T.C.
MARMARA ÜNİVERSİTESİ

TEKNOLOJİ FAKÜLTESİ

MEKATRONİK MÜHENDİSLİĞİ BÖLÜMÜ

GÖRME ENGELLİ BİREYLER İÇİN NESNE TESPİTİ VE DERİNLİK TAHMİNİ

ALİ OSMAN KAYA, İLKNUR KARACA

ÖĞR. GÖR. DR. GAZİ AKGÜN

İSTANBUL, 2021

T.C.
MARMARA ÜNİVERSİTESİ

TEKNOLOJİ FAKÜLTESİ

MEKATRONİK MÜHENDİSLİĞİ BÖLÜMÜ

GÖRME ENGELLİ BİREYLER İÇİN NESNE TESPİTİ VE DERİNLİK TAHMİNİ

ALİ OSMAN KAYA, İLKNUR KARACA
170217052, 170217032

ÖĞR. GÖR. DR. GAZİ AKGÜN

İSTANBUL, 2021

MARMARA ÜNİVERSİTESİ
TEKNOLOJİ FAKÜLTESİ
MEKATRONİK MÜHENDİSLİĞİ BÖLÜMÜ

Marmara Üniversitesi Teknoloji Fakültesi Mekatronik Mühendisliği Öğrencileri Ali Osman KAYA ve İlknur KARACA'nın "Gerçek Zamanlı Nesne Tespiti ve Derinlik Algılama" başlıklı bitirme projesi çalışması 01/07/2021 tarihinde sunulmuş ve jüri üyeleri tarafından başarılı bulunmuştur.

Jüri Üyeleri

Marmara Üniversitesi Öğr. Gör. Dr. Gazi AKGÜN (Danışman).....(İMZA)……

…… (Üye)
Marmara Üniversitesi ..... (İMZA)……

…….. (Üye)
Marmara Üniversitesi ..... (İMZA)……

TEŞEKKÜR

Çalışmamızın her aşamasında bize yol gösteren, tüm bilgi birikimini paylaşıp her türlü desteği veren, yardım ve rehberliğini esirgemeyen saygıdeğer danışmanımız Öğr. Gör. Dr. Gazi AKGÜN'e, lisans eğitimimiz süresince katkılarından ötürü tüm bölüm hocalarımıza teşekkürlerimizi sunarız.

Temmuz, 2021 Ali Osman KAYA, İlknur KARACA

ÖZET

Yapay zekâ, günümüze ulaşana kadar pek çok aşamadan geçmiş ve gelişmiştir. Günümüzde ise birçok alanda kullanımı mümkün olan ve işlerimizi kolaylaştıran bir durumdadır. Bu alanlardan birisi de "Bilgisayarlı Görü" anlamına gelen "Computer Vision" alanıdır. Girdi görüntü üzerinden özellik çıkarımına dayalı olan bu alanda, görüntülerde nesne tespiti veya sınıflandırması gibi işlemler mümkün hâle gelmiştir.

Dünya üzerinde görme bozukluğu yaşayan yaklaşık 253 milyon kişi bulunmaktadır. 36 milyonu ise görme engelli bireyleri kapsamaktadır. Projemizin, yapay zekanın "Computer Vision" alanını kullanarak bu insanlara yardımcı olacağı düşünülmektedir.

İnsanlar, günlük aktivitelerini gerçekleştirebilmeleri için gözleri ile nesneleri tespit etmesi, sınıflandırması ve derinliğini tahmin etmesi gerekmektedir. Projemizde bu gereksinimleri göz önünde tutarak nesne tespiti için en iyi modellerden biri olan "YOLOv5" ve derinlik için oldukça başarılı ve hızlı olan "PyD-Net"i birleştirerek kullanılmasına karar verilmiştir. Aynı zamanda tespit edilen nesnelerin kullanıcıya bildirilmesi için metinden sese çevrim yapan "Google Text-To-Speech" kullanılmıştır.

Anahtar kelimeler: Yapay Zekâ, Gerçek Zamanlı Nesne Tespiti, Gerçek Zamanlı Derinlik Tahmini, Derin Öğrenme, YOLOv5, PyDnet, Metinden Sese Çevrim.

Temmuz, 2021 Ali Osman KAYA, İlknur KARACA

ABSTRACT

Artificial intelligence has gone through many stages and developed until it reaches the present day. Today, it is in a situation that can be used in many areas and makes our work easier. One of these fields is "Computer Vision". In this field, which is based on feature extraction from the input image, operations such as object detection or classification in images have become possible.

There are approximately 253 million people in the world who have visual impairment. 36 million of them include visually impaired individuals. It is thought that our project will help these people by using the "Computer Vision" field of artificial intelligence.

Humans need to detect, classify and estimate the depth of objects with their eyes in order to carry out their daily activities. Considering these requirements in our project, it was decided to combine "YOLOv5", one of the best models for object detection, and "PyD-Net", which is very successful and fast for depth estimate. At the same time, "Google Text-To-Speech", which converts from text to voice, is used to notify the user of the detected objects.

Keywords: Artificial Intelligence, Real-Time Object Detection,Real-Time Depth Estimation, Deep Learning, YOLOv5, PyDnet, Text to Speech.

Temmuz, 2021 Ali Osman KAYA, İlknur KARACA

İÇİNDEKİLER

SEMBOLLER LİSTESİ

x1 : input
TP : Gerçek Pozitif
FP : Yanlış Pozitif
FN : Yanlış Nagatif
Σ : Toplam
ϵ: Kapsama
interp : İnterpolasyon

KISALTMALAR LİSTESİ

IBM : International Business Machines
YOLO : You Look Only Once
YOLOv5 : You Look Only Once version 5
CNN : Convolutional Neural Networks
COVID-19 : CoronaVirus Disease - 19
2D : 2 Dimensional
3D : 3 Dimensional
CPU : Central Processing Unit
PyD-NET : Python Dynamic Module Networks
YSA : Yapay Sinir Ağları
R-CNN : Region Based Convolutional Neural Networks
F-CNN : Faster Convolutional Neural Networks
YOLOv4 : You Look Only Once version 4
FPS : Frame Per Second
MB : Megabyte
CSP Network : Cross Stage Partial Network
SSP Block : Spatial Pyramid Pooling Block
PANet : Path Aggregation Network
GIoU : Generalized Intersection of Union
YOLOv3 : You Look Only Once version 3
COCO : Common Object in Context
FPN : Future Pyramid Network
IoU : Intersection over Union
AP : Average Precision
mAP : ;Mean Average Precision
GPU : Graphics Processing Unit
gTTS : Google Text-To-Speech
ReLU : Rectified Linear Unit

ŞEKİL LİSTESİ

BÖLÜM 1: GİRİŞ

Günümüzde gelişen teknoloji ile birlikte oluşan veri miktarı ivmelenerek artmıştır. Bunun nedenlerinden biri günlük yaşantımızda kullandığımız cihaz sayısının artmasıdır. Küçülen mikroişlemci ve sensör yapısı sayesinde her cihazda birden fazla sensör kullanılabilmekte ve bu sensörlerden üretilen verileri işleyebilecek mikroişlemciler de bulunabilmektedir. Bir diğer neden ise internet kullanımının artması diyebiliriz. Özellikle sosyal medya kullanıcı sayısının artması ve her kullanıcının aktif olup gün içinde paylaşımlar yapması ile internet üzerindeki veri miktarı da artmıştır.

Yapay zekâ ise temelde verilerden tahmin çıkarabilme üzerine dayanmaktadır. Bu bağlamda verinin arttığı günümüzde yapay zekanın gelişimi ve tahmin doğruluğu da artmıştır. Özellikle son dönemde popülerleşen "Derin Öğrenme (Deep Learning)", yapay zekanın bir alt alanı olup büyük miktarda ses, görüntü veya yazı verisi kullanılarak çalışan bir uygulama yapısına sahiptir.

Dünya üzerinde görme bozukluğu yaşayan yaklaşık 253 milyon insan bulunmaktadır. 36 milyonu ise görme engelli bireyleri kapsamaktadır. Bu bireyler günlük aktivitelerini gerçekleştirmek için yardımcı ekipmanlara ihtiyaç duyabilmektedir. Beyaz baston veya rehber köpekler örnek verilebilir. Bu ekipmanlar temelde bireylerin engellere çarpamadan yürüyebilmesini amaçlamaktadır. Projemiz ise, engelli bireylerin görüş açısında olan nesnelerin tespitini ve uzaklıklarının tahminini içermekte, aynı zamanda bu bilgileri bireye sesli biçimde aktarmayı amaçlamaktadır.

Derin öğrenmenin alt alanı olan ve görüntü verileri ile çalışan "Bilgisayarlı Görü (Computer Vision)" algoritmalarının birleştirilmesine dayanan projemizde nesne tespiti için "YOLOv5" ve derinlik tahmini için "PyD-Net" kullanılmıştır. Bu iki algoritmanın birleştirilmesi ile gözümüzün yaptığı derinlik algılaması ve nesne tespiti işlemleri mümkün hâle gelmiştir. Böylece bireylerin engellerinin büyük ölçüde aşılacağı düşünülmektedir.

1.1.Yapay Zekâ Nedir?

1955 yılında yapay zekâ alanında öncü isimlerden biri olan John McCarthy, "Yapay zekânın amacı makineleri, onların zekâları varmış gibi geliştirmektir." diyerek kabaca ilk tanımlardan birini yapmıştır.[1] Gerçekten de günümüzde geliştirilen birçok makine neredeyse insan zekâsına denk, hatta bazı durumlarda ondan daha hızlı ve etkili olabilmektedir. Bu sebeple birçok alanda kullanımı mevcuttur. Sağlık, ulaşım, güvenlik, finans, mühendislik vb. alanlar örnek verilebilir.

Tarihçe olarak baktığımızda ilk olarak 1931 yılında Kurt Gödel, birinci dereceden yüklem mantığında tüm gerçek ifadelerin türetilebilir olduğunu göstermiştir ve böylece yapay zekâda ilk basamağı atmıştır. Ardından 1937'de Alan Turing, bu makinelerin belirli sınırları olacağını açıklamıştır. Bu bağlamda Turing testini geliştirerek bir kriter belirlemiştir. 1943'te ise McCulloch, Pitts ve Hebb, nörobilim sonuçlarına dayanan sinir ağlarının ilk matematiksel modelini çıkarmıştır. Günümüzde yapay zekânın geldiği son noktalardan biri ise, 2016'da ünlü Go şampiyonu Lee Sedol'ın Google Deepmind ekibi tarafından geliştirilen AlphaGo programı tarafından yenilmesidir.[1]

Yıllar içinde yapay zekânın gelişimi ivmelenerek artmıştır. Bu artışın nedenlerinden birisi "Big Data" olarak adlandırılan, verinin büyük bir hacme sahip olduğunu ifade eden bu kavram ile geliştirilen modellerin doğruluk oranlarının artmasıdır. Bu veri hacminin sebebi ise dünya çapında kullanımı artan "Internet of Things" uygulamalarıdır.[2] Bu uygulamalardan birisi ise cep telefonları diyebiliriz. Mikrofon, kamera ve GPS gibi birçok sensör barındıran bu cihazların dünya çapında kullanımı bulunması ciddi manada bir veri kitlesi oluşturmaktadır.[3]

Doğruluk oranlarının artmasıyla birlikte yapay zekâda makine-insan etkileşimi de aynı oranda artmıştır. Örnek olarak son dönemlerde otomotiv endüstrisinde kullanıma açılan otonom sürüş özelliğini verebiliriz. Günümüz taşımacılığına yön vereceği düşünülen bu özellik, "Lidar" gibi çevre hakkında bilgi veren birçok sensörle birlikte çalışmaktadır. Bir başka örneği ise sağlık alanında verebiliriz. Sağlık asistanlığı, kanser tespiti ve yeni ilaçlar geliştirmek gibi birçok konuda yapay zekâ kullanılmaktadır. Bu alanda geliştirilmiş en ünlü yapay zekâlardan birisi IBM Watson robotudur. IBM ekibinin büyük bir veri ile besleyerek geliştirdiği bu robotun ün kazanması ile sağlık alanında yapay zekâ algoritmaları analiz ve işleme için kullanılmaktadır.[2]

1.2.Nesne Tespiti

Nesne tespiti bir girdiğin içinde barındırdığı nesnelerin tespit edilip etiketlenmesi üzerine kurulan bir yöntemdir. Nesne tespiti ve nesne tanıma için farklı algoritmalar ve yöntemler geliştirilmiştir. Dijital görüntülerdeki nesnelerin hızlı tespitini etkin biçimde gerçekleştiren ilk algoritma Viola Jones [4] olmuştur. Fakat yıllar boyunca gelişen teknoloji sayesinde yeni teknolojiler ile daha fazla doğruluk oranı oluşturularak yeni yöntemler geliştirildi.

Projede kullanılan YOLO algoritması tek adımlı bir nesne dedektörüdür. Algoritma yayınlandığı günden bu yana birçok versiyonu çıkmıştır. [4] YOLOV5 olarak bilinen son algoritma geniş kullanım alanına sahiptir.

YOLOV5 giriş görüntülerinden çıkarım yaparak karar verir. Alt yapısında bulunan veri setini kullanarak resim, video ve kameradan aldığı girdilerden nesneyi tespit edip etiketler. Bu için ise derin öğrenme algoritmalarını kullanılarak çözümler bulunur. [5]

1.3.Derinlik Tahmini

Derinlik tahmini, uzun zamandır üzerine uğraşılan bir konu hâline gelmiştir. Öyle ki her sene dünyanın sayılı şirketlerinden olan Microsoft, Google ve Facebook'un araştırma ekiplerinin yanında birçok saygın üniversitenin de bu alan ile ilgili yeni araştırmalar ve uygulamalar sunduğu görülmektedir. Robotik, otonom navigasyon, artırılmış gerçeklik ve daha birçok uygulama alanı bulması bu alan üzerindeki çalışmaların başlıca nedenlerinden diyebiliriz.

Derinlik için geliştirilmiş birçok sensör bulunmaktadır. Örnek olarak LIDAR ve Kinect verilebilir. Bazı durumlar için hâlâ etkili olan bu sensörler gün geçtikçe popülerliğini pasif sensörler olan "Binocular/Multi-View stereo", "Structure from motion" ve "monocular depth" sensörlerine bırakmıştır. Maliyet, boyut ve ağırlık ile de aktif sensörlerden daha uygundurlar. Son yıllarda ise Computer Vision alanındaki gelişmeler doğrultusunda derinlik tahmininde CNN kullanımı ile daha iyi bir performans göstermiştir.[6]

1.4.Literatür Özeti

1.4.1.Nesne Tespiti

Nesne tespiti çok amaçlı bir algoritma olduğu için hayatın her alanında kullanıma girmiştir. Gerek YOLO gerek başka algoritmalar ile farklı alanlarda işlevlerine özgü farklı tanımlar yer almaktadır.

YOLOv5 'in kullanıldığı bir araştırmada, zehirli mantar analizi yapılmıştır. En zehirli 8 mantar türünü içeren bir veri seti oluşturulup, sağlık alanında bir çalışma yapılmıştır. [4]

Bir başka araştırmada ise COVID-19 üzerine bir sağlık çalışması yapılmıştır. İnsan Yüzünün sınırlarını algılayarak uygun maskenin kullanımını analiz eder. [7]

Yine bir başka çalışmada da son teknoloji 5. nesil YOLO algoritması kullanılarak nesne tespiti üzerinde durulmuştur. Buğday tanıma çalışması yapılmıştır. [8]

Kaktüs hastalıklarını tespit etmek için de yine YOLOv5 algoritması kullanılarak çalışmalar yapılmıştır. Hastalıklar 5 kategori üzerinde sınıflandırıp, tedavi ve önlemeye yönelik girişimler amaçlanmıştır. [9]

1.4.2.Derinlik Tahmini

Derinlik tahmini için literatürde birçok çalışma mevcuttur ve her sene başka yayınlar ile de genişlemektedir. Yapılan araştırmalardan biri, elde taşınabilen bir cihaz ile görme engelli bireylere yardımcı olmayı amaçlamıştır. Bu cihaz çevresini algılayıp sesli mesaj ile uyarı verebilmektedir. Çevre algılama için görüntü işleme kullanmakta ve objelerle aradaki mesafeyi sesli olarak ifade etmektedir. Aynı zamanda nesnenin ismini de söyleyebilmektedir. Donanım olarak ultrasonik sensör, Arduino Uno, Raspberry Pi, Pi kamera gibi araçları kullanmaktadır. Gerçek zamanlı nesne algılaması 3 aşamada gerçekleşmektedir: mesafe hesaplama, nesne tanıma ve sesli çıkış.[10]

Bir başka araştırma, yardımcı köpekler ve bastonları destekleyecek şekilde geliştirilmiştir. Sistem, iki stereo kameradan ve çevreyi algılama için taşınabilir bir bilgisayardan oluşmaktadır. Amaç, statik ve dinamik nesneleri tanımak ve akustik sinyallere çevirmektir. Bu sinyaller stereophonic kulaklıklar ile kullanıcıya çevre hakkında bilgi vermeyi sağlamaktadır. [11]

Tek kamera ile çalışan derinlik tahmini çalışmalarına örnek olarak Google Brain ekibinin 2018 yılında yayınladığı “vid2depth” isimli çalışmadır. Denetimsiz öğrenme ile çalışan bu model, öğrenmeyi derinlik ve ego-motion ile yapmaktadır. Aynı zamanda kayıp olarak 2D yerine 3D kayıp kullanarak yerel çalışmak yerine tüm sahne ve sahnenin geometrisinde çalışabilmektedir.[12]

Bir başka örneği ise CPU üzerinde çalışabilecek kadar hafif ve doğruluğu da yüksek olan PyD-Net isimli modeldir. Denetimsiz öğrenme ile derinlik tahmini sunan bu çalışma Raspberry Pi gibi gömülü sistemlerde bile çalışabilmektedir. Piramit şeklinde bir mimariye sahip olan bu modelde birden fazla küçük decoder bulunmakta ve her biri ayrı çözünürlükte çalışmaktadır. [7]

1.5.Amaç

Görme engelli bireyler, günlük aktiviteleri için dışarıdan yardıma ihtiyaç duyabilmektedir. Örnek olarak herhangi bir yere gitmek istediklerinde beyaz baston veya rehber köpek yardımı ile hareket etmektedirler. Çevrelerini algılamada diğer duyu organlarını kullansalar da yeterli olamayabilmektedir. Günümüzde yapay zekânın bir alt dalı olan "Computer Vision" alanının gelişmesi birçok yeniliği de beraberinde getirmiştir. Projemizde bu yeniliklerden derinlik algılama ve nesne tespitini kullanarak bir kamera ile görme engelli bireylerin çevresini daha iyi algılayabilmesini amaçlıyoruz. Herhangi bir kamera vasıtasıyla, örneğin cep telefonu kamerası, modelin görüntü üzerinde derinlik algılaması ve nesne tespitini yapması sonucu sesli bir çıkış olarak bireye iletmesi amaçlanmıştır. Gelişme aşamasında olan modelimiz bilgisayar kamerasında bahsettiğimiz özellikler ile çalışabilmektedir.

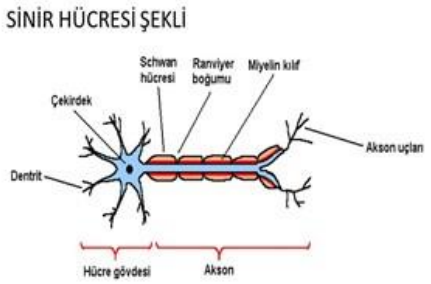
1.6.Orijinal Katkı

Literatür taraması sonucu derinlik ve nesne tespitinin incelenip birçok çalışmanın mevcut olduğu ve çok çeşitli alanlarda kullanılarak başarıya ulaştığı görülmüştür. Fakat iki işlemin PyDnet ve YOLOV5 kullanılıp kameradan alınan görüntüde hem nesne tanıma hem de derinlik konusunun işlenmesinin olmadığı gözlemlenmiştir. İki farklı durumun işlendiği sistemimiz kamera verilerini kullanıp gerçek zamanlı nesne algılamanın sağlanıp ve aynı zamanda da gerçek zamanlı derinlik algılama süzgecinden geçmesine dayanmaktadır. Nesneyi ve derinliği algılamasının yanı sıra sesli olarak nesnenin adını da söylemektedir. Yapılan sistemimiz görme engelli bireylerin hayatını kolaylaştırıp nesneyi seslendirmesi ile de bireyin hayattan kopmamasını sağlayacaktır. Görme engelli bireylerin hareketlerinin sınırı azalacak ve kişinin hayatına olumlu katkılar yaparak hayata daha fazla dahil olmasını sağlayacaktır.

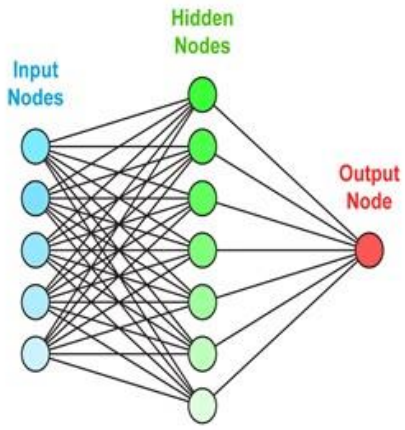
BÖLÜM 2: MATERYAL VE METOT

2.1.Yapay Sinir Ağları ve CNN

Yapay zekanın temeli insan beynine dayanmaktadır. Yapay sinir ağlarının (YSA) anlaşılabilmesi için sinir sistemiyle birlikte değerlendirilip incelenmesi gerekmektedir. İnsan sinir sisteminde dentritten algılanan uyarı elektriksel olarak taşınıp akson ucundan bir tepki oluşturulması mantığına dayanmaktadır.

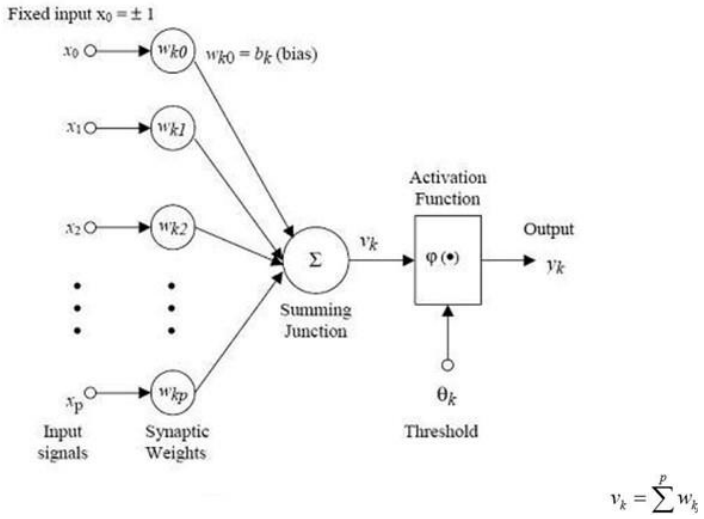


Şekil 1.Sinir Hücresi [13]



Şekil 2.Yapay Sinir Ağları [14]

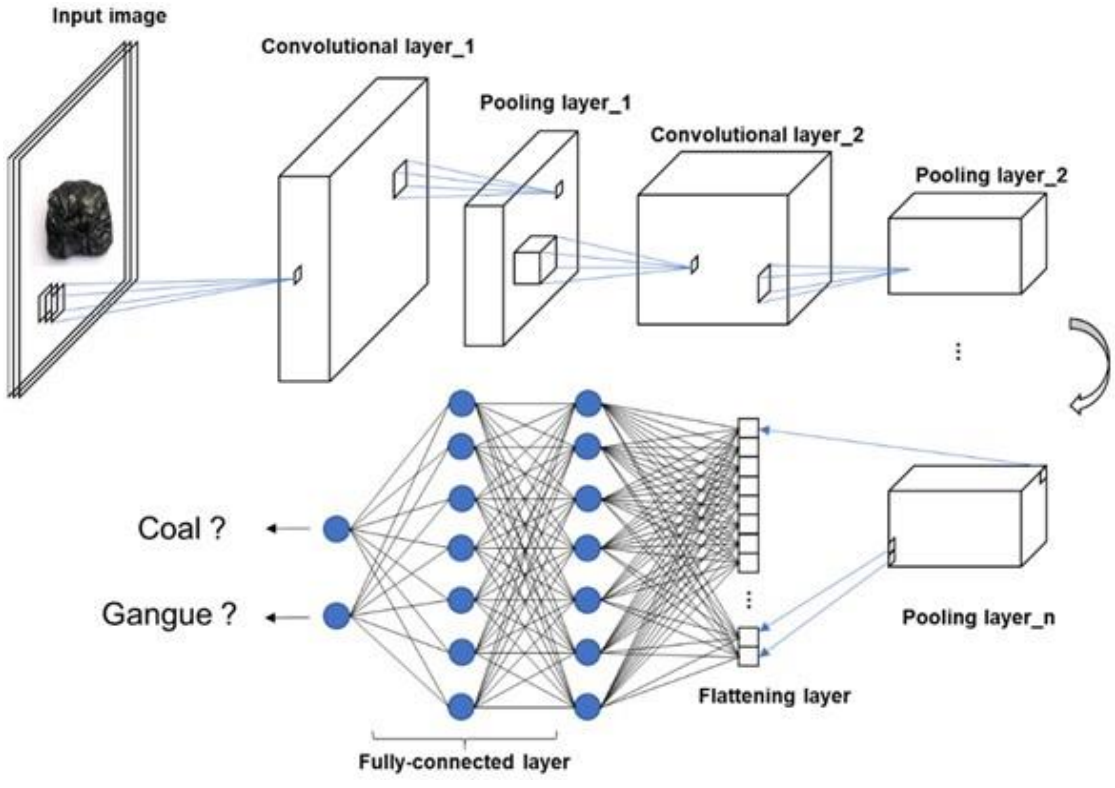
Yapay sinir ağları ile ulaşılmak istenen duruma göre eğitim gerçekleştirilip, insan beyninin anlama, değerlendirme ve uygulama sisteminin klonlanması mantığına dayanmaktadır. Yapay sinir ağları giriş katmanı, gizli (ara) katman ve çıkış katmanı olmak üzere üç ana katmandan oluşmaktadır. Alınan bilgiler girdi katmanından ağa iletip, gizli katmanında işlemlere tabii tutularak çıkış katmanına ulaşır ve çıktı oluşur.



Şekil 3.Yapay Sinir Ağları Matematiksel Modeli [15]

Şekilde bir sinir hücresinin matematiksel modeli görülmektedir. Bu model sinir sisteminde olduğu gibi input (x1) verisini alarak işlemler yaptırır ve çıkışa uygun hale getirir.

Yapay sinir sinir ağlarında katmanların iletişim şekline göre birçok çeşidi bulunmaktadır ve görüntüyü farklı katmanlarda analiz eden Evrişimli Sinir Ağları (CNN) da bunlardan biridir.



Şekil 4. CNN Mimari Yapı [16]

Bir başka ifadeyle Konvolüsyonel Sinir Ağları (CNN) Çok Katmanlı Algılayıcıların (MLP) bir türüdür.[20] İleri Yönlü Sinir Ağı olarak bilinen CNN algoritması hayvanların görme merkezinden esinlenilmiştir. Bu nedenle matematiksel konvolüsyon işlemi yapılıp bir nöronun uyarı bölgesinden uyaranlara verdiği cevap olarak işleme girmiştir. [16]

CNN, bir ya da daha fazla konvolüsyonel katman, altörnekleme (subsampling) katmanı ve standart çok katmanlı bir sinir ağı gibi tamamen bağlı katmandan oluşur.[16]

CNN algoritması görüntü işleme alanında tercih edilmiş, başarılı sonuçlar elde edilmiştir ve aynı zamanda da eğitimi hızlı bir sistemdir. R-CNN, F- CNN, EffiencientDet, ATTS, ASSF, CenterMask, YOLO olmak üzere CNN çeşitleri mevcuttur.

2.2.YOLOv5

Son yıllarda adı sıkça duyulan konvolüsyonel sinir ağlarını kullanarak çalışan YOLO algoritması, gerçek zamanlı nesne tespiti yapabilmesi ve diğer nesne tespit algoritmalarına göre genel ortalama kesinlik (mAP) değeri yüksekliği sayesinde bu kadar popülerdir.

YOLO birçok nesneyi algılayıp nesnelere sınırlar çizerek isimlerini etiketlemektedir. YOLO algoritması resmi bir kerede nöral ağdan geçirerek resimdeki tüm koordinatları ve nesne sınıfını tahmin edebilmesi sayesinde hız konusunda da diğer algoritmaları geride bırakmaktadır. Başka bir deyişle hızının sebebi tek regresyon işleminin uygulanışıdır.

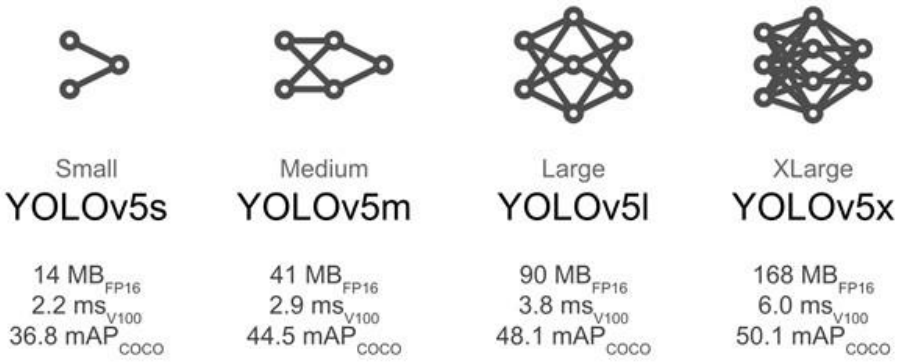
YOLOv5'i diğer versiyonlardan, özellikle YOLOv4'ten, ayıran özellikler incelendiğinde 5. versiyonun daha avantajlı olduğu görülmüştür. YOLOv5 önceki sürümlerden farklı olarak Python diliyle yazılmıştır. Bu durum ise IOT cihazlarında kurulum ve entegrasyon kolaylığı sağlamaktadır.

Ayrıca Pytorch topluluğu DarkNet topluluğundan daha büyük olduğundan daha fazla büyüme potansiyeline sahiptir.

YOLOv4'ün FPS değeri 50 iken YOLOv5'in FPS değeri 140'tır. Ayrıca boyutu da küçük ve 27 MB'dir.[17]

YOLOv5 özetlenecek olursa omurga, boyun, baş kısmından oluşur. Omurga odak yapısı ve CSP ağı, boyun SSP bloğu ve PANet, baş kısmı ise GIoU kullanan YOLOv3 kafasından oluşmaktadır. [17]

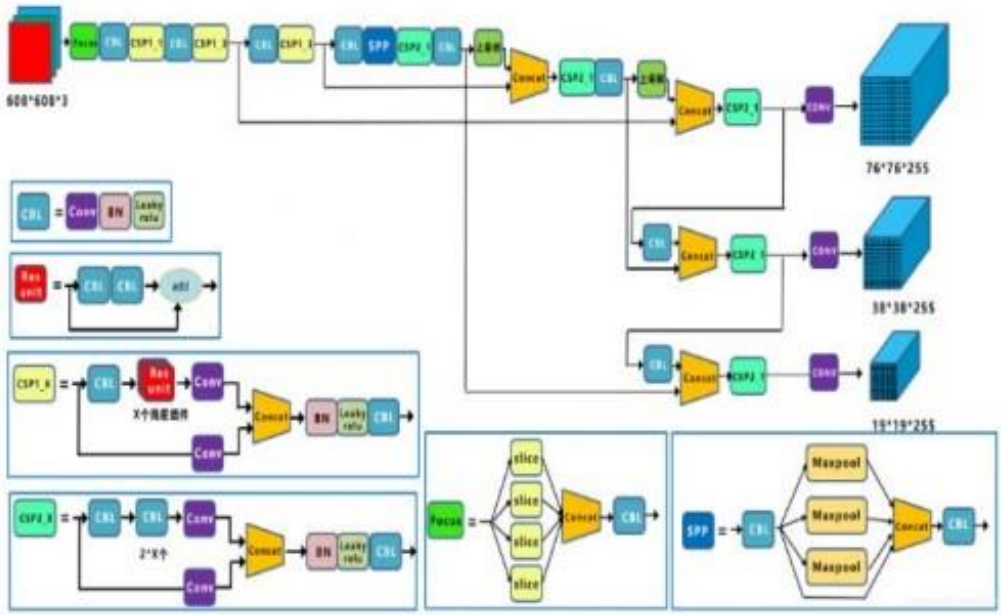
Farklı k değerleriyle k ortalama kümeleme algoritması kullanılarak içinden uygun olanlar seçilmiştir. YOLOv5 için COCO veri seti kullanılmıştır ve bu veri seti 80 farklı sınıftan oluşmuştur. [17]



Şekil 5.YOLOv5 Modelleri [18]

YOLOv5'in dört farklı modeli vardır. (Şekil 5: YOLOv5 modelleri) Bu modeller önceden eğitilmiş 80 sınıf bulunan MSCOCO veri setini kullanmaktadır ve doğruluk payı en yüksek olan ve en geniş model Yolo5x'tir. [4]

2.2.1.YOLO katmanlar



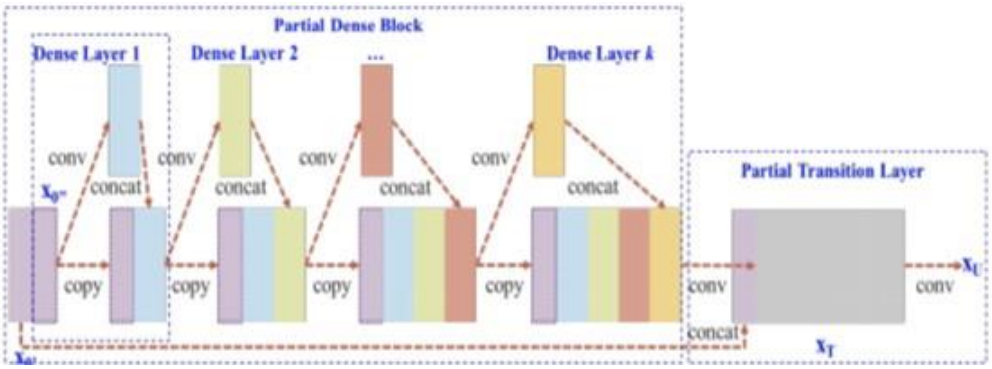
Şekil 6.YOLO Katmanlar [7]

YOLOv5 özetlenecek olursa omurga, boyun, baş kısmından oluşur. Omurga odak yapısı ve CSP ağı, boyun SSP bloğu ve PANet, baş kısmı ise GIoU kullanan YOLOv3 kafasından oluşmaktadır. [7]

2.2.1.1.Omurga

Farklı özelliklerdeki görüntüleri toplayan CNN ağıdır. Omurga CSP Network katmanından oluşur.

2.2.1.1.a.CSP Network



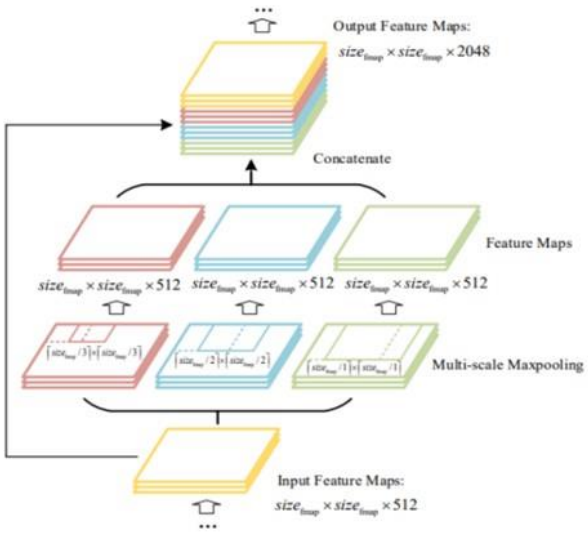
Şekil 7.CSP Network Mimarisi [8]

CSP DenseNet ile aynı çalışma politikasına dayanan bir sistemdir. Sırayla katmanlar arasında geçiş yapılıp farklı yoğun katmanlarda farklı yoğunluklarda işlenirler. [8]

2.2.1.2.Boyun

Görüntüyü tahmin etmek etmek yapılan bir dizi katmanlar bütünüdür. Boyun SSP Blocklarının bir dizi işlemler gerçekleşmesi yaptığı kısımdır.

2.2.1.2.a.SSP Network



Şekil 8.SSP Network Mimarisi [8]

Şekil 8’de gösterilen yeni versiyon YOLO SSP bloğu, öznitelik haritalarının çok ölçekli maksimum havuzlama yapıldıktan sonra klasik SSP bloğundan farklı olarak tek boyutlu bir vektöre dönüştürür.

SSP girdi boyutundan bağımsız sabit boyutlu bir üretim amacı olarak ortaya çıkmıştır. Giriş özellikleri haritaların sürümüne kopyalandı, farklı boyutlardaki çekirdekleriyle maksimum havuzlama gerçekleştirilir.[8]

Yeni SSP bloğu omurgaya entegre edilmiştir. Giren özellik haritalarının sayıları azaltılıp omurga SSP arasında kullanılır. Daha sonra özellik haritaları çoğaltılır ve klasik SSP ile aynı havuzlama işlemine tabii tutulur. Haritanın boyutu sizefmapxsizefmapx512 dir. [4]

2.2.1.2.b.PANet

Omurga sayesinde daha düşük katmanlardan aldığı giriş görüntülerini işler, karmaşık hale getirilen daha derin katmandır. Bu esnada resmin anlam karmaşıklığı artarken, özellik haritalarını uzamsal çözünürlüğü alt örneklemden dolay azalacaktır. Bu özellikleri korumak için Feature Pyramid Network (FPN) mimarisi boyun kısmına uygulanmıştır.[4]

Path Aggregation Network (PAN) FPN’nin gelişmiş bir sürümüdür.

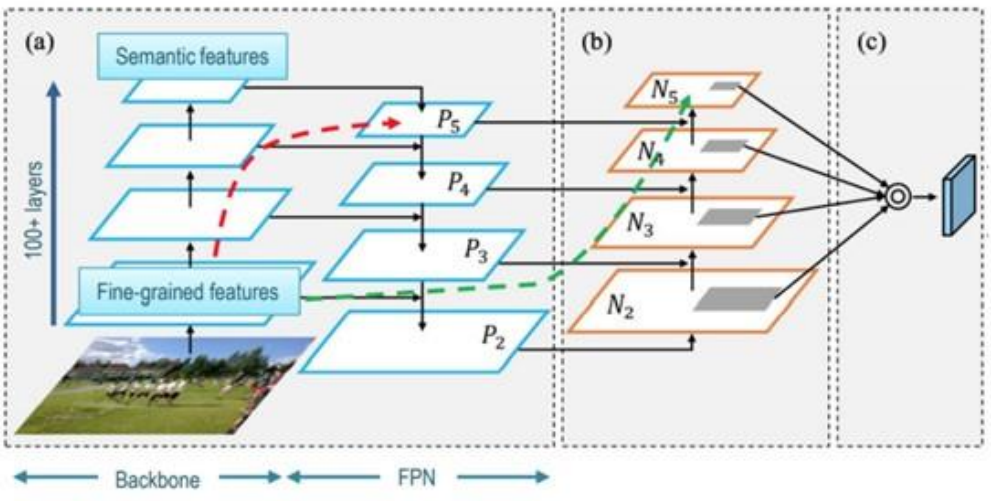


Figure 25. PANet architecture including (a) FPN backbone, (b) bottom-up path augmentation, (c) adaptive feature pooling. (Liu, et al., 2018)

Şekil 9. PANet Mimarisi a)FPN Omurga Yapısı b)Aşağıdan-Yukarıya Çoğaltma Yolu c)Uyarlanabilir Özellik Havuzu [4]

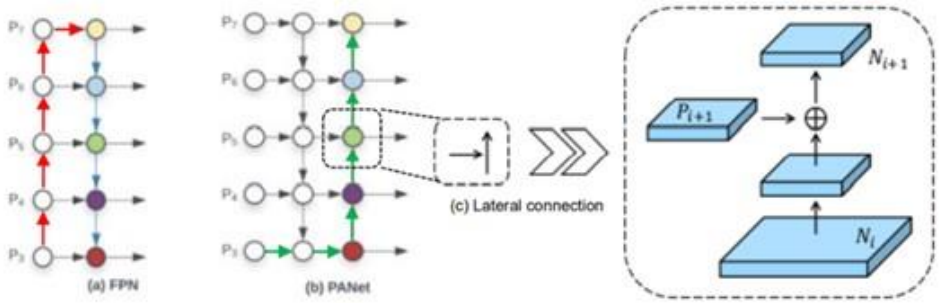


Figure 26. (a) FPN architecture. (b) PAN architecture. (c) Connection in bottom-up augmentation path. (Solawetz, 2020)

Şekil 10.a)FPN Mimarisi b)PAN mimarisi c)Aşağıdan-Yukarıya Çoğaltma Yolu Bağlantısı [8]

Günümüzde derin sinir ağları omurgaları çok sayıda katman içerir. Şekildeki kırmızı yol düşük seviyeden yüksek seviyeye geçişi ifade eder. Kısayol oluşturulup, bilgi akışını kolaylaştırmak amaçlanmıştır.

2.2.1.3.Baş

Boyundan aldığı bilgiler dahilinde sınıflandırma tahmini ve kutulama işlemini yapar.

2.2.1.3.a.YOLOv3

Tek aşamalı bir dedektör için, kafanın işlevi yoğun tahminler yapmak ve sınırlayıcı kutu çizmektir. Son tahmin birimi ve sonuç işlem merkezidir. Baş kısmında koordinatlar (merkez, yükseklik, genişlik), tahmin güven puanı ve olasılık sınıfları işlenir. YOLOv5 YOLOv3 ile aynı kafayı taşır. [8]

2.3.YOLO'nun Arka Planındaki Matematiksel İşlemler

Bir nesne algılama modelinin performansının değerlendirilmesi esnasında birçok kriterler bulunmaktadır.

Intersection over Union (IoU): Bilgi açıklamaları ile gerçek nokta arasındaki farkı bulan ölçümdür. Bu metrik sistem, son teknoloji nesne algılama algoritmalarında kullanılmaktadır. Nesne algılama esnasında model, algıladığı nesne olabileceklere sınırlı kutular çizer. IoU değeri eşik değerinden büyükse nesne alarak algılanır ve kutu çizilir.[4]

IOU = Area of union / Area of intersection

Precision (Hassasiyet): Doğru tahminleri ölçmek için kullanılır.

Precision= TP/(TP+FP)

Recall (Geri Çağırma): Gerçek bir pozitif oran olan geri çağırma kesin referans nesnelerinin tespit edilme olasılığıdır.

Recall= TP/(TP+FN)

Average Precision (AP/Ortalama Hassasiyet): Nesne değerlerini ölçer. Hassasiyet ve geri çağırma değerlerini kullanarak nümerik analiz yapar.

$$\mathrm{AP} = \frac{1}{11\Sigma r\epsilon(0,0.1,0.2,\ldots,1)\rho\mathrm{interp(r)}}$$

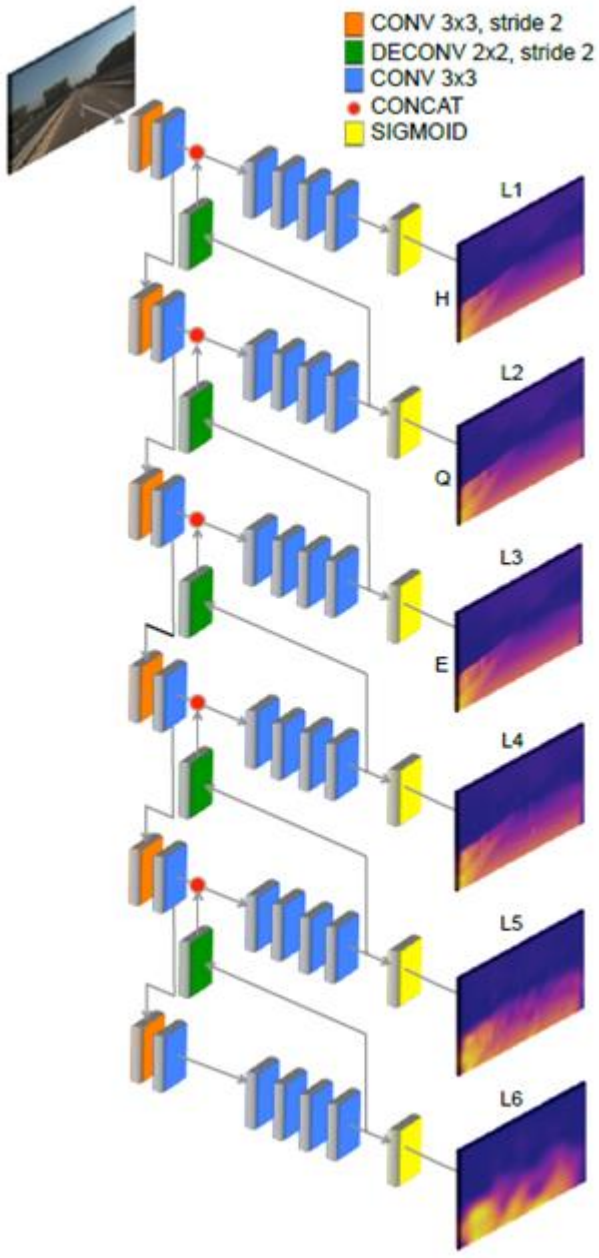
mAP: Veri seti M sınıf kategorisi içeriyorsa, Müzerinden AP ortalaması alınır.

$$mAP = \frac{1}{M\sum_{j=1}^{M} APj}$$

2.4.PyD-Net

Araştırmalarımızda derinlik tahmini için birçok model ve yayın incelenmiştir. Çoğu modelin ancak GPU desteği ile istenen FPS hızlarına ulaştığı görülmüştür. Gerçekleştirmek istediğimiz model ise basit cihazlarda bile yeterli hızlara ulaşarak tahmin yapabilmesidir. Bu bağlamda PyD-Net modeli uygun görülmüştür.

PyD-Net mimarisi Şekil 1'de görüldüğü üzere piramidal özellik çıkarıcı (Pyramidal features extractor) ve derinlik çözücüden (Depth decoder) oluşmaktadır.[6]



Şekil 11.PyD-Net Mimarisi [6]

2.4.1.Piramidal Özellik Çıkarıcı

Piramit özellik çıkarıcı, girilen görüntülerden CNN kullanılarak özellik çıkarılması esasına dayanır. 12 evrişimsel katmandan (Convolutional layer) oluşan küçük kodlayıcılar ile çalışır. İlk katman 2 kayma (Stride) ile ardından gelen katmana bağlanmaktadır ve böylece piramidin ilk katmanını oluşturmaktadır. Bu düzende devam ederek 6 katlı bir piramit oluşmaktadır. Piramidin her katı ile birlikte görüntü çözünürlüğü de 2'nin üssü biçiminde küçülmektedir (1/2'den 1/64'e). Katmanların filtre sayısı piramidin alt katlarına inildikçe sırasıyla 16, 32, 64, 96, 128 ve 196 olarak artmaktadır. Aktivasyon katmanı olarak ise 0.2 alfa değeri ile "leaky ReLU" kullanılmaktadır. Küçük kodlayıcıların kullanılmasının yanı sıra kabadan inceye (Coarse-to-fine) stratejisi ile daha yüksek seviyede düşük seviyedeki gibi ayrıntıları iyileştirmeyi sağlamaktadır.[6]

2.4.2.Derinlik Çözücü

Piramidin en üst katından başlayarak her katta çıkarılan özellikler 4 katmandan oluşan derinlik çözücüden geçerek özellik haritaları üretilir. Bu katmanlardaki filtre sayısı sırasıyla 96, 64, 32 ve 8'dir. Bu çözücünün çıkışı iki amaç için kullanılır. İlki Sigmoid katmanından geçerek derinlik haritasının oluşturulması, diğeri ise çözünürlüğü 2 katına çıkaran 2 kaymalı 2x2 Deconvolution katmanı ile işlenmiş özellikleri piramidin bir sonraki katına taşımaktır. Bir sonraki aşamada ise girilen görüntü ile işlenmiş özellikleri bir birleştirme (Concatenate) işlemi ile yeni bir çözücüye aktarır. Bu en yüksek çözünürlüğe kadar sürmektedir. Her evrişimsel katman "leaky ReLU" aktivasyon katmanı tarafından devam ettirilmiştir. Son katmanlarda ise çıktı görüntülerini normalize etmek için "Sigmoid" katmanı kullanılmıştır.[6]

2.4.3.Model Eğitiminde Kullanılan Kayıp

Projemizde PyD-Net geliştiricileri tarafından eğitilmiş hazır model kullanılmıştır. Bu modelin eğitim aşamasında kullanılan kayıp (Loss) parametresi, farklı ölçeklerde hesaplanan değerlerin toplamı olan çok ölçekli bir parametredir.

$$\mathcal{L}_S = \alpha_{AP}\left(\mathcal{L}_{ap}^{l} + \mathcal{L}_{ap}^{r}\right) + a_{ds}\left(\mathcal{L}_{ds}^{l} + \mathcal{L}_{ds}^{r}\right) + a_{lr}\left(\mathcal{L}_{lr}^{l} + \mathcal{L}_{lr}^{r}\right)$$

Piramidin her katında hesaplanan kayıp sinyali, üç dağıtımın hesaplanmış ağırlık toplamına eşittir. $\mathcal{L}_{ap}$, yeniden düzenleme kaybını ifade eder.

$$\mathcal{L}_{ap}^{l} = \frac{1}{N}\sum_{i,j} \alpha \frac{1 - SSIM\left(I_{i,j}^{l}, \tilde{\mathrm{I}}_{i,j}^{l}\right)}{2} + (1-a)||\left(I_{i,j}^{l}, \tilde{\mathrm{I}}_{i,j}^{l}\right)||$$

Eşitsizlik yumuşaklığı terimi olan $\mathcal{L}_{ds}$, L1 cezasına göre derinlik süreksizliğini caydırır.

$$\mathcal{L}_{ds}^{l} = \frac{1}{N}\sum_{i,j} |\delta_x d_{i,j}^{l}| e^{-\left\|\delta_x I_{i,j}^{l}\right\|} + |\delta_y d_{i,j}\,| e^{-\left\|\delta_y I_{i,j}^{l}\right\|}$$

Son terim ise sağ-sol tutarlılığını kontrol etmektedir.[6]

$$\mathcal{L}_{lr}^{l} = \frac{1}{N}\sum_{i,j} |d_{i,j}^{l} - d_{i,j+d_{i,j}^{l}}^{r}|$$

2.5.Metinden Sesli Çıkış Elde Edilmesi

Metinden sesli çıkış elde edilmesi kısmında çeşitli kütüphaneler tarandıktan sonra "gTTS (Google Text-To-Speech)" kütüphanesine karar verilmiştir.[19] Çalışma prensibi, girdi olarak aldığı "String" ifadeleri önceden programlanmış bir "Speaker" aracılığıyla sesli bir çıktı vermesine dayanır.

Projemizde nesne tespiti için kullandığımız YOLOv5 modeli, algıladığı her nesneyi bir "String" çıktısı olarak aktaracak şekilde kodlanmıştır. Kod içerisine eklediğimiz bir fonksiyon sayesinde algıladığı her nesneyi sesli olarak çıktı verecek şekilde ayarlanmıştır. PyD-Net modelinde ise herhangi bir "String" çıktısı veren bir koda rastlanmamıştır. Bu sebeple kod içerisine çıktı verdiği görüntü matrisi üzerinden, belirli bir eşik değeri aşıldığı takdirde sesli çıktı verecek şekilde bir kod eklenmiştir.

BÖLÜM 3: BULGULAR VE TARTIŞMA

3.1.YOLOv5

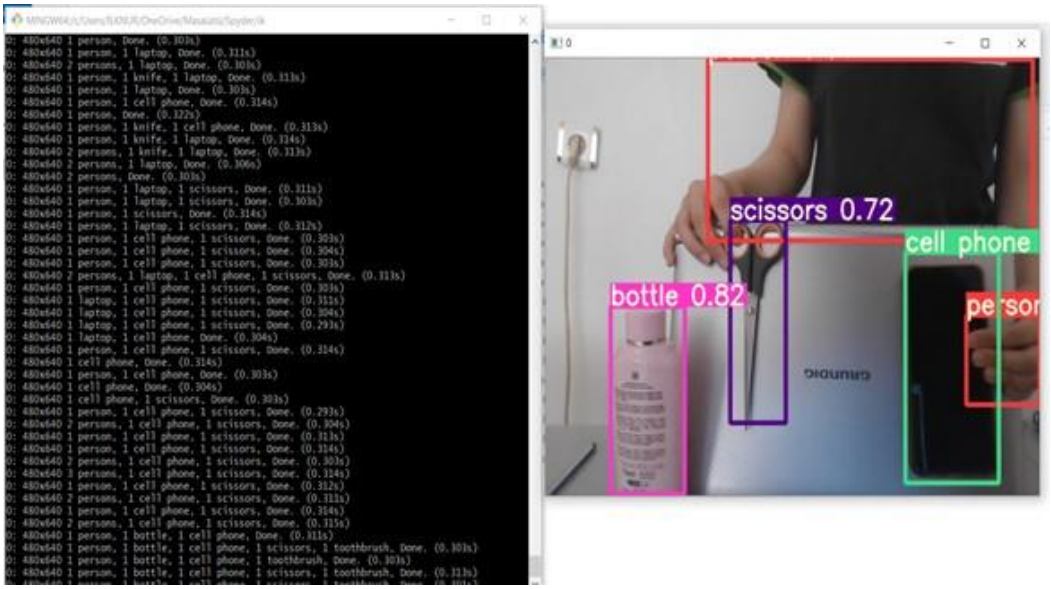
Sistemimizde YOLOv5 modeli kullanılmıştır. Modelimiz YOLO serisinin 5. versiyonu olup Python dilini kullanılarak oluşturulmuş ve hızı ile de dikkat çekmektedir. YOLOv5 COCO veri setini kullanmaktadır ve bu set 80 adet tanımlanmış nesneden oluşmaktadır.

COCO veri seti çok büyük ölçekli bir algılama ve segmentasyon içerir. 330k görüntüye sahip, 80 nesne kategorisi bulunan, süperpiksel malzeme segmentasyonuna sahip bir settir.

YOLOv5 olan 5. nesil YOLO ise diğer versiyonlara göre daha hızlı çalışan ve dilinin Python olup Pytorch geliştiricileri tarafından şekillendirildiği için gelişime daha açık bir versiyondur. YOLO algoritmasını resim, video ve kamera üzerinde kullanabilir ve doğru sonuçlar elde edebiliriz.

Çalışmamızda gerçek zamanlı nesne tanıma mevcut olduğundan komut isteminde "--source 0" parametresiyle kameraya erişim sağlayarak anlık olarak nesnelerin etiketlenmesini sağlayıp nesneleri görebiliriz. Aynı zamanda da kameradan aldığı görüntülerde nesne etiketlemesi gerçekleştirdiği gibi komut isteminde de nesnelerin tek tek isimleri ve sayıları görünmektedir. İsimler başka bir deyişle etiket verileri ise seslendirme kısmında kullanılmak üzere alınmaktadır.

Şekil 12.YOLOv5 Örnek Çalışma 1



Şekil 13.YOLOv5 Örnek Çalışma 2

3.2.PyD-Net

PyD-Net modelinin kullanılması kararlaştırıldıktan sonra çalışmalar başlamıştır. Modelin 4 adet değişken parametresi bulunmaktadır. Bunlar; çözünürlük, kontrol dosyası, genişlik ve yükseklikliktir.

Çözünürlük (resolution) parametresi 3 farklı seçenekte sunulmuştur. Bu seçenekler çıktı görüntü kalitesini ve FPS hızını doğrudan etkilemektedir. “1”, “2” veya “3” şeklinde seçim yapılırken “1” en yüksek kalite, “3” ise en düşük kaliteyi temsil etmektedir. Çalışmamızda görüntü kalitesi yerine hız önem arz ettiğinden “3” seçilmiştir.

Kontrol dosyası (checkpoint) parametresi, önceden eklenmiş ve modelin çalışması için referans olan dosyaları içermektedir.

Genişlik (width) ve yükseklik (height) parametreleri, çıktı görüntünün boyutlarının belirlenmesini sağlamaktadır. Büyük boyutlu bir çıktı, FPS hızını olumsuz etkilemektedir fakat derinlik haritasının daha doğru çıkarılmasını da sağlamaktadır. Bu sebeple çalışmamızda orta boyutlarda, genişliği 704 ve yüksekliği ise 512 alarak FPS hızını ve derinlik haritası doğruluğu korunmaya çalışılmıştır.

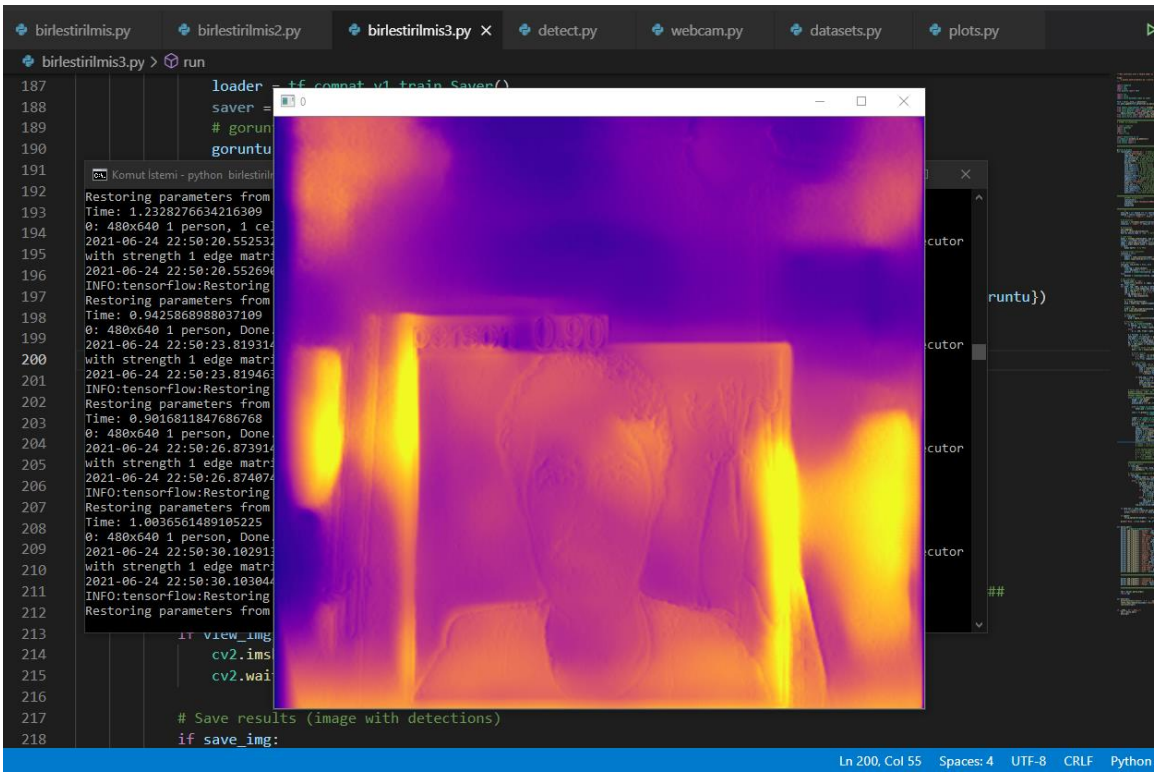
Şekil 14.PyD-Net Örnek Çalışma

3.3.Modellerin Birleştirilmesi

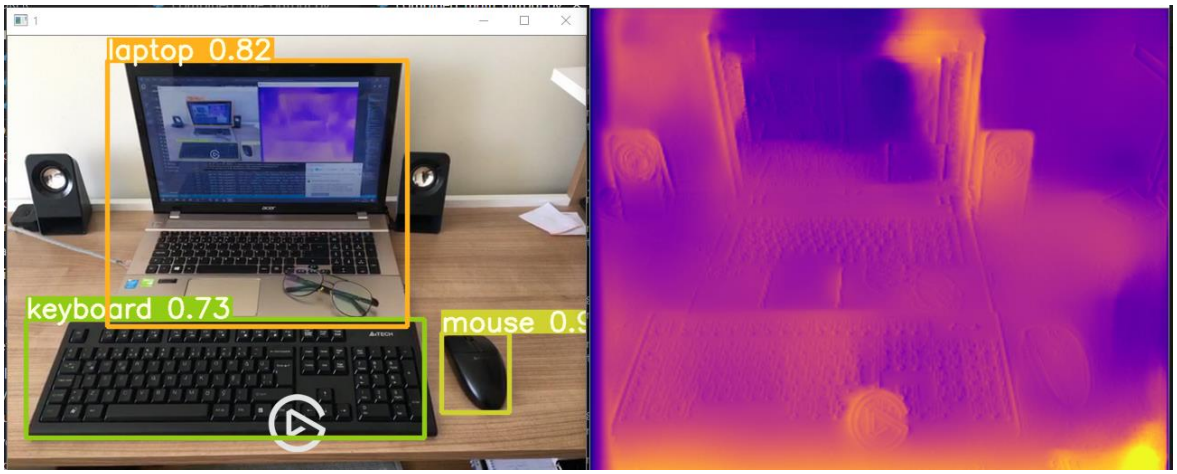
YOLOv5 ve PyD-Net modellerinin birleştirilmesi adına çeşitli fikirler tartışılmış ve denenmiştir. Bu fikirlerden birisi iki farklı kamera kullanılması olmuştur. Maliyet ve ekipman fazlalığı oluşturacağından başka fikirler tartışılmıştır. İki farklı cihazda çalışması da yine bu sebepten atlanmıştır. Bir başka fikir ise Python modülü olan "subprocessing" veya "multiprocessing" kütüphanelerinin kullanımı ile işlemciyi ikiye bölmek ve iki modeli aynı anda çalıştırmak olmuştur. Birçok deneme ve üzerine araştırmalar yapılsa da bir sonuca ulaşılamayacağı görülmüştür.

İki modelin de kameraya ulaşarak çalışması ve işlemcinin aynı anda kamera için tek modele izin vermesinden dolayı model kodlarının doğrudan birleştirilmesi fikri güçlenmiştir. Her modelin kodları satır satır incelenerek çalışma mantığı çıkarılmıştır. YOLOv5 modelinin, kameraya ulaştıktan sonra çeşitli algoritma ve fonksiyonlar ile kameradan aldığı görüntü üzerinde sırasıyla modelin uygulanması, sınıflandırma, tespit etme ve sınırlayıcı kutu çizme işlemlerini uyguladığı görülmüştür. PyD-Net modelinin ise

kameraya ulaştıktan sonra modelini uygulaması ve renk haritasının basılması ile çıktı verdiği görülmüştür. Kod karmaşıklığının daha az olmasından dolayı PyD-Net modelinin YOLOv5 modeli içine atanabileceği tespit edilmiştir.



Şekil 15.Birleştirilmiş Model Çalışması 1



Şekil 16.Birleştirilmiş Model Çalışması 2

BÖLÜM 4: SONUÇLAR VE TARTIŞMA

Çalışmamızda YOLOv5 nesne tespiti ve PyD-Net derinlik tahmini modelleri birleştirilmiş ve model çıktıları sesli biçimde dışarı verilmiştir. Elde ettiğimiz sonuç için birçok model incelenmiştir. Hız, boyut ve doğruluk gibi parametreler dikkate alınarak nesne tespiti için YOLOv5 ve derinlik tahmini için PyD-Net seçilmiş ve birleştirilmiştir.

Model birleştirilmesinde, iki modele ait asgari dosyalar seçilmiş ve tek klasöre indirgenmiştir. Ardından modellerin birlikte çalışabilmesi adına fikir alışverişi yürütülmüştür. YOLOv5 algoritmasının kameraya ulaştığı ve hapsettiği görüntünün kopyasının PyD-Net modeline girdi olarak verilmesi kararlaştırılmıştır. Böylece iki model aynı kod içinde bağımsız çalışabilmiştir. Çıktı olarak ise iki modelin de çıktıları görüntülenmiştir. Alternatif olarak ise YOLOv5 modelinin çıktısının PyD-Net modeline girdi olarak verilmesi ve tek çıktı görüntü alan bir kod da hazırlanmıştır.

Sesli çıkış adına kod içerisinde hazırlanan bir fonksiyon, tespit edilen nesneleri sırasıyla söylemektedir. Derinlik haritasının sesli çıkışı adına düşünülen yöntem ise çıktı matrisinde ortalama işlemi yapmak ve belirli bir değerin üzerinde seyreden ortalama değer (yakınlık / sarı renk çoğunluklu) durumunda “Yakında” olarak sesli çıkış sağlanmıştır.

KAYNAKLAR

[1] Ertel, Wolfgang. *Introduction to artificial intelligence*. Springer, 2018.

[2] Caiming Zhang, Yang Lu, Study on artificial intelligence: The state of the art and future prospects, Journal of Industrial Information Integration,Volume 23, 2021.

[3] Kansal, Aman, Michel Goraczko, and Feng Zhao. "Building a sensor network of mobile phones." *2007 6th International Symposium on Information Processing in Sensor Networks*. IEEE, 2007.

[4] CENGİL, Emine, and Ahmet ÇINAR. "Poisonous Mushroom Detection using YOLOV5." *Turkish Journal of Science and Technology* 16.1 (2021): 119-127.

[5] Tan, Fatma Gülşah, et al. "Derin Öğrenme Teknikleri ile Nesne Tespiti Ve Takibi Üzerine Bir İnceleme." Avrupa Bilim ve Teknoloji Dergisi 25 (2021): 159-171.

[6] Poggi, Matteo, et al. "Towards real-time unsupervised monocular depth estimation on cpu." *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018.

[7] Yang, Guanhao, et al. "Face Mask Recognition System with YOLOV5 Based on Image Recognition." *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*. IEEE, 2020.

[8] Thuan, Do. "Evolution of yolo algorithm and yolov5: the state-of-the-art object detection algorithm." (2021).

[9] Kaweesinsakul, Kanlayanee, Siranee Nuchitprasitchai, and Joshua M. Pearce. "Open source disease analysis system of cactus by artificial intelligence and image processing." *arXiv preprint arXiv:2106.03669* (2021).

[10] Sreeraj M, Jestin Joy, Alphonsa Kuriakose, Bhameesh M B, Anoop K Babu, Merin Kunjumon, VIZIYON: Assistive handheld device for visually challenged, Procedia Computer Science,Volume 171, 2020.

[11] Dunai, Larisa, et al. "Real-time assistance prototype—a new navigation aid for blind people." *IECON 2010-36th Annual Conference on IEEE Industrial Electronics Society*. IEEE, 2010.

[12] Mahjourian, Reza, Martin Wicke, and Anelia Angelova. "Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

[13] Sinir Sistemi, https://slideplayer.biz.tr/slide/10375037/, (Erişim tarihi: 25.05.2021)

[14] Yapay Sinir Ağları, https://arpit3043.medium.com/reasons-to-use-random-forest-over-a-neural-network-comparing-machine-learning-versus-deep-learning-c6b8380d6c7f, (Erişim tarihi: 05.05.2021)

[15] Heger, D. “An Introduction to Artificial Neural Networks (ANN) Methods, Abstraction and Usage, (2015).

[16] Ihsan, Candra Nur. "Klasifikasi Data Radar Menggunakan Algoritma Convolutional Neural Network (CNN)." *DoubleClick: Journal of Computer and Information Technology* 4.2 (2021): 115-121.

[17] Şeker, Abdulkadir, Banu Diri, and Hasan Hüseyin Balık. "Derin öğrenme yöntemleri ve uygulamaları hakkında bir inceleme." *Gazi Mühendislik Bilimleri Dergisi (GMBD)* 3.3 (2017): 47-64.
[18] YOLOv5 Modelleri, https://pytorch.org/hub/ultralytics_yolov5/, (Erişim tarihi: 10/05/2021)
[19] Google Text-To-Speech, https://cloud.google.com/text-to-speech, (Erişim tarihi: 29.06.2021)

EKLER

Ek-1 Yazıdan Sese Çeviren Fonksiyonun Kodu

```
from gtts import gTTS
import playsound
import os

def text_to_speech(text):

    tts = gTTS(text=text, lang="tr")
    filename="voice.mp3"
    tts.save(filename)
    playsound.playsound(filename)
    os.remove(filename)
```

Ek-2 Modelin Akış Diyagramı