T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

UYDU GÖRÜNTÜLERİNDEN ALINAN HALKA AÇIK BİNALARIN DERİN ÖĞRENME YÖNTEMLERİYLE SINIFLANDIRILMASI VE PERFORMANS ÖLÇÜMÜ

Şeyma KARABULUT

Yüksek Lisans Tezi

EKOBİLİŞİM ANABİLİM DALI

EYLÜL 2022

T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Ekobilişim Anabilim Dalı

Yüksek Lisans Tezi

UYDU GÖRÜNTÜLERİNDEN ALINAN HALKA AÇIK BİNALARIN DERİN ÖĞRENME YÖNTEMLERİYLE SINIFLANDIRILMASI VE PERFORMANS ÖLÇÜMÜ

Tez Yazarı
Şeyma KARABULUT

Danışman
Doç. Dr. Derya AVCI

EYLÜL 2022
ELAZIĞ

T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Ekobilişim Anabilim Dalı
Yüksek Lisans Tezi

Başlığı: Uydu Görüntülerinden Alınan Halka Açık Binaların Derin Öğrenme Yöntemleriyle Sınıflandırılması ve Performans Ölçümü

Yazarı: Şeyma KARABULUT

İlk Teslim Tarihi: 14.06.2022

Savunma Tarihi: 30.09.2022

TEZ ONAYI

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

İmza

Danışman: Doç. Dr. Derya AVCI — Onayladım
Fırat Üniversitesi, Teknoloji Fakültesi

Başkan: Doç. Dr. Eser SERT — Onayladım
Malatya Turgut Özal Üniversitesi, Mühendislik ve Doğa Bilimleri

Üye: Dr. Öğr. Üyesi Murat AYDOĞAN — Onayladım
Fırat Üniversitesi, Teknoloji Fakültesi

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

İmza

Prof. Dr. Burhan ERGEN
Enstitü Müdürü

BEYAN

Fırat Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırladığım " Uydu Görüntülerinden Alınan Halka Açık Binaların Derin Öğrenme Yöntemleriyle Sınıflandırılması ve Performans Ölçümü " Başlıklı Yüksek Lisans Tezimin içindeki bütün bilgilerin doğru olduğunu, bilgilerin üretilmesi ve sunulmasında bilimsel etik kurallarına uygun davrandığımı, kullandığım bütün kaynakları atıf yaparak belirttiğimi, maddi ve manevi desteği olan tüm kurum/kuruluş ve kişileri belirttiğimi, burada sunduğum veri ve bilgileri unvan almak amacıyla daha önce hiçbir şekilde kullanmadığımı beyan ederim.

30.09.2022

Şeyma KARABULUT

ÖNSÖZ

Kurumsal Kaynak Planlaması (KKP) Sistemlerinde var olan iş zekâsı çözümleri daha çok istatistik bilgi vermeye yöneliktir. Coğrafik bir konuda görsel olmadan yapılan istatistik bilginin kullanımı zordur ve bu bilgi gelecek tahmini yapmayı kullanıcıya bıraktığı için de işlemler uzun zaman almakta ve karmaşıklaşmaktadır. Aynı zamanda bu tip bir yazılım oldukça maliyetlidir. Bu çalışmada bina tespiti gibi uygulamaların görüntüler üzerinden yapılması için halka açık binaların uydu imgelerinin sınıflandırılması adına derin öğrenme yöntemleri önerilmiş ve performans ölçümü gerçekleştirilmiştir. Coğrafik uygulamaların akıllı teknolojiler ile yapılmasına hizmet etmesi için daha önce üstünde çalışılmış derin öğrenme tabanlı You Only Look Once (YOLO) algoritması kullanılmıştır.

YOLO algoritmasının üzerinde çalıştığı Darknet framework çatısı altında sanal bir Nvidia CUDA marka GPU ile erişime açık, uydu görüntüleri üzerinde öğrenme ve sınıflandırma işlemleri yapılmıştır. Uydu imgeleri ile seçimi belirginleştirmek adına 5 sınıf belirlenerek her sınıf için 60 imge olmak üzere toplamda 300 örnek imge toplanmıştır ve bu veri seti üzerinde çalışılmıştır.

Çalışmalarım sırasında yakın ilgi, anlayış gördüğüm ve beni destekleyip motive eden danışman hocam Doç. Dr. Derya AVCI ’ya ve tez konusu için önerilerde bulunan Prof. Dr. Engin AVCI’ ya en içten teşekkürlerimi sunarım.

Çalışmam süresince manevi desteklerini esirgemeyen aileme ve arkadaşlarıma teşekkür eder, saygılarımı sunarım.

.

Şeyma KARABULUT
ELAZIĞ, 2022

İÇİNDEKİLER

Sayfa

ÖZET

Uydu Görüntülerinden Alınan Halka Açık Binaların Derin Öğrenme Yöntemleriyle Sınıflandırılması ve Performans Ölçümü

Şeyma KARABULUT

Yüksek Lisans Tezi

FIRAT ÜNİVERSİTESİ
Fen Bilimleri Enstitüsü

Ekobilişim Anabilim Dalı
Eylül 2022, Sayfa: xi + 40

Günümüzde yapay zekânın her konuda uygulanabilir bir teknoloji olması nedeniyle birçok görsel ve ses verisi için kullanımı oldukça artmıştır. Gerek özel, gerekse kamu kurumları var olan yapılarının dışına çıkıp işlerin daha kolay ve hızlı yürütülebilmesi için çeşitli sınıflandırma algoritmaları ile çalışan sistemler geliştirir hale gelmiştir. Özellikle belediyelerde şehir ve bölge planlaması için yapılan çalışmalarda kullanılan görsel sınıflandırmalar, istenilen sonuçlara ulaşılması, detaylandırılması ve hızları bakımından yetersiz bulunmaktadır. Bu tip bina tespiti çalışmasının uydu verileri üzerinden sağlanması verilerin gerçekliğine katkı sağlarken, derin öğrenme algoritmasının test edilip en iyi performans ile gerçekleştirilmesi bu çalışmanın birçok kuruma ve/veya sektöre uyarlanabileceğini göstermektedir. İmge üzerinde yapılmış olan çalışmada halka açık (hastane, okul, hava limanı, cami ve saha) alanların tanımlanması sağlanmıştır.

Kullanıma açık uydu görüntülerinden alınan 5 sınıfa ayrılmış toplam 300 imge üzerinde, derin öğrenme tabanlı YOLOv3 ve YOLOv5 kütüphanesi ile bina sınıflandırılası ve tespiti analiz edilmiştir.

Veri seti üzerinde derin öğrenme için test ve eğitim verileri ayrılmıştır. Python dilinde tasarlanmış YOLOv3 ve YOLOv5 kütüphanesi algoritmaları ile imgeler eğitilmiştir. Eğitim adımı öncesinde ve sonrasında doğru sınıflandırılmış veri oranının kontrolü için daha hızlı bölgesel-evrişimsel sinir ağlar uygulanmıştır. Sınıflandırma sonuçları karşılaştırılmıştır.

Bu halka açık binaların tespiti için Darknet-53 açık kaynaklı sinir ağı çerçevesi ile gerçek zamanlı nesne algılama sistemi YOLOv3 ve YOLOv5 derin öğrenme modelleri kullanılmıştır. Belirtilen bu binaların sınıflandırılması gerçekleştirilerek nitelikli ağırlık modeli oluşturulmuştur. Böylelikle ileriki ekobilişim alanındaki uzay görüntülerinde nesne tespitine katkı sağlanmıştır.

Anahtar Kelimeler: Derin Öğrenme, Makine Öğrenmesi, Daha hızlı Bölgesel-Evrişimsel Sinir Ağları, Python, YOLO, Darknet

ABSTRACT

Classification and Performance Measurement of Public Buildings from Satellite Images with Deep Learning Methods

Şeyma KARABULUT

Master's Thesis

FIRAT UNIVERSITY
Graduate School of Natural and Applied Sciences

Department of Eco-informatics
September 2022, Pages: xi + 40

Today, since artificial intelligence is a technology that can be applied in every subject, its use for many visual and audio data has increased considerably. Both private and public institutions have started to develop systems that work with various classification algorithms in order to go out of their existing structures and to carry out work easier and faster. Visual classifications used in studies for city and regional planning, especially in municipalities, are insufficient in terms of achieving desired results, elaboration and speed. While providing this type of building detection study through satellite data contributes to the authenticity of the data, testing the deep learning algorithm and performing it with the best performance shows that this study can be adapted to many institutions and / or sectors. In the study carried out on the image, it was ensured that the areas open to the public (hospital, school, airport, mosque and field) were defined. Building classification and detection were analyzed with the deep learning-based YOLOv3 and YOLOv5 library on 300 images, divided into 5 classes, taken from satellite images available for use. Test and training data are separated for deep learning on the data set. Images are trained with the algorithms of the YOLOv3 and YOLOv5 library designed in Python. Faster regional-convolutional neural networks were applied to control the correctly classified data rate before and after the training step. Classification results were compared. Darknet-53 open source neural network framework and real-time object detection system YOLOv3 and YOLOv5 deep learnings model were used to detect these public buildings. A qualified weight model was created by classifying these buildings. Thus, it contributed to object detection in space images in the future ecoinformatics field.

Keywords: Deep Learning, Machine Learning, Faster Regional-Convolutional Neural Networks, Python, YOLO, Darknet

ŞEKİLLER LİSTESİ

Sayfa

TABLOLAR LİSTESİ

Sayfa

SİMGELER VE KISALTMALAR

Kısaltmalar

DL : Derin Öğrenme
YSA : Yapay Sinir Ağları
AP : Ortalama Hassasiyet
SSD : Tek Atış Detektörü
YOLO : Sadece Bir Kere Bak
CNN : Evrişimsel Sinir Ağı
GPU : Grafik İşlemci Birimi
CPU : Merkezi İşlem Birimi
RAM : Rastgele Erişimli Bellek
FPN : Özellik Piramit Ağları
RCNN : CNN Özellikli Bölgeler
GAN : Çekişmeli Üretici Ağlar
MLP : Çok Katmanlı Algılayıcılar
SPPNet : Mekânsal Piramit Havuzlama Ağları

1. GİRİŞ

Günümüzde birçok alanda insana özgü nitelikler kazanmış insan düşünme sistemiyle donatılmış, insan gibi düşünebilen, karar alabilen ve harekete geçen robotlar, sistemler veya uygulamalar geliştirilmektedir. Bu insan gibi davranışlar gösteren uygulamaların mühendisliğe kazandırılmasının temelinde makine öğrenmesi teknolojisi vardır. Makine öğrenmesi, insan beyninin çalışma düzeneğine benzetilmiş mühendislik çözümlerinde uygulama yapılmasına yer verilmiş bir öğrenme modeli olarak bilinmektedir. Çalışma prensibi bir insanın ömrü boyunca öğrendikleriyle hayatını kolaylaştıracak seviyeye ulaşması örnek alınarak geliştirilmiştir. Tespit ve ileriye dönük tahmin yapabilme sonuçlarıyla özellikle endüstriyel alanda anormal giden durumlarda, üretim aşamasında süreçlerin hızlandırılmasında, ürünlerin sınıflandırılmasında ve ürün kalitesinin artırılmasında makine öğreniminin tercih edilmesi insanlığa büyük kolaylık sağlamaktadır. Medikal alanda insan sağlığı için teşhis ve tanı koyarken işlemlerin hızlandırılması, benzer şekilde güvenlik alanında görüntü, ses gibi medyaların tanınması ve işlenmesi bu alanda da insan hayatına kolaylık sağlamaktadır.

İnsanın sahip olduğu gözler, kulaklar, burun, deri ve dil devamlı olarak beynin görme, ses, koku, dokunma ve tatla çevirdiği çeşitli veri formlarını toplar. Sonrasında beyin, duyusal organlar yoluyla aldığı çeşitli ham veri işlerini işler ve alınan ham verilerin doğası hakkında görüş bildirmek için kullanılan konuşmaya çevirir. Günümüz dünyasında, makinelere bağlı sensörler veriler toplamak için kullanılmaktadır. Ayrıca internet üzerinden veya sosyal paylaşım sitelerinden birçok veri toplanılmaktadır. Birden fazla kaynaktan toplanan bu zengin veri biçimleri, öngörü kazanılabilmesi ve daha anlamlı bir hale getirilmesi için işlem gerektirir. Bu öngörünün kazanılabilmesi daha net bir ifade ile makinenin bir insan gibi basit tek veya birkaç parçalı işlemleri yapabilmesi olgusu yapay zekânın çıkış noktasıdır [1]. İnsan beynindeki sinir hücrelerinin çalışması şeklinden yararlanılarak çalışma prensibini oluşturmuş makine öğrenmesi beyindeki nöronların yaklaşımı ile makinenin öğrenmesini gerçekleştirmektedir. Bu benzetilmiş modele yapay sinir hücresi modeli denmektedir [2]. Yapay sinir hücreleri, yapay sinir ağları (YSA) şeklinde karışık bir yapı ile derin öğrenme uygulamalarının temelini oluşturmaktadır.

Derin öğrenmenin makine öğreniminin bir kolu olduğu görülmektedir. Makine öğreniminin tarihine bakıldığında ise derin öğrenme mimarileri, yapay zekâya olan ilgi giderek artmasıyla günümüze kadar gelişerek gelen ve en popüler yapay zekâ algoritmalarını olarak ortaya konmuştur. Yapay zekâ sorunlarının çözülmesi noktasında sürekli gelişmekte olan derin öğrenme mimarileri farklı yaklaşımları ile katkı sağlamaktadır.

Günümüzdeki kullanımının yaygınlığı bakımından derin öğrenme; sanayide, robotikte, bankacılıkta, medikal ve tıp alanlarında, güvenlikte, adli vakalarda görüntü işleme, bilgisayar

görmesi, nesne ve ses tespiti gibi birçok alanda kendisini göstermekte ve sorunlara hızlı, etkin ve akıllı çözümler sunmaktadır[2].

Gündelik yaşamda karşılaşılan birçok problemin çözümünde hareketsiz görüntüler veya video görüntüleri akıllı algoritmalar ile işlenmesi gerekmektedir. Örneğin büyük şehirlerde güvenliğin sağlanması nüfusun yoğunluğuna bağlı olarak zorluğu artan bir sorundur. Bu yüzden şehirlerde sokaklar, caddeler ve kamuya açık ortak kullanım alanları güvenlik kameralarıyla donatılmıştır. Ayrıca askeri maksatlı olarak kullanılan insansız kara, hava ve deniz araçları veya benzeri güvenlik amaçlı uygulamalarda da görüntülerin akıllı algoritmalar ile otomatik işlenmesi önemlidir [3].

İnsan yapısını modellemek amacıyla ilk defa McCulloch-Pitts tarafından insan sinir sisteminden esinlenerek beyin fonksiyonlarının işleyişinin mantıksal olarak hesaplayan bir model ortaya konulmuştur. Bu aynı zamanda insan sinir sisteminin bir taklidi olan YSA temelini oluşturmuştur.1980'lerde, sinir ağı araştırmaları yeniden paralel dağıtık işlem olarak ortaya çıktı ve bugünün Derin Öğrenme temeli de o yıllarda ortaya atılmış oldu. O yıllarda yapay sinir ağlarını eğitmek için geri yayılım algoritması başarıyla kullanılmış ve bu kullanım yaygınlaştırılmıştır[4]. Şu anda en popüler derin öğrenme (DL) tabanlı ağ mimarisi, Evrişimli Sinir Ağı (CNN) tabanlı mimaridir [5]. CNN tabanlı nesne algılama yöntemleri arasında iki aşamalı detektörler (örneğin, CNN özellikli bölgeler (RCNN) [6], mekânsal piramit havuzlama ağları (SPPNet) [7], hızlı RCNN [6], daha hızlı RCNN ve özellik piramit ağları (FPN) [8]) ve tek aşamalı detektörler (örneğin YOLO) single shot multibox detector (SSD) [9] ve RetinaNet [8]) mevcuttur. Bu başarılı detektörler, optik görüntülerdeki nesneleri algılamak ve izlemek için geliştirilmiştir. Bunların arasında, tek aşamalı detektörlü YOLO modelleri, gerçek zamanlı optik görüntü algılamada avantajlara sahiptir ancak küçük nesneler için nispeten düşük doğruluğa sahiptir. İki aşamalı detektörler (R- CNN tabanlı modeller) yüksek yerelleştirme ve doğruluk sunar fakat çıkarım hızı nispeten yavaştır [10-15].

2. DERİN ÖĞRENME

2.1. Derin Öğrenme

Derin öğrenmenin tanımı günümüze kadar birçok farklı kaynakta farklı tanımlarla ifade edilmiştir. Çeşitli kaynaklardan alınan tanımlamalara göre;

Derin öğrenme, insan beyninin algılama ve karar verme özelliğini taklit eden bir makine öğrenmesi sınıfıdır. Bilgisayarların, deneyimlerden öğrenmelerini ve dünyayı, kavramların hiyerarşisi açısından anlamalarını sağlayan bir makine öğrenimi olarak tanımlamıştır.

Özetle; derin öğrenme, insan beyninin karmaşık problemler için gözlemleme, analiz etme, öğrenme ve karar verme gibi yeteneklerini taklit eden, denetimli veya denetimsiz olarak özellik çıkarma, dönüştürme ve sınıflandırma gibi işlemleri büyük miktarlardaki verilerden yararlanarak yapabilen bir makine öğrenmesi tekniğidir.

Geliştirilen bu teknik, derin ağların dinamik çalışmasını ve katmanların farklı işlerde yoğunlaşmasını sağlar. Derin ağların bu yapısı, her bir veriyi aynı model ile analiz eden yapay sinir ağlarından ayrılan en önemli özelliğidir. Derin sinir ağının tasarımı, yapay sinir ağı tasarımına benzerdir. Fakat derin ağlar için özel olarak geliştirilen yöntemler, ağın tüm katmanlarını aynı anda çalıştırmak yerine, analiz edilecek verilere özel katmanları çalıştırmayı amaçlamaktadır. Derin öğrenme sistemleri yapay sinir ağlarının özel bir türü olduğundan, yapay sinir ağlarındaki bazı geleneksel problemler de bu sistemlerde mevcuttur[10]. Yapay sinir ağlarında olduğu gibi gizli katmanlar, aktivasyon fonksiyonları, öğrenme algoritmaları, kayıp fonksiyonları, öğrenme katsayısı gibi hiper parametreler kullanılmaktadır. Bu hiper parametreler, entegrasyon, evrişim, otomatik kodlayıcı katmanları ve bırakma gibi hiper parametrelerin eklenmesiyle oluşturulur. Derin ağların eğitiminde ve uygulamasında yeni hiper parametreler kullanılırken, derin ağların ve yapay sinir ağlarının eğitim süreci benzerdir. Antrenman sırasında aşırı ve yetersiz uyum problemlerini önlemek için benzer çalışmalar yapılmaktadır[4].

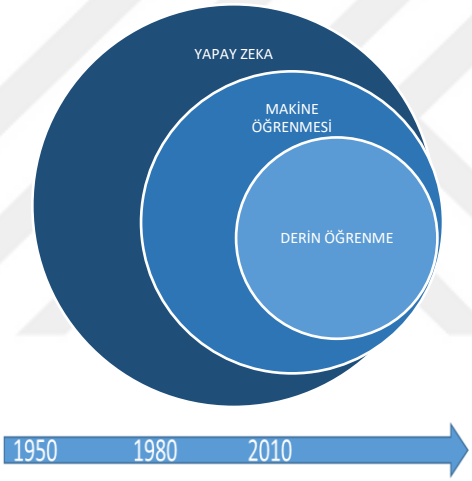
Derin öğrenme günümüzde görüntü, video, doküman ve doğal dil işleme gibi konularda da kullanılmaktadır. Büyük veri ve GPU'ların geliştirilmesiyle farklı derin öğrenme modelleri tasarlanmasına olanak sağlanmıştır. Tasarlanan bu modeller giriş verisinden kullanıcı tarafından belirlenen özellikler olmadan öğrenme işlemini kendisi yapmaktadır. Bu öğrenme işlemini farklı katmanlarda veriye ait farklı özellikler keşfetmekle elde etmektedir[16]. Derin öğrenme, YSA ilk çıktığı yıllardan itibaren farklı görevleri yerine getirmek üzere çeşitlenerek 8 farklı kategoride meydana gelmiştir.

1- Tek Katmanlı Algılayıcılar
2- Çok Katmanlı Algılayıcılar
3- Evrişimsel Sinir Ağları

4- Tekrarlanan Sinir Ağları
5- Uzun/Kısa Süreli Bellek Ağları
6- Sıralı Modeller
7- Sığ Derinlikli Ağlar
8- GAN Ağları

2.2. Derin Öğrenme Tarihçesi

Yapay zekâya dayalı yöntemler 1950'li yıllardan itibaren yavaş bir gelişim gösterirken 2000'li yılların başında derin öğrenme mimarileriyle yeniden ortaya çıkmış ve gelişimi hızlandırılmıştır. Yapay zekânın yıllar içerisinde özelleşerek derin öğrenmeye doğru giden gelişimi Şekil 2.1'de gösterilmektedir.



Şekil 2.1. Yıllar içerisinde yapay zekâ gelişimi

1940 – 1960 arası yıllarda başlayan yapay zekâ teknolojisi Sibernetik(otonom) olarak ortaya çıkmıştır. İlk genel, makine öğrenmesi algoritması, 1965 yılında denetimli derin beslemeli çok katmanlı algılayıcı modeli ile Ivakhnenko ve Lapa adındaki araştırmacıların bir yayınında ortaya çıkarılmıştır. Ivakhnenko ve Lapa, söz konusu çalışmada en iyi özellikleri her katmanda seçen metotlar ile seçip bir sonraki katmana iletmesini sağlamışlardır[17]. Burada en iyi özellik çıkarımının yapılması için en küçük kareler yöntemi kullanılmış olup yapay sinir ağlarını uçtan uca eğitmek için geriye yayılım yöntemi tercih edilmemiştir. 2000'li yılların başında yapay zekâ metotları, donanımsal kifayetsizlikten ötürü her ne kadar YSA bir takım gizli katman ve düğüm sayılarının artırımı şeklinde iyileştirilmeye gidilmiş olsa da yetersiz kalmıştır. Fakat bilgisayar üretiminde GPU ve benzeri donanımsal iyileştirmeler ile birlikte karmaşık bir yapıya sahip olan ve

gizli katman sayıları artan yapay sinir ağlarının hesaplamalarında maliyet tasarrufu sağlanmış ve yapay zekâ işlevselliğini geri kazanmıştır.

1970'te XOR problemi ile çok katmanlı ve tek katmanlı model olarak kendini gösteren yapay zekâ 1980-1990 yılları arasında yerini bir alt dalı olan makine öğrenmesine bırakmıştır. 1980'li yıllarda temelleri atılmış olan derin öğrenme, 1990'lı yıllarda donanım kaynaklı engeller nedeniyle bir duraklama devri geçirmiş olsa da günümüzde yapay zekânın en popüler alt dalı olma özelliğini taşımaktadır. Her geçen gün gelişmekte olan bilgisayarlı görü, doğal dil işleme, otonom araçlar, müzik, sanat, savunma sanayii, güvenlik ve finans gibi birçok çalışma alanında uygulamaları bulunan derin öğrenme alanında çalışma yaparken yüksek hızlı grafik işlem birimlerine (GPU), yüksek kapasiteli belleklere ve/veya bulut çalışma ortamlarına ihtiyaç duyulmaktadır [18]. 2006 yılında ise öznitelik çıkarımı işlemi modelin içine alınmış olup, 2010 senesinin başında makine öğrenmesinin yetersiz kalan kısımları derin öğrenme modelleri ile geliştirilmiştir.

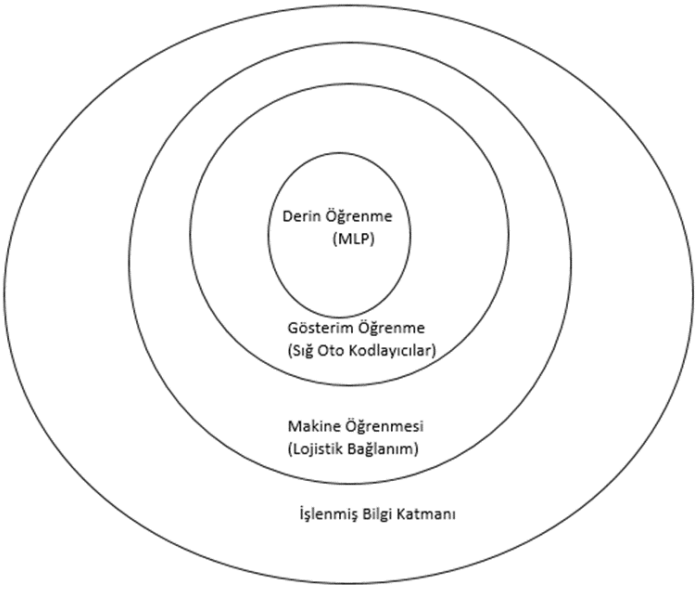
2.3. Yapay Zekâ

Yapay zekâ insan beyninin biyolojik özelliklerinden esinlenerek bir dananım ya da yazılıma insan gibi davranabilme kabiliyetinin kazandırılmasıdır [19]. Yapay zekâ, matematiksel bir yöntem olmayıp onun da ilerisinde yaklaşımsal bir yöntemdir. Burada hammaddemiz veridir. Söz konusu veriyi elde etmek ve rafine etmek oldukça önemlidir

Yapay zekâ iki temel bileşene sahiptir. Bunlar;

- Klasik Öğrenme Kuralları
- Biyolojik Öğrenme Kuralları

Makine öğrenmesi, kural tabanlı ve kümeleme mantıklı klasik öğrenme kurallarını çoğunlukla uygularken, derin öğrenme biyolojik öğrenme kurallarını kapsamaktadır. Yapay zekâ katmanları Şekil 2.2'de kapsamlarıyla verilmektedir.



Şekil 2.2. Yapay zekâ katmanları

Yapay sinir ağlarının ilk ortaya atıldığı günden itibaren günümüze gelene kadar birçok modellemesi yapılmıştır. XOR problemlerini çözebilmek için yapılan deneysel çalışmalar neticesinde Çok Katmanlı Algılayıcılar (MLP) ortaya çıkmıştır.

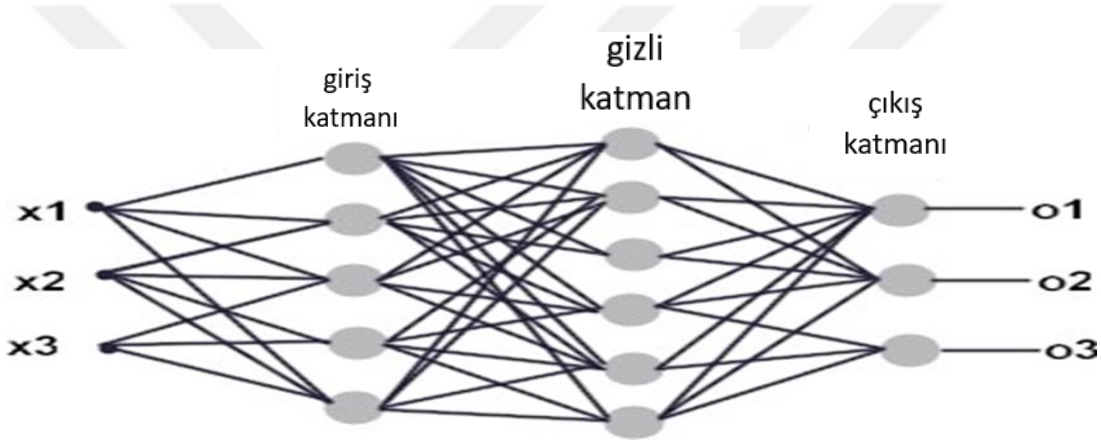
Katman sayılarına göre sınıflandırma yapıldığında ortaya çıkan modellerde tek katmanlı ve çok katmanlı yapay sinir ağları modeli oluşturulmuştur. Tek katmanlı algılayıcılar kendi içlerinde ikili modelleme ve algılayıcılar olarak ayrılmıştır. Çok katmanlı algılayıcılar ise en az bir adet gizli sinir ağı katmanı ve hesaplama kabiliyeti bulunan sinir ağlarından oluşan yapıya denilmektedir[20-22].

Bir yapay sinir ağının çok katmanlı olabilmesi için bir adet giriş, en az bir adet gizli ve bir adet çıkış veya sınıflandırma katmanı bulunması gerekmektedir. Giriş katmanında alınan değerler birinci nörondan ağırlık değerleri ile gizli katmana iletilerek matematiksel hesaplama işlemi gerçekleştirilir. Sonucunda kullanılan aktivasyon fonksiyonları ile çıkış katmanına iletilirler. İletilen değerlere göre bir sınıflandırma yapılır ve hata oranı tespit edilir. Elde edilen hata oranına göre ağırlık ve giriş değerlerinde güncelleme yapılır ve aynı işlem tekrar edilir. Elde edilen sonuç en uygun seviyeye geldiğinde ağ eğitim sonuçlandırılır ve yeni gelecek ağın daha önceden işlem yapmadığı veriler ile tahmin işlemi gerçekleştirilir. Çok katmanlı algılayıcılar öğrenmeyi en küçük kareler yöntemine göre gerçekleştirmektedir. Bu yöntem ise Delta Öğrenme Kuralının geliştirilmiş halidir. Delta öğrenme kuralı iki aşamada gerçekleştirilmektedir[23-25].

1- İleri Doğru Hesaplama
2- Geriye Doğru Hesaplama

İleri doğru hesaplama ağda bulunan girdi değerleri ile girdilerin ağırlık değerlerinin hesaplanması ile gizli katmanda kullanılan aktivasyon fonksiyonları ile belirli bir değer göstererek çıktı katmanına iletir. Çıktı katmanında elde edilen sonuç ile girdi katmanındaki veriye ait gerçek sonuç karşılaştırılarak hata değeri elde edilmektedir. Bundan sonraki hesaplamalar geriye doğru hesaplama yapılarak en uygun değer elde edilmeye çalışılır.

Bu hesaplama da elde edilen hata değerlerine göre ağ girdi değerlerini güncelleyerek ağın eğitilmesi için en uygun ağırlık değerlerini hesaplar ve ağ güncellemelerden sonra yeni gelen veriler üzerinde kestirimler yapılarak sınıflandırma işlemi gerçekleştirilmektedir. Özellikle sınıflandırma ve genelleme yapma durumlarında çok katmanı denetleyiciler (MLP) etkin çalışır. Girdi ile çıktı arasında bir linerlilik söz konusudur[26-32]. Derinliği arttırdıkça performansının arttığı gözlemlenmiştir. İleri beslemeli derin ağlar Şekil 2.3'te şematize edilmiştir.



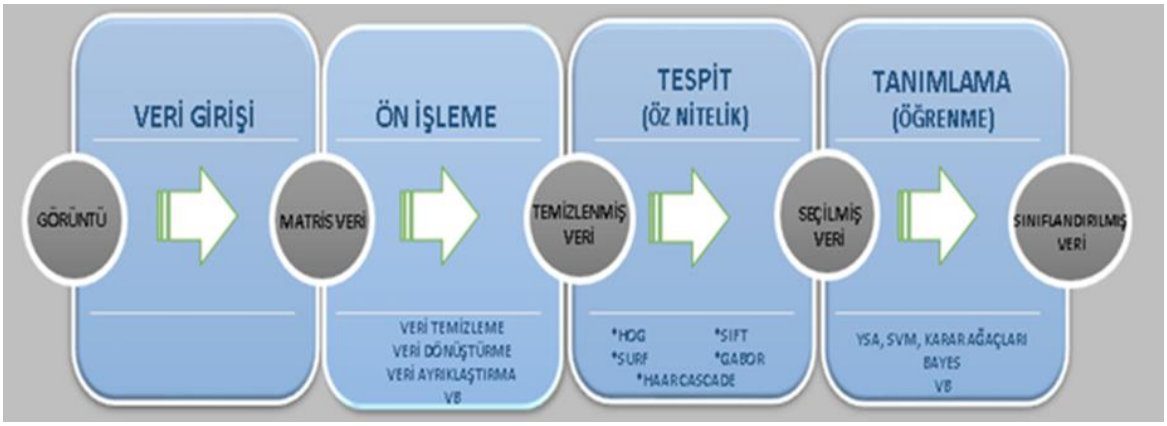
Şekil 2.3. İleri beslemeli derin ağlar

2.4. Derin Öğrenme Süreçleri

Başından sonuna kadar bir eğitim sırasında kaydedilen aşamalar aşağıdaki gibidir;

1. Öncelikle bir verinin Girdi ve Çıktısı oluşturulmalıdır. Girdi verilerine matematiksel olarak x, çıktı verilerine ise y denildiği durumda x→ y giriş ve çıkış verilerinin tanımlı olup olmadığı sorgulanmaktadır.
2. Daha sonra verilerden farklı olanlar çıkarılıp atılmaktadır.
3. İlk iki aşamayı geçtikten sonra hangi derin öğrenme algoritması uygulanacağının tespiti yapılmaktadır.
4. Bu aşamada artık eğitim işlemi başlamaktadır, veriler eğitime tabi tutulmaktadır. Eğitim sonucunu ölçen performans kıstaslarına göre ise eğitim aşamasının geçip geçmediği anlaşılmaktadır.
5. Nihai olarak son aşamada uygulama kısmına geçilmektedir.

Derin öğrenmenin söz konusu aşamaları Şekil 2.4'te gösterilmektedir.



Şekil 2.4. Derin öğrenme aşamaları

2.5. Derin Öğrenme Mimarileri

Binlerce imge kullanılarak eğitime tabi tutulmuş farklı derin öğrenme modelleri bulunmaktadır. Bu modellerden öne çıkanlar; AlexNet, GoogleNet, ResNet, VGG gibi derin öğrenme mimarileridir. Bu metotların hepsi birbirinden değişik model geliştirerek eğitim süreçlerini sürdürmektedir. Bunlardan AlexNet, bir milyondan fazla imge ile eğitimini tamamlayarak büyük bir hacme ulaşmıştır. Bununla birlikte AlexNet mimarisiyle imgeler 1000 farklı kategoriye ayrılabilmektedir. AlexNet mimarisi beş konvolüsyon katmanı ve üç tam bağlı katman olmak üzere 8 katmanlı bir yapı göstermektedir[4]. Önemli mimarilerden biri olan VGG16 modeli ise milyonlarca imge ile eğitilme tabi tutulmuştur. Bu mimari de binlerce farklı sınıfı ayırt edebilir ve 5 konvolüsyon katmanlıdır. AlexNet ve VGG-16 dışındaki diğer mimariler ise; Google Net, VGG-19, Derin Oto Kodlayıcılar, RBM, LSTM, RNN, LeNet, CNN, ZFNet, ResNet olarak bilinmektedir[33-40]. Geliştiricileri ile birlikte geliştirildikleri yıllar içerisindeki derin öğrenme mimarileri Tablo 2.1'de gösterilmektedir.

Tablo 2.1.Yıllara göre derin öğrenme mimarileri ve geliştiricileri

Yıl	CNN	Geliştirici
1998	LeNet	Yann LeCun
2012	AlexNet	Alex Krizhevsky, Geoffrey Hinton, Ilya Sutskever
2013	ZFNet	Matthew Zeiler and Rob Fergus
2014	Google Net	Google
2014	VGG Net	Simonyan, Zisserman
2015	ResNet	Kaiming He

2.6. Derin Öğrenmenin Kullanıldığı Alanlar

Yapay zekânın en önemli kullanım alanı görüntü işleme, ses tanıma ve doğal dil işleme olarak bilinmektedir. Bu alanlardaki kullanım, sektörlere göre açıklandığında otomotiv sektöründeki makine öğrenimi kullanımını sensörlerden alınan verilere dayanan insansız araçlara benzer örnek verilebilmektedir.

Konuşma tanıma teknolojisi, yüzde 95 civarında bir doğruluk oranı ile kullanıcıları dinleyerek öğrenme işlemini gerçekleştirmektedir.

Fen bilimleri ve tıp alanlarında ise derin öğrenme modelleri kullanımı, doktorların yaptığı teşhisi destekler nitelikte katkı sağlayarak MR cihazlarındaki tarama verileri üzerinden kanserli hücre göstergelerini tespit etmesi şeklinde örneklerde söz konusu olmaktadır.

Edebiyat alanında Osmanlıca harfleri tanımaya ilişkin bir YSA tasarlamış ve uygulamıştır[13].

Pazarlama alanında, makine öğrenimi ile e-postaların duygu analizini yapılmaktadır. Geliştiriciler tarafından fotoğraf ve video verileri eğitilerek gerçek zamanlı duygu tespiti yapılabilmektedir.

Siyaset bilimi alanında da yine duygu analizi, sosyolojik açıdan izleyicilerin duygusal reaksiyonlarına göre içeriklerin işlenmesi gibi örneklerin ileride kullanılması beklenmektedir.

2.7. Günümüzde Derin Öğrenme

Günümüzde derin öğrenme, yazılım devi olan Google, Microsoft, Apple, NEC, IBM gibi birçok teknoloji şirketi tarafından etkin ve karlı bir şekilde kullanılmakta ve güçlü bir yazılım alt yapısına entegre edilerek geliştirilmektedir.

PyLearn2, Torch, DistBelief, MXNet, TensorFlow, YOLO, Caffe gibi yazılım kütüphaneleri mevcuttur.

3. NESNE TANIMA ALGORİTMALARI

Nesne tanımada, doğruluk ve hız performansı artırımı için farklı algoritmalar geliştirilmiştir. Bilime emek veren araştırmacılar tarafından nesne tespiti için derin öğrenme algoritmaları kümeli bir şekilde gelişmektedir. Yaya algılama, tıbbi görüntüleme, robotik, sürücüsüz arabalar, yüz algılama gibi çeşitli popüler uygulamalar birçok alanda insanlara kolaylık sağlamaktadır. Nesne tanımanın geniş alanda incelenmesi ve sürekli gelişen son teknolojiler nedeni ile tümünü tek bir perspektifte toplamak zor bir işlemdir. Önceki çalışmalarda uzaktan algılanmış görüntüler üzerinde şablon oluşturma ve benzerlik ölçümlerinin altyapısını oluşturduğu bu yöntemin bir kolu olan katı şablon eşleştirmesini kullanarak binaların, kıyı kesiminin ve su tanklarının ve yolların tespiti başarılmıştır [14]. Bu çalışmada, derin öğrenme yöntemlerinin görüntü işleme alanındaki kullanışlı olması ve uygulanabilirliği gözetilmiş ve temel olarak derin öğrenme yöntemleri ve görüntü işleme algoritmaları kullanılmıştır.

3.1. Tensorflow Algoritması

Tensorflow Google'ın Brain ekibi tarafından geliştirilen, genel itibariyle makine öğrenimi için kullanılan çoğunlukla Python dilinde yazılmış açık kaynak kodlu ücretsiz bir yazılım kütüphanesidir. Aynı zamanda bir derin öğrenme kütüphanesi de denilebilen TensorFlow'un en önemli özelliği, herhangi bir platforma bağlı kalmadan hesaplama yapabilme yeteneğidir.

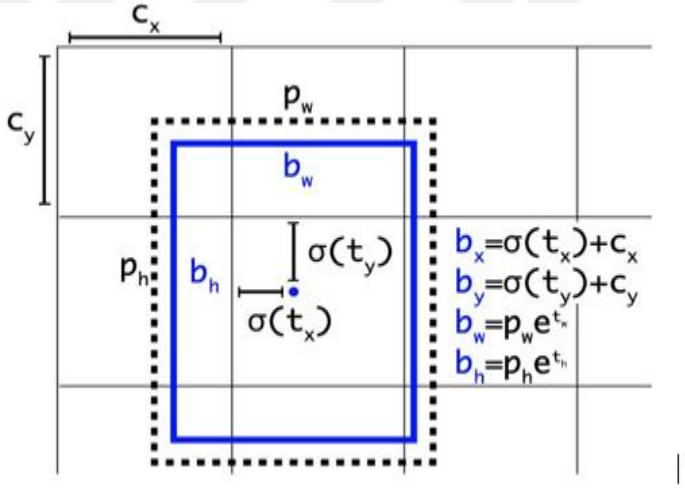
TensorFlow, Python dilinde geliştirilmiş olmasına karşın, son zamanlarda Java, JavaScript, C++, C# gibi dilleri de desteklemektedir. Öğrenme tabanlı bu kütüphane kullanıcılara herhangi bir tarayıcıdan çalışma imkânı sağlarken Google, çalışma ortamlarında pek çok kullanım ve geliştirim imkânı sağlamaktadır. TensorFlow, bir ya da daha fazla merkezi işlem birimi (CPU) ve grafik işlemlerini yürüten işlemcileri (GPU) kullanarak, harekete geçirme, konuşlandırma (dağıtma) işlemlerinin yapılmasını yürütmektedir[41].

3.2. YOLO Algoritması

Açılımı 'You Only Look Once' olan YOLO evrişimsel sinir ağları kullanarak nesne tespiti yapan bir algoritmadır. YOLO algoritması nesne tespitini çok hızlı ve tek seferde yapabildiği için ona bu ad verilmiştir. Bu algoritma diğer nesne tespiti algoritmalarının önüne geçmiştir. Çünkü YOLO diğer algoritmalardan farklı olarak imgenin tamamını tek bir seferde sinir ağdan geçirmektedir. Bu da ona hız ve performans yüksekliği sağlamaktadır. Tek aşamalı nesne algılama yöntemlerinin hızla gelişmesiyle birlikte, çok ölçekli tahmin kutuları ve derin bir omurga ağı ile gelişmiş performans sergileyen YOLO 'nun en popüler ve kararlı sürümü Redmon ve Farhadi (2018) tarafından tanıtıldı [16]. YOLOv4'e birkaç şaşırtıcı yeni özellik sundu ve YOLOv4,

doğruluk ve hız açısından YOLOv3'ü büyük bir farkla geride bıraktı. Yakın zamanda Jocher ve ark. (2020), olağanüstü iyileştirmelerle YOLOv5'i ve YOLOv5'in PyTorch tabanlı bir sürümünü tanıttı. Uçtan uca ağ yapısı, diğer ağlara göre daha yüksek bir algılama oranı sağlar [17].

YOLO algoritması görüntü üzerinde tespit edilecek nesnelerin çevresini sınırlayıcı kutu adı verilen kutular ile çevrelemektedir. Çalışma prensibi olarak YOLO sisteme girdi olarak verilen görüntüyü NxN'lik ızgaralara bölerek işleme başlamaktadır. Bu her ızgara kendi içerisinde var olan nesnelerin tespitini ve bu nesnelerin merkez noktasının o ızgara içerisinde olduğunun veya olmadığının tespitini yapmaktadır [18-19]. Izgaralardan nesnenin kendi içinde olan var ise ve merkez noktasına sahip olduğuna karar vermişse o nesnenin sınıfını, yüksekliğini ve genişliğini bulup o nesnenin çevresini sınırlayıcı kutu ile çevrelemektedir. Izgaralardan birçoğunun içinde nesne bulunabilmesi durumunda imgede çok fazla sınırlayıcı kutular oluşma durumu ortaya çıkabilmektedir. Fakat tüm sınırlayıcı kutuların güven skoru bulunmaktadır. Güven değeri en yüksek olan kutu ekrana 'en yüksek olmayan bastırma' algoritması aracılığıyla çizilmektedir. Şekil 3.1'de boyut öncelikleri ve konum tahmini içeren sınırlayıcı kutu gösterilmektedir.



Şekil 3.1. Boyut öncelikleri ve konum tahmini içeren sınırlayıcı kutu

Bu şekilde bx: Nesnenin orta noktasının x koordinatı, by: Nesnenin orta noktasının y koordinat, bw: Nesnenin genişliği, bh: Nesnenin yüksekliği anlamlarına gelmektedir.

YOLOv3 algoritması ise tahmin yaparken nesne tespitini tek bir regresyon problemi olarak değerlendirmesi temeline dayalıdır. YOLOv3 algoritması, kutuları 3 farklı ölçekte tahmin etmektedir. Her ölçü için 3 farklı kutu tahmin edip giriş görüntüsünü NxN'lik ızgaralara bölmektedir.

$$N*N*N(3*(A+B+C)) \quad (3.1)$$

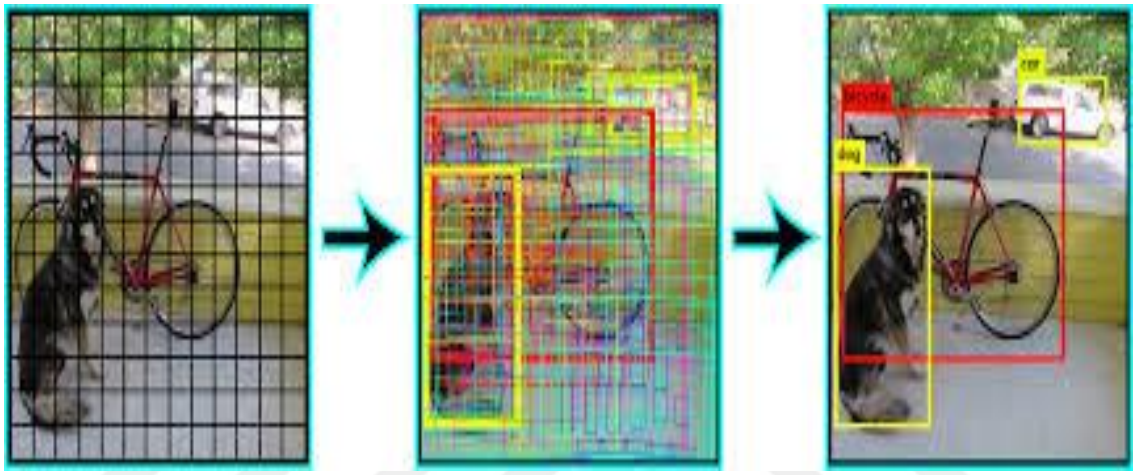
Burada gösterilmekten olan harfli değerler şöyledir;

A: Sınırlayıcı kutu uzaklıkları

B: Nesne tahmini

C: Sınıf tahminleri

YOLOv3'te ağın başından beri bir özellik çıkarımı yapılarak özellik haritası belirlenir. Daha sonra birleştirme kullanılarak bu harita örneklenmiş özellikler ile bütünleştirilir. Bu şekilde işleyen metodun amacı bir önceki özellik haritasından daha anlamlı özelliklerin alınabilmesini ve özellik haritasını daha anlamlı ve detaylı tarayabilmektir. Bir imge üzerinde köpek, kuş ve araba nesnelerini tanınması örneği Şekil 3.2'de gösterilmektedir.



Şekil 3.2. YOLO nesne tespitinde ızgara örneği

4. MATERYAL VE METOT

Uydu imgelerinin çeşitli sınıflara göre kategorilendirerek toplanması, bu görüntülerin sınıflarının etiketlenmesi, derin öğrenme metotları ile sınıflandırılması, nesne tanıma algoritmalarının kullanılması ve performanslarının ölçümü için çeşitli işlemler ve araçlar kullanılmıştır. Bu bölümde imgelerin toplanmasından sınıflandırılmasına, uygulanan algoritmalara ve ortaya çıkan grafiklere kadar tez için kullanılan materyal ve metotlar açıklanmaktadır.

4.1. Materyal

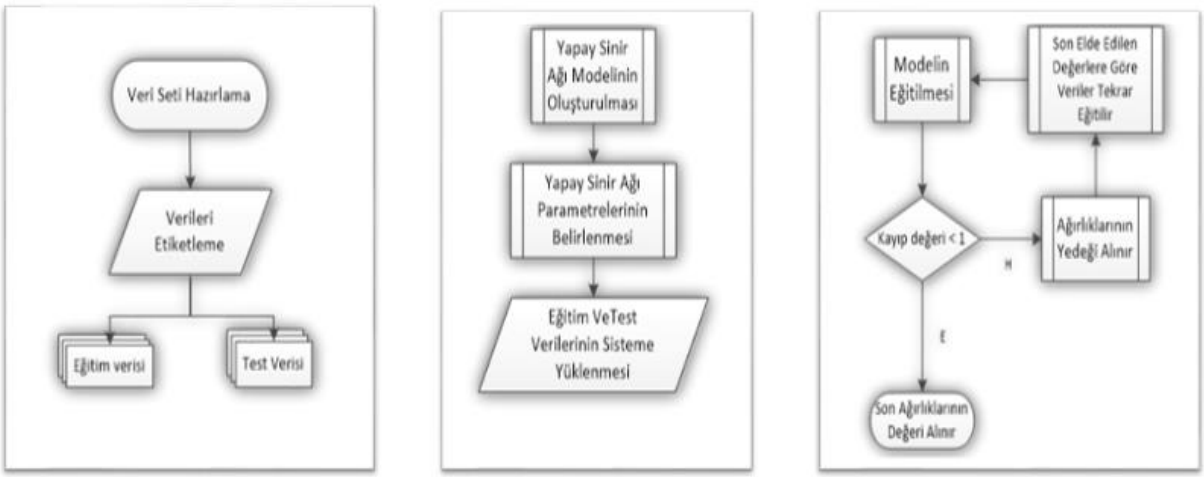
Tezin temel amacı uydu imgelerinin sınıflandırılma performansını arttırmak üzere imgelere derin öğrenme yöntemler ile nesne tanıma algoritmaları uygulamaktır. Bu çalışmada özgün ve gerçek zamanlı veri seti oluşturmak adına uydu imgeleri kullanılmıştır. Ana sınıflar belirlenirken uydu imgelerinden yararlanılmış ve ayırt edici özellikleri bulunan cami, hastane, havalimanı, okul ve saha olmak üzere toplamda 5 sınıf belirlenmiştir.

Sınıflandırmada performans ölçümü değerlendirme amaçlı Türkiye'deki çeşitli illerden cami, hastane, havalimanı, okul ve saha sınıflarına ait, her sınıf için 60 imge açık erişim imkânı bulunan uydu görüntüleri yardımı ile toplanmıştır. Toplamda 300 imge üzerinde etiketleme ve nesne tanıma algoritmaları uygulanmıştır.

Öncelikle açık erişim imkânı sunan Google Earth Pro programı kullanılarak bu çalışmaya uygun uzay görüntüleri seçilmiştir. Elde edilen bu imgelere grafiksel bir görüntü açıklama aracı olan LabelImg (Windows_v1.8.0) aracılığıyla etiket verilmiştir. Etiketleme işleminin ardından YOLOv3 ve YOLOv5 algoritması kullanarak verilerin işlenmesi için Google Colaboratory laboratuvarı kullanılmıştır. Tez çalışmasında makine öğrenmesi uygulamaları için bir sanal makine üzerinden GPU kullanılmıştır. Google Colab kullanımı ücretsiz bulut, GPU ve sanal makine imkânı sağlamıştır. Böylece kullanılan bilgisayardan bağımsız bir RAM ve diske istenilen her cihazdan erişim sağlanmıştır. Dezavantajı ise GPU kullanımı için dinamik değişen kullanıcı yoğunluk durumuna göre giriş yapılmasıdır. Buna ek olarak GPU'ya sürekli bağlantı sağlanamamasıdır.

4.2. Metot

Bu tez çalışması Şekil 4.1'de görüldüğü gibi üç aşamadan oluşmaktadır. İlk olarak veri setinin hazırlanması ile alakalı süreçler, ikinci aşamada derin öğrenme için kullanılacak evrişimsel sinir ağının oluşturulması ve uyum bütünleşmesi, üçüncü aşamada ise veri setinin eğitilmesi ve diğer ortamlarda da kullanılmak üzere ağırlık parametrelerinin elde edilmesi gerçekleştirilmiştir.

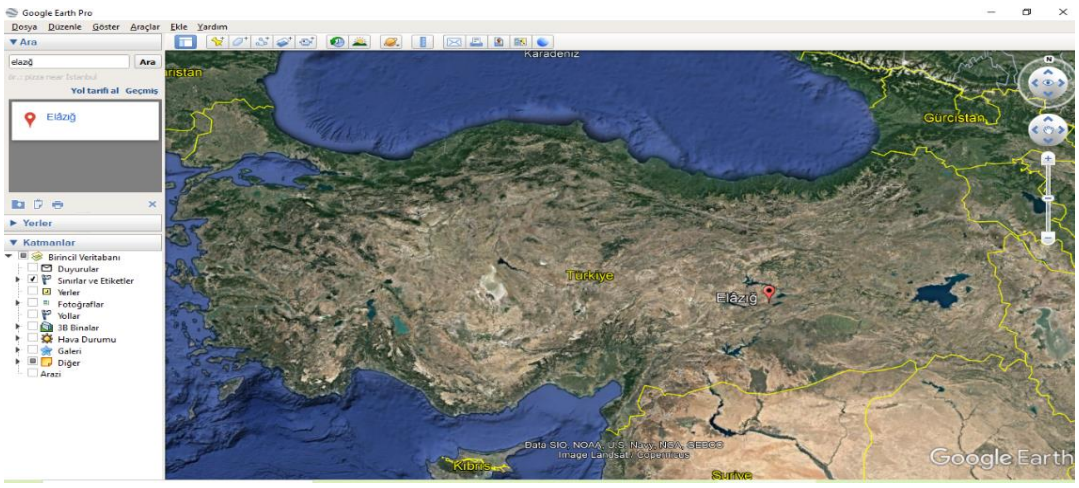


Şekil 4.1. İş akış diyagramı

4.2.1. Veri Seti Hazırlama ve Etiketleme

Bu çalışmada veri seti internet üzerinden Google Earth [42] aracılığı ile tarafımızca oluşturulmuştur. Veri hazırlama ve etiketleme aşamasında, uzay görüntülerinden elde edilen imgeler belirli işlemlerden geçirilerek etiketli bir veri setine dönüştürülmüştür.

Görüntüler Türkiye'nin çeşitli bölgelerinde bulunan hastane, okul, saha, cami, havalimanı gibi halka açık, sürekli kullanılan yerlerden sağlanmıştır. Çalışmanın veri setini oluşturan bu görüntüler Google Earth Pro programı aracılığıyla gerçek ve güncel imgeler şeklinde temin edilmiştir. Türkiye'nin birçok yerinden elde edilen bu uzay görüntülerine ait örnek görseller Şekil 4.2'de mevcuttur.



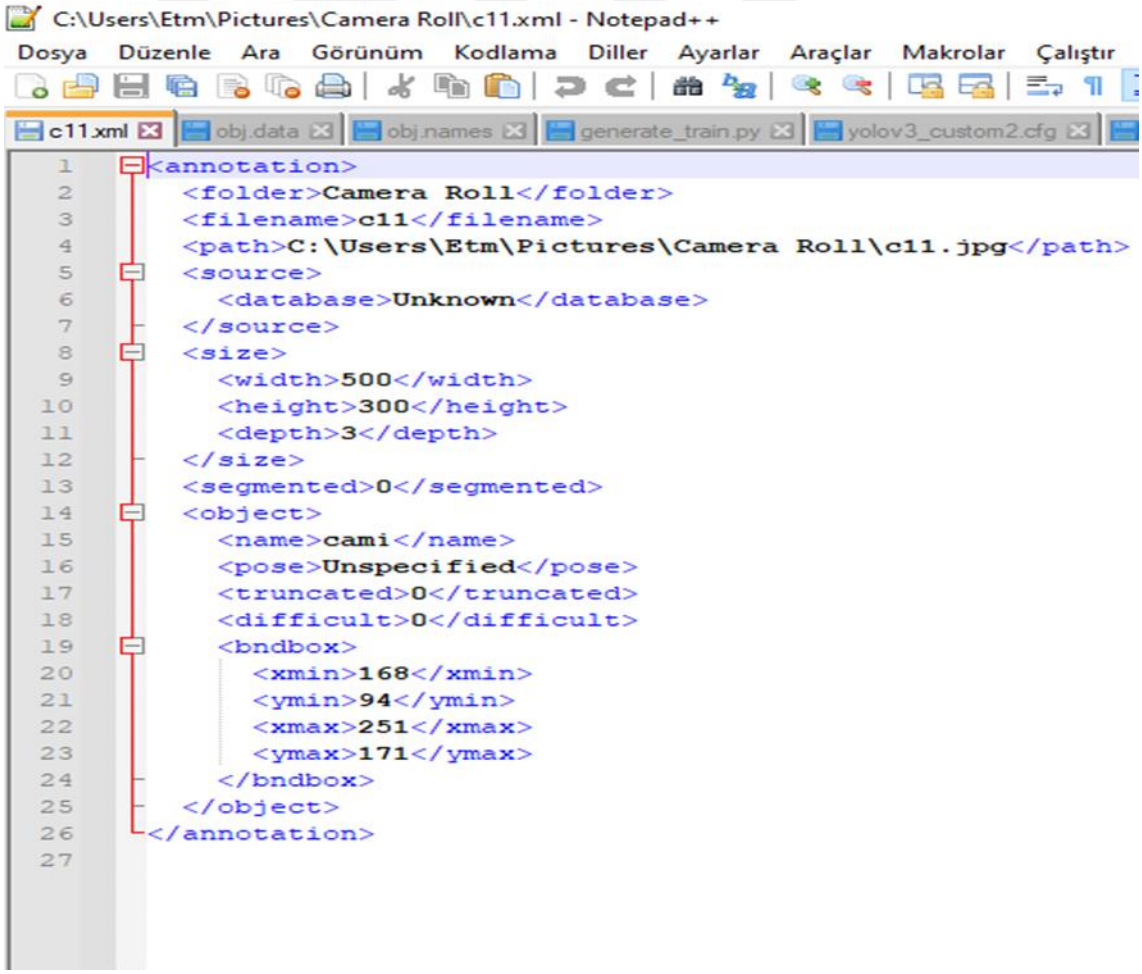
Şekil 4.2. Google Earth aracılığıyla Türkiye

Derin öğrenme yöntemiyle sınıflandırılmasını ve performans ölçümünü amaçlayan bu tez çalışmasında elde edilen görüntülerin makine öğrenmesine uygun hale getirilmesi gerekmektedir. Görüntü üzerindeki nesneleri tanıması için öncelikle nesnelerin belirtilmesi gerekmektedir. Bunun için tanınması istenen nesnelerin isimleri etiketleme yöntemi ile belirtilerek eğitim sonrasında aynı isimlerin kullanılması amaçlanmaktadır. Tüm bu amaçlar için öncelikle uydu imgeleri toplanarak bir veri seti oluşturulmuştur. Programın araçları yardımıyla çeşitli ikon ve işaretlerden temizlenerek belirlenen yerlerin görüntüleri alınıp jpg dosya uzantısı şeklinde kaydedilmiştir. Bu işlem sonunda 300 görüntülük bir veri seti oluşturulmuştur.YOLOv3 ve YOLOv5 algoritmalarına dayalı eğitim modelinde toplamda 300 imge ilk önce 5 farklı sınıfa ayrılarak her bir sınıf için etiketleme yapılmıştır. Veri setinin kolay anlaşılabilir olması için tespit edilmesi istenen ana nesneyi temsil edecek nesnenin baş harfi ve numaralar verilmiştir. Buna göre 60 cami, 60 hastane, 60 havalimanı, 60 okul ve 60 saha olmak üzere 5 sınıfa ayrılmıştır. Şekil 4.3'te veri setinden örnek görüntüler gösterilmektedir.

Şekil 4.3. Veri setinden örnek görüntüler

Görüntüleri etiketlemek için etiketleme aracı olan LabelImg kullanılmıştır. Etiketleme işlemi yapılırken sonraki adımlarda Python programlama dili ile çalışan YOLO algoritması kullanılacağı için etiketlerin Türkçe yazı karakteri içermemesine dikkat edilmiştir. LabelImg uygulamasının son sürümü sayesinde etiketleme işlemi YOLO için uygun olan seçenek işaretlenerek gerçekleştirilmiştir. Etiketlenmiş imgeler, etiket işlemi yapılmamış imgeler ile aynı dosyanın içine xml uzantısı ile aynı isimlerde kaydedilmiştir. Bu şekilde görüntü dosyaları etiketlenerek xml formatında text dosyalarına dönüştürülmüştür. Görüntüler içerisindeki etiketlerin koordinatları text dosyaları içerisine kaydedilmiştir.

Örnek bir c11 adlı görüntü dosyasının içeriğini Şekil 4.4.'te görmek mümkündür. Burada imgenin genişlik(widht), yükseklik(height), etiket adı(name) ve etiketin görüntünün neresinde konumlandığı bilgisini veren satırlar (Xmax, Xmin, Ymax, Ymin) ihtiva edilmektedir. Bu işlem 300 verinin hepsine uygulanarak jpg formatlı dosyaları ile birlikte toplam 600 veriden oluşan bir veri seti hazırlanmıştır.



Şekil 4.4. Görüntünün text dosyası içeriği

Örnek teşkil etmesi için seçilen cami nesnelerinin etiketlenmesi işlemi Şekil 4.5' te verilmiştir. Programın sol alt köşesinde bulunan ünite aracılığıyla nesne seçimi yapılmakta ve seçilen nesneye bir isim tayin edilmektedir. Bu isim o nesnenin etiketi olarak kaydedilmektedir. Bununla birlikte hastane, okul, havalimanı ve saha etiketleme işlemleri de Şekil 4.6'da gösterilmektedir.

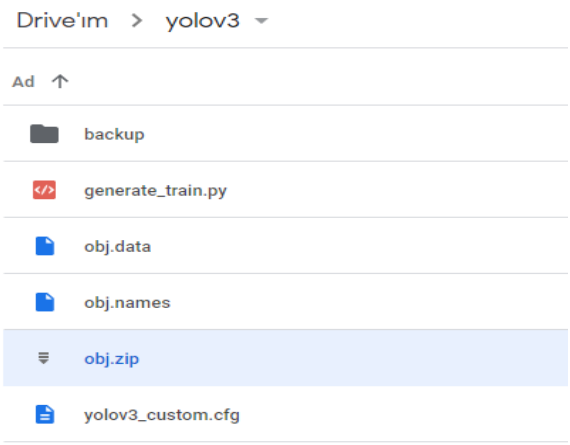
Şekil 4.5. LabelImg aracı ile etiketleme işlemi

Şekil 4.6. Etiketlenmiş görüntü örnekleri

Bu etiketleme aşaması bitirildikten sonra veri seti, %80'i eğitim, %20'i test verisi olacak şekilde ayrıştırılmıştır.

4.2.2. Veri Setinin Sanal Sunucuya Yüklenmesi

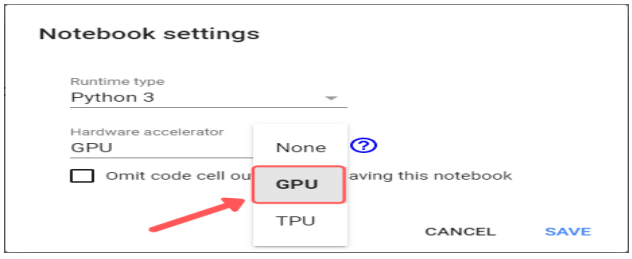
Bu aşamada hesap doğrulaması yapılarak giriş işlemi gerektiren Google Drive kullanarak Google Colab not defteri üzerinden işlem yapmak amacıyla veri seti, obj. zip adlı bir dosyaya kaydedilerek sıkıştırılmıştır. Daha sonra bu sıkıştırılmış veriler Google Drive'a yüklenilmiştir. Google Drive'da obj.zip dosyasının aldığı yer Şekil 4.7'de gösterilmiştir.



Şekil 4.7. Obj.zip dosyasının yeri

4.2.3. Sanal GPU Tercih Edilmesi

Google Drive aracılığıyla Google Colab ortamında işlemlerin hızlı sonuç verebilmesi için sanal ve boş bir GPU'ya bağlanılmıştır. Şekil 4.8'de GPU seçme işlemi gösterilmektedir. Google Colab çalışmamıza başlamadan önce GPU seçimi yapılarak sürücünün içeriğine erişebilmesini sağlama amacıyla bulut sanal makineye yerleştirmek için Şekil 4.9'daki komut hücresi çalıştırılmıştır.



Şekil 4.8. Sanal GPU seçme işlemi

```
[ ]  %cd ..
     from google.colab import drive
     drive.mount('/content/gdrive')

     /content
     Mounted at /content/gdrive
```

Şekil 4.9. Sanal makineye bağlanma komut satırı

Çalışmamızda en hızlı ve yeni teknoloji ile tespit edilebilirliği içerdiği için YOLOv3 Darknet framework üzerinde kullanılmıştır. Makefile dosyası ve OPENCV Kütüphanesi eklenmiştir. Bununla birlikte Nvidia tarafından geliştirilmiş bilgisayarın işlem performansına yüksek oranda katkı yapan bir paralel programlama platformu Cuda kurulmuştur. Kurulum kodu Şekil 4.10' da verilmiştir.

```
# verify CUDA
!/usr/local/cuda/bin/nvcc --version
```

```
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2020 NVIDIA Corporation
Built on Mon_Oct_12_20:09:46_PDT_2020
Cuda compilation tools, release 11.1, V11.1.105
Build cuda_11.1.TC455_06.29190527_0
```

Şekil 4.10. Cuda kurulumu için komut satırı

Darknet klonlanarak GPU ve OpenCv kütüphanesini etkinleştirmeye yönelik Şekil 4.11' deki komut dizini çalıştırılmıştır.

```
[ ]  # clone darknet repo
     !git clone https://github.com/AlexeyAB/darknet
```

```
fatal: destination path 'darknet' already exists and is not an empty directory.
```

```
[ ]  # change makefile to have GPU and OPENCV enabled
     %cd darknet
     !sed -i 's/OPENCV=0/OPENCV=1/' Makefile
     !sed -i 's/GPU=0/GPU=1/' Makefile
     !sed -i 's/CUDNN=0/CUDNN=1/' Makefile
```

```
/darknet
```

Şekil 4.11. Darknet klonlanarak GPU ve OPENCV etkinleştirme komut satırı

4.2.4. Konfigürasyonların Yapılması

Uzay görüntülerinden oluşturulmuş veri seti obj adlı bir klasör ile önce Google Drive'a daha sonra darknetin veri klasörüne yükleyerek sanal makinede işlem yapılması ve darknetin özelleştirilmiş bir config dosyası hazırlanıp eklenerek eğitim için gerekli konfigürasyon sağlanmıştır. Bu dosya içeriğinde Şekil 4.12'den anlaşılacağı üzere sınıf sayısı 5, filter sayısı, (sınıf sayısı+5)*3 formülüyle 30 değerini almıştır. Max batch değeri ise 10.000 olarak hesaplanmıştır.

```
[yolo]
mask = 6,7,8
anchors = 10,13,  16,30,  33,23,  30,61,  62,45,  59,119,  116,90,  156,198,  373,326
classes=5
num=9
jitter=.3
ignore_thresh = .7
truth_thresh = 1
random=1
```

Şekil 4.12. Konfigürasyon dosyası içeriği

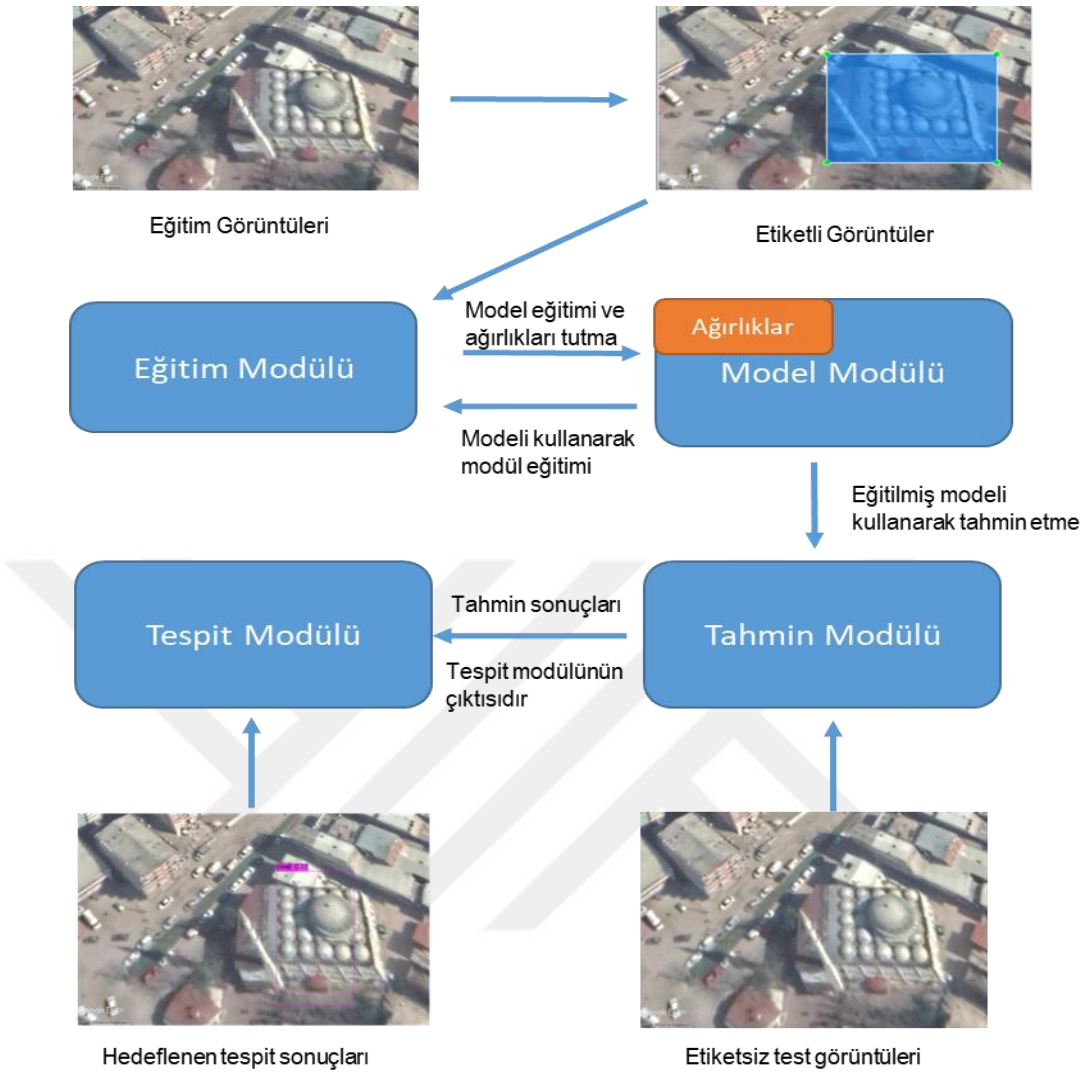
Gerekli konfigürasyon dosyası yüklendikten sonra 'content/gdrive/My\Drive/' klasörü ile '/mydrive' klasörü arasında sembolik bir bağlantı oluşturularak sürücüye erişilmesi için Şekil 4.13'te belirtildiği gibi sadece bir mydrive kısa yolu oluşturulmuştur.

```
[ ] # this creates a symbolic link so that now the path /content/gdrive/My\ Drive/ is equal to /mydrive
    !ln -s /content/gdrive/My\ Drive/ /mydrive
    !ls /mydrive
```

Şekil 4.13. Mydrive kısayolu oluşturma komut dizini

4.2.5. YOLO Modeli Oluşturma ve Eğitim

YOLO algoritmasının çalışma akış diyagramı Şekil 4.14'te görüldüğü gibi etiketleme işleminden sonra eğitim gerçekleştirilir. Eğitilen model tekrar eğitilebilir halde işleme girdi olarak dâhil edilir. İşlem basamaklarına adım denilen bu eğitimler en az 100 adım ilerleyince ağırlıkları tekrar kullanmak üzere yedeklenir. Model ne kadar çok eğitime maruz kalırsa tahmin değeri o kadar artacağı bilinmektedir. Sisteme verilen herhangi bir etiketli nesnelerin bulunduğu görselde istenilen nesneyi tespit edip doğruluk oranı verecektir.



Şekil 4.14. YOLO algoritması akış diyagramı

Eğitime başlamak için öncelikle drive içerisine obj.name ve obj. data dosyaları Şekil 4.15'te gösterildiği gibi yüklenmiştir.

```
[ ]  # upload the obj.names and obj.data files to cloud VM from Google Drive
     !cp /mydrive/yolov3/obj.names ./data
     !cp /mydrive/yolov3/obj.data  ./data

     # upload the obj.names and obj.data files to cloud VM from local machine (uncomment to use)
     #%cd data
     #upload()
     #%cd ..
```

Şekil 4.15. İsim ve veri dosyaları yüklenmesi için komut dizini

Ağırlıkların yedeklenebilmesi amacıyla ise drive içerisine bir backup dosyası açılmıştır. Eğitim için gerekli olan generate_train.py adlı python dosyası da yine drive içerisine yüklenmiştir. Drive içerisinde bulunan dosyalar Şekil 4.16' da gösterilen komut satırı çıktısında yer almaktadır. Daha sonra evrişim katmanları için önceden eğitilmiş ağırlıkları indirilmiştir. Bu adım ile YOLOv3 ağının evrişimli katmanlarının ağırlıkları indirilmiştir. Bu ağırlıklar kullanılarak oluşturulacak özgün nesne detektörünün çok daha doğru olmasına ve uzun süre antrenman yapılmasına gerek kalmamasına yardımcı olması sağlanmıştır.

```
[ ]  # upload pretrained convolutional layer weights
     !wget http://pjreddie.com/media/files/darknet53.conv.74

     URL transformed to HTTPS due to an HSTS policy
     --2022-05-23 13:19:34--  https://pjreddie.com/media/files/darknet53.conv.74
     Resolving pjreddie.com (pjreddie.com)... 128.208.4.108
     Connecting to pjreddie.com (pjreddie.com)|128.208.4.108|:443... connected.
     HTTP request sent, awaiting response... 200 OK
     Length: 162482580 (155M) [application/octet-stream]
     Saving to: 'darknet53.conv.74'

     darknet53.conv.74   100%[===================>] 154.96M  55.4MB/s    in 2.8s

     2022-05-23 13:19:37 (55.4 MB/s) - 'darknet53.conv.74' saved [162482580/162482580]
```

Şekil 4.16. Evrişim katman ağırlıkları yüklenmesi

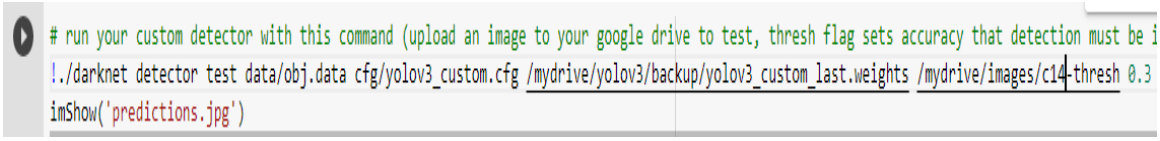
Eğitimin işleminin gerçekleşebilmesi için son aşamada darknet detector train komutu ile antrenman gerçekleştirilmiştir. Bu antrenman komut satırı Şekil 4.17'de gösterilmektedir.

```
[ ]  # train your custom detector
     !./darknet detector train data/obj.data cfg/yolov3_custom.cfg darknet53.conv.74 -dont_show

     Görüntülenen çıkış son 5000 satıra kısaltıldı.
      total_bbox = 53142, rewritten_bbox = 0.000000 %
     v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.710965), count: 1, class_loss = 1.670344, iou_loss = 0.520169, total_l
     v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.000094, iou_loss = 0.000000, total_l
     v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.000005, iou_loss = 0.000000, total_
      total_bbox = 53143, rewritten_bbox = 0.000000 %
     v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.583346), count: 1, class_loss = 1.372681, iou_loss = 0.208360, total_l
     v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 94 Avg (IOU: 0.000000), count: 1, class_loss = 0.000024, iou_loss = 0.000000, total_l
     v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 106 Avg (IOU: 0.000000), count: 1, class_loss = 0.000007, iou_loss = 0.000000, total_
      total_bbox = 53144, rewritten_bbox = 0.000000 %
     v3 (mse loss, Normalizer: (iou: 0.75, obj: 1.00, cls: 1.00) Region 82 Avg (IOU: 0.554448), count: 1, class_loss = 1.116804, iou_loss = 0.748227, total_l
```

Şekil 4.17. YOLO eğitimini başlatan komut dizini

Yapılan çalışmada görüntüler etiketlendikten sonra DarkNet-53 aracılığı ile Google Colab üzerinden özgün YOLO modeli eğitilmiştir. Eğitim aşamasından geçirilerek yaklaşık 1400 adımlık bir öğrenme sağlanmıştır. Bu işlem yaklaşık 5 saatte gerçekleştirilmiştir. Nesne tespiti için ise verilen herhangi bir uzay görüntüsünde tespit yapması istenen kod ise Şekil 4.18'deki gibidir. Örnek olarak c14.jpg görüntü dosyası tespit amacıyla verilmiştir.



Şekil 4.18. C14 dosyasında nesne tespiti için komut dizini

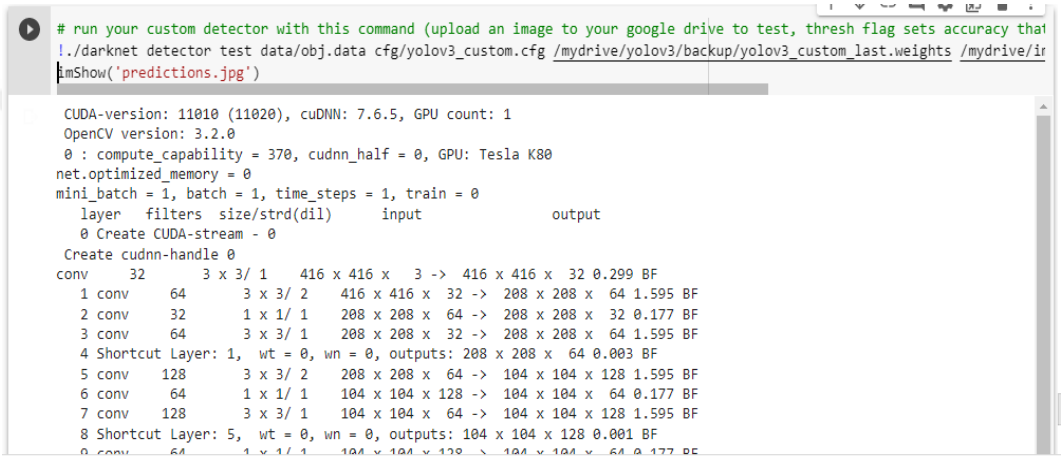
Eğitim grafikleri elde edildikten sonra imgelerin doğruluk oranları tespit edilmiştir. Uydu görüntülerinden elde edilen sınıflandırma sonuçlarının doğruluklarının etiketlenmesi, uzaktan algılama verilerinden elde edilen haritaların kalitesinin ve kullanılabilirliğinin değerlendirilmesi için gereklidir. Elde edilen görsel çıktılar, predictions. jpg adıyla gösterilmiştir. Sonuçları, bulgular ve tartışma bölümünde açıklanmıştır.

5. BULGULAR VE TARTIŞMA

Bu bölümde yapılan deneysel çalışmaların elde edilen çıktılarına ayrıntılarıyla yer verilmektedir.

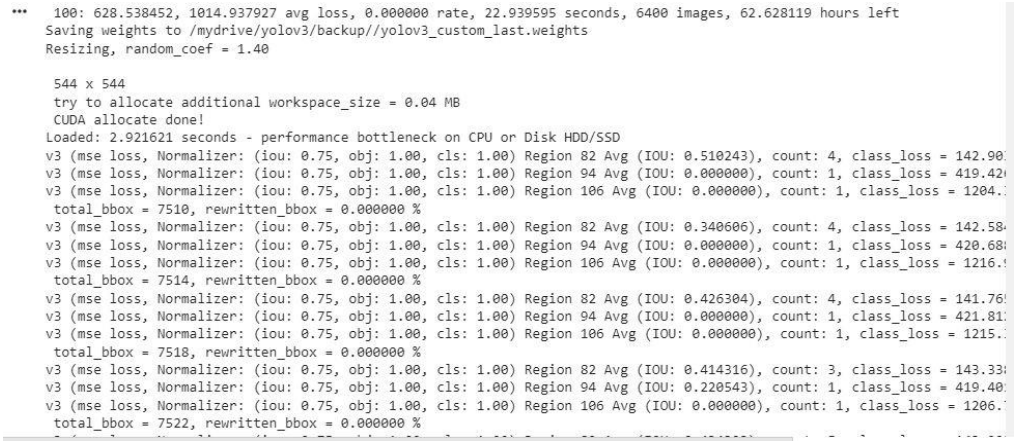
5.1. YOLOv3 Modeli Oluşturma ve Eğitim

YOLOv3 eğitimi başlarken ilk görüntü Şekil 5.1'de gösterilmiştir.



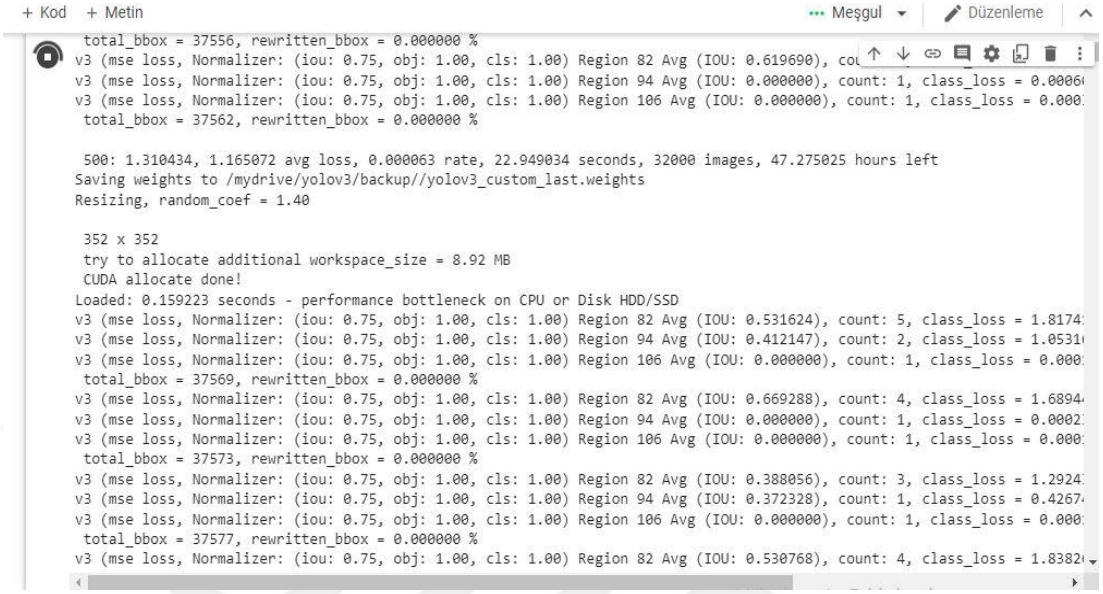
Şekil 5.1. Eğitimin başlangıcı

Eğitimin 100. adım değerleri Şekil 5.2'de gösterilmiştir. Burada 100. adımda ortalama kayıp değeri 628.5 ile 1014.9 arasındadır.



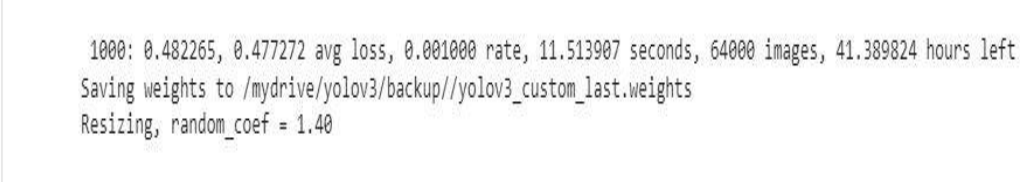
Şekil 5.2. Eğitimin 100. adımı

Eğitimin 500. Adım değerleri ise Şekil. 5.3'te verilmiştir. Burada 500. adımda ortalama kayıp değeri 1.31 ile 1.16 arasındadır.



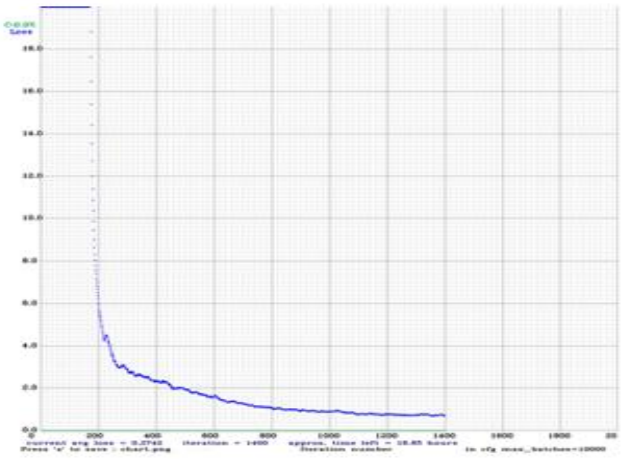
Şekil 5.3. Eğitimin 500. adımı

Eğitimin 1000. adım değerleri ise Şekil 5.4.'te verilmiştir. Burada 1000. adımda ortalama kayıp değeri 0.48 ile 0.47 arasındadır.



Şekil 5.4. Eğitimin 1000. adım

Eğitimin yaklaşık 5 saat sürekli devam etmesi halinde 1400 iterasyon gerçekleştirilmiştir. Şekil 5.5'te gösterilen grafikten anlaşılacağı üzere eğitimin ilk adımlarında hata oranı 18'den 0,27 değerine doğru bir düşüş yaşandığı gözlemlenmiştir. Bu verilere göre bu çalışmada performansta bir iyileşme görüldüğü söylenebilmektedir. Eğitim sonunda uygun ağırlıklar belirlenmiştir.



Şekil 5.5. Eğitim hata değeri grafiği

Eğitim tamamlandıktan sonra elde edilen model verileri kullanılarak nesne tespiti gerçekleşmiştir. Şekil 5.6'da modelin cami nesnesi tahmin çerçevesi ve tahmin oranı gösterilmektedir.

Şekil 5.6. Cami nesnesi tahmini

Şekil 5.7'de ise hastane nesnesi olan bir imge üzerinde bu nesnenin tahmini kare içerisine alınıp 0.32 olarak hesaplanan tahmin oranı belirtilmektedir.

Şekil 5.7. Hastane nesnesi tahmini

Bir başka tahmin işlemi saha nesnesi bulunan imge üzerinde %41 tahmin oranıyla Şekil 5.8'de gösterilmektedir.

Şekil 5.8. Saha nesnesi tahmini

İki saha nesnesi olan Şekil 5.9'daki görüntüde her iki nesnenin de ayrı ayrı kutulara alınarak farklı doğruluk oranlarıyla gerçekleşen tespit işlemi gösterilmektedir.

Şekil 5.9. Bir görüntüde iki nesne tahmini

Şekil 5.10'da cami içeren bir görüntüdeki cami nesnesi tespiti 0.40 oranında gerçekleştiği belirtilmektedir.

Şekil 5.10. Cami nesnesi tahmini

5.2. YOLOv5 Modeli Oluşturma ve Eğitim

Çalışma devam ederken YOLO sürümlerinin güncellenmesi ve geliştirilmesi sayesinde daha yüksek performans ölçebilmek için YOLOv5 eğitim modeli kullanılmıştır. Daha önce YOLOv3 modelinde olduğu gibi 300 imgeden 240 tanesi eğitim yapılması için 60 tanesi ise test işlemi için kullanılmıştır.

Sadece nesnenin tanınması üzerine yapılan bir araştırmada ise eğitimin sonuçları Şekil 5.11'de gösterilmiştir. İstenilen nesneye herhangi bir karışıklık olmadan doğru bir şekilde ulaşılabilmektedir.

Şekil 5.11. YOLOv5 nesne tahmini sonuçları-1

YOLOv5 modeli ile nesne tanımadaki doğruluk oranları %90'a varan performansa ulaşılabilmektedir. Görüntü verilerinin eğitim sonuçları yaklaşık 1 saat süren eğitim ile şekil 5.12'de ve şekil 5.13' teki sonuçların alınması sağlanmıştır.

Şekil 5.12. YOLOv5 nesne tahmini sonuçları-2

Şekil 5.13. YOLOv5 nesne tahmini sonuçları-3

Görüldüğü gibi yüzde 90'a varan tahmin değerlerinde tahmin edilen nesne özellik çıkarım algoritması ile kare içine alınmıştır. Kuşbakışı bu görüntülerin bulut, toz veya fotoğraf kalitesi

düşüklüğü nedeniyle nesne tespitinin yapılması zorlaşır iken elde edilen değerler tatmin edici ölçekte ve bir sonraki çalışmaya ışık tutacak niteliktedir.

5.2.1. Başarı Ölçüm Değerleri

Derin öğrenme sistemlerinde modelin eğitim süreci sonunda elde ettiği başarı ve performansı belli ölçütlere göre değerlendirilmektedir. Bu başlıkta çalışmada kullanılan modelin değerlendirilmesi için doğruluk (accuracy), kesinlik (precision), duyarlılık (recall) ve mAP metrik hesaplamalarına değinilmiştir. Metriklerin hesaplanması için aşağıda verilecek formüllerde kullanılan Doğru Pozitif (DP) ve Doğru Negatif (DN) , modelin doğru olarak tahmin edildiği, Yanlış Pozitif (YP) ve Yanlış Negatif (YN) ise modelin yanlış olarak tahmin edildiği alanlardır.

Doğruluk (accuracy) değeri modelde doğru tahmin ettiğimiz alanların toplam veri kümesine oranı ile hesaplanmaktır. Denklem 5.1. ‘de doğruluk formülü verilmiştir.

$$Doğruluk = \frac{\mathrm{DP+DN}}{DP+YP+DN+YN} \tag{5.1}$$

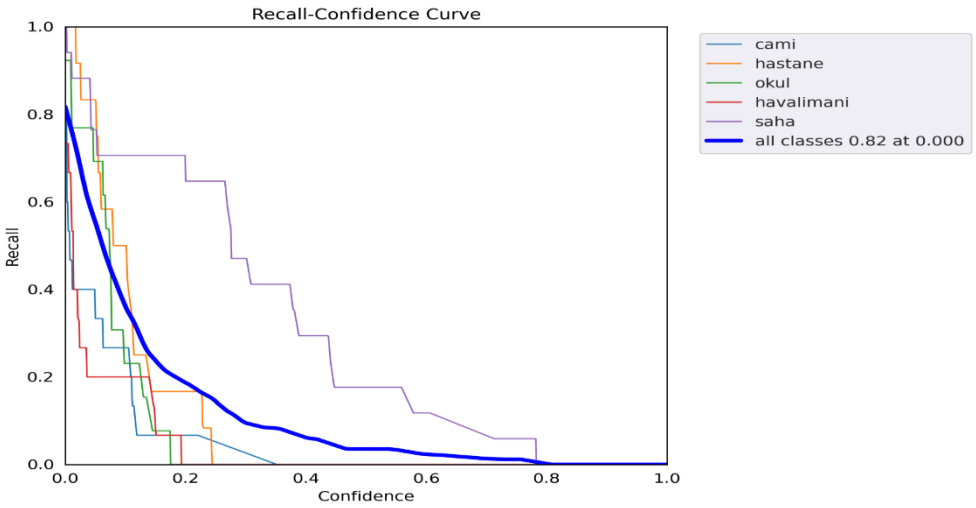
Kesinlik (Precision) ise Pozitif olarak tahmin edilen değerlerin gerçekten kaç adedinin Pozitif olduğunu göstermektedir. Denklem 5.2. ‘de kesinlik formülü verilmiştir.

$$Kesinlik = \frac{\mathrm{DP}}{DP+YP} \tag{5.2}$$

Duyarlılık (Recall) ise Pozitif olarak tahmin etmemiz gereken işlemlerin ne kadarını Pozitif olarak tahmin ettiğimizi gösteren bir metriktir. Mümkün olduğunca yüksek olması gereklidir. Denklem 5.3. ‘te duyarlılık hesaplama formülü verilmiştir.

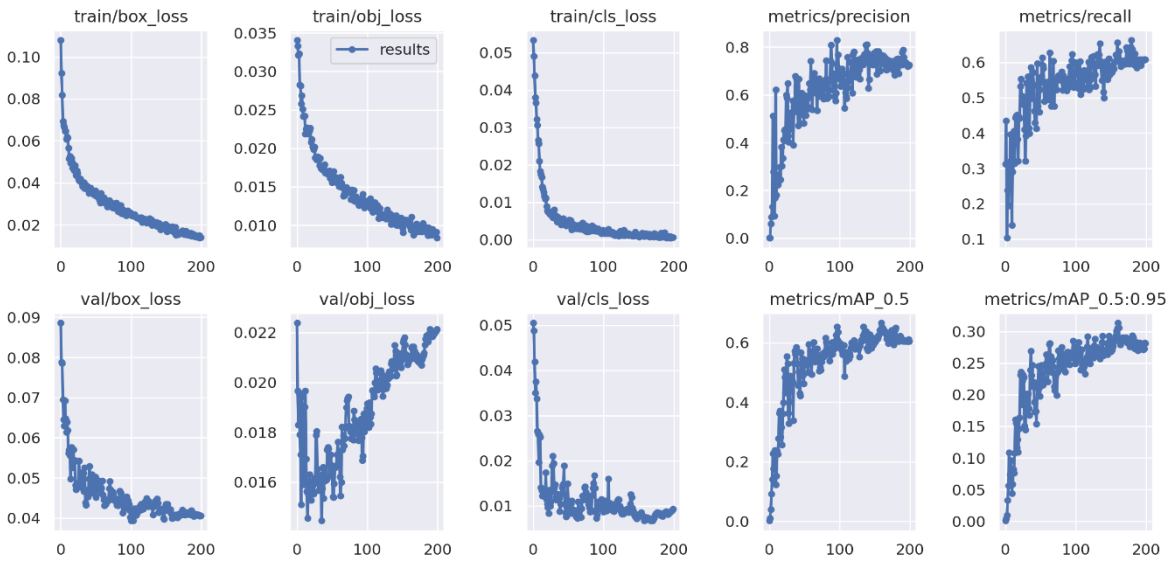
$$Duyarlılık = \frac{\mathrm{DP}}{DP+YN} \tag{5.3}$$

Yapılan çalışmada cami, hastane, okul, havalimanı ve saha sınıflarının duyarlılık değerlerinin grafiksel gösterimi Şekil 5.14’te gösterilmektedir.



Şekil.5.14. Tüm sınıfların duyarlılık ölçümü

Bu eğitim sonucunda 200 adım sayısı ile elde edilen kayıp değerlerinin ve genel ortalama kesinlik (mAP) değerlerinin ve duyarlılık değerlerinin grafikleri Şekil 5.15'te yer almaktadır. Burada adım sayısı arttıkça performansın yükseldiği gözlenmektedir. İlk adımlarda öğrenme daha fazla olmaktadır.



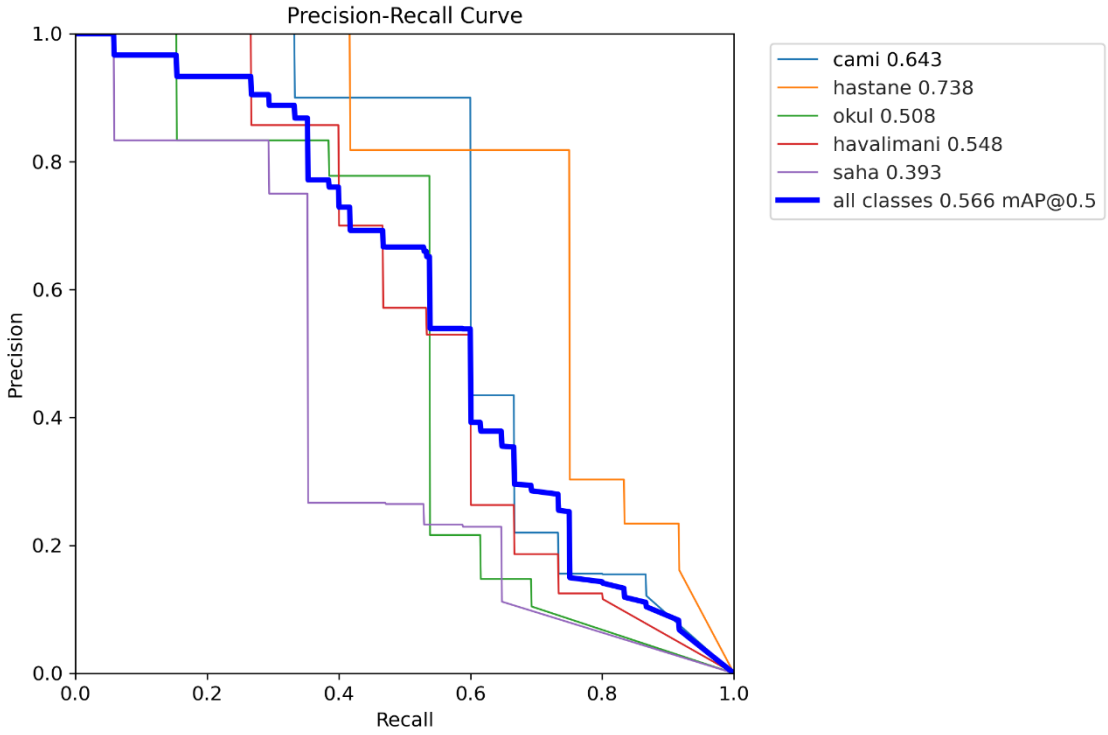
Şekil.5.15. YOLOv5 eğitimi sonucu elde edilen ölçümler

Test verisindeki her bir örnek için bir test çalıştırılır ve elde edilen sonuca göre recall(r) ve precision(p) değerleri yüzdelik olarak hesaplanır. Bu değerlere göre bir r-p grafiği oluşur. Oluşturulan r-p grafiği altında kalan alan Average Precision (Ortalama hassasiyet - AP) olarak

hesaplanmaktadır. Ortalama Hassasiyet (Average Precision) formülü Denklem 5.4.'te gösterilmektedir.

$$AP = \int_0^1 p(r)dr \tag{5.4}$$

Çalışmanın konusu 5 sınıf için r-p grafiği Şekil 5.16'da gösterilmektedir.



Şekil.5.16. Tüm sınıflar için kesinlik- duyarlılık grafiği

Her bir sınıf için AP hesabı yapıldıktan sonra tüm sınıflar için N kadar sınıf sayısına bölünerek Genel Ortalama Hassasiyet (Mean Average Precision) değeri Denklem 5.5' teki gibi hesaplanır[37].

$$mAP = \frac{1}{\mathrm{N}} \sum_{i=1}^{N} APi \tag{5.5}$$

Bu hesaplamalara göre çalışmadaki modelin 200 adım sonrası her sınıfı için ayrı ortalama değerleri mevcuttur. Cami, hastane, okul, havalimanı ve saha sınıflarının genel ortalama değerleri Tablo 5.1'de gösterilmektedir.

Tablo 5.1. Her sınıf için mAP değerleri

Sınıf	**İmge Sayısı**	**Durum Sayısı**	**mAP (%)**
cami	60	15	64.3
hastane	60	12	73.8
okul	60	13	50.8
havalimanı	60	15	54.8
saha	60	17	39.3
genel ortalama	60	72	56.6

Her bir adımda ölçümler performansı olumlu yönde etkilemiştir. YOLOv5 modelinin 1. , 100. , 150. ve 200. adımları Tablo 5.2'de gösterilmektedir.

Tablo 5.2. Adımlara göre metrikler

Epoch	Train/box_loss	Precision	Recall	mAP
1	0.10541	0.0023214	0.57033	0.0029404
100	0.013255	0.6406	0.47318	0.24217
150	0.018450	0.69308	0.55107	0.27392
200	0.0096626	0.76149	0.45245	0.26546

Yapılan bu çalışmalarda algoritmanın üst sürümü ile daha yüksek performans alındığı gözlemlenmektedir. YOLOv5'in YOLOv3'e üstünlükleri Tablo 5.3'te gösterilmektedir. YOLOv5, YOLOv3'ün 1/4 süresiyle ve 1/7 kadarlık adım sayısı giderek performansını % 46 oranında iyileştirmiştir. Doğruluk oranı arttıkça tahmin edilebilirlik artmış, zamandan ve işlem yoğunluğundan kazanım sağlanmıştır.

Tablo 5.3. YOLOv3 ve YOLOv5'in performans karşılaştırması

Performans/Algoritma	Eğitim Süresi	Adım Sayısı (epoch)	Doğruluk (Accuracy) Oranı (min/ max)
YOLOv3	1 saatte	14000	0.32 – 0.44
YOLOv5	4 saatte	200	0.40 - 0.90

Derin öğrenme algoritmaları günümüzde nesne tespiti için kullanılan algoritmaların başında gelmektedir. Bu kullanım yaygınlığının sebepleri başarı oranının iyi durumda olmasıdır. Hazırlanılan model ile cami, hastane, okul, havalimanı ve saha gibi tespit edilirliği daha belirgin nesnelerin yapay sinir ağları sayesinde rakamsal değerler ifade edilebilen tespitlerinin geliştirilmesi bu alanda yapılan çalışmalara büyük ölçekte katkı sağlamıştır.

6. SONUÇLAR

Bu çalışmada, nesne tespit etme alanında kullanılan YOLOv3 ve YOLOv5 modelleri ile halka açık binaların tespiti yapılmıştır. YOLOv3 algoritması ve YOLOv5 algoritması performans karşılaştırması yapılıp YOLOv5'in daha iyi performansla öğrenimi gerçekleştirmesi sağlanmıştır.

Bu tez çalışması ile belirli halka açık binaların tespiti yapılabilmekte ve tespit edilen nesnenin görüntü içerisinde hangi konumda olduğu bilgisine erişilebilmektedir. YOLO modeli ile nesneler farklı boyutlardan ve açılardan çekilmiş görüntüleriyle tespitinin performansına olumlu etki etmesi amaçlanmıştır. Deneysel çalışmalar sonucunda gerçek zamanlı nesne tanımı ve takibi yapabilen YOLO algoritması çalışmanın ilk aşamalarında kullanılan tensorflow kütüphanesini geride bırakmıştır. Anaconda komut dizininde gerçekleştirilen Python tabanlı tensorflow kütüphanesi sürüm uyumsuzlukları ve donanımsal yetersizlikleri tolore edememe sebebiyle geliştirilmeye devam etmesi gerektiği yapılan uygulama ile tespit edilmiştir. YOLO versiyonlarının diğer nesne tanıma modellerinden çok daha iyi sonuç verdiği görülmektedir. Sanal makine ortamı kullanılarak imgelerin nesne tanıma hızı ve performansı arttırılmıştır. Eğitim işlemi adımından sonra imgelerin doğru sınıflandırılmalarındaki oranı gözlemleyebilmek için işlem adım sayısı artırılmış ve kayıp değeri 1'in altına düşünceye dek eğitim sürdürülmüştür. Bu şekilde yüksek performans değerlerine sahip yeni bir model ortaya konmuştur.

Bu çalışmada, ekobilişim alanında oluşturulan yeni veri seti gelecek çalışmalara ışık tutar nitelikte katkı sağlamaktadır. Bu çalışmadan yola çıkarak başka halka açık alanlarda denenmesi ve daha iyi sonuçlar alınması planlanmaktadır. Bu çalışmada, derin öğrenme yöntemleri halka açık binaların tespit edilmesi alanında kullanılmıştır. Kullanılan veri tespit etme algoritmaları en uygun şartlarda çalışmaya uygun hale getirilerek tespit etme doğruluğunun arttırılması ve çalışma sonucunda elde edilen evrişimsel sinir ağı ağırlıklarının farklı platformlarda kullanılarak belediyelerde ve farklı çeşitli kuruluşlarda kullanılabilen akıllı uygulamalar geliştirilmesi planlanmaktadır. Gerçek zamanlı olarak tespit edilen binalara ait konum bilgileri elde edilebilmektedir. Bu konum bilgilerine göre farklı medya bileşenlerini de ihtiva eden yardımcı teknolojiler yapılması planlanmaktadır. Gelecek çalışmalarda konut, çeşme, karakol, benzinlik gibi birçok navigasyon bilgisi gereken nesneler hakkında bilgilendirme yapılması planlanmaktadır. Yapılması planlanan uygulamalar sayesinde ilgili kurumlar çalışmalarında yapay zekâ modüllerini daha etkin ve pratik bir şekilde kullanabileceklerdir.

KAYNAKLAR

[1] Karakuş, Serkan, et al. Derin Öğrenme Yöntemlerinin Kullanılarak Dijital Deliller Üzerinde Adli Bilişim İncelemesi. 2018.

[2] Doğan, Ferdi, and İbrahim Türkoğlu. "Derin Öğrenme Modelleri ve Uygulama Alanlarına İlişkin Bir Derleme." DÜMF Mühendislik Dergisi 10.2: 409-445.

[3] Kizrak, M. A. & Bolat, B. (2018). Derin Öğrenme İle Kalabalık Analizi Üzerine Detaylı Bir Araştırma. Bilişim Teknolojileri Dergisi, *11*(3), 263-286.

[4] Özkan, İnik., & Ülker, E. (2017). Derin öğrenme ve görüntü analizinde kullanılan derin öğrenme modelleri. Gaziosmanpaşa Bilimsel Araştırma Dergisi, 6(3), 85-104.

[5] Zhong, Y., Hu, X., Luo, C., Wang, X., Zhao, J., & Zhang, L. (2020). WHU-Hi: UAV-borne hyperspectral with high spatial resolution (H2) benchmark datasets and classifier for precise crop identification based on deep convolutional neural network with CRF. Remote Sensing of Environment, 250, 112012

[6] Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 652-660).

[7] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE transactions on pattern analysis and machine intelligence, 37(9), 1904-1916.

[8] Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2117-2125

[9] Liu, M., Wang, X., Zhou, A., Fu, X., Ma, Y., Piao, C., 2020. UAV-YOLO: Small object detection on unmanned aerial vehicle perspective. Sensors 20, 1–12. doi:10.3390/s20082238.

[10] Çalışkan, D., & Demir, Ö. Derin Öğrenme Yöntemleri ile Şüpheli Davraniş Tespiti. International Periodical of Recent Technologies in Applied Engineering, 3(1), 26-41.

[11] Song, Q., Li, S., Bai, Q., Yang, J., Zhang, X., Li, Z., & Duan, Z. (2021). Object Detection Method for Grasping Robot Based on Improved YOLOv5. Micromachines, 12(11), 1273.

[12] Şeker, Abdulkadir; Diri, Banu; Balık, Hasan Hüseyin. Derin Öğrenme Yöntemleri Ve Uygulamaları Hakkında Bir İnceleme. Gazi Mühendislik Bilimleri Dergisi (GMBD), 2017, 3.3: 47-64.

[13] Pehlivan, R. (2014). *Resim tabanlı osmanlıca belgelerde sınıflandırma* (Doctoral dissertation, İstanbul Kültür Üniversitesi/Fen Bilimleri Enstitüsü/Matematik Bilgisayar Anabilim Dalı).

[14] Öçer, N. E. (2020). Uzaktan algılama görüntülerinde derin öğrenme temelli yaklaşımlar kullanarak nesne tespiti.

[15] Dıkbayır, H. S., & Bülbül, H. İ. Derin Öğrenme Yöntemleri Kullanarak Gerçek Zamanlı Araç Tespiti. Tübav Bilim Dergisi, 13(3), 1-14.

[16] Karacı, A. (2022). X-ışını görüntülerinden omuz implantlarının tespiti ve sınıflandırılması: YOLO ve önceden eğitilmiş evrişimsel sinir ağı tabanlı bir yaklaşım. *Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, *37*(1), 283-294.

[17] Huang, Y. Q., Zheng, J. C., Sun, S. D., Yang, C. F., & Liu, J. (2020). Optimized YOLOv3 algorithm and its application in traffic flow detections. *Applied Sciences*, *10*(9), 3079.

[18] Kuznetsova, A., Maleva, T., & Soloviev, V. (2020). Using YOLOv3 algorithm with pre-and post-processing for apple detection in fruit-harvesting robot. *Agronomy*, *10*(7), 1016.

[19] Kavzoğlu, T., & Çölkesen, İ. (2010). Destek vektör makineleri ile uydu görüntülerinin sınıflandırılmasında kernel fonksiyonlarının etkilerinin incelenmesi. Harita Dergisi, 144(7), 73-82.

[20] Chan, Tsung-Han, et al. PCANet: A simple deep learning baseline for image classification?. IEEE transactions on image processing, 2015, 24.12: 5017-5032.

[21] Toraman, Suat. "Derin Öğrenme ile İnsansız Hava Aracı Görüntülerinden Yaya Tespiti." Journal of Aviation 2.2: 64-69.

[22] Kayaalp, K.; Süzen, A. A. Derin Öğrenme ve Türkiye'deki Uygulamaları. Yayın Yeri: IKSAD International Publishing House, Basım sayısı, 2018, 1.

[23] Resul, DAŞ, Polat, B., & Tuna, G. (2019). Derin öğrenme ile resim ve videolarda nesnelerin tanınması ve takibi. Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 31(2), 571-581.

[24] Hanbay, Kazım. Evrişimsel sinir ağı ve iki-boyutlu karmaşık gabor dönüşümü kullanılarak hiperspektral görüntü sınıflandırma. Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, 2020, 35.1: 443-456.

[25] Eser, SERT, (2021). A deep learning based approach for the detection of diseases in pepper and potato leaves. Anadolu Tarım Bilimleri Dergisi, 36(2), 167-178.

[26] Henderson, P., & Ferrari, V. (2016, November). End-to-end training of object class detectors for mean average precision. In Asian Conference on Computer Vision (pp. 198-213). Springer, Cham.

[27] Cesur, Muhammed, Cesur, Elif, Cedimoğlu, İ., Torkul, O., & Okuyan, A. (2018). Product Identification System Design Based on Deep Learning. International Journal of Scientific and Technological Research, 4(10).

[28] Ortaç, G., & Özcan, G. (2018). A Comparative Study for Hyperspectral Data Classification with Deep Learning and Dimensionality Reduction Techniques. Uludağ Üniversitesi Mühendislik Fakültesi Dergisi, 23(3), 73-90.

[29] Süberk, Nilay Tuğçe; Ateş, Hasan Fehmi. Deep Learning for Building Density Estimation in Remotely Sensed Imagery. In: 2019 4th International Conference on Computer Science and Engineering (UBMK). IEEE, 2019. p. 423-428.

[30] Koç, Ebubekir, et al. Yapay Sinir Ağları, Kelime Vektörleri ve Derin Öğrenme Uygulamaları. 2018.

[31] Alan, Mehmet. Karar ağaçlarıyla öğrenci verilerinin sınıflandırılması. Atatürk Üniversitesi İktisadi ve İdari Bilimler Dergisi, 2014, 28.4.

[32] Adarsh, P., Rathi, P., & Kumar, M. (2020, March). YOLO v3-Tiny: Object Detection and Recognition using one stage improved model. In 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS) (pp. 687-694). IEEE.

[33] Aktaş, A., Doğan, B., & Demir, Ö. (2020). Derin öğrenme yöntemleri ile dokunsal parke yüzeyi tespiti. *Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, *35*(3), 1685-1700

[34] Özel, M. A., Baysal, S. S., & Şahin, M. Derin Öğrenme Algoritması (YOLO) ile Dinamik Test Süresince Süspansiyon Parçalarında Çatlak Tespiti. *Avrupa Bilim ve Teknoloji Dergisi*, (26), 1-5.

[35] Li, K., Huang, Z., Cheng, Y. C., & Lee, C. H. (2014, May). A maximal figure-of-merit learning approach to maximizing mean average precision with deep neural network based classifiers. In 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 4503-4507). IEEE.

[36] Meschede, C., Gros, D., Habermann, T., Kirstein, G., Ruhrberg, S. D., Schmidt, A., & Siebenlist, T. (2018). Anaphora Resolution: Analysing the Impact on Mean Average Precision

and Detecting Limitations of Automated Approaches. International Journal of Information Retrieval Research, 8(3), 33-45.

[37] Prasanna, S., & El-Sharkawy, M. (2023). Improving Mean Average Precision (mAP) of Camera and Radar Fusion Network for Object Detection Using Radar Augmentation. In Proceedings of Seventh International Congress on Information and Communication Technology (pp. 51-60). Springer, Singapore.

[38] Mahamud, S. S., Ayve, K. N., Uddin, A. H., & Arif, A. S. M. (2023). Tomato Leaf Disease Recognition with Deep Transfer Learning. In Emerging Technologies in Data Mining and Information Security (pp. 203-211). Springer, Singapore.

[39] Cengiz, A., & Derya, A. V. C. I. (2021). Uydu İmgelerine Derin Öğrenme Tabanlı Süper Çözünürlük Yöntemlerinin Uygulanması. Afyon Kocatepe Üniversitesi Fen Ve Mühendislik Bilimleri Dergisi, 21(5), 1069-1077.

[40] Jiang, C., Ren, H., Ye, X., Zhu, J., Zeng, H., Nan, Y., ... & Huo, H. (2022). Object detection from UAV thermal infrared images and videos using YOLO models. *International Journal of Applied Earth Observation and Geoinformation*, *112*, 102912.

[41] Musaoğlu, N., Göksel, Ç., Kaya, Ş., Saroğlu, E., & Bektaş, F. (2005). İstanbul Anadolu Yakasi 2b Alanlarinin Uydu Görüntüleri İle Analizi. TMMOB Harita ve Kadastro Mühendisleri Odası, 10. Türkiye Harita Bilimsel ve Teknik Kurultayı.

[42] www.google.com/earth