**T.C.**

**SAKARYA ÜNİVERSİTESİ**

**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**

BSM 498 BİTİRME ÇALIŞMASI

BİLGİSAYARLI GÖRÜ  
İLE İŞ SAHASI GÜVENLİĞİ

G171210070 – Fatih GÜLŞEN

|  |  |  |
| --- | --- | --- |
| Fakülte Anabilim Dalı  Tez Danışmanı | :  : | BİLGİSAYAR MÜHENDİSLİĞİ  Dr. Ögr. Üyesi Serap ÇAKAR |

2021-2022 Bahar Dönemi

T.C.

SAKARYA ÜNİVERSİTESİ

**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**

BİLGİSAYARLI GÖRÜ  
İLE İŞ SAHASI GÜVENLİĞİ

BSM 498 - BİTİRME ÇALIŞMASI

Fatih GÜLŞEN

|  |  |  |
| --- | --- | --- |
| Fakülte Anabilim Dalı | : | BİLGİSAYAR MÜHENDİSLİĞİ |

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

|  |  |  |
| --- | --- | --- |
|  |  |  |
| ………………. | ………………. | ………………. |
| Jüri Başkanı | Üye | Üye |

ÖNSÖZ

Bu çalışmanın hazırlanması sürecinde, bana yardımcı olan, yol gösteren, emeği geçen hocam Dr. Ögr. Üyesi Serap ÇAKAR’a, bu zamana kadar bana maddi manevi destek olan başta Dila ŞENOL olmak üzere aileme, okul hayatım boyunca yanımda olan arkadaşlarıma teşekkür ederim.

İÇİNDEKİLER

[ÖNSÖZ iii](#_Toc104323269)

[İÇİNDEKİLER iv](#_Toc104323270)

[SİMGELER VE KISALTMALAR LİSTESİ vi](#_Toc104323271)

[ŞEKİLLER LİSTESİ vii](#_Toc104323272)

[TABLOLAR LİSTESİ ix](#_Toc104323273)

[ÖZET x](#_Toc104323274)

[BÖLÜM 1. GİRİŞ 1](#_Toc104323275)

[1.1. İş Sahası Güvenliği 1](#_Toc104323276)

[1.2. Literatür Çalışmaları 1](#_Toc104323277)

[BÖLÜM 2. BİLGİSAYARLI GÖRÜ MİMARİ YAPISI 5](#_Toc104323278)

[2.1. Görüntü İşleme 5](#_Toc104323279)

[2.1.1. Görüntü nedir? 5](#_Toc104323280)

[2.1.2. Görüntü işleme nedir? 5](#_Toc104323281)

[2.2. Bilgisayarlı Görü Nedir? 5](#_Toc104323282)

[2.2.1. Kullanım alanları 6](#_Toc104323283)

[2.2.2. Bilgisayarlı görünün iş güvenliğindeki yeri 6](#_Toc104323284)

[2.3. Derin Öğrenme 7](#_Toc104323285)

[2.3.1. Yapay sinir ağları 9](#_Toc104323286)

[2.3.1.1. Yapay sinir ağları temel bileşenleri 10](#_Toc104323287)

[2.3.1.2. Yapay sinir ağı modelleri 10](#_Toc104323288)

[2.4. Bilgisayarlı Görü Algoritmaları 11](#_Toc104323289)

[2.4.1. Evrişimli Sinir Ağları (CNN) 11](#_Toc104323290)

[2.4.1.1. Convolutional layer 12](#_Toc104323291)

[2.4.1.2. Activation layer 13](#_Toc104323292)

[2.4.1.3. Pooling layer 13](#_Toc104323293)

[2.4.1.4. Flattening layer 14](#_Toc104323294)

[2.4.1.5. Fully-Connected layer 15](#_Toc104323295)

[2.4.1.6. Yaygın uygulamalar 15](#_Toc104323296)

[2.4.2. Region based CNN (R-CNN) 16](#_Toc104323297)

[2.4.2.1. Seçici arama (Selective Search) 17](#_Toc104323298)

[2.4.2.2. Non-Max Supression 18](#_Toc104323299)

[2.4.2.3. Intersection over union (IoU) 18](#_Toc104323300)

[2.4.3. Fast R-CNN 19](#_Toc104323301)

[2.4.4. Faster R-CNN 19](#_Toc104323302)

[2.4.5. Mask R-CNN 20](#_Toc104323303)

[BÖLÜM 3. GELİŞTİRİLEN YAZILIM 23](#_Toc104323304)

[3.1. Kullanılan Teknolojiler 23](#_Toc104323305)

[3.1.1. Python dili 23](#_Toc104323306)

[3.1.1.1. OpenCV 23](#_Toc104323307)

[3.1.1.2. Torch 24](#_Toc104323308)

[3.2. Geliştirilen Algoritma 24](#_Toc104323309)

[3.3. Uygulamanın Kısaca Tanıtımı 27](#_Toc104323310)

[3.3.1. Vücut duruş yapısı tespiti 27](#_Toc104323311)

[3.3.2. Nesne ve insan tespiti 27](#_Toc104323312)

[3.3.3. Güvenlik alanı seçimi 27](#_Toc104323313)

[3.3.4. Uygulama içinden görüntüler 28](#_Toc104323314)

[BÖLÜM 4. MODELİN TEST EDİLMESİ VE DEĞERLENDİRİLMESİ 30](#_Toc104323315)

[4.1. Kullanılan Model 30](#_Toc104323316)

[4.1.1. ResNet-50 30](#_Toc104323317)

[4.2. Uygulama Performans Değerlendirmesi 36](#_Toc104323318)

[BÖLÜM 5. SONUÇ VE DEĞERLENDİRME 37](#_Toc104323319)

[KAYNAKLAR 38](#_Toc104323320)

[EKLER 43](#_Toc104323321)

[EK A: Yürütme planı 43](#_Toc104323322)

[ÖZGEÇMİŞ 44](#_Toc104323323)

[BSM 498 BİTİRME ÇALIŞMASI](#_Toc104323324)

[Değerlendİrme ve Sözlü Sınav Tutanağı 45](#_Toc104323325)

SİMGELER VE KISALTMALAR LİSTESİ

|  |  |
| --- | --- |
| CPU | : Central processing units |
| GPU | : Graphics processing units |
| RGB | : Red – Green - Blue |
| DSÖ | : Dünya sağlık örgütü |
| MR | : Manyetik rezonans |
| PET | : Positron emisyon tomogrofisi |
| HOG | : Histogram of Oriented Gradients |
| SVM | : Support Vector Machine |
| YOLO | : You Only Look Once |
| CNN | : Convolutional Neural Network |
| CV | : Computer Vision |
| R-CNN | : Region Base Convolutional Neural Network |
| SSD | : Single-Shot Detector |
| YSA | : Yapay sinir ağları |
| DNN | : Deep Neural Network |
| ReLu | : Rectified Linear Unit |
| IoU | : Intersection Over Union |
| RoI | : Region of Interest |
| RPN | : Region Proposal Network |
| API | : Application Programming Interface |
| 2D | : 2 Dimensions |

ŞEKİLLER LİSTESİ

[Şekil 1.1. İş Sahası Güvenliği 7](#_Toc104323348)

[Şekil 2.1. Derin öğrenme modelinin ürüne dönüşümü 8](#_Toc104323349)

[Şekil 2.2. Yapay Sinir Ağları Yapısı 9](#_Toc104323350)

[Şekil 2.3. Evrişimli Sinir Ağları Yapısı 11](#_Toc104323351)

[Şekil 2.4. Convolutional Layer Çalışma Örneği 12](#_Toc104323352)

[Şekil 2.5. Pooling Layer Çalışma Örneği 14](#_Toc104323353)

[Şekil 2.6. Flattening Layer Çalışma Örneği 14](#_Toc104323354)

[Şekil 2.7. LeNet-5 Mimarisi 15](#_Toc104323355)

[Şekil 2.8. AlexNet Mimarisi 16](#_Toc104323356)

[Şekil 2.9. R-CNN mimarisi 17](#_Toc104323357)

[Şekil 2.10. Seçici arama çalışma örneği 18](#_Toc104323358)

[Şekil 2.11. Fast R-CNN mimarisi 19](#_Toc104323359)

[Şekil 2.12. Faster R-CNN mimarisi 20](#_Toc104323360)

[Şekil 2.13. Mask R-CNN diğer ağlar ile gösterimi 21](#_Toc104323361)

[Şekil 2.14. Semantik ve Örnek Segmentasyon farkı 22](#_Toc104323362)

[Şekil 3.1. Setup Akış Diyagramı 25](#_Toc104323363)

[Şekil 3.2. Run Akış Diyagramı 26](#_Toc104323364)

[Şekil 3.3. Uygulama içi alan dışı görüntü 28](#_Toc104323365)

[Şekil 3.4. Uygulama içi sadece Keypoints alan içi görüntü 28](#_Toc104323366)

[Şekil 3.5. Uygulama içi alan içi görüntü 29](#_Toc104323367)

[Şekil 4.1. Görüntü Tanıma İçin Deep Residual Learning 31](#_Toc104323368)

[Şekil 4.2. Artık Blok Yapısı 31](#_Toc104323369)

[Şekil 4.3. 1x1 Evrişim katmanı içeren ve içermeyen ResNet Bloğu 33](#_Toc104323370)

[Şekil 4.4. ResNet Mimarisi 34](#_Toc104323371)

[Şekil 4.5. ResNet-50 mimarisi 35](#_Toc104323372)

[Şekil 4.6. ResNet karşılaştırma tablosu 35](#_Toc104323373)

TABLOLAR LİSTESİ

[Tablo 2.1. Biyolojik ve Yapay Sinir Ağları Karşılaştırma 9](#_Toc104323374)

[Tablo 4.1 Sistem değerlendirme FPS tablosu 36](#_Toc104323375)

ÖZET

Anahtar kelimeler: İş Güvenliği, Bilgisayarlı Görü, Derin Öğrenme

Günümüzde işletmeler çalışanlarının güvenlikleri için birçok önlem almaktadır. Mevcut çalışanlarına iş güvenliği eğitimleri ve güvenlik önlemleri (Güvenlik alanı, iş giysileri vb.) bunların başında gelmektedir. Bu eğitimler ve önlemler %100 etkili olmadığından iş kazaları kaçınılmazdır.

Bu gibi kazaların yaşanmaması adına artık yapay zekadan faydalanarak insan inisiyatifine bırakılmadan belli önlemler alınabilmektedir. Bunların başında yapay zekâ destekli iş güvenliği uygulamaları gelmektedir.

Bu çalışmada yapılan projenin amacı, kişilerin belirtilen alana girişinin tespitinin yapılmasıdır. Proje OpenCV ve PyTorch kullanılarak gerçekleştirilmiştir. Vücut duruş pozisyon tespiti ile can güvenliği açısından riskli bölgelere giren insanlar tespit edilerek denetlenir. Vücut duruş pozisyonu için ResNet-50 mimarili ve COCO veri seti ile eğitilmiş bir model tercih edilmiştir. Çevrede bulunan objelerin tespiti için Mask R-CNN kullanılmış olup bu modelde vücut duruş pozisyonu gibi ResNet-50 mimarisine sahip COCO veri seti ile eğitilmiştir.

# GİRİŞ

* 1. İş Sahası Güvenliği

İş sahası güvenliği ister masa başı ister ağır makinelerle çalışılıyor olsun belli başlı tehlikeleri kapsar. Her şirket güvenli çalışma ortamı sağlayarak kendisini, çalışanını ve müşterisini korumuş olur. Güvenliğin sağlanmadığı durumlarda özellikle ağır makinelerde çalışanlarda bir kaza anında ölüme varan sonuçlar yer alabilir.

Her iş yerinde bu güvenlikten sorumlu şahıslar veya departmanlar vardır. Bu şahıslar veya departmanlar büyük işletmelerde anlık kontrol sağlayamadıklarından son zamanlarda güvenlik kameraları ile kontrollerini sürdürmektedir. Bu güvenlik önlemlerinin zaman ve insan maliyeti artmasından dolayı görüntü işleme kullanılarak iş sahası güvenliği sağlamak için adımlar atılmaktadır [3].

* 1. Literatür Çalışmaları

Sendipan Dey tarafından yapılan nesne tespiti çalışması 4 farklı yöntemle ele alınmıştır. Bu yöntemler sırasıyla şu şekildedir:

* HOG/SVM
* YOLO v3
* Faster R-CNN
* Mask R-CNN

Bu yöntemleri anlatırken nasıl çalıştığına dair bilgiler verip görsel olarak bu yöntemlerin çalışma prensiplerini desteklemiştir. Eğitim aşamasında veri seti kullanmamıştır önceden eğitilmiş modelleri tercih etmiştir. Nasıl kodlanabileceğine dair bilgi vermiştir [6].

Michael Avendi yaptığı nesne tespiti çalışmasında bilgisayarlı görünün derin öğrenme içeren kısımlarına odaklanmıştır. Bu çalışmada Python diline ait PyTorch kütüphanesinden yararlanmıştır. Nesne tespiti yaparken nasıl veri setleri oluşturulacağını ve bunları nasıl okuyup eğitim yapılacağından bahsetmiştir. Bu çalışmada kodlamanın nasıl yapılması gerektiğine ayrıca yer vermiştir [7].

V Kishore Ayyadevara ve Yeshwanth Reddy’nin yaptığı çalışmada nesne tespitini 2 ayrı temel bölüme ayrılmıştır. Bu bölümlerden ilki basit nesne tespiti ikincisi ileri düzey nesne tespiti olarak geçmektedir. İlk bölümünde temel olarak veri seti oluşturma ve bu veri setini R-CNN ve Fast R-CNN temelli eğitiminden bahsedilmektedir. İkinci bölümde Faster R-CNN, YOLO ve SSD (Single-Shot Detector) temelli eğitimleri ele almıştır. Her iki bölümde de bu algoritmaların nasıl kodlanacağından ayrı ayrı bahsetmişlerdir [8].

Alexander Toshev ve Christian Szegendy tarafından yapılan çalışmada insan vücut duruşu DNN tabanlı regresyon problemi ile formüle edilmiştir. Yüksek hassasiyetli insan vücut duruşu tespiti sağlayan DNN regresörlerinden oluşan bir dizi sunar. Bu yaklaşım yayınlandıktan sonra birçok alanda bu algoritma kullanılmaya başlamıştır [25].

Ching-Hang Chen ve Deva Ramanan tarafından yapılan çalışmada CNN ile insan vücut pozisyonu algılanması. Bu algılanan pozisyonun 2 boyutlu olduğundan dolayı 3 boyutlu vücut pozisyonu içeren veri seti sayesinde eğitilen başka bir derin ağ ile bu 2 boyutlu vücut duruşunun 3 boyutluya dökülmesi sağlanmıştır [26].

M.G. Dorrer ve A.E. Tolmacheva tarafından yapılan çalışmada üretilen akıllı buzdolabının içerisindeki ürünleri kontrolü için bilgisayarlı görü test edilmiştir. Bu testlerde YOLOv3 ve Mask R-CNN performans olarak birbirleri ile karşılaştırılmıştır. Canlı görüntü üzerinde hızlı işlem yapabilmesi için testlerde Mask R-CNN daha başarılı olmuştur ve bu yüzden Mask R-CNN tercihi yapılmıştır [27].

Alexander Shustanov ve Pavel Yakimov tarafından yapılan çalışmada bilgisayarlı görü ile trafik işaret ve işaretçilerinin tanınması sağlanmıştır. Burada CNN tabanlı mimariler test edilmiştir ve sonuçları aktarılmıştır [28].

Mustafa Tüfekçi ve Doç. Dr. Fatih Karpan bu çalışmasında CNN, RNN, LTSM, RBM ve AE üzerinde testler gerçekleştirip bu mimarileri doğruluk yüzdesi olarak belli başlı metotlar ile değerlendirmeye almıştır. Testleri gerçekleştirdikten sonra bilgisayarlı görü için en iyi mimarinin çalışmanın gerçekleştirildiği koşullarda CNN olduğuna kanaat getirmişlerdir [29].

Firgat Muradli tarafından hazırlanan bu çalışmada derin öğrenme kullanarak insan duruşunun tespitini yapmıştır. Birkaç farklı veri setinin performanslarını değerlendirmiştir. Uygulama tarafında Python dilinden yararlanarak açık kaynak kodlu kütüphane olan Keras’ı tercih etmiştir [30].

Kaiming He, Georgia Gkioxari, Piotr Dollár ve Ross Girshick yaptığı ortak çalışmada Mask R-CNN COCO veri seti ile nasıl eğitebileceğini ve performans değerlendirmeleri yapılmıştır. Faster R-CNN ile eğitim ve tahmin açısından karşılaştırma yapılmıştır. İlk bu çalışma ile Mask R-CNN piyasaya sürülmüştür [39].

Bilgisayarlı görü sistemi ile iş güvenliği sağlayan birçok şirket vardır. Sercan Esen ve Serhat Çillidağ tarafından kurulan Intenseye firması iş güvenliğini bilgisayarlı görü sistemi ile sağlamaktadır. Şirket New York ve İstanbul’da sahip olduğu ofisler ile Arçelik, Borçelik, COAST ve King Hawaiian gibi firmalar ile çalışma ortaklığı göstermektedir [9].

2015 yılında Silikon Vadisinde Emrah Gültekin ve Hakan Gültekin tarafından kurulan CHOOCH firması da bilgisayarlı görü sistemi ile iş güvenliği ve birçok alanda hizmet vermektedir. Yapılan bu girişimin en önemli müşterilerini ABD Hükümeti ve Fortune 500 şirketleri yer almaktadır [10].

San Diego’da kurulan AlwaysAI.co firması da bilgisayarlı görü sistemi ile iş güvenliği ve depo yönetimi gibi birçok alanda faaliyet göstermektedir [11].

Carlos Anchia, Elizabeth Spears ve Logan Spears tarafından kurulan Plainsight firması tarım, tedarik zinciri, enerji ve ilaç sanayi gibi farklı alanlarda bilgisayarlı görü sistemi ile hizmet vermektedir. Tarım sektöründe bilgisayarlı görü sisteminin kullanılması ile canlı hayvan ve mahsul üreticilerinin verimliliklerinin hızlı ve kolay bir şekilde arttırılması ve daha fazla gelir elde edilmesi amaçlanmaktadır [12].

Bilgisayarlı görü sistemi kullanılarak düzenlenen yarışmalar dünyada çeşitlilik göstermektedir. İlk olarak, dünya da en çok kullanılan görüntü işleme kütüphanelerinden biri olan OpenCV tarafından gerçekleştirilen yarışma da 2021 yılında 400.000 USD ödül dağıtılmıştır. Düzenlenen bu yarışmanın sponsoru Intel ve Microsoft Azure’dur [13].

Bir diğer yarışma örneği olarak ise, COCO, her sene veri seti üzerinde iyileştirilmeler yapılması için yarışmalar düzenlemektedir. 2020 yılında yapılan yarışmada ödüller en iyi sonuç ve en iyi fikir olarak iki ayrı dalda verilmiştir [14].

Son olarak, ImageNet 2010 yılından 2017 yılına kadar Kaggle üzerinden düzenlenen yarışmalarda görüntü sınıflandırma ve konumlandırma üzerinde çalışılan en yaygın konulardır. Ayrıca Kaggle sitesi üzerinden bu gibi birçok yarışmalar gerçekleştirilmektedir [15,16].

# BİLGİSAYARLI GÖRÜ MİMARİ YAPISI

Bilgisayarlı görünün temelinde derin öğrenme ve YSA yatmaktadır. YSA temelli algoritmalar ile başarı gösterdiği için Bölüm 2.3 ve Bölüm 2.3.1 de bu mimariler ayrıntılı şekilde açıklanmıştır. Görüntü işleme Bölüm 2.1’de temel olarak giriş yapılmıştır.

* 1. Görüntü İşleme
     1. Görüntü nedir?

Görüntü, tek bir parça gibi gözükse bile birden fazla parçadan oluşmaktadır. Bu parçalara piksel denmektedir. Bu görüntüler 2 boyutlu matris olarak da ifade edilebilir. Her bir piksel renk yoğunlukları ayarlanarak oluşan yeni renklerin birleşmesi ile görüntüler oluşur. Günlük kullanımda sık sık karşılaşılan 1920x1080, 1280x720 gibi oranlar piksel sayısını belirtmektedir [2].

* + 1. Görüntü işleme nedir?

Görüntü işleme, bilgisayarların görüntüler üzerinde anlamlı ifadeler elde edilmesini sağlayan bilgisayar bilim dalıdır. Bu işlem pikseller üzerinde belli algoritmalara dayanarak uygulanan matematiksel işlemler ile gerçekleştirilir. Temel amacı insanların yeteneklerini kopyalayarak hız kazanıp maliyetlerin daha aşağı çekilmesini sağlamaktır [1],[2].

* 1. Bilgisayarlı Görü Nedir?

Bilgisayarlı görme, görüntü işleme ile birbirlerine çok benzeseler dahi aslında farklı şeylerdir. Temel olarak görüntü işleme mevcut görüntüden yeni görüntüler elde etmek (içeriği basitleştirme veya geliştirme gibi) iken bilgisayarlı görü görüntünün fotometrik (parlaklık, renk gibi) özelliklerini normalleştirme gibi örneklendirilebilir. Son zamanlarda makine öğrenmesi ve derin öğrenmenin daha basit kodlanabilir olması ve bilgisayarların (CPU ve GPU) daha çok gelişmesi ile sonuçların gerçek zamanlı alınabilmesinden dolayı çok daha fazla kullanılmaya başlanmıştır.

* + 1. Kullanım alanları

Bilgisayarlı görünün başlıca kullanım alanları şunlardır:

* Güvenlik: Gelişen donanım ve yapay zekâ algoritmaları ile birçok kurum bilgisayarlı görü güvenlik amacıyla tercih etmektedir. En yaygın kullanılan teknoloji olarak yüz tanıma sistemleri gösterilebilir [1],[2].
* Tıp: En geniş kullanım alanı olarak görülebilir. Bilgisayarlı tomografi, MR, PET alanlarında bilgisayarlı görü kullanılan başlıca alanlardır. Birçok sonucu büyük DSÖ veri tabanı sayesinde anında hata oranı çok düşük şekilde doktor kontrolü gerekmeden alınabilir [1],[2].
* Savunma sanayi: Askeri alanda bilgisayarlı görü teknolojilerine başvurulur. Uydu görüntülerinin iyileştirilmesi ve anlamlandırılması, hedeflerin tespiti ve takibinde kullanılır [1],[2].
* Üretim: Ürünlerin kalite kontrolü, ambalaj kontrolü ve araçların bakım kontrolünde sık sık kullanılır [1],[2].
* Otonom araçlar: Kendi kendine giden araçların etrafında olan biten olaylar için gerçek zamanlı olarak nesne tanımlama ve izleme yaparak aracı bu olaylara göre yönlendirir [1],[2].
  + 1. Bilgisayarlı görünün iş güvenliğindeki yeri

İş güvenliğinin her geçen gün öneminin artmasından ve personel zamanının çok değerli olmasından dolayı güvenliğin otomasyon kontrolüne geçmesi üzerinde çalışmalar vardır. Donanım ve kurulum ek masraflarından kaçınmak için olan sistem üzerinde en iyi çalışabilecek sistemler bilgisayarlı görü sistemleridir. Her şirketin çalışma alanlarında güvenlik kameraları mevcuttur. Bu güvenlik kameraları bilgisayarlı görü için çok güzel bir olanaktır. Bu yüzden son zamanlarda bulut teknolojisinin de gelişmesi ile bu çalışmaları yürüten birçok girişim ortaya çıkmıştır.

Şekil 1.1. de gösterilen fotoğraf Intenseye firmasının Area Control ürününden örnek bir görüntüdür. Burada bilgisayarlı görü ile iş güvenliği nasıl sağlandığının bir örneğidir [9].

iç mekan, cihaz içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 1.1. İş Sahası Güvenliği [9]

* 1. Derin Öğrenme

Derin öğrenme, verilen veri setinin sonuçlarını veren bir makine öğrenmesi yöntemidir. Derin öğrenme, makine öğrenmesi, yapay zekâ farklı kavramlardır. Derin öğrenme makine öğrenmesi alt dalı iken maine öğrenmesi yapay zekâ alt dalıdır [5].

Derin öğrenme eğitimi 3 farklı şekilde gerçekleştirilebilir.

Bunlar şu şekildedir:

- Gözetimli öğrenme

- Yarı gözetimli öğrenme

- Gözetimsiz öğrenme

Gözetimli öğrenmede veri seti etiketli olarak verilir. Gözetimsiz öğrenmede veri seti tamamen etiketsiz verilir ve kendi kendine öğrenmesi sağlanır. Yarı gözetimli öğrenmede gözetimli ve gözetimsiz öğrenmenin ortasında yer alır. Eğitim veri setinde daha geniş etiketsiz veri seti, sınıflandırma ve özellik çıkarımında etiketli veri seti kullanılır. Bunları örneklendirmemiz gerekir ise veri setimizde 100 fotoğraf olduğunu var sayalım. Gözetimli öğrenmede 80 fotoğraftaki kişileri belirleyip eğitmeliyiz ve geri kalan 20 fotoğrafın sonuçlarını isteyebilir. Gözetimsiz öğrenmede hiç etiketleme yapmadan 80 fotoğrafı eğitime sokarız ve geri kalan 20 fotoğraftaki kişileri sorarız. Bu bir kümeleme yaparak bize benzer sonuçları ortaya koyar. Yarı gözetimlide ise kişilerin bulunduğu sadece birkaç fotoğraf etiketleyerek 80 fotoğrafı eğitime sokarız ve geri kalan 20 fotoğrafın sonuçlarını gözlemleriz [4],[32].

Aşağıdaki yöntemler derin öğrenme algoritmalarına örnek olarak verilebilir:

* Convolutional Neural Network (CNN)
* Deep Neural Network (DNN)
* Deep Belief Network (DBN)
* Convolutional Deep Belief Network (CDBN)
* Deep Boltzmann Machines (DBM)

Bir derin öğrenme modelinin ürün olana kadar geçirdiği sürecin kısa bir çizimi Şekil 2.1.’de gösterildiği gibidir.



Şekil 2.1. Derin öğrenme modelinin ürüne dönüşümü [4]

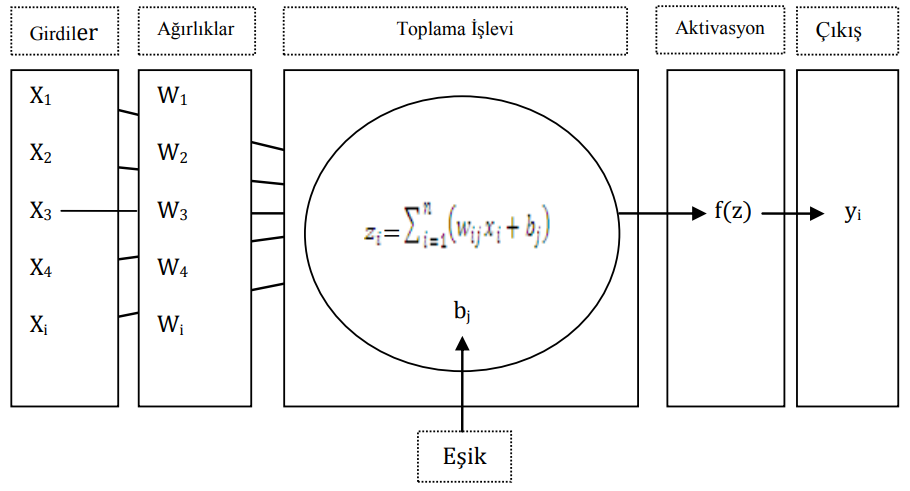
* + 1. Yapay sinir ağları

YSA, insan beyninin öğrenme yapısını taklit ederek öğrenme, hatırlama, genelleme yaparak yani veriler üretmeye yarayan yazılımlardır. YSA insan beynindeki sinir yapısından ilham alınarak geliştirilmiş bir yapıdır. YSA temel olarak 3 katmandan oluşur. Bu katmanlar girişi katmanı, çıkış katmanı ve gizli katmandır [21],[22],[23].

Yapay sinir ağları ile biyolojik sinir ağları karşılaştırması Tablo 2.1.’de gösterilmiştir.

Tablo 2.1. Biyolojik ve Yapay Sinir Ağları Karşılaştırma [21]

|  |  |
| --- | --- |
| **Biyolojik Sinir Ağları** | **Yapay Sinir Ağları** |
| Nöron | İşlemci Eleman |
| Dentrit | Toplama Fonksiyonu |
| Hücre Gövdesi | Transfer Fonksiyonu |
| Aksonlar | Yapay Nöron Çıkışı |
| Sinaps | Ağırlıklar |



Şekil 2.2. Yapay Sinir Ağları Yapısı [21]

* + - 1. Yapay sinir ağları temel bileşenleri

Yapay sinir ağları, öğrenme algoritması ve aktivasyon fonksiyonu olarak iki temel kısma ayrılır [21],[22],[23].

* Öğrenme algoritması: YSA’nın en önemli özelliklerin başında veriyi öğrenme gelir. Ağdaki veriler ağırlık diye adlandırdığımız uçlarda saklanır. Ağırlıklar veriler arasında en uygun değeri bulmaya çalışır bu da ağın eğitilmesi demektir [21],[22],[23].
* Aktivasyon fonksiyonu: Yapay sinir hücrelerine girdi olarak verilen değerin buna karşılık gelen çıktı değerini üretir. Bu fonksiyonlar genellikle doğrusal değillerdir. Fonksiyonları verilere ve algoritmalara uygun seçilmesi önemlidir. Bu fonksiyonlar doğrudan öğrenmeye etki ederler [21],[22],[23].
  + - 1. Yapay sinir ağı modelleri

Yapay sinir ağı modelleri 2 farklı şekilde incelenebilir [21],[22],[23].

* İleri beslemeli yapay sinir ağları: İleri beslemeli YSA’larında girilen bilgiler ileri doğru yani girişten çıkışa doğru hareket eder. Bir katmandan gelen düğümler sadece kendinden sonra gelen katmana gider, ardından sonuç katmanına geçer. Giriş verileri giriş katmanında herhangi bir işleme uğramaz. Orta ve son katmanda işlemlere uğrarlar ve buraların bir ağırlık değeri olur [21],[22],[23].
* Geri beslemeli yapay sinir ağları: Geri beslemeli yapay sinir ağlarında en az bir düğümün çıkışı diğer düğümün girişi olarak verilir. Bundan dolayı gecikmeler genellikle elaman üzerinde yapılır. Besleme işlemi bir katmandaki düğümler üzerinde olmayabilir, bu da doğrusal ilişkiden söz edilememesi sağlamaktadır. Geri beseleme yapısı girilen veri setine göre değişiklik gösterebilir [21],[22],[23].
  1. Bilgisayarlı Görü Algoritmaları

Bilgisayarlı görü için kullanılan, en sık tercih edilen algoritmalar derin öğrenme temelli algoritmalar olmaktadır. En sık kullanılan mimarilerin başında CNN gelmektedir [17].

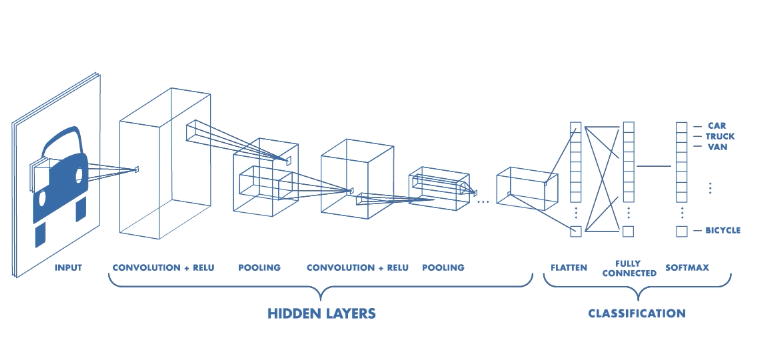
* + 1. Evrişimli Sinir Ağları (CNN)

Evrişimli sinir ağları derin öğrenmenin bir alt dalıdır. Bu sinir ağının 1 girişi 1 çıkışı bulunmaktadır. Bunun yanı sıra birden fazla gizli katman bünyesinde barındırır. Katmanları şu şekildedir:

* Convolutional Layer
* Activation Layer
* Pooling (Downsampling) Layer
* Flattening Layer
* Fully-Connected Layer

Bu katmanları sırasıyla incelemek gerekir [17],[18],[19],[20].

Evrişimli sinir ağları yapısı temel olarak Şekil 2.3’te gösterilmiştir.



Şekil 2.3. Evrişimli Sinir Ağları Yapısı [17]

* + - 1. Convolutional layer

Convolutional layer, temel olarak özellikleri saptamak için kullanılır. CNN’nin yapı taşıdır. CNN mimarisi ismini bu katmandan almaktadır. Bu katman görüntüyü ilk ele alır ve özellikleri saptamak için belli başlı filtreler uygular. Bu filtre örneğin kenar algılama, blur ekleme, görselleri keskinleştirme olabilir. Filtre uygulandıktan sonra oluşan yeni matrise Feature Map denir [17],[19],[20].

* Stride (Adım): Stride (Adım) değeri CNN modellerinde parametre olarak değiştirilebilir bir değerdir. Bu parametre filtrenin ana görsel üzerinde kaç piksel kayacağını belirler [17],[19],[20].
* Padding (Dolgu): Bir görsele filtre uygulandıktan sonra çıktı değerler orijinal görselden daha küçük boyutlarda olurlar. Bunu önlemek amacıyla padding uygulanır. Padding, çıkan görselin 4 tarafını tamamen 0’lar ile doldurur [17],[19],[20].

metin, skorbord içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 2.4. Convolutional Layer Çalışma Örneği [17]

* + - 1. Activation layer

Activation (Aktivasyon) layer isminin nedeni bu katmanda aktivasyon fonksiyonlarının kullanılmasından dolayıdır. Bu katmanda birçok aktivasyon fonksiyonu kullanılabilir. Geçmişte en sık kullanılanlar sigmoid ve tanh iken şu anda en sık kullanılan aktivasyon fonksiyonu ReLu fonksiyonudur [17],[19],[20].

ReLu temel olarak şu şekilde çalışır. Eğer giren değer 0 veya 0’dan küçük ise çıktı değerini 0 olarak ver. Eğer giren değer 0’dan büyük ise aynı şekilde çıktı ver. Denklem 2.1’de matematiksel olarak gösterilmiştir.

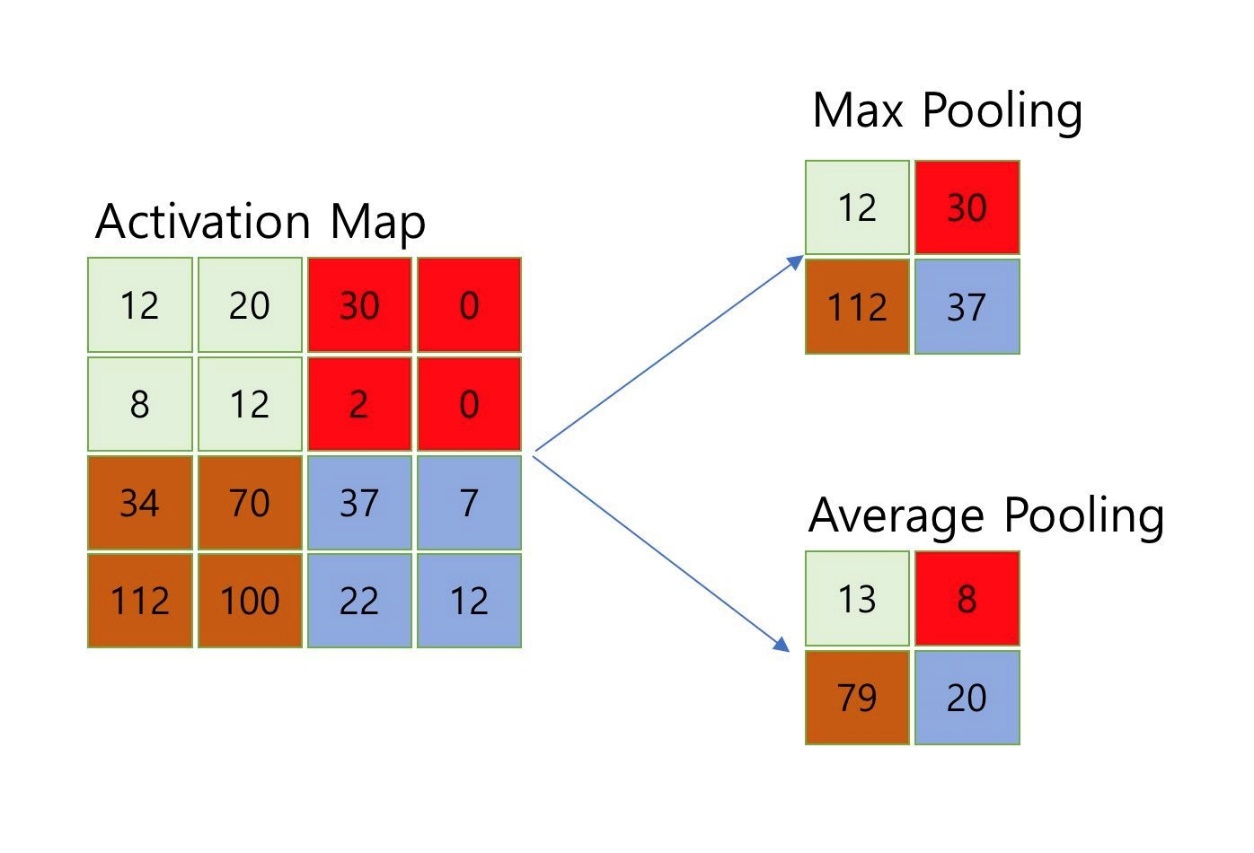
**(2.1)**

* + - 1. Pooling layer

Pooling layer’ın genel olarak görevi ağ içindeki hesaplama sayısı, parametre azaltmak için kullanılır. Birçok pooling işlemi vardır ama en sık kullanılanı Max Pooling işlemidir. Aynı prensip ile çalışan Average Pooling ve L2-norm Pooling algoritmaları da mevcuttur [17],[19],[20].

Bu katman çok sık kullanılan bir katman değildir. Bu katmanı birçok kişi tercih etmez. Bu katman yerine Convolutional layer katmanının Stride (Adım) değeri yükseltilmesi tercih edilir [17],[19],[20].

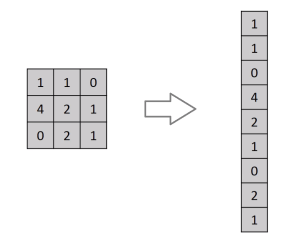
Şekil 2.5.’de pooling örneği gösterilmiştir. Burada uygulanan havuzlama boyutu 2x2 olarak örneklendirilmiştir.



Şekil 2.5. Pooling Layer Çalışma Örneği [17]

* + - 1. Flattening layer

Flattening layer görevi basit olarak son katmana verileri hazırlamaktır. Bu katmanın amacı gelen matrisi tek boyutlu matrise indirgeyip Fully-Connected layer’a göndermesidir. Tek boyuta indirgenmesindeki amaç Fully-Connected layer’ın tek boyutlu olarak giriş istemesidir [17],[19],[20].



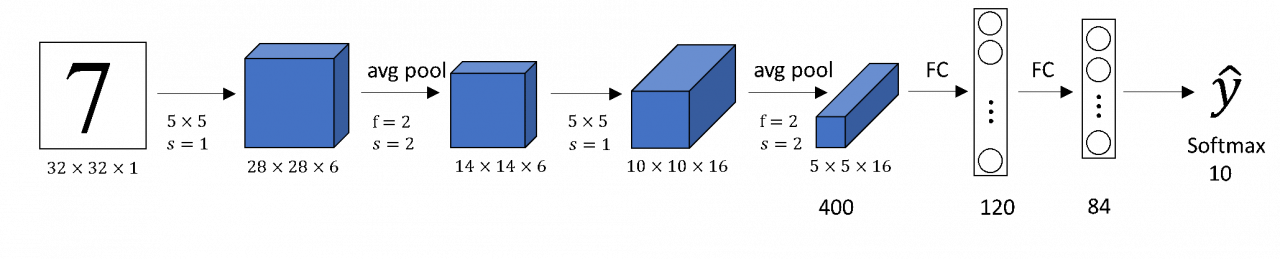
Şekil 2.6. Flattening Layer Çalışma Örneği [17]

* + - 1. Fully-Connected layer

Girdiler Flattening layer’dan sinir ağlarına uygun şekilde geldikten sonra klasik sinir ağları öğrenmesi gerçekleşir [17],[19],[20].

* + - 1. Yaygın uygulamalar
* LeNet: Bu ağ, Convolutional Networks’ün ilk başarılı uygulaması olarak görülür. 1990’larda Yann LeCun tarafından posta kodları ve basit basamakları okumak için geliştirilip kullanıldı [17],[19],[20].
* AlexNet: Bu ağ, 2012 yılında ImageNet ILSVRC yarışmasında sunuldu. Diğer ağlara nazaran çok daha iyi bir performans göstermiştir [17],[19],[20].
* GoogLeNet: ILSVRC 2014’ün kazanını bu ağ olmuştur. Ağdaki parametrelerin sayısını azaltmak amacıyla Average pooling katmanı kullanılmıştır [17],[19],[20].
* VGGNet: Bu ağ sayesinde ağ derinliğinin sinir ağları için ne kadar önemli olduğunu göstermiştir. 16 adet Convolutional layer bulunur [17],[19],[20].
* ResNet: ResNet 2015 yılında Kaiming He, Xiangyu Zhang, Shaoqing Ren ve Jian Sun tarafından tanıtılan bir mimaridir [45]. Daha az katman sayısı ve karmaşıklık ile yüksek hız ve başarı sağlamıştır.

LeNet mimari yapısı Şekil 2.7.’te, AlexNet mimari yapısı Şekil 2.8.’da gösterilmiştir.



Şekil 2.7. LeNet-5 Mimarisi [24]

metin, saat içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 2.8. AlexNet Mimarisi [25]

* + 1. Region based CNN (R-CNN)

CNN mimarisinde sonuçların sadece sınıf dönmesinden dolayı tek bir fotoğraf, video karesinde sadece tek bir sınıf tespit edilebilmektedir. Tek bir karede birden fazla sınıf olduğunda ise CNN bu problemleri çözememektedir. Bu yüzden R-CNN mimarisi bulunmuştur. R-CNN iki adımdan oluşmaktadır. İlk adım seçici arama ile 200 bölgeye ayrılır ve ikinci adımda her bir bölgeye ayrı ayrı CNN uygulanır [33],[35], [36].

metin, işaret içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 2.9. R-CNN mimarisi [35]

* + - 1. Seçici arama (Selective Search)

Seçici arama görüntülerde yakalanması gereken bölgelerin belirlenmesi için kullanılan bir algoritmadır. Bu algoritma sabit bir algoritmadır bu kısımda hiçbir öğrenme gerçekleşmez [33],[35].

İlk olarak küçük bölgeler belirlenir ve ardından birbirine benzeyen bölgeler birleştirilir. İterasyon yapıldığından dolayı bu olay birçok kez tekrarlanır ve bölgeler kümelenmiş olur. Bölgeler R-CNN için belirlendikten sonra bu bölgelerin her biri CNN ağlarına giriş olarak verilir. Bu algoritma sonucunda yaklaşık 2000 bölge elde edilir ve bu 2000 bölgeye tek tek CNN uygulanır. CNN ağlarının çıktıları SVM modellerinde sınıfı belirlenir ve regresyon modellerinde sınırlayıcı kutular belirlenir [33],[35].



Şekil 2.10. Seçici arama çalışma örneği [33]

* + - 1. Non-Max Supression

Giriş yapılan görsellerde 2000 bölge olsa dahi bu bölgelerin hepsi kullanılmaz. Doğru sınırlayıcı kutuları (bounding box) belirlemek için non-max supression kullanılır [33],[35].

Non-Max supression elde edilen sınırlayıcı kutuların IoU değerleri 0,5’ten büyük olanları tutar küçük olanları ise bastırır. Eğer bir cisim için birden fazla 0,5 ten büyük değer çıkar ise sadece en büyük değere sahip olanı alır [33],[35].

* + - 1. Intersection over union (IoU)

IoU değeri sınırlayıcı kutuların doğruluk değeri olarak ifade edilebilir. Saptanan sınırlayıcı kutular ile gerçek sınırlayıcı kutular karşılaştırılır ve sonucuna IoU değeri denir. Bu değer Denklem 2.2. de gösterildiği şekilde hesaplanır [33],[35].

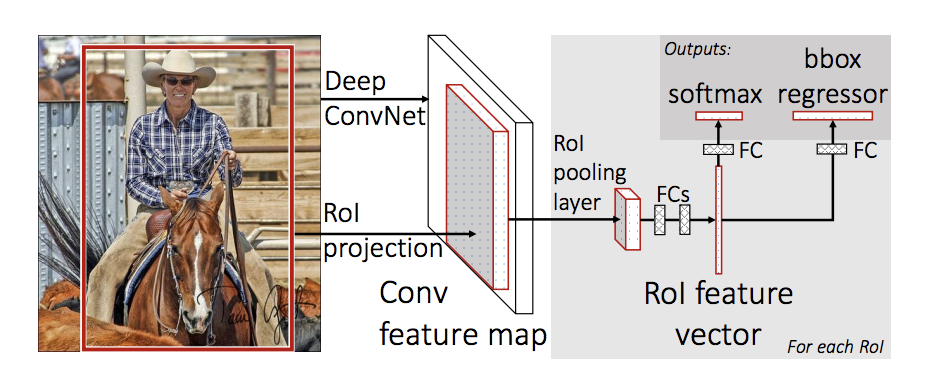
**(2.2)**

: Tahmin edilen alan

: Gerçek alan

* + 1. Fast R-CNN

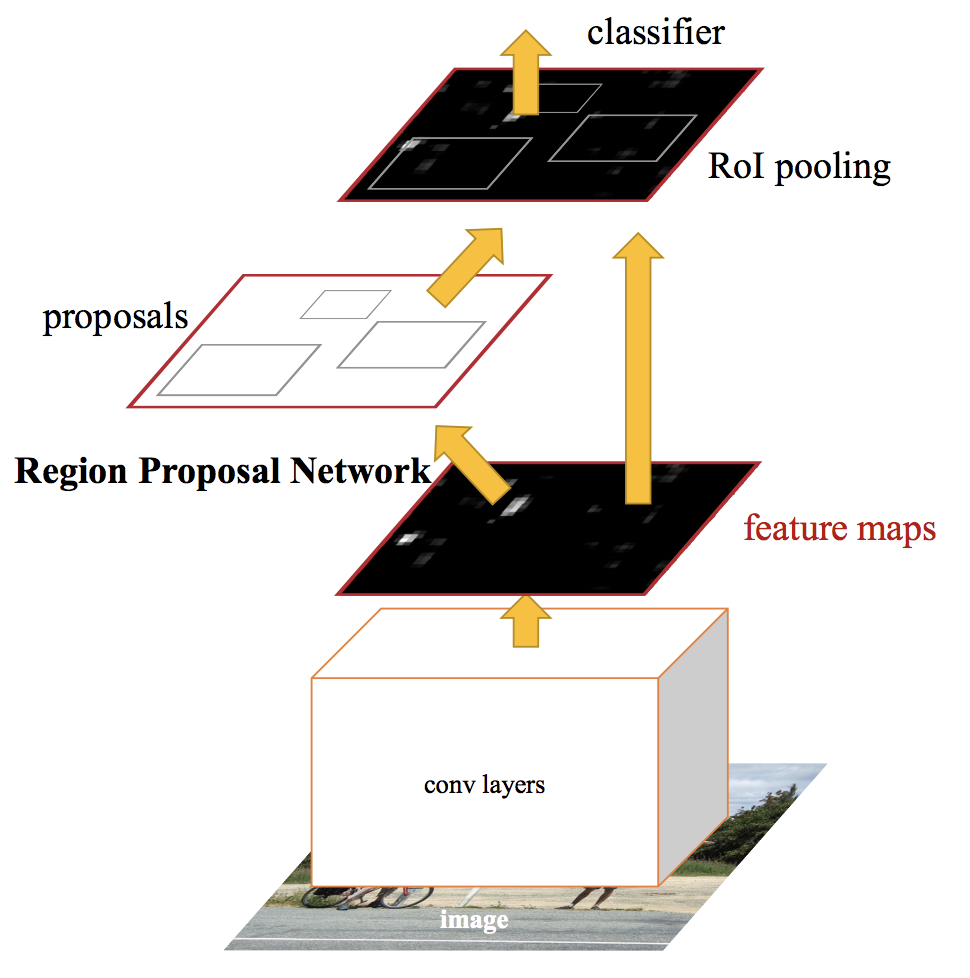
R-CNN mimarisinde 2000 bölgeye tek tek CNN uygulanmasından dolayı zaman maliyeti çok yüksek olan bir modeldir. Bu yüzden Fast R-CNN mimarisinden ayrı ayrı 2000 adet CNN’den vazgeçilip tek bir adet model kullanılmaktadır. Fast R-CNN’de tüm görüntü CNN ile işlenip bir adet öznitelik haritası çıkarılır. Her bölge için öznitelik haritaları toplanır. Bu öznitelik haritalarına max pooling uygulanır ve belirli bir boyuta indirgenir. Bu pooling katmanının ismi ise Region of Interest (RoI) layer olarak adlandırılmıştır. Bu öznitelik haritaları tek boyutta bir vektör haline indirgenir. Bu vektörler sinir ağına girer ve softmax ile bölgedeki sınıf, bounding box regressor ile belirlenen objenin sınırlayıcı kutusu belirlenir [33],[34],[35], [36].



Şekil 2.11. Fast R-CNN mimarisi [35]

* + 1. Faster R-CNN

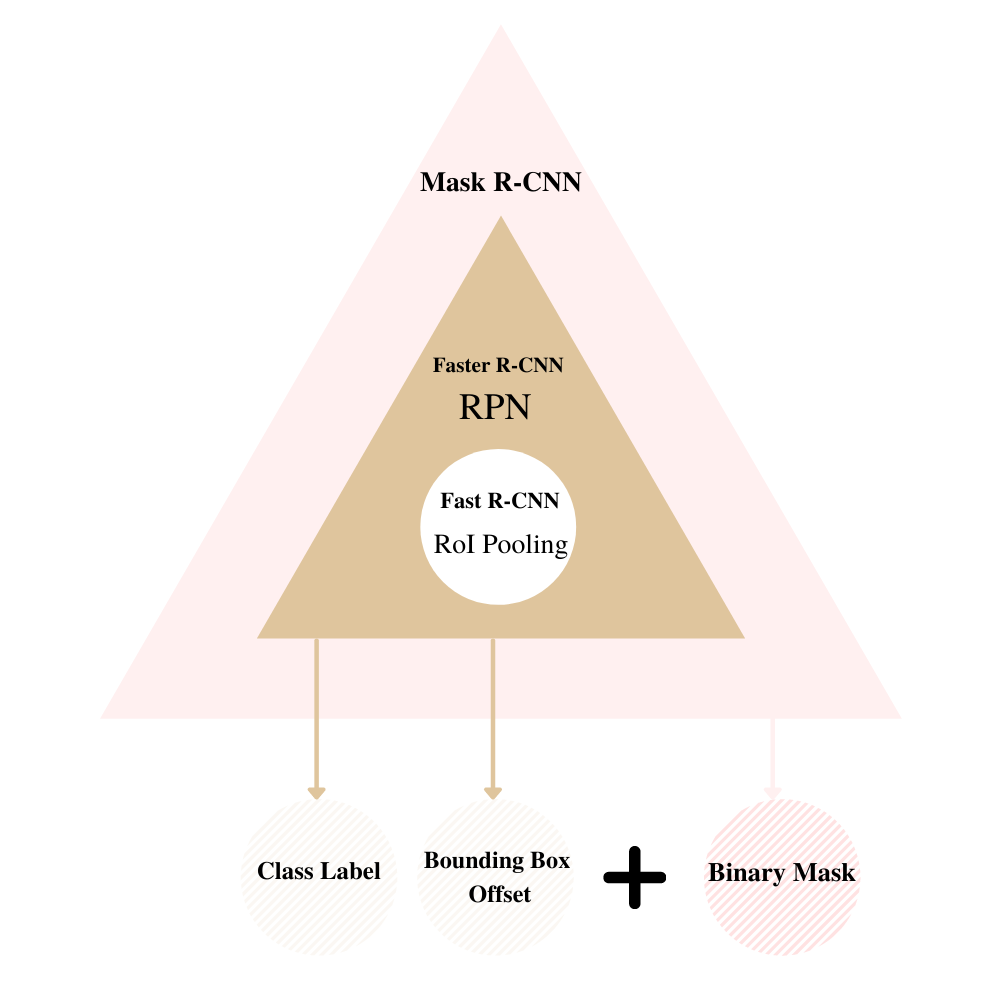
Faster R-CNN, maliyetleri yüksek olsan seçici arama yerine daha hızlı ve maliyetsiz geliştirilen Region Proposal Network (RPN) yöntemini kullanır. RPN bölge önermeye yarayan bir evrişimli sinir ağıdır (CNN). Giriş olarak herhangi bir boyutta girdiyi alır ve cisim skoruna göre bir dizi nesnelere ait olabilecek dikdörtgenler sunar. Bu öneriler CNN tarafından oluşturulan öznitelik haritası üzerinde küçük bir ağı kaydırarak yapar. Bu önerilen alındıktan sonra bu hesaplamaları Bölüm 2.2.3.’te bahsedilen Fast R-CNN mimarisine girdi olarak verir ve sonuçlarını o mimari yapısına göre alır [33],[34],[35],[36].



Şekil 2.12. Faster R-CNN mimarisi [35]

* + 1. Mask R-CNN

Faster R-CNN çıkışında 2 tür vardır. Bunlardan ilki sınıf etiketi ikincisi ise sınırlayıcı kutudur. Mask R-CNN’de buna ek olarak nesne maskesi çıkaran ek bir dal mevcuttur. Bu maske sınıf etiketi ve sınırlayıcı kutudan çok farklıdır. Ek olarak Mask R-CNN’in farkı pikselden piksele hizalama yönteminin kullanılmasıdır. Mask R-CNN Faster R-CNN’in iki aşamayı benimser ve ikinci aşamaya paralel olarak her bir RoI değeri için ikili bir maske çıkarır [37],[38].

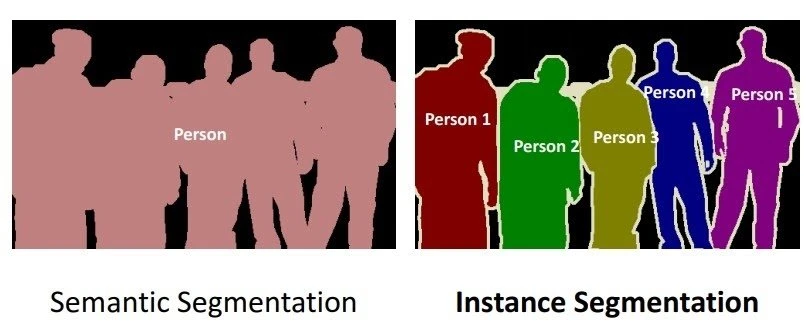


Şekil 2.13. Mask R-CNN diğer ağlar ile gösterimi [38]

Mask R-CNN kapsamında 2 ayrı segmentasyon vardır [37].

* Semantik Segmentasyon (Semantic Segmentation): Nesne örneklerini ayırt etmeden her bir pikseli sabit bir kategori ile sınıflandırır [37].
* Örnek Segmentasyon (Instance Segmentation): Tüm nesneleri doğu tespit ederken o nesnelerin örneklerini de hassas bir şekilde ayırt etmeye çalışır. Bu nedenle nesne algılama, nesne sınıflandırma ve nesne lokalizasyonunun birleşimidir [37].

Bu iki segmentasyonun örnek görseli Şekil 2.14’te gösterilmiştir.



Şekil 2.14. Semantik ve Örnek Segmentasyon farkı [37]

Mask R-CNN avantajları:

* Basitlik: Eğitim kolaydır.
* Performans: Her görevde diğer tüm tek giriş modellerden daha iyi performans sergiler
* Verimlilik: Faster R-CNN’e yalnızca küçük bir ek adım ekler
* Esneklik: Diğer görevlere genellemek kolaydır. Örneğin insan poz algılanması kolaylıkla yapılabilir.

# GELİŞTİRİLEN YAZILIM

* 1. Kullanılan Teknolojiler

Uygulama tasarlanırken kullanılan teknolojilerde kısaca bahsedilmiştir.

* + 1. Python dili

Python, nesne yönelimli ve yüksek seviyeli bir programlama dilidir. Neredeyse bütün işletim sistemlerinde çalışabilir (Linux, Unix, Windows, Macintosh OS). Python ile veri analizi, ağ programlama, kullanıcı arabirimi programlama, web programlama, veri tabanı yazılımı ve bilgisayarlı görü gibi birçok alanda yazılım geliştirilebilir [40].

* + - 1. OpenCV

OpenCV, açık kaynak kodlu olarak geliştirilmiş C++ programlama dili ile yazılmış bir bilgisayarlı görü kütüphanesidir. Bu kütüphane birçok programlama dili ile aktif olarak kullanılabilmektedir [41].

OpenCV ’nin sıkça tercih edildiği alanlar [41]:

* Mobil robotik
* Hareket takibi
* Nesne tespiti
* Yüz tanıma sistemleri
* Arttırılmış gerçeklik

OpenCV şu algoritmaları bünyesinde barındırır [41]:

* Karar ağaçları
* K-en yakın komşu
* Yapay sinir ağları
* Rastgele orman
* Destek vektör makinaları (SVM)
* Derin sinir ağları (DNN)
  + - 1. Torch

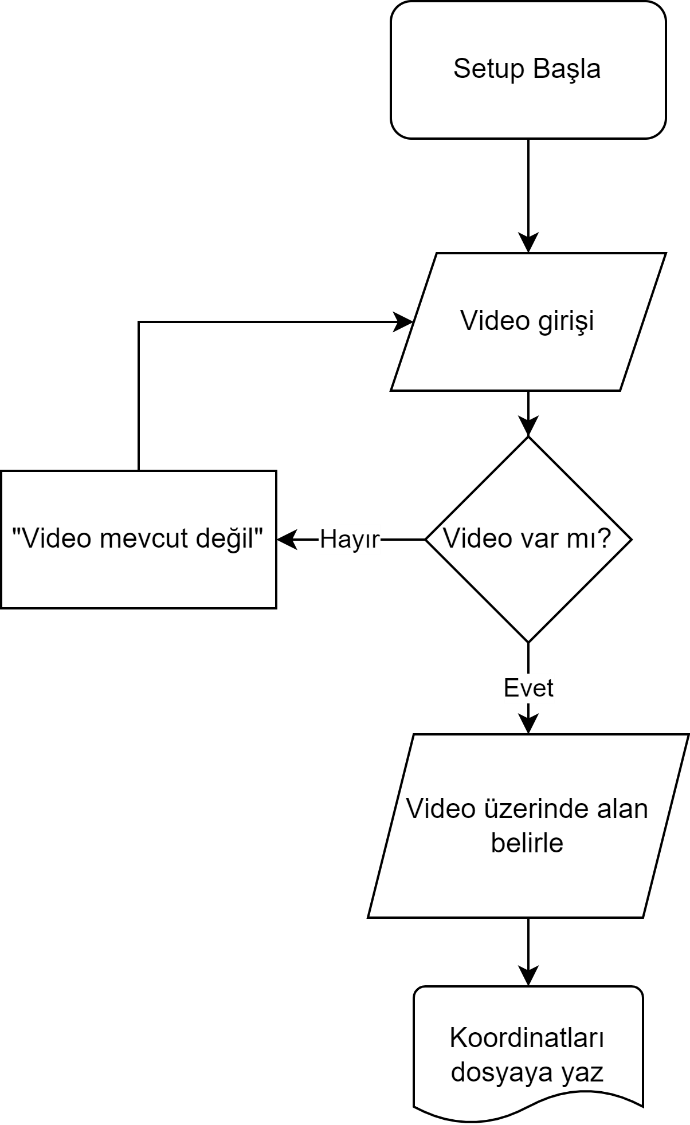
Torch, açık kaynak kodlu bir makine öğrenmesi kütüphanesidir. Lua programlama dili ile programlanmıştır ancak birçok dile API’ları sayesinde destek verebilmektedir. PyTorch bunların en sık kullanılan API’larından biridir. Python desteklemesi sayesinde OpenCV ile bilgisayarlı görü alanında sıkça tercih edilmektedir [42].

PyTorch’un bir parçası olan torchvision kısmı sayesinde bilgisayarlı görü alanında hazır kod parçaları ve eğitilmiş modeller sayesinde hızlıca ürün ortaya konabilmektedir [43].

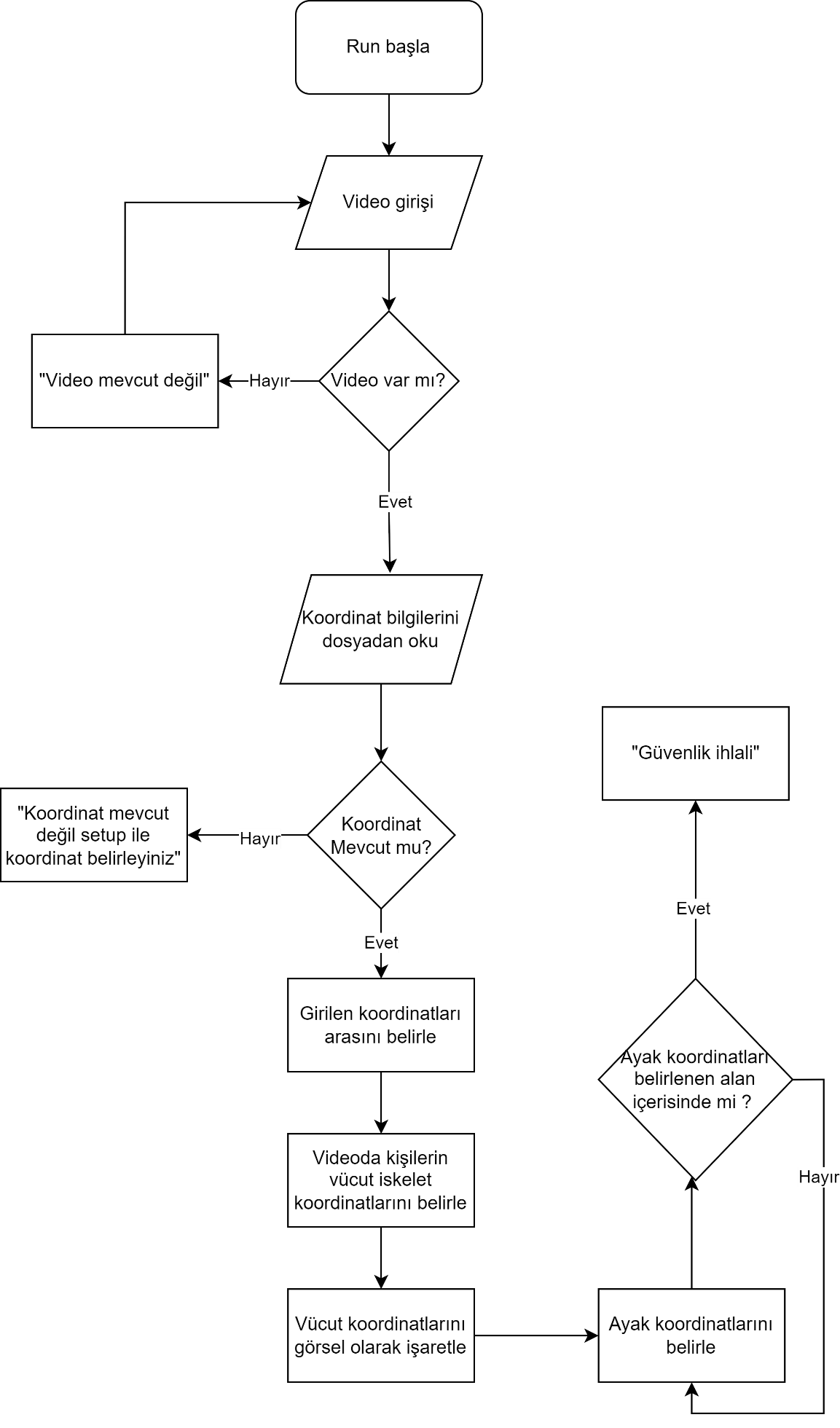
* 1. Geliştirilen Algoritma

Tasarlanan uygulamada temel olarak iki kısma ayrılmaktadır. Bunlar setup ve run olarak isimlendirilmiştir. Setup kısmında kullanıcı video girişi (video dosya yolu) yapıp bu videonun koordinatlarını fare ile işaretlenmesi sağlanmaktadır. Run kısmında ise koordinatlar okunup videodaki insanların vücut omurga yapıları ekrana yansıtılarak ayakları belirlenen alana giriş sağladığında uyarı verilmesi sağlanmıştır.

Şekil 3.1. ve Şekil 3.2.’de akış diyagramları gösterilmiştir.



Şekil 3.1. Setup Akış Diyagramı



Şekil 3.2. Run Akış Diyagramı

* 1. Uygulamanın Kısaca Tanıtımı

Uygulama python programlama dili ile yazılmış olup temel olarak PyTorch ve OpenCV üzerine oturtulmuştur. İnsanların vücut yapısı algılanması için PyTorch keypoints detection kullanılmış olup önceden eğitilmiş model tercih edilmiştir. Önceden eğitilmiş model tercih edilmesinin sebebi başarı oranının yüksek olup kendini kanıtlamış bir model olması ve bu sayede uzun süren eğitimlerden kurtarması. Buna ek olarak hazır kullanarak zamandan tasarruf sağlayıp test sürelerinin artmasının sağlanması.

* + 1. Vücut duruş yapısı tespiti

Vücut duruş pozisyonu diğer bir deyişle poz algılaması PyTorch’ un hazır eğitilmiş modeli ile yapılmıştır. Vücut poz algılanmasında bu uygulamada en önemli olan noktalar 15-16 numaralı bölgelerdir. Bu bölgeler ayakları göstermektedir. Ayakların önemli olmasının sebebi ise 2 boyutlu uzayda çalışmalar yapıldığından güvenlik alana girişleri ayakların o alana girmesi ile doğruluğu sağlamasındandır.

* + 1. Nesne ve insan tespiti

Nesne ve insan tespitinde PyTorch’ un hazır eğitilmiş modeli tercih edilmiş olup Mask R-CNN algoritması kullanılmıştır. Nesne ve insan tespitinin kullanılmasının nedeni eğer uygulama ilerletilmek istenirse yeni modeller eğitilerek insanların üzerinde güvenlik ekipmanlarının varlığını tespit eden yapıyı kolayca uygulamaya eklenmesini sağlamaktır.

* + 1. Güvenlik alanı seçimi

Güvenlik alanı seçimi OpenCV ROI fonksiyonu ile sağlanmıştır. ROI kullanılarak güvenlik alanının koordinatları alınarak dışarı Pickle dosyası olarak aktarılır. Her bir seferde birden çok güvenlik alanı seçilebilir.

* + 1. Uygulama içinden görüntüler

metin, yer, iç mekan, kalabalık içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 3.3. Uygulama içi alan dışı görüntü

metin, iç mekan, yer, kalabalık içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 3.4. Uygulama içi sadece Keypoints alan içi görüntü

yer, iç mekan, yeşil, renkli içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 3.5. Uygulama içi alan içi görüntü

# MODELİN TEST EDİLMESİ VE DEĞERLENDİRİLMESİ

* 1. Kullanılan Model

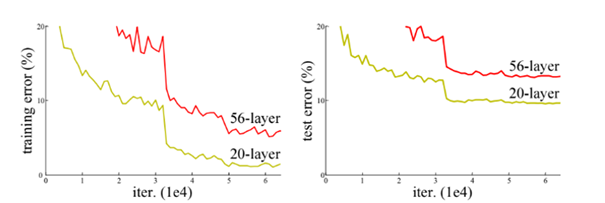
Bölüm 3.3.1 ve Bölüm 3.3.2’de bahsedilen ve uygulamada aktif olarak kullanılan iki yöntemde aynı mimaride çalışmaktadır. Burada birçok nesne algılamada algoritmasında aktif olarak kullanılan Resnet-50 mimarisi tercih edilmiştir. Ve bu model COCO 2017 veri seti ile eğitilmiştir. COCO 2017 veri seti ile hazır eğitilmiş model tercih edilmiştir.

* + 1. ResNet-50

ResNet, 2015 yılında Kaiming He, Xiangyu Zhang, Shaoqing Ren ve Jian Sun tarafından önemli ölçüde daha derin olan ağların eğitimini kolaylaştırmak için “Deep Residual Learning for Image Recognition” makalesinde tanıtılan belirli bir sinir ağı türüdür [45].

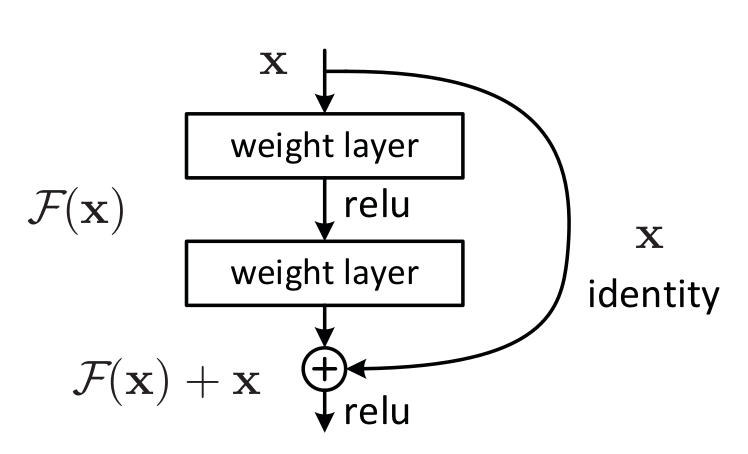
ResNet başarıları şu şekildedir:

* ImageNet veri setinde VGG ağlarından derin ancak daha anlaşılır ve daha az karmaşıklığa sahip 152 katmandan oluşan derin ağda %3.57 hata oranı ile ILSCRC 2015 sınıflandırma görevinde 1. Oldu
* Faster R-CNN’de VGG-16 katmanlarının ResNet-101 ile değiştirilmesi sonucunda COCO veri setinin nesne algılamasında %28 iyileştirilme gözlemlendi.
* ILSVRC ve COCO 2015 yarışmasında ImageNet algılama, ImageNet yerelleştirme, COCO algılama ve COCO segmentasyonun da 1. oldu.



Şekil 4.1. Görüntü Tanıma İçin Deep Residual Learning [45]

Şekil 4.1’de görünen iki grafikte sol grafik eğitim hatası sağ grafikte test hatası gözlemlenmektedir. İki grafikte de 56 katmanlı bir ağın 20 katmanlı ağdan daha fazla hata yaptığı söylenebilir. Buradan gözlemlendiği gibi ağ sayısındaki artış hata oranını düşürmemektedir [44, 45]. Bunun nedeni çok derin ağların eğitilmesi yakınsamayı engelleyen kaybolan gradyanlar nedeniyle zordur. Teoride ağ katmanı arttıkça doğruluk artması beklenirken bunun tam tersi olduğu gözlemlenir. Ağ doğruluğa doyar ve bundan sonraki her eğitimde doğruluk düşer. Bu aşırı öğrenme (overfitting) değildir. Bu olaya derin ağlarda bozulma/optimizasyon problemi denir.

**

Şekil 4.2. Artık Blok Yapısı

Şekil 4.2’de gösterilen x katman girdisinin toplama işlemine taşıyan yola artık bağlantı veya kısayol bağlantısı denir. Kısayol bağlantıları bir veya birden çok katman atlayabilir.

Kısayol bağlantısı kullanmadan önce “x” girdimiz katmanın ağırlıklarıyla çarpılır ve bir ön yargı terimi eklenir. Ardından f() aktivasyon fonksiyonundan geçer ve çıktıyı H(x) olarak alırız.

**(4.1)**

Kısayol bağlantısı ile çıktı şu şekilde olur:

**(4.2)**

Blok artık, aynı sayıda çıktı kanalın sahip 2 adet 3x3 evrişimli katmana sahiptir. Her evrişimli katmanı bir normalleştirme katmanı ve bir ReLu izler. Ardından iki evrişim işlemi atlanıp girdi doğrudan ReLu işleminde önce eklenir.

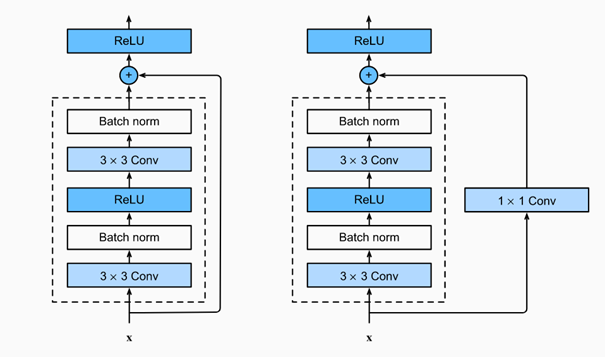
Kısaca girdi boyutu aradaki havuzlama katmanları ve evrişim katmanı ile farklı boyutlarda olabilir. Bu sorun iki farklı şekilde ele alınabilir. Bu yaklaşımlar şu şekildedir:

* Atlama bağlantısı, boyutu arttırmak amacıyla 0 ile doldurulabilir.
* Kanal sayısı değiştirilmek istenirse girdiyi toplama işlemi için istenen şekle dönüştürerek ek bir 1x1 evrişim katmanı eklenir.

Bu yaklaşımda ek parametreye ihtiyaç duyulur. Yeni çıktı şu şekilde olur:

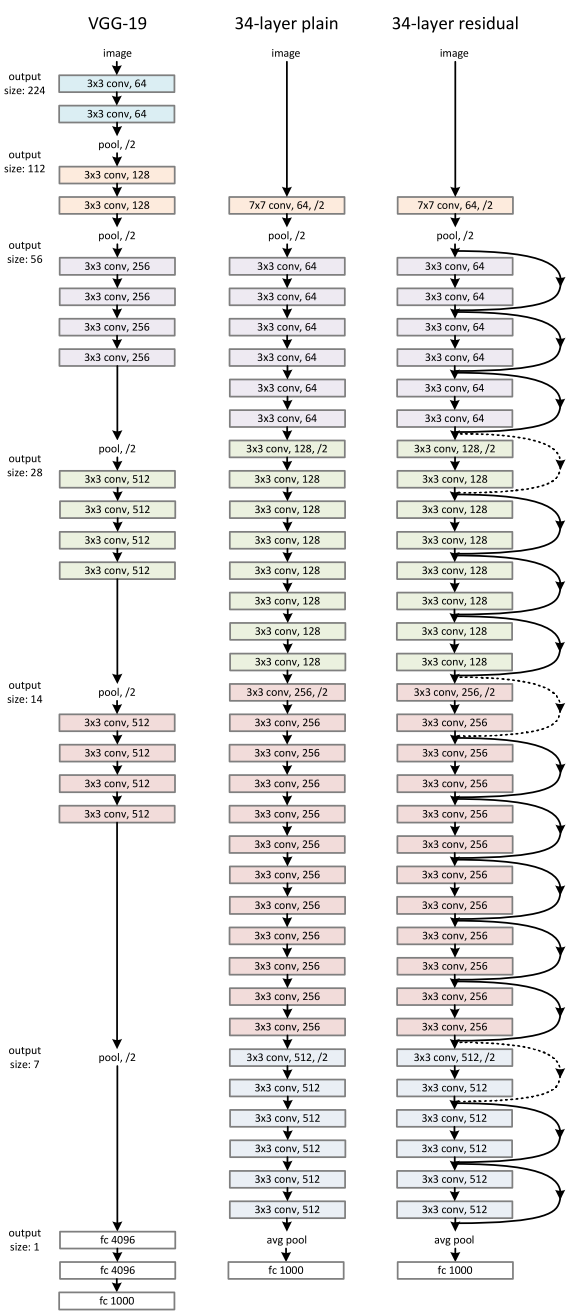
**(4.3)**

Şekil 4.3’te bu içeren ve içermeyen bloklar ayrı ayrı gösterilmiştir.



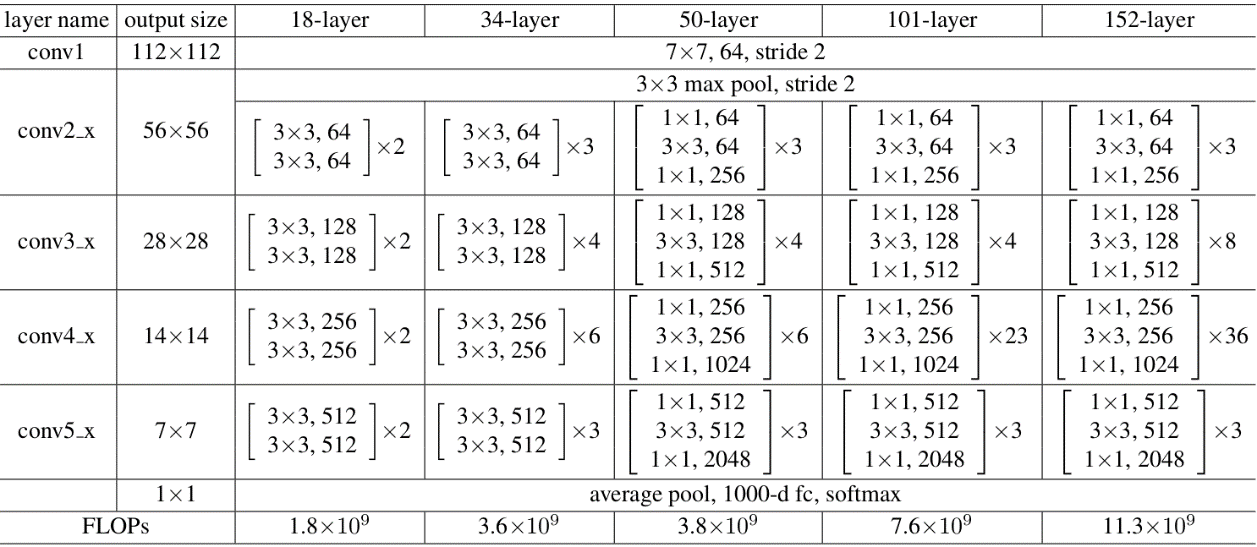
Şekil 4.3. 1x1 Evrişim katmanı içeren ve içermeyen ResNet Bloğu

Şekil 4.4’te gösterilen mimaride VGG-19 mimarisi esas alınarak ondan daha az filtreye ve karmaşıklığa sahip olarak 34 katmanlı düz ağ mimarisi gösterilmiştir.



Şekil 4.4. ResNet Mimarisi

Şekil 4.5’te uygulamada da kullanılan ResNet-50 mimari yapısı gösterilmiştir.



Şekil 4.5. ResNet-50 mimarisi [44]

Şekil 4.6’da birden çok mimari yapının ImageNet doğruluk oranları verilip sayısal karşılaştırma sağlanmıştır.

tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 4.6. ResNet karşılaştırma tablosu [44]

ResNet mimarisinde sayılar katman sayısını temsil etmektedir. ResNet-50 ile ResNet-101 arasındaki temek fark katman sayısında yatmaktadır.

* 1. Uygulama Performans Değerlendirmesi

Tablo 4.1. de gösterildiği gibi maskeleme ve vücut duruş pozisyonun aktif olup olmamasına göre GPU ve CPU performansı karşılaştırılmıştır.

Tablo 4.1 Sistem değerlendirme FPS tablosu

|  |  |  |  |
| --- | --- | --- | --- |
| Maske Tespiti | Vücut duruş Pozisyonu Tespiti | GPU (Nvidia RTX 2070S) | CPU (Intel i7 10875H) |
| Açık | Açık | 2.6 FPS | 0.18 FPS |
| Kapalı | Açık | 8.0 FPS | 0.49 FPS |
| Açık | Kapalı | 3.1 FPS | 0.28 FPS |

Tablo 4.1.’de de gözlemlendiği gibi maskeleme yapmak performansı gözle görülecek şekilde düşürmüştür. Eğer cisim algılanması gerekmiyor ise (makinelerin işaretlenmesi, kişilerin kasket takıp takmadığının kontrolü gibi) maskele kullanmak gereksiz performans düşüşüne neden olmaktadır. Gereksiz denmesinin sebebi vücut duruş pozisyonunda insanların zaten işaretleniyor olması.

CPU ve GPU performansını karşılaştıracak olursak CPU neredeyse görüntü işleme imkânı vermiyor. Bunun sebebi derin ağlarda çok fazla matematiksel işlem gerçekleştiriliyor olması. Nvidia firmasının CUDA çekirdekleri ile bu basit matematiksel işlemler hızlıca gerçekleştiriliyor. Eğer kütüphanede (projede PyTorch kullanıldı) GPU desteği yok ise zorunlu olarak CPU kullanılmaktadır.

# SONUÇ VE DEĞERLENDİRME

Bu çalışmada, derin öğrenme tabanlı nesne tespiti ve vücut duruş pozisyonu algılanarak belirtilen alana giriş tespiti yapılmıştır. Maskeleme ve vücut duruş pozisyonu algılamada PyTorch kütüphanesi kullanılarak ResNet-50 mimarisinde COCO veri seti ile eğitilmiş model kullanılmıştır.

Model canlı bir güvenlik kamerası üzerinde test edilmediğinde canlı görüntü performans testleri gerçekleştirilemedi. Birkaç hazır video ve kaydı alınan bir görüntü üzerinde test edilmiştir.

Çalışmayı daha ileriye götürebilecek adımlar şu şekilde olabilir:

* İş güvenliği için giyim tespiti (Yelek, kasket, eldiven vb.)
* Endüstriyel otomasyon (PLC vb.) ile haberleşme sağlanarak uyarı sayesinde makinelerin kapatılması. Örnek görüntülerde olan lazer cihazları çalışırken alana giriş ile lazer kesimin duraklatılması.
* Uyarılar veri tabanında depolanarak çok fazla ihlal gerçekleşen alana ek önlemler alınabilir.
* Model buluta aktarılarak donanım ihtiyacı uzak sunucuda karşılanır. Her bir bölge için donanım ihtiyacı uzak sunucuya API sayesinde iletilerek karşılanabilir.

KAYNAKLAR

|  |  |  |
| --- | --- | --- |
| [1] |  | Görüntü İşleme Nedir?, Erişim Adresi: <https://azure.microsoft.com/tr-tr/overview/what-is-computer-vision/>, Erişim Tarihi: 08.03.2022 |
| [2] |  | Görüntü işleme nedir ve nerelerde kullanılır, Erişim Adresi: <https://www.ceyrekmuhendis.com/goruntu-isleme-nedir-ve-nerelerde-kullanilir/>, Erişim Tarihi: 08.02.2022 |
| [3] |  | Workplace safety tips, Erişim Adresi: <https://www.indeed.com/career-advice/career-development/workplace-safety-tips>, Erişim Tarihi: 09.03.2022 |
| [4] |  | Makine öğrenmesi, Erişim Adresi: <https://www.ibm.com/tr-tr/cloud/learn/machine-learning>, Erişim Tarihi: 09.03.2022 |
| [5] |  | Derin öğrenme nedir?, Erişim Adresi: <https://nyilmazsimsek.medium.com/derin-%C3%B6%C4%9Frenme-deep-learning-nedir-ve-nas%C4%B1l-%C3%A7al%C4%B1%C5%9F%C4%B1r-2d7f5850782>, Erişim Tarihi: 09.03.2022 |
| [6] |  | DEY, Sandipan. Python Image Proccesing Cookbook. Packt Publishing Limited, 2020. |
| [7] |  | AVENDI, Michael. PyTorch Computer Vision Cookbook: Over 70 Recipes to Master the Art of Computer Vision with Deep Learning and PyTorch 1. x. Packt Publishing Limited, 2020. |
| [8] |  | AYYADEVARA, V. Kishore; REDDY, Yeshwanth. Modern Computer Vision with PyTorch: Explore deep learning concepts and implement over 50 real-world image applications. Packt Publishing Ltd, 2020. |
| [9] |  | Intenseye, [www.intenseye.com](http://www.intenseye.com) |
| [10] |  | CHOOCH, chooch.ai |
| [11] |  | AlwaysAI, alwaysai.co |
| [12] |  | PlainSight, plainsight.ai |
| [13] |  | OpenCV, opencv.org |
| [14] |  | COCO Dataset, cocodataset.org |
| [15] |  | ImageNet, image-net.org |
| [16] |  | Kaggle, kaggle.com |
| [17] |  | CNN nedir, nasıl çalışır?, Erişim Adresi: <https://medium.com/@tuncerergin/convolutional-neural-network-convnet-yada-cnn-nedir-nasil-calisir-97a0f5d34cad>, Erişim Tarihi: 13.03.2022 |
| [18] |  | Derin Öğrenme Nedir ve Uygulama Alanları, Erişim Adresi: <https://www.elektrikde.com/derin-ogrenme-nedir-ve-uygulama-alanlari/>, Erişim Tarihi: 13.03.2022 |
| [19] |  | Introduction to Convolutional Neural Networks, Erişim Adres: <https://rubikscode.net/2018/02/26/introduction-to-convolutional-neural-networks/>, Erişim Tarihi: 13.03.2022 |
| [20] |  | Convolutional Neural Networks, Erişim Adresi: <https://en.wikipedia.org/wiki/Convolutional_neural_network>, Erişim Adres: 13.03.2022 |
| [21] |  | Yapay Sinir Ağları ve Uygulamaları, Erişim Adresi: <https://www.gtech.com.tr/yapay-sinir-aglari-ve-uygulamalari-1/>, Erişim Tarihi: 15.03.2022 |
| [22] |  | Yapay Sinir Ağları, Erişim Adresi: <https://www.veribilimiokulu.com/yapay-sinir-aglari/>, Erişim Tarihi: 15.03.2022 |
| [23] |  | Yapay Sinir Ağları, Erişim Adresi: <https://tr.wikipedia.org/wiki/Yapay_sinir_a%C4%9Flar%C4%B1>, Erişim Tarihi: 15.03.2022 |
| [24] |  | Bilgisayarlı Görü, Erişim Adresi: https://www.datasciencearth.com/bilgisayarli-goru-computer-vision/, Erişim Tarihi: 16.03.2022 |
| [25] |  | TOSHEV, Alexander; SZEGEDY, Christian. Deeppose: Human pose estimation via deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2014. p. 1653-1660. |
| [26] |  | CHEN, Ching-Hang; RAMANAN, Deva. 3d human pose estimation= 2d pose estimation+ matching. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017. p. 7035-7043. |
| [27] |  | DORRER, M. G.; TOLMACHEVA, A. E. Comparison of the YOLOv3 and Mask R-CNN architectures’ efficiency in the smart refrigerator’s computer vision. In: Journal of Physics: Conference Series. IOP Publishing, 2020. p. 042022. |
| [28] |  | SHUSTANOV, Alexander; YAKIMOV, Pavel. CNN design for real-time traffic sign recognition. Procedia engineering, 2017, 201: 718-725. |
| [29] |  | YILDIRIM, Muhammed; ÇINAR, Ahmet. Classification of 40 Different Human Movements with CNN Architectures and Comparison of Their Performance. Turkish Journal of Science and Technology, 2021, 16.1: 103-112. |
| [30] |  | MURADLI, Firgat, et al. Derin öğrenme kullanılarak görüntülerden insan duruş tespiti. 2021. Master's Thesis. Sakarya Üniversitesi. |
| [31] |  | STEVENS, Eli; ANTIGA, Luca; VIEHMANN, Thomas. Deep learning with PyTorch. Manning Publications, 2020. |
| [32] |  | Derin Öğrenme, Erişim Adresi: <https://blog.dopinger.com/tr/derin-ogrenme-nedir/#Derin_Ogrenme_Kullanim_Sekilleri>, Erişim Tarihi: 16.03.2022 |
| [33] |  | R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms, Erişim Adresi, <https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e>, Erişim Tarihi: 16.03.2022 |
| [34] |  | Understanding Fast R-CNN and Faster R-CNN for Object Detection, Erişim Adresi: <https://towardsdatascience.com/understanding-fast-r-cnn-and-faster-r-cnn-for-object-detection-adbb55653d97>, Erişim Tarihi: 16.03.2022 |
| [35] |  | R-CNN vs Fast R-CNN vs Faster R-CNN | ML, Erişim Adresi: <https://www.geeksforgeeks.org/r-cnn-vs-fast-r-cnn-vs-faster-r-cnn-ml/>, Erişim Tarihi: 16.03.2022 |
| [36] |  | Getting Started with R-CNN, Fast R-CNN, and Faster R-CNN, Erişim Adresi: <https://uk.mathworks.com/help/vision/ug/getting-started-with-r-cnn-fast-r-cnn-and-faster-r-cnn.html>, Erişim Tarihi: 16.03.2022 |
| [37] |  | Mask R-CNN: A Beginner’s Guide, Erişim Adresi: <https://viso.ai/deep-learning/mask-r-cnn/>, Erişim Tarihi: 16.03.2022 |
| [38] |  | Nesne Tespit Algoritmalarının Gelişimi (Mask R-CNN) — 2, Erişim Adresi: <https://dilaraozdemir.medium.com/nesne-tespit-algoritmalar%C4%B1n%C4%B1n-geli%C5%9Fimi-mask-r-cnn-2-b622f6f4c2a8>, Erişim Tarihi: 16.03.2022 |
| [39] |  | HE, Kaiming, et al. Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. 2017. p. 2961-2969. |
| [40] |  | Python (Programlama Dili), Erişim Adresi: https://tr.wikipedia.org/wiki/Python\_(programlama\_dili),  Erişim Tarihi: 17.03.2022 |
| [41] |  | OpenCV, Erişim Adresi: <https://en.wikipedia.org/wiki/OpenCV>, Erişim Tarihi: 17.03.2022 |
| [42] |  | Torch, Erişim Adresi: <https://en.wikipedia.org/wiki/Torch_(machine_learning)>, Erişim Tarihi: 17.03.2022 |
| [43] |  | PyTorch, Erişim Adresi: <https://en.wikipedia.org/wiki/PyTorch>, Erişim Tarihi: 17.03.2022 |
|  |  |  |
| [44] |  | Understanding ResNet50 architecture, Erişim Adresi: <https://iq.opengenus.org/resnet50-architecture/>, Erişim Tarihi: 02.05.2022 |
| [45] |  | HE, Kaiming, et al. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. p. 770-778. |

EKLER

EK A: Yürütme planı

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **İş Paketi No** | **İş Paketi Adı** | **Haftalar** | | | | | | | | | | | | | |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** |
| **1** | Proje yol haritasının belirlenmesi | **x** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **2** | Bilgisayarlı görü ve teknolojilerinin teorik araştırması |  | **x** | **x** |  |  |  |  |  |  |  |  |  |  |  |
| **3** | Bilgisayarlı görü için OpenCV ve PyTorch kullanımının araştırılması |  | **x** | **x** |  |  |  |  |  |  |  |  |  |  |  |
| **4** | Hazır video görüntülerinin elde edilmesi |  |  |  | **x** |  |  |  |  |  |  |  |  |  |  |
| **5** | Araştırma sonuçlarının analiz edilmesi |  |  |  | **x** |  |  |  |  |  |  |  |  |  |  |
| **6** | Bilgisayarlı görü algoritmasının seçilmesi |  |  |  | **x** |  |  |  |  |  |  |  |  |  |  |
| **7** | Bilgisayarlı görü algoritmasının test edilmesi |  |  |  |  | **x** | **x** | **x** |  |  |  |  |  |  |  |
| **8** | Güvenlik alanının video içerisinden seçilebilmesi ve kaydedilmesi |  |  |  |  |  |  | **x** | **x** |  |  |  |  |  |  |
| **9** | Güvenlik alanına girişin saptanması |  |  |  |  |  |  |  |  | **x** | **x** |  |  |  |  |
| **10** | Hazır birkaç video içerisinde test ve optimizasyon |  |  |  |  |  |  |  |  |  | **x** | **x** |  |  |  |
| **11** | Çalışan uygulamanın scripte dökülmesi (Setup ve Run) |  |  |  |  |  |  |  |  |  |  |  | **x** |  |  |
| **12** | Hata tespitleri |  |  |  |  |  |  |  |  |  |  |  | **x** |  |  |
| **13** | Son aşama testleri optimizasyon |  |  |  |  |  |  |  |  |  |  |  | **x** | **x** |  |
| **14** | Proje çıktısının alınması |  |  |  |  |  |  |  |  |  |  |  |  | **x** |  |
| **15** | Proje değerlendirmelerinin yapılması |  |  |  |  |  |  |  |  |  |  |  |  | **x** | **x** |

ÖZGEÇMİŞ

Fatih Gülşen, 07.01.1999 da Bursa’da doğdu. İlk ve orta eğitimini Osmangazi’de tamamladı. Lise eğitimini ise Nilüfer’de tamamladı. 2022 yılı itibarıyla Sakarya Üniversitesi, Bilgisayar ve Bilişim Bilimleri Fakültesinde eğitimine devam etmektedir. 2020 yılında Trade Atlas Bilgi Sistemleri Şirketinde yazılım stajını yaptı ve staj bitiminden itibaren bir sene burada çalışmaya devam etti. Şu anda aktif olarak Mol Kimya Ltd. Şti.’nin ERP sisteminin tasarımı ve veri tabanı sistemi yöneticiliğini gerçekleştirmektedir.

BSM 498 BİTİRME ÇALIŞMASI

Değerlendİrme ve Sözlü Sınav Tutanağı

KONU: BİLGİSAYARLI GÖRÜ İLE İŞ SAHASI GÜVENLİĞİ

ÖĞRENCİLER (Öğrenci No/AD/SOYAD): G171210070 – Fatih GÜLŞEN

|  |  |  |  |
| --- | --- | --- | --- |
| 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** |  |  |  |

Danışman (Jüri adına):

danışman imzası: