüntü işleme görevlerinde de başarıyla kullanılmaktadır. Modelin iki temel bileşeni bulunmaktadır:

- **Encoder (Kodlayıcı) Kısmı:** Girdiyi daha düşük çözünürlüklü bir temsil haline getirmek için ardışık evrişim ve maks pooling katmanlarından oluşur. Bu kısım, giriş görüntüsünün özelliklerini çıkarır.
- **Decoder (Kod Çözücü) Kısmı:** Encoder'dan gelen düşük çözünürlüklü temsili, orijinal çözünürlüğe geri getirmek için up-sampling ve birleştirme (skip connection) katmanlarını kullanır. Bu kısım, detayları geri kazandırır ve segmentasyon haritasını oluşturur.

3.3.1 U-Net'in Şerit Tespiti İçin Kullanımı:

Şerit tespiti, yol üzerindeki şeritlerin doğru şekilde tanımlanmasını gerektirir. Bu görev, görüntü segmentasyonu kapsamında değerlendirilir. U-Net'in şerit tespiti için avantajları:

- **Yüksek Çözünürlüklü Çıktılar:** U-Net, çıktının detay seviyesini koruyarak segmentasyon yapar. Bu, şeritlerin doğru şekilde algılanmasını sağlar.
- **Skip Connections:** Encoder ve decoder katmanları arasındaki bağlantılar, detay kaybını önler ve modelin doğruluğunu artırır.
- **Eğitim Verimliliği:** U-Net, sınırlı miktarda veriyle etkili bir şekilde çalışabilir.
- **U-Net Modelinin Özelleştirilmesi:** Proje kapsamında U-Net, CARLA simülasyonundan elde edilen segmentasyon haritaları ile eğitilecektir. Özelleştirmeler şunları içerebilir:
  - **Veri Genişletme:** Eğitim sırasında veri çeşitliliğini artırmak için döndürme, ölçekleme ve parlaklık ayarlama gibi teknikler kullanılabilir.
  - **Daha Hafif Modeller:** Gerçek zamanlı uygulamalar için U-Net'in daha hafif versiyonları (ör. MobileNet tabanlı U-Net) tercih edilebilir.

  U-Net tabanlı segmentasyon ile şerit tespiti, otonom araçların yolda doğru bir şekilde yönlendirilmesini sağlamak için kritik bir bileşen olarak projede yer alacaktır.

3.4 Python – Programlama Dili

Proje, Python programlama dili kullanılarak gerçekleştirilecektir. Python'un geniş kütüphane desteği ve derin öğrenme araçları bu projede avantaj sağlamaktadır.

3.5 Veri Kümeleri

Projenin başarısı için yüksek kaliteli ve kapsamlı veri kümelerine ihtiyaç vardır. Kullanılacak veri kümeleri şunlardır:

- **CARLA Simülasyon Veri Kümesi:** CARLA, sürücüsüz araç araştırmaları için hazır veri kümeleri sunmaktadır. Bu veri kümeleri, otonom araç senaryolarına uygun nesne anotasyonlarını içerir.
- **Şerit Anotasyonları:** Şerit tespiti için detaylı segmentasyon anotasyonları.
- **Trafik İşareti ve Araç Verileri:** YOLOv8 için uygun biçimde anotasyonlanmış görüntüler.
- **Sensör Verileri:** Lidar, radar ve kamera verileri.
- **Ek Veri Kümeleri:** CARLA dışındaki açık kaynak veri kümeleri, modeli daha çeşitli durumlarda test etmek için kullanılabilir. Örnekler:
  - **COCO Veri Kümesi:** Genel nesne tespiti.
  - **KITTI Veri Kümesi:** Otonom araç araştırmaları için yaygın olarak kullanılan bir veri seti.

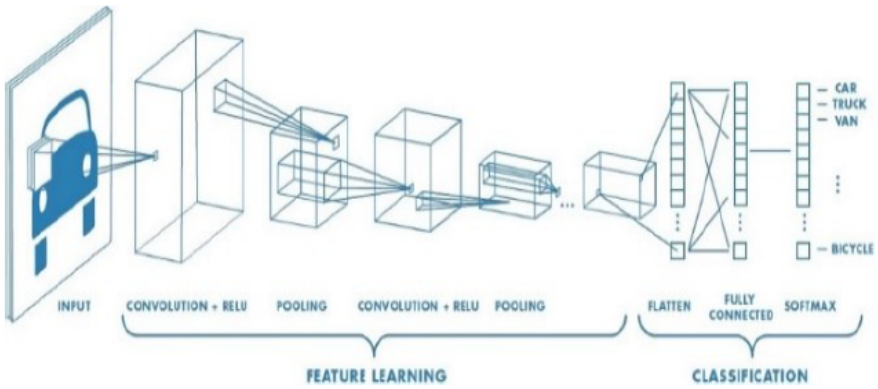
- **Model Geliştirme Süreci:** Proje kapsamında geliştirilecek modeller ve metodolojiler şu şekilde planlanmıştır:
  - **Model Eğitimi:**YOLOv8: Trafik işareti, yaya ve araç tespiti. U-Net: Şerit tespiti. Multitask Modeller: Farklı görevleri bir arada gerçekleştirecek şekilde head ve backbone yapılarının özelleştirilmesi.
  - **Model Testi ve Performans Analizi:** Geliştirilen modeller, CARLA simülasyonu ve gerçek dünya verileri ile test edilerek doğruluk ve performans ölçümleri yapılacaktır.

Bu proje, sürücüsüz araçlar için nesne tanıma ve çevre algılama alanında yeni yöntemler geliştirmeyi amaçlamaktadır. Kullanılacak teknolojiler, veri kümeleri ve simülasyon araçları, gerçek dünya uygulamalarına yönelik yüksek kaliteli algoritmaların geliştirilmesine olanak sağlayacaktır. Proje sonunda, elde edilen sonuçlar analiz edilerek otonom araç teknolojilerine katkıda bulunulacaktır

4 İlgili Çalışmalar / Literatür Araştırması

Bu makalede, otonom sürüşte hızlı ve doğru şerit tespiti için yeni bir yaklaşım sunulmuştur. Geleneksel yöntemlerin yüksek çözünürlüklü görüntülerde hesaplama açısından yetersiz kalmasını çözmek için, yalnızca görüntülerin dilimlenmiş (slice) kısımları işlenmiştir.[10]

Bu çalışma, otonom araçlar için şerit algılama görevini uçtan uca (end-to-end) öğrenmeye yönelik bir örnek segmentasyon (instance segmentation) problemi olarak ele almıştır. LaneNet adlı model, şerit piksellerini ayrıştırmak için ikili segmentasyon (binary segmentation) ve gömülü bir kümeleme (clustering) yaklaşımı kullanır.[11]



Şekil 2: CNN'in mimarisi [1]

CARLA simülatöründe nesne tespiti için kullanılan yöntemlerden biriside R-CNN algorithmasıdır. R-CNN algorithmasında rastgele seçilen yaklaşık iki bin bölge, CNN ile işlenir ve SVM kullanılarak sınıflandırılır.[12] Ancak yavaş çalışıtığı ve çok fazla bellek tüketiği için Fast R-CNN ve Faster R-CNN algorithmaları geliştirilmiştir. Fast R-CNN'de Selective Search ile bölgeler belirlenir ve Softmax yöntemi ile sınıflandırma yapılır. Faster R-CNN'de ise Region Proposal Network (RPN) kullanır. Lakin her ikisi de gerçek zamanlı uygulamalarda yeterince hızlı değildir. Bu yüzden Nesne tesbitini tek bir adımda ve çok hızlı gerçekleştiren SSD (Single Shot MultiBox Detector) algorithması kullanılmıştır.[12]

Bir başka çalışmada CARLA'dan alınan görüntüler ile konvolüsyonel sinir ağı tabanlı nesne tespiti algoritmaları (YOLOv4, CenterNet, Faster R-CNN) kullanılarak, nesne tespit performansı ölçümleri gerçekleştirilmiştir. [12] YOLOv4'de Transfer Learning yöntemi, CenterNet de Anchor-Free Model ve Hourglass ağı, Faster R-CNN de ise Anchor boyutları ve Öğrenme oranında değişiklikler yapılmıştır. Aralarından YOLOv4'ün en iyisi olduğu lakin veri setindeki sorunlar nedeniyle istenilen sonuç bulunamamıştır.[12]

SSD MobileNet V1: Design Specifications	
Initial Learning Rate	0.004
Activation Function	ReLU (Rectified Linear Unit)
Batch Size	10
Regulariser	L2 Regulariser
Epochs	4000
Loss Function	RMSE (Root Mean Squared error)

Tablo 1: SSD MobileNETV1 tasarım özellikleri [6]

Bir başka makalede ise SSD MobileNet, SSD ResNet, Faster R-CNN, Mask R-CNN, YOLO ve CenterNet algorithmaları karşılaştırılmıştır. SSD MobileNet çerçevesi, gömülü cihazlarda ve mobil uygulamalarda verimli çalışmak için tasarlanmıştır. [6] Makalede de SSD MobileNet'in bu özelliği ve tüm sınıflandırma işlemleri sinir ağından yalnızca bir kez geçme özelliği kullanılmıştır. SSD ResNet karmaşık nesne tespiti görevlerinde doğruluğu artırmak için kullanılmıştır. Faster R-CNN için bölge önerisi için öğrenilebilir bir mekanizma olan RPN kullanılmıştır.[6]

Method		# parameters	GPU memory requirement (unit: GByte)	#FLOPs
U-Net		23.7526×10^6	2.547	1.9803×10^9
SegNet		29.4963×10^6	6.647	3.3913×10^9
FCN-8s		134.2777×10^6	1.386	1.6037×10^9
DeepLab V3		17.8307×10^6	4.401	0.9103×10^9
CED-Net	First model	1.3529×10^6	0.978	0.0212×10^9
	Second model	1.353×10^6	3.919	0.8069×10^9
	Third model	1.3529×10^6	0.978	0.0212×10^9
	Fourth model	1.353×10^6	3.919	0.8069×10^9
Modified U-Net		36.9692×10^6	4.613	5.8297×10^9
MTS-CNN	First model	24.9744×10^6	2.236	2.0727×10^9
	Second model	24.9747×10^6	2.244	2.0743×10^9

Tablo 2: CNN performans grafiği [7]

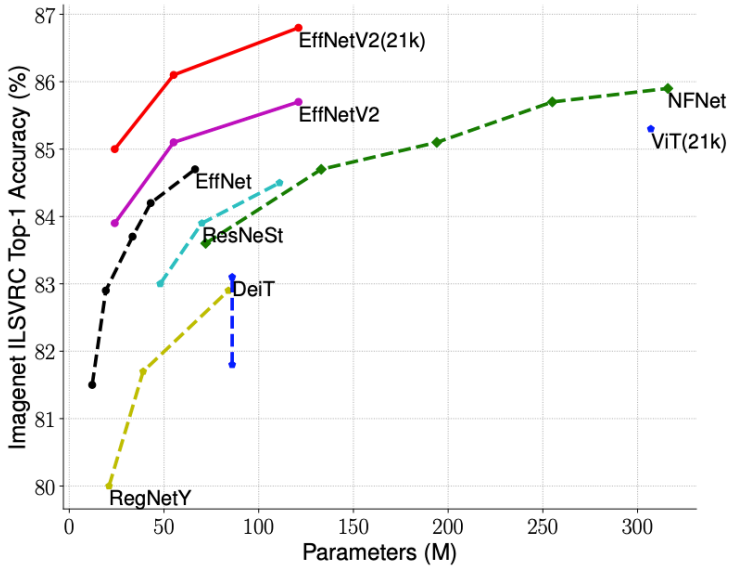
Mask R-CNN, Faster R-CNN ile aynı şekilde çalıştırılmıştır. Lakin bu algorithma da sınıflandırılan nesnelerin yalnızca sınırlarını değil, aynı zamanda piksel düzeyindeki maskeleri de baz alınmıştır. YOLO, nesne tespitini bir regresyon problemi olarak ele alınmıştır. CenterNet'te Anahtar Nokta Algoritması kullanılmıştır. Karşılaştırılma sonunda SSD MobileNet'in en hızlı ve en yüksek doğrulukta tespit yaptığı belirlenmiştir. [6] SSD ResNet SSD MobileNet kadar yüksek olamamıştır. Lakin karmaşık nesneler de daha iyi bir performans göstermiştir. Faster R-CNN Doğruluk açısından başarılıdır, ancak hız açısından yavaştır. Mask R-CNN Piksel bazlı maskeler sağladığı için segmentasyon görevlerinde güçlüdür. Lakin daha fazla işlem gerektirdiğinden çok yavaş çalışmıştır. YOLO orta seviyede doğruluk sunmuştur. CenterNet'te doğruluk ve hız değerleri YOLO ve SSD MobileNet'in gerisinde kalmıştır.[6]

Model	AP	AP_{50}	AP_{75}	AP_M	AP_L
$CARLA^{st}_{train}$	11.7	25.4	10.6	4.8	21.4
$MVD20^{st}_{train}$	1.2	3.8	0.3	0.2	2.3
$CARLA^{st}_{train}$ + $MVD20^{st}_{train}$	16.2	33.8	16.3	7.3	27.7
$MVD120^{st}_{train}$	14.4	35.5	8.4	5.9	25.0
$CARLA^{st}_{train}$ + $MVD120^{st}_{train}$	19.1	39.5	18.2	9.0	32.2
$MVD600^{st}_{train}$	25.9	53.6	21.6	13.2	42.1
$CARLA^{st}_{train}$ + $MVD600^{st}_{train}$	26.3	52.6	23.6	15.1	40.4
$MVD2000^{st}_{train}$	34.4	63.6	32.2	19.6	52.0
$CARLA^{st}_{train}$ + $MVD2000^{st}_{train}$	35.5	63.9	34.2	21.5	53.1
$MVDFull^{st}_{train}$	41.7	71.3	43.4	27.6	58.1
$CARLA^{st}_{train}$ + $MVDFull^{st}_{train}$	40.1	67.9	39.7	24.1	58.7

Tablo 3: Carla ortalama keskinlik tablosu [8]

Bu çalışmada, CARLA simülatörünün sentetik veri üretimi için etkili bir araç olduğu vurgulanmaktadır. Sentetik veri, özellikle nadir görülen, yenilikçi veya genellikle ihmal edilen nesneler için veri setlerinin zenginleştirilmesinde kullanılmıştır. Simülatörde uyg ulanan domain randomization (alan rastgeleleştirme) teknikleri, modelin farklı senaryolara uyum sağlamasına yardımcı olmuştur. [8] Sentetik verilerin özellikle büyük nesneler üzerinde daha yüksek doğruluk sağladığı belirtilmiştir. Bu çalışmada, nesne tespiti için Detectron2 kütüphanesi ve Faster R-CNN kullanılmıştır. Faster R-CNN, ResNet-101 ve FPN kombinasyonu sayesinde, farklı ölçeklerdeki nesneler için yüksek doğruluk ve etkili tespit yetenekleri sergilemiştir. Detectron2, büyük nesneler üzerinde en iyi performansı sağlamıştır.[8]

Segmentasyon ve obje tespiti gibi farklı görevleri birleştirmek, multi-task learning ile modelin paylaşılmış özelliklerden faydalanmasını sağlar ve hesaplama maliyetlerini düşürür. Bu bağlamda, EfficientNetV2, optimize edilmiş compound scaling ve hızlı eğitim stratejileriyle ideal bir backbone olarak öne çıkar. [2] Hem düşük seviyeli (segmentasyon) hem de yüksek seviyeli (obje tespiti) özellikleri aynı anda etkili bir şekilde çıkarması, EfficientNetV2'yi multitask uygulamalar için güçlü ve verimli bir seçenek haline getirir.[2]



Şekil 3: EfficientNetV2 modelinin parametre sayısına göre başarı oranı. [2]

Multi-task görevleri üzerine yapılan bir araştırmada MTS-CNN gibi çoklu görev öğrenimi tabanlı yaklaşımlardan esinlenerek segmentasyon ve algılama doğruluğunu artırmaya yönelik stratejiler önem taşır.[13] MTS-CNN çalışmasında, ekin ve yabani ot segmentasyonu için çoklu görev yaklaşımı benimsenmiş, VGG-16 tabanlı U-Net mimarisiyle tek aşamalı ve iki aşamalı eğitim stratejileri ile daha hassas sonuçlar elde edilmiştir. Buna benzer şekilde, EfficientNetV2'nin güçlü özellik çıkarma yeteneklerinden yararlanarak, şerit ve trafik tabelası gibi farklı nesne türlerinin algılanmasını optimize eden çoklu görev modelleri geliştirilebilir.[13]

EfficientNet modelleri, derin öğrenme tabanlı sınıflandırma algoritmalarında yüksek doğruluk oranlarıyla dikkat çekmekte ve özellikle sağlık gibi kritik alanlarda başarıyla kullanılmaktadır. Literatürde, EfficientNet'in parametre verimliliği ve dikkatlice ölçeklendirilmiş yapısıyla yüksek performans sunduğu gösterilmiştir. [14] Örneğin, yukarıda sunulan çalışmada EfficientNet B0 modeli, çok merkezli bir veri kümesi üzerinde 0,98 doğruluk oranına ulaşarak, akciğer anormalliklerinin sınıflandırılması konusunda üstün bir başarı sergilemiştir. Bu sonuçlar, EfficientNet'in mimari tasarımının görüntü sınıflandırma görevlerinde genelleştirilebilirliğini ve doğruluğunu vurgulamaktadır.[14]

Model	Image Size	Epoch	Training Accuracy (%)	Testing Accuracy (%)
VGGNet19	224*224	30	90.36	90.12
ResNet50	224*224	30	93.59	91.66
ResNet152	224*224	30	94.12	93.91
MobileNetV2	224*224	30	96.34	95.67
DenseNet169	224*224	30	95.59	94.21
EfficientNet B0	224*224	30	97.84	96.13

Tablo 4: CNN modellerin performans karşılaştırıkması. [9]

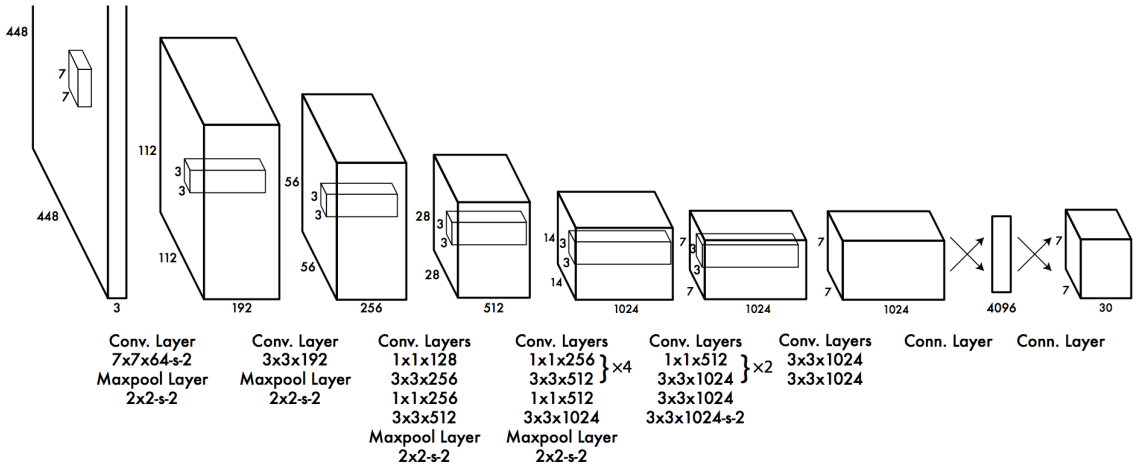
EfficientNet modelleri, tarımsal hastalık tespiti gibi karmaşık görüntü sınıflandırma problemlerinde yüksek doğruluk ve verimlilik sağlayarak önemli bir yer edinmiştir. Literatürde, bu modellerin dikkat mekanizmalarıyla entegre edilmesi, özellikle hassas desenlerin ve görüntü özelliklerinin yakalanmasını optimize ettiği için üstün performans göstermektedir. [9] Örneğin, EfficientNet B0 ile Convolutional Block Attention Module (CBAM) entegrasyonu sayesinde, bu çalışma üç ana buğday mantar hastalığını %98,70 doğruluk oranıyla tespit ederek, mevcut literatürdeki %93 ile %97 arasında değişen doğruluk oranlarını aşmıştır. Ayrıca, ortalama F1 skoru %98, hassasiyet ise %99 olarak rapor edilmiştir.[9]

5 Metodoloji ve Çözüm Teknikleri

5.1 U-Net Tabanlı Şerit Tespiti

U-Net, biyomedikal görüntü segmentasyonu için geliştirilmiş bir derin öğrenme modelidir, bu çalışmada yol şeritlerinin tespiti için uyarlanmıştır. U-Net'in amacı yoldaki şeritleri tespit etmek ve görüntüden segmentasyon yaparak belirgin hale getirmektir. U-Net'in çalışabilmesi için öncelikle modelin görüntüyü giriş olarak alması gerekir daha sonra her pikselin bir sınıfa ait olup olmadığını (örneğin şerit veya arka plan) belirlemek için çalışır. Bu süreç, özellikle yol üzerindeki şerit işaretlerini algılamak için tasarlanmıştır. [15] U-Net'in ne kadar etkin olduğunu öğrenmek için Carla simülatörü tarafından sağlanan farklı senaryolarda kapsamlı bir şekilde test edilmiştir. Test sürecinde, modelin şeritlerin doğru bir şekilde tespit edilmesi ve piksellerin hassas bir şekilde sınıflandırılması üzerinde yoğunlaşılmıştır. Çeşitli hava koşulları (örneğin, güneşli, yağmurlu veya sisli) ve farklı yol tiplerinde (örneğin, düz yollar, virajlı yollar, çok şeritli yollar) modelin performansı değerlendirilmiştir. [15] Sonuçlar, U-Net'in hem görsel doğruluk açısından hem de piksel düzeyinde sınıflandırma başarısında üstün performans gösterdiğini ortaya koymuştur. Model, özellikle karmaşık trafik senaryolarında, şeritlerin yerini doğru bir şekilde belirlemiş ve çevresel faktörlere karşı dayanıklılığını kanıtlamıştır. Bu etkileyici performans, U-Net'in otonom sürüş uygulamalarında güvenilir bir segmentasyon aracı olarak kullanılabileceğini göstermektedir.[15]

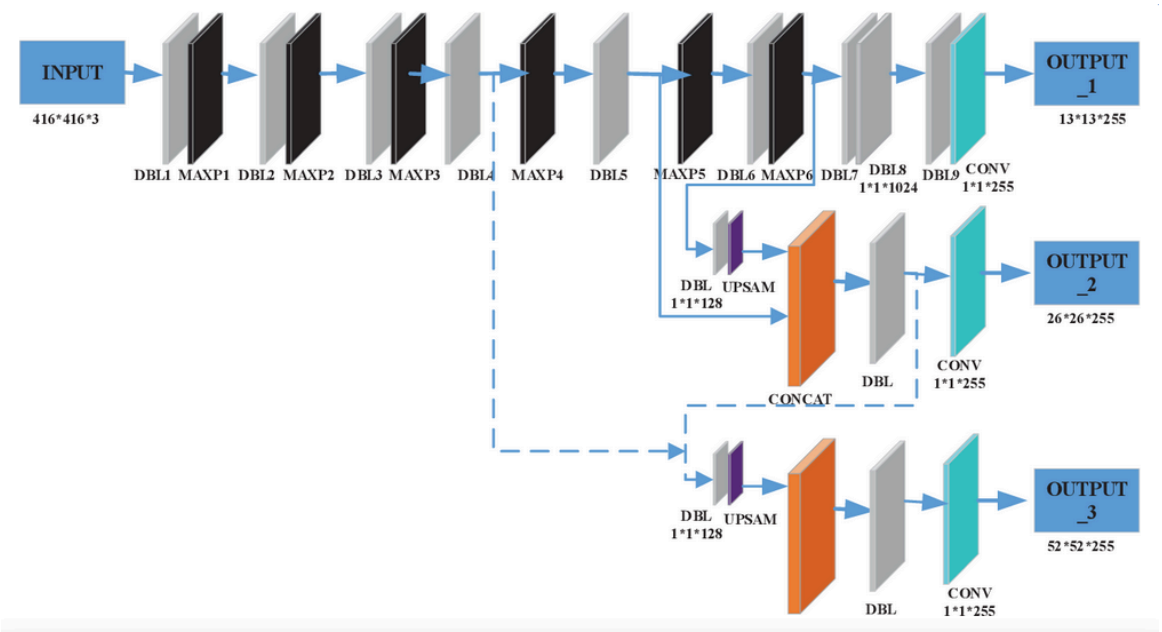
5.2 YOLO Tabanlı Trafik İşareti ve Yaya Tespiti



Şekil 4: Yolo'nun Mimarisi [3]

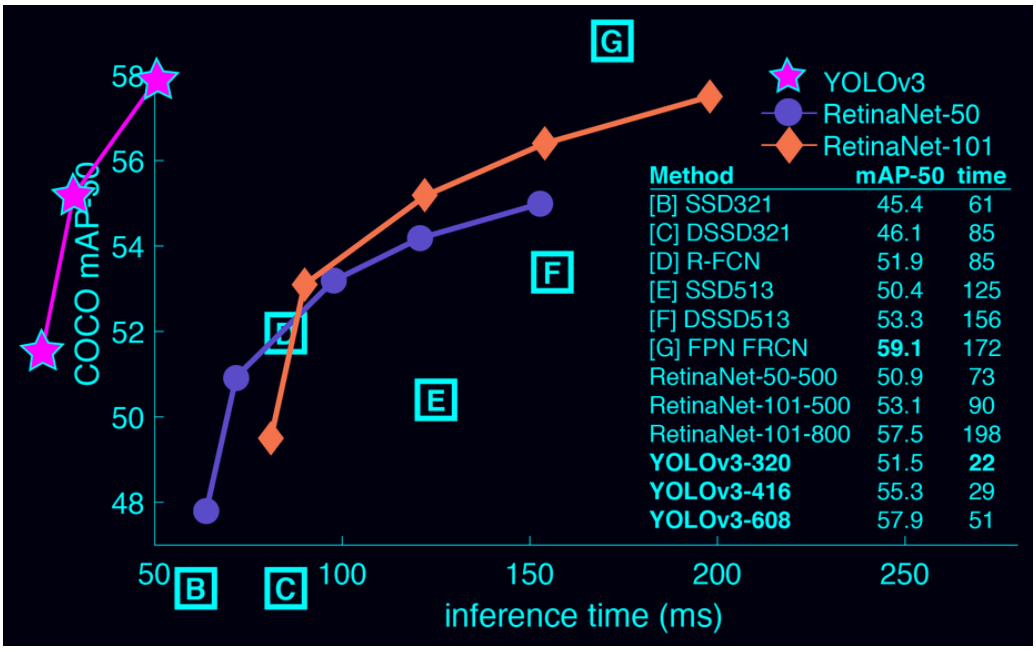
Yolo, gerçek zamanlı nesne tespiti için en iyi algoritmalardan biridir ve diğer tespit yöntemlerine kıyasla hızlı ve doğru sonuçlar sağlar. Yolo açık kaynaklı bir sinir ağı çerçevesi olan Darknet

üzerinde de çalışır. [3] Yolo'nun Yolov3-tiny modeli , Yolov3'ün daha hafif ve hızlı bir versiyonudur. Hızlı olması nedeniyle bu model, özellikle CPU ortamında daha avantajlıdır. CARLA simülatöründe Yolov3-tiny modeli hız tabelalarının (30 km/h, 60 km/h, 90 km/h) tespiti ve tanımlanması için kullanılacaktır.[3]



Şekil 5: Geliştirilmiş YOLOv3-tiny ağ mimarisi [4]

Burada kullanılan Yolov3-tiny modeli, [16] CARLA simülatörü için özelleştirilebilir ve hız tabelası algılama performansını artırmak amacıyla modifiye edilecektir. Standart YOLO v3 modeli, 75 derin evrişim katmanına sahiptir. Bu çalışmada kullanılan hafif versiyonda, katman sayısı 16'ya düşürülerek hem hesaplama yükü azaltılmış hem de simülasyon verimliliği artırılacaktır. YOLO, CARLA'daki hız tabelalarını tanımak için özel bir veri setiyle eğitilmiştir. Eğitim veri seti, CARLA'daki 30 km/s, 60 km/s ve 90 km/s hız tabelalarının farklı açılardan ve koşullardan görüntüler alınmıştır. [16] YOLO'nun kayıp fonksiyonu, hız tabelası algılama performansını optimize etmek için değiştirilecektir. Örneğin, sınır kutularının doğruluğuna daha fazla ağırlık verilecektir. Çalışmamızda, araç önüne monte edilen bir kamera, her beş karede bir çevrenin RGB görüntülerini alacak ve YOLO modeline analiz için iletecektir. YOLO modeli, bu RGB görüntülerdeki hız tabelalarını algılar ve tanımlar. Algılama sonuçları, gerçek zamanlı olarak ekran üzerinde görüntülenecektir. [16]



Şekil 6: Yolo performans grafiği [5]

Aynı zamanda YOLO tabanlı bir Konvolüsyonel Sinir Ağı (CNN) kullanılarak gerçek zamanlı nesne algılama ile trafik işaretlerinin tanınması için kullanılabilir. Bunun yanında, veri toplama, model eğitimi, sensör birleşimi ve test süre çlerini içeren bütünleşik bir yaklaşım benimsenmiştir. Yolov3-tiny modelin de daha karmaşık ve büyük veri kümelerinde eğitim yapmak zorunda kaldığımız durumlarda YOLOv5 modeli kullanılacaktır. [17] Yolov5 otonom araç uygulamalarında kritik olan düşük gecikme süresi sunması, farklı veri kümeleri ve özel uygulamalar için kolayca uyarlanabilir olması ve kaynak kısıtlaması olan cihazlarda etkili bir şekilde çalışabilmesi. YOLOv5 modelini tercih etmemizin sebepleridir. Aynı zamanda hız işaretleri, yayalar, hayvanlar ve çukurlar gibi çeşitli nesneleri algılayacak şekilde özelleştirilecektir. [17]

5.3 Multi-Task Learning Yöntemleri

Multi-task learning (MTL), birden fazla görevi aynı model altında çözerek özellik paylaşımından faydalanmayı, hesaplama maliyetlerini azaltmayı ve model performansını artırmayı hedefler. Bu projede, segmentasyon (şerit tespiti) ve obje tespiti (trafik tabelası tanıma) görevleri için EfficientNetV2, ortak bir backbone olarak kullanılacaktır.

Stage	Operator	Stride	#Channels	#Layers
0	Conv3x3	2	24	1
1	Fused-MBConv1, k3x3	1	24	2
2	Fused-MBConv4, k3x3	2	48	4
3	Fused-MBConv4, k3x3	2	64	4
4	MBConv4, k3x3, SE0.25	2	128	6
5	MBConv6, k3x3, SE0.25	1	160	9
6	MBConv6, k3x3, SE0.25	2	256	15
7	Conv1x1 & Pooling & FC	-	1280	1

Tablo 5: EfficientNetV2 mimarisi. [2]

[2] EfficientNetV2, compound scaling prensibiyle katman derinliği, genişliği ve çözünürlüğü optimize edilmiş bir mimaridir. Özellikle MTL'de hem düşük seviyeli (segmentasyon) hem de yüksek seviyeli (obje tespiti) özellikleri etkili bir şekilde çıkarmasıyla öne çıkar. Tan et al. (2021) tarafından yapılan çalışmalar, EfficientNetV2'nin önceki EfficientNet sürümlerine göre %20-30 daha hızlı eğitim süresi ve daha düşük parametre sayısıyla daha yüksek doğruluk sunduğunu göstermektedir. [2] Bu performans, gerçek zamanlı otonom sürüş senaryolarında ihtiyaç duyulan verimliliği ve doğruluğu sağlaması açısından kritik öneme sahiptir. EfficientNetV2'nin güçlü özellik çıkarımı sayesinde, segmentasyon ve obje tespiti görevleri aynı anda etkili bir şekilde gerçekleştirilebilir, bu da projede hesaplama maliyetini ve model karmaşıklığını önemli ölçüde azaltır. [2]

6 Proje Yönetimi

6.1 İş Paketleri (İP), Görev Dağılımı ve Süreleri

İş Paketi			**Zaman (Aylar)**									
No	**Adı**	**Sorumlu kişi(ler)**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
İP1	Literatür araştırması	Ali Enes Doğan Şerafettin Doruk Sezer Dilara Çelik										
İP2	Segmentasyon modellerinin oluşturulması ve test edilmesi	**Dilara Çelik** Ali Enes Doğan										
İP3	Nesne tespiti modellerinin oluşturulması	**Şerafettin Doruk Sezer** Dilara Çelik										
İP4	Çoklu görev algoritmasının oluşturulması	**Ali Enes Doğan** Şerafettin Doruk Sezer										
İP5	CARLA simülasyonu üzerinde testlerin yapılması	Ali Enes Doğan Şerafettin Doruk Sezer Dilara Çelik										

Tablo 6: İş Zaman Çizelgesi

6.2 Riskler ve Alınacak Tedbirler

IP No	Risk(ler)in Tanımı	Alınacak Tedbir (ler) (B Planı)
1	U-Net tabanlı algoritmanın %80 IOU(Intersection over Union) altında performans vermesi	Segmentasyon R-CNN algoritmasına kullanılacaktır.
2	Yolov3-tiny modelin de daha karmaşık ve büyük veri kümelerinde eğitim yapmakta zorlandığımız zaman.	YOLOv5 veya daha üst YOLO sürümleri kullanılacaktır.
3	EfficientNetV2 modelinin %80 IOU altında performans vermesi	ResNet modeli kullanılacaktır.
4	Eğitim veri setinde yanlış veya eksik anotasyonlar, modelin doğruluğunu olumsuz etkileyebilir.	Anotasyonların manuel olarak kontrol edilecek ve doğrulanacaktır.
5	Algoritmaların gerçek zamanlı olarak düşük gecikme ile çalışamaması veya çok yoğun trafik veya karmaşık sahnelerde işlem sürelerinin uzaması.	Daha hafif ve optimize edilmiş modellerin (ör. YOLOv5n veya SSD MobileNet) kullanılacaktır.

Tablo 7: Risk Analizi ve B Planı

7 Yaygın Etki

- **Trafik Güvenliği:** Trafik işaretleri ve yayaların doğru bir şekilde algılanması sayesinde, araçların çevresine duyarlı bir şekilde hareket etmesi sağlanır. Şerit takibi ve engel algılama özellikleri, araçların yol dışına çıkmasını veya kazalara neden olabilecek hataları minimize eder.
- **Yakıt ve Enerji Verimliliği:** Araçların trafik işaretlerine ve şeritlere uygun hareket etmesiyle daha akıcı bir trafik akışı sağlanır. Bu, yakıt tüketimini azaltır ve karbon ayak izini küçültür.
- **Ulaşım Verimliliği:** Araçların trafik işaretlerine ve yollara duyarlı bir şekilde hareket etmesi, trafik sıkışıklığının azaltılmasına yardımcı olur.
- **Toplu Taşıma Sistemleri:** Projede geliştirilen teknolojiler, akıllı toplu taşıma sistemlerinde kullanılabilir. Örneğin, otobüslerin duraklarda güvenli bir şekilde durmasını sağlayacak otonom sistemler için temel oluşturabilir.
- **Ticari Araçlar:** Yük taşımacılığında otonom kamyonlar için benzer algoritmalar kullanılabilir ve trafik güvenliği artırılabilir.

Kaynakça

[1] A. Saenong, Z. Zainuddin, ve M. Niswar, "Identification of Poultry Reproductive Behavior Using Faster R-CNN with MobileNet V3 Architecture in Traditional Cage Environment", içinde 2023 International Seminar on Intelligent Technology and Its Applications (ISITIA), 2023, ss. 456-461. doi: 10.1109/ISITIA59021.2023.10221017.

[2] M. Tan ve Q. V. Le, "EfficientNetV2: Smaller Models and Faster Training", CoRR, 2021, [Çevrimiçi]. Erişim adresi: https://arxiv.org/abs/2104.00298

[3] M. Juanola, "Speed traffic sign detection on the CARLA simulator using YOLO", 2019. [Çevrimiçi]. Erişim adresi: https://api.semanticscholar.org/CorpusID:209090264

[4] H. Gong, H. Li, K. Xu, ve Y. Zhang, "Object Detection Based on Improved YOLOv3-tiny", 2019 Chinese Automation Congress (CAC), ss. 3240-3245, 2019, [Çevrimiçi]. Erişim adresi: https://api.semanticscholar.org/CorpusID:211207232

[5] J. Redmon ve A. Farhadi, "YOLOv3: An Incremental Improvement", arXiv, 2018.

[6] D. Niranjan, B. C. VinayKarthik, ve Mohana, "Deep Learning based Object Detection Model for Autonomous Driving Research using CARLA Simulator", içinde 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC), 2021, ss. 1251-1258. doi: 10.1109/ICOSEC51865.2021.9591747.

[7] N. Jain, S. Yerragolla, T. Guha, ve Mohana, "Performance Analysis of Object Detection and Tracking Algorithms for Traffic Surveillance Applications using Neural Networks", 2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), ss. 690-696, 2019, [Çevrimiçi]. Erişim adresi: https://api.semanticscholar.org/CorpusID:212703964

[8] T. Bu, X. Zhang, C. Mertz, ve J. M. Dolan, "CARLA Simulated Data for Rare Road Object Detection", içinde 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), 2021, ss. 2794-2801. doi: 10.1109/ITSC48978.2021.9564932.

[9] S. Nigam, R. Jain, V. K. Singh, S. Marwaha, A. Arora, ve S. Jain, "EfficientNet architecture and attention mechanism-based wheat disease identification model", Procedia Computer Science, c. 235, ss. 383-393, 2024, doi: 10.1016/j.procs.2024.04.038.

[10] J.-M. Guo ve H. Markoni, "Deep Learning Based Lane Line Detection and Segmentation Using Slice Image Feature", içinde 2021 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), 2021, ss. 1-2. doi: 10.1109/ISPACS51563.2021.9651012.

[11] D. Neven, B. D. Brabandere, S. Georgoulis, M. Proesmans, ve L. V. Gool, "Towards End-to-End Lane Detection: an Instance Segmentation Approach", CoRR, 2018, [Çevrimiçi]. Erişim adresi: http://arxiv.org/abs/1802.05591

[12] W. Gao, J. Tang, ve T. Wang, "An Object Detection Research Method Based on CARLA Simulation", içinde The 2021 2nd International Conference on Internet of Things, Artificial Intelligence and Mechanical Automation (IoTAIMA 2021), Hangzhou, China: IOP Publishing Ltd, May. 2021, s. 12163. doi: 10.1088/1742-6596/1948/1/012163.

[13] Y. H. Kim ve K. R. Park, "MTS-CNN: Multi-task semantic segmentation-convolutional neural network for detecting crops and weeds", Computers and Electronics in Agriculture, c. 199, s. 107146, Ağu. 2022, doi: 10.1016/j.compag.2022.107146.

[14] K. Kansal, T. B. Chandra, ve A. Singh, "ResNet-50 vs. EfficientNet-B0: Multi-Centric Classification of Various Lung Abnormalities Using Deep Learning", Procedia Computer Science, c. 235, ss. 70-80, 2024, doi: 10.1016/j.procs.2024.04.007.

[15] D. Niranjan, B. C. VinayKarthik, ve Mohana, "Deep Learning based Object Detection Model for Autonomous Driving Research using CARLA Simulator", içinde 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC), 2021, ss. 1251-1258. doi: 10.1109/ICOSEC51865.2021.9591747.

[16] Y. Valeja, S. Pathare, D. Patel, ve M. Pawar, "Traffic Sign Detection using Clara and Yolo in Python", içinde 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS), 2021, ss. 367-371. doi: 10.1109/ICACCS51430.2021.9442065.

[17] S. Sarumathi, M. Sabir, M. Suhail, M. Umarulla, ve M. Yousuf, "Enhancing Transportation Safety with YOLO-Based CNN Autonomous Vehicles", içinde 2024 International Conference on Electronics, Computing, Communication and Control Technology (ICECCC), 2024, ss. 1-8. doi: 10.1109/ICECCC61767.2024.10593914.