

T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

ISE 401 BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ BİTİRME
ÇALIŞMASI

BİLGİSAYARLI GÖRÜ TEKNOLOJİLERİNİ
KULLANARAK İŞ YERLERİNDE KKD KULLANIM
DURUMUNU DENETLEYEN KAMERA

B181200376 – Şeyda GÜNDOĞDU
B191200370 – Esra AKSU
B191200371 – Liva Nur PULAT

Bölüm : **BİLİŞİM SİSTEMLERİ**
MÜHENDİSLİĞİ
Danışman : **Doç. Dr. İhsan Hakan SELVİ**

2021-2022 Bahar Dönemi

T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

BİLGİSAYARLI GÖRÜ TEKNOLOJİLERİNİ
KULLANARAK İŞ YERLERİNDE KKD KULLANIM
DURUMUNU DENETLEYEN KAMERA

ISE 402 - BİTİRME ÇALIŞMASI

Şeyda GÜNDOĞDU

Esra AKSU

Liva Nur PULAT

Fakülte Anabilim Dalı : BİLİŞİM SİSTEMLERİ
MÜHENDİSLİĞİ

Bu tez .. / .. / ... tarihinde aşağıdaki jüri tarafından oybirliği / oyçokluğu ile kabul edilmiştir.

.....
Jüri Başkanı

.....
Üye

.....
Üye

ÖNSÖZ

Teknolojik gelişmeler hayatımızın birçok noktasında yeniliğe ve değişikliğe yol açmaktadır. Sadece getirdiği kazanç ve kolaylık için değil, aynı zamanda iş güvenliği alanında da işletmeler açısından büyük önem taşımaktadır. Günümüzde hala ciddi bir problem olan iş kazalarının başlıca sebebi ihmalkârlıktır ve bu alanda denetim yapacak sistemlere ihtiyaç duyulmaktadır. Bu çalışmada da Bilgisayarlı Görü sistemi kullanılarak çalışanların iş sahasında baret takıp takmadıkları kontrol edilecek ve iş güvenliği sağlanacaktır.

İÇİNDEKİLER

ÖNSÖZ.....	ii
İÇİNDEKİLER.....	iii
SİMGELER VE KISALTMALAR LİSTESİ.....	v
ŞEKİLLER LİSTESİ.....	vi
TABLolar LİSTESİ.....	vii
ÖZET.....	viii

BÖLÜM 1.

GİRİŞ.....	1
------------	---

BÖLÜM 2.

BİLGİSAYARLI GÖRÜ İLE YAPILAN İŞ GÜVENLİĞİ UYGULAMALARI.....	11
2.1. EWSAI (Electric Work Safety with Artificial Intelligence).....	11
2.2. CORTES.....	11
2.3. Manuel Pres Hatlarında Yapay Zekâ Güvenlik Sistemi	12
2.4. COGNITIWE.....	12
2.5. RING	12
2.6. HGS DIGITAL	12
2.7. INTENSEYE	13

BÖLÜM 3.

ÖNERİLEN SİSTEM.....	14
3.1. Yönetim.....	15
3.2. Makine Öğrenmesi.....	15
3.3. Derin Öğrenme.....	16
3.3.1. Evrişimsel Sinir Ağları (CNN).....	16
3.4. Nesne Tespit ve Takip Metotları	16
3.4.1. Nesne takip yöntemlerinin sonuçlarını değerlendirme.....	18

3.4.2. Literatürde nesne tespiti ve takibi için kullanılan yöntemler.....	20
BÖLÜM 4.	
PROJE FİKRİNİN UYGULANMASI.....	26
4.1.Kullanılan Teknolojiler.....	26
4.1.1. Python ve OpenCV	26
4.1.2.Kullanılan algoritma.....	30
4.1.3.Google Colab.....	34
4.1.4.Anaconda&Spyder IDE.....	35
4.1.5.Darknet.....	35
4.2.Deney Düzenegi.....	35
4.2.1. Veri Seti.....	35
4.2.2. Konfigürasyon İşlemleri.....	36
4.2.3. Eğitim ve Sonuç.....	38
BÖLÜM 5.	
SONUÇLAR VE ÖNERİLER.....	41
KAYNAKLAR.....	42
ÖZGEÇMİŞLER.....	44
BSM 401 BİLGİSAYAR MÜHENDİSLİĞİ BİTİRME	
ÇALIŞMASIDEĞERLENDİRME VE SÖZLÜ SINAV	
TUTANAĞI.....	45

SİMGELER VE KISALTMALAR LİSTESİ

İSG	: İş Sağlığı ve Güvenliği
KKD	: Kişisel Koruyucu Donanım
DL	: Deep Learning (Derin Öğrenme)
ML	: Machine Learning (Makine Öğrenmesi)
AI	: Artificial Intelligence (Yapay Zekâ)
CNN	: Convolutional Neural Network (Evrişimsel Sinir Ağı)
TP	: True Positive
TN	: True Negative
FP	: False Positive
FN	: False Negative
GB	: Gigabyte
RAM	: Random Access Memory (Rastgele Erişimli Hafıza)
RGBA	: Red-Green-Blue-Alpha (Kırmızı-Yeşil-Mavi renklerin baz alındığı renk standardı)

ŞEKİLLER LİSTESİ

Şekil 3.1.	Use-Case Diyagramı.....	16
Şekil 3.2.	KKD İzleme Sisteminin Akış Diyagramı.....	17
Şekil 3.3.	Kesişme durumunda TP, FP ve FN değerlerin hesaplanması	20
Şekil 3.4.	Nesne takip algoritmaların genel diyagramı	22
Şekil 3.5.	Geçici değişiklikler yöntemi, a: $t-1$ zamanındaki görüntü ($It-1$), b: t zamanındaki görüntü (It), c: t zamanındaki fark görüntüsü $Nt=(It)-(It-1)$	24
Şekil 3.6.	Bir video çerçevesindeki nesne tespiti için arka plan görüntüsü çıkartma yöntemi	25
Şekil 4.1	YOLO algoritmasındaki sınırlayıcı kutucukların Özellikleri.....	33
Şekil 4.2.	YOLO algoritmasının çalışma mantığı.....	33
Şekil 4.3.	YOLO tarafından tespit edilen nesneler.....	34
Şekil 4.4.	YOLO ve diğer nesne tespit algoritmalarının MS COCO veri seti üzerindeki performanslarının karşılaştırması.....	35
Şekil 4.5.	Makesense'in nesne algılama için resim etiketleme arayüzü.	36
Şekil 4.6.	helmet.names dosyası içeriği.....	36

Şekil 4.7.	Konfigürasyon dosyası içerisindeki batch ve subdivisions değerleri.....	37
Şekil 4.8.	helmet.data dosyası içeriği.....	37
Şekil 4.9.	Eğitimin 4. Saatine ait loss grafiği.....	38
Şekil 4.10.	Colab’de elde edilen, modele ait istatistiki bilgiler.....	38
Şekil 4.11.	Helmet Detector’un resim üzerinden bareti tespit etmesine dair ekran görüntüsü.....	39
Şekil 4.12.	Helmet Detector’un bilgisayar kamerası üzerinden gerçek zamanlı nesne tespiti.....	40

TABLÖLAR LİSTESİ

Tablo 3.1.	True Positive (TP), False Positive (FP), False Negative (FN), True Negative (TN) değerlerin hesaplanması.....	20
------------	---	----

ÖZET

Anahtar kelimeler: Derin Öğrenme, Bilgisayarlı Görü, İş Sağlığı ve Güvenliği

Geçmişten günümüze kadar gelen ve iş hayatındaki önemini koruyan iş sağlığı ve güvenliği alanında gelişmeler devam etmektedir. Ancak her ne kadar ekipmanlar daha dayanıklı, kontroller daha sıkı yapılmaya çalışılmış olsa da insan gücü yetersiz kalmıştır. Gelişen teknolojiye rağmen gözden kaçan hatalar ve ihmaller yüzünden kaza ve ölüm sayılarında büyük bir değişiklik olmamıştır.

Projemiz için tasarlayacağımız Bilgisayarlı Görü sistemi ile bu probleme çözüm getirmeyi amaçlıyoruz. Bilgisayarlı görüş, dijital görüntüler veya videolar üzerinde işlem yapan ve elde ettiği verilen üzerinden anlamlar çıkarmaya çalışan bir sistemdir. Diğer bir deyişle, yaptığı analizler sonucunda ulaştığı sonuçları bir insan gibi yorumlamasıdır [1]. Bilgisayarlı görüş ve makine öğrenimi kullanarak otomatik baret tespiti için bir model geliştirilmesi ve herhangi bir ihmal karşısında uyarı vermesini sağlanması hedeflenmektedir. Bu çalışmada, nesne tespiti yapmak için DL algoritmaları kullanılarak yeterli büyüklükte bir veri seti ile model bir insanın KKD kullanımını algılayabilecek seviyeye getirilmiştir.

ABSTRACT

Keywords: Deep Learning, Computer Vision, Occupational Health and Safety

Developments continue in the field of occupational health and safety, which has remained important in business life from the past to the present. However, despite the fact that the equipment was more durable and the controls were tried to be tighter, the manpower was insufficient. Although the developing technology, there has not been a big change in the number of accidents and deaths because there are many mistakes and omissions that are overlooked.

We aim to bring a solution to this problem with the Computer Vision system we will design for our project. Computer vision is a system that processes digital images or videos and tries to make sense of the data it obtains. In other words, it is interpreting the results it has reached as a result of the analyzes it has made, like a human being. It is aimed to develop a model for automatic helmet detection using computer vision and machine learning and to provide a warning in case of any negligence. In this study, the model created with a sufficiently large data set by using DL algorithms for object detection has been brought to a level that can detect a person's use of PPE.

BÖLÜM 1. GİRİŞ

İş sağlığı ve güvenliği; iş ve işçiyi korumak adına oluşturulmuş kurallar bütünüdür. İşyerlerinde, çalışanlarına hem fiziksel hem de ruhsal yönden sağlıklı çalışma ortamı sunmak, meydana gelecek olumsuz etkenlerden korumak ve işletme için en iyi üretim verimini en az riskle almayı amaçlamaktadır [2]. Özetle; iş sağlığı ve güvenliğinin amacı, işyerlerinde meydana gelebilecek tüm uzun ve kısa vadeli sağlık problemlerini önlemek ve meslek hastalığını ortadan kaldırmaktır.

Ancak bilimsel ve teknolojik ilerlemeye rağmen, Uluslararası Çalışma Örgütü (ILO) tarafından sağlanan istatistikler, birçok ülkede çalışma koşullarının iş kazaları sorununu önemli ölçüde azaltacak derecede değişmediğini göstermektedir. Meydana gelen kazalar incelendiğinde Yapı ve Madencilik en çok kayıp veren sektörler olurken yaranın alındığı bölgeler arasında ilk sırada yüz ve kafa bölgesi gelmektedir. Sadece Amerika Birleşik Devletleri'nde her yıl yaklaşık 1,7 milyon insan travmatik beyin hasarı (TBH) nedeniyle hastaneye kaldırılmakta veya ölmektedir [3].

Kişisel Koruyucu Donanım kullanımının büyük ölçüde güvenlik sağladığı bilinmesine rağmen, işyerlerinde kullanımı ihmal edilmektedir. Yine Uluslararası Çalışma Örgütü'nün raporuna göre dünyada her yıl iş kazası nedeniyle ölen 2,34 milyon insanın önemli bir kısmı ihmalden kaynaklandığı tahmin edilmektedir [3]. Bunun önüne geçmek için para cezaları ve çeşitli yaptırımların uygulanmasıyla birlikte üst düzey bir çalışan tarafından denetim yapmaya çalışılmıştır. Ancak tüm çalışma süresi boyunca operatörlerin kontrol edilmesi neredeyse imkânsız olduğu için verimli bir çözüm olamamıştır.

Baş koruma türleri olarak başlıca endüstriyel güvenlik baretleri, koruyucu başlıklar ve itfaiyeci kaskları gelmektedir. Genellikle yeterince sabit olmayan nesnelerle çalışıldığında ve düşme riski olan taşımalar gerçekleştirildiğinde kullanılmaktadır. Bir kaskın koruyucu yapısı, darbe aldığı anda kendisini sıkıştırarak kuvveti emer ve zamanla

eski haline döner. Bu sayede çarpışma şiddeti azalır ve meydana gelen kaza en az hasarla atlatılmış olur.

Kişisel Koruyucu Donanım bu kadar önem arz ederken hala ihmal ediliyor olması ciddi bir problemdir. Bu nedenle, bilgisayarlı görü sistemi ile kullanım denetimi yaparak iş sağlığı ve güvenliği alanında gelişme kaydetmek hedeflenmiştir.

BÖLÜM 2. BİLGİSAYARLI GÖRÜ İLE YAPILAN İŞ GÜVENLİĞİ UYGULAMALARI

Araştırmalara göre, iş güvenliğini sağlamak için çeşitli kanunlar ve uygulamalar devreye sokulsa da her yıl binlerce kişi iş kazalarında yaralanmakta veya hayatını kaybetmektedir. Yapay zekâ ve bilgisayar görüşü alanındaki son gelişmeler sayesinde standart süreçleri gerçek zamanlı olarak kontrol altına alınabilir ve güvenlik ihlallerini anında tespit edilebilir hale gelmiştir. Aşağıda, bu teknolojileri kullanarak iş sağlığı ve güvenliği alanında gelişme sağlayan bazı uygulamalar incelenmiştir.

2.1. EWSAI (Electric Work Safety with Artificial Intelligence)

Türk Ar-Ge girişim şirketi Ayvos tarafından geliştirilen proje, elektrik üretim ve dağıtım sektöründe meydana gelen elektrik kaynaklı kazaların önlenmesini amaçlamıştır. Makine öğrenimi ve derin öğrenme tabanlı görüntü işleme altyapısıyla; trafo, pano, elektrik hattı alanlarında işlem öncesi ve sırasında personellerin kıyafet/ekipman uygunluğunun akıllı şekilde analizini yaparak uygun olmayan veya eksik ekipman/kıyafet tespiti yaparak anlık olarak amir pozisyonundaki kişilere durum hakkında uyarı verilmesini sağlayan bir sistemdir.

2.2. CORTES

Ayvos tarafından geliştirilen bu uygulama ile her sektör için çalışanların güvenliği denetlenmektedir. Anlık kontrol ve uyarı sistemleri ile olası ihlallerde yetkililer anlık bildirilmektedir. Aynı zamanda detaylı raporlamalar ile gelişime açık noktalar tespit edilerek gerekli iyileştirmeler yapılmaktadır.

2.3. Manuel Pres Hatlarında Yapay Zekâ Güvenlik Sistemi

Coşkunöz ve Event Gates tarafından gerçekleştirilmiştir. Proje, insan ve uzuvlarını kamera görüntüleri vasıtasıyla nesne tanımlama algoritmaları ile işleyerek, tanımlı

alan ile insan çerçevesinin kesişmesi halinde ihlal durumu oluşturmakta ve prese hata sinyali göndererek, hata devam ettiği sürece dışarıdan müdahale ile çalıştırılmasını engellemektedir. Ayrıca sistemde herhangi bir arıza olması veya devre dışı kalması durumunda presi de devre dışı bırakacak bir poka-yoke uygulanmaktadır.

2.4. COGNITIVE

Yapay zekâ destekli video analiz çözümleri ile iş güvenliği, kalite kontrol, akıllı stok takibi gibi birçok alanda hizmet sunmaktadır. Ana sektör olarak üretim ve perakendeye odaklı çalışmaktadır. Üretim sektöründe baret, yelek gibi güvenlik ekipmanlarının kontrolü, sosyal mesafe kontrolü, maske denetimi, tehlikeli alan giriş çıkış kontrolü, kampüs içi forklift denetimi, kalite kontrol gibi birçok başlıkta çözümler geliştirmektedir.

2.5. RING

Projenin önceliği şantiye güvenliği sağlamaktır. Güvenlik kameralarına, ışıklarına ve hareket sensörlerine bağlanarak her zaman açık bir güvenlik sistemi oluşturmaktadır. Aynı zamanda oluşabilecek acil durum yardımları için otomatik arama yapmaktadır.

2.6. HGS DIGITAL

Kablosuz kameralar ile yakalanan görüntülerle iş yerindeki anormallikler tespit edilmesi ve bölgeye en yakın sorumluya bildirim göndererek uyarı yapılması sağlanmaktadır. Yerel güvenlik sorumlusunun müdahalesi ile sistem uyarısı doğrulanırsa sorunun olduğu bölge özellikle takip edilmektedir. Amaç, zaman içinde otomasyonu artırmaktır.

2.7. INTENSEYE

Kesintisiz iş yeri güvenliği sağlayabilen bir yapay zekâ platformudur. Üretim tesisleri ve depolarda var olan kameralara bağlanarak gerçek zamanlı yapay zekâ analizleri yapmaktadır. Bu sayede üretim tesisi içinde yaşanabilecek tehlikeli durumları ve güvenlik ihlallerini yapay zekânın bildirimleriyle tespit etmektedir. Çalışanların yaşlarını, cinsiyetlerini ve duygu durum değişimleriyle birlikte vücut poz analizi ve uzuv takibi yapabilmektedir. Özellikle ağır sanayi, otomobil üretim, enerji ve gıda sektörleri gibi iş kazalarının yoğunluklu yaşandığı sektörlerde başta olmak üzere her sektör için uygun bir hizmet sunmaktadır.

BÖLÜM 3. ÖNERİLEN SİSTEM

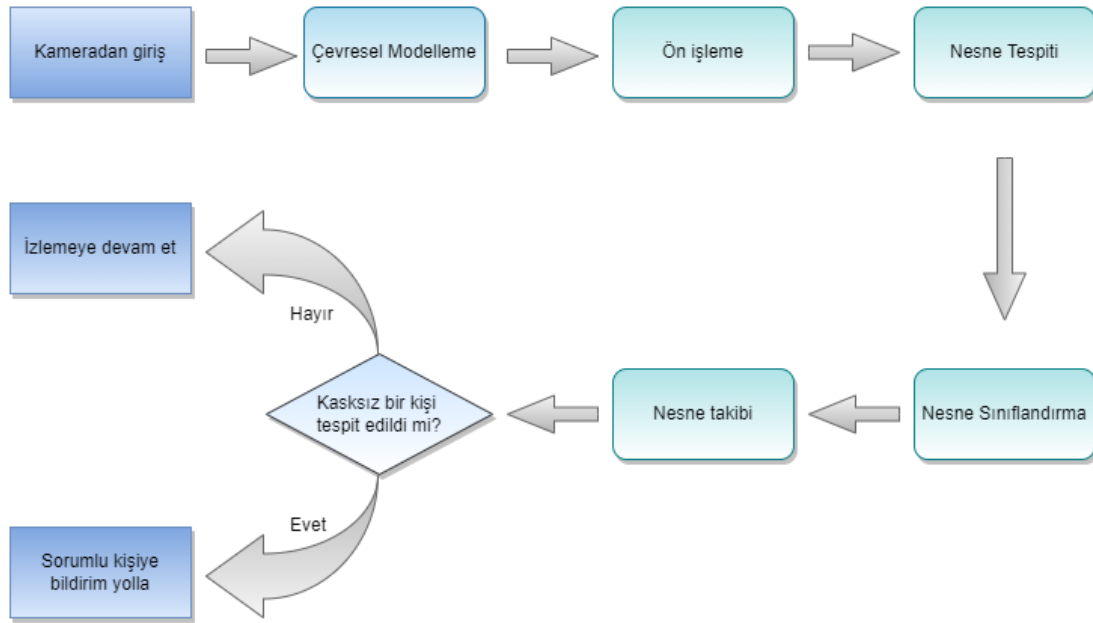
KKD kullanımının söz konusu olduğu iş yeri inceleyen bir kamera düşünülmüştür. Bu kamera işçilerin KKD kullandığının tespiti yapabilecek şekilde eğitilmiştir. KKD kullanmayan bir çalışan tespit ettiğinde İSG sorumlusunun kullandığı web ve mobil uygulamalara bildirim gönderir. Bu proje fikri ile KKD kullanılmamasından kaynaklı iş kazalarının önüne geçileceği düşünülmektedir.

Proje için düşünülen use-case (kullanım senaryosu) ve input-output diyagramı Şekil 3.1’de gösterilmiştir.



Şekil 3.3. Use-Case Diyagramı

Sisteme ait input-output diyagramı ise Şekil 3.2’de görüldüğü gibidir (izlenen adımların açıklamaları ‘Nesne Tespit ve Takip Metotları’ başlığı altında yapılmıştır).



Şekil 3.2. KKD izleme sistemine ait akış diyagramı

3.1. Yöntem

Bilgisayarlı görü en temel şekilde insan beyninin görmeyle ilgili yapabildiklerini makineye yaptırmaya çalışmaktır. En temel görevi de nesneleri tanımak, gruplayabilmek ve girdilerden anlamlı bilgiler çıkararak eylemde bulunmasını sağlayabilmektir. Bilgisayarlı görü çok fazla veriye ihtiyaç duyar. Ayrımları ayırt edene ve nihayetinde görüntüleri tanıyana kadar veri analizini tekrar tekrar çalıştırır. Bunları yapabilmesi için makine öğrenmesi ve CNN adı verilen derin öğrenme algoritmaları kullanılır.

3.2. Makine Öğrenmesi

ML, insanların öğrenme şekillerini taklit etmek için veri ve algoritmaların kullanımına odaklanıp doğruluğunu kademeli olarak artıran bir yapay zekâ (AI) ve bilgisayar bilim dalıdır. Makine öğrenmesi, büyüyen veri bilimi alanının önemli bir bileşenidir. İstatiksel yöntemler kullanılarak, algoritmalar; sınıflandırmalar veya tahminler yapmak üzere eğitilir ve veri madenciliği projelerinde temel içgörülerini ortaya çıkarmaktadır. Bu içgörüler, sonrasında uygulamalar ve işler dahilinde karar verme sürecini teşvik ederek, ideal anlamda önemli büyüme ölçütlerini etkiler.

Makine öğreniminde bilgisayarla görme, derin öğrenmede, bir görüntüde ilgilenilen nesneyi gösteren açıklamalı görüntüler aracılığıyla veri setlerini analiz etmek için kullanılır. Denetimli makine öğrenimi algoritmaları eğitimi için etiketlenmiş binlerce veya milyonlarca görüntüyü besleyen görsel verileri anlamak için kalıpları tanıyabilir.

3.3. Derin Öğrenme

DL; nesne tanıma, doğal dil işleme gibi alanlarda çok katmanlı yapay sinir ağlarını kullanan bir yapay zeka yöntemi olup makine öğrenmesinin çeşitlerinden biridir. Derin öğrenme, geleneksel makine öğrenmesi yöntemlerinden farklı olarak kodlanmış kurallar ile öğrenmek yerine; resim, video, ses ve metinlere ait verilerin simgelerinden otomatik olarak öğrenebilmektedir. Esnek yapıda olduklarından, ham resim ya da metin verisinden de öğrenebilmekte ve verinin büyüklüğüne göre tahmin doğrulukları artabilmektedir. Bununla birlikte derin öğrenme, örnekler üzerinden öğrenme işlemini gerçekleştirmektedir.

3.3.1. Evrişimsel Sinir Ağları (CNN)

CNN, bir girdi görüntüsünü alıp, görüntüdeki çeşitli görünüşleri/nesneleri birbirinden ayırabilen derin öğrenme algoritmasıdır. Evrişimli sinir ağları, temel olarak görüntüleri sınıflandırmak (örneğin gördüklerini isimlendirmek), benzerlikler kümelemek (fotoğraf arama) ve sahnelerde nesne tanıma yapmak için kullanılan derin yapay sinir ağlarıdır.

Evrişimsel sinir ağları yapısı gereği input olarak resim ya da videoları alır. Tabi ki resimleri alırken ilgili formata çevrilmiş olması gerekir. Örneğin bir konvolüsyonel sinir ağına bir resim veriyorsak bunu matris formatında vermemiz gerekmektedir.

3.4. Nesne Tespit ve Takip Metotları

Görüntü İşleme, görüntüyü dijital form haline getirmek ve bazı işlemleri gerçekleştirmek için geliştirilmiş, spesifik görüntü elde etmek veya ondan bazı yararlı bilgiler çıkarmak için kullanılan bir yöntemdir. Bu yöntemin girdisi video kesiti ve ya

fotoğraf gibi bir görüntüdür. Çıktısı ise görüntünün istenilen ya da dikkat edilmesi gereken bölümüne karşılık gelir. Takip edilecek nesnenin değişken bir ortam içinde bulunması nesne takibi ve analizini zorlaştıran temel problemdir. Bu problemleri çözmek ve nesnenin başarılı bir şekilde takip edilmesi için birçok farklı yöntem gelişmiştir. Bu kısımda nesne takibi için kullanılan yöntemleri ve aşamalarını gerekli şekilde açıklanacaktır.

Günümüz teknolojisinde birçok cihazda kamera erişimi söz konusudur, bu da görsel olarak çok sayıda veri üretilmesi demektir. Bu yüzden de elde edilen görüntü verisinin analizi ve bu veriden bir sonuç elde edebilmek ayrı bir öneme sahip olmuştur. Görüntü işleme teknikleri ile ham veriden anlamlı sonuçlar çıkartabilmek mümkündür. Örneğin akıllı telefonlardaki kameralar sayesinde yüz tanımlama [4] veya trafikteki araçların plaka tespiti [5] gibi günlük uygulamalar görüntü işleme alanının en belirgin uygulamalarındandır.

Görüntü işleme uygulamalarının önemli konulardan biri de nesne takibidir. Nesne takibi bir görüntü dizisi ya da bir videodaki önceden belirlenmiş ya da belirlenmemiş [6] nesnelerin konum, hız veya doğrultu gibi bilgilerinin edinilmesidir. Nesne takibi için literatürde birçok yöntem vardır. Bir video içerisindeki arka arkaya gelen iki imgenin birbirinden çıkarılması gibi basit uygulamalardan başlayarak günümüzde popüler olan derin öğrenme yöntemlerine kadar pek çok yöntem bulunmaktadır [6]. Mevcut video içerisinde nesnenin bulunduğu ortamın arka planının çıkartılmasıyla nesne hareketi takip edilebilir [7]. Yine bir görüntü dizisindeki takip edilecek nesnenin arka plan ve nesne olarak sınıflandırılmasıyla nesne takibi sağlanabilmektedir [8].

Günümüzde hareketli ortam, değişen ışık koşulları ve kalite seviyesi düşük görüntüde bile gerçek zamanlı olarak nesne takibi yapabilen derin öğrenme algoritmaları oldukça önem kazanmıştır. Son zamanlarda artan işlem gücü ve grafik işlemcilerdeki gelişmelere paralel olarak, derin öğrenme yöntemleri, büyük veri analizinde, konuşma tanımlama, görüntü sınıflandırması ve nesne takibi gibi çeşitli alanlarda kullanılmaya başlamıştır. Bazı şirketler (ör. Google ve Facebook) ayrıca büyük miktarda veriyi

günlük olarak toplamak ve analiz etmek yoluyla derin öğrenme ile ilgili projeleri başlatmıştır [9]. Eğitimlerinin zaman alıcı olmasına rağmen sonuçları oldukça başarılıdır.

3.4.1. Nesne takip yöntemlerinin sonuçlarını değerlendirme

Yapılan nesne takip çalışmalarında sonuçların değerlendirilmesi için en yaygın kullanılan metotlardan biri hassasiyettir (Precision) [10]. Bu metot video üzerinde çalıştırılan yöntemin belirli olan tüm nesnelerden bulduğu nesnelerin doğru bulunma oranı şeklinde tanımlanabilir. Sık kullanılan diğer başarı ölçme metodu ise anımsamadır (Recall). Bu metod algoritmada bulunan nesnelerin kaçının doğru olduğunu ölçer. Bu metotların hesaplanmasında True Positive (TP), False Positive (FP), False Negative (FN), True Negative (TN) değerleri kullanılmaktadır. Değerler Tablo 1’de açıklanmıştır [11].

Tablo 3.1. True Positive (TP), False Positive (FP), False Negative (FN), True Negative (TN) değerlerin hesaplanması

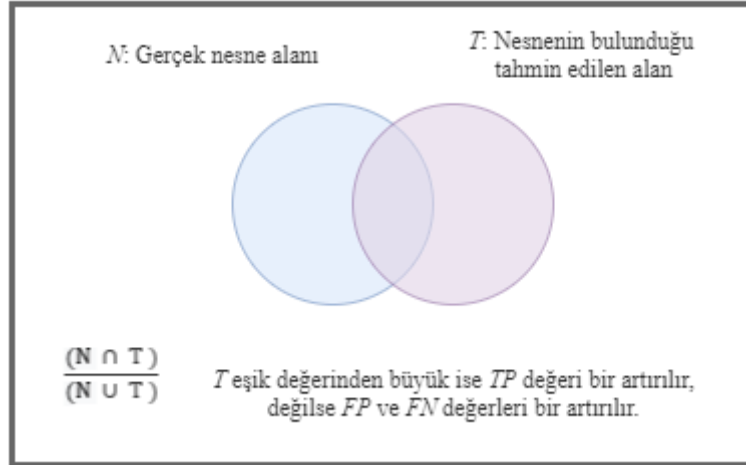
	Gerçek değer: Nesne var	Gerçek değer: Nesne yok
Algoritma tahmini: Nesne var	TP	FP
Algoritma tahmini: Nesne yok	FN	TN

TP : Algoritmamızın nesne olarak bulduğu ve nesnenin olduğu durum sayısı, FN : Algoritmamızın nesne olarak bulamadığı fakat nesne olduğu durum sayısı, FP : Algoritmamızın nesne olarak bulduğu fakat nesne olmadığı durum sayısı, TN :

Algoritmamızın nesne olarak bulamadığı ve nesnenin de olmadığı durum sayısı olarak hesaplanır.

Nesne takip algoritmaları farklı nedenlerden dolayı nesneyi bire bir örtüşecek şekilde bulamayabilir. Bu durumda bulunan şeyin nesne olup olmaması durumu değerlendirme işlemi aşağıdaki 4 adımla yapılabilir:

1. Algoritmada nesne olarak bulunan alan ile nesnenin gerçekten bulunduğu alanın kesişimi sonucu elde edilen alanın piksel sayısı hesaplanır ($N \cap T$).
2. Algoritmada nesne olarak bulunan alan ile nesnenin gerçekten bulunduğu alanın birleşimi ile elde edilen alanın piksel sayısı hesaplanır ($N \cup T$).
3. Kesişen alan birleşim alanına bölünür.
4. Elde edilen oran, daha önceden belirlenmiş olan T eşik oranı ile kıyaslanır. Eğer T değerinden büyük ise tespit etme durumu kabul edilir (TP artırılır), değilse ret edilir (FP , FN değerleri artırılır) (Şekil 1).



Şekil 3.3. Kesişme durumunda TP , FP ve FN değerlerin hesaplanması

Recall ve Precision değerleri TP , FP , FN , TN değerlerinden faydalanılarak aşağıdaki matematiksel formül ile hesaplanır:

$$Recall = \frac{TP}{TP+FN} \quad Precision = \frac{TP}{TP+FP} \quad (3.1.)$$

Yukarıdaki performans ölçme kriterleri nesne takip algoritmalarının son aşamalarını ifade etmektedir. Literatürde bunlarda farklı ölçütler de bulunmaktadır. Nesne takip yönteminin performansı kurulan algoritma içerisindeki işlem adımlarına bağlıdır. Özellikle işlenmemiş veriden anlamlı özelliklerin çıkartılması ve nesneye ait ayırt edici özelliklerin algılanabilmesi problemin en önemli kısmıdır.

3.4.2.Literatürde nesne tespiti ve takibi için kullanılan yöntemler

Literatürde detaylı bir inceleme yapıldığında nesne takibi çalışmalarının 4 aşamada incelendiği görülmektedir. Bu aşamalar ön işleme, nesne tespiti, nesne sınıflandırma ve nesne takibi olarak sıralanabilir (Şekil 3).



Şekil 3.4. Nesne takip algoritmaların genel diyagramı

Bu aşamalardan özellikle nesne tespiti önemli bir yere sahiptir. Çünkü modelin genel performansı bu aşamaya bağlıdır. Bu işlem genel olarak nesnenin belirlenmesi ve işlenecek olan nesnenin arka plandan ayrılması olarak açıklanabilir. Bir diğer ifade ile bu aşamada bir segmentasyon işlemi gerçekleştirilir.

3.4.2.1. Veri ön işleme

Verileri kullanım amacına uygun bir şekilde hazırlamak ve uygulamanın ilerleyişinde herhangi bir engel teşkil etmeyecek hale getirilmesidir. Bu aşamada veri türüne ve kullanım amacına göre farklı yöntemler kullanılabilir. Amaç algoritmanın doğru bir şekilde eğitilmesini sağlayacak veri setini ortaya çıkarmaktır. Veri ön işlemede kullanılan yöntemlere birkaç örnek vermek gerekirse; Regresyon, esikleme, kümeleme, filtreleme, binning, karar ağaçları vb. diyebiliriz. Önemli bir diğer konu ise

bazen bu algoritmalarından biri tek başına yeterli olmayabilir, böyle durumlardan birden çok algoritmanın peş peşe kullanılması söz konusudur.

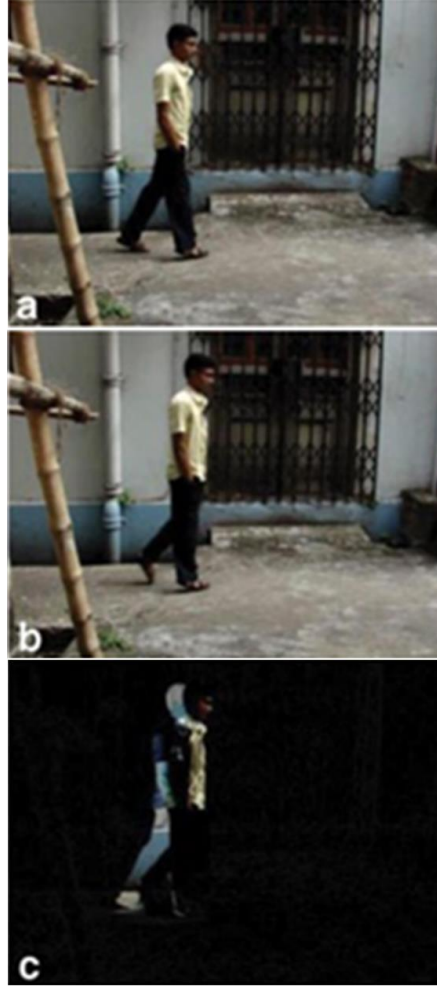
3.4.2.2. Nesne tespiti

Belirli bir sınıfa ait nesneyi ve bunun örneklerini algılamakla ilgilidir. Bilgisayarla görme ve görüntü işlemeden farklı olarak algılanan nesnenin görüntü üzerinde koordinatlarının bulunmasını içerir. Nesne tespiti aşamasıyla ilgili literatürde oldukça fazla sayıda yöntem bulunmaktadır. Bu yöntemleri genel itibariyle aşağıdaki gibi 3 gruba ayırabiliriz:

1. Çerçeveler arasındaki fark
2. Optik akış
3. Arka plan modeli çıkartma

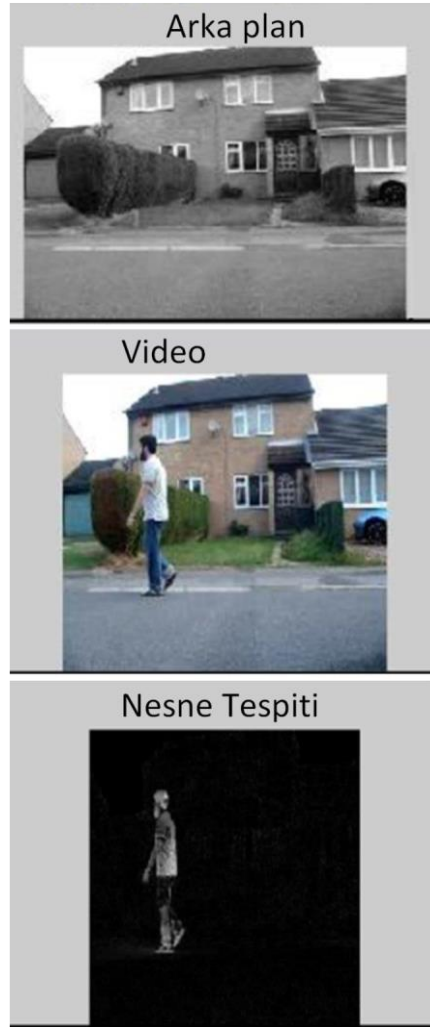
Bir görsel verideki nesnenin tespiti için iki temel bilgi kullanılır. Bunlar görsel öznitelik (renk, doku, şekil vb.) ve hareket bilgileridir. Bu bilgilerin elde edilmesi ve amaca uygun kullanımı için belli bir işlem gücü ve zaman harcanmaktadır. Nesne tespitinde kullanılan bilginin çeşitliliği başarıyı arttırırken aynı zamanda işlem süresinde de bir artışa sebep olmaktadır. Bu durum gerçek zamanlı uygulamalarda nesne takibi için göz ardı edilemeyecek kadar önemli bir problemdir.

Bu dökümanda açıklanacak olan nesne takip yöntemlerinden ilki ve en çok kullanılanlardan başlıca geleni videoda art arda gelen iki görüntü arasındaki geçici değişiklikleri bulma yöntemidir (temporal frame differencing). Bir video verisindeki art arda gelen sahnelerin birbirinden çıkartılmasıyla elde edilen bilgidir (Şekil 4). Bu yöntem oldukça basit ve hızlıdır fakat ışık değişimi ve gürültüye karşı da çok hassastır.



Şekil 3.5. Geçici değişiklikler yöntemi, a: $t-1$ zamanındaki görüntü (I_{t-1}), b: t zamanındaki görüntü (I_t), c: t zamanındaki fark görüntüsü $Nt=(I_t)-(I_{t-1})$ [12]

Bir nesne tespiti uygulamasından kullanılan geçici değişiklikler yönteminin hata oranı yüksek çıkabilmektedir. Bu yüzden bu yöntem nazaran daha az hata oranına sahip yöntemlerden biri olan arka plan çıkartma yöntemi kullanılmak istenebilir. Bu yöntemde belirlenmiş nesnenin olduğu görüntüden sadece arka planın çıkarılmasıyla elde edilen fark görüntüsü belli bir eşik değerde geçilir ve istenmeyen nesneler elenir. Bu yöntemle elde edilen yeni görüntüdeki gürültüden kurtulmak için çeşitli morfolojik işlemler yapılır ve istenen nesne ortaya çıkarılmış olur (Şekil 5).



Şekil 3.6. Bir video çerçevesindeki nesne tespiti için arka plan görüntüsü çıkartma yöntemi [13]

Nesne tespiti için oluşturulacak iyi bir arka plan modelinin değişken ışıklı ortamlarda başarı gösterebilmesi gerekmektedir. Arka plan görüntü modeli için tek tip arka plan görüntüsü kullanıldığında model model ışık değişimine karşı oldukça hassas olacaktır bu da modelin başarı oranını düşürür.

Arka plan ve nesnenin ortaya çıkartılması ile nesne tespiti yapan bir diğer yöntem ise optik akış yöntemidir. Bu yöntem ile art arda gelen iki görüntüde piksel hareketliliğine bakılarak arka plan çıkartılmakta ve bu şekilde nesne tespiti yapılabilmektedir. Bu yöntemin başarısı homojen olmayan arka plan görüntülerinde düşmektedir.

3.4.2.3. Nesne sınıflandırma

Görüntü işlemede sınıflandırma, nesnelerin kendine has öznitelikleri dikkate alınarak sınıflara ayrılması işlemidir. Kullanılan öznitelik nesnenin tanımlanması için kullanılan sayısal değerlerden oluşmaktadır. Bu değerler bir pikselin değeri veya bir görüntüdeki ortalama yoğunluk değeri gibi anlamlı ifadeler olabilmektedir. Nesne takibi veya sınıflandırması yöntemlerinde öznitelik seçimi önemli bir adımdır. Öznitelikler seçilirken başarıyı arttırmasının yanı sıra getirdiği işlem yüküne de bakılması gerekmektedir.

Sınıflandırma algoritmaları nesne takip yöntemlerinde önemli bir yere sahiptir. Özellikle çoklu nesne takip yöntemlerinde takip edilen her bir nesnenin diğer nesnelerle karıştırılmaması için sınıflandırma algoritmaları kullanılır. Literatürde nesne takibi için en yaygın kullanılan öznitelikler renk, kenar, doku, derinlik, süper piksel, hareket ve optik akış şeklinde sıralanabilir. Literatürde yapılan bazı çalışmalarda bu öznitelikler birlikte de kullanılmaktadır.

3.4.2.4. Nesne takibi

Nesne takibi, video kaydı içerisinde hareket halinde olan bir nesnenin hareket yörüngesini tahmin etme işlemidir. Nesne takip yöntemleri 3 kategoriye ayrılır: 1) nokta tabanlı; 2) çekirdek tabanlı; 3) silüet tabanlı.

Nokta tabanlı yöntemde takip edilecek nesne noktalar ile ifade edilir. Bu noktaların bir sonraki görüntüdeki konumları ve birbirine olana uzaklıkları gibi verilerin sonraki görüntüde de birbirine paralel devam etmesi gerekir. Bu yöntemde temel amaç nesnenin video çerçevesi içinde tespit edilmesi ve bir önceki çerçevede kullanılan nokta benzerliklerin hesaplamasıdır.

Çekirdek tabanlı yöntemlerde bir geometrik şekil yardımıyla takip edilecek nesne çerçevelenir. Bu çerçeve içerisinde bulunan nesne parçasının anlamlı bilgileri hesaplanarak başlangıçtaki şekil yardımıyla nesne takip edilir. Bu yöntemde nesnenin şeklinden ziyade kullanılan geometrik şeklin içerisinde bulunan nesne bilgilerinin

ıkarılması yeterli olabilmektedir. Bu Őekil iinde bulunan piksellerin hesaplanan olasılık yoęunluk bilgileri veya histogram zellikleri gibi bilgileri sonraki video erevelerinde takip edilebilmektedir.

Siluet tabanlı yntemler genellikle takip edilen nesnenin insan ya da hayvan gibi belli bir geometrik Őekille ifade edilemedięi durumlarda kullanılır. Bu yntemin temel amacı nesneyi tanımlayacak kenar bilgisi ya da Őekil bilgisi ıkartılarak sonraki imgelerde bu bilgiyi aramaktır. Bu yntem Őekil deęiŐiklięine karŐı olduka hassas olmaktadır.

ekirdek ve siluet tabanlı yntemler kıyaslandığında, ekirdek tabanlı yntemlerin daha dŐk iŐlem zamanına ve daha yksek baŐarı oranlarına sahip oldukları grlmektedir. Bu sebepten dolayı alıŐmalarda ekirdek tabanlı yntemler geniŐ bir kullanım alanına sahiptir. Nokta tabanlı yntemler dięer yntemlere oranla daha dŐk iŐlem zamanına sahip olmakla birlikte daha dŐk baŐarı oranına sahiptirler.

BÖLÜM 4. PROJE FİKRİNİN UYGULANMASI

4.1. Kullanılan Teknolojiler

4.1.1. Python ve OpenCV

90'lı yıllarından başında geliştirilmeye başlanan Python programlama dili, adını MonthyPython isimli komedi grubundan almaktadır. Nesne yönelimli, modüler, etkileşimli ve yorumsal bir seviyede dil olan bu programlama dilinde makine mantığına yatkın olarak daha hızlı çalışma gerçekleşmektedir. Ancak, makine mantığından yaklaşıp, insan mantığından uzaklaşmaya başlamak ile de daha zor bir dil haline gelmektedir. Bir programlama dilinin makine mantığı yerine daha çok insan mantığına yakınlaşması, bu dilin yüksek seviyede bir dil olduğunu göstermektedir. Python dili de bu özelliği ile yüksek seviyede bir programlama dilidir. Diğer birçok programlama diline göre öğrenilmesi kolay olan Python'un, yorumsal olması nedeniyle diğer dillerin aksine derlenme gereksinimi bulunmamaktadır. Program geliştirmeyi daha kolay hale getirdiği gibi bilgisayardaki görme sorunlarını çözmede kullanılan OpenCV Python, sayısal işlemler için optimize edilen ve Python bağlamalarından oluşan bir kütüphanedir.

OpenCV açık kaynak kodlu bir görüntü işleme kütüphanesidir. İlk olarak C dili kullanılarak geliştirilmiş daha sonra geliştirilmeye C++ dili ile devam edilmiştir. Java, C++, Python dilleriyle kullanılabilir. OpenCV kütüphanesinde ayrıca derin sinir ağı modülü bulunmaktadır. Bu sayede istenildiği takdirde başka bir yapay zeka kütüphanesi kullanmadan sadece OpenCV kullanılarak hem görüntü işleme hem bilgisayarlı görü işlemlerini gerçekleştirebiliriz. OpenCV'de bulunan başlıca modüller şunlardır:

- **Core:** OpenCV'nin temel fonksiyonlarını ve size, matris, point gibi veri yapılarını bulundurur.
- **HighGui:** Resim görüntüleme, pencereleri yönetme ve grafiksel kullanıcı arabirimleri için gerekli olabilecek metotları barındırır. Bu modül ile örnek tek bir fonksiyon ile bir resmi görüntüleyebiliriz.

- **Imgproc:** Filtreleme operatörleri, kenar bulma, nesne belirleme, renk uzayı yönetimi, renk yönetimi ve eşikleme gibi neredeyse tüm fonksiyonları içine alan bir pakettir.
- **Imgcodecs:** Dosya sistemi üzerinden resim ve video okuma/yazma işlemlerini yerine getiren metotları barındırmaktadır.
- **Videoio:** Kameralara ve video cihazlarına erişmek ve görüntü almak ve görüntü yazmak için gerekli metotları barındırır. OpenCV 3 sürümü öncesinde bu paketteki birçok metot video paketi içerisindeydi. [14]

OpenCV Python kütüphanesinde 2000'den fazla algoritma yer almaktadır. Bunlar, yüz tanıma, hareketleri tespit etme, nesneleri sınıflandırma ve ayırt etme, plaka tanıma, görüntü üzerinde işlem yapma, görüntüleri karşılaştırma gibi işlemleri rahatlıkla yapabilen algoritmalarlardır. Projede kullanılan fonksiyonlardan bahsetmek gerekirse :

VideoCapture()

Bu fonksiyonu ile görüntüyü yakalama işlemi yapıyoruz. Bu fonksiyonun içine 0 değeri verilirse birincil kameramızdaki görüntüyü alacaktır (ikincil bir kamera varsa 1 de yazılabilir). Kameramızın çözünürlük değerini öğrenmek için genişlik ve yükseklik değerlerini alıyoruz. Bu değerler video kaydederken işimize yarayacaktır. [15]

Rectangle() Fonksiyonu: Imgproc modülünde bulunuyor. Bu fonksiyon ile bir resim üzerinde istenilen yerlere dikdörtgenler çizdirebiliriz. En genel haliyle tanımı şu şekildedir: void rectangle(Mat mat, Point nokta1, Point nokta2, Scalar renk, int kalınlık) mat parametresi üzerinde işlem yapılacak resmi ifade eder. nokta1 çizeceğimiz dikdörtgenin başlangıç sınırlarını ifade eden Point türünden bir değerdir. Point sınıfının yapıcı fonksiyonu x ve y koordinatlarını alır. nokta1'e çizeceğimiz dikdörtgenin sırasıyla x ve y başlangıç konumlarını veriyoruz. nokta2'ye ise bitim noktalarını veriyoruz. renk parametremiz Scalar türünden bir değer. Scalar kullanılan renk uzayındaki katman sayısına göre farklı sayıda parametre alıyor. OpenCV bu renk uzaylarını bilindik sıralamada kullanmıyor. RGB uzayından bir renk tanımlarken BGR sırasıyla, RGBA renk uzayından bir renk tanımlarken ABGR sırasıyla tanımlamalıyız.

Son parametremiz olan kalınlık da int türünden bir değer. Aldığı değerle doğru orantılı olarak çizdiğimiz dikdörtgenin çizgi kalınlığının artmasını sağlıyor. [16]

putText() Fonksiyonu: Rectangle fonksiyonu gibi Imgproc modülünde yer alıyor ve onunkine benzer bir işlevi var. Bir resmin üzerine istenilen konumda yazı yazdırılmasına yarıyor. void putText(Mat resim, Point baslangic, int yazıSitili, int yazıBoyutu, Scalar renk) İlk parametreye üstünde işlem yapılacak resmi, ikinci parametreye resmin hangi noktasından itibaren yazının yazdırılacağını, 3. parametreye yazı stilini veriyoruz. 14 Yazı stilleri ImgProc sınıfında static üye değişken olarak tanımlı. Buraya direkt olarak tamsayı değeri yazmamıza gerek yok. 4. parametremiz tamsayı değeri alıyor, yazının boyutunu belirlememize yarıyor. 5. ve son parametremiz ile yazının rengini belirliyoruz. [16]

cv2.imread() Fonksiyonu: Belirtilen dosyadan bir görüntü yükler. Görüntü okunamıyorsa (eksik dosya, uygun olmayan izinler, desteklenmeyen veya geçersiz biçim nedeniyle) bu yöntem boş bir matris döndürür. İki parametre alır ilk parametre yüklenecek görüntünün yolunu alır ikinci parametre ise resmin nasıl okunacağını belirtir, belirtilmezse varsayılan değeri alır.

cv2.imshow() Fonksiyonu: Görüntüyü bir pencerede görüntülemek için kullanılır. Pencere otomatik olarak görüntü boyutuna sığar. İki parametre alır ilk parametreye görüntülenecek pencereye verilen isim, ikinci parametreye ise görüntülenecek resmin bulunduğu değişken yazılır.

cv2.waitKey() Fonksiyonu: Bu metod, cv2.imshow() ile görüntülediğimiz resmin ekranda kalma süresini belirler. Parantez içine ms cinsinden görüntünün ekranda kalma süresini belirleyebiliriz.

cv2.waitKey(1500) → 1.5sn ekranda görünür ve kapanır.

cv2.waitKey() / cv2.waitKey(0) → Bir klavye hareketine kadar görüntü ekranda kalır.

OpenCV DNN Modülü: DNN modülü ile birlikte popüler Derin Öğrenme framework'leri OpenCV ile birlikte kullanılır hale gelmiştir. Bu sayede OpenCV ile çeşitli kütüphanelerin önceden eğitilmiş çıkarım yapabilen çıktıları -modelleri kolayca ve platform bağımsız olarak kullanılabilir. DNN modülünden 3 adet fonksiyon kullanılmıştır, bunlar:

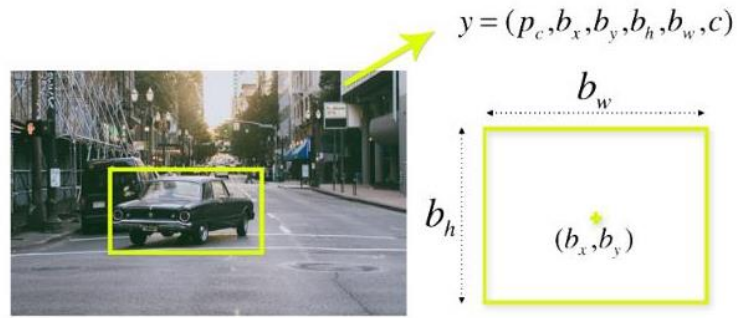
- **cv2.dnn.blobFromImage():** Görüntülerin okunarak sinir ağı girişine hazırlayan okuma fonksiyonudur.
- **cv2.dnn.readFromDarknet():** Kullanılacak modeli algoritmaya dahil etmek için kullanılır. Parametre olarak konfigürasyon(cfg) ve ağırlık(weights) dosyalarının yollarını alır.
- **cv2.dnn.NMSBoxes():** Non Maximum Supression yöntemini uygulamak için gerekli fonksiyondur. En yüksek güvenilirliğe sahip bounding box'ların idlerini döndürür ve bir listeye atar.

4.1.2.Kullanılan Algoritma

Son yıllarda, geliştirilen derin öğrenme teknikleri nesne tespiti için şaşırtacak derecede hızlı sonuçlar verebilmektedir. Bu tekniklerden biri ve bizim de projemizde kullanacağımız ise YOLOv4 algoritmasıdır. “You Only Look Once” yani YOLO, gerçek zamanlı olarak nesne tespitini tek bir uçtan-uca modelle gerçekleştirebilen, ve verimli sonuçlar elde eden CNN (Evrişimsel Sinir Ağları) ailesindendir.

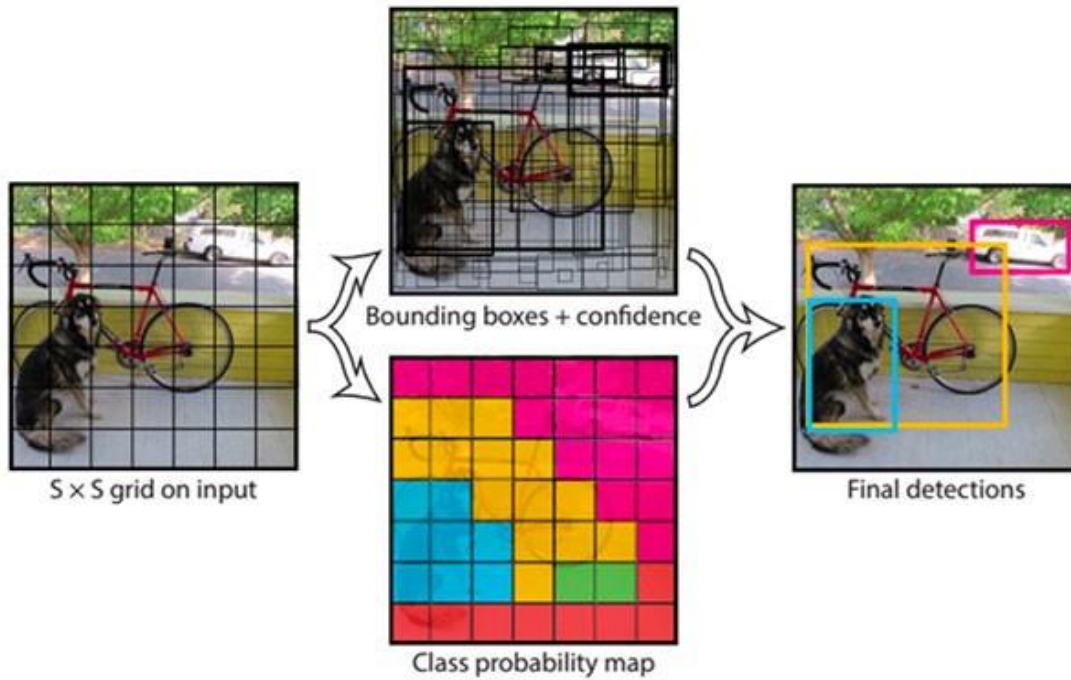
Nesne algılamadaki amaç bir görüntüde ilgili nesneyi bulmak ve onu kutucuk içerisinde belirtmektir. YOLO algoritmasını anlayabilmek için öncelikle neyin tahmin edildiğini belirlemek gerekir. Sonuçta, nesnelerle dolu bir görselde tespit edilmesini istediğimiz nesnenin konumunu belirten sınırlayıcı kutuyu tahmin etmeyi amaçlamaktayız. Ayrıca nesne konumunun belirlenmesinden öncesinde de nesnenin hangi sınıfa ait olduğunun tahmininin yapılması gerekmektedir. Nesne konumunu belirten her sınırlayıcı kutu dört özellik kullanarak tanımlanabilir:

- sınırlayıcı kutunun merkezi (bx, by)
- genişlik (bw)
- yükseklik (bh)
- c değeri (bir nesnenin sınıfına karşılık gelir. Örneğin: araba, trafik ışıkları, vb.).



Şekil 4.1. YOLO algoritmasındaki sınırlayıcı kutucukların özellikleri [17]

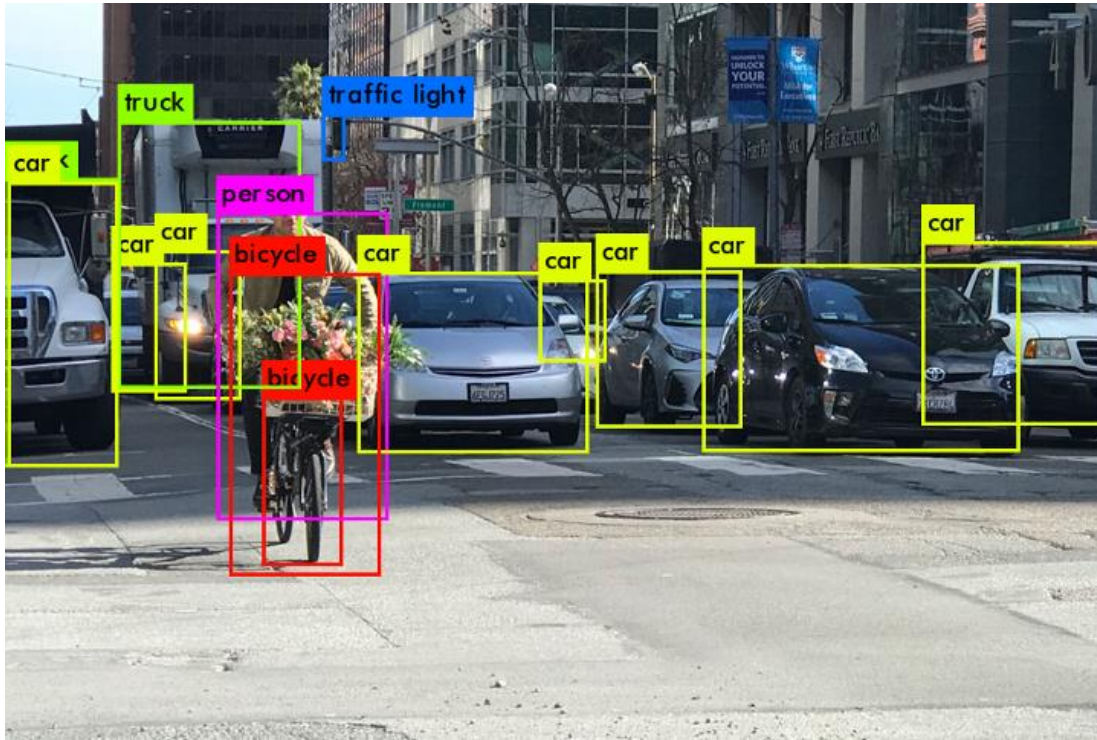
Bunlara dışında sınırlayıcı kutuda bir nesne olma ihtimaline karşılık gelen pc değeri de tahmin edilmelidir.



Şekil 4.2. YOLO algoritmasının çalışma mantığı [18]

YOLO algoritması ile çalışılırken görüntüde nesne içerebilme ihtimali olan bölgeler aranmak yerine görüntü $N \times N$ 'lik ızgaralara bölünür ve her ızgara kendi içerisinde nesne olup olmadığını ve nesne var olduğunu düşünüyorsa merkez noktasının kendi

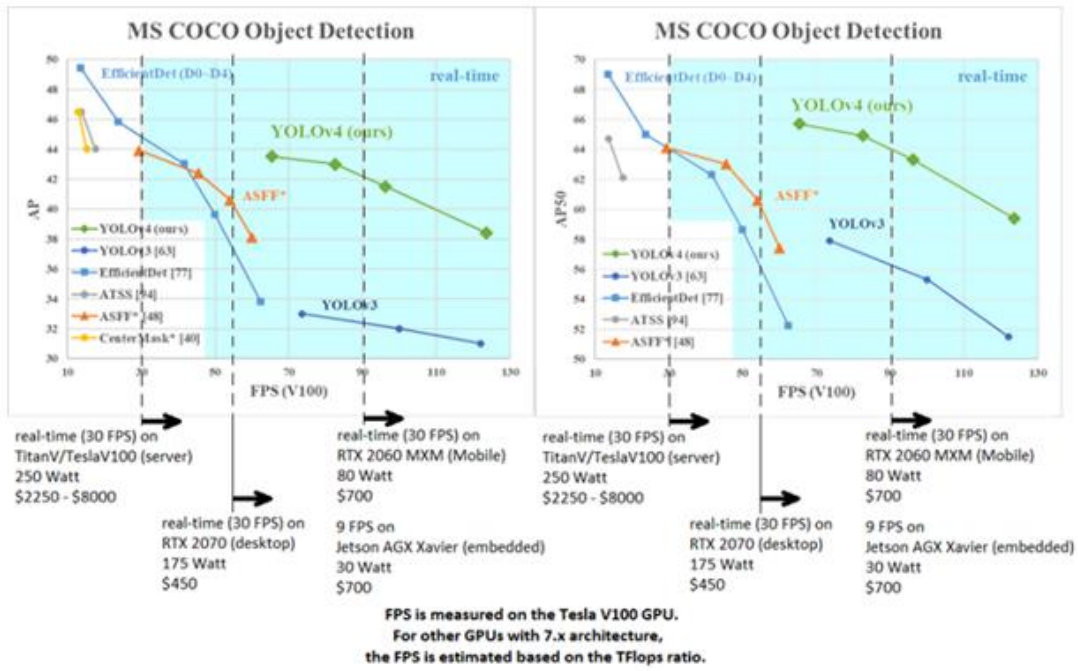
alanında olup olmadığını düşünmektedir [19]. Nesnenin merkez noktasına sahip olduğuna karar veren ızgara o nesnenin sınıfını, yüksekliğini ve genişliğini bulup o nesnenin çevresine bounding box çizmelidir [19]. Normal olarak bu hücrelerin çoğu bir nesne içermeyecektir. Bundan dolayı nesne bulundurma ihtimali düşük olan sınırlayıcı kutuları kaldırmaya yarayan *pc* değerinin tahmin edilmesi önem arz etmektedir.



Şekil 4.3. YOLO tarafından tespit edilen nesneler [19]

Son derece iyi performanslı ve gerçek zamanlı olarak 9000’den fazla sınıfın nesnelerini algılayabilen YOLOv3 birçok yöntemden daha iyi sonuçlara sahip son teknoloji bir nesne algılama sistemidir [20]. Ayrıca YOLO bilgisayar programlama temeli olan herkesin anlayabileceği şekilde açık ve iyi tanımlanmış bir yöntemdir. Hatta temelinde tanımlanmamış sınıflara ait nesnelerin nasıl sisteme dahil edileceğini, yeni bir modelin nasıl işleneceğini ve eğitileceğini göstererek farklı amaçlara uyarlamaya imkan tanır.

Aşağıdaki grafiklerde YOLO ve diğer bazı algoritmaların MS COCO veri seti için nesne tespiti performanslarını görmekteyiz.



Şekil 4.4. YOLO ve diğer nesne tespit algoritmalarının MS COCO veri seti üzerindeki performanslarının karşılaştırması [19]

Grafiklerde de görüldüğü gibi sınıflandırıcı sayısının eşit olduğu bir case düşünürsek YOLOv4 rakiplerine göre neredeyse 3 kat fark atmış durumda [19].

Bu nedenle projemizi gerçekleştirirken YOLOv3'ü temel amaç olarak kullanarak, KKD kullanım durumunu otomatik olarak takip eden yeni bir model eğittik. Bu amaçla en önemli ve yaygın olarak kullanım şartı bulunan KKD eşyası olan kaskların işçiler tarafında takılı olup olmamasının tespiti ile ilgilendik.

4.1.3. Google Colab

Herhangi bir kullanıcının düzenleyicisine kaynak kodu yazmasına ve tarayıcıdan çalıştırmasına izin veren bir IDE'dir. Google Colab Jupyter Notebook'un Google'ın bulut sunucularında barındırılan ve birden fazla kullanışlı özelliğe sahip, güçlendirilmiş bir sürümüdür. Özellikle Python programlama dilini destekler ve makine öğrenimi görevlerine, veri analizine, eğitim projelerine vb. yöneliktir. Verileri eğitmek için oldukça kullanışlıdır çünkü Tesla K-80 GPU'sunu sağlamaktadır. Bu sayede eğitim işlemi oldukça hızlı ve doğru bir şekilde gerçekleştirilebilecektir.

4.1.4 Anaconda&Spyder IDE

Anaconda, veri bilimi ve benzeri bilimsel uygulamalar için Python kullanmak isteyenlere hazırlanmış tümleşik bir Python dağıtımıdır. Veri bilimi, yapay zeka vb konularında sıkça kullanılan kütüphanelerin yanı sıra Jupyter Notebook ve Spyder gibi araçları da barındırır.

Spyder, Python ile yazılmış ve Python geliştirme için kullanabileceğiniz, açık kaynak bir IDE'dir. Spyder, hata ayıklama işlemi, gelişmiş düzenleme yapabilme, etkileşimli test edebilme özelliklerinden dolayı Python için oldukça güçlü bir geliştirme ortamıdır.

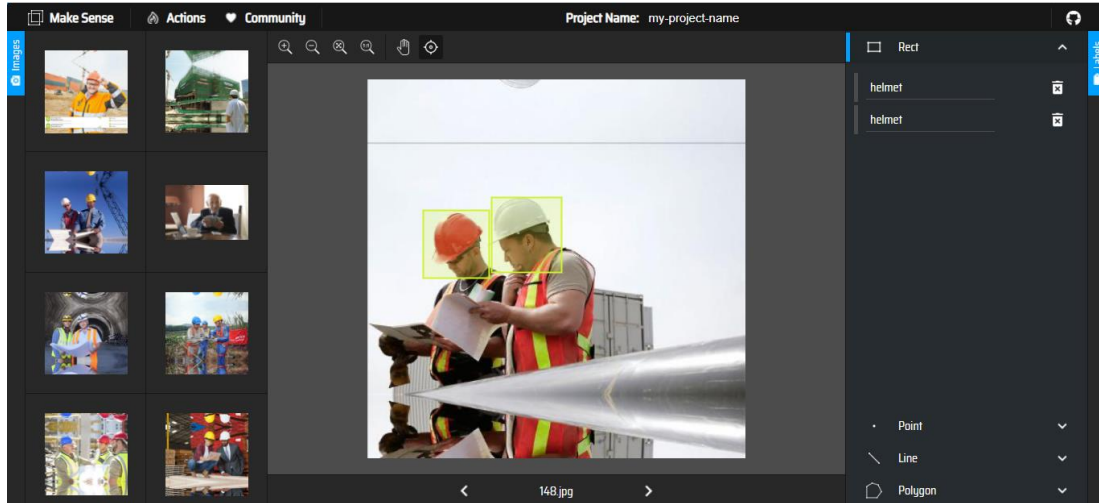
4.1.5.Darknet

Darknet, açık kaynaklı bir sinir ağı çerçevesidir. Gerçek zamanlı nesne algılama (görüntüler için de kullanılabilir) için hızlı ve son derece doğru bir framework'tür. Hızlı olmasının en önemli nedeni C ve CUDA ile yazılmasıdır.

4.2. Deney Düzenegi

4.2.1. Veri Seti

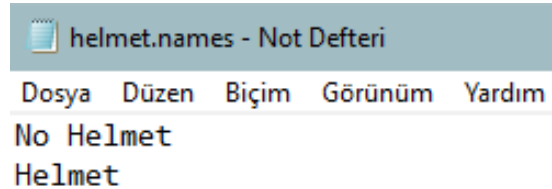
Diğer ML algoritmalarında da olduğu gibi YOLO için de bilinmeyen bir nesnenin nasıl görüldüğünü öğrenmesini sağlayacak bir veri seti gereklidir. Proje kapsamında toplamda 950 adet görüntü kullanılmıştır. Kaskın nasıl görüldüğünü makineye öğretmek amacıyla 758 adet görüntü kullanılmıştır. Modeli test etme işlemi de aynı veri setinden geriye kalan 192 adet görüntü ile yapılmıştır. Tüm görüntüler Kaggle'dan ve daha birçok farklı siteden elde edildi. Görüntülerdeki kaskın konumu makinenin öğrenmesi amacıyla manuel olarak belirtildi. Etiketleme yapılırken makesense isimli web uygulaması kullanıldı.



Şekil 4.5. Makesense'in nesne algılama için resim etiketleme arayüzü

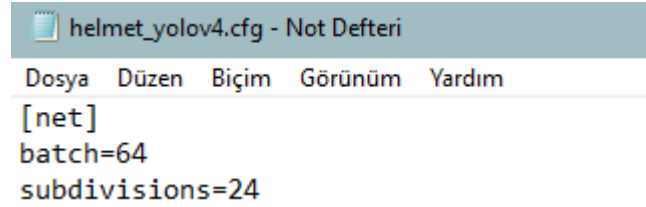
4.2.2. Konfigürasyon İşlemleri

İlk olarak Github'dan YOLO'yu kullanarak eğitim yapmamıza ortam sağlayan darknet dosyaları indirilmiştir. Daha sonra data klasöründe name uzantılı dosya oluşturulmuştur ve buraya class isimleri satır satır girilmiştir. Bizim modelimizde “No Helmet” ve “Helmet” olarak toplamda 2 adet class kullanılmıştır.



Şekil 4.6. helmet.names dosyası içeriği

Daha sonra darknet ile birlikte gelen “yolov4.cfg” dosyasında verilerimizin train edilebilmesi için özel ayarlamalar yapılmıştır. Bu ayarlamalardan biri de batch ve subdivision değerlerini eğitimimize uygun bir şekilde değiştirmektir. Bacth her iteration da kaç resim üzerinden geçileceğini gösterir [21]. Subdivision ise batchleri mini-batch gruplarına ayırır [21]. Max batches sayısı ise class sayısı*2000 formülünden yola çıkarak 4000 olarak belirlemiştir, bu aynı zamanda bizim iterasyon sayımızdır.



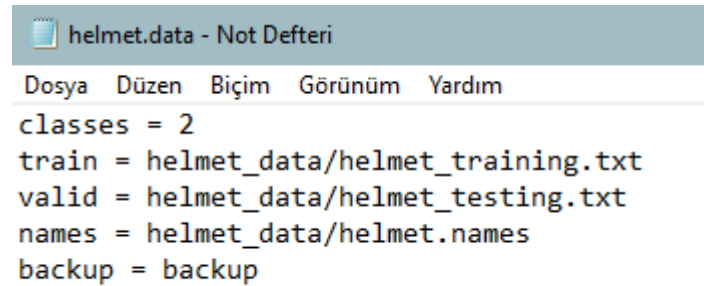
```
helmet_yolov4.cfg - Not Defteri
Dosya  Düzen  Biçim  Görünüm  Yardım
[net]
batch=64
subdivisions=24
```

Şekil 4.7. Konfigürasyon dosyası içerisindeki batch ve subdivisions değerleri

Ayrıca bu .cfg uzantılı konfigürasyon dosyamızda her bir yolo katmanındaki class değişkeninin değerini train edeceğimiz sınıf sayısı ile değiştirilir ve her bir yolo katmanının üzerindeki katmanda filters değerini class sayımıza göre değiştirilmiştir.

$$\text{Yolov4 için filters} = (\text{Number of class} + 5) * 3$$

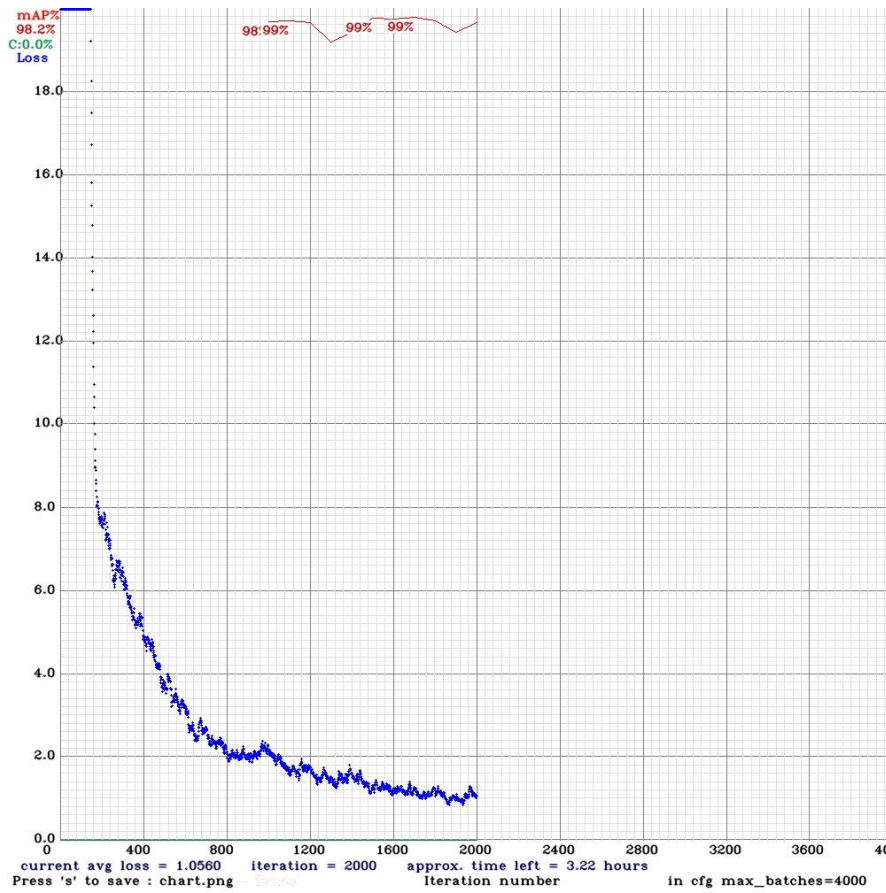
Sonraki adımda data klasörü içerisinde .data uzantılı bir dosya oluşturularak burada class sayısı, train.txt, test.txt ve names dosyalarının yolu verilmiştir.



```
helmet.data - Not Defteri
Dosya  Düzen  Biçim  Görünüm  Yardım
classes = 2
train = helmet_data/helmet_training.txt
valid = helmet_data/helmet_testing.txt
names = helmet_data/helmet.names
backup = backup
```

Şekil 4.8. helmet.data dosyası içeriği

4.2.3. Eğitim ve Sonuç



Şekil 4.9. Eğitimin 4. Saatine ait loss grafiği

YOLOv4 algoritmasıyla ağ bağlantısı ve 8 GB RAM' e sahip bir bilgisayar ile model yaklaşık 7 saat boyunca eğitildi. Modelin eğitimi bittiğinde makine görüntüden kaskı tespit edebilir duruma geldi.

```
detections_count = 592, unique_truth_count = 444
class_id = 0, name = No Helmet, ap = 95.86% (TP = 152, FP = 20)
class_id = 1, name = Helmet, ap = 98.63% (TP = 281, FP = 15)

for conf_thresh = 0.25, precision = 0.93, recall = 0.98, F1-score = 0.95
for conf_thresh = 0.25, TP = 433, FP = 35, FN = 11, average IoU = 77.40 %

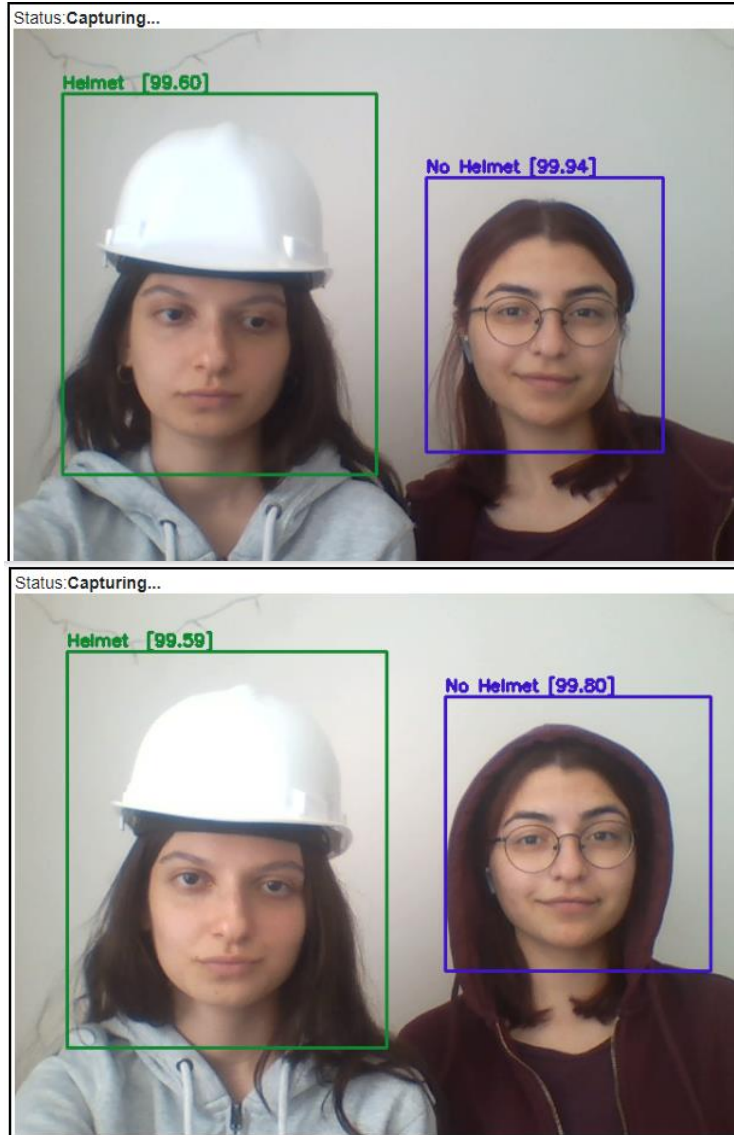
IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
mean average precision (mAP@0.50) = 0.972428, or 97.24 %
Total Detection Time: 20 Seconds
```

Şekil 4.10. Colab'de elde edilen, modele ait istatistiki bilgiler



Şekil 4.11. Helmet Detector'un resim üzerinden bareti tespit etmesine dair ekran görüntüsü

Aynı zamanda modelimizi bir kameranın akışına uygulayarak da test ettik, bu işlemi öncelikle Spyder isimli editörde gerçekleştirdik ancak video akış hızı çok yavaş olduğu için sonrasında Google Colab'de Tesla K80 GPU'su üzerinden aynı işlemi gerçekleştirdik ve daha iyi sonuçlar aldık. Uygulamamız gerçek zamanlı olarak nesne tespiti yapar ve algılama hızı şu an için yeterli seviyededir.



Şekil 4.12. Helmet Detector'un bilgisayar kamerası üzerinden gerçek zamanlı nesne tespiti

Modelin eğitimi bitti sırada ortalama hata (current avg loss) değeri 0.5661'e kadar düşmüştür. Genel olarak projeye bakıldığında bir iş yeri ortamında ki işçilerin kask takıp takmadığının tespitini yapan bir kamera mevcuttur. Bu kamera kasklı bir insanı algıladığında her hangi bir reaksiyon vermezken kasksız bir işçi algıladığı takdirde eğer 20 saniye içerisinde işçi kaskını takmazsa İSG sorumlusunun kullandığı KKD Takip Sistemi'nin mobil ve web platformlarını uyarı bildirimi göndermektedir.

BÖLÜM 5. SONUÇLAR VE ÖNERİLER

Projemizde, YOLO ile gerçek zamanlı nesne tanıma işlemi yapılmıştır. Tanımlanan KKD'nin olması gerektiği gibi takılıp takılmadığını kontrol eden bir sistem geliştirilmiştir. Bu sayede, işyerlerinde gerçekleşebilecek kaza hasarları en aza indirgenecek ve manuel insan kontrolüne göre çok daha güvenli bir hal alacaktır. Çünkü bu uygulama sayesinde çalışma saatleri içerisinde kimlerin KKD kullanmadığı belirlenecek ve uyarı gönderilecektir. Çalışanları sadece fiziksel olarak korumakla kalmayıp, oluşacak daha sağlıklı ortam ile birlikte ruhsal olarak da motive edecektir.

Bu geliştirilen uygulama ile uzun ve kısa vadeli oluşabilecek kazaların önüne geçilmesi ve işçinin çalışma ortamındaki verimini artırırken tehlike oranının düşürülmesi planlanmıştır. Kontrol yetersizliği ve ihmaller ortadan kalktığında KKD kullanımının ne derece hayat kurtardığı gösterilmektedir. Yapılan çalışmada, DL algoritmaları kullanılarak yeterli büyüklükte bir veri seti ile model, bir insanın KKD kullanımını algılayabilecek seviyeye getirilmiştir.

Bu çalışmada YOLO gerçek zamanlı nesne tanımlama yapılmıştır çünkü YOLO diğer algoritmalara göre daha hızlı ve küçük nesnelerin tespitinde daha iyi sonuç vermektedir. YOLO algoritması görüntüyü eşit parçalara bölmekte ve bu kesit alanı içerisinde nesnenin varlığını kontrol etmektedir.

Sonuç olarak, kamera görüntülerini alıp işleyerek, oluşturulan model ile işçilerin kask takıp takmadığını kontrol eden bir proje tasarlanmıştır. Bu alanda daha fazla çalışma yapılması ve yapılan çalışmaların da desteklenmesi önemli olacaktır. Ayrıca bu çalışmaların farklı alanlara uyarlanması birçok konuda kolaylaştırıcı uygulamalar geliştirilebilir.

KAYNAKLAR

- [1] Avcı, İ. & Yıldırım, M. (2021). Görme Engelli Bireyler İçin Derin Öğrenme Tabanlı Nesne Tanıma Modeli. Avrupa Bilim ve Teknoloji Dergisi, (28), 220-227.
- [2] TÜLÜ M. (2014). İş Sağlığı ve Güvenliği Hizmetlerinde İSG Profesyonellerinin Algı ve Beklentileri. Genel Bilgiler, s. 2-3.
- [3] C.B. Souto Maior, J.M. Santana, L.M. Nascimento, J.B. Macedo, M.C. Moura & D.L. Isis (2018). Personal protective equipment detection in industrial facilities using camera video streaming (Yayımlanmamış doktora tezi). Üretim Mühendisliği Bölümü, Brazilya.
- [4] Hjeltnæs E., Kee L.B., Face Detection: A Survey, Computer vision and image understanding, 83, 236–274., 2001
- [5] Anagnostopoulos C.-N.E., Anagnostopoulos I.E., Psoroulas I.D., Loumos, V., Kayafas, E., License Plate Recognition From Still Images and Video Sequences: A Survey, IEEE Transactions on intelligent transportation systems, 9, 3:377–391, 2008
- [6] Wang D., Unsupervised video segmentation based on watersheds and temporal tracking, IEEE Transactions on Circuits and Systems for video Technology, 8.5:539–546, 1998
- [7] Risha K.P., Kumar A.C., Novel Method of Detecting Moving Object in Video, Procedia Technology, 24:1055–1060, 2016
- [6] Avidan S., Support vector tracking, IEEE transactions on pattern analysis and machine intelligence, 26.8: 1064-1072, 2004
- [8] Avidan S., Support vector tracking, IEEE transactions on pattern analysis and machine intelligence, 26.8: 1064-1072, 2004
- [9] Chen Y., Yang X., Zhong B., Pan S., et al., CNNTracker: Online discriminative object tracking via deep convolutional neural network, Applied Soft Computing, 38:1088–1098, 2016
- [10] Hanbay ve Üzen, Nesne tespit ve takip metotları: Kapsamlı bir derleme, Tr. Doğa ve Fen Derg. – Tr. J. Nature Sci. 2017 Vol. 6 No. 2
- [11] Szegedy C., Liu W., Jia Y., Sermanet P., Going Deeper With Convolutions, In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1–9, 2015

- [12] Shaikh S.H., Saeed K., Chaki N., Moving Object Detection Approaches, Challenges and Object Tracking, In: Moving Object Detection Using Background Subtraction. Springer International Publishing, p. 5-14, 2014
- [13] Aldhaheri A.R., Edirisinghe E.A., Detection and Classification of a Moving Object in a Video Stream, In: Proc. of the Intl. Conf. on Advances in Computing and Information Technology-ACIT, 2014.
- [14] <https://pythondunyasi.com/opencv-python-nedir/>, Eriřim Tarihi: 25.12.2021.
- [15] <https://senolomer0.medium.com/opencv-i%CC%87le-g%C3%B6r%C3%BCnt%C3%BC-i%CC%87%C5%9Fleme-1-d7879e2386e1>, Eriřim Tarihi: 25.12.2021.
- [16] AKBAL A. Y., Tensorflow ve OpenCV Kütüphanesi ile Android Platformunda Nesne Tespiti, Sakarya Üniversitesi, 2019
- [17] <https://medium.com/teknopar-akademi/yolo-ile-opencv-ve-python-kullanarak-obje-tan%C4%B1ma-6303f0376c25>, Eriřim Tarihi: 23.12.2025.
- [18] <https://medium.com/operations-management-t%C3%BCrkiye/rcnn-yolo-opencv-ile-nesne-tan%C4%B1ma-fa3856e69fb3>, Eriřim Tarihi: 23.12.2025.
- [19] <https://www.odakarge.com/yolo-nedir.html>, Eriřim Tarihi: 22.12.2025.
- [20] <https://ichi.pro/tr/yolo-yolov2-ve-simdi-yolov3-ile-gercek-zamanli-nesne-algilama-44743782178952>, Eriřim Tarihi: 22.12.2025.
- [21] <https://medium.com/@ahmetxgenc/yolo-darknet-ile-object-detection-19f21d8eb31b>, Eriřim Tarihi: 25.12.2021.

ÖZGEÇMİŞLER

Şeyda Gündoğdu, 18.04.1996'da İstanbul'da doğdu. İlk, orta ve lise eğitimini İstanbul'da tamamladı. 2017 yılında İstanbul Üniversitesi, Peyzaj Mimarlığı Bölümü'nde lisans eğitimine başladı. 2019 yılında ise Sakarya Üniversitesi, Bilişim Sistemleri Mühendisliği Bölümü'ne geçiş yaptı. 2021 yılında CRS Soft Yazılım Hizmetleri A.Ş.'de Web Programlama alanında yazılım stajını yapmıştır. Hâlâ Sakarya Üniversitesi, Bilişim Sistemleri Mühendisliği Bölümü'nde lisans eğitimine devam etmektedir.

Esra Aksu, 03.02.1999'da İstanbul'da doğdu. İlk, orta ve lise eğitimini Üsküdar'da tamamladı. 2017 yılında Çamlıca Kız Anadolu İmam Hatip Lisesi'nden mezun oldu. 2017 yılında Işın Üniversitesi Endüstri Mühendisliği Bölümü'nü kazandı. Ancak 2019 yılında Sakarya Üniversitesi Bilişim Sistemleri Mühendisliği Bölümü'ne geçiş yaptı. SAÜ Bilişim Sistemleri Mühendisliği Bölümü'nde öğrenimine devam etmektedir.

Liva Nur Pulat, 14.08.1999 da Bolu'da doğdu. İlk,orta ve lise eğitimini Bolu'da tamamladı. 2017 yılında Bolu Anadolu İmam Hatip Lisesi'nden mezun oldu. 2018 yılında Gazi Üniversitesi Endüstriyel Tasarım Mühendisliği Bölümü'nü kazandı. 2019 yılında Sakarya Üniversitesi Bilişim Sistemleri Mühendisliği'ne geçiş yaptı. 2020 yılında Bolçi Çikolata A.Ş.'de İşletme Stajı'nı ve 2021 yılında Hira&Parl Yazılım ve Bilişim Şirketi'nde Yazılım Stajı'nı yapmıştır. Sakarya Üniversitesi Bilişim Sistemleri Mühendisliği Bölümü'nde eğitime devam etmektedir.

ISE 402 BİTİRME ÇALIŞMASI DEĞERLENDİRME VE SÖZLÜ SINAV TUTANAĞI

KONU :

ÖĞRENCİLER (Öğrenci No/AD/SOYAD):

Değerlendirme Konusu	İstenenler	Not Aralığı	Not
Yazılı Çalışma			
Çalışma klavuza uygun olarak hazırlanmış mı?	x	0-5	
Teknik Yönden			
Problemin tanımı yapılmış mı?	x	0-5	
Geliştirilecek yazılımın/donanımın mimarisini içeren blok şeması (yazılımlar için veri akış şeması (dfd) da olabilir) çizilerek açıklanmış mı?			
Blok şemadaki birimler arasındaki bilgi akışına ait model/gösterim var mı?			
Yazılımın gereksinim listesi oluşturulmuş mu?			
Kullanılan/kullanılması düşünülen araçlar/teknolojiler anlatılmış mı?			
Donanımların programlanması/konfigürasyonu için yazılım gereksinimleri belirtilmiş mi?			
UML ile modelleme yapılmış mı?			
Veritabanları kullanılmış ise kavramsal model çıkarılmış mı? (Varlık ilişki modeli, noSQL kavramsal modelleri v.b.)			
Projeye yönelik iş-zaman çizelgesi çıkarılarak maliyet analizi yapılmış mı?			
Donanım bileşenlerinin maliyet analizi (prototip-adetli seri üretim vb.) çıkarılmış mı?			
Donanım için gerekli enerji analizi (minimum-uyku-aktif-maksimum) yapılmış mı?			
Grup çalışmalarında grup üyelerinin görev tanımları verilmiş mi (iş-zaman çizelgesinde belirtilebilir)?			
Sürüm denetim sistemi (Version Control System; Git, Subversion v.s.) kullanılmış mı?			
Sistemin genel testi için uygulanan metotlar ve iyileştirme süreçlerinin dökümü verilmiş mi?			
Yazılımın sızma testi yapılmış mı?			
Performans testi yapılmış mı?			
Tasarımın uygulamasında ortaya çıkan uyumsuzluklar ve aksaklıklar belirtilerek çözüm yöntemleri tartışılmış mı?			
Yapılan işlerin zorluk derecesi?	x	0-25	
Sözlü Sınav			
Yapılan sunum başarılı mı?	x	0-5	
Soruları yanıtlama yetkinliği?	x	0-20	
Devam Durumu			
Öğrenci dönem içerisindeki raporlarını düzenli olarak hazırladı mı?	x	0-5	
Diğer Maddeler			
Toplam			

DANIŞMAN (JÜRİ ADINA):

DANIŞMAN İMZASI: