MARMARA ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

DERİN ÖĞRENME YÖNTEMLERİ İLE MEDİKAL GÖRÜNTÜLERDE KANSERLİ DOKU TESPİTİ

MUSTAFA YANCI

YÜKSEK LİSANS TEZİ

Elektrik - Elektronik Mühendisliği Anabilim Dalı

Elektrik - Elektronik Mühendisliği Yüksek Lisans Programı (Türkçe)

DANIŞMAN

Dr. Öğr. Üyesi Önder DEMİR

EŞ-DANIŞMAN

Dr. Öğr. Üyesi Kazım YILDIZ

İSTANBUL, 2019

Tarih : 28.02.2020

TEZ TESLİM FORMU

Sayı : EK38F3A3C3

Konu : Tez Teslim

Enstitü : Fen Bilimleri Enstitüsü

Ana Bilim Dalı : Elektrik - Elektronik Mühendisliği

Bilim Dalı : Elektrik - Elektronik Mühendisliği Yüksek Lisans Programı(Türkçe)

Hazırlayan : MUSTAFA YANCI

Danışman : Dr. Öğr. Üyesi Önder Demir & Dr. Öğr. Üyesi Kazım Yıldız

Tezin Adı : Derin Öğrenme Yöntemleri ile Medikal Görüntülerde Kanserli Doku Tespiti

Kısıt Tarihi :

İMZA

Yukarıda bilgileri bulunan öğrencinin tezi elektronik ortamda teslim alınmış ve kütüphanelerimize herhangi bir borcu bulunmamaktadır.

Teslim Alan :

İmza :

MARMARA ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Marmara Üniversitesi Fen Bilimleri Enstitüsü Yüksek Lisans Öğrencisi Mustafa Yancı'nın "**Derin Öğrenme Yöntemleri ile Medikal Görüntülerde Kanserli Doku Tespiti**" başlıklı tez çalışması, 13 Ocak 2020 tarihinde savunulmuş ve jüri üyeleri tarafından başarılı bulunmuştur.

Jüri Üyeleri

Dr.Öğretim Üyesi Önder Demir (Danışman)
Marmara Üniversitesi .....(İMZA).....

Doç.Dr. Ahmet Emin Kuzucuoğlu (Üye)
Marmara Üniversitesi .....(İMZA).....

Prof.Dr. Serhat Özekes (Üye)
Üsküdar Üniversitesi .....(İMZA).....

ONAY

Marmara Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 12.02.2020 tarih ve 2020/06-02 sayılı kararı ile Mustafa Yancı'nın Elektrik-Elektronik Mühendisliği Anabilim Dalı Elektrik-Elektronik Mühendisliği Programında Yüksek Lisans derecesi alması onanmıştır.

Fen Bilimleri Enstitüsü Müdürü

Prof. Dr. Bülent Ekici

MARMARA ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

DERİN ÖĞRENME YÖNTEMLERİ İLE MEDİKAL GÖRÜNTÜLERDE KANSERLİ DOKU TESPİTİ

MUSTAFA YANCI

523115028

YÜKSEK LİSANS TEZİ

Elektrik - Elektronik Mühendisliği Anabilim Dalı

Elektrik - Elektronik Mühendisliği Yüksek Lisans Programı (Türkçe)

DANIŞMAN

Dr. Öğr. Üyesi Önder DEMİR

EŞ-DANIŞMAN

Dr. Öğr. Üyesi Kazım YILDIZ

İSTANBUL, 2019

ÖNSÖZ

Tez çalışmam boyunca benden bilgisini ve desteğini esirgemeyen hocalarım Dr. Önder Demir'e ve Dr. Öğr. Üyesi Kazım Yıldız'a, bana bu süreçte manevi desteğini esirgemeyen aileme ve arkadaşlarıma teşekkür ederim.

Haziran, 2019 **Mustafa YANCI**

İÇİNDEKİLER

ÖZET

DERİN ÖĞRENME YÖNTEMLERİ İLE MEDİKAL GÖRÜNTÜLERDE KANSERLİ DOKU TESPİTİ

Kanser, insanlarda sıkça görülen ve ölüm sebepleri arasında ikinci sırada olan bir hastalıktır. Ülkemizde de kanser vakalarında ciddi bir artış yaşanmaktadır.

Kanserin erken evrelerde teşhis edilmesinin tedavinin başarısını arttırmak için çok kritik olduğu bilinmektedir. Ancak günümüzde kanser vakalarının artışı testlere ayrılan inceleme süresinin kısalmasını zorunlu hale getirmekte ve en doğru teşhisi koyma şansının azalmasına sebep olmaktadır. Ülkemizde onkoloji alanındaki uzman doktor sayısının da yetersiz olması da incelemelerde yetersizliğe sebep olmaktadır.

Medikal görüntülemede ciddi gelişmeler yaşanmasının yanında, bu görüntülerin incelenmesinde de teknolojik gelişmelerden faydalanılmaktadır. Bu teknolojik gelişmelerden biri yapay zekanın günümüzdeki karşılığı olan derin öğrenmedir.

Bu çalışmada medikal görüntüler üzerinde derin öğrenme yöntemleri kullanılarak kanserli doku tespiti yapmak amaçlanmıştır. Çalışmada bir Konvolüsyonel Sinir Ağı mimarisi olan YOLO kullanılmıştır. Gerçek zamanlı bir nesne tespit aracı olan YOLO, doğru konfigürasyonlarla medikal görüntülerde kanserli hücreleri tespit etmeyi başarmıştır. Çalışmada kullanılan veri setindeki tümür hücre görüntülerinin tespitinde %70-75 arası başarı elde edilmiştir.

Haziran,2019 **Mustafa YANCI**

ABSTRACT

CANCERIOUS TISSUE DETECTION ON MEDICAL IMAGES BY USING DEEP LEARNING METHODS

Cancer is one of the most frequent disase and the second cause of death. The number of cancer cases is increasing in our country too.

It is known that early detection of cancer is critical to increase the success of treatment. But nowadays the increase in the number of cancer cases causes examination periods to be shorter and the chance of making the right diagnosis to be decreased. Additionaly, the insufficient number of expert doctors in oncology area in our country causes inadequacy in the examinations.

In addition to serious developments in medical imaging, technological developments are also utilized in the examination of these images. One of these technological developments is deep learning which is current equivalent of artificial intelligence.

In this study, it was aimed to detect cancerous tissue on medical images by using deep learning methods. YOLO, one of the Artificial Neural Network architectures is used in the study. A real-time object detection tool, YOLO succeeded detection of cancer cells in medical images with the correct configuration.

About 70 -75% success rate was achieved in the detection of tumor cell images in the data set used in the study.

July,2019 **Mustafa YANCI**

SEMBOLLER

$\sum$	:	Toplam
μ	:	Mikro

KISALTMALAR

AI	: Artificial Intelligence / Yapay Zeka
ML	: Machine Learning / Makine Öğrenmesi
DL	: Deep Learning / Derin Öğrenme
ABD	: Amerika Birleşik Devletleri
GB	: Gigabyte
Hz	: Hertz
mAP	: Mean Average Precision
IDC	: Invasive Ductal Carcinoma
DCIS	: Ductal Carcinoma in Situ
ANN	: Artificial Neural Network
CPU	: Central Processing Unit
GPU	: Graphical Processing Unit
TPU	: Tensor Processing Unit
MRI	: Magnetic Resonance Imaging
BreCaHAD	: Breast Cancer Histopathological Annotation and Diagnosis
RGB	: Red – Green – Blue
PNG	: Portable Network Graphics
TIF	: Tagged Image Format
JSON	: JavaScript Object Notation
XML	: eXtensible Markup Language
YOLO	: You Only Look Once

ŞEKİL LİSTESİ

SAYFA

TABLO LİSTESİ

SAYFA

1. GİRİŞ

Son yıllarda kanser vakalarında ciddi bir artış yaşanmaktadır. Bu artış tıbbi incelemelerde yetersizliğe sebebiyet vermektedir.

Sağlık bakanlığının son verilerine göre erkeklerde akciğer kanseri, kadınlarda ise meme kanseri en çok görülen kanser türleridir. Meme kanseri kadınlarda en yüksek sayıda ölüme sebep olan kanser türüdür. [1]

Birçok hastalıkta olduğu gibi hem diğer kanser türlerinde hem de meme kanserinde erken teşhiş çok önemlidir. Kanser teşhisinde hem gözle veya elle muayene hem de çeşitli test ve tetkik yöntemleri uygulanmaktadır.

Teknolojinin imkanlarından birçok alanda olduğu gibi sağlık alanında da faydalanılmaktadır. Kanserde tanı koymak için çeşitli test ve tahlil yöntemlerinin yanında tıbbi görüntüleme yöntemleri de kullanılmaktadır.

Kanser teşhis ve tedavi sürecinde hastalar birçok kez Röntgen, MR, CT, PET, Mamografi, Ultrasonografi gibi çeşitli testler yaptırmaktadırlar. Gelişen görüntüleme yöntemleri sayesinde teşhisi zor birçok detayı görüntülemek günümüzde mümkündür. Ancak hasta ve vaka sayısındaki artış aynı zamanda hasta verisinde artış demektir. Hasta sayısı ve hasta verisindeki ciddi artış, uzmanların verilere incelemeye ayıracakları zamanı maalesef azaltmaktadır. Dolayısıyla da görüntüleme teknolojilerindeki gelişim her zaman kesin sonuca ulaşmak anlamına gelememektedir.

Hasta verilerini incelemekte insan gözüne olan ihtiyacı azaltmak için bilgisayar gücünden faydalanma uzun zamandır tercih edilen bir yoldur. Sağlık alanında tıbbi görüntülerde hastalık tespitinde görüntü işleme teknikleri sıkça kullanılmaktadır.

Uzun zamandır popüler bir teknolojik anlayış olan yapay zeka kavramı son yıllarda hızlı bir şekilde gelişen bilgisayar donanımları sayesinde görüntü işlemede de kendine yer bulmuş ve sağlık dahil birçok alanda insanların işini kolaylaştırmanın yanında birçok yeniliğin de ortaya çıkmasını sağlamıştır.

Yapay zekanın günümüzdeki varyasyonu olan makine öğrenmesi ve derin öğrenme kavramları görüntü işlemede çoğunlukla belli nesneleri tespit etme ve tespit edilen nesneye göre karar verme amacıyla kullanılmaktadır. Bu amaçlardan biri de tıbbi görüntülerde hastalık tespit etmektir.

1.1. Kanser Nedir?

Kanser; büyüme hücrelerinin düzenli çalışmasından sorumlu genlerdeki anormal değişiklikler veya mutasyonların sonucu olarak ortaya çıkan, kontrolsüz, durmaksızın çoğalan, diğer dokulara da yayılan ve sağlıklı şekilde hayatına devam eden hücrelerdir. [2][3]

Trilyonlarca hücreden oluşan insan başta olmak üzere hayvansal sisteme sahip canlıların vücudunda sağlıklı her dokunun hücreleri kendi içinde ihtiyaç olduğunda(büyüme, onarma vs.) yeni hücreler üretmek için bölünür, eski hücreleri öldürür ve vücuttan uzaklaştırır. Böylelikle büyüme, yenilenme ve onarma gibi biyolojik fonksiyonlar yerine getirilmiş olur. Bu bölünme işlemi ve hücrenin diğer biyolojik faaliyetleri hücrenin çekirdeğindeki DNA adlı zincir molekülün kontrolünde gerçekleşir. Bu molekülde meydana gelecek hasarlar hücrenin biyolojik fonksiyonlarında bozulmalara yol açar. Bu bozuklukların en yaygınlarından biri kanser olarak karşımıza çıkmaktadır. Bozuk bir DNA'lı hücre kontrolsüz bir şekilde bölünme kararı alıp, etrafındaki sağlıklı hücrelerin fonksiyonlarından bağımsız şekilde kendi biyolojik faaliyetlerini sürdürmeye başlar.

Bağımsızlığını ilan eden bozuk DNA'lı hücre veya hücreler tümör olarak adlandırılırlar. Tümörler durmaksızın bölünmeye devam ettikleri gibi etrafındaki sağlıklı hücrelere de zarar verirler. Bu zarar dokusal boyutta fonksiyonu yerine getirememe olarak kendini göstermesinin yanında hastaya da acı verebilmekte, hastayı ciddi şekilde olumsuz etkileyebilmektedir.

Bulunduğu noktada büyümekle kalmayan bu kanserli dokular zamanla dokusal bütünlüklerini de koruyamazlar ve bozuk hücrelerden bazıları dokudan koparak kan ve lenf damarları yoluyla diğer dokulara taşınabilirler. Tümörden ayrılıp başka sağlıklı dokulara taşınan kanser hücreleri ulaştıkları noktalarda yeni kanserli dokular üremesine sebebiyet verebilmektedirler. Kanserli dokuların bu yayılabilmesi durumuna **metastaz(saçılma)** denilir.

1.1.1. Kanser Türleri

Kanserin birçok türü vardır. En çok görülen kanser türlerinden birkaçı şunlardır.

- Meme kanseri
- Akciğer kanseri
- Prostat kanseri
- Kolon ve rektal kanser
- Cilt kanseri
- Mesane kanseri
- Pankreas kanseri
- Tiroid kanseri
- Lenf kanseri

1.1.2. Kanser ile İlgili Bazı Sayısal Veriler

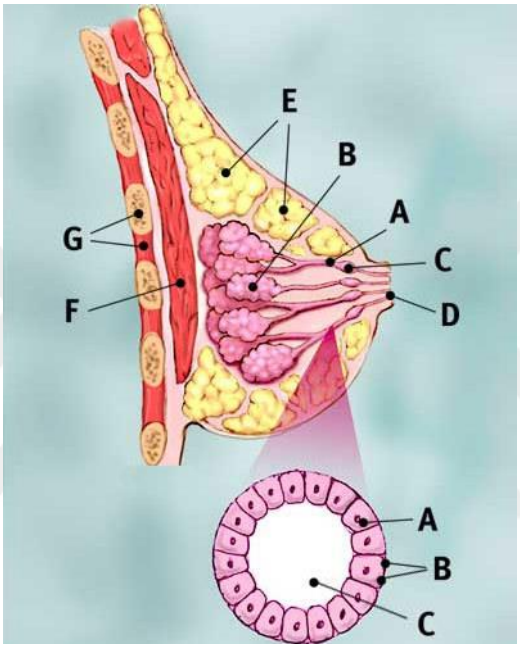
Dünya Sağlık Örgütü'nün (World Health Organization - WHO) kanser ile ilgili verilerinden bazıları aşağıdaki gibidir. [4]

- 2015 yılında yaşanan ölümlerin yaklaşık %17'si kanser kaynaklı olmuştur. 8,8 milyon insan kanserden yaşamını yitirmiştir.
- Kanser ölümlerinin %70'i düşük ve orta gelirli ülkelerde olmuştur.
- 2018 itibariyle dünya genelinde erkeklerde ölüme en çok sebep olan kanser türleri akciğer, karaciğer, kolorektal ve prostat kanserleridir.
- 2018 itibariyle dünya genelinde kadınlarda ölüme en çok sebep olan kanser türleri meme, akciğer, kolorektal, serviks ve mide kanserleridir.
- 2018 yılında kanserden 9.6 milyon insanın hayatını kaybedeceği tahmin edilmektedir.
- 2010 yılında kanserin ekonomik maliyeti yaklaşık 1,16 Trilyon dolar olmuştur (1 000 000 000 000 $)

1.2. Meme Kanseri

Meme kanseri meme hücrelerinde meydana gelen hücresel bozulma ve bozuk hücrelerin kontrolsüz bir şekilde çoğalmasıdır. Memedeki tümör direkt meme hücrelerindeki başkalaşım ile meydana geleceği gibi başka bir organdaki kanser hücrelerinin metastaz yapması sonucu da oluşabilir.

Şekil 1.1'de sağlıklı bir memenin anatomik yapısı gösterilmiştir.



Şekil 1. 1 Meme Anatomisi

Meme Profili (Üstteki şekil) : **A:** Tüpler(Süt kanalları – Ducts) **B:** Süt Bezleri (Lobules) **C:** Süt tutma kanalları **D:** Meme Başı (Nipple) **E:** Yağ dokuları **F:** Pektoral kas/Göğüs kası **G:** Göğüs duvarı ve göğüs kafesi/kaburgalar

Süt kanalının genişletilmiş profili (Alttaki şekil) A: Normal süt kanal hücreleri B: Kanal zarı C:Süt Kanalı(Lumen) [36]

1.2.1. Meme Kanseri ile İlgili Sayısal Veriler

- Meme kanseri tüm kanser türleri içinde %12.3'lük oranla en çok görülen ikinci kanser türüdür.
- 2018 Yılında yaklaşık 2.1 milyon kadın meme kanserine yakalanmıştır.

- Meme kanserinin en yaygın olduğu ilk on ülke sırasıyla Belçika, Lüksemburg, Hollanda, Fransa, Lübnan, Avusturya, İtalya, Yeni Zelanda'dır. [6]

- Meme kanserinde tedavi süreci uzun sürse de kurtulma oranı yüksektir. 2012 – 2016 yılları arasındaki verilere göre kurtulma oranı % 89.9'dur [7]

- 2019'da ABD'de yaklaşık 270 000 meme kanseri vakasının ortaya çıkması, yaklaşık 42 000 kişinin öleceği tahmin edilmektedir.

1.2.2. Meme Kanseri Türleri

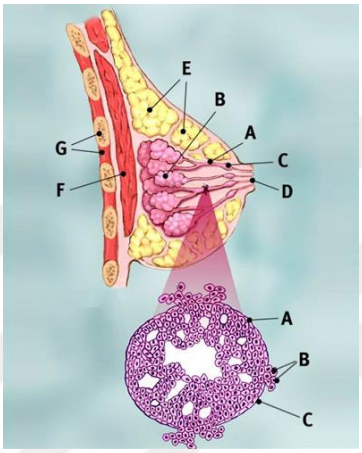
Meme kanserinin türleri aşağıdaki gibidir.

Ductal Carcinoma in Situ (DCIS) : İn Situ Duktal Karsinom olarak Türkçeleştirilmiş olan bu kanser türü süt kanallarında oluşur. **Ductal;** süt kanallarının içinde oluşan demektir. **Carcinoma** cilt veya diğer dokularda meydana gelen ve iç organlar üzerinde çizgi şeklinde veya organı kaplar şekilde çoğalan kanser türleri için kullanılan bir ifadedir. **Situ** ise kanserin bulunduğu organda oluşmaya başladığını belirten ifadedir. Yani kanserin başka bir organdan metastaz yaparak oluşmadığını ifade eder. Özetle DCIS memedeki süt kanallarında oluşmaya başlamış ve burada çoğalmış meme kanseri türüdür. DCIS aynı zamanda meme kanserinin sıfırıncı evresi olarak da bilinir.(Stage 0)

Invasive Ductal Carcinoma(IDC) : Invazif Duktal Karsinom olarak Türkçeleştirilen bu tür meme kanserlerinin %80'ini oluşturur. Saldıran, istila eden anlamlarına gelen **Invasive** kelimesi kanserin oluştuğu organda yayıldığını ifade eder. **Ductal,** süt kanallarının içinde oluşan demektir. **Carcinoma** cilt veya diğer dokularda meydana gelen ve iç organlar üzerinde çizgi veya kaplar şekilde çoğalan kanser türleri için kullanılan bir ifadedir. IDC süt kanallarını istila eden kanser türüdür. Bu kanser türü meme kanserinin birinci evresi olarak da bilinir.(Stage 1) DCIS zamanla IDC'ye dönüşebilir. Şekil 1.2'de IDC şematize edilmiştir. IDC'nin çeşitli tipleri vardır.

- **IDC Type - Tubular:** Memede süt kanallarında oluşan ve sağlıklı hücrelere yayılan kanser türüdür. Görülme oranı %1-2 civarındadır.
- **IDC Type – Medullary:** Medüler karsinom beynin medula tabakasına benzediği için bu isimle anılırlar. Beynin medula tabakası gibi yumuşak ve kanamaya meyillidir. IDC'ler içinde yaklaşık %4-5'lik orana sahiptir.

- **IDC Type – Mucinous:** Mucin denilen glikoproteinlerin içindeki anormal hücrelerden türeyen IDC türüdür. %1'lik bir orana sahiptir.
- **IDC Type – Papillary:** Menopoz sonrası yaşlı kadınlarda görülür. Normal sağlıklı meme hücreleri gibi görünmesine rağmen birinci evre kanser hücreleridir ve çok hızlı çoğalırlar. %1-2'lik görülme oranına sahiptirler.
- **IDC Type – Cribriform:** İsveç peyniri gibi delikli bir yapıya sahip olduğu için bu isim(cribriform – kalbur gibi) verilmiştir. Süt kanalları(ducts) ile süt bezleri(lobules) arasında oluşurlar. %5'lik bir görülme oranına sahiptirler.



Şekil 1. 2 Invasive Ductal Carcinoma (IDC)

Meme Profili (Üstteki şekil) : **A:** Tüpler(Süt kanalları – Ducts) **B:** Süt Bezleri (Lobules) **C:** Süt tutma kanalları **D:** Meme Başı (Nipple) **E:** Yağ dokuları **F:** Pektoral kas/Göğüs kası **G:** Göğüs duvarı ve göğüs kafesi/kaburgalar
Invasive Ductal Carcinoma (IDC) (Alttaki şekil) **A:** Normal süt kanal hücreleri **B:** Kanal zarını delip yayılmış kanser hücreleri **C:**Süt Kanalı Zarı [37]

***Invasive Lobular Carcinoma (ILC)*:** IDC'den sonra en çok görülen ikinci türdür. IDC süt kanallarında oluşurken ILC süt üretiminden sorumlu lobüllerde yani süt bezlerinde oluşurlar. Invasive olması yayılan yapıda olduklarını ifade eder. Meme kanserlerinin %10'lık bir kısmı ILC'dir.

***Inflammatory breast cancer (IBC)*:** İltihaplı meme kanseri, en ciddi kanser türlerinden biri olmakla birlikte %1'lik bir oranda görülür. Memede kızarma ve şişme olarak kendini gösterir. Çok hızlı yayılırlar günler hatta saatler içinde ilerleyebilir.

***Lobular Carcinoma In Situ (LCIS)*:** DCIS gibi bulunduğu organda oluşan ve henüz yayılmamış kanser türüdür. DCIS'den farkı, DCIS süt kanallarında oluşurken LCIS süt bezlerinde oluşur.

***Male Breast Cancer*:** Erkeklerde görülen meme kanseri türüdür. Erkeklerde meme kanseri 1/1000 görülme oranına sahiptir. 2019'da 2670 erkeğe meme teşhisi konulacağı tahmin edilmektedir.

***Meme Kanserinin Moleküler Alt Tipleri*:** Genetik sebeplere dayalı kanser türleridir. Luminal -A, Luminal - B, Triple-negative/basal-like, HER2-enriched, Normal-like şeklinde beş ana başlıkta incelenirler.

***Paget's Disease of the Nipple*:** Kanser hücreleri meme ucunda oluşmuş ve toplanmıştır. İlk başta meme ucundaki süt kanallarını etkiler, daha sonra meme başına yayılır. %4-5 görülme oranına sahiptir.

Phyllodes Tumors of the Breast: Phyllodes(filloid) kelimesi Yunanca'da "yaprağa benzeyen" anlamına gelmektedir. Bu kanser türünün bu kelime ile adlandırılmasının sebebi, tümörlerin yaprak şeklinde büyümesidir. Çok hızlı büyümelerine rağmen nadiren meme dışına yayılırlar. Görülme oranı %1'in altındadır.

Metastatic Breast Cancer: Kanserin dördüncü evresi (Stage IV) olarak da bilinen metastatik meme kanseri memede oluşup vücudun diğer organlarına yayılabilen tümörlerdir. Metstatik meme kanseri çoğunlukla karaciğer, beyin, kemikler ve akciğerlere metastaz yapar. Erken teşhis koyulan kadınların %30'u ilerleyen dönemlerde metastatik meme kanseri teşhisiyle de karşılaşmaktadırlar.

1.3. Yapay Zeka, Makine Öğrenmesi, Derin Öğrenme

Günümüzde veri analizlerinde kullanılan en popüler teknoloji trendlerinden biri derin öğrenme (deep learning - DL)dir. Derin öğrenme, makine öğrenmesi(machine learning - ML) ve yapay zeka (artificial intelligence - AI) kavramları her ne kadar aynı anlamlarda kullanılsa da farklı kavramlardır.

Yapay zekâ kavramı resmi olarak ilk defa 1955 yılında Honever, New Hampshire Dartmouth College'da yapılan bir konferansta John McCarthy tarafından kullanılmıştır[9].

Yapay zekâ, kendi kendine öğrenebilme, muhakeme yapabilme ve mantıklı kararlar verme gibi insana ait becerileri makinelere uyarlamayı amaçlayan bir bilim alanıdır. Yapay zekânın tarihsel gelişimine bakıldığında ise 1970 yılının bir dönüm noktası olduğu

görülmektedir. Bu tarihten önce birçok araştırmanın yapıldığı ve 1969 yılında XOR probleminin çözülememesi nedeni ile araştırmaların durduğu görülmektedir. 1970 yılından sonra sınırlı sayıda araştırmacının çalışmalarını sürdürmeleri ve XOR problemini çözmeleri sonucunda yapay sinir ağlarına olan ilgi yeniden alevlenmiştir. Bu çalışmalar hem yapay zekâ hem de donanım teknolojisindeki gelişmeler ile de desteklenmiştir. Artık bilgisayarların da öğrenebileceğini herkes kabul etmekte ve bu teknolojiden faydalanmak istemektedir[4]

Yapay zekadan derin öğrenme kavramına gelinen günümüzde, bu gelişimin bir kanadını gelişen teknoloji, diğer kanadını ise yapay sinir ağları oluşturmaktadır.

1.3.1. Yapay Sinir Ağı

Yapay sinir ağı, biyolojik sinir ağlarının fonksiyonalite ve yapısını taklit etmeyi amaçlayan matematiksel bir modeldir. Tüm yapay sinir ağlarının temel yapı taşı basit bir matematiksel model olan **yapay nöron**lardır. Yapay nöronlar ya da kısaca nöronlar **node (düğüm)** olarak da adlandırılırlar.

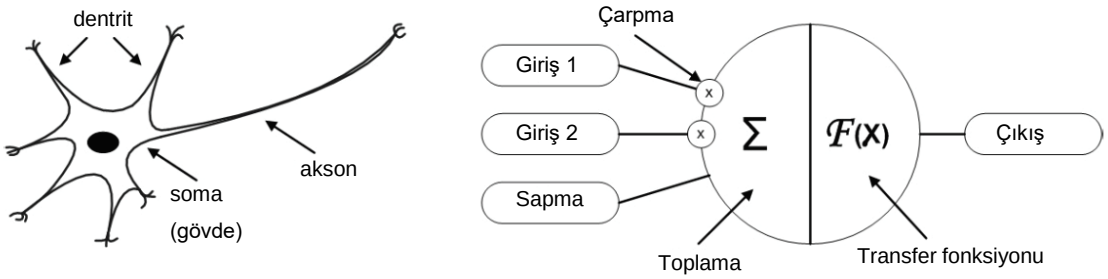
Bu matematiksel modelin çarpma, toplama ve aktivasyon olmak üzere 3 temel kural seti vardır. Bu kural setleri sırasıyla şu şeklîde çalışır;

***Çarpma*:** Yapay nöronun giriş kısmında tüm giriş değerleri ağırlıklandırılır yani her giriş değeri o giriş kanalına özel bir ağırlık katsayısı ile çarpılır.

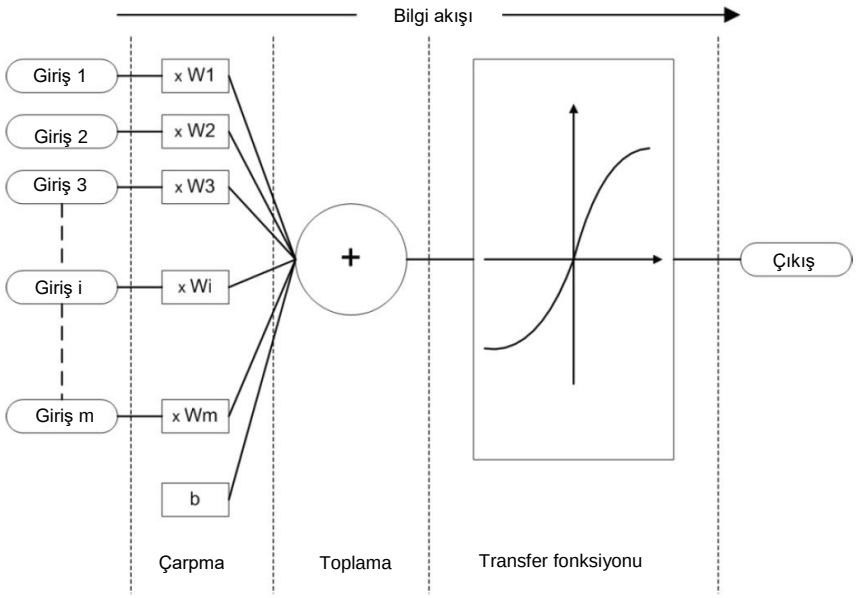
***Toplama*:** Yapay nöronun orta bölgesinde toplama fonksiyonu bulunur. Ağırlık katsayıları ile çarpılmış giriş değerleri ve sapma burada toplanır.

***Aktivasyon(Transfer) Fonksiyonu*:** Ağırlıklandırılmış girişler sapma ile toplandıktan sonra bu toplam, aktivasyon fonksiyonundan geçirilerek çıkışa iletilir. Aktivasyon fonksiyonu transfer fonksiyonu olarak da bilinir.

Özetle yapay nöron girişine uygulanan girdileri ağırlıklandırıp toplayıp gövdesindeki fonksiyondan geçirerek çıktı üreten, bir veya birden fazla girişi ve bir çıkışı olan bir fonksiyondur.



Şekil 1. 3 Biyolojik Nöron ve Yapay Nöron [38]



Şekil 1. 4 Yapay Nöronun Çalışma prensibi [38]

Yapay nöronun matematiksel tanımı şu şekildedir.

$$y(k) = F\left(\sum_{i=0}^{m} w_i(k)\, x_i(k) + b\right)$$

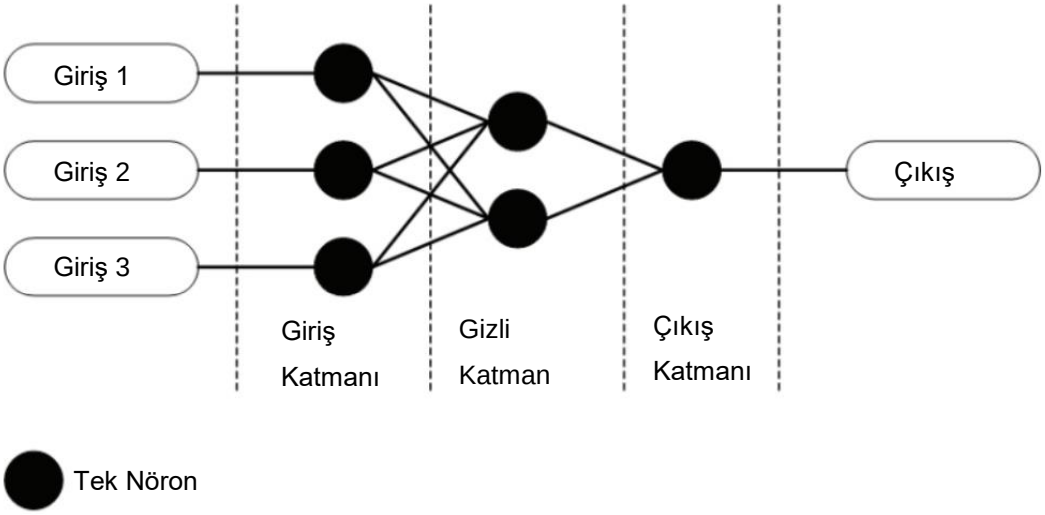
$x_i(k)$: i 0'dan m'ye giderken k ayrık zamanındaki giriş değeri
$w_i(k)$: i 0'dan m'ye giderken k ayrık zamanındaki ağırlık değeri
b : sapma
F : Transfer fonksiyonu
$y(k)$: k ayrık zamanındaki çıkış değeri

Transfer fonksiyonu, matematiksel modeli gerçek hayata benzetme amacıyla kullanılan çeşitli fonksiyonlardır. Aktivasyon fonksiyonu olarak da adlandırılırlar. Bunlardan bazıları basamak (step), doğrusal(linear), sigmoid, ReLU, Tanh fonksiyonlarıdır.

Örneğin; Basamak fonksiyonu giriş eşik değerden büyükse 1, eşik değerden küçükse 0 çıktısı üretir. Genelde eşik değer 0 seçilir ve 0 - 1 çıktısı almak beklenir. Aktivasyon fonksiyonu basamak olan bir nöronun ağırlıklı girişlerinin toplamı 0'dan küçükse 0, 0'dan büyükse 1 çıktısı üreten ikilik sistemde çalışan bir nöron tasarlanabilir.

Yapay nöron tek başına bir fonksiyondan ibarettir ancak birden fazla nöron çeşitli amaçlarla ve belli kurallara göre birbirine bağlandığında büyük bir hesaplama gücü ortaya çıkarabilmektedir. Yapay nöronların birbirlerine bağlanmasıyla elde edilen büyük ve kompleks matematiksel modellere Yapay Sinir Ağları (Artificial Neural Networks - ANN) denir.

Nöronlar birbirlerine rastgele bağlanmazlar. Bir nöronun çıkışı, başka bir nörona girdi olarak bağlanarak bir veya daha fazla fonksiyon zincirleri oluşturulur.



Şekil 1. 5 Basit Bir Yapay Sinir Ağı Örneği [38]

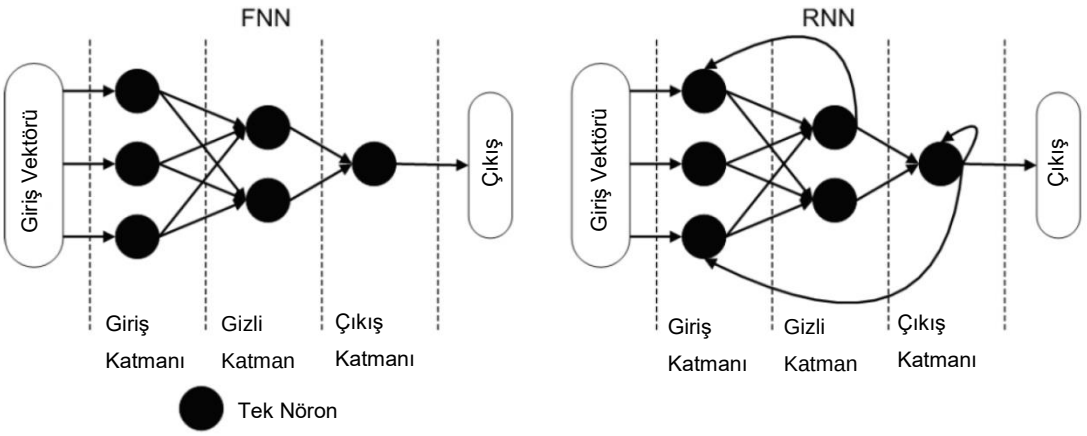
Yapay sinir ağları giriş katmanı, gizli katman veya katmanlar ve çıkış katmanından oluşan bir yapıya sahiptir. Şekil 1.5'de basit bir yapay sinir ağı modeli görülmektedir. Çeşitli topolojilere göre nöron dizilimi yapılmış bir yapay sinir ağı tek bir sistem olarak düşünüldüğünde giriş veya girişler ve çıkıştan ibaret bir sistemdir. Bu sistemde girdilerin uygulandığı nöronların oluşturduğu ilk katman giriş katmanı, sistemden çıkış alınan katman ise çıkış katmanı olarak adlandırılır. Giriş ve çıkış katmanları arasında ara işlemleri yapan nöronların bulunduğu katman veya katmanlar ise gizli katman olarak adlandırılır. Bir yapay sinir ağında gizli katman sayısı yapılacak işleme göre değişebilir.

Basit bir işlem için bir tane gizli katman yetebilirken kompleks bir işlem için onlarca hatta yüzlerce gizli katman kullanılabilir.

Biyolojik nöronla benzerlik kurmak gerekirse, nöronlar birer sinir hücresidir ve canlı vücudunda sinyal iletimini sağlarlar. Bir nöronun dentritine gelen sinyal nöronun gövdesinde işlenir ve akson denilen uçtan diğer bir nöronun dentritine aktarılır. Burada bir nöronun görevi kendisine uygulanan sinyali bir takım işlemlerden(örneğin genliğini artırmak gibi) başka bir nörona girdi olarak taşımaktır. Birden fazla nöron bir araya gelerek hem sinyal iletimini sağlayan, hem de sinyalin çeşidine göre algılamayı üstlenen bir sinir ağını oluşturmaktır. Örneğin; sesi kulağımızla algılarız ama kulaktaki reseptörlerle algılanan ses sinyali nöronlar üzerinde çeşitli işlemler uygulanarak elektrik sinyali şeklinde beyne iletilir ve yine nöronlardan oluşan beyin bu sinyali işler. Kulaktan beyne giden sinir ağında birçok katman çeşitli görevleri (sinyali güçlendirme, sinyalin türünü belirleme vs.) üstlenebilir. Sonuçta bu ağ ses sinyalini algılamayı amaçlamaktadır, giriş (kulaktaki alıcılar) ve çıkış (beyin) arasındaki fonksiyonlara gizli katman diyebiliriz.

Benzer bir örneği yapay sinir ağı için verirsek; bir resimdeki nesnenin şeklini ve rengini belirleyip ismini söyleyen bir yapay sinir ağı tasarladığımızı düşünelim. Bu ağın girdisi resimdir. Resmin uygulandığı girişler giriş katmanıdır. Ağımızın içinde resimdeki nesnenin rengini algılayan bir fonksiyon, şeklini algılayan başka bir fonksiyon(üçgen,kare,daire vs.) olduğunu düşünürsek, bu iki fonksiyon ağımızın gizli katmanlarını oluşturur. Daha sonra bu iki fonksiyondan alınan çıktılar, çıkış fonksiyonunda rengi ve ismi ile bir yazıya çevrilerek çıktı olarak (örn: kırmızı kare) üretilir. Bu dört katmanı olan(giriş katmanı, 2 tane gizli katman ve çıkış) bir yapay sinir ağıdır.

Değişik amaçlarla çeşitli yapay sinir ağı tasarlama topolojileri geliştirilmiştir. Convolutional Neural Network(CNN), Feed-Forward(ileri beslemeli) Neural Network (FNN) ve Recurrent(geri dönen) Neural Network(RNN) en çok kullanılan topolojilerden ikisidir. Şekil 1.6'da FNN ve RNN topolojilerine ait basit birer örnek verilmiştir.

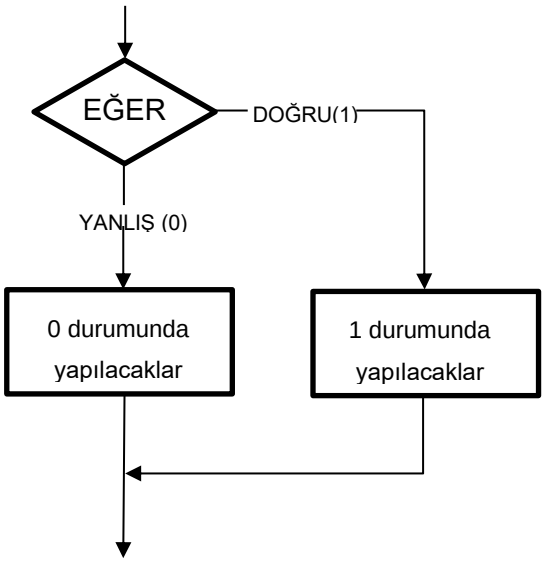


Şekil 1. 6 FNN ve RNN Topolojileri [38]

1.3.2. Yapay Sinir Ağından Yapay Zekaya

Yapay zeka herhangi bir canlı organizmadan faydalanmaksızın, tamamen yapay araçlar kullanılarak oluşturulan(çoğunlukla programlanan), canlı(genelde insan) gibi davranışlar sergileyebilen makinelerin(bilgisayarların) geliştirilmesi teknolojisinin genel adıdır.

Burada canlı(insan) gibi davranışlar sergilemekten kasıt, makinenin canlı gibi karar verebilmesidir. Programlamada bu amaçla kullanılan EĞER (if) yapısı bulunmaktadır. Bütün programlama dillerinde if yapısı vardır. Bu yapının temel çalışma şekli şekil 1.7'de görüldüğü gibidir.



Şekil 1. 7 EĞER (IF) diyagramı

Programlamada IF bloğu kontrol ve sonrasında karar verme amacıyla kullanılmaktadır. Kontrol ettiği koşulun doğru (True - 1) veya yanlış (false -0) sonuç vermesine göre programın akışını yönlendirir. Örneğin bir sayının mutlak değerini veren bir fonksiyon fonksiyonu bir IF bloğu ile yazabiliriz. Bunun için giriş sayısının 0'dan büyük veya eşit olup olmama koşulunu verebiliriz. Daha sonra koşulun çıktısına göre şöyle iki çıktı üretebiliriz.

- Eğer sayı 0'dan büyükse veya 0'a eşitse sayıyı aynı şekilde çıkar(hiç bir şey yapma)
- Eğer sayı 0'dan küçükse sayıyı -1 ile çarp

Aynı karar yapısını sayının 0'dan küçük olma koşuluna göre de kurabiliriz. Bu sefer kodumuz sayının sırıfdan küçük olmasını kontrol edecek.

- Eğer sayı 0'dan küçükse sayıyı -1 ile çarp
- Eğer sayı 0'dan küçük değilse aynı şekilde çıkar

EĞER yapısı yapay zeka kavramını tam olarak doyuramasa da makinede zeka kavramından bahsedebilmemizi sağlayan en önemli unsurdur. Burada içinde her EĞER bloğu bulunduran kodun yapay zekaya sahip olduğu iddia edilmemektedir. Ancak uzun yıllar boyunca yapay zekaya sahip sistemlerin gelişimi bu blok sayesinde gerçekleştirilmiştir.

Bir EĞER bloğunun çıktısı başka bir EĞER bloğunun girdisi olarak(tıpkı yapay sinir ağlarındaki gibi) kullanılırsa ve EĞER'lerden oluşan bir kontrol zinciri ile yapay zekaya sahip makineler üretilebilir. Ancak EĞER blokları ile makinenin neyi kontrol edeceğini ve kontrol sonucuna göre nasıl davranacağına aslında kodu yazan karar vermektedir.

Yapay zeka kavramının daha insansı bir noktaya gelmesi ve gelişmesi yapay sinir ağları sayesinde olmuştur. EĞER bloklarının içinde yapılan sayısal kontroller, gerçek hayattaki veriler üzerinden yapıldıkça makinenin yapay zekaya sahip olması fikri daha da gelişecektir.

Yapay zekadaki en önemli amaç, makinenin koşullara göre karar verme ve sınıflandırma yapabilmesidir. Örneğin; bir hayvanın bir at olup olmadığına karar veren minik bir yapay zekalı sistem geliştirilmek istenirse; dört ayaklı olması bir koşul olabilir. Ancak bir hayvanın dört ayaklı olması at olarak sınıflandırılması için yeterli değildir. Bacak

kemiklerinin oranı da bir kontrol koşulu olarak eklenebilir. Veya gözlerinin konumunun yüzüne oranı da bunlara eklenebilir. Bu durumda bir eşek de at olarak sınıflandırılabilir. Canlının boy verisi de kontrol edilebilir. Bu sefer bir zebra da at olarak sınıflandırılabilir. O zaman renk kontrolü de gerekebilir.

Görüleceği üzere makineden insana gibi bir karar verebilme yeteneği kazanması için ona kontrol etmesi gereken noktalar dışarıdan verilmektedir. Yani bu kontrollerin yapılacağı EĞER blokları geliştirici tarafından verilmektedir. .

Yapay nöron, bir EĞER bloğu gibi çalışsa da en önemli farkı ağırlıkları da dikkate almasıdır. Aslında yapay nöron bünyesinde EĞER bloğunu da barındıran ama aynı zamanda çeşitli hesaplamalar da yapan bir matematiksel fonksiyondur. Dolayısıyla EĞER bloğundan fazlasını sunmaktadır. Örneğin; cep telefonu, tablet bilgisayar, LCD-LED monitör ve televizyonu sınıflandıran bir sistem geliştirilmiş olsaydı ve bu sistem sadece EĞER blokları ile tasarlansaydı, kontrol koşullarında dört köşeli olma, diktörtgen olma gibi durumları kontrol ettirilseydi, bu sistem muhtemelen bir sınıflandırma yapamazdı. Ancak bu sistemi yapay nöronlarla tasarlarsak, çeşitli noktaları kontrol eden nöronların kodlarında hangi nesnenin hangi özelliklere sahip olduğunun kodlanması gerekecek ve daha sonra nesnenin detaylarının verilmesi yetecekti. Tasarlanan bu sistemde bir nöron bu veriler ışığında en boy oranını hesaplayacak ama aynı anda diğer detaylara da bakacaktır. Başka bir nöron ise nesnenin boyutlarının aralığını kontrol edecektir. Başka bir nöron nesnenin özelliklerine göre sınıflandırma yapacaktır. Günümüzde birçok ekran 16:9 oranında üretilmekte. Dolayısıyla ekran oranı yeterli bir hesaplama olmayacağı için ekran boyutu da dikkate alınacaktır. 16:9 oranına sahip, ekran boyutu 7 inç'ten büyük ise tablet bilgisayar, monitör veya televizyon olarak sınıflandırılabilecek şekilde tasarlanabilir. Bu sefer cihazın anten/uydu girişinin olup olmaması koşuluna bakılacaktır. Televizyonlar ağırlıklı olarak yani çoğunlukla bir anten/uydu girişine sahiptir. Dolayısıyla bu kontrolü yapan nöronda televizyon için bu noktanın ağırlığı yüksek olacaktır.

Özetle, makinenin yani bilgisayarın daha da zeki hale gelmesi Yapay Nöron kavramı ile olmuştur. Daha fazla nöron daha fazla kontrol demektir. Daha fazla kontrol de yapılan sınıflandırmanın doğruluk payının daha da artması demektir.

1.3.3. Makine Öğrenmesi – Machine Learning(ML)

Günümüzde yapay zeka ifadesi kullanım olarak popülerliğini sürdürse de çalışma prensibini yeni bir yaklaşıma bırakmıştır. Bu yaklaşım yapay zekanın bir alt dalı olan Makine Öğrenmesidir. 1970 yılından sonraki 10 yıl içerisinde birbirinden farklı yeni yapay zeka modelleri geliştirilmiştir. Makine öğrenmesi(machine learning) modeli bu modellerin içinde en çok ilgi görenlerinden biri olmuştur.

Önce makinenin bir zekaya sahip olabileceği fikri ile makinelere insan gibi sınıflandırma ve karar verebilme yeteneği kazandırılmıştır. Ancak makinenin neyi nasıl sınıflandıracağı, nasıl kontrol edip karar vereceğini insan kodlamaktaydı. Bu yeni yaklaşım ise makinenin öğrenip, öğrendikleri doğrultusunda kararlar verebilme yeteneğine sahip olmasını amaçlamaktadır.

Makine öğrenmesi veri madenciliği ile ortak noktalarda buluşmasına rağmen çalışma şekilleri ve kulanım amaçları bakımından farklılık göstermektedirler.

Veri madenciliği birçok alanda büyük verilerin analizleri ve istatistikleri yapılarak tahmin yürütme, pazarlama stratejilerini geliştirme amacıyla izlenilen, bilgisayar bilimi ve istatistiğin disiplinler arası bir alt dalıdır. Analizi yapılacak veri arttıkça daha çok ortak özellik ortaya çıkarılır ve veri ile ilgili daha çok bilgi çıkarmak mümkün olmaktadır.

Veri madenciliği pazarlama başta olmak üzere biyoloji, bankacılık, sigortacılık, borsa, perakendecilik, telekomünikasyon, genetik, sağlık, bilim ve mühendislik, kriminoloji, endüstri, istihbarat vb. birçok dalda kullanım alanına sahiptir. Birçok şirket yeni kampanya ve promosyonları tasarlamak, yeni ürün konseptlerini belirlemek için geçmişteki alışveriş verilerinden faydalanmaktadır. Bu amaçla müşterileri ile ilgili bir çok detayı veri tabanlarında saklamaktadırlar. Veri madenciliği uygulamalarına birkaç örnek vermek gerekirse;

- Bankalar geçmişte kredi almış müşterilerine ait yaş, meslek, medeni durum, ev-araba sahibi olup olmama, çocuk sayısı, çekilen kredi miktarı, geri ödeme miktarı, geri ödeme düzenliliği vs. gibi birçok detayı saklamaktadırlar. Bu bilgiler ışığında bazı müşteri profilleri çıkarılır. Kredi almak isteyen yeni müşterilerin detay bilgilerinin daha önce oluşturdukları profillere uyup uymadığına göre müşteriye olumlu veya olumsuz cevap vereceklerine karar verirler. Veya yine bu tip veriler ışığında yeni kredi kampanyalarını oluştururlar.

- Telekomünikasyon şirketleri müşterilerinin geçmişteki kullanım bilgilerini ticari amaçlarla saklarlar. Müşterinin yaş, cinsiyet, meslek, aylık Ses-SMS-Veri kullanım miktarı, paket ücreti gibi detaylara göre analizler yapıp yeni paketleri tasarlarlar. Örneğin; müşterilerin zamanla SMS paketlerini daha az kullandığı, internet paketlerini bitirdiği tespiti yapıldıysa, yeni kampanyalarda daha az SMS, daha çok ses ve internet kullanım hakkı veren paketler tasarlayabilirler.
- Pazarlama alanında, müşterilerin alışveriş detayları (fiş - fatura detayları) veri tabanlarında saklanmaktadır. Bir A ürününü alan müşterilerin kaçının hangi ürünü de beraberinde aldıkları tespit edilebilir. Buna göre yeni promosyonlar oluşturularak satışların artması amaçlanmaktadır.
- Sağlık alanında, hastalara ait çeşitli detaylar toplanarak, hangi koşullarda hangi hastalık riskinin daha çok olduğu gibi verilere ulaşılabilir. Yüksek riskli hastalıklara karşı çeşitli tedaviler geliştirmede bu analizlerden faydalanılabilir.

Makine öğrenmesi ya da makine öğrenimi temelde veri madenciliği ile yapay zekanın kesiştiği bir nokta olarak görülebilir. Veri madenciliği ve makine öğrenmesinde çoğunlukla aynı algoritmalar kullanılır ancak önemli bir fark vardır; veri madenciliğinde geçmişte toplanmış veriler ışığında bazı çıkarımlar yapılarak geleceğe dair çeşitli planlamalar yapılması amaçlanmakta iken, makine öğrenmesinde geçmişteki veriler makineye öğretilerek, gelecekteki yeni verileri eski veriler ışığında değerlendirmek, sınıflandırmak, kararlar vermek amaçlanmaktadır.

Makine öğrenmesi; yapay sinir ağlarını kullanarak öğrenme verisini(training data) analiz ederek bir model oluşturur. Bu model oluşturma işlemi makinenin öğrenme aşamasıdır. Daha sonra yeni veriler, oluşturulan bu model kullanılarak çözümlenir, sınıflandırılır, çeşitli tespitler yapılır.

Makine öğrenimi algoritmaları giriş verisinin tipine göre, amaca göre, çözülmesi beklenen probleme göre farklılık gösterirler. Bu algoritmalardan bazıları şu şekildedir.

- ***Denetimli Öğrenme (Supervised Learning) :*** Giriş-çıkış bilgisi eşleştirilmiş yani etiketlenmiş verilerle yapılan öğrenme şeklidir. Başka bir ifadeyle, makineye öğreneceği verileri ve çıktı olarak istenenlerin birlikte verilmesiyle yapılan öğrenme şeklidir [11]. Örneğin; türleri ile etiketlenmiş gitar fotoğrafları ile öğrenmenin gerçekleştirilmesi ve sonrasında yeni gitar fotoğraflarının hangi tür

(elektro, klasik, akustik, bass vs.) gitara ait olduğunun belirlenmesi amacıyla denetimli makine öğrenmesi algoritması kullanılmalıdır.

- ***Denetimsiz Öğrenme (Unsupervised Learning) :*** Giriş – çıkış ilişkisi kurulmamış yani etiketlenmemiş veri ile gerçekleştirilen öğrenme şeklidir. Bu öğrenme şeklinde belli bir sınıf belirlenmez, makineden öğrenme verisini sınıflandırması ve daha sonra gelecek verileri öğrendiği verilere göre sınıfını belirlemesi beklenir [11]. Örneğin; Ud, bağlama ve gitar resimlerinden oluşan bir veri setini etiketlemeden makinenin öğrenip sınıflandırması bir Denetimsiz bir öğrenme şeklidir. Öğrenme sonunda makine ud, gitar, bağlama şeklinde öğrenmez, üç farklı sınıf öğrenir. Burada denetimli öğrenme ile vereceği sonuç bakımından farklılık gösterir. Ud, bağlama, bass gitar, elektro gitar, klasik gitar, akustik gitar olmak üzere 6 farklı türde enstrümandan oluşan bir verisetinde tüm gitar türlerini özel tür adları belirlenmeden "Gitar" etiketi ile etiketlenip Denetimli Öğrenme ile öğretilse, sonuçta "4 sınıf"lık bir öğrenme gerçekleşir. Denetimsiz öğrenmede ise 6 farklı enstrüman türünden oluşan bu veri setinden 6'dan fazla sınıfa sahip bir öğrenme gerçekleşebilir. Çünkü gitarlar bass, elektro, klasik, akustik gibi türlerinin yanında bu türler de görünümsel özelliklerine göre de farklılıklar gösterebilmektedir.(Klasik ve akustik gitarların kasa yapısına göre kesik(cutaway), düz vs. elektro gitarların gövde yapısına göre strat, Les Paul, cazz vs. bir çok çeşidi mevcuttur) Bu yüzden sadece gitarları türlerine göre ayırmanın yanında aynı tür gitarları da görünümsel özelliklerine göre de sınıflandıracaktır. Ayrıca gitarların bu sayılanlar dışında da birçok türü mevcuttur ve etiketlenmeyen her veri yeni bir sınıfın ortaya çıkmasına sebep olabilir. Bu durum, amaca göre avantaj veya dezavantaj olabilir. Dolayısıyla amaca uygun algoritmayı seçmek çok önemlidir.
- ***Pekiştirmeli Öğrenme (Reinforcement Learning) :*** Makinenin amaca yönelik ne yapılması gerektiğini çevre ile etkileşim halinde öğrendiği bir makine öğrenmesi yaklaşımıdır [12]. Bu yaklaşımda ödül(reward) ve ceza(punishment) kavramları vardır[11]. Bu kavramlar, öğrenilen yöntem veya yolların ödüllendirilmesi veya cezalandırılması yani iyi veya kötü olmasına göre sınıflandırılır. Bu sınıflandırma veri gruplarının sınıflandırılması olarak görülebilir. Bu öğrenme yöntemi oyun, robot kontrolü gibi alanlarda sıkça kullanılır. Bu öğrenme şeklinde izlenilen her

yol yeni bir veridir ve izlenilecek en doğru yöntem sürekli değişebilmekte ve gelişebilmektedir. Örneğin; bir savaş oyununda, rakiplerin oyuncuya nasıl ataklarda bulunacağı, nasıl vuracağı, nasıl saklanacağı başta kodlanmış olsun. Pekiştirmeli Öğrenmeye dayalı bir algoritma ile desteklendiyse, rakipler davranışlarını oyuncudan öğrenerek geliştirebilir. Kodlarında hiç olmayan bir aksiyonu, oyuncunun davranışlarından öğrenebilir. Bu yöntem otonom araçlarda da kullanılabilir. Bir ülkenin trafik kurallarına ve yol şartlarına göre eğitilmiş bir otonom sürüş sistemi, başka bir ülkenin trafik kural ve yol şartlarını öğrenmemiş olabilir. Ancak bu yöntemle, sürücüsünün davranışlarından yeni davranışlar öğrenebilir.

- ***Anomali Tespiti(Anomaly Detection) :*** Veri madenciliğinde de kullanılan bu yöntem, veri setinde beklenen davranışa uymayan davranış veya kalıp bulmayı amaçlamakta ve sağlamaktadır[13]. Bu beklenmedik durumlara literatürde outliers (aykırı değerler), exceptions (istisnalar) veya anomaliler denilmektedir. Örneğin yöntem telekomünikasyon ve bankacılık sektörlerinde fraud(kaçak) tespitinde kullanılmaktadır. Banka müşterilerinin harcamalarındaki davranışlarından genel profilini çıkarır ve hesaplarında beklenenin dışında bir harcama yakalayıp, müşteriyi uyarabilmekte, hesabı geçici olarak askıya alabilmekte, müşteriyi veya bankayı dolandırıcılığa karşı koruyabilmektedir. Aynı durum telekomünikasyon alanında da uygulanabilmektedir. Bazı müşteriler telekomünikasyon şirketlerinin sistemlerindeki bazı açıkları yakalayıp, bunu kendi faydalarına kullanabilmektedir. Telekomünikasyon şirketleri, müşterilerinin kullanımlarını analiz ederek, normal dışı kullanımlarından kaçak kullanım tespitleri yapabilmektedir.

Yukarıda örneklerle açıklanan makine öğrenmesi yöntemlerinden daha fazla yöntem bulunmaktadır. Ortaklık kuralları(Association rules), Özellik öğrenimi(Feature learning), küme analizi ya da kümeleme (Cluster analysis or clustering) gibi yöntemler bunlardan birkaçıdır.

Çeşitli makine öğrenmesi metodları vardır. SVM(Support Vector Machine – Destek Vektör Makinesi), Classification(Sınıflandırma), Regression(Regresyon - Geri çekme, Düzeltme), Association(Ortaklık kurma), Clustering(kümeleme), Apriori(olası), K-means(K-ortalamaları), Naïve Bayes, KNN (K-Nearest Neighbors - En yakın K komşular)

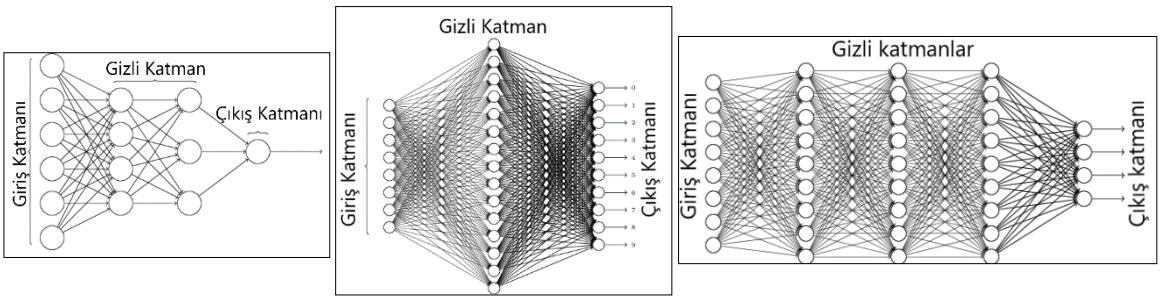
vb. bu algoritmalardan bazılarıdır. Yapay sinir ağları da makine öğrenme metodlarından bir tanesidir. Bu metodların bazıları makine öğrenmesi için geliştirilmiş, bazıları ise veri madenciliğinde hali hazırda kullanılan ve makine öğrenmesinin de gelişmesine katkı sağlayan algoritmalardandır.

1.3.4. Derin öğrenme – Deep Learning (DL)

Yapay sinir ağları veri madenciliğinde kullanılan ve makine öğrenmesinin derin öğrenmeye doğru evirilmesinde en önemli rolü oynayan makine öğrenmesi yöntemidir.

Yapay sinir ağları giriş katmanı, gizli katman ve çıkış katmanından oluşmaktadır. Bu katmanlar yapay nöronlardan oluşmaktadır. Bu nöronlar ise çeşitli matematiksel işlemler yapmaktadır. Nöronlardaki tanımlar aslında birer yazılım parçalarıdır ve yaptıkları işlemler bilgisayarın işlemcisinde(CPU) gerçeklemektedir.

Örneğin, girişinde 3, gizli katmanında 5, çıkışında 1 nörondan oluşan bir yapay sinir ağı tasarladığımızı düşünelim. Her nöronun 1 birin bilgisayar gücüne ihtiyaç duyduğunu düşünelim. Basit bir kombinasyon hesabıyla bu sinir ağımızın 3x5x1=15 birimlik bilgisayar gücüne ihtiyaç duyduğunu düşünebiliriz. Girişinde 5, gizli katmanında 10, çıkış katmanında 3 nöron olsa 5x10x3=150 birim işlemci gücü demek olur. Bu bilimsel bir hesaplama değil ancak yapay sinir ağlarında katmanların genişlemesi yani nöronların artmasının bilgisayara bindireceği gücü ciddi bir şekilde artıracağını anlamak için bir örnek olabilir.



Şekil 1. 8 Çeşitli Yapay Sinir Ağı örnekleri [39] [40]

Yapay sinir ağlarındaki gizli katman sayısının artması ise sinir ağının işlem gücünü artırabileceği gibi ihtiyaç duyacağı işlemci gücünü de artıracaktır. Özellikle de görüntü işlemede daha çok katmana ihtiyaç duyulmaktadır.

Yakın zamana kadar bilgisayar işlemcilerinde ciddi gelişmeler yaşanmıştır. Frekans (GHz) hızlarındaki artış sayesinde birim zamanda seri(peş peşe) yapılabilecek işlem sayısı artmış, çekirdek(core) sayılarındaki artış sayesinde ise bilgisayarların paralel(aynı anda) yapabileceği işlem sayısı artmıştır.

2019 itibariyle masaüstü ve dizüstü kişisel bilgisayarlarda 8 çekirdekli CPU'lar kullanılmakta, bu işlemcilerin frekans hızları 4 GHz'leri geçebilmektedir.

İşlemcilerdeki bu gelişim yapay sinir ağlarında katmanlardaki nöron sayısının daha da artırmaya olanak tanımasının yanında katman sayılarının artmasına olanak tanımış ve çok katmanlı sinir ağlarının tasarlanmaya başlanmıştır.

İşlemcilerdeki gelişimin yanında özellikle bilgisayarda grafik oyun oynamak başta olmak üzere grafik işlemede kullanılan grafik işlemci birimlerinde de (GPU – Graphical Processing Unit) ciddi gelişmeler yaşanmıştır. Grafik işlem performansının artırmak için GPU'ların frekans değerleri ve çekirdek sayılarında ciddi artışlar yaşanmıştır. GPU'daki gelişmeler özellikle oyun ve grafik tasarım sektörlerine hitap eden ticari hamleler olsa da günümüzde birçok alanda fayda sağlamaktadır.

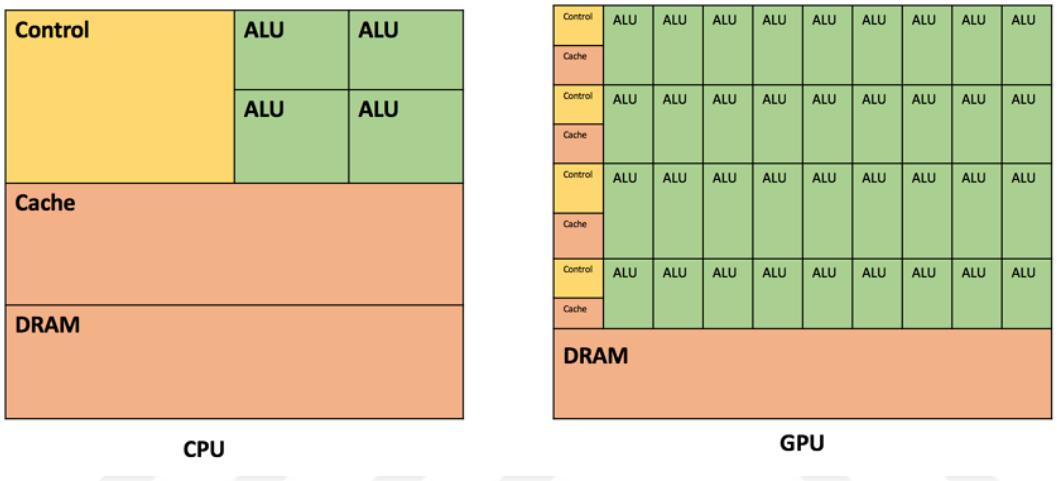
Derin öğrenme kavramının ortaya çıkmasında yapay sinir ağları kadar grafik işlemcilerin gelişiminin de payı büyüktür. Bunu anlayabilmek için GPU'ların nasıl çalıştığını basitçe anlamak gerekir.

GPU'lar aslında özel amaçlarla üretilmiş işlemci birimleridir. CPU'lar bilgisayardaki temel işleri yönetmek, kontrol etmek, çeşitli hesaplar yapmak için geliştirilmişlerdir. Zamanla artan ihtiyaçlara yönelik bilgisayarların seri ve paralel işlem kapasiteleri artırılmıştır. Bundan 20 sene önce işlemci kapasitelerinde MHz değerleri konuşulurken günümüzde işlemci saat hızları (clock-rate) GHz seviyelerine ulaşmıştır. [14]

CPU'larda matematiksel işlemlerin yanında bir çok işlem de gerçekleştirilir. Bilgisayarda çevre birimlerini sürekli kontrol etme, ses işleme, grafik birimlerini kontrol etme ve yönetme bunlardan bazılarıdır. CPU yaptığı matematiksel işlemlerini ise bünyesinde bulunan Aritmetik Mantık Birimi (ALU – Aritmetik Logical Unit) sayesinde gerçekleştirilir. Her işlemcide bir tane bulunan bu yapı, çok çekirdekli işlemcilerde ise her çekirdekte bir tane bulunur. Örneğin; bir toplama işlemini 160 kere hesaplanması isteniyor olsun. Tek çekirdekli bir işlemci bunu 160 döngüde yapar. Yani işlemcideki ALU bu işlemi 160 döngüde gerçekleştirir. Bir işlem 1 birim sürede gerçekleşse, 160 birimde gerçekleşir. 8 çekirdekli bir işlemci ise her bir çekirdeğinde ALU bulunması sayesinde bu 160 işlemi 8 ALU üzerinde 20 döngüde gerçekleştirir. Sadece bu basit örnekte bile çekirdek sayısının yani ALU sayısının 8 katına çıkması sayesinde ciddi bir hız kazanılmış olur. Bu gibi bir işlemin 160 değil de 160 milyon kez yapılmak istendiğini düşünürsek, her bir işlem 1 µs sürse bile tamamlanması tek çekirdek işlemcide 160 saniye, 8 çekirdek işlemcide 20 saniye sürer.

GPU'lar içinde çok fazla (yüzlerce, hatta günümüzde binlerce) ALU barındırırlar. Şekil 1.9'da CPU ve GPU mimarileri karşılaştırılmıştır. Temel matematiksel işlemler tek boyutta hatta boyutsuz işlemlerdir. Ancak görüntü işleme iki boyutta gerçekleşir. Resim ve videolar iki boyutlu verilerdir ve bilgisayarın GPU'larında işlenmektedir. Video bilgisayar için birim sürede peş peşe akan resimlerdir. Resmin veya fotoğrafın bilgisayardaki karşılığı yani matematiksel gösterimi ise matrislerdir.

GPU'lar matris olarak ifade edilen görüntüleri küçük parçalara ayırıp tüm parçalara aynı işlemi aynı anda uygulama yeteneğine sahip, çok çekirdekli işlemcilerdir.



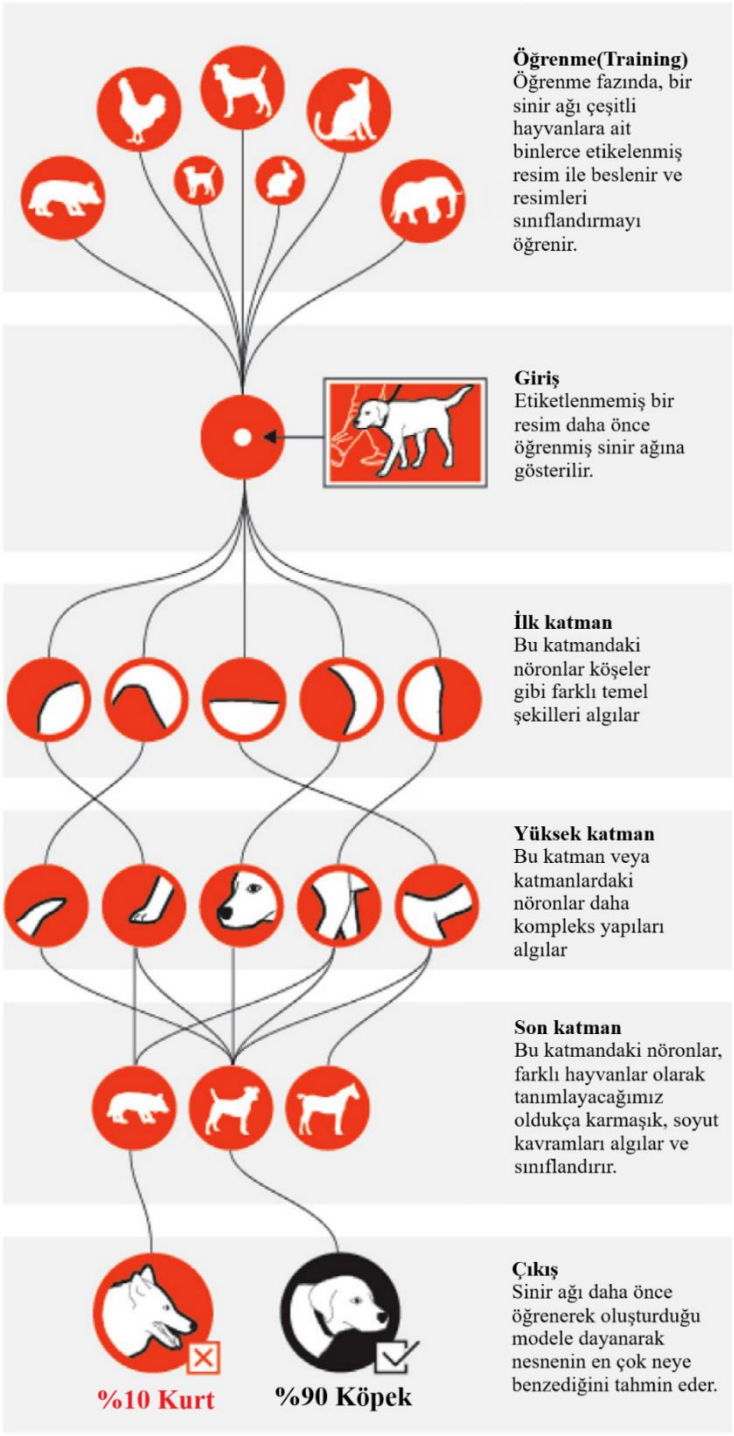
Şekil 1. 9 CPU – GPU mimarisinin karşılaştırılması [41]

GPU'lardaki bu mimari sayesinde bilgisayarların paralel işlem yeteneği çok hızlı artmış ve bu sayede yapay sinir ağlarında çok fazla katman kullanabilmek, her katmanda çok fazla nöron kullanabilmek mümkün olmuştur. Yapay sinir ağlarının çok katmanlı yapıya ulaşması ile sinir ağlarının yeni bir adı olmuştur; Deep Neural Networks yani Derin Sinir Ağları.

Derin sinir ağları sayesinde geliştirilen makine öğrenmesi yöntemi derin öğrenme(deep learning - DL) olarak adlandırılmaktadır.

Derin öğrenme sayesinde özellikle görüntülerde nesne tespitinde ciddi başarılı sonuçlar elde edilmektedir. Şekil 1.10'da derin öğrenme ile nitelik ve nesne tespitinin nasıl çalıştığına basit bir örnek verilmiştir. Gerçek bir derin öğrenme ile tespitte şekil 1.10'dakinden çok fazla nöron olur ve çok fazla detay üzerinde çalışılmaktadır.

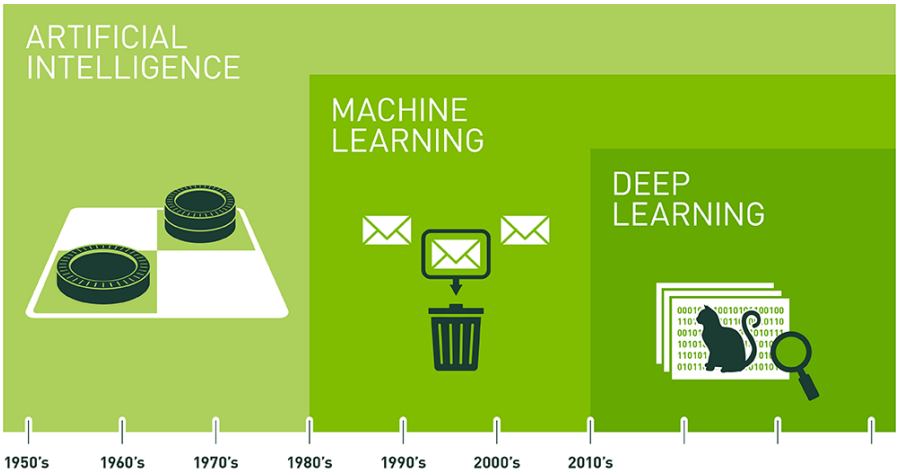
Derin öğrenme ile çok fazla sayıda(on binlerce, yüz binlerce) resmi hızlı işlemek mümkün hale gelmiştir ve başarıda ciddi artış sağlanmıştır.



Şekil 1. 10 Derin öğrenme ile nesne tespitin çalışma şekline basit bir örnek [42]

Derin öğrenme makine öğrenmesinin bir alt dalıdır. Makine öğrenmesi de yapay zekanın bir alt dalıdır. Yapay zekadan derin öğrenmeye evirilen kavram günümüzde derin öğrenme

olarak yoluna devam etmektedir. Şekil 1.11'de yapay zekadan derin öğrenmeye gelinen sürecin basit bir kronolojisi görülmektedir.



Şekil 1. 11 Yapay zeka, Makine Öğrenimi ve Derin Öğrenme kronolojisi. [43]

1.3.5. CUDA ve cuDNN

GPU'ların paralel işleme yeteneği grafik işlemenin yanında derin öğrenme ve kripto para madenciliğinde çok fazla kullanılmalarını sağlamıştır. Zamanla daha fazla ALU barındıran GPU'lara ihtiyaç duyulmuştur ve son yıllarda GPU'ların gelişimine büyük bir hız verilmiştir. GPU üreticileri donanım kapasitesini artırmaya ek olarak yazılımsal geliştirmeler de sunmuşlardır.

İki büyük GPU üreticisinden biri olan NVIDIA (Diğeri AMD), bu noktada uzun zamandır iki ürün geliştirmektedir. Bunlar CUDA ve cuDNN'dir.

CUDA, GPU'larda genel hesaplamalar yapma amacıyla NVIDIA tarafından geliştirilmiş bir paralel hesaplama platformudur. Geliştiriciler CUDA sayesinde GPU'ların gücünden faydalanarak ciddi hız elde etmişlerdir. GPU ile hızlandırılmış uygulamalarda sıralı işlemler CPU'da tek paralel/tek parça olarak peş peşe işlenir. Her işlemin altında yapılacak hesaplama işlemleri ise binlerce GPU çekirdeğinde işlenir. Geliştiriciler C, C++, Fortran, Python ve MATLAB gibi popüler programlama dillerinde yazdıkları kodlarda CUDA sayesinde birkaç temel anahtar kelime ile paralel işlem yapabilmektedirler. NVIDIA, CUDA Toolkit(araç kiti) GPU ile hızlandırılmış uygulamalar geliştirmek için ihtiyaç olan herşeyi içinde barındırmaktadır.

NVIDIA cuDNN (NVIDIA Deep Neural Network) kütüphanesi Derin Sinir Ağları için GPU ile hızlandırılmış temel bir kütüphanedir. cuDNN derin öğrenmede kullanılan çeşitli

fonksiyonlar(konvolüsyon, normalizasyon, aktivasyon katmanları vs.) için büyük ölçüde ayarlanmış uygulamalar sunmaktadır. Caffee, Caffee 2, Keras, Chainer, TensorFlow, MxNet, MATLAB, PyTorch gibi çok kullanılan derin öğrenme çatıları(framework) cuDNN ile hızlandırılmıştır.

1.4. Yapılmış Çalışmalar

Derin öğrenme günümüzün en çok kullanılan veri analiz yöntemlerindendir. Birçok alanda olduğu gibi sağlık alanında çeşitli analiz ve teşhisleri daha akıllı hale getirmek için kullanılmaktadır.

Bu çalışmada derin öğrenme yöntemleri tıbbi görüntülerden kanserli bölge tespitinde kullanılacaktır. Literatürde yapay zeka, makine öğrenimi ve derin öğrenme algoritmaları ile kanser üzerine yapılmış birçok çalışma bulunmaktadır.

Aaron N. Richter ve Taghi M. Khoshgoftaar'ın [16] 2018 yılında yaptıkları çalışmada onkoloji araştırmacılarına, istatistikçilere ve veri bilimcilerine, hastaların kanser olma ve tekrar etme riskini tahmin etmede kullanılan güncel yöntemler hakkında bilgi verme amaçlı bir araştırma gerçekleştirmişlerdir. Çalışmada çeşitli kaynaklardan veriler toplanmıştır. Bu veriler hastaların moleküler, klinik ve pratik verilerinin yanında sosyal ve yaşam tarzlarıyla ilgili verileri de içermektedir. Bu verilerin bazıları belli kanser türlerinin riskleri için birbirleriyle ilişkilendirilmiştir. Örneğin; sigara tüketimi ile akciğer kanseri, alkol tüketimi ile karaciğer kanseri, ultraviyole ışınlara maruz kalma ile cilt kanseri ilişkilendirilmiştir. Ayrıca bu verileri bazı özellikleri beş kategoride gruplandırmışlardır. Bu kategoriler aşağıdaki gibidir.

- Demografik veriler: Bu çalışmada sadece yaş ve cinsiyet
- Laboratuvar verileri: Beyaz kan hücre sayıları, hemoglobin, glikoz, trigliseritler vs.
- Histopatolojik veriler: Bölge, tümör boyutu, metastaz, evre, marjinler gibi kanser ve tümörle ilişkili bilgiler
- Klinik veriler: Tedaviler, aile geçmişi, hayati organlar, diğer kategorilere uymayan ve rutin olarak alınan klinik bilgiler vs.
- Yaşam tarzıyla ilgili veriler: Sigara ve alkol tüketimi gibi sosyal geçmiş ile ilgili bilgiler

Veriler hazırlandıktan sonra model geliştirme anlatılmış ve bazı gelişmiş veri madenciliği teknikleri kullanılarak bu modellerin performanslarının nasıl artırılabileceği anlatılmıştır.

Alexander W. Forsyth ve arkadaşlarının[17] 2018 yılında yaptıkları çalışmada elektronik sağlık kayıtlarından meme kanseri semptomlarının dokümantasyonunun makine öğrenme metotları kullanılarak çıkarılması çalışılmıştır. Çalışmada kullanılan veri setinde iki akademik kanser merkezinden alınmış, 1996 Mayıs - 2015 Mayıs arasındaki paklitaksel (meme, yumurtalık, akciğer, mesane, prostat, gırtlak kanseri gibi kanser türlerinin tedavisinde kullanılan kemoterapi ilacı) içeren kemoterapi tedavisi almış 2965 adet meme kanseri hastasına ait klinik notlarından elde edilen 103 564 cümle bulunmaktadır. Bu cümlelerin 10.000 tanesinin rastgele alanları aktif semptom(pozitif etiket), semptom yokluğu (negatif etiket) ve hiçbir semptom ile ilişkisi olmayan (nötr etiket) şeklinde işaretlenmiştir.

Örneğin; "patient *describes* having *nausea,* vomiting, *and pain*,"("hastada mide bulantısı, kusma ve ağrı var") cümlesindeki kelimelerin etiketleri sırasıyla ["nötr," "*nötr*," "nötr," "*pozitif*," "pozitif," "*pozitif*"] şeklinde olacaktır. Veya "patient *denies* having *nausea*,"("hasta mide bulantısı olmadığını iddia ediyor") cümlesindeki kelimelerin etikerleri sırasıyla ["nötr," "*nötr*," "nötr," "*negatif*"] şeklinde olacaktır.

İnsan eliyle kodlanan bu 10.000 adet veri cümle öğrenme, doğrulama ve test verisi olarak bölünmüştür. Verinin %20'si(2000 cümle) test verisi olarak ayrılmıştır. Daha sonra bazı makine öğrenimi algoritmaları ile CRF(Conditional Random Field – Şartlı Rastgele Alan) modeli oluşturularak 10000 cümlenin 8000 tanesi makineye öğretilmiştir. Öğretilen cümlelerden sonra sistemin verileri işlemesi sağlanmış ve insan-makine karşılaştırması yapılmıştır. Aşağıdaki örneklerde ilk cümle insan tarafından, ikinci cümle ise makine tarafından etiketlenmiştir. (kalın kelimeler pozitif(**P**), eğik yazılar negatif (*N*), normaller nötr(0))

İnsan	she denies *fever chills* but has **rigors**	(0 0 *N N* 0 0 **P**)
Makine	she denies *fever chills* but has rigors	(0 0 *N N* 0 0 0)
	(*ateşli titremesinin* olmadığını iddia ediyor ancak **titremesi** var)	

İnsan	ct dye caused a high **fever** and she has a **rash and blistering**	(0 0 0 0 0 **P** 0 0 0 0 **P P**)
Makine	ct dye caused a high **fever** and she has a rash and **blistering**	(0 0 0 0 0 **P** 0 0 0 0 0 **P**)
	(ct boyası yüksek **ateşe** neden oldu ve **kızarıklık** ve **kabarması** var)	

İnsan	she has a **cough** with green sputum and **dysuria** (0 0 0 **P** 0 0 0 **P**)
Makine	she has a **cough** with green sputum and dysuria (0 0 0 **P** 0 0 0 0)
	(yeşil balgamlı **öksürüğü** ve **ağrılı idrar** var)

Sanjay Purushotham ve arkadaşları[18] 2018'de yaptıkları çalışmada büyük medikal veri setleri üzerinde derin öğrenme modellerinin performanslarını kıyaslamışlardır. Detaylı kıyaslamayı MIMIC-II veri seti üzerinde gerçekleştirmişlerdir. Derin öğrenme yöntemlerinin mevcuttaki analiz modelleriyle kıyaslandığında en iyi performansı verdiğini ortaya koymuşlardır.

Zulong Hu ve arkadaşları[19] 2018'de yaptıkları çalışmada görüntü tabanlı kanser tespiti ve teşhisinde derin öğrenme uygulamalarını araştırmışlar ve incelemişlerdir. Yakın zamanda derin öğrenme uygulamalarının kullanıldığı çalışmaları 9 kategoride sınıflandırmışlar. Bu konudaki akademik araştırmaların büyük çoğunluğu meme, akciğer, cilt, prostat, beyin, kolon, rahim ağzı, mesane ve karaciğer kanserleri üzerine yapıldığını tespit etmişlerdir. Görüntü tabanlı derin öğrenme modellerinin en popüler dördü Konvolüsyonel Sinir Ağları(Convolutional Neural Networks - CNN), Tam Konvolüsyonel Ağlar(Fully Convolutional Networks – FCN), Otomatik Kodlayıcılar(Auto-encoders) ve Derin İnanç Ağları(Deep Belief Networks)'dır.

Azam Hamidinekoo ve arkadaşları [20] 2018'de yaptıkları çalışmada mamografi ve meme dokubiliminde derin öğrenmeyi ele almışlar ve bu alandaki gelecek trendleri incelenmiştir. Çalışmada önce derin öğrenme tabanlı modeller hakkında temel bilgi verilmiştir, daha sonra mamografi ve histopatolojik görüntü analizlerinde kullanılan mevcut derin öğrenme tabanlı yaklaşımlar incelenmiştir. Çalışmanın devamında mamografi ve histoloji görüntü arasında kurulabilecek ilişkiler incelenmiştir. Son olarak mamografik ve histolojik özellikler ve fenotipler(kalıtımsal dış görünüş) arasında bağlantı kuran bir derin öğrenme modeli geliştirilebileceği anlatılmıştır. Bu planlanan modele "Mammography - Histology - Phenotype - Linking - Model" (Mamografi - Histoloji - Fenotip - Bağlantı - Modeli) olarak adlandırılmış ve ($ML_{M<->H}$) şeklinde sembolize edilmiştir.

Barış Geçer ve arkadaşları[21] 2018'de yaptıkları çalışmada tam kesit meme histopatoloji görüntülerinden kanser tespiti ve sınıflandırmasında derin konvolüsyonel ağlar (Deep Convolutional Networks - DCN) kullanmışlardır. Çalışmada görüntüleri kanserli –

kansersiz diye sınıflandırmak yerine çok sınıflı senaryolar ile beş teşhis grubunda sınıflandırmışlardır. Çalışmayı MathConvNet kütüphanesi kullanarak, donanımı NVIDIA GeForce GTX-970 GPU, Intel Xeon ES-2630 2.60 GHz CPU ve 64GB RAM olan bir bilgisayarda gerçekleştirmişlerdir.

Shengfeng Liu ve arkadaşları [22] 2019'da yaptıkları çalışmada popüler derin öğrenme mimarilerini tanıtmış ve daha sonra Ultrasonografi görüntülerinde hangi derin öğrenme yönteminin hangi amaçlarda daha çok kullanıldıkları, hangi algoritmanın daha popüler olduğu, hangi algoritmanın hangi organ görüntüsünde daha fazla kullanıldığı gibi çeşitli analiz ve karşılaştırmalar yapmışlardır. Karşılaştırmalarını sınıflandırma(classification), tespit etme(detection) ve bölümleme(segmentation) olmak üzere üç ayrı algoritma grubunu karşılaştırmışlar ve bu yöntemlerin çoğunlukla fetüs, meme ve prostat ultrason görüntüleri üzerinde kullanıldığını göstermişlerdir. Yıllara göre dağılımda bu üç algoritma grubunun kullanımında 2016 yılı itibariyle ciddi bir artış başlamış olup, 2017'de 2016'nın iki katından fazla çalışma yapıldığı sonucuna varılmıştır. Üç algoritma grubunda da en çok kullanılan yapay sinir ağı mimarisinin Konvolüsyonel Sinir Ağı (Convolutioal Neural Network - CNN) olduğu sonucuna varılmıştır.

Alexander Selvikvåg Lundervold ve Arvid Lundervold [23] 2018'de yaptıkları çalışmada derin öğrenme mimarilerini karşılaştırmışlar ve Konvolüsyonel Sinir Ağı (Convolutioal Neural Network - CNN) kullanarak MRI(Magnetic Resonance Imaging - emar) görüntülerinde bozuklukları düzeltme, eksikleri yerine koyma, organ tespiti, sınıflandırma gibi çeşitli uygulamaları gerçekleştirmişlerdir.

Srivarna Settisara Janney ve Sumit Chakravarty [24] 2019'da yaptıkları çalışmada derin öğrenmenin medikal ve cerrahi enstrümanlardaki kullanım alanlarını, geçmişi, günümüzdeki durumunu derleyip gelecekte olabilecek gelişmelerden bahsetmişlerdir. Otonom cerrahi operasyonların derin öğrenme sayesinde gelişeceğini, derin öğrenme ile birçok hastalığa yeni ilaçların keşfedilebileceğini, Sanal Gerçeklik(Virtual Reality –-VR) ve Artırılmış Gerçeklik (Augmented Reality - AR) ile derin öğrenmenin birleşimi ile tıp alanında sanal görselleştirmede ciddi gelişmeler yaşanılacağı savunulmuştur. Savundukları bu ve daha birçok maddeyi, birçok markanın bu alanlarda yaptığı endüstriyel çalışmalardan örnekler vererek desteklemişlerdir.

2. MATERYAL VE YÖNTEM

Bu bölümde derin öğrenme ile kanser tespitinde kullanılan materyaller ve izlenen yönyem anlatılmıştır.

2.1. Kullanılan Veri Seti

Bu çalışmada BreCaHAD adlı veri seti kullanılmıştır. [27] Veri setinin adı **Bre**ast **Ca**ncer **H**istopathological **A**nnotation and **D**iagnosis (Meme Kanseri Histopatolojik Açıklama ve Teşhis) ifadesinin kısaltmasından gelmektedir.

Histoloji: Yunancada zar ya da doku anlamına gelen histos ve bilim anlamına gelen logos kelimelerinin birleşmesi ile dilimize doku bilimi olarak çevrilir. Histoloji canlıya ait yapıların özelliklerini, yapılarla fonksiyonların ilişkilerini mikroskobik olarak inceleyen bilim dalıdır. [25]

Patoloji: Hastalık (Yunanca pathos) ve bilim (Yunanca logos) kelimelerinin birleşmesi ile oluşmuş hastalıklar bilimi anlamına gelen bir sözcüktür. Patoloji (hastalıkbilim) özellikle altta yatan hastalıkla ilgili hücrelerdeki, dokulardaki ve organlardaki yapısal ve işlevsel değişikliklerin tanınması, araştırılması ve incelenmesiyle ilgilenir. [26]

Histopatoloji: Hastalıklı dokunun histolojik incelenmesinde uzmanlaşan, patolojik histoloji olarak da bilinen patolojinin bir alt dalıdır.

Veri seti 162 adet meme kanseri histopatoloji görüntüsünden oluşmaktadır. Veri seti indirildiğinde aşağıdaki dosyalar gelmektedir.

- annotation_details.xlsx : Veri setindeki sınıflar ile ilgili bilgiler
- original.png : işaretlenmemiş bir örnek imaj dosyası
- annotated.png : örnek imaj dosyasının işaretlenmiş kopyası
- data.json : işaretlenmiş imaj örneğinin işaretine ait sayışal verileri içeren JSON formatlı dosya
- BreCaHAD.zip : Tüm veri seti (TIF formatında işaretlenmemiş ve PNG formatında işaretlenmiş imaj dosyaları, işaretleme ile ilgili JSON dosyaları vs.)

BreCaHAD.zip dosyası dışarı çıkarıldığında(uncompress-unzip) birkaç dizin altında veri setine ait dosyalara erişilebilmektedir.

"*BreCaHAD/images*" dizini altında 1360 x 1024 çözünürlükte ve 8 bit derinlikli 3 kanallı RGB renklendirmeye sahip sıkıştırılmamış formatta (.tif uzantılı) 162 adet piyopsi görüntüsü bulunmaktadır.
"*BreCaHAD/groundTruth_display*" dizini altında aynı görüntüler aynı çözünürlük ve renklendirme özellikleri ile PNG formatında(sıkıştırılmış) ve işaretlenmiş olarak bulunmaktadır.
"*BreCaHAD/groundTruth*" dizini altında ise işaretlenmiş görüntülerin işaretlerine ait koordinat bilgileri JSON formatında ve [0,1] aralığında normalize edilmiş halde bulunmaktadır. Bu normalizasyon noktanın merkezinin x ve y piksel değerinin en ve boy pixel değerine bölünmesi ile hesaplanmaktadır.
Şekil 2.1'de veri setinden bir görüntü işaretlenmemiş ve işaretlenmiş şekilde JSON formatlı koordinat bilgileri ile birlikte gösterilmiştir.

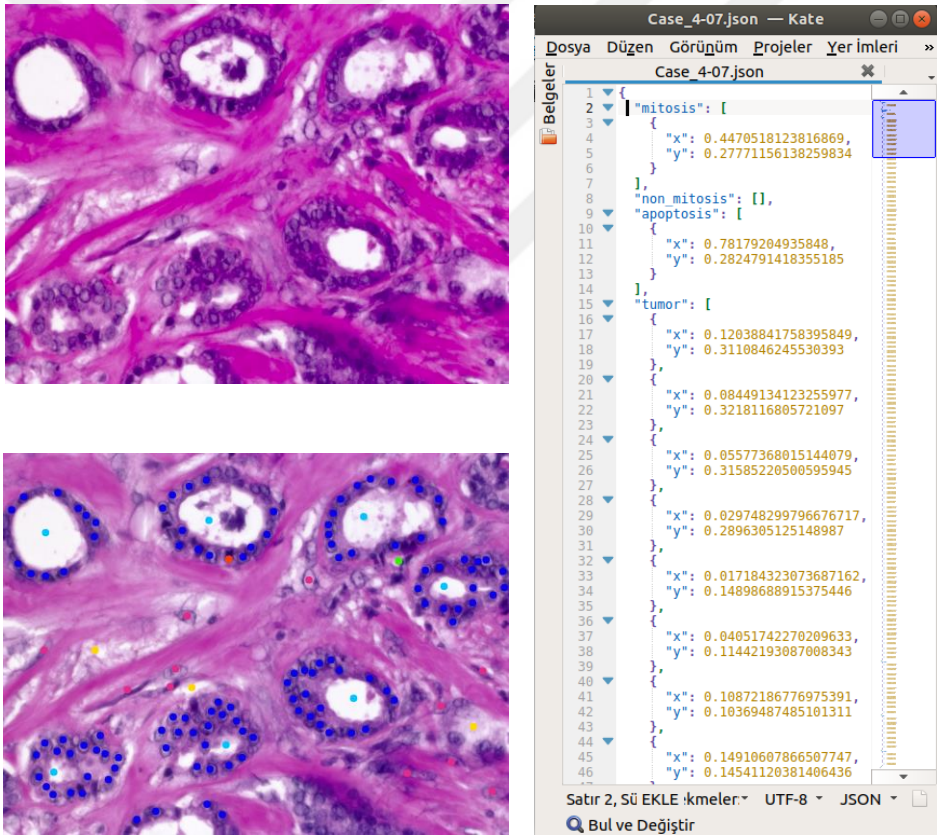
Biyopsi görüntüleri üzerinde hücrelerin çekirdeği hücre tiplerine göre farklı renkte nokta ile işaretlenmiştir. İşaretlerin renkleri ve etiketlemenin yapıldığı hücre türü ile veri seti içindeki sayıları Tablo 2.1'de gösterilmiştir.

Bu veri setinde bulunan JSON formatlı koordinat bilgisi bu çalışmada yeterli olmadığı için veri setinde bazı değişiklikler yapılmıştır. İlerleyen başlıklarda anlatılacak olan bu değişiklikler ile veri setinin zenginleştirilmesi ve başka çalışmalara da katkı sağlayacak hale gelmesine de katkı sağlanmıştır.

Veri setindeki görüntülerde birçok kanser ve kanser olmayan hücre türü vardır. Veri setinde Mitoz, Apoptoz, Tümör çekirdeği, Tümör olmayan hücre çekirdeği, Tübül ve Nontübül olmak üzere 6 farklı hücre türü işaretlenmiştir. Dolayısıyla veri setimizde 6 sınıf vardır. Görüntülerde ductal carcinoma, lobular carcinoma, mucinous carcinoma veya tubular carcinoma gibi hücre türleri de bulunmaktadır ancak bu tür hücreler işaretlenmemiştir. Dolayısıyla bu hücre türleri veri setine dahil edilmemiştir.

Tablo 2. 1. BreCaHAD veri setindeki işaretlenmiş hücre tipleri, işaret rengi ve hücre sayıları

Hücre Türü	**İşaret(Nokta) Rengi**	**İşaretli Hücre sayısı**
Mitosis	Kırmızı	112
Apoptosis	Yeşil	263
Tumor nuclei	Koyu Mavi	20117
Non-tumor nuclei	Pembe	1917
Tubule	Açık Mavi	486
Non-tubule	Sarı	601
TOPLAM		**23549**



Şekil 2. 1 BreCaHAD Veri setinden bir işaretlenmemiş ve işaretlenmiş bir örnek ve işaretlemelerin JSON formatlı [0,1] aralığında normalize edilmiş koordinatları

2.2. Konvolüsyonel Sinir Ağı (CNN) ve Tensör

Konvolüsyonel sinir ağları derin öğrenmede kullanılan yapay sinir ağı çeşitlerinin başında gelir. Yapay sinir ağları öğrenilecek resimleri tensör tipinde işlerler. Bu bölümde önce Tensörler kısaca anlatılacak, daha sonra CNN'den bahsedilecektir.

2.2.1. Tensör

CNN'i daha iyi anlayabilmek için Tensör kavramını anlamak gerekmektedir.

Matematikte sayısal büyüklükler skaler ve vektörel olarak ikiye ayrılmaktadırlar. Skaler büyüklükler sadece sayısal veriden ibaret iken, vektörel büyüklükler ise sayısal verinin yanında yön bilgisini de taşıyan büyüklüklerdir. Hız, ivme, Momentum, Kuvvet gibi fiziksel büyüklükler vektöreldir. İki boyutlu vektörler x-y, üç boyutlu vektörler x-y-z eksenlerinde gösterilirler. Vektörel büyüklükler boyutları kadar bileşenden oluşurlar. Örneğin x-ekseni ile 53 derece açı yapan ve 5 birim büyüklükteki iki boyutlu bir vektör x ekseninde 3 birim, y ekseninde 4 birimlik bileşene sahiptir. (3-4-5 üçgeni olarak bilinen konu)

Fizikte vektörel işlemler bir noktadan başka bir noktaya olacak şekilde ifade edilmektedirler. Ancak vektörlerin temsil etmekte yetersiz olduğu büyüklükler vardır.

Örneğin 1280x1024 piksel boyutlu 8-bit renk derinlikli bir resim. Bilgisayarda tüm renkler kırmızı, yeşil ve mavinin belli sayısal değerlerde toplanmasıyla elde edilmektedir. Bu örnek resmimiz 1280x1024 boyutlu bir matristir ve her bir hücresinde bir renk barındırmaktadır. Her bir renk ise kırmızı-yeşil ve mavi bilgisinin toplamı olan bir sayısal değerdir. Bu sayısal değer ise üç farklı renk kodunu barındıran bir veridir.

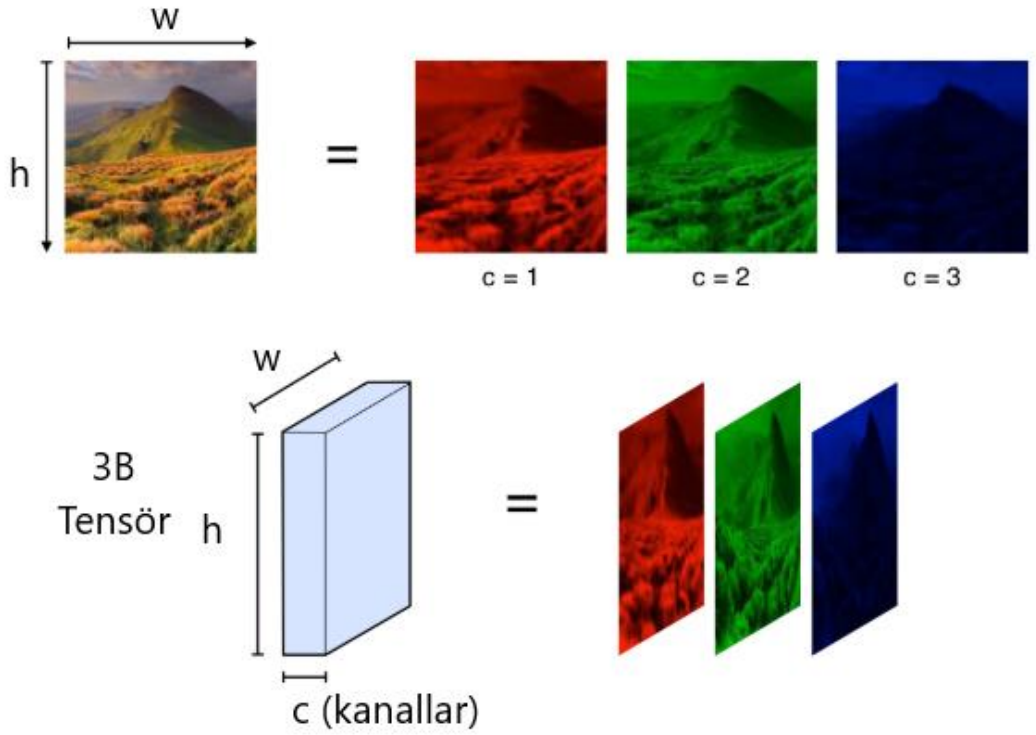
Daha doğru bir ifadeyle, her bir piksel renk değerlerini barındıran 1x3'lük bir matristir. Resmimiz ise 1280x1024 boyutunda, her bir hücresinde 1x3'lük bir matris barındıran başka bir matristir. Özetle resmimiz 1280x1024x3 boyutunda üç boyutlu bir matristir.

Elimizdeki veri daha fazla boyutta da olabilir. Örneğin, bir oyun geliştirme ortamında üç boyutlu bir matris olan bir resim dosyasının belli bir yönde hareket etmesiyle, günlük dördüncü bir boyut olarak hız boyutu da gelebilir. Hızın-yolun bir açı bilgisi varsa, bu sefer yön bilgisi iki eksende de ifade edilebileceğinden verimiz 5 boyuta çıkabilir.

Buradaki boyut kavramı günlük hayattaki en-boy-yükseklik bilgisinden farklı olarak, bir veriye ait birçok detayın tek bir matris ile ifade edilmesidir.

Cebirsel olarak ifade edilirse, vektörlerden oluşan çok boyutlu matrisler Tensör olarak adlandırılır.

Derin öğrenme resim verilerini tensör olarak kullanır. Tensörlerin değişik alanlarda değişik tanımları vardır ancak derin öğrenmede resim üzerinde tespitler yapacağımız için resim verilerini 3 boyutlu tensörler olarak anlamamız yeterlidir.



Şekil 2. 2 Bir resmin tensör olarak gösterimi [44]

2.2.2. CNN

Bu çalışmada kullanılacak olan derin öğrenme aracı (YOLO), CNN tabanlı bir yazılımdır. Konvolüsyonel Sinir Ağının nasıl çalıştığını anlamak için önce konvolüsyonu anlamak gerekir.

Konvolüsyon, Türkçe'ye evirişim olarak çevrilmiş, sinyal teorisinde konvolüsyon toplamı ve konvolüsyon integrali olarak iki başlıkta incelenen bir matematiksel işlemdir. Dürtü cevabı (impulse response) bilinen bir sistemin girişine başka bir sinyal

uygulandığında çıkışında yeni bir sinyal elde etmeyi amaçlayan bir işlemdir. Elektronikte filtreler konusu bu işlemin uygulamadaki karşılıklarından biridir.

Konvolüsyon eğer ayrık zamanlı(discrete time) sinyallerde yapılıyorsa Konvolüsyon toplamı, sürekli zamanlı (continuous time) sinyallerde yapılıyorsa konvolüsyon integrali uygulanır. Konvolüsyon'un genel formülleri aşağıdaki gibidir.

- Konvolüsyon Toplamı;

$$\boldsymbol{y[n] = x[n] * h[n] = \sum_{m=-\infty}^{\infty} x[m].\, h[n-m]}$$

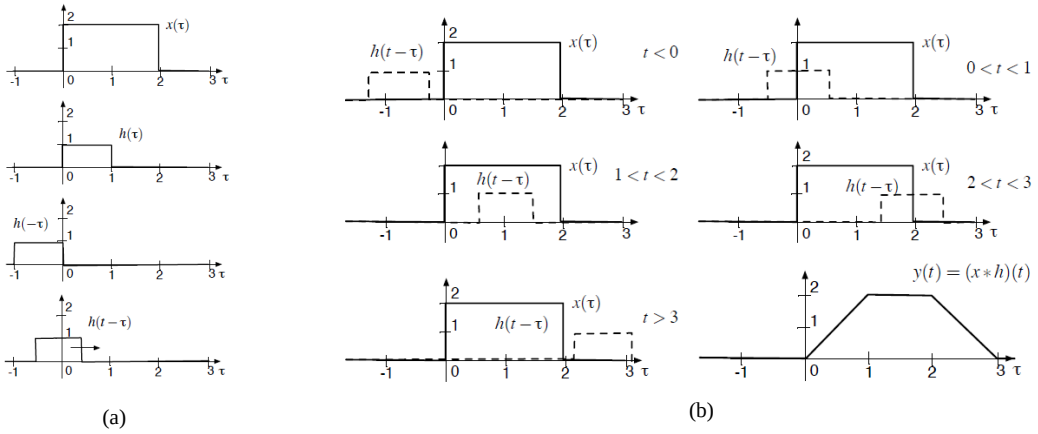
h[n] : Ayrık sistemin dürtü fonksiyonuna cevabı
x[n] : Giriş sinyali
y[n] : Çıkış sinyali
n : Ayrık zaman değişkeni
* : konvolüsyon işareti

- Konvolüsyon İntegrali;

$$\boldsymbol{y(t) = x(t) * h(t) = \int_{\tau=-\infty}^{\infty} x(\tau) h(t-\tau) d\tau}$$

h(t) : Sürekli sistemin dürtü fonksiyonuna cevabı
x(t) : Giriş sinyali
y(t) : Çıkış sinyali
t : Sürekli zaman değişkeni
* : konvolüsyon işareti

İki formülde de görüleceği üzere konvolüsyon işleminde dürtü cevabı giriş zamanda ters çevrilir. Daha sonra giriş sinyali ile tüm zaman değerlerinde çarpılarak, çarpımların toplamı hesaplanır. Bu toplam işlemi ayrık zamanda tek tek toplam iken, sürekli zamanda integral olarak karşımıza çıkmaktadır. Konvolüsyonu bir sistemin dürtü cevabını başka bir sinyalin üzerinde gezdirerek çıkış sinyalini elde etmek olarak da anlayabiliriz. Şekil 2.2'de sürekli zamanda basit bir konvolüsyon işlemi örneği verilmiştir.

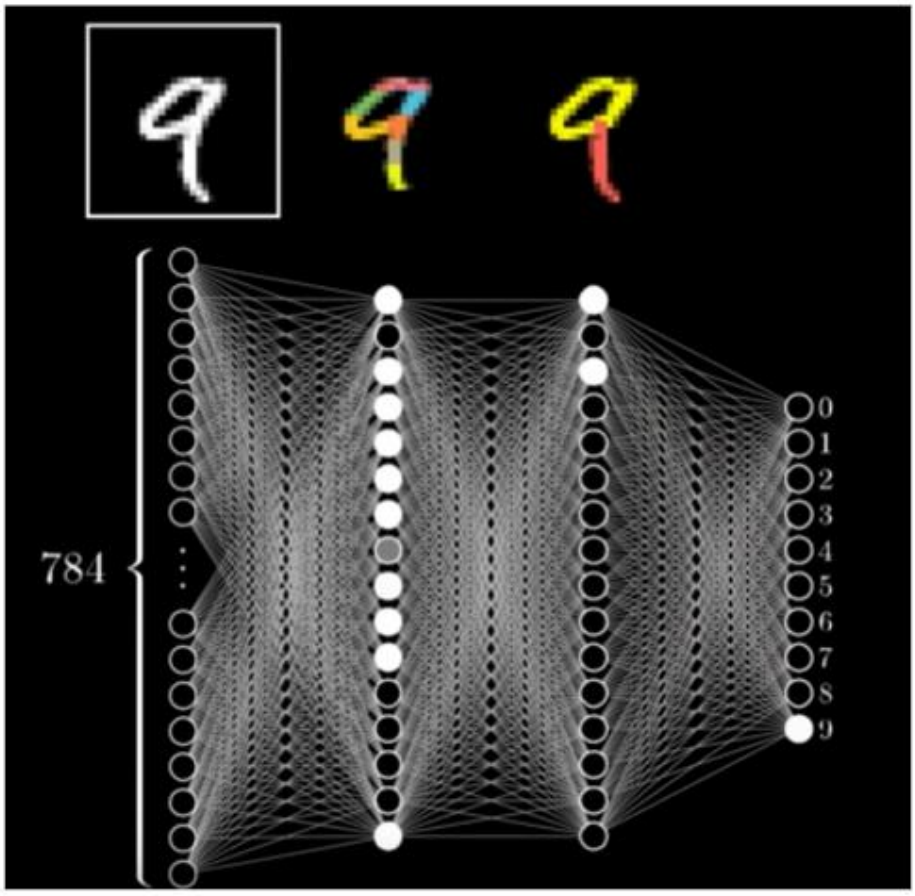


Şekil 2. 3 (a) *x(τ)* : Giriş sinyali *h(τ)* : Sistemin dürtü cevabı ve zamanda ters çevrilmesi (b) Ters çevrilen dürtü cevabının, zaman ekseni boyunca ilerletilmesi ve çıkış sinyalinin elde edilmesi, *y(τ)* : çıkış sinyali [45]

Konvolüsyonel Sinir Ağları (CNN) , içerisinde konvolüsyonel katman veya katmanlar bulunduran yapay sinir ağlarıdır. Konvolüsyonel katmanlarda sinyal teorisindeki konvolüsyon işlemine benzer bir mantığı temel alarak nitelik çıkarmayı amaçlayan işlemlerin gerçekleştiği nöronlar vardır. Bu nöronlar esasında sinir ağlarının filtreleme işlemlerini gerçekleştirmektedir. Konvolüsyonel katmandaki nöronların her biri kendisinden önceki katmandan gelen veriyi(bu çoğunlukla bir resim) üzerinde çeşitli detaylar(köşe, kenar, ovallik, düzlük, yuvarlaklık, üçgenlik, renk, vs.) tespit etmektedirler. Tespit ettikleri niteliklerde ortaklıkların ağırlıkları arttıkça da sistemin öğrenmesi artmaktadır. Şekil 2.4'teki örnekte, girişe verilen bir görüntünün "9" olduğu, her konvolüsyonel katmanda görüntüye ait çeşitli nitelikler tespit edilmekte ve her detayın hangi gruba daha yakın olduğu karşılaştırılarak tespit edilmektedir.

CNN'deki konvolüsyonel katmanları oluşturan nöronların her biri farklı bir detayı tespit edecek filtreleri barındırmaktadır. Bu filtreler tek başlarına bir anlam ifade etmeyebilirler ancak tespit edilecek nesnelerin çeşitli detaylarının tek seferde bir görüntüde bulunması, o görüntüdeki tespit edilecek nesnenin tahmin edilme şansını artırmaktadır. Basit bir örnek vermek gerekirse, görüntülerdeki şekillerin dikdörtgen ve üçgen olup olmamasını tespit eden bir sistemimiz olduğunu varsayalım. Filtrelerimizden biri görüntüde bir "L" şekli yani bir köşe tespit etti. Bu tek başına makinenin şekli sınıflandırması için yeterli olmayabilir. Veya yine aynı görüntüde sağ alt veya sağ-sol üstteki dik açılı köşe tespiti

yapması da yeterli olmayabilir. Ancak aynı anda aynı görüntüde dört tane farklı noktalarda dik açılı köşe yakalaması görüntüyü diktörtgen olarak sınıflandırması için yeterlidir. Veya tek bir dik açı, iki tane dar açılı köşe yakalaması da görüntünün üçgen olarak sınıflandırılmasını sağlayacaktır.

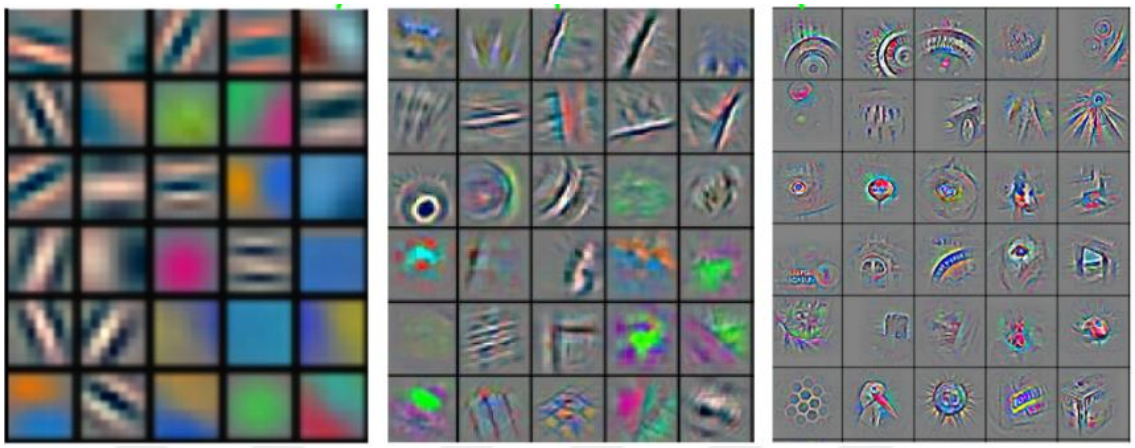


Şekil 2. 4 Bir CNN'deki konvolüsyonel katmanların "9" şeklindeki çeşitli detayları tespit ederek sınıflandırması [46]

Günümüzde Machine Vision(makine görmesi) olarak bilinen ve makinenin insan gibi görebilmesini temsil eden kavramın derin öğrenmedeki karşılığı CNN'deki olduğu gibi küçük detayları tespit edip bu detayları birleştirerek en çok ortak özelliklerden görüntünün ne olduğunun tespiti olarak karşımıza çıkmaktadır.

Şekil 2.5'te her bir sinir ağındaki her katmanda tespit edilen farklı detaylar görülmektedir. Görüleceği üzere filtrelerin tespit edeceği nitelikler her katmanda daha kompleksleşmekte ve daha özel şekillere odaklanmaktadır. Örneğin, bu ağımızda arı peteği tespiti 3.

konvolüsyonel katmanda yapılabilecekken, göz tespiti 2. konvolüsyonel katmanda gerçekleşecektir. Eğer gözün hangi canlıya ait olduğu bilgisi de aranıyorsa 3. katmanda hatta belki de daha ilerideki bir katmanda filtrelenecektir.



Şekil 2. 5 Bir CNN'deki konvolüsyonel katmanlardaki çeşitli filtreler[47]

2.2.3. CNN'deki Katmanlar

CNN'de Konvolüsyonel katman dışında başka katmanlar da bulunur. Konvolüsyonel katman tek başına nesne tespitinde yeterli olamayacağı için başka işlemlerin de gerçekleştiği katmanlar kullanılmaktadırlar. CNN katmanları özetle şunlardır.

- ***Konvolüsyonel Katman:*** Görüntüde filtreleme işlemlerinin yapıldığı katmandır.

- ***Pooling:*** Görüntüde indirgeme yapan katmandır. Görüntü üzerinde belli büyüklükte bir filtre gezdirilir ve bu filtrenin gezdiği alandan aranılan sayısal değeri çıkarır. Örneğin 2x2'lik bir Max Pooling filtresi gezdiriliyorsa, filtrenin gezdiği 2x2'lik bölgedeki en büyük sayısal değere sahip piksel alınır. Veya 3x3'lük Average Pooling filtresi gezdiriliyorsa, filtrenin gezdiği her 9 pikselin sayısal değerlerinin ortalaması yeni piksel değeri olarak alınır.

- ***Fully Connected Layer(Bütünüyle Bağlı Katman) :*** Önceki katmanlardan elde edilen çeşitli özniteliklerle oluşturulan sınıfların yanyana getirilmesi ile oluşturulan katmandır. Örneğin; 0-9 arası sayıların tespitini yapan bir CNN'de Fully Connected Layer'da 10 tane sınıf olur. Bu sınıflar her sayının şeklen sahip olduğu öz niteliklerden oluşturulmuştur.

CNN'de bu katmanlar çeşitli parametrelerle konfigüre edilerek sinir ağının başarı ve hız gibi yetenekleri değişmektedir.

2.3. Darknet ve YOLO

Darknet, Washington Üniversitesi – Bilgisayar Bilimi & Matematik bölümü mezunu Joseph Redmon tarafından C dili ve CUDA kullanılarak geliştirilmiş, açık kaynak kodlu bir Sinir Ağı Çatısı(Neural Network Framework)'dır. [28]

Joseph Redmon, Darknet'in kaynak kodları başta olmak üzere Darknet ile ilgili birçok bilgiyi kendi kişisel sitesi [28] ve GitHub sayfası[29] üzerinden pjreddie takma adı ile paylaşmaktadır.

Darknet hızlı çalışan, yüklemesi ve kullanması çok kolay bir sistemdir. Hem CPU hem de GPU'da işlem yapmayı desteklemektedir. Ayrıca OpenCV desteği vardır. Bu sayede bir çok görüntü formatı ile işlem yapmak mümkündür. Ayrıca OpenCV desteği sayesinde kameradan alınan anlık görüntü üzerinde de çalışmak mümkündür.

Darkneti hazır haliyle kullanabilmek için bir Linux veya Unix sisteme ihtiyaç duyulmaktadır. Darknet'i Windows üzerinde de kullanmak mümkündür ancak GitHub pjreddie sayfasındaki kaynak kodları ile Windows'ta hızlı bir başlangıç mümkün değildir. Darknet'i Windows'ta kullanabilmek için GitHub'da AlexeyAB takma adlı kullanıcının pjreddie(Joseph Redmon) 'nin kaynak kodundan çatallaştırdığı(forking) kaynak kodu kullanılabilir.[30] AlexeyAB, pjreddie'nin kaynak kodunu Microsoft Visual Studio ile derlenebilir bir versiyonunu geliştirmiş ve Windows üzerinde de kullanılabilir hale getirmiştir. Ayrıca Joseph Redmon'ın sitesinde Darknet'e dair değinmediği birçok noktayı açıklamıştır.

Bu çalışmada pjreddie'nin kodu kullanılmış olup, AlexeyAB'nin paylaştığı bilgilerden de faydalanılmıştır.

2.3.1. Darknet'in Yüklenmesi ve Konfigürasyonu

Bu çalışma Ubuntu 18.04.2 LTS işletim sistemi üzerinde gerçekleştirilmiştir.

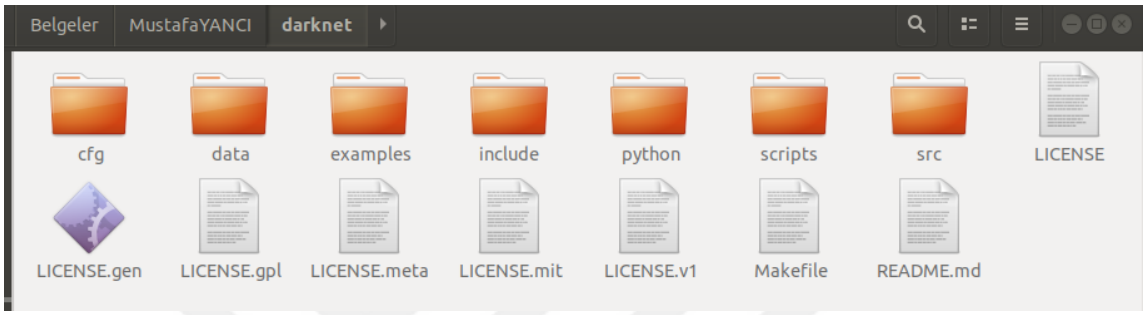
Darknet'i kurmak için önce GitHub'daki kaynak kodunun indirilmesi gerekmektedir. Herhangi bir dizinde aşağıdaki komut verilerek Darknet indirilebilir.

```
git clone https://github.com/pjreddie/darknet.git
```

```
mustafayanci@mustafayanci:~/Belgeler/MustafaYANCI$ git clone https://github.com/pjreddie/darknet.git
'darknet' dizinine çoğaltılıyor...
remote: Enumerating objects: 5901, done.
remote: Total 5901 (delta 0), reused 0 (delta 0), pack-reused 5901
Nesneler alınıyor: 100% (5901/5901), 6.16 MiB | 1.08 MiB/s, bitti.
Farklar çözülüyor: 100% (3922/3922), bitti.
mustafayanci@mustafayanci:~/Belgeler/MustafaYANCI$ cd darknet
mustafayanci@mustafayanci:~/Belgeler/MustafaYANCI/darknet$
```

Şekil 2. 6 Darknet'in GitHub'dan indirilmesi

Darknet indirildiğinde aşağıdaki gibi dosya ve dizinler gelmektedir.



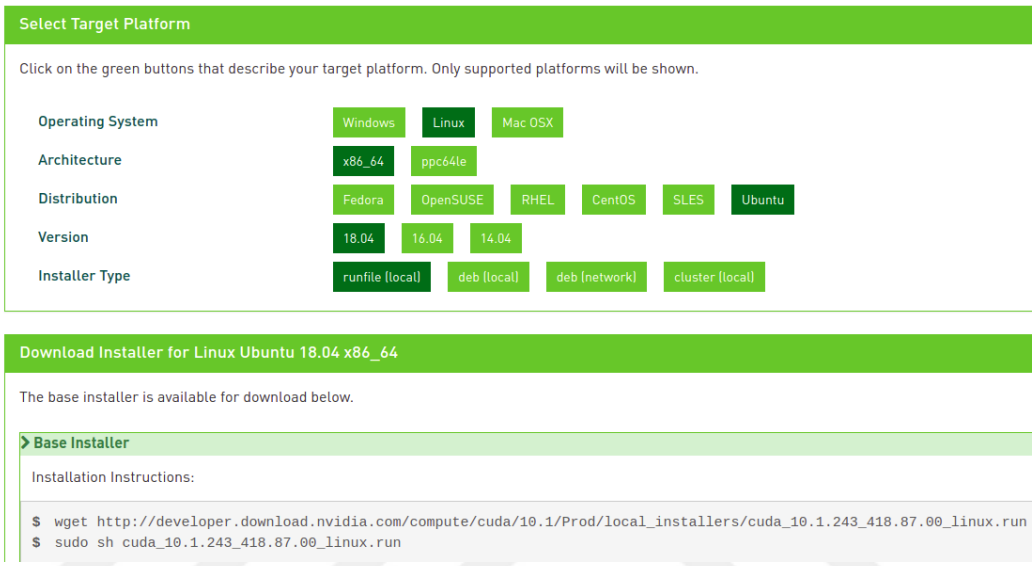
Şekil 2. 7 Darknet'in dosya ve dizinleri(derlenmemiş hali)

Eğer CPU üzerinde çalışılacaksa darknet dizinine girilerek make komutu ile kod derlenerek çalışmaya başlanabilir. Bu çalışmada CUDA destekli bir NVIDIA marka grafik kartı kullanılmıştır ve darknet'i derlemeden önce bazı ön hazırlıklar yapılmıştır. Bu ön hazırlıklar aşağıdaki gibidir.

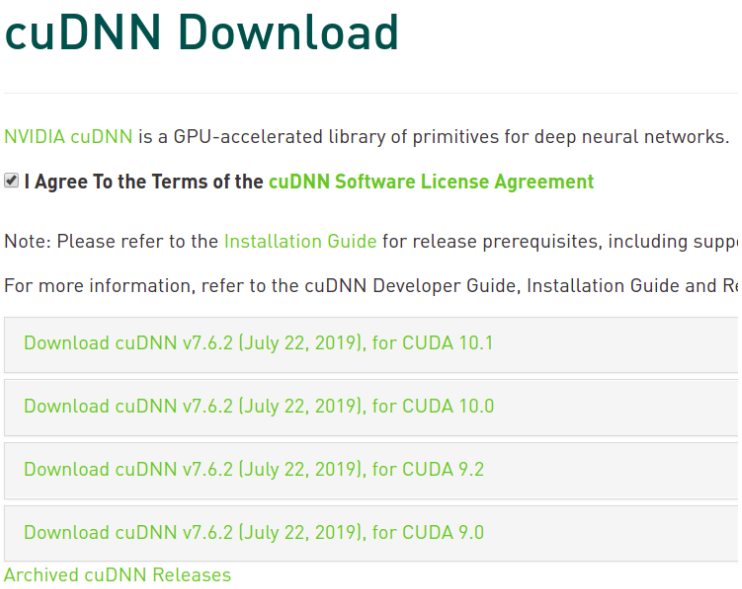
- Ubuntu üzerinde minimal bir grafik kartı sürücüsü mevcuttur. Grafik kartının özelliklerinden yararlanabilmek için güncel sürücünün yüklenmesi gerekmektedir. NVIDIA'nın sitesinden grafik kartının modeli seçilip, işletim sistemi olarak Linux 64-bit seçilirse "NVIDIA-Linux-x86_64-XXX.YY.run" isimli bir dosya inecektir. Bu çalışmada 418.56 versiyonu kullaılmıştır. Bu inen dosyayı "*sudo ./NVIDIA-Linux-x86_64-418.56.run --dkms -s*" şeklinde çalıştırarak sürücü yüklenir.

- Sürücü yüklendikten sonra CUDA yüklemek gereklidir. https://developer.nvidia.com/'dan şekil 2.8'de gösterildiği gibi gerekli bilgiler seçilerek kurulum dosyası indirilir ve sayfada belirtildiği şekilde kurulur.

- CUDA'yı da yükledikten sonra cuDNN yüklemek gerekmektedir. https://developer.nvidia.com/cudnn adresinden mevcutta yüklü olan driver ve CUDA ile uyumlu cuDNN versiyonu indirilir. (cuDNN indirme üye girişi gerektirmektedir.) Bu çalışma yapıldığı sırada 418.56 versiyon numaralı sürücü ve 10.1 versiyon numaralı CUDA yüklüydü. Uyumlu cuDNN versiyonu ise 7.6'dır.

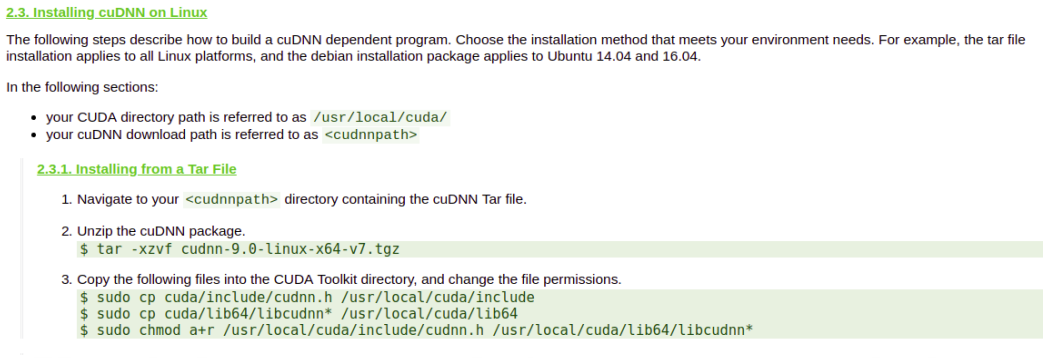


Şekil 2. 8 CUDA'nın indirilmesi ve kurulması



Şekil 2. 9 cuDNN indirme sayfası

İndirilen tar dosyası şekil 2.10'da gösterildiği şekilde açılır ve cuDNN kütüphane dosyaları CUDA'ya dahil edilir. (Nvidia'nın sitesinde yükleme CUDA 9.8 versiyonu için anlatılmıştır ancak 10.1 için de aynıdır.)



2.3. Installing cuDNN on Linux

The following steps describe how to build a cuDNN dependent program. Choose the installation method that meets your environment needs. For example, the tar file installation applies to all Linux platforms, and the debian installation package applies to Ubuntu 14.04 and 16.04.

In the following sections:

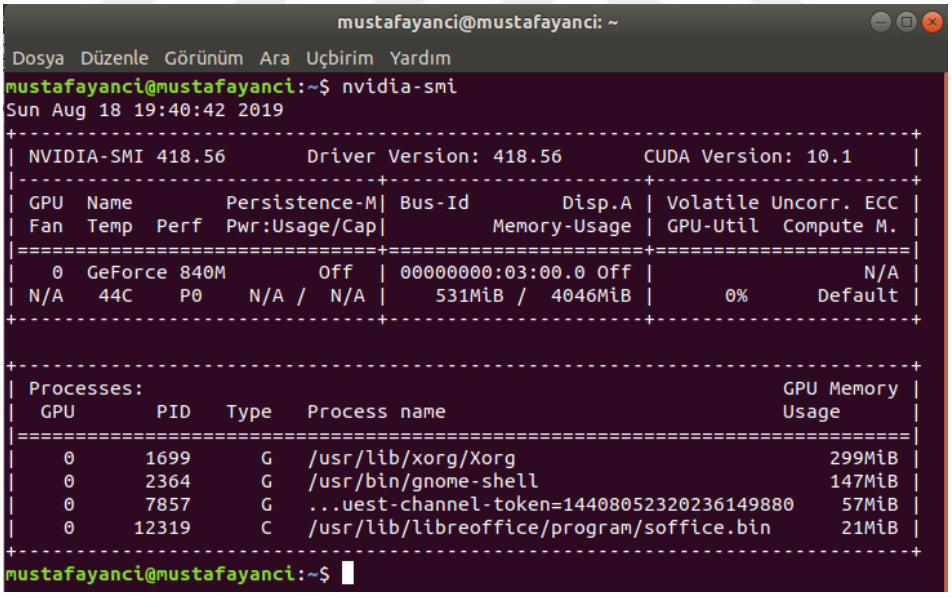
- your CUDA directory path is referred to as `/usr/local/cuda/`
- your cuDNN download path is referred to as `<cudnnpath>`

2.3.1. Installing from a Tar File

1. Navigate to your `<cudnnpath>` directory containing the cuDNN Tar file.
2. Unzip the cuDNN package.
```
$ tar -xzvf cudnn-9.0-linux-x64-v7.tgz
```
3. Copy the following files into the CUDA Toolkit directory, and change the file permissions.
```
$ sudo cp cuda/include/cudnn.h /usr/local/cuda/include
$ sudo cp cuda/lib64/libcudnn* /usr/local/cuda/lib64
$ sudo chmod a+r /usr/local/cuda/include/cudnn.h /usr/local/cuda/lib64/libcudnn*
```

Şekil 2. 10 cuDNN'in CUDA'ya eklenmesi

- Grafik kartı ile ilgili yüklemeler yapıldıktan sonra "nvidia-smi" komutu verilerek grafik sürücü ve CUDA sürümleri kontrol edilebilir. Bu komut ile grafik kartının anlık bellek kullanımı ve grafik kartı üzerindeki işlem yükleri de görüntülenebilir.

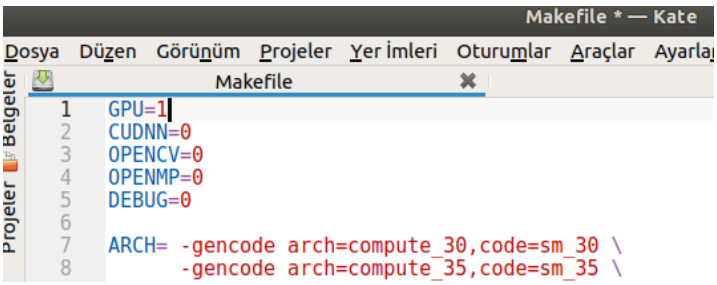


```
mustafayanci@mustafayanci: ~
Dosya  Düzenle  Görünüm  Ara  Uçbirim  Yardım
mustafayanci@mustafayanci:~$ nvidia-smi
Sun Aug 18 19:40:42 2019
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 418.56       Driver Version: 418.56       CUDA Version: 10.1     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  GeForce 840M        Off  | 00000000:03:00.0 Off |                  N/A |
| N/A   44C    P0    N/A /  N/A |    531MiB /  4046MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|    0      1699      G   /usr/lib/xorg/Xorg                           299MiB |
|    0      2364      G   /usr/bin/gnome-shell                         147MiB |
|    0      7857      G   ...uest-channel-token=14408052320236149880    57MiB |
|    0     12319      C   /usr/lib/libreoffice/program/soffice.bin      21MiB |
+-----------------------------------------------------------------------------+
mustafayanci@mustafayanci:~$
```

Şekil 2. 11 nvidia-smi ekranı

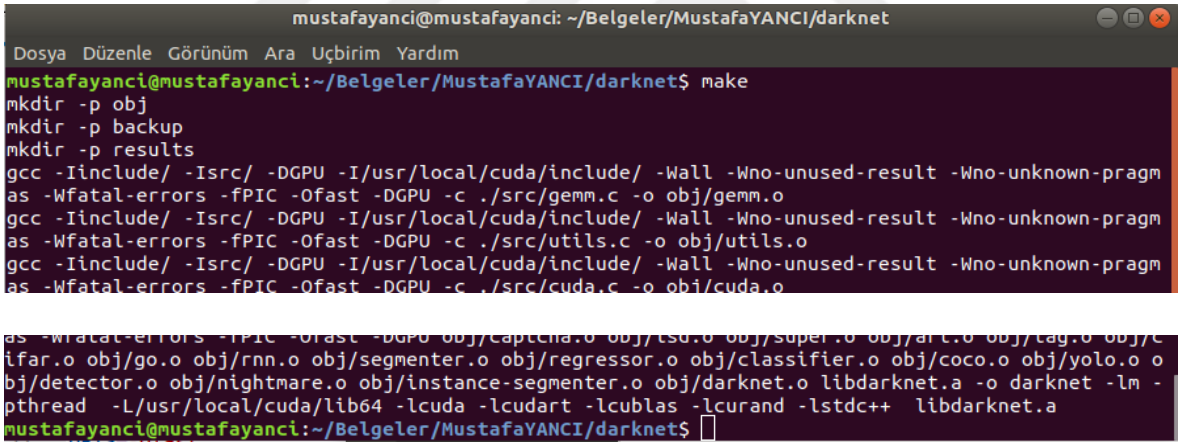
- Grafik kartı ile ilgili yüklemelerde bir problem yoksa darknet'i GPU'da çalışabilecek şekilde derlenebilir. Derleme işlemi yapmadan önce darknet dizininde bulunan MakeFile dosyasında şekil 2.12'de görüldüğü gibi ufak bir değişiklik yapılması gerekmektedir. Bu değişiklik GPU=0 olan satırı GPU=1 olarak değiştirmektir. MakeFile veya makefile şeklinde adlandırılan bu dosya C

ve C++ dillerinde kodlanmış projelerin derlenirken ihtiyaç duyacağı bilgileri saklamakla görevlidir. Unix ve Linux sistemlerin büyük çoğunluğunda bulunan make uygulaması bu dosyadaki konfigürasyonları okuyarak derlemeyi gerçekleştirir.



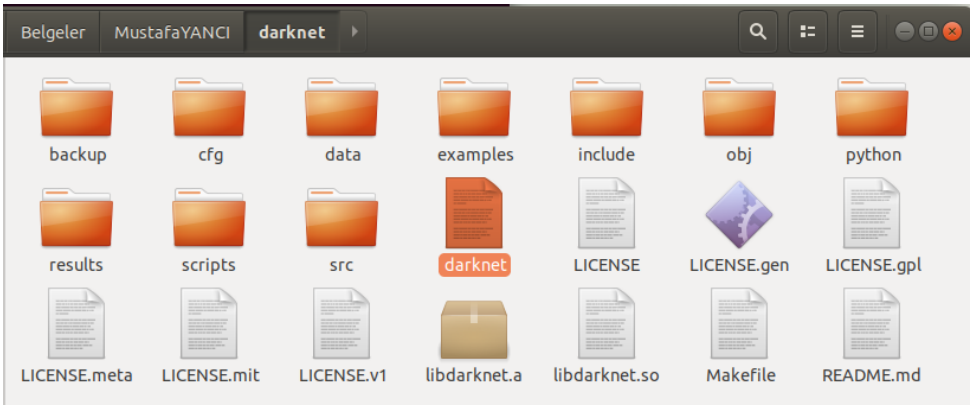
Şekil 2. 12 Darknet'in GPU'da çalışacak şekilde konfigüre edilmesi

GPU=1 şeklinde değiştirdikten sonra MakeFile dosyası kaydedilir ve darknet dizininde bir shell (komut satırı-kabuk) açılarak **make** komutu verilir. Make uygulaması projeyi GPU'da çalışacak şekilde derler. Bu işlem bir kaç dakika sürebilir. İşlem tamamlanana kadar beklenmelidir.



Şekil 2. 13 Darknet'in derlenmesi

Derleme işlemi tamamlandığında darknet dizini altında **darknet** adında çalıştırılabilir bir dosya oluşur. Darknet'in tüm fonksiyonları bu program ile çalıştırılır.



Şekil 2. 14 Darknet'in dosya ve dizinleri(derlenmiş hali)

Darknet doğru şekilde derlendiğinde "./darknet" komutu verildiğinde şekil 2.15'te gösterildiği şekilde bir cevap döner.

```
mustafayanci@mustafayanci:~/Belgeler/MustafaYANCI/darknet$ ./darknet
usage: ./darknet <function>
mustafayanci@mustafayanci:~/Belgeler/MustafaYANCI/darknet$
```

Şekil 2. 15 "./darknet" komut çıktısı

2.3.2. Darknet ile ilgili genel bilgiler

Darknet bir sinir ağı çatısıdır. Çeşitli konfigürasyonlarda sinir ağlarını çalıştırır. Çalıştıracağı sinir ağını .cfg uzantılı dosyalardan okur. Bu dosyalar çalışılacak sinir ağına ait katmanların parametrelerini içerir. Düzenli olması açısından .cfg dosyaları darknet/cfg dizini altında tutulur.

Darknet .cfg uzantılı dosyadan okuduğu sinir ağını .weights uzantılı ağırlık dosyalarını kullanarak çalıştırır. .weights uzantılı dosyalar sinir ağındaki nöronlar arasındaki katsayıları barındırır. Darknet'in sitesinde çeşitli konfigürasyonlara sahip sinir ağı .cfg dosyaları ve önceden öğretilmiş .weights dosyaları paylaşılmaktadır. Darknetin hem öğrenme hem de öğrenilene dayalı tespit işleminde sitede paylaşılan cfg ve weights dosyalarından faydalanmak ciddi bir işlem hızı kazandırmaktadır. Ancak kullanıcı isterse kendi özel konfigürasyonunu oluşturabilir, kendi verisi ile darkneti eğiterek yeni bir .weights dosyası oluşturabilir.

darknet/cfg dizini altında cfg dosyalarının yanısıra öğretilecek veri seti ile ilgili konfigürasyon bilgilerinin tutulduğu dosyalar da bulunur.

Darknet açık kaynak kodlu bir yazılımdır. Kodlarında değişiklikler yapılarak çeşitli fonksiyonlar kazandırılabilir, gerekli bulunmayan fonksiyonlar deaktif edilebilir.

darknet/data dizini altında kullanılan veri setleri ile ilgili konfigüratif bilgiler

Darknet'in birçok avantajı vardır. Bunlardan bazıları;

- Darknet öğrenme aşaması olmadan hazır cfg ve weight dosyaları ile nesne tespitine imkan sağlamaktadır. Yani derin öğrenmenin de ötesinde, derin öğrenilmiş bilgiyi hızlıca kullanmaya olanak tanır.
- Darknet kod yazmayı minimuma indiren bir sistemdir. Hiç kod yazmadan, sadece konfigürasyon dosyasını düzenleyerek (ki çoğunlukla buna bile ihtiyaç olmadan), öğretilecek veri setini hazırlayarak öğretmek mümkündür.

2.3.3. YOLO (You Only Look Once) : Gerçek Zamanlı Nesne Tespiti

Joseph Redmon ve arkadaşlarının geliştirdikleri YOLO, darknet ile çalışabilen bir sinir ağıdır ve nesne tespitinde yeni bir yaklaşımdır. [31]

İsmi "Sen Sadece Bir Kez Bak" şeklinde çevrilebilecek olan "You Only Look Once" ifadesindeki kelimelerin baş harfleri ile oluşturulmuştur. YOLO mevcut tespit sistemlerinden farklı bir yaklaşımla çalışmaktadır. Mevcut nesne tespit araçları bir görüntü üzerinde çeşitli işlemler gerçekleştirerek, sınıflandırıcı fonksiyonlarını tekrar tekrar kullanarak nesne tespiti gerçekleştirir. YOLO ise tek bir sinir ağı ile direkt tüm görüntü üzerinde hem sınırlama kutularını(bounding boxes) hem de sınıf olasılıklarını tahmin eder.

YOLO birleşik mimarisi sayesinde ileri derecede hızlıdır. YOLO'nun ilk versiyonu saniyede 45 kare (fps) işleyebilecek yetenektedir. Fast YOLO olan versiyonu ise saniyede 155 kare işleyebilmektedir. Bu sayesinde gerçek zamanlı nesne tespitinde çok başarılı bir sistemdir.

YOLO'nun 2016 yılında çıkan versiyonu YOLO9000 ise ilk versiyondan hem daha hızlı hem de daha çok nesneyi tespit edebilecek yetenektedir. YOLOv2 olarak da bilinen bu sistem, 9000 nesne kategorisini tespit edebilecek yetenekte olduğu için YOLO9000 olarak adlandırılmıştır. [32]

YOLO'nun güncel versiyonu olan YOLOv3 ise YOLO9000'a göre bazı değişiklikler yapılmış halidir.

Bu çalışmada YOLOv3 kullanılmıştır.

2.3.4. Veri Setinin YOLO için hazırlanması

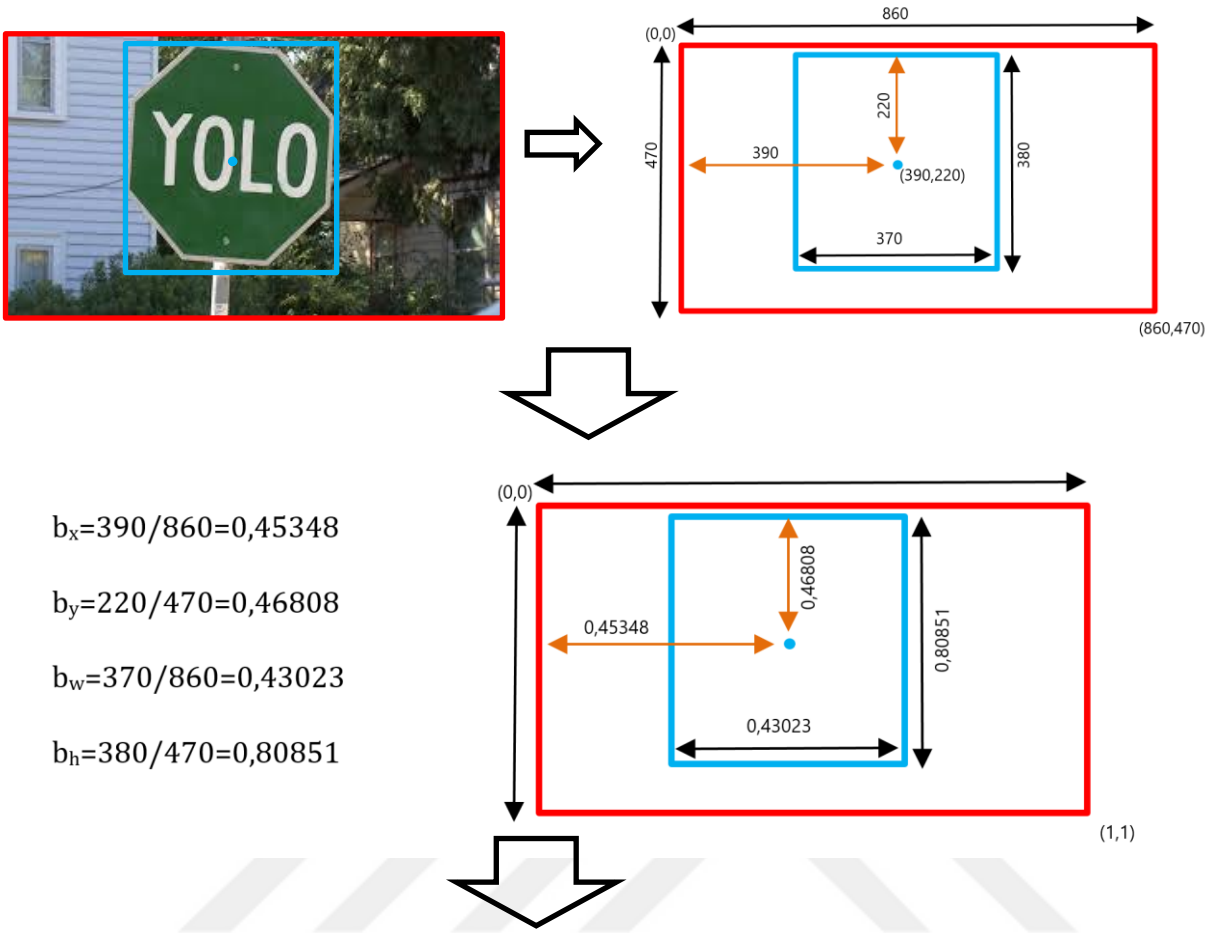
YOLO ile öğrenme için veri setinin YOLO'nun okuyabileceği formatta hazırlanması gerekmektedir. Mevcut birçok derin öğrenme sistemi görüntünün piksel değerleri ile ilgilenmektedir. YOLO ise bir görüntüde öğrenilecek nesnenin hangi piksellerde bulunduğu ile değil, görüntünün neresine, hangi bölgesinde bulunduğu ile ilgilenmektedir. YOLO öğrenilecek nesne ile ilgişi şu bilgilerin hepsini görüntü ile aynı anda almayı bekler.

<nesnenin sınıfı> <b_x> <b_y> <b_w> <b_h>

Burada nesnenin sınıfı,verisetindeki sınıf sınıf sayısında etiket sırasından 1 eksik tam sayı ile gösterilir. Örneğin, 8 sınıflı bir verisetinde, 6. sınıf YOLO formatında 5 ile, ilk sınıf 0 ile temsil edilir.

b_x ve b_y değerleri bir görüntü içindeki öğrenilecek nesnenin bir dikdörtgen içine alındığında bu dikdörtgenin merkez koordinatlarının tüm görüntüye göre (0,1) aralığında normalize edilmiş halidir. Genişlik b_w(genişlik) ve b_h(yükseklik) ise, bu dikdörtgenin genişlik ve yükseklik değerlerinin tüm görüntüye oranlanmış halinin (0,1) arasında normalize edilmiş halidir. Şekil 2.15'te bu yapı görsel olarak açıklanmıştır. Hesaplama formülleri ise şu şekildedir.

b_x=(nesnenin bulunduğu diktörgenin x değeri)/(bütün resmin genişliği) = x/w
b_y=(nesnenin bulunduğu diktörgenin y değeri)/(bütün resmin yüksekliği) = y/h
b_w=(nesnenin bulunduğu diktörgenin genişliği)/(bütün resmin genişliği) = w_i/w
b_h=(nesnenin bulunduğu diktörgenin yüksekliği/(bütün resmin yüksekliği) = h_i/h



Şekil 2. 16 Görüntüdeki nesnenin koordinat bilgilerinin YOLO'nun işleyebileceği formata dönüştürülmesi.

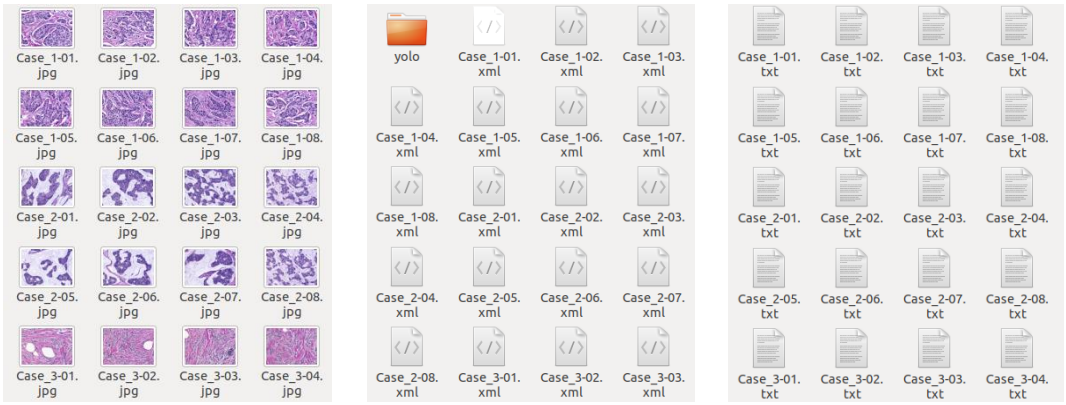
YOLO nesne ile ilgili bilgileri bu formatta ve büyük görüntünün dosya adıyla aynı isimde bir .txt dosyasına kaydedilmiş olarak beklemektedir. Bir görüntü içinde birden fazla nesne varsa hepsi için bu bilgiler aynı dosya içinde olmalıdır.

YOLO'daki bu formatın en büyük avantajı, öğrenilecek görüntünün çözünürlüğü değişse bile, nesnenin görüntünün neresinde bulunduğu bilgisi değişmeyecek olmasıdır. Özellikle çok fazla imaj ile çalışıldığında, imajların çözünürlükleri yüksek olduğunda GPU'da bellek dolması problemi ile karşılaşılabilmektedir. YOLO'da bu problemi aşmak için öğrenme öncesi görüntüleri yeniden ölçeklendirme yapılmaktadır. Bu format sadece çözünürlüğün değişmesinde değil, görüntünün en boy oranı değiştiğinde bile bilginin kaybolmaması açısından önemlidir.

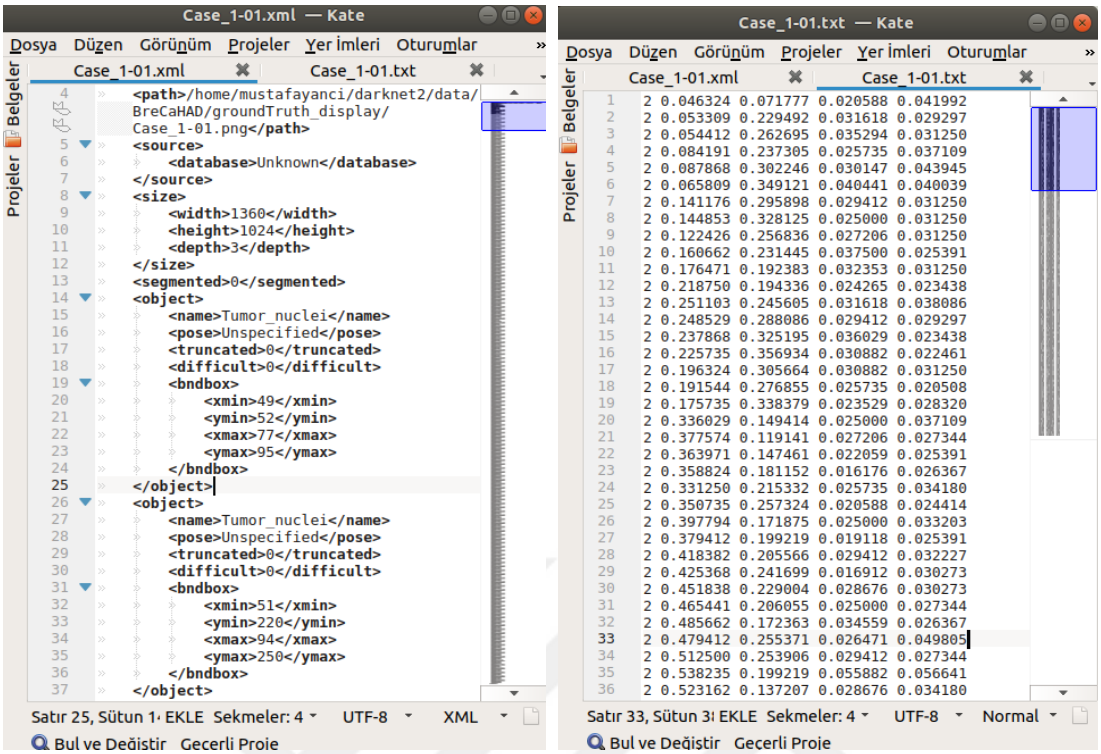
Çalışmada kullanılan veri setinde hücrelerin koordinatları piksel cinsinden verilmiştir ve hücrelerin sadece çekirdekleri işaretlenmiştir. JSON formatındaki etiket bilgilerinde ise sadece çekirdek koordinatları YOLO formatında kayıtlıdır ancak en-boy bilgisi yoktur.

Burada veri setini YOLO'nun beklediği formata çevirmek için Python (v3) dilinde yazılmış olan labelImg adlı uygulama kullanılmıştır[34]. Bu uygulama imajların etiketlerini hem en çok kullanılan nesne veri setlerinden biri olan Pascal VOC veri setinin formatı olan XML formatında, hem de YOLO formatında kaydedebilmektedir. Mevcutta bu formatlardan sadece birinde kayıt yapmakta iken, kodlarında ufak değişiklikler yapılarak BreCaHAD verisetinin iki formatta da (hem XML hem YOLO) etiket dosyaları oluşturulmuştur. (Şekil 2.16)

Uygulama kullanılırken önce çekirdekleri renkli nokta ile işaretlenmiş görüntülerin bulunduğu dizin uygulamada açıldı. Daha sonra renklere göre hücreler dikdörtgen içine alınarak sınıfı belirlendi. Etiketlenen her bölgenin etiketi uygulama tarafından ilgili etiket dosyalarına otomatik olarak kaydedilmektedir. Veri setini etiketleme işlemi bilgisayar faresi ile çok zor olduğu ve uzun sürdüğü için bir çizim tableti kullanılarak etiketleme işlemi gerçekleştirilmiştir. Veri setindeki yaklaşık 23500 hücre bu şekilde etiketlenmiştir.



Şekil 2. 17 BreCaHAD Veri setinin görüntüleri ile VOC(.xml) ve YOLO (.txt) formatındaki etiket dosyaları



Şekil 2. 18 BreCaHAD Veri setinin VOC(.xml) ve YOLO (.txt) formatındaki etiket yapısı

2.3.5. YOLO'nun BreCaHAD veri seti ile eğitilmesi

Eğitim(training) işlemi başlatılmadan önce aşağıdaki adımlar uygulanmıştır.

- Veri seti darknet/data dizini altına kopyalandı. YOLO formatında hazırlanan etiket dosyaları görüntü dosyaları ile aynı dizine kopyalanmıştır. (YOLO görüntü ile birlikte aynı isme sahip txt dosyasını da beklemektedir.)

- Veri seti öğrenme ve test verisi olarak rastgele bölünmüştür Verinin %90'ı öğrenme verisi olacak şekilde ayarlanıp öğrenme ve test verilerine ait dosya yolları train.txt ve test.txt dosyalarına alınmıştır. (Şekil 2.17)

- brecahad.data adında veri tanımlama dosyası hazırlanmıştır ve darknet/cfg altına kopyalanmıştır. (Şekil 2.18) Veri seti tanımlama dosyası (brecahad.data) hazırlanmıştır. Bu dosyada veri setindeki sınıf sayısı, öğrenme ve test verilerinin bilgileri, veri setinde kaç sınıf olduğu bilgisi ve backup dizin yolunun tanımını içermektedir. Backup dizini öğrenme sırasında belli sayıda veri öğrenildikçe (iteration-epoch) o ana kadar öğrenilen veriye dair ağırlık dosyası(.weigths) nın bir yedeğinin alındığı dizindir. Varsayılan olarak darknet/backup dizini verilmiştir. Özel bir dizin verilmek isteniyorsa, dizinin okuma-yazma yetkileri kontrol edilmelidir.

train.txt

```
1  /home/mustafayanci/darknet/data/BreCaHAD/JPEGImages/Case_1-05.jpg
2  /home/mustafayanci/darknet/data/BreCaHAD/JPEGImages/Case_1-08.jpg
3  /home/mustafayanci/darknet/data/BreCaHAD/JPEGImages/Case_2-01.jpg
4  /home/mustafayanci/darknet/data/BreCaHAD/JPEGImages/Case_2-05.jpg
5  /home/mustafayanci/darknet/data/BreCaHAD/JPEGImages/Case_2-06.jpg
6  /home/mustafayanci/darknet/data/BreCaHAD/JPEGImages/Case_3-02.jpg
7  /home/mustafayanci/darknet/data/BreCaHAD/JPEGImages/Case_3-04.jpg
8  /home/mustafayanci/darknet/data/BreCaHAD/JPEGImages/Case_3-06.jpg
9  /home/mustafayanci/darknet/data/BreCaHAD/JPEGImages/Case_3-07.jpg
10 /home/mustafayanci/darknet/data/BreCaHAD/JPEGImages/Case_3-08.jpg
```

Şekil 2. 19 Öğrenme veri listesi

brecahad.data

```
classes= 6
train  = data/BreCaHAD/train.txt
valid  = data/BreCaHAD/test.txt
names = data/brecahad.names
backup = backup/
```

Şekil 2. 20 brecahad.data dosyası

- Veri setindeki sınıfların etiketlerinin bulunduğu brecahad.names adlı dosya darknet/data dizini altında oluşturulmuştur. (Şekil 2.19)

brecahad.names

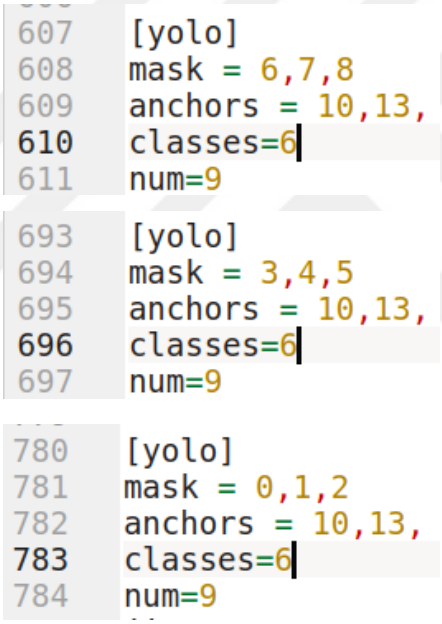
```
1  Mitosis
2  Apoptosis
3  Tumor_nuclei
4  Non_tumor_nuclei
5  Tubule
6  Non_tubule
7
```

Şekil 2. 21 brecahad.names dosyası

- Konfigürasyon dosyası olarak darknet ile birlikte gelen yolov3.cfg dosyasının bir kopyası oluşturulup yolov3_obj.cfg adıyla kaydedildi. Darknet'in sitesinde paylaşılan yolov3.weights dosyası indirildi.

Öğrenme(training) işlemi başlamadan önce yolov3_obj.cfg dosyasında bazı düzenlemeler yapılmıştır. Bu düzenlemeler AlexeyAB'nin GitHub'daki darknet sayfasından alınmıştır. [30] Yapılan düzenlemeler şu şekildedir;

- Sınıf sayısı düzenlenmiştir. Mevcut yolov3.cfg dosyası 80 sınıflı VOC veri seti ile eğitilerek yolov3.weight dosyası oluşturulmuştur. Bizim veri setimizde ise 6 sınıf bulunmaktadır. Bu yüzden yolov3_obj.cfg dosyasındaki tüm yolo katmanlarının classes parametresi 6 olarak değiştirilmiştir. (Şekil 2.20)



Şekil 2. 22 YOLO katmanlarında sınıf sayısının değiştirilmesi

- Tüm yolo katmanlarından önce gelen konvolüsyonel katmanlarda filters parametresi 33 olarak düzenlenmiştir. (Şekil 2.21) Bu sayı filters = (classes + 5) x 3 formülü ile hesaplanmıştır. Bu formül YOLOv3'e özel olup, daha önceki yolo

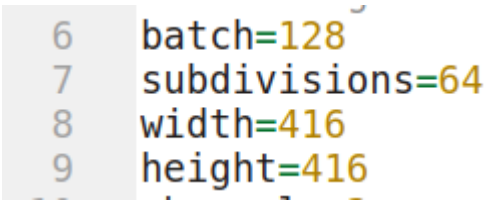
versiyonlarında farklıdır. Bu çalışmadaki veri setinde 6 sınıf olduğu için filters = (6+5)x3=33 olarak hesaplanmıştır.

```
599  [convolutional]
600  size=1
601  stride=1
602  pad=1
603  filters=33
604  activation=linear

685  [convolutional]
686  size=1
687  stride=1
688  pad=1
689  filters=33
690  activation=linear

772  [convolutional]
773  size=1
774  stride=1
775  pad=1
776  filters=33
777  activation=linear
```

Şekil 2. 23 YOLO katmanlarından önce gelen konvolüsyonel katmanlarda filters parametresinin değiştirilmesi

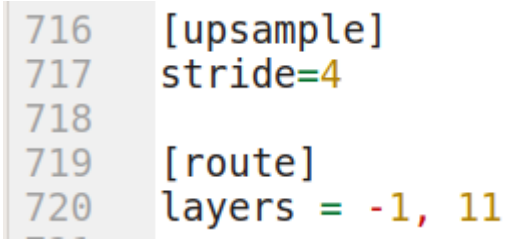
- Katmanlardan önceki en baştaki konfigürasyonlardan width ve height değerleri 416 olarak düzenlenmiştir. Bu ayar, tüm görüntüleri hangi çözünürlükte olursa olsun 416x416 çözünürlükte kare imaja çevirmektedir. [0,1] aralığında normalize edilen boyut bilgileri sayesinde çözünürlük ve en-boy oranı değişse bile nesneye ait bilgi (nesnenin resmin bütününe göre nerede bulunduğu, resmin bütününe oranı vs.) kaybolmamaktadır. Aynı zamanda batch parametresi 128, subdivisions parametresi 64 olarak ayarlandı. (Şekil 2.21) Bu sayede darknet her döngüde 128 görüntü alacak ve her görüntüyü 64 parçaya bölerek inceleyecektir. Bu iki ayar optimum değerler olmayıp, bilgisayarın grafik kartının bellek boyutuna göre değiştirilebilir. 4 GB belleğe sahip NVIDIA GeForce 840M GPU'ya sahip bir dizüstü bilgisayarda bu ayarlarla sorunsuz çalışılmıştır.
- Aynı kısımda learning_rate (öğrenme oranı) değeri de belirlenmelidir. Bu değer YOLO'nun öğrenirken veride yaşadığı kaybı gösterir. Öğrenme süresince bu değer çok düşük iken, öğrenme başarılı hale geldikçe bu değer yükselerek

belirlediğimiz değere ulaşmaya çalışır. YOLO bir nesne tespit aracıdır ve günlük hayatta tespiti kolay olan nesneler için bu değer yüksek (örneğin 0.01 gibi) belirlenebilir. Ancak kanser hücrelerinin şekli günlük hayattaki kedi, araba, tabak gibi nesnelere göre daha kompleks olduğundan 0.01 gibi bir değer çok yüksektir. Bu çalışmada learning_rate değeri, 0.001 olarak belirlenmiştir.



Şekil 2. 24 Katmanlardan önce işlenecek görüntülerin hangi çözünürlüğe ölçekleneceğinin, her döngüde kaç imaj işlenip her imajın kaça bölüneceğinin ayarlanması

- Girişe verilen görüntü 416x416 çözünürlüğe ölçeklendiğinde tespit edilecek nesnelerin boyutu eğer 16x16 pikselden küçükse çok ince bir ayar daha yapılmalıdır. Cfg dosyasında son yolo katmanından önce gelen upsample katmanının stride değeri 4, route katmanının layers parametresi ise -1, 11 olarak değiştirilmelidir[30]. Bu parametreler yolov3.cfg dosyasında ve bizim kopyalayıp konfigüre edilen yolov3_obj.cfg dosyasında 717. Ve 720. satırlara denk gelmektedir. Bu ayar yapılmadan çalıştırılan öğrenme sonrasında YOLO bu veri setinde hiç bir tespit yapamamıştır.



Şekil 2. 25 YOLO'nın çok küçük şekilleri yakalayabilmesi için upsample ve route katmanlarında değişiklik yapılması

Bu ayarlar yapıldıktan sonra darknet dizininde bir shell açılarak aşağıdaki şekilde öğrenme başlatılmıştır.

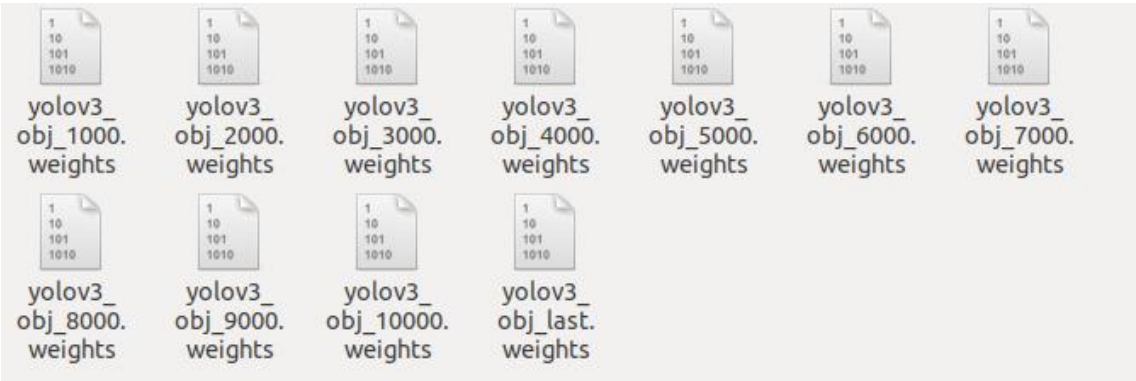
sudo ./darknet detector train cfg/brecahad.data cfg/yolov3_obj.cfg darknet53.conv.74

```
mustafayanci@mustafayanci:~/darknet1$ ./darknet detector train cfg/brecahad.data cfg/yolov3_brecahad.cfg yolov3.weights
yolov3_brecahad
layer     filters    size              input                output
    0 conv     32  3 x 3 / 1   416 x 416 x   3   ->   416 x 416 x  32  0.299 BFLOPs
    1 conv     64  3 x 3 / 2   416 x 416 x  32   ->   208 x 208 x  64  1.595 BFLOPs
    2 conv     32  1 x 1 / 1   208 x 208 x  64   ->   208 x 208 x  32  0.177 BFLOPs
    3 conv     64  3 x 3 / 1   208 x 208 x  32   ->   208 x 208 x  64  1.595 BFLOPs
    4 res    1                 208 x 208 x  64   ->   208 x 208 x  64
    5 conv    128  3 x 3 / 2   208 x 208 x  64   ->   104 x 104 x 128  1.595 BFLOPs
    6 conv     64  1 x 1 / 1   104 x 104 x 128   ->   104 x 104 x  64  0.177 BFLOPs
    7 conv    128  3 x 3 / 1   104 x 104 x  64   ->   104 x 104 x 128  1.595 BFLOPs
    8 res    5                 104 x 104 x 128   ->   104 x 104 x 128
    9 conv     64  1 x 1 / 1   104 x 104 x 128   ->   104 x 104 x  64  0.177 BFLOPs
   10 conv    128  3 x 3 / 1   104 x 104 x  64   ->   104 x 104 x 128  1.595 BFLOPs
   11 res    8                 104 x 104 x 128   ->   104 x 104 x 128
   12 conv    256  3 x 3 / 2   104 x 104 x 128   ->    52 x  52 x 256  1.595 BFLOPs
   13 conv    128  1 x 1 / 1    52 x  52 x 256   ->    52 x  52 x 128  0.177 BFLOPs
   14 conv    256  3 x 3 / 1    52 x  52 x 128   ->    52 x  52 x 256  1.595 BFLOPs
   15 res   12                   52 x  52 x 256   ->    52 x  52 x 256
   16 conv    128  1 x 1 / 1    52 x  52 x 256   ->    52 x  52 x 128  0.177 BFLOPs
   17 conv    256  3 x 3 / 1    52 x  52 x 128   ->    52 x  52 x 256  1.595 BFLOPs
   18 res   15                   52 x  52 x 256   ->    52 x  52 x 256
```

Şekil 2. 26 YOLO eğitim aşaması

Öğrenme süresi bilgisayar konfigürasyonuna göre birkaç gün sürebilir

Öğrenme süresince, belirli sayıda görüntü işlendikten sonra darknet/backup dizini altına .weights dosyaları atmaya başlar. Bu dosyalar o döngüye kadar öğrenilmiş verilere ait ağırlık dosyalarıdır. Öğrenme süresince atılan bu .weights dosyalarından hangisinin en başarılı ağırlık dosyası olduğuna deneyerek karar vermek mümkün. Ancak .cfg dosyamızda belirlediğimiz learning_rate parametresine ulaşıldığında last.weights dosyası oluşmaktadır ve öğrenme bitmektedir. Bu çalışmada /yolov3_obj_6000.weights dosyası kullanılarak başarılı tespitler yapılmıştır. Diğer .weights dosyaları da tespit yapabilmektedir ancak en başarılı sonuçlar 9000 döngüde öğrenilebilmiştir.



Şekil 2. 27 YOLO eğitim sırasında atılan .weights dosyalarından birkaçı

3. DENEYSEL ÇALIŞMALAR VE BULGU

Bu çalışmada bir yapay sinir ağı çatısı(framework) olan darknet ve gerçek zamanlı nesne tespit aracı olan YOLO, medikal görüntüler üzerinde kanserli hücre tespiti yapmak amacıyla kullanılmıştır. Veri seti olarak meme kanseri hastalarından alınan biyopsi görüntüleri ile oluşturulmuş BreCaHAD veri seti kullanılmıştır. Veri seti YOLO'nun okuyacağı formatta olmadığı için tüm veriler YOLO formatında tekrar etiketlenmiştir.

YOLO ile derin öğrenme yapmadan önce darknet'in okuyacağı sinir ağı konfigürasyon dosyasında bazı değişiklikler yapılarak, sinir ağının küçük nesneleri de tespit edebilecek hale gelmesi sağlanmıştır.

YOLO'nun eğitilmesi GPU üzerinde gerçekleştirilmiştir. Böylelikle zamandan ciddi kazanç sağlandı. Aynı veri seti ile CPU üzerinde çalışılsaydı, öğrenmenin bitmesi çok uzun sürebilirdi.

3.1. Çalışmada Kullanılan Bilgisayar hakkında genel bilgiler

Bu çalışmada YOLO, 4GB 128 bit veri yoluna sahip NVIDIA GeForce 840M GPU, Intel i5 4210U model işlemciye sahip, Lenovo marka bir dizüstü bilgisayar kullanılmıştır. Bilgisayar üzerine dual-boot özelliği ile Windows'un yanına Ubuntu 18.04.2 LTS kurulmuş ve çalışma Ubuntu üzerinde gerçekleştirilmiştir. Öğrenme aşaması çok daha iyi GPU'ya sahip bir bilgisayar üzerinde gerçekleştirilebilir ve çıkan .weights dosyaları başka bilgisayarlar üzerinde tespit yapma amacıyla kullanılabilir

3.2. Öğrenme Sonrası YOLO ile Hücre Tespit İşlemi

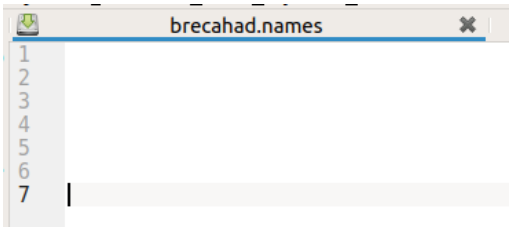
Öğrenme tamamlandıktan sonra YOLO eğitilirken kullanılan dosyalar brecahad.data ve yolov3_obj.cfg dosyaları ile öğrenme sonrası oluşan .weight dosyası ile tespit işlemi yapılabilir. Tespit için aşağıdaki şekilde komut verilebilir.

./darknet detector test <.data dosyası> <.cfg dosyası> <.weights dosyası> <görüntü>

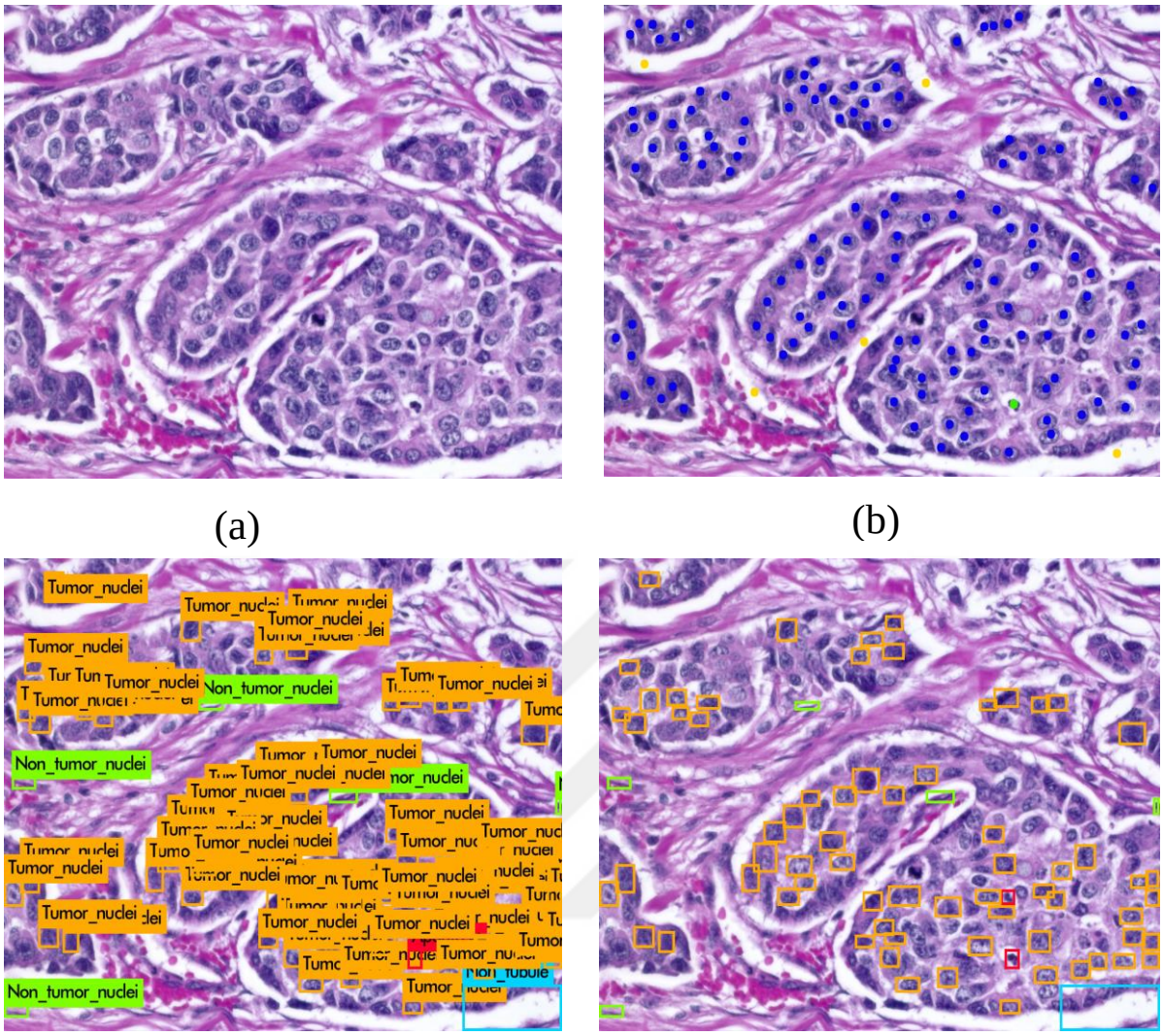
```
./darknet detector test cfg/brecahad.data cfg/yolov3_brecahad.cfg  backup/yolov3_brecahad_last.weights  data/BreCaHAD/JPEGImages/Case_3-03.jpg
```

Komut çalıştığında darknet dizini altında predictions.jpg adında bir dosyaya tespit edilen edilen görüntüler işaretlenmiş şekilde atılır.

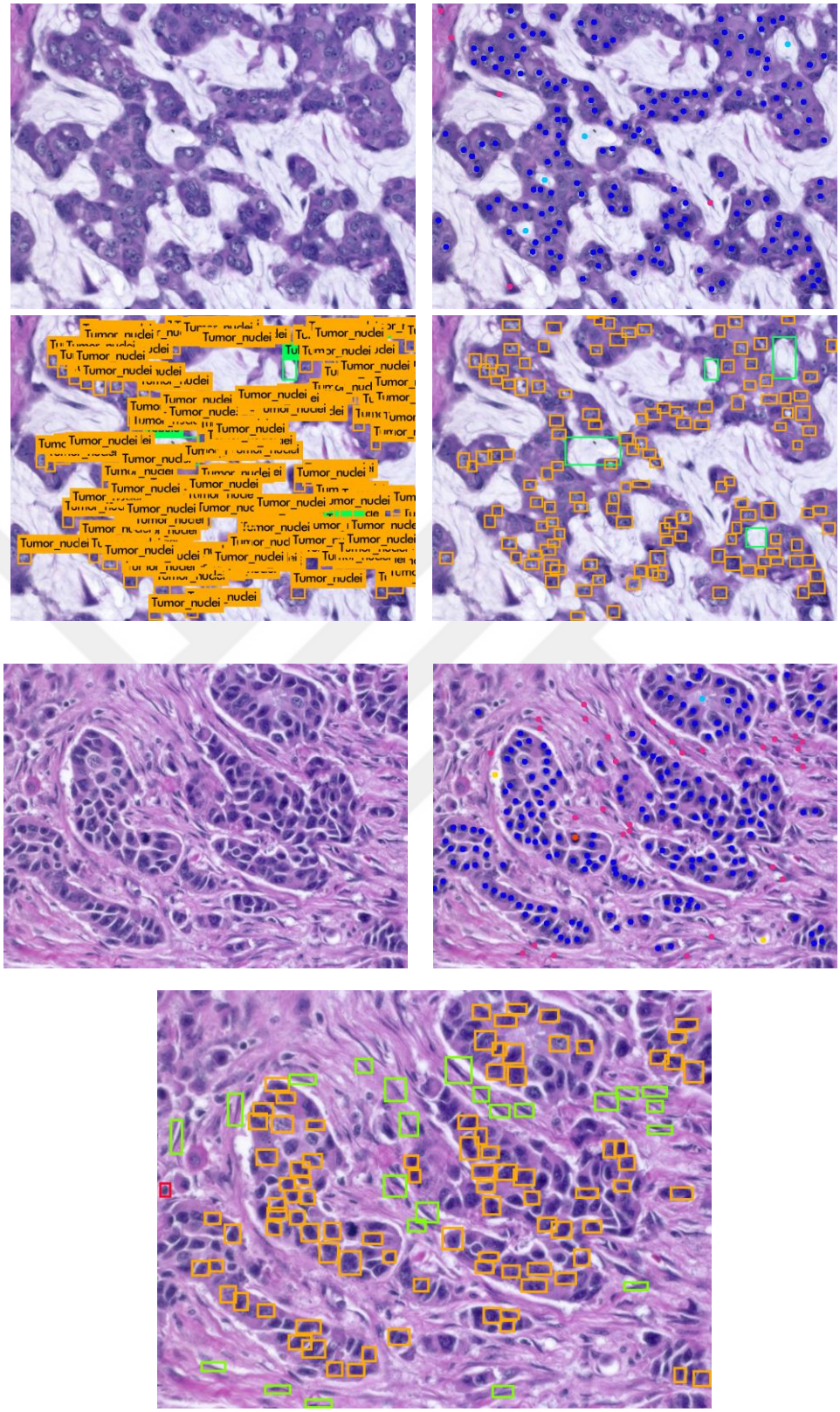
Şekil 2.26’da kullanılan görüntü ve tespit edileni görülmektedir. Kanserli hücrelerin görüntü üzerinde çok fazla olmasından dolayı, işaretlemenin yanında etiketinin de olması çirkin bir görüntü oluşturmaktadır. Bu yüzden tespit işleminde kullanılan brecahad.names dosyasındaki etiketler silindi ve 6 tane boş satır eklendi ve tekrar test yapıldı Bu sefer şekil 2.26.d’deki gibi daha düzgün bir görüntü elde edildi.



Şekil 3. 1 Boş brecahad.names dosyası



Şekil 3. 2 (a) YOLO ile üzerinde tespit yapılacak görüntü (b) Aynı görüntünün veri setinde çekirdek merkezi işaretlenmiş kopyası (c) YOLO'nun tespit ettiği hücreler (etiketli) (d) YOLO'nun tespit ettiği hücreler (etiketsiz)



Şekil 3. 3 Veri setinden örnekler ve YOLO ile tespit edilen hücreler

Bu deneyde eğitim **darknet53.conv.74** dosyası ile gerçekleştirilmiştir. Ancak tespit işlemi öğrenme sonrası oluşan yeni .weights dosyaları ile gerçekleştirilmiştir. Bu dosya artık 6 tane sınıfa göre eğitilmiş, sadece bu veri setindeki sınıflara ait görüntüler üzerinde çalışabilecek bir ağırlık dosyasıdır. Bundan sonra bu tür görüntülerde veri setindeki sınıflardan biri aranıyorsa son .weights dosyası kullanılarak hızlıca tespit yapılabilir.

Eğer bu sistemin yeni bir hücre türünü daha tanıması istenirse, veri setine yeni hücre türüne ait görüntüler etiketleri ile eklenip, son oluşan yani öğrenmiş .weights dosyası ile tekrar öğrenme başlatılabilir. Bu özellik sayesinde en baştan öğrenme yerine öğrenilmişi geliştirebilmek mümkün olmaktadır.

3.2.1. Average Precision & Mean Average Precision

Derin öğrenmede öğrenme başarısı Average Precision değeri üzerinden hesaplanmaktadır. Bu skala için bazı temel kavramlar vardır. Bu kavramlardan dördü gerçekleşen öğrenmeyi test verisi üzerinde uygulandığında elde edilen tahmini sınıflandırmak için kullanılmaktadır. Bu kavramları kanser hücresi tespiti üzerinden yorumlayarak, test verisinin tespitinin kanser olup olmamasına göre şöyle açıklanabilir.

Kanserli bölge pozitif, kansersiz bölge negatif kabul edilirse;

- ***True Positive(Doğru Pozitif - TP) :***Bölge pozitif, sistem pozitif sınıfta tespit etmiş. Yani kanserli hücreyi kanserli olarak tespit etmiş.
- ***False Positive(Yanlış Pozitif - FP) :*** Bölge pozitif, sistem negatif sınıfta tespit etmiş. Yani kanserli hücreyi kanserli değil olarak tespit etmiş.
- ***True Negative(Doğru Negatif - TN) :*** Bölge negatif, sistem negatif sınıfta tespit etmiş. Yani kanserli olmayan hücreyi kanserli değil olarak tespit etmiş.
- ***False Negative(Yanlış Negatif – FN) :*** Bölge negatif, sistem pozitif sınıfta tespit etmiş. Yani kanserli olmayan hücreyi kanserli olarak tespit etmiş.

Bu terimler kullanılarak her bir örnek için Precision(hassasiyet) ve Recall(geri çağırma) değerleri hesaplanır. Formülleri aşağıdaki gibidir.

$$Recall = r = \frac{Doğru\ Pozitifler}{Tespit\ edilen\ kanserli\ hücre\ sayısı} = \frac{TP}{TP + FN}$$

$$Precision = p = \frac{Doğru\ Pozitifler}{Toplam\ pozitif\ sonuçlar} = \frac{TP}{TP + FP}$$

Recall ve Precision değerleri yüzdelik olarak hesaplanmaktadır. Yani değerleri 0-1 arasındadır. Test verisindeki her bir örnek için bir test çalıştırılır ve elde edilen sonuca göre recall(r) ve precision(p) değerleri hesaplanır. Bu değerlere göre bir r-p grafiği oluşur. r-p grafiği altında kalan alan Average Precision(Ortalama hassasiyet - AP) olarak hesaplanmaktadır. Bu alan aşağıdaki formül ile hesaplanır.

$$AP = \int_0^1 p(r)dr$$

Bu formül pratikte sonlu bir toplam olarak aşağıdaki şekilde kullanılır.

$$AP = \sum_{k=1}^{n} P(k).\Delta r(k)$$

n : nesne sayısı

P(k) : Her bir k değeri için precision

Δr(k) : Her bir k değeri için recall değerindeki değişim.

Her bir sınıf için AP hesabı yapıldıktan sonra tüm sınıflar için Mean Average Precision(mAP) değeri şu şekilde hesaplanır.

$$mAP = \frac{\sum_{i\ \in\ sınıflar} AP_i}{Toplam\ Sınıf\ Sayısı}$$

3.3. YOLO'nun Başarısı ve Çalışmaya Dair Sonuçlar

Bu çalışmada YOLO ile yapılan öğrenme işleminde veri seti %90 öğrenme ve %10 test verisi olarak ayrılmıştır. Tablo 3.1'de öğrenme ve test verisindeki hücre tiplerinin sayısal dağılımı görülmektedir.

Tablo 3. 1. Veri setindeki öğrenme ve test verisi dağılımı

Hücre Tipi	**Öğrenme verisi**	**Test Verisi**	**Veri Seti Toplam**
Mitosis	103	9	112
Apoptosis	237	26	263
Tumor_nuclei	17924	2193	20117
Non_tumor_nuclei	1640	277	1917
Tubule	407	79	486
Non_tubule	547	54	601
TOPLAM	**20858**	**2638**	**23496**

Öğrenmenin başarısını ölçmek için aşağıdaki komut öğrenilen .weights dosyası ile çalıştırılmalıdır.

./darknet detector test <.data dosyası> <.cfg dosyası> <öğretilmiş .weights dosyası>

Derin öğrenme ile tespitte başarılı sonuç elde etmeyi etkileyen birçok faktör vardır. Bunlardan en önemlisi ise veri setidir. Bir sistemin bir nesneyi ne kadar iyi öğrenip tanıması isteniyorsa o kadar çok örnek ile beslenmelidir. Özellikle de kompleks görüntülerin tespiti bekleniyorsa, sistem çok fazla görüntü ile öğretilmelidir.

Bu çalışmada YOLO'nın ürettiği .weights dosyalarında en yüksek başarı 6000-10000 tekrarlar(iteration) arasında elde edilmiştir.

Tablo 3.2'de 1000-10000 tekrar sonrasında üretilen .weights dosyalarının başarıları görülmektedir. Burada başarı AP(Average Precision – ortalama hassasiyet) olarak hesaplanmıştır.

Tablo 3. 2. YOLO'nun başarısı

TP:True Positive FP : False Positive AP : Average Precision mAP : Mean Average Precision

Tekrar (Iteration)	Mitosis			Apoptosis			Tumor_nuclei			Non_tumor_nuclei			Tubule			Non_tubule			mAP(%)
	AP(%)	TP	FP	AP(%)	TP	FP	AP(%)	TP	FP	AP(%)	TP	FP	AP(%)	TP	FP	AP(%)	TP	FP	IoU:0.50
1000	0,21	0	0	1,95	0	0	28,62	205	76	6,94	2	3	65,2	6	1	19,97	0	0	20,48
2000	0,79	0	0	11,49	0	2	67,64	1753	1576	26,98	32	39	**78,71**	33	8	29,53	2	2	35,86
3000	0,63	0	0	12,58	0	0	61,04	415	97	19,08	2	1	**75,37**	8	1	25,08	1	0	32,30
4000	1,83	0	0	9,62	0	0	**70,00**	988	261	28,86	11	3	**81,63**	29	3	26,21	1	0	36,36
5000	3,42	0	0	9,60	0	1	**72,07**	1205	384	32,28	23	8	**80,88**	35	4	29,19	1	1	**37,91**
6000	5,34	0	0	15,32	0	0	**73,87**	1787	1132	35,01	34	26	**79,71**	37	8	25,15	4	3	**39,07**
7000	2,45	0	0	7,12	0	0	**67,88**	957	268	26,12	4	2	**81,65**	26	4	23,13	1	0	34,72
8000	1,45	0	1	11,18	0	3	**69,31**	1844	1724	27,88	62	77	**77,79**	46	13	19,56	3	5	34,53
9000	4,37	0	0	17,28	1	1	**75,26**	1783	990	35,4	71	66	**82,55**	39	8	21,28	4	6	**39,36**
10000	6,27	0	1	12,44	0	3	**73,12**	1617	801	33,13	42	33	81,41	37	4	22,26	3	4	**38,10**

mAP(Mean Average Precision – Ortalama hassasiyetlerin ortalaması) değeri tüm hücre tipleri üzerinden hesaplandığında **%35-40** arasında çıkmaktadır. Ancak burada bu değerin düşük çıkması, tümör hücreleri dışında da hücre tiplerinin olması ve her hücre tipinden yeterli sayıda bulunmamasından kaynaklı olarak YOLO'nun her hücre tipini yeterince öğrenememesinden kaynaklanmaktadır.

Tablo 3.2. den de görüleceği üzere, örnek sayısı az olan kompleks hücre tiplerinin öğrenimi çok zordur ve az örnek ile öğrenim mümkün olmamaktadır. Bu yüzden YOLO, Mitoz hücre tipini öğrenememiştir. Öğrenmesi için çok fazla örneğe ihtiyaç vardır.

Diğer yandan Tübül, Nontübül ve Apoptoz ise görüntü olarak çok kompleks olmadıkları için az miktarda veri ile öğrenimi mümkündür ve tespiti kolaydır.

Bu çalışmanın asıl amacı olan tümör tespitinde ise yeterli miktarda veri ile makul seviyelerde denilecek bir başarı elde edilmiştir. Yaklaşık 18000 tümör hücresi ile eğitilen sistem **%70-75** hassasiyetle öğrenebilmiştir.

Bu çalışmada mevcut veri setindeki 6 sınıf ile öğrenmeye ek olarak, veri setinden tümör hücreleri dışındaki hücre tipleri çıkarılıp tek sınıf ile de öğrenim gerçekleştirilmiştir. Tablo 3.3'te görüleceği üzere tek sınıf ile öğrenmede daha az tekrarda yüksek başarılı bir öğrenme elde edilmiştir. 3000 tekrarda hem %72'lik başarı hem de tespit edilen tümör hücre sayısı çok yüksek çıkmıştır. Bu öğrenmede tek bir sınıf olduğu için AP değeri aynı zamanda mAP değerini vermektedir.

Tablo 3.3. YOLO'nun tek sınıf ile öğretildiğinde başarısı

TP:True Positive FP : False Positive FN: False Negative

AP : Average Precision mAP : Mean Average Precision

Tekrar(Iteration)	mAP(%)	TP	FP	FN
1000	26,95	151	112	2042
2000	62,16	1479	1221	714
3000	**72,53**	1523	716	670
4000	65,96	1523	842	670
5000	65,63	1528	883	665
6000	64,16	1463	669	730
7000	63,55	1471	743	722

Tek sınıflı eğitim hem Linux hem de Windows üzerinde çalışılmış olup, Windows'ta daha yüksek başarı elde edilmiştir. Tablo 3.3'te Windows üzerinde yapılan öğrenme sonucu görülmektedir. Linux üzerinde yapılan eğitimin sonuçları Ek A'da paylaşılmıştır.

Tek sınıflı eğitim yapılırken .cfg dosyası 1 sınıf için tekrar düzenlenmiştir. YOLO katmanlarındaki classes değeri 1, YOLO öncesi convolutional katmanlarının filters değeri 18 (3x(1+5)) olarak ayarlanmıştır.

3.4. Karşılaşılan sorunlar

- ***Out of memory hatası:*** Grafik kartımızın belleğinin belli bir sınırı var. Darknet'i geliştiren kişiler öğrenme işlemini 12GB bellekli 4 adet Titan X model grafik kartı ile yaptıkları için paylaşılan .cfg dosyasındaki konfigürasyonlarla bu hatayı almamış olabilirler. Eğer bu hata ile karşılaşılıyorsa .cfg dosyasındaki batch, subdivisions, width ve height değerlerinde değişiklik yapmak gereklidir. Batch, darknet'in tek seferde kaç görüntüyü belleğe yükleyeceğini, subdivisions ise her görüntünün kaç küçük kareye bölüneceği bilgisini verir. Width ve height değerleri ise alınan görüntünü hangi çözünürlüğe ölçekleneceğini gösterir. Batch ve subdivisions değerlerini çok düşürmek de çok yükseltmek de öğrenmeyi olumsuz etkileyebilir. Burada kullanılan bilgisayara göre optimum değerler deneyerek bulunmalıdır.

- ***Öğrenme sonrası hiç bir hücrenin tespit edilememesi :*** Bu problem YOLOv3 cfg dosyasını hiç konfigüre etmeden direkt kullanma ile alakalıdır. Bölüm 2.3.5'te anlatılan adımlar uygulanırsa hücre tespitinde başarılı sonuçlar alınabilir.
- ***Öğrenme sırasında loss(kayıp) değerinin sonsuza gitmesi :*** Bu hata da YOLOv3 dosyasını doğru konfigüre etmemekten kaynaklanmaktadır. Aynı zamanda Darknet'in Windows üzerinde derlenmiş versiyonunda da bu hata ile karşılaşılmıştır. Ayrıca, öğrenme belli bir döngü sayısından sonra da loss değeri yükselmekte ve sonsuza gitmektedir.

3.5. Gelecekte Yapılabilecekler

YOLO ile birçok uygulama yapılabilir. Bunlardan biri, YOLO'yu OpenCV desteğiyle kameradan gerçek zamanlı görüntü alınarak biyopsi incelemelerinde kullanılan görüntüleme cihazına YOLO'yu entegre etmek olabilir. Bu sayede kanserli hücrelerin tespiti biyopsi incelemesi sırasında anlık olarak yapılabilir.

Yapılabilecek başka bir çalışma ise YOLO ile geliştirilen kanser tespit sistemi çevrimiçi olarak bir sunucuda çalıştırılıp, bir internet sitesi üzerinden yayına açılarak çevrimiçi olarak görüntülerde tespit yapma hizmeti sunulabilir.

3.6. YOLO'nun Başarısını Artırma Önerileri

YOLO'nun başarısını artırabilmek için aşağıdaki gibi bazı yollar izlenebilir.

- Derin öğrenmenin olmazsa olmazı veridir. YOLO'nun hangi tür nesnede ne kadar başarılı tespitler yapması bekleniyorsa, o nesne ile ilgili mümkün olduğu kadar fazla örnek sağlanmalıdır. Örneğin, bu çalışma yaklaşık 18000 veri ile değil de 60000 veri ile gerçekleştirilseydi başarı çok daha yüksek olacaktı.
- Bu çalışmada derin öğrenme ile kanserli hücre tespiti amaçlandı ve bu amaç doğrultusunda tatmin edici bir başarı elde edildi. Tüm veri seti büyütülemiyorsa bile şekli zor olan hücre türleri için veri sayısını artırmak önemlidir.
- .cfg dosyasını konfigüre edebilmeyi iyi bilmek gerekir. Darknet bu dosyadan okuduğu parametrelerle tespit algoritmasını çalıştırdığı için, bu dosyadaki parametrelerin doğru ayarlanmasıyla daha başarılı sonuçlar elde edilebilir.
- Öğretilecek veri setini iyi tanımak gerekir. .cfg dosyasını düzenlerken öğretilecek veriye uygun konfigürasyon yapılması önemlidir. Bunun en büyük örneklerinden

biri bu çalışmadır. Mevcut yapısı ile hiçbir tespit yapamamış olan YOLO ile, doğru (ama en doğru değil) konfigürasyonlarla başarılı tespitler yapılabilmiştir.

- Güçlü bir bilgisayar kullanmak tespitin başarısını doğrudan etkilemese de hızlı sonuç almayı sağlar. Kişisel bilgisayarlarda yüksek donanımın maliyetli olmasından dolayı çevrimiçi ve ücretsiz olarak sunulan hizmetlerden faydalanmayı tavsiye ederiz. Google Colab bu noktada tercih edilebilecek bir örnektir.[35] Hem Python hem de Shell dillerini destekleyen Google Colab, çevrimiçi ve ücretsiz olarak CPU, GPU ve TPU'da işlem yapabilme imkanı sunmaktadır.
- Bu çalışmada Joseph Redmon'un kaynak kodu ile çalışılmıştır ancak AlexeyAB [30]'nin de sunduğu farklı yöntemler vardır. Bu çalışmada mevcut YOLO (YOLOv3) AlexeyAB'nin önerileri ile tümör tespitinde kullanılır hale getirilebilmiştir. YOLOVv3'te üç adet YOLO katmanı bulunmaktadır. AlexeyAB'nin kendi GitHub sayfasında ise 5 YOLO katmanı bulunan bir konfigürasyon dosyası paylaşılmıştır. Bu çalışmaya dahil edilmedi ama başarılı sonuçlar elde edilebilir. Ek A'da bu konfigürasyon ile yapılmış bir eğitimin sonuçları paylaşılmıştır. Yüksek başarı elde edilememiş olmakla beraber, konfigürasyonda bir oynama yapılmamıştır. Doğru konfigürasyon ile başarı artırılabilir.
- Joseph Redmon YOLO'nun Tiny YOLO adında minimal bir versiyonunu da sunmuştur. Bu versiyon tespiti kolay nesnelerde hızlı tespit amacıyla kullanılmak üzere geliştirilmiştir. YOLOv3'te 3 adet, Tiny YOLOv3'te ise 2 adet YOLO katmanı ve YOLOv3'e göre çok daha az konvolüsyonel katman bulunmaktadır. AlexeyAB ise GitHub sayfasında 3 katmanlı bir Tiny YOLO konfigürasyonu sunmuştur. Bu çalışmada tespitte hız ihtiyacı duyulmamış olsa da, bu çalışmadakine yakın başarıda, belki de daha başarılı sonuçları 3 YOLO katmanlı Tiny YOLO ile daha hızlı elde etmek mümkün olabilir.

4. SONUÇ

Bu çalışmada gerçek zamanlı bir nesne tespit aracı olan YOLO medikal görüntülerde tümör tespiti amacıyla kullanılmıştır ve başarılı sonuçlar elde edilmiştir. YOLO ile yapılan çalışmalar çoğunlukla günlük hayatta gözle görülebilen nesnelerin makineye öğretilip tespit etmesi amacıyla kullanılmaktadır. OpenCV desteği sayesinde kameradan alınan görüntüler üzerinde de çalışabilen Darknet sayesinde seyir halindeki bir aracın çevresindeki nesneleri ve trafik işaretlerini tespit edip, tespit ettiği nesnelere göre davranış sergileyebilmesini amaçlayan çok fazla çalışma mevcuttur. Ancak Joseph Redmon'un YOLO'yu ilk duyurduğu yayının sonuç bölümünde YOLO'nun genel amaçlı bir nesne algılama sistemi olduğunu ve yeni birçok alanda bir kullanılmasıyla birlikte nesne tespitinde ideal bir bakış açısı olabileceğini ifade etmiştir. Bu çalışma sayesinde bu ifade de desteklenmiştir. Bir nesne tespit aracı olan YOLO ile kanserli hücre tespitinde %70'in üzerinde başarı elde edilmiştir.

Derin öğrenmede başarıyı etkileyen en önemli faktörlerden biri eğitim verisinin çok olmasıdır. Bu çalışmada veri seti %70 eğitim - %30 test şeklinde ayarlanarak eğitim gerçekleştirildiğinde, mAP değeri %40'ın altında çıkmıştır. %70'in üzerindeki başarıya eğitim verisini %90'a çıkarıldığında ulaşılmıştır. Daha büyük bir veri seti ile başarının daha da yüksek çıkması beklenmektedir.

Bilgisayar gücünün makine öğrenmesi için fazlasıyla yeterli olduğu günümüzde en önemli ihtiyaç veridir. Makine ne kadar çok veri ile eğitilirse, canlı davranışının makinelerce taklit edilebilmesi o kadar iyi sağlanabilir. Günümüzde sosyal medyanın da gücü sayesinde çok fazla veri birikmektedir. Etik kurallarına da uyarak, kişisel veri olmayacak verilerden veya kişisel verileri barındırmayacak şekilde toplanacak verilerle veri setleri oluşturulması ve bu verilerin gelecekte yapılacak bilimsel çalışmalara da fayda sağlaması için paylaşılması önemlidir. Bu çalışmada kullanılan BreCaHAD veri setine XML ve YOLO formatında iki yeni etiket yapısı kazandırılmış olup, bu veriler paylaşılacaktır.

Bu çalışmada kullanılan .cfg dosyasındaki değerler optimum değerler değildir. Ancak mevcut hali ile hiç tespit yapamayan YOLO ile tümör hücreleri tespit edilebilmiştir. .cfg

dosyasında yapılabilecek başka ayarlar ve YOLO katman sayısının artırılması gibi değişiklikler YOLO'nun başarısını artırabilir.

YOLO tek sınıflı bir veri ile eğitildiğinde yüksek başarıya daha hızlı ulaşabildiği görülmüştür.

Bu çalışmada sadece meme kanserinin tespiti yapılmıştır. YOLO ile başka kanser türleri üzerinde de çalışılabilir ve farklı başarı oranlarına ulaşılabilir. Ayrıca YOLO, çeşitli kanser türleri ile eğitilerek kanserin hangi organa ait olduğunun ve bununla birlikte kanser ile ilgili birçok bilginin de tespiti(iyi-kötü huylu olması, seviyesi vs.) üzerinde de çalışılabilir.

KAYNAKLAR

[1] www.saglik.gov.tr,2016, "Kanser İstatistikleri"

https://www.saglik.gov.tr/Eklenti/8635,kanser-istatistikleridocx.docx

[2] www.cancer.gov, 2015, "What Is Cancer?" https://www.cancer.gov/about-cancer/understanding/what-is-cancer

[3] www.breastcancer.org, 2018, "What Is Breast Cancer?", https://www.breastcancer.org/symptoms/understand_bc/what_is_bc

[4] World Health Organization(WHO), 2017," 10 facts about cancer", https://www.who.int/features/factfiles/cancer/en/

[5] World Cancer Research Fund (WCRF) International,2018," Worldwide cancer data", https://www.wcrf.org/dietandcancer/cancer-trends/worldwide-cancer-data

[6] World Cancer Research Fund (WCRF) International,2018," W Breast cancer statistics", https://www.wcrf.org/dietandcancer/cancer-trends/worldwide-cancer-data

[7] National Cancer Institute, 2018, "Cancer Stat Facts: Female Breast Cancer", https://seer.cancer.gov/statfacts/html/breast.html

[8] Breastcancer.org,2018," Types of Breast Cancer", https://www.breastcancer.org/symptoms/types

[9] McCarthy, J., Minsky, M. L., Rochester, N., & Shannon, C. E. (2006). A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955. AI magazine, 27(4), 12.

[10] Krenker,A., Bešter, J., Kos, A., Introduction to the Artificial Neural Networks, Artificial Neural Networks - Methodological Advances and Biomedical Applications, (2011)

[11] Russell, S., Norvig P., (2010) Artificial Intelligence: A Modern Approach

[13] Erol, S., medium.com, 2018, "Pekiştirmeli Öğrenmeye Giriş Serisi-1", https://medium.com/deep-learning-turkiye/pekiştirmeli-öğrenmeye-giriş-serisi-1-8f5c35b6044

[13] Chandola, V., Banerjee, A., Kumar, V., Anomaly Detection: A Survey

[14] www.computerhope.com, 2019, "Computer processor history" https://www.computerhope.com/history/processor.htm9.

[15] developer.nvidia.com, "CUDA Zone", https://developer.nvidia.com/cuda-zone

[16] Richter, A. N., Khoshgoftaar, T. M., (2018), A review of statistical and machine learning methods for modeling cancer risk using structured clinical data , Artificial Intelligence In Medicine, 90, 1-14.

[17] Forsyth, A. W., Barzilay, R., Hughes, K. S., Lui, D., Lorenz, K. A., Enzinger, A., Tulsky, J. A., Lindvall, C., (2018), Machine Learning Methods to Extract Documentation of Breast Cancer Symptoms From Electronic Health Records, Journal of Pain and Symptom Management, 55, 1492-1499.

[18] Purushotham, S., Meng, C., Che, Z., Liu, Y., (2018), Benchmarking deep learning models on large healthcare datasets, Journal of Biomedical Informatics,83,112-134

[19] Hu, Z., Tang, J., Wang, Z., Zhang, K., Zhang, L., Sun, Q., (2018), Deep learning for image-based cancer detection and diagnosis −A survey, Pattern Recognition, 83,134-149.

[20] Hamidinekoo, A., Denton, E., Rampun, A., Honnor, K., Zwiggelaar, R., (2018), Deep learning in mammography and breast histology, an overview and future trends , Medical Image Analysis, 47, 45-67.

[21] Gecer, B., Aksoy, S., Mercan, E., Shapiro, L. G., Weaver, D. L., Elmore, J. G., (2018), Detection and classification of cancer in whole slide breast histopathology images using deep convolutional networks, Pattern Recognition, 84, 345-356.

[22] Liu, S., Wang, Y., Yang, X., Lei, B., Liu, L., Xiang Li, S., Wang, T. (2019). Deep Learning in Medical Ultrasound Analysis: A Review., Engineering. 261-275

[23] Selvikvåg Lundervold, A., & Lundervold, A. (2018). An overview of deep learning in medical imaging focusing on MRI. Zeitschrift Für Medizinische Physik. 102-127

[24] Janney, S. S., & Chakravarty, S. (2019). Deep learning in medical and surgical instruments. Bioelectronics and Medical Devices, 833–855.

[25] tr.wikipedia.org,"Histopatoloji", https://tr.wikipedia.org/wiki/Histopatoloji

[26] tr.wikipedia.org,"Patoloji", https://tr.wikipedia.org/wiki/Patoloji

[27] Aksac A., Demetrick D. J., Ozyer T., Alhajj R., BreCaHAD: a dataset for breast cancer histopathological annotation and diagnosis

[28] pjreddie.com, "Darknet: Open Source Neural Networks in C", https://pjreddie.com/darknet/

[29] github.com,"Darknet", https://github.com/pjreddie/darknet

[30] github.com, "Yolo-v3 and Yolo-v2 for Windows and Linux", https://github.com/AlexeyAB/darknet

[31] Redmon J., Divvala S., Girshick R., Farhadi A., You Only Look Once: Unified, Real-Time Object Detection

[32] Redmon J., Farhadi A., (2016) YOLO9000: Better, Faster, Stronger

[33] Redmon J., Farhadi A., (2018) YOLOv3: An Incremental Improvement

[34] github.com, "LabelImg", https://github.com/tzutalin/labelImg

[35] Google Colab, https://colab.research.google.com

[36] https://www.breastcancer.org/illustrations/i0013

[37] https://www.breastcancer.org/illustrations/i0061

[38] Krenker,A., Bešter, J., Kos, A., Introduction to the Artificial Neural Networks, Artificial Neural Networks - Methodological Advances and Biomedical Applications, (2011)

[39] https://machinelearnings.co/text-classification-using-neural-networks-f5cd7b8765c6

[40] https://www.quora.com/In-multilayer-neural-networks-are-weights-in-hidden-layers-closer-to-the-input-updated-more-strongly-than-weights-in-layers-closer-to-the-output

[41] https://www.datascience.com/blog/cpu-gpu-machine-learning

[42] Selvikvåg Lundervold, A., & Lundervold, A. (2018). An overview of deep learning in medical imaging focusing on MRI. Zeitschrift Für Medizinische Physik. 102-127

[43] https://developer.nvidia.com/deep-learning

[44] Neural Networks, Convolutions, Architectures Andrei Bursuc - Florent Krzakala - Marc Lelarge , https://www.di.ens.fr/~lelarge/dldiy/slides/lecture_6/

[45] Prof. Paul Cuff - Signals and Systems- Slides courtesy of John Pauly (Stanford) Lecture 3

[46] https://www.youtube.com/watch?v=aircAruvnKk

[47] https://medium.com/@ayyucekizrak/derine-daha-derine-evrişimli-sinir-ağları-2813a2c8b2a9

Ek A. YOLO ile Yapılmış Farklı Eğitim Sonuçları

i. Weights dosyası vermeden eğitim

Bu eğitim .weights dosyası verilmeden yapılmıştır ve başarısı çok düşüktür. Öğrenilmiş weigths dosyasının üzerine eğitim daha başarılı sonuçlar vermektedir. Burada bu çalışmada yüksek başarı elde edilen konfigürasyon kullanılmıştır.

Tekrar (Iteration)	Mitosis			Apoptosis			Tumor_nuclei			Non_tumor_nuclei			Tubule			Non_tubule			mAP(%)
	AP(%)	TP	FP	AP(%)	TP	FP	AP(%)	TP	FP	AP(%)	TP	FP	AP(%)	TP	FP	AP(%)	TP	FP	IoU:0.50
1000	0,03	0	0	0,03	0	0	9,33	8	27	1,40	0	0	0,02	0	0	0,02	0	0	1.81
2000	0,74	0	0	0,26	0	0	29,45	449	634	6,07	0	0	7,28	0	0	0,59	0	0	7.40
3000	0,14	0	0	1,96	0	0	7,33	25	52	0,29	0	0	5,69	0	0	2,37	0	0	2.96
4000	0,21	0	0	1,54	0	0	49,58	658	306	9,75	0	0	51,70	0	0	3,79	0	0	19.43
5000	0,82	0	0	1,54	0	0	48,93	461	124	10,67	1	6	50,68	4	1	7,41	0	0	20.01
6000	1,27	0	0	5,25	0	0	54,43	787	335	18,42	33	62	57,41	21	4	5,07	1	22	23.64
7000	11,11	0	0	9,49	0	0	55,90	1156	857	17,67	5	10	58,04	11	5	4,77	1	9	26.16
8000	4,94	0	0	9,77	0	0	46,61	503	159	16,43	17	26	47,38	7	3	3,91	0	1	21.51
9000	2,78	0	0	2,97	0	0	36,66	205	54	14,56	1	0	46,16	3	0	5,92	0	7	18.17
10000	0,00	0	0	0,00	0	0	0,00	0	0	0,00	0	0	16,28	1	1	2,62	1	8	3.15
11000	0,00	0	0	0,00	0	0	0,00	0	0	0,07	0	0	26,92	4	3	4,04	0	4	5.17
last	0,00	0	0	0,00	0	0	0,00	0	0	0,00	0	0	22,61	4	1	3,22	0	5	4.31

ii. 5 Adet YOLO katmanlı konfigürasyon ile eğitim

Bu eğitim AlexeyAB'nin 5 YOLO katmanlı konfigürasyonu ile yapılmıştır ve tatmin edici bir sonuç elde edilememiştir. (Kendisi hem küçük hem de büyük nesnelerin tespitinde bu konfigürasyonun kullanılmasını önermektedir.)

Tekrar (Iteration)	Mitosis			Apoptosis			Tumor_nuclei			Non_tumor_nuclei			Tubule			Non_tubule			mAP(%)
	AP(%)	TP	FP	AP(%)	TP	FP	AP(%)	TP	FP	AP(%)	TP	FP	AP(%)	TP	FP	AP(%)	TP	FP	IoU:0.50
2000	0,20	0	0	0,05	0	0	6,32	19	40	0,68	0	0	0,03	0	0	0,01	0	0	1,21
3000	0,00	0	0	0,12	0	0	34,89	295	183	2,91	0	0	3,65	0	0	0,25	0	0	6,97
4000	0,00	0	0	0,69	0	0	46,63	474	216	9,95	0	0	21,28	1	1	5,36	0	0	13,99
5000	0,06	0	0	4,93	0	0	50,52	1015	727	9,80	0	0	50,78	12	4	12,18	1	1	21,38
6000	0,00	0	0	5,42	0	0	55,94	779	343	17,67	0	0	66,22	20	5	21,78	4	3	27,84
7000	0,52	0	0	5,80	0	0	58,26	964	453	15,32	1	1	59,39	31	11	16,08	6	11	25,89
8000	5,24	0	0	12,19	1	0	56,28	1035	513	10,10	0	3	68,46	21	7	17,81	3	4	28,34
9000	0,56	0	0	7,98	0	2	50,43	668	326	13,30	18	25	62,26	32	13	17,52	10	20	25,34
10000	0,00	0	0	3,75	0	1	56,53	924	472	10,32	5	13	66,92	26	7	15,89	11	15	25,57
final	0,58	0	0	12,87	0	0	55,40	687	246	9,22	0	2	65,33	27	7	20,83	3	3	27,37
last	11,11	0	0	9,01	0	0	54,55	449	118	11,52	4	10	69,32	24	6	16,26	5	6	28,63

iii. YOLO'nun tek sınıflı veri ile eğitimi

Bu eğitimde veri seti sadece tümör hücrelerinden oluşacak şekilde ayarlanmış, aynı zamanda YOLO katmanlarının classes değeri 1, YOLO öncesi convolutional katmanlarının filters değeri 18 olarak ayarlanmıştır. Bu çalışma Ubuntu Linux üzerinde gerçekleşmiş olup, Windows üzerinde daha yüksek başarı elde edildiği için Windows sonuçları çalışma sonuçlarında paylaşılmıştır.

Tekrar(Iteration)	mAP(%)	TP	FP	FN
1000	23,28	69	49	2124
2000	39,13	114	87	2079
3000	57,50	871	375	1322
4000	62,00	962	371	1231
5000	61,36	1531	1301	662
6000	59,34	838	262	1355
7000	57,69	1572	1828	621
8000	57,47	1607	1963	586
9000	35,74	259	87	1934

iv. YOLO'nun tek sınıflı veri ile eğitimi -2

Bu eğitimde 6 sınıflı veri seti ile eğitilen YOLO konfigürasyonu kullanılmıştır. Ancak veri setinden tümör hücreleri dışındaki hücre tipleri çıkarılmıştır. Eğitim Ubuntu üzerinde gerçekleştirilmiştir.

Tekrar(Iteration)	mAP(%)	TP	FP	FN
1000	18.31	0	2	2193
2000	55.44	867	454	1326
3000	53.66	601	190	1592
4000	60.75	1438	1060	755
5000	53.86	469	99	1724
6000	58.42	1549	1495	644
7000	63.40	1113	493	1080
8000	61.31	1242	709	951
9000	45.94	453	134	1740

ÖZGEÇMİŞ

Adı Soyadı **:** Mustafa YANCI

Doğum Yeri ve Tarihi **:** İstanbul - 1989

Yabancı Dili **:** İngilizce

E-Posta **:** mustafayanci@gmail.com

Öğrenim Durumu

Derece	**Bölüm/Program**	**Üniversite/Lise**	**Mezuniyet Yılı**
Lise	Anadolu Lisesi	Ümraniye Anadolu Lisesi	2009
Lisans	Elektronik Mühendisliği (%100 Burslu - %100 İngilizce)	İstanbul Kültür Üniversitesi	2015

İş Deneyimi

Yıl	**Firma/Kurum**	**Görevi**
2016	Türk Telekom / AVEA	Mobil Faturalı Hatlar Gelir Yönetimi Uzmanı