T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

DERİN ÖĞRENME ALGORİTMALARINI KULLANARAK GÖRÜNTÜ SINIFLANDIRMA

Utku Kubilay ÇINAR

YÜKSEK LİSANS TEZİ

İstatistik Anabilim Dalı

İstatistik Programı

Danışman

Prof. Dr. Gülhayat GÖLBAŞI ŞİMŞEK

Haziran, 2020

T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

DERİN ÖĞRENME ALGORİTMALARINI KULLANARAK GÖRÜNTÜ SINIFLANDIRMA

Utku Kubilay ÇINAR tarafından hazırlanan tez çalışması 26.06.2020 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü İstatistik Anabilim Dalı, İstatistik Programı **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Prof. Dr. Gülhayat GÖLBAŞI ŞİMŞEK
Yıldız Teknik Üniversitesi
Danışman

Jüri Üyeleri

Prof. Dr. Gülhayat GÖLBAŞI ŞİMŞEK, Danışman
Yıldız Teknik Üniversitesi _____

Prof. Dr. Fatma NOYAN TEKELİ, Üye
Yıldız Teknik Üniversitesi _____

Doç. Dr. Seher ARIKAN TEZERGİL, Üye
Marmara Üniversitesi _____

Danışmanım Prof. Dr. Gülhayat GÖLBAŞI ŞİMŞEK sorumluluğunda tarafımca hazırlanan Derin Öğrenme Ağlarını Kullanarak Görüntü Sınıflandırma başlıklı çalışmada kaynaklardan aldığım bilgileri ana metin ve referanslarda eksiksiz gösterdiğimi, araştırma verilerine ve sonuçlarına ilişkin çarpıtma ve/veya sahtecilik yapmadığımı, çalışmam süresince bilimsel araştırma ve etik ilkelerine uygun davrandığımı beyan ederim. Beyanımın aksinin ispatı halinde her türlü yasal sonucu kabul ederim.

Utku Kubilay ÇINAR

TEŞEKKÜR

Başta bana her konuda, her zaman yardımcı olan danışman hocam Prof Dr. Gülhayat GÖLBAŞI ŞİMŞEK' e, Dr. Fahrettin Fırat GÖNEN'e, sevgili aileme ve ayrıca bugüne kadar bana emeği geçen herkese çok teşekkür ederim.

Utku Kubilay ÇINAR

İÇİNDEKİLER

KISALTMA LİSTESİ vi

ŞEKİL LİSTESİ vii

TABLO LİSTESİ ix

ÖZET x

ABSTRACT xii

1 GİRİŞ 1

1.1 Literatür Özeti ... 1
1.2 Tezin Amacı ... 8
1.3 Hipotez ... 8

2 KONUYLA ALAKALI TEMEL KAVRAMLAR 9

2.1 Makine Öğrenmesi ... 9
2.1.1 Denetimli Öğrenme ... 11
2.1.2 Denetimsiz Öğrenme ... 12
2.1.3 Yarı Denetimli Öğrenme ... 12
2.1.4 Takviyeli Öğrenme ... 12
2.2 Derin Öğrenme ... 12
2.3 Transfer Öğrenme ... 14
2.4 ImageNet Veri Seti ... 17
2.5 Keras ... 20
2.6 Tensorflow ... 21
2.7 Bilgisayarlı Görü için Derin Öğrenme ... 23
2.7.1 Evrişimsel Sinir Ağları (CNN) ... 23
2.7.2 İletim Sönümleme (Dropout) ... 29
2.7.3 Az Öğrenme ve Ezberleme (Underfit ve Overfit) ... 31
2.7.4 Görüntü İşleme Modellerinde Kullanılan Hiper Parametreler ... 33

2.7.5 Aktivasyon Fonksiyonları 38
2.7.6 Veri Artırma 44
2.7.7 Derin Öğrenmede Görüntü Ön İşleme 48
3 UYGULAMA **50**
3.1 Veri Seti 50
3.2 Yöntem ve Bulgular 51
4 SONUÇ VE ÖNERİLER **61**
4.1 Tartışma ve Sonuç 61
4.2 Öneriler 62
KAYNAKÇA **63**
TEZDEN ÜRETİLMİŞ YAYINLAR **67**

KISALTMA LİSTESİ

BERT	Bidirectional Encoder Representations from Transformers
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DNN	Deep Neural Network
EKK	En Küçük Kareler
ESA	Evrişimsel Sinir Ağları
GAN	Generative Adversarial Networks
GPU	Graphics Processing Unit
ILSVRC	ImageNet Large Scale Visual Recognition Competition
MNIST	Modified National Institute of Standards and Technology
NLP	Natural Language Processing
RL	Reinforcement Learning
SGD	Stochastic Gradient Descent
TL	Transfer Learning
TPU	Tensor Processing Unit
YSA	Yapay Sinir Ağları
YZ	Yapay Zeka

ŞEKİL LİSTESİ

Şekil 1.1 Derin öğrenmenin tarihçesi 5
Şekil 2.1 Zetabayt cinsinden yıllık veri hacmi.....10
Şekil 2.2 Makine öğrenmesi kelimesinin Google üzerinde aranma eğilimi.....11
Şekil 2.3 Derin öğrenme kelimesinin Google üzerinde aranma eğilimi.....14
Şekil 2.4 Inception v3 derin öğrenme mimarisi.....16
Şekil 2.5 Alexnet mimarisi.....17
Şekil 2.6 Zfnet mimarisi.....18
Şekil 2.7 Googlenet mimarisinin bir kesiti.....19
Şekil 2.8 Örnek bir keras derin öğrenme modeli.....21
Şekil 2.9 Üç renk kanalı.....23
Şekil 2.10 Esa yapısı.....24
Şekil 2.11 3x3 boyutunda filtre katmanı.....24
Şekil 2.12 3x3 boyutunda filtrenin görüntü verisine uyguladığı konvolüsyon işlemi 25
Şekil 2.13 Örnek dolgulu evrişim işlemi.....26
Şekil 2.14 Örnek padding kullanılmayan evrişim işlemi.....26
Şekil 2.15 Esa katman sıralaması ve örnek mimarisi.....27
Şekil 2.16 Maksimum ortaklama işlemi.....28
Şekil 2.17 Maksimum ve ortalama ortaklama işlemi.....28
Şekil 2.18 Düzleştirme (flattening).....29
Şekil 2.19 İletim sönümleme (dropout).....30
Şekil 2.20 Az öğrenme - aşırı uyum ile yanlılık - varyans ilişkisi.....31
Şekil 2.21 Yanlılık – varyans ilişkisi.....32
Şekil 2.22 Toplam hata – varyans ve yanlılık ilişkisi.....33
Şekil 2.23 Makine öğrenmesi model geliştirme süreci.....34
Şekil 2.24 Olasılıksal eğim düşümü.....36
Şekil 2.25 Öğrenme oranı ve tekrar sayısı.....37
Şekil 2.26 Öğrenme oranı ve lokal minimum.....37
Şekil 2.27 Görüntünün dijitalleştirilmiş bilgisayar görüsündeki karşılığı.....40

Şekil 2.28 Doğrusal aktivasyon fonksiyonu ve türevi 40
Şekil 2.29 Tek kutuplu ve çift kutuplu adım aktivasyon fonksiyonu 41
Şekil 2.30 Sigmoid aktivasyon fonksiyonu 41
Şekil 2.31 Hiperbolik tanjant aktivasyon fonksiyonu 42
Şekil 2.32 Doğrusal birim aktivasyon fonksiyonu 42
Şekil 2.33 Aktivasyon fonksiyonlarının düzlem üzerinde gösterimi 43
Şekil 2.34 Aktivasyon fonksiyonları 44
Şekil 2.35 Döndürme işlemi yaparak sentetik veri üretme 45
Şekil 2.36 Autoaugment yöntemi ile sentetik veri üretme 46
Şekil 2.37 Nesne belirleme yaparak sentetik veri üretme 47
Şekil 2.38 Gri renk tonları ve bilgisayar görüsündeki karşılıkları 48
Şekil 2.39 Bazı renklerin rgb değerleri 49
Şekil 3.1 Kurulan birinci modelin mimarisi 53
Şekil 3.2 Kurulan ikinci modelin mimarisi 54
Şekil 3.3 Kurulan üçüncü modelin mimarisi 55
Şekil 3.4 İlk kurulan modelin başarı - eğitim ve validasyon kayıp değerleri 56
Şekil 3.5 İkinci kurulan modelin başarı - eğitim ve validasyon kayıp değerleri 58
Şekil 3.6 Tensorflow lite ile geliştirilebilecek mimarilerin boyutları - başarıları ve tahmin süresi 59

TABLO LİSTESİ

Tablo 2.1 2012-2017 yılları arasında ILSVRC yarışması sonuçları.....20
Tablo 3.1 İlk kurulan modelin çapraz doğrulama tablosu.....57
Tablo 4.1 Model kıyaslama tablosu 61

ÖZET

Derin Öğrenme Algoritmalarını Kullanarak Görüntü Sınıflandırma

Utku Kubilay ÇINAR

İstatistik Anabilim Dalı

Yüksek Lisans Tezi

Danışman: Prof. Dr. Gülhayat GÖLBAŞI ŞİMŞEK

Günümüzde bilgisayar teknolojilerinin gelişmesi, Merkezi İşlem Birimi ve Grafik İşlemci Ünitelerinin görsel / fotoğraf / video analizlerinde kullanılmasıyla beraber bilgisayarla görü süreçleri, daha az maliyetle ve daha yüksek başarılarla yapılabilmektedir. Bu çalışmada amaçlanan, yapay zeka tabanlı derin öğrenme algoritmalarını kullanarak fotoğrafları sınıflandırmaktır. Bu sınıflandırma işlemi Python programı kullanılarak yapılmıştır.

Bu çalışmada kullanılacak veri seti olarak, her evde rahatça bulunabileceğinden dolayı ilaç kutuları seçilmiştir. 10 farklı ilaç kutusunun, 200 tane fotoğrafı eğitim için ve 50 tane fotoğrafı test verisi için çekilmiştir. Toplamda 10 farklı ilaç kutusunun, 2500 adet fotoğrafı çekilmiştir. Etiketleme ve klasörleme işleminin ardından ilk olarak eğitim verisinin klasörü oluşturulmuştur. Daha sonra aynı işlemler test verisi için de uygulanmıştır. Bu işlemlerin ardından bütün fotoğrafların boyutları değiştirilmiştir. Görüntü işleme süreçlerinde yaygın kullanılan boyut, 224 x 224 x 3 piksel boyutudur. Bütün fotoğraflar bu boyutta olacak şekilde yeniden düzenlenmiştir. Model kurulmadan önce veriyi artırmak için yapay fotoğraflar

üretilmiştir. Her fotoğraf eğitilmeden önce belirlenmiş kriterlere göre kendini değiştirir ve sisteme yeni bir fotoğrafmış gibi girer. Böylece veri miktarı artmış ve sınıflandırma başarısı da yükselmiş olur. Sınıflandırma yapmak için Evrişimsel Sinir Ağları algoritması ile beraber transfer öğrenme kullanılmıştır. Uygun model ve mimari sonucunda geliştirilen model; 500 fotoğrafı, 10 sınıfa %97'i aşan bir başarı oranıyla sınıflandırmıştır.

Anahtar Kelimeler: Sınıflandırma, görüntü işleme, evrişimsel sinir ağları

ABSTRACT

Image Classification Using Deep Learning Algorithms

Utku Kubilay ÇINAR

Department of Statistics

Master of Science Thesis

Advisor: Prof. Dr. Gülhayat GÖLBAŞI ŞİMŞEK

Nowadays, thanks to the development of computer technologies and the use of Central Processing Unit (CPU) and Graphics Processor Units (GPU) on visual / photo / video analyses, computer vision processes can be achieved with less cost and higher success. The aim of this study is classification of photographs using deep learning algorithms based on artificial intelligence. This classification was made using Python.

As the data set to be used in this study, pillbox were chosen because they can be found easily in every house. 200 photographs of 10 different medicine boxes were taken for training data and 50 photographs for test data. In total, 2500 photographs of 10 different medicine boxes were taken. Following labeling and foldering, the training data folder is first created. The same procedures were applied for the test data. After this process all photos have been resized. The widely accepted dimesions in the image processing is 224 x 224 x 3 pixels. All photos were resized accordingly to this dimesions. Before the model was built, artifical images were produced to increase the data. Each photo changes itself according to the specified criteria before training and enters the system as if it were new photo. Thus, data amount increases

and classification success also increases. In order to classify, CNN (Convolutional Neural Network) algorithm and Transfer Learning were used. Developed as a result of the appropriate model and architecture, the model classified 500 photographs with an accuracy rate exceeding 97% in 10 classes.

Keywords: Classification, image processing, convolutional neural network

1
GİRİŞ

1.1 Literatür Özeti

Günümüzde bilgisayar gücünün artması ve verinin çeşitlenmesiyle beraber, hemen hemen her alanda veri analizleri ve gelişmiş sınıflandırma modelleri kurulabilmektedir. Teknolojide yaşanan gelişmeler sayesinde, insan beyninin objeleri sınıflandırma yeteneği kadar olmasa da, görseller üzerinden nesneleri sınıflandırma yeteneği artık bilgisayarlara geçmiş durumdadır. Derin öğrenme algoritmaları, görüntü sınıflandırma problemlerindeki yüksek başarılarıyla kendilerini defalarca kez ispatlamıştır. Son yıllarda görüntü işleme projeleri ya da yapay zekada bilgisayar görüsü, herkesin dikkatini çekmiş ve ilgi odağı haline gelmiştir.

Bilgisayarlar gelişmiş yapay zeka algoritmalarını kullanarak, insan kararıymışçasına görüntüleri ya da objeleri sınıflandırabilmektedir. Fotoğraf kalitelerinin ve piksel sayılarının artmasıyla, verinin hacmi ve işlem gücü bilgisayarlar açısından zorlaşmıştır. Teknolojinin bu problemi giderebilmesi için bilgisayarların GPU denilen Grafik İşlemci Üniteleri'nin kullanımı yaygınlaşmıştır.

Derin öğrenme algoritmaları, görüntü işleme süreçlerinde piksel olarak adlandırılan noktaları, bir matris formatı gibi düşünür ve işlemlerini gerçekleştirir. Herhangi bir görseli / fotoğrafı bilgisayarların işleyebilmesi için güçlü işlemcilere ihtiyaç duyulmaktadır. Bunun en temel sebeplerinden biri ise fotoğrafı matris formatına dönüştürüp, ilişkileri yakalamaya çalışmasıdır. Bu sebeple, derin öğrenme algoritmaları yoğun miktarda çok boyutlu matris işlemleri kullanır.

Derin Öğrenme kavramı ilk olarak, 2000 yılında Igor Aizenberg ve arkadaşlarının yaptığı çalışmada kullanılmıştır [1]. 1950'li yıllarda yapay zeka ile ilgilenen akademik çevreler, bilgisayarlı görü için iki yöntem öne sürmüşlerdir. Bunlar; Yapay Sinir Ağları ve Karar Ağaçlarıdır. Yapay Sinir Ağları insan beyninin çalışma mekanizmasından esinlenilerek ortaya atılmış bir yöntemdir ve sinir hücrelerinin

çalışma şekli simüle edilerek çok katmanlı ağ yapısı oluşturulmuştur. Bu yapı 1980'li yılların sonundan 2000'li yılların başına kadar geçen sürede büyük başarılar elde etmiştir [2] [3].

Literatür derinlemesine incelendiğinde, son 5 yılda görüntü işleme ve obje tanımlama alanlarında ciddi gelişmeler olduğu görülmektedir. Derin öğrenme algoritmalarından biri olan Evrişimsel Sinir Ağları, derin öğrenmenin özelleştirilmiş bir halidir. Ayrıca günümüzde görüntü işlemede en yaygın kullanılan algoritmaların başında gelmektedir. Kabaca, bir görüntüden faydalı bilgiler elde etme ve modelleme süreçlerine, görüntü işleme denilmektedir.

İşlenebilecek görüntü kameralardan, videolardan ve tarayıcılardan elde edilebilir. Bu görüntüler matris formatında sayısallaştırılarak algoritmaya sokulur ve bilgiler elde edilmeye çalışılır. Makine öğrenmesi sistemleri elle programlanmak yerine eğitilirler. Derin öğrenme algoritmaları göreve yönelik verilen örneklere bakarak, istatistiksel örüntüleri ortaya çıkarıp görevi otomatikleştirecek kurallar ortaya koyar [4].

Bu çalışmanın amacı, kameralar yardımıyla elde edilen görüntüleri sınıflandırmaktır. Bu sınıflandırmayı yapabilmek ve yüksek başarı elde edebilmek için çalışmada, Evrişimsel Sinir Ağları ve transfer öğrenme yöntemi kullanılmıştır. Çalışma kapsamında elde edilen görseller öncelikle veri ön işleme adımları izlenerek, boyutları değiştirilmiş ve sayısallaştırılmıştır. Ardından her görsel algoritmaya sokularak görüntü öznitelikleri çıkarılmıştır. Çıkarılan bu öznitelikler ile görseller sınıflarına ayrıştırılmıştır.

Modelin doğru sınıflandırma performansını artırmak için her görsel, modele girmeden önce yapay olarak artırılmıştır. Veri artırma işleminde; her görselin boyutları, odaklandığı yerler ve açısal döndürmeler gibi işlemler yapılarak farklı bir görselmiş gibi modele girer ve veri zenginleştirilmiş olur. Böylelikle daha az efor ile daha fazla veri modele girer ve modelin başarısı artırılmaya çalışılır.

Bilgisayarlı görü, derin öğrenme ve görüntü işleme denilince akla ilk MNIST veri seti gelmektedir. MNIST veri setinde, yaklaşık 60.000 eğitim ve 10.000 test verisi bulunmaktadır ve el yazılarının sınıflandırılması amaçlanan bir veri setidir. Bu veri

seti, görüntü işleme algoritma başarılarının kıyaslanması için sıklıkla kullanılır ve ortak bir değerlendirme verisidir [5].

Veri bilimi (data science), verilerden anlam üretmek, değer elde etmek ve veri analizleri süreçleri ile bilimsel yöntemleri kullanan disiplinlerarası bir uzmanlık alanıdır. Çeşitli kaynaklardan veriler kullanarak, talep edilen şeyi etkileyecek tüm durumları belirleyip, farklı yöntemler ile tahmin edilmesini sağlayan, büyük verilerin analizinin yapıldığı bilim dalıdır.

Veriler analiz edilirken, genellikle istatistik ve bilgisayar teknolojileri kullanılır. Kısaca veri bilimi, veriyi anlamlandırma bilimidir. Veri biliminde; sinyal işleme, makine öğrenmesi, istatistiksel öğrenme ve modelleme, veri madenciliği, bilgisayar bilimleri ve disiplinlerarası teknik ve teorik bilgiler kullanılmaktadır.

Veri bilimi aslında istatistik yöntemleri kullanmaktadır; fakat uygulama alanlarında bazı farklılıklar vardır. Veri biliminde kullanılan veriler genellikle heterojen yapıdadır. Veri biliminde veriler, yapılandırılmış, yapılandırılmamış ya da yarı yapılandırılmış olabilir.

Veri bilimi, özellikle son 5 yılın en popüler araştırma ve uygulama alanlarının başında gelmektedir. Disiplinlerarası olmasından kaynaklı veri bilimi ile ilgili olan araştırmacıların matematiksel alt yapısı ve bilgisayar teknolojilerine ilgisi olmalıdır. Veri bilimci (data scientist), çeşitli veri kaynaklarını kullanarak veri üzerinde çeşitli sorular soran, cevaplar arayan, analitik kabiliyeti yüksek, veriyi modelleyen, katma değeri yüksek bilgiler elde etmeye çalışan, karmaşık veriden fayda üretmeye çalışan, sorunların kaynaklarını araştıran ve çözüm önerileri sunan, sayısal tahminler geliştiren, çalıştığı kurum ve kuruluşların stratejilerini belirleyebilen, makine öğrenmesi algoritmaları kullanarak yapay zeka tabanlı ürünler geliştirebilen ve programlama bilgilerini kullanarak çeşitli analizleri gerçekleştiren, verileri görselleştiren ve sunan kişidir. Ayrıca veri bilimci, bir yazılımcıdan daha çok istatistik, bir istatistikçiden daha çok yazılım bilmelidir. Yapısal olmayan veri yığınlarını kullanarak, bilgiyi nasıl elde edebileceğini ve kullanabileceğini bilen kişidir.

Bir veri bilimci, bilgisayar teknolojilerinden en çok Python ve R programlama dillerine ihtiyaç duymaktadır. Python programlama dili, genellikle yazılım, web programlama alanlarında popülerken, R programlama dili, veri görselleştirme ve istatistiksel analizler alanlarında yaygın kullanılmaktadır. Veri biliminde, veriler üzerinden model kurulmadan önce ciddi bir manipülasyon sürecinden geçilmelidir. Veri, model kurulmaya hazır hale getirilmelidir.

Python programlama dilinde büyük verilerin manipülasyonu daha hızlıyken, R programlama dili daha yavaş fakat yazım dili daha kolaydır. Veri manipülasyonu sürecinde Python programında "Pandas" gibi kütüphaneler kullanılırken, R programlama dilinde "tidyverse" gibi kütüphaneler kullanılmaktadır. R ve Python programlama dillerinin son 5-10 yılda popüler olmasının temel sebeplerinden bazıları ise, lisanssız, ücretsiz ve açık kaynak kodlu olmasıdır. Python ve R programlama dilleri, kolay yazımı, basitliği, farklı kütüphanelerin varlığı ve geniş kullanıcı ekosistemi olduğu için veri biliminde kullanılan en yaygın bilgisayar teknolojilerinin başında gelmektedir.

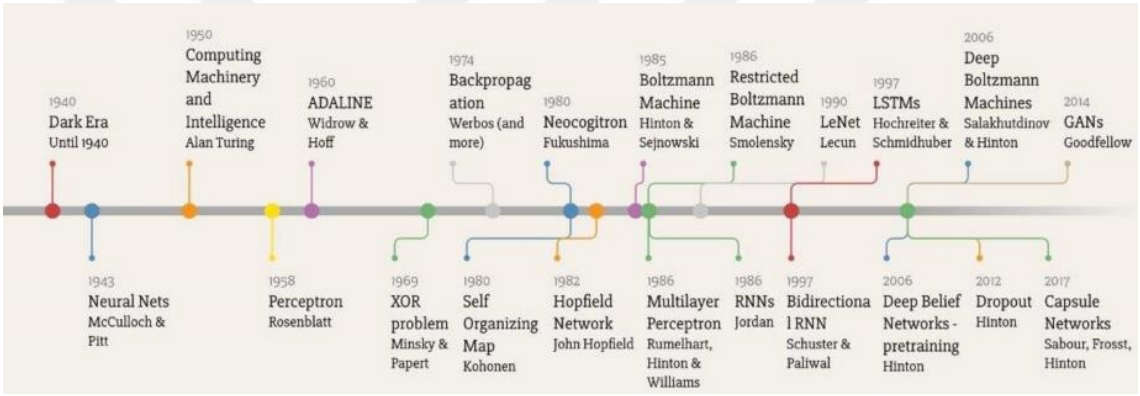
Veri bilimi proje süreçleri kısaca aşağıda belirtildiği gibidir;

- Problemi anlamak
- Problemi çözmek için nelere ihtiyacının olduğunu belirlemek
- Problemin yaşandığı ve konuyu iyi bilen kişilerle görüşerek problemi anlamlandırmak
- Veri kaynaklarına erişerek, veriyi elde etmek
- Problemin çözümünde kullanmak ve geliştirilecek modelin performansını yükseltebilmek için veriyi zenginleştirmek
- Literatürü araştırmak, geçmiş çalışmaları incelemek
- Veriyi manipüle etmek. Model öncesi veriyi, model kurmaya hazır hale getirmek
- Uç değerleri temizlemek
- Kayıp verileri silmek ya da doldurmak
- Keşifçi veri analizleri yapmak, yorumları ve sonuçları paylaşmak
- Probleme uygun modeller geliştirmek ve kıyaslamak
- Sonuçları paylaşmak ve raporlamak

Şeklinde özetlenebilir.

Derin öğrenme, makine öğrenmesinin alt başlıklarından biridir. Derin öğrenme algoritmaları sayesinde, görüntü verisinde çeşitli dönüşümler yapılır. Görüntünün sahip olduğu özellikler ve doğrusal olmayan ilişkiler, derin ve farklı katmanlarca işlenir. Her sıralı katman, bir önceki katmanın çıktısını girdi alır [6] [7].

Yapay Sinir Ağları'nda gerçekleşen öğrenme süreci, insan beyninden etkilenilerek ortaya atılmıştır. Bu öğrenme sürecindeki yaklaşım ilk olarak, 1943 yılında insan beynindeki nöronların görevi ve iletim sistemlerinin çalışma mantığı, matematiksel modellemeler ile gerçekleştirilmiştir [8]. Bu matematiksel modellemeler, eşik mantığı adı verilen matematik ve algoritma tabanlı sinir ağlar için hesaplamalar yapmaktadır.



Şekil 1.1 Derin öğrenmenin tarihçesi [9]

Yapay Sinir Ağları'nda çoklu gizli katmanlara sahip, ileri beslemeli sinir ağları veya çok katmanlı yapılar, genellikle Derin Sinir Ağları (DNN) olarak adlandırılmaktadır. Evrişimsel Sinir Ağları (CNN) ileri ve geri beslemeli çalışabilen bir sinir ağıdır. 1960'larda, Hubel ve Wiesel, bir kedinin görme sisteminde hassas yönelim - seçici kullanan nöronları farketmişlerdir. Geri beslemeli sinir ağlarının karmaşıklığını etkili bir şekilde azaltabilmek için kedinin nöronlarındaki özel ağ yapısının kullanılabileceği belirtilmiştir. Bu yapıya Evrişimsel Sinir Ağı denmektedir.

Evrişimsel Sinir Ağları, görüntünün sahip olduğu ilişkileri ve yapıları belirlemede yaygın kullanılan bir algoritmadır. Basit yapısı, az parametreye ihtiyaç duyması ve uygulanabilirliği gibi birçok avantaja sahiptir. Günümüzde ses analizi ve görüntü tanıma gibi konularda sıklıkla bu algoritma tercih edilmektedir. Ayrıca Evrişimsel

Sinir Ağları, modelin biyolojik sinir ağlarına benzer olmasını sağlamaktadır, böylelikle ağ modelinin karmaşıklığını azaltır [10].

Çok katmanlı denetimli derin öğrenme nöronları Ivakhnenko ve Lapa tarafından 1965 yılında tanıtılmıştır [11]. Lapa ve Ivankhnenko'nun yaptığı çalışmada her katman, çıkardığı nitelikleri bir sonraki katmana iletmektedir. Önceki katmanlardan alınan bilgi, sonraki katmanlara iletilirken En Küçük Kareler (EKK) yöntemi kullanmaktadır. Günümüzde bu işlem için genellikle geri yayılım (backpropagation) tercih edilmektedir.

Yapay Sinir Ağları'nın tarihçesi, insanların nörobiyoloji konusuna ilgi duyması ve elde ettikleri bilgileri bilgisayar bilimine uyarlamasıyla başlamaktadır. Yapay Sinir Ağları'nın tarihini 1970 öncesi ve sonrası diye ikiye ayırmak gerekir. 1970 yılı, Yapay Sinir Ağları yaklaşımının tarihinde bir dönüm noktası olmuş ve o zamana kadar kullanılamaz denilen Yapay Sinir Ağları yeniden doğmuştur [2].

Görüntü işlemede sıklıkla kullanılan derin öğrenme algoritmaları, Geoffrey Hinton ve arkadaşlarının yayınladıkları çalışma ile ortaya atılmıştır [12] [13]. Ayrıca bu çalışmada overfit /aşırı uyum / ezberleme olmaması için dropout / iletim sönümleme kavramı ortaya atılmıştır. Dropout kavramı ise oluşturulan yapay sinir ağı modelinde nöronların rastgele silinmesidir. Dropout kullanılan çalışmalarda başarı seviyesinin yükseldiği gözlemlenmiştir.

2012 yılında yapılan bir araştırmada, logsig transfer fonksiyonu ile Çok Katmanlı Yapay Sinir Ağları kullanılarak patates görüntülerinin sınıflandırılması yapılmıştır. Elde edilen patates görüntüleri, eşitlenerek gerekli bilgiler sütun matrislerine dönüştürülmüştür. Bu çalışmada, ileri beslemeli - geri yayılım Yapay Sinir Ağları modelleri kullanılmış ve bir ya da daha fazla katmanlı Yapay Sinir Ağı mimarileri oluşturulmuştur. Bu çalışma ile webcam kullanılarak, gerçek zamanlı patates büyüklükleri ve kaliteleri sınıflandırılmaya çalışılmıştır [14].

Geri yayılım algoritmaları önceki yıllarda ortaya atılmış olsa da ilk başarılı derin sinir ağı uygulamasını Yann LeCun ve arkadaşları gerçekleştirmiştir. Bu çalışmada, posta kutusunun üzerindeki yazılara odaklanılmıştır. Her ne kadar ağ başarılı

çalışsa da, model eğitimi yaklaşık 3 gün sürdüğü için pratikte uygun olmadığı görülmüştür [15] [7].

ImageNet, 15 milyondan fazla etiketli, yüksek çözünürlüklü görüntü veri setidir. Görüntüler internetten toplanıp, etiketlenmiştir. 2010'dan itibaren Pascal Görsel Nesne Yarışması'nın bir parçası olarak, ImageNet Büyük Ölçekli Görsel Tanıma Yarışması (ILSVRC) adı verilen yarışma düzenlenmektedir. ILSVRC, 1000 kategoriden ve her sınıftan yaklaşık 1000 görüntü içeren, ImageNet verilerini kullanan bir alt veri kümesidir. ImageNet veri setinde, yaklaşık 1.2 milyon eğitim görüntüsü, 50.000 doğrulama görüntüsü ve 150.000 test görüntüsü bulunmaktadır [16].

VGG16 derin öğrenme algoritması, Oxford Üniversitesi'ndeki Görsel Geometri Grubu (VGG) tarafından ImageNet Büyük Ölçekli Görsel Tanıma 2014 Yarışması' nda daha iyi sonuç vermesi için geliştirilmiştir. VGG16 mimarisi, 13 konvolüsyon ve 3 tam bağlantılı katmandan oluşmaktadır. Pooling / ortaklama, tam bağlantı, iletim sönümleme ve konvolisyon katmanlarıyla beraber toplamda 41 katmandan oluşmaktadır. Girdi katmanı yani her görsel, 224 x 224 x 3 olacak şekilde boyutlandırılmalıdır; çünkü girdi katmanı bu boyutlu verilerde çalışmaktadır. Son katman ise sınıflandırılmanın yapıldığı katmandır [12].

2018 yılında, bitki yapraklarının sınıflandırılması için derin öğrenme algoritmaları kullanılmıştır. 32 sınıftan toplamda 1900 görsel ile yapılan çalışmada; AlexNet, VGG16, ResNet50 ve GoogleNet gibi algoritmalar kullanılmıştır. Iterasyon / tekrar sayıları değiştirilerek doğru sınıflandırma başarıları değiştiği gözlemlenmiştir. Yapılan çalışmada, AlexNet %99.72 doğru sınıflandırma ile en başarılı algoritma olduğu görülmüştür. Ayrıca GPU kullanılarak her eğitim süreci 1-3 saat sürerken, CPU kullanılarak eğitilen modellerin 24 saatten uzun sürdüğü saptanmıştır [12].

Evrişimsel ağlar kullanılarak heyelan sınıflandırılması konu alınan çalışmada, Evrişimsel Sinir Ağlar kullanarak heyelan fotoğrafları sınıflandırılmasında ilk girişim olduğu ve ümit verici sonuçlar elde edildiği belirtilmiştir. Bu çalışmada veri sayısının az olması, karşılaşılan en büyük zorluk olarak ifade edilmiştir. Bu sorunu gidermek için VGG16 mimarisine dayanan bir Evrişimsel Sinir Ağları modeli

kullanılmıştır. Bu çalışmada kullanılan modelde yüksek başarı elde edildiği belirtilmiştir [17].

1.2 Tezin Amacı

Tezin amacı, derin öğrenme modellerini ve çeşitli mimarileri kullanarak çok boyutlu görüntü verileri üzerine sınıflandırma modeli geliştirebilmektir. Görüntü sınıflandırma modelleri Python programı üzerinden geliştirilmiştir. Derin öğrenme alanında araştırmacıların karşısına çıkabilecek yaygın kavramlar tanıtılmış ve uygulaması yapılmıştır. Son yıllarda oldukça popüler olan derin öğrenme modelleri kıyaslanmıştır ve her yönüyle araştırılmıştır.

1.3 Hipotez

Veri bilimi ve derin öğrenmede sıklıkla kullanılan görüntü sınıflandırma modellerinin, ilaç kutuları görüntüleri üzerinde sınıflandırma başarıları gözlenmek istemiştir. Bu kapsamda derin öğrenme algoritmaları ve derin katmanlarda öğrenme işlemlerinin nasıl yapıldığı incelenip, en uygun modelin elde edilmesi üzerine çalışılmıştır. Özellikle Tensorflow Lite kullanılarak geliştirilen modeller ve çeşitli yöntemler test edilmiştir. Bu sebeple bu tez, benzer çalışmalardan öne çıkmaktadır.

2
KONUYLA ALAKALI TEMEL KAVRAMLAR

2.1 Makine Öğrenmesi

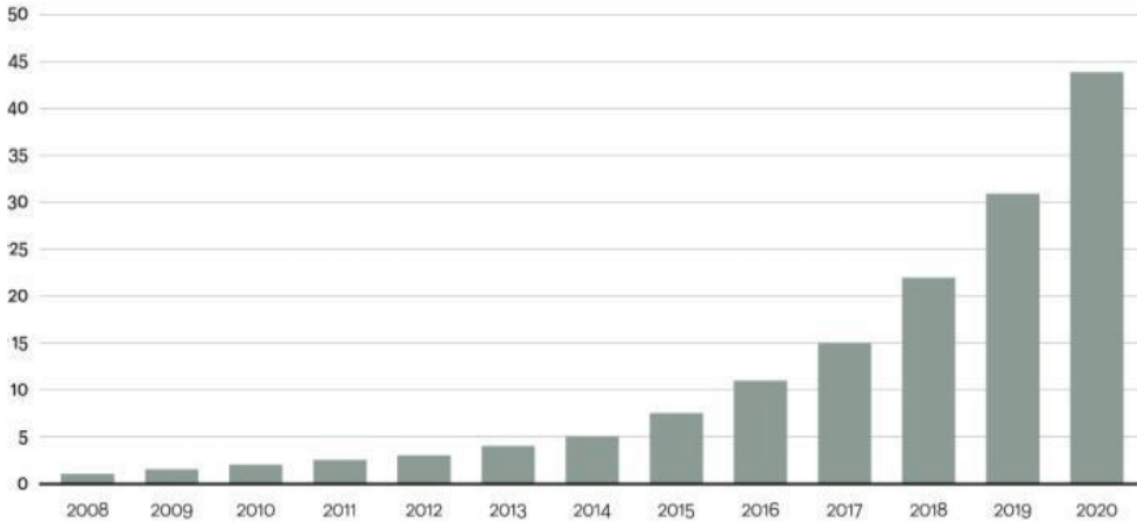
Makine öğrenmesi; matematiksel ve istatistiksel yöntemler kullanarak mevcut verilerden çıkarımlar yapan, bu çıkarımlarla bilinmeyene dair tahminlerde bulunan, modelleme ve algoritmalardan oluşan yapay zekanın bir alt dalıdır [18]. Günümüzde verinin hacmi ve karmaşıklığının artmasıyla beraber, insan gözünün verilerdeki yapıyı öğrenmesi ve fark etmesi zorlaşmıştır. Büyük veri yığınları içinde yapıyı görmek ve ortaya çıkarmak için makinelere ve bilgisayarların işlem gücüne ihtiyaç duyulmaktadır.

Makine öğrenmesinde kullanılan her algoritma, kendilerine verilen her veriyi inceleyerek, verinin yapısını ve karakterini öğrenmeye çalışır. İncelenen her veri ile birlikte genelleme yapabilme yeteneği kazanarak, hangi koşullarda / etkenlerde ve hangi durumlarda ne gibi sonuçların oluştuğunu öğrenir.

Makine öğrenmesi modelleri, geçmişte öğrendiği ve bildiği durumları göz önüne alarak tahminler üretmektedir. İstatistiksel öğrenmede verilerden yorum ve çıkarım yapabilme ön planda iken, makine öğrenmesinde tahmin başarısı ön plandadır. Geçmişte öğrendiği verilere dayalı tahmin üreten makine öğrenmesi algoritmaları, gerçekleşen ile bilinen tahmin değerleri arasındaki farka dikkat ederek, yapıyı tekrar öğrenmeyi ve genelleme yapabilmeyi öğrenir. Yani makine öğrenmesi, gerçekleşen ile tahmin arasındaki farkları değerlendirerek bir sonuç üretir ve tahmin değerlerini geri bildirim yaparak günceller. Başarısız olduğu alanları geliştirmeye çalışır. Bu süreci birçok kez tekrarlayarak en doğru tahminleri üretir.

Son yıllarda hayatımıza büyük verinin girmesi ile beraber, klasik istatistik ve veri analizi yöntemlerinin yetersiz kalmasıyla, makine öğrenmesine ve derin öğrenmeye olan ilgi artmıştır. İnsan gözünün algılayamayacağı boyut ve karmaşıklıktaki verinin

karakterini anlamak ve mevcut olan yapıyı / ilişkiyi görmek zorlaşmıştır. Bu sebeple makine öğrenmesi algoritmalarında ve büyük veri analitiği konularında büyük gelişmeler olmuştur. Şekil 2.1 üzerinden, 2020 yılına kadar dünya genelinde mevcut verinin büyüme ivmesi ve boyutu gösterilmiştir.



Şekil 2.1 Zetabayt cinsinden yıllık veri hacmi [19]

Dünyadaki veri üstel bir şekilde büyürken, klasik veri analizi yaklaşımları yetersiz kalmaktadır. Bu sebeple veriyi analiz etme ve ilişkileri görebilmek için insanoğlu, bu yeteneği bilgisayarlara kazandırmaya çalışmıştır. Makine öğrenmesi algoritmaları, son yıllarda daha hızlı ve daha başarılı tahminler üretmesiyle, dünya genelinde daha popüler hale gelmiştir.

Şekil 2.2'de "Machine Learning" kelimesinin Google Trends kullanılarak, aranma sıklığı gösterilmiştir.



Şekil 2.2 Makine öğrenmesi kelimesinin Google üzerinde aranma eğilimi

2.1.1 Denetimli Öğrenme

Makine öğrenmesinde veriler, etiket durumuna göre 4'e ayrılır. Makine öğrenmesinde, herhangi bir girdi d eğerinin çıktısı biliniyorsa bu tip öğrenmeye denetimli / gözetimli / etiketli öğrenme denilmektedir. Günlük hayatta analistlerin çalıştığı birçok veri, etiketli veridir. Denetimli öğrenme basitçe, hangi koşullarda ya da durumlarda nasıl bir sonucun oluştuğunun bilindiği verilerdir. Diğer bir ifade ile tahmin edilmek istenilen değişkenin / bağımlı değişkenin geçmiş değerlerinin bilinmesidir. Regresyon ve sınıflandırma problemleri, denetimli öğrenmeye verilebilecek en doğru örneklerdendir.

Kredi kartı dolandırıcılığı tahmin problemi, denetimli öğrenmeye verilebilecek güzel örneklerden biridir; çünkü bu problem aslında ikili sınıflandırma problemidir. Geçmişte kredi kartı dolandırıcılığı (fraud detection) yapan kişilerin, geçmiş davranış verileri ve dolandırıcı olup - olmadığı bilinmektedir. Kredi kartı dolandırıcılığı tahmin problemi; kişilerin geçmiş davranış karakteri incelenip, dolandırıcı olup - olmadığı etiket verisine göre analiz edilerek, gelecekte bu davranış karakterini sergileyebilecek kişilerin belirlenmesidir.

Regresyon problemlerinde de yaklaşım bu şekildedir. Bu öğrenme yönteminde makine öğrenmesinde kullanılan algoritmalar, beklenen çıktı değeri üzerine eniyileme yaparlar.

2.1.2 Denetimsiz Öğrenme

Makine öğrenmesinde tahmin edilmek istenen değişken yoksa ya da veri seti sadece girdi değişkenlerinden oluşuyorsa, bu tarz öğrenme şekillerine denetimsiz / gözetimsiz / etiketsiz öğrenme denilmektedir. Kümeleme, çok boyutlu ölçekleme ve anomali tespiti denetimsiz öğrenmeye verilecek örneklerdir.

2.1.3 Yarı Denetimli Öğrenme

Yarı denetimli öğrenmede bir miktar etiketlenmemiş ve etiketlenmiş veri kullanılır. Çalışma mantığı denetimli öğrenmeye benzemektedir. Makine öğrenme algoritmaları, etiketli veriler ile beraber etiketsiz veriyi etiketlemeyi öğrenir.

2.1.4 Takviyeli Öğrenme

Takviyeli / pekiştirmeli öğrenmede, sistem ödül üzerinden çalışır ve deneme-yanılma yolu ile makine belirtilen hedefe ulaşmayı amaçlar. Makine geçmiş deneyimleri inceleyerek, en çok ödülün nasıl kazanıldığını öğrenmeye ve hatayı azaltmaya çalışır.

2.2 Derin Öğrenme

Makine öğrenmesinin alt başlıklarından biri olan derin öğrenme, insan beyninin yapısından ilham alan algoritmaları kullanarak, derin katmanlarda öğrenme işlemlerinin yapıldığı bir alt disiplindir. Karmaşık veriler ve görüntü verilerindeki deseni ve yapının karakterini anlama yeteneği ile son zamanlarda popülerlik kazanmıştır. Derin öğrenme algoritmaları, kendi başlarına kararlar alabilmeleri için Yapay Sinir Ağı oluşturularak katmanlar halinde veri modellenir ve yapıyı öğretilir.

Makine öğrenmesi ile derin öğrenmeyi birbirinden ayıran en önemli özellik, Yapay Sinir Ağları mimarisindeki katmanlardır. Öğrenme, derin katmanlarda gerçekleştiği için derin öğrenme olarak adlandırılmaktadır. Yapay zeka uygulama alanlarında

derin öğrenme modelleri; genellikle görüntü tanıma, nesne algılama / belirleme, ses tanıma, doğal dil işleme ve genetik gibi birçok farklı alanda kullanılmaktadır.

Derin öğrenme, makine öğrenmesinin bir alt türüdür. Derin öğrenmede, sıralı katmanlar birbirlerine bilgilerini aktararak daha iyi öğrenmeyi öğrenir ve yapıyı keşfeder. Derin öğrenme yöntemlerinde öğrenme şekli, genellikle temsili öğrenmedir. Öğrendikleri bilgiler ile veriyi temsil edebilme gücü artırılmaya çalışılır. Derin öğrenme problemleri, görüntü işleme alanlarında sıklıkla kullanılmaktadır. Görüntü işlemede gerek yapısı gerek ilişkilerin öğrenilmesi, karmaşık olduğu için derin öğrenme algoritmalarında çok sayıda öznitelik oluşturma işlemleri yapılmaktadır.

Derin öğrenmede, her katman geri yayılım yaparak bir önceki katmandan gelen çıktıları düzenler ve öğrenme iyileşir. Bazen bu öğrenme işlemi, ezberlemeye dönüşmektedir. Derin öğrenmede amaç yapıyı ezberlemek değil, genelleştirmeyi sağlamaktadır. Kısaca derin öğrenme, insan beyninin çalışma mantığına benzer bir şekilde, doğrusal olmayan ilişkileri çok sayıda katman kullanarak derin bir ağ mimarisi oluşturur ve girdi / çıktı verileri arasındaki doğrusal ve doğrusal olmayan modellerle formüle eder. Bu sebeple verinin boyutu ve miktarı artırıldıkça derin öğrenme yöntemlerinin daha başarılı olduğu gözlemlenmiştir.

Derin öğrenme algoritmalarında, nöronlar arasında birbirini izleyen sinyallerin ilettikleri bilgi ile karar mekanizması oluşturulmaya çalışılır. Derin öğrenme modellerinde, modelin sahip olduğu mimari özellikleriyle bilgi, nöronlar / birimler arasında aktarılmakta ve diğer katmanda bu bilgi güncellenmektedir. Böylece derin öğrenme algoritmaları, mimariye bağlı olarak performansları değişmekte ve modelin eğitim süresi etkilenmektedir.

Son yıllarda bilgisayarların veri işleme gücünün artmasıyla dünyada yapay zeka uygulama alanları ve derin öğrenme algoritmalarının popülerliği hız kazanmıştır. Artık birçok kurum ve kuruluş, derin öğrenme tabanlı modeller kullanmaktadır. Dünyada özellikle son yıllarda derin öğrenmenin bu kadar gelişmesinin temel sebeplerinden biri, teknolojinin ve bilgisayarların işlem güçlerinin ciddi oranda gelişmesidir.

GPU ve CPU'da yaşanan gelişmeler sayesinde bilgisayarlar artık aynı anda çok fazla işlemi bir arada yapabilmektedir. Derin öğrenme algoritmaları, verinin karmaşıklığı ve çok sayıda işleme ihtiyaç duyması sebebiyle, bilgisayarların donanımsal özelliklerine bağlı kalmaktadır. Derin öğrenme problemlerinde, yüzbinlerce ya da milyonlarca öznitelik çıkarımı yapılmaktadır. Geliştirilen model, bu öznitelikleri kullanarak verinin sahip olduğu yapıyı öğrenmeye çalışmaktadır. Bu nedenle derin öğrenme algoritmaları, bilgisayar gücüne ihtiyaç duymaktadır.

2012 yılında İletim Sönümleme [13], 2014 yılında Çeklişmeli Üretken Ağlar [20] ve üretken ağların geliştirilmesi ile dünya genelinde, derin öğrenme konusuna büyük bir ilgi oluşmuştur. Şekil 2.3'te "Deep Learning" kelimesinin Google Trends kullanılarak, aranma sıklığı gösterilmiştir.



Şekil 2.3 Derin öğrenme kelimesinin Google üzerinde aranma eğilimi

Şekil 2.3 üzerinden de anlaşıldığı üzere 2014 yılında, dünya genelinde derin öğrenme kelimesinin aranma sıklığında ciddi bir artış görülmüştür.

2.3 Transfer Öğrenme

Makine öğrenmesi ve derin öğrenme modelleri, bir problem için geliştirildiğinde geçmişte öğrendiği bir bilgiyi, başka bir yerde kullanabilir. Günlük hayatımızda insanlar, öğrendiği herhangi bir bilgiyi başka bir durumda ya da koşulda kullanabilmektedir. İlişkiyi kurabilmek için problemlerin benzer olması, insanlar için yeterlidir. Geçmiş deneyimlerden öğrenilen bir bilgiyi / gerçeği başka bir

konuya aktarabilir. Böylece hem bilgi aktarımı yapılmış hem de yeni bir problemle karşılaşıldığında ne yapılması gerektiği biliniyor olur. Geçmiş bilgi saklanıp, gelecekte benzer bir problemi çözmek için kullanılabilir [21].

Transfer öğrenme / öğrenme aktarımı, makine öğrenmesi ve yapay zeka alanlarında sıklıkla kullanılan bir yöntemdir. Bunun sebebi, hem eğitim süresinin kısalması hem de daha az eğitim verisi kullanılarak daha başarılı modellerin geliştirilmesine imkan vermesidir. Transfer öğrenme kısaca, önceden geliştirilmiş bir modelin geçmiş bilgilerini depolayıp benzer bir problemle karşılaşıldığında bu bilgi deposunun kullanılmasıdır. Model her şeyi sıfırdan öğrenmez, öğrenme aktarımının kullanılmadığı çalışmalarda, model her şeyi sıfırdan öğrenir. Araştırmacının çözmek istediği probleme göre geçmiş bilgiler, geliştirilen model ile tekrardan güncellenir. Bu işlem model eğitimi süresini büyük ölçüde azaltır.

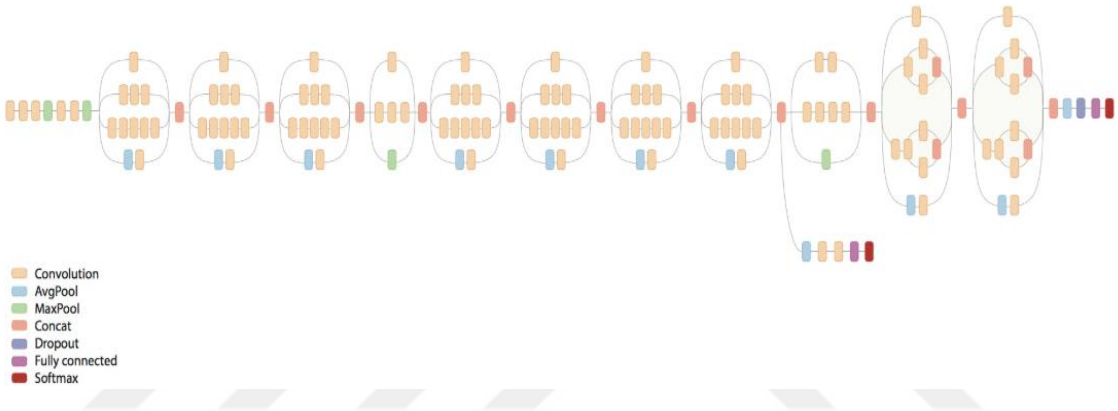
Öğrenme aktarımının avantajlarından biri de geçmişte öğrenilen veriden, ilgili bilgi ve özellikler, araştırmacının hedef aldığı çalışmalarda kullanılabilir olmasıdır. Transfer öğrenme ile beraber daha önceden öğrenilmiş bilgi, yeni kurulan modele aktarılır. Bu sebeple, adına transfer öğrenme yani öğrenme aktarımı denilmektedir. Yeni kurulan modelin eğitim süresini kısaltan ve model performansını artıran bir yöntemdir. Görüntü işlemede CNN algoritmaları, görüntü üzerinden çok fazla sayıda öznitelik çıkarımı yapar. Farklı mimariler ile kurulan derin öğrenme algoritmaları, milyonlarca öznitelik çıkartır. Bu öznitelikler arasında modellerin, veri setini anlayabilmesi ve genelleştirebilmesi için büyük maliyetlere ihtiyaç duyar.

Görüntü işleme modellerinin eğitildiği büyük ve zorlu veri setleri göz önüne alındığında, öğrenme aktarımının bu alan için çok önemli olduğu görülmekte ve sıklıkla kullanılmaktadır. Görüntü verilerini girdi olarak kullanan modellerde, öğrenme aktarımı kullanmak son yıllarda yaygınlaşmıştır. Transfer öğrenmede özellikle ImageNet verisi üzerinden yapılan çalışmalar, veri bilimi sektöründe çokça kullanılmaktadır.

ImageNet verisi yaklaşık 1.000 sınıflı görüntü verisidir. Görüntü tanıma ve sınıflandırma sistemlerinde derin öğrenme modellerinin geliştirilebilmesi için yaygın olarak kullanılan akademik bir veri setidir. Bu veri seti ile eğitilmiş modellerin bilgileri saklanarak, farklı görüntü işleme problemlerinde

kullanılmaktadır. ImageNet alt yapısı bulunan modellerin sakladıkları bilgiler indirilebilir ve yeni modellerde girdi olarak kullanılabilir. Bu sebeple öğrenme işlemi sıfırdan yapılmaz.

Öğrenme aktarımında önemli bir diğer konu ise, hangi veri seti üzerinde model kurulduğudur. ImageNet gibi farklı veri setleri vardır. Diğer önemli konu ise, eğitilmiş derin öğrenme modellerinin mimarisidir. Günümüzde araştırmacının problemine bağlı olarak VGG, Inception ve Resnet tabanlı mimariler ve veri setleri vardır. Şekil 2.4 üzerinden derin öğrenmede kullanılan Inception V3 mimarisi gösterilmiştir.



Şekil 2.4 Inception v3 derin öğrenme mimarisi [22]

Derin öğrenme ve görüntü işleme problemlerinde sıklıkla kullanılan transfer öğrenme ve derin öğrenme mimarileri, aynı zamanda metin analizi gibi alanlarda da sıklıkla kullanılmaktadır. Doğal Dil İşleme (NLP) çalışma alanlarında da öğrenim aktarımının kullanımı yaygındır. NLP çalışmalarında, görüntü işlemede de kullanıldığı gibi derin öğrenme mimarileri kullanılmaktadır.

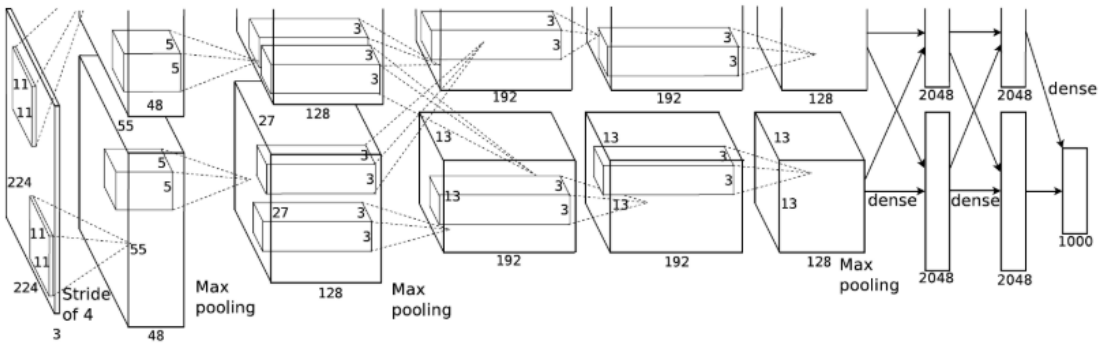
Google'ın geliştirmiş olduğu "word2vec" ya da "GloVe" gibi yöntemler, uzun süredir kullanılmaktadır. 2018 Ekim ayında Google, etkin bir şekilde transfer öğrenme yöntemlerinin kullanıldığı ve NLP için geliştirdikleri "Bert" algoritmasını tanıtmışlardır. Susam Sokağı çizgi dizisi karakterlerinden olan Büdü (orjinalinde Bert), çalışmaya ismini vermektedir. İki Yönlü Kodlayıcı Transformatörü (Bert), Google ekibi tarafından arama sonuçlarının eniyilemesi amacıyla geliştirilmiş bir yöntemdir. Bert, 24 transformer bloğu, 1024 gizli katmanlı ve 340 Milyon parametresi bulunan bir algoritmadır [23][24].

2.4 ImageNet Veri Seti

ImageNet, 2012 yılından bu yana Büyük Ölçekli Nesne Tanıma Yarışması (ILSVRC) bilgisayarlı görü projelerinde ve nesne tanıma araştırmalarında kullanılması için tasarlanmış, büyük bir görsel veritabanıdır [25].

Görüntü tanıma / sınıflandırma problemlerinde araştırmacının elinde yeterince veri seti bulunmuyorsa, model performansını artırmak için transfer öğrenme yöntemlerinin uygulanması avantaj sağlamaktadır. ImageNet veritabanı 15 milyon farklı görüntüden oluşan, binlerce farklı nesne kategorisine sahip bir görsel veri setidir. Bu veri seti, 2012 yılında yapılan yarışma ile dikkatleri üzerine çekmiştir. Günümüzde çeşitli görüntü işleme algoritmalarının ve çeşitli mimarilerin başarılarının test edildiği, araştırmacıların ImageNet verisi sınıflandırma başarısına göre kendi yöntemlerini ölçümleyebildiği büyük bir veritabanı olmuştur.

Derin öğrenme ilk olarak 1998 yılında Yann LeCun ve arkadaşlarının yayınlamış olduğu çalışma [26] ile ilk uygulaması gerçekleştirilmiştir. Fakat derin öğrenme algoritmalarının ve bilgisayar teknolojilerinin gelişmesiyle derin öğrenme, 2010 yılında sonra popüler olmuştur. 2012 yılında gerçekleştirilen ImageNet yarışması, dünya çapında duyulmasına katkıda bulunmuştur. 2012 yılında düzenlenen Büyük Ölçekli Nesne Tanıma Yarışması'nı AlexNet modeli kazanmıştır [27].



Şekil 2.5 Alexnet mimarisi [28]

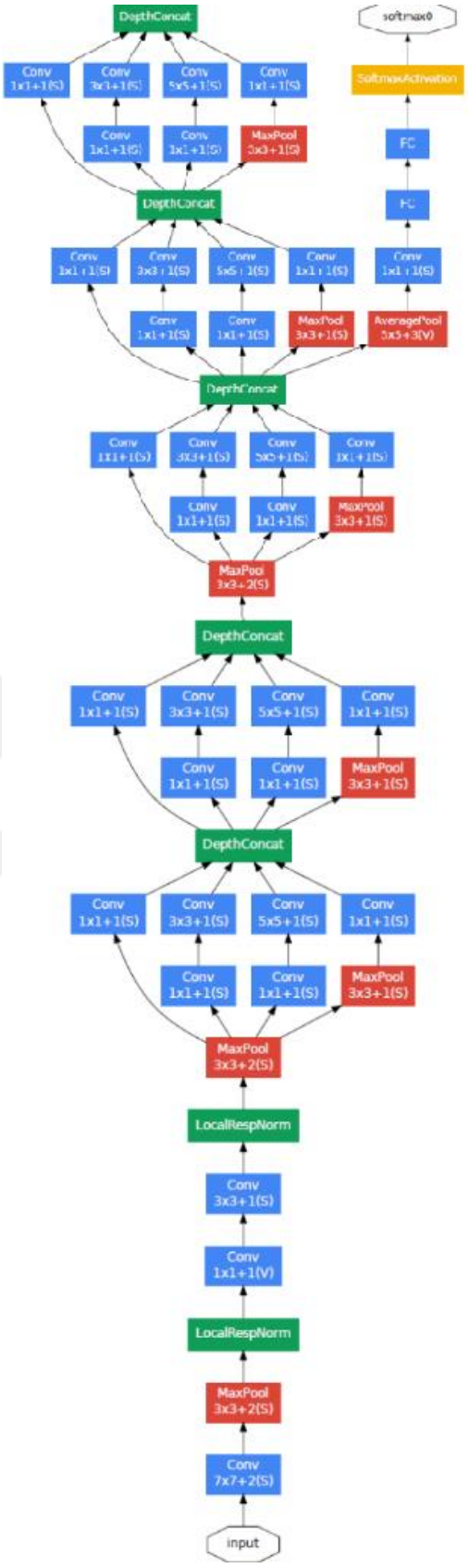
2013 yılında düzenlenen yarışmada ise ZFNet modeli birinci olmuştur [29]. AlexNet ilk konvolüsyon filtreleme katmanında 11 x 11'lik filtreleme kullanmasına karşın, ZFNet 7 x 7 boyutunda filtre kullanmıştır. ZFNet modelinde aktivasyon fonksiyonu

olarak Relu ve model eğitimi için Stokastik Gradyan İnişi (SGD) kullanılarak, 12 gün boyunca eğitilmiştir.



Şekil 2.6 Zfnet mimarisi [29]

2014 yılında düzenlenen ImageNet yarışmasını GoogLeNet mimarisi kullanılarak Christian Szegedy ve arkadaşları kazanmıştır [30]. GoogLeNet mimarisi, 22 katmana sahip, ardışık bir yapıda konvolüsyon ve havuzlama katmanlarını üst üste kullanmaktan uzaklaşan ilk Evrişimsel Sinir Ağı mimarilerinden birisidir. [27]. GoogLeNet 5 milyon parametreye ve AlexNet'e göre kıyaslandığında 12 kat daha az parametreye sahiptir.



Şekil 2.7 Googlenet mimarisinin bir kesiti [30]

2015 yılında yapılan ImageNet Büyük Ölçekli Nesne Tanıma Yarışması'nın kazananı Xiangyu Zhang ve arkadaşları olmuştur [31]. Xiangyu Zhang ve arkadaşları, Microsoft ResNet mimarisi kullanmıştır. Residual Network (ResNet) mimarisi, diğer derin öğrenme mimarilerine göre daha büyük ve daha derin katmanlı tasarlanmıştır. ResNet mimarisi, artık bağlantıları kullanır. Resnet modelleri artık değerleri kullanarak, kendinden bir sonraki katmanları besleyen bloklara eklenmesiyle oluşmaktadır. Bu sebeple daha önceki mimarilerden daha farklı bir yapıya sahiptir.

Tablo 2.1 2012-2017 yılları arasında ILSVRC yarışması sonuçları [32]

Yıl	Algoritma	Başarım(Top–5 Doğruluk)
2012	AlexNet	%84.7
2013	ZFNet	%85.2
2014	VGGNet	%92.7
2014	GoogLeNet	%93.33
2015	ResNet	%96.43
2016	Trimp	%97.01
2017	SENet	%97.75

2.5 Keras

Keras, 2015 yılında François Chollet tarafından Python'da geliştirilmiş açık kaynak kodlu yapay zekada en sık kullanılan sinir ağı kütüphanelerinden biridir [33] ayrıca farklı modülleri bir arada sunmaktadır.

Keras, popüler bir sinir ağı modeli geliştirme API'lerinden biridir ve TensorFlow, R, Microsoft Cognitive Toolkit, Theano ya da PlaidML teknolojileri ile birlikte çalışabilir. Kullanıcı dostu olması, daha basit Yapay Sinir Ağı modelleri oluşturulabilmesi, modüler ve geliştirmeye açık olması, makine öğrenmesi alanında çokça kullanılmasının alt sebeplerinden bazılarıdır.

Keras kullanılarak sıralı derin ağ modellerinin oluşturulması kolay olduğu için derin öğrenme çalışmalarında sıkça kullanılmaktadır. Araştırmacı derin ağ modelleri

oluşturmak istediğinde, modelin mimarisini kolaylıkla belirleyebilmektedir. Doğrusal olmayan ve ilişkilendirmenin zor olduğu veri setlerinde derin öğrenme modelleri, yapıyı anlamak için kullanılmaktadır. Keras'ın araştırmacıya büyük bir kolaylık sunduğu, model mimarisinin kolayca değiştirilebildiği, sınıflandırma ya da regresyon problemlerini kolaylıkla belirleyebildiği, aktivasyon ve maliyet fonksiyonlarını kişiselleştirebildiği bir ortam sağlar. Ek olarak Google, Microsoft, Amazon, Apple, Nvidia ve Uber tarafından da desteklenmektedir.

```
from keras.models import Sequential
from keras.layers import Conv2D, MaxPool2D, Dense, Flatten, Dropout

model = Sequential([
    Conv2D(filters=32, kernel_size=(5,5), activation='relu', input_shape=x_train.shape[1:]),
    Conv2D(filters=32, kernel_size=(5,5), activation='relu'),
    MaxPool2D(pool_size=(2, 2)),
    Dropout(rate=0.25),
    Conv2D(filters=64, kernel_size=(3,3), activation='relu'),
    Conv2D(filters=64, kernel_size=(3,3), activation='relu'),
    MaxPool2D(pool_size=(2, 2)),
    Dropout(rate=0.25),
    Flatten(),
    Dense(256, activation='relu'),
    Dropout(rate=0.5),
    Dense(10, activation='softmax')
])
```

Şekil 2.8 Örnek bir keras derin öğrenme modeli

Şekil 2.8'de 10 farklı kategoriye ait bir sınıflandırma modeli oluşturulmuştur. Sınıflandırma işleminde softmax aktivasyon fonksiyonu tercih edilmiş ve evrişim katmanlarında relu aktivasyon fonksiyonu kullanılmıştır. 2 kez en büyük ortaklama yapılarak veri matrisinin boyutu düşürülmüştür.

2.6 Tensorflow

Tensorflow, 2015 yılında Google tarafından geliştirilen, makine öğrenmesi ve sinir ağları gibi algoritmaların / yöntemlerin olduğu ve veri biliminde en sık kullanılan açık kaynaklı bir yazılım kütüphanesi ve veri çerçevesidir [34]. Tensorflow'un bu

kadar popüler olmasının temel sebeplerinden biri, Tensorflow kullanarak uygulama oluşturmanın ve model geliştirmenin kolaylığıdır.

Tensorflow, sayısal hesaplamalarda ve büyük ölçekli makine öğrenmesi problemlerinde kullanılan güçlü bir veri çerçevesi (dataframe) ve Python kütüphanesidir. Tensorflow olarak adlandırılmasının sebebi ise tensor olarak isimlendirilen çok boyutlu dizileri, n boyutlu matris, girdi olarak alır ve istenilen / belirtilen akıştan geçirir, ardından çıktı olarak araştırmacıya sunar. Tensorflow ile eğitilen derin ağ yapılarının model gelişimi, Tensorboad kullanılarak grafiğe dönüştürülebilir. Bu durum araştırmacıya büyük kolaylık sağlar ayrıca CPU, GPU ve TPU üzerinde çalışabilir.

Tensorflow, ses tanıma, görüntü işleme, Google Translate çeviri, metin tabanlı analizler ve video algılama alanlarında yaygın olarak kullanılmaktadır. Tensorflow'un kullanılmasının önemli sebeplerden bir diğeri ise, dünya çapında bir ekosisteme sahip olmasıdır. Araştırmacı model geliştirirken karşılaştığı bir hatayı internette aratarak, çözümü kolayca bulabilmektedir.

Derin öğrenmenin ve makine öğrenmesinin son zamanlarda yaşadığı büyük gelişme ile yapay zeka, büyük bir etki alanı ve finansal katma değer oluşturmuştur. 2013 yılında Oxford Üniversitesi'nin yaptığı bir araştırmada, gelecekte yapay zekanın Amerika'daki tüm işlerin yaklaşık yarısını alabileceği öngörülmüştür. Şu an dünyadaki hemen hemen her teknoloji şirketinde veri bilimciler çalışmaktadır. Özellikle pazarlama, araştırma ve geliştirme merkezlerinde büyük verilerin analizine ve anlamlandırılmasına ihtiyaç vardır ve bu ihtiyaç, verinin üstel bir şekilde büyümesiyle artmaktadır.

Büyük verinin derinlemesine analiz edilebilmesi için güçlü bir bilgisayar donanımına ihtiyaç duyulmaktadır. Büyük veri analitiğinde genelden özele ya da özelden genele gibi yaklaşımlarla veriler analiz edilmektedir. Veri biliminin tarihi incelendiğinde, 1962'lere dayanmaktadır. John Tukey, Veri Analizinin Geleceği yazısında [35] şöyle belirtmiştir, "Kendimi uzun süre özelden genele doğru çıkarımlar yapan bir istatistikçi olarak düşünürdüm. Matematiksel İstatistik'teki gelişmeleri merakla ve şüpheyle seyrettim. Veri analizi, deneye dayalı bir bilimdir.

Elektronik bilgisayarlar, veriyi artırmıştır ve hiç şüphesiz veri analitiğinde bilgisayarların büyük bir önemi vardır.”

2.7 Bilgisayarlı Görü için Derin Öğrenme

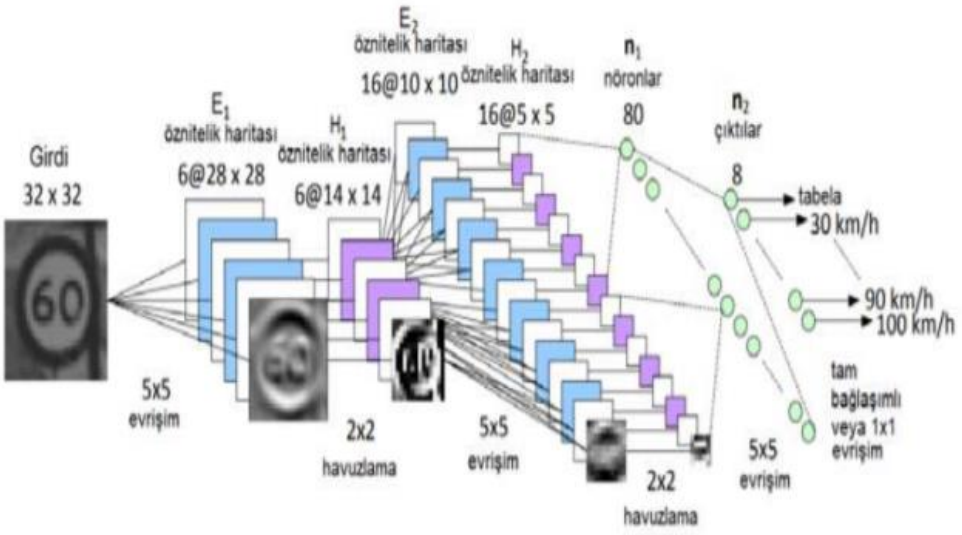
2.7.1 Evrişimsel Sinir Ağları (CNN)

Yapay zekada kullanılan derin öğrenme algoritmaları geliştirilirken, insan beyninden ilham alınmıştır. Bir ya da daha fazla evrişim işlemi gerçekleştirilen sinir ağlarına, Evrişimsel Sinir Ağlar denmektedir. Evrişimsel Yapay Sinir Ağları, 3 renk Kırmızı, Yeşil ve Mavi (RGB) kanallarından matrisler oluşturarak, görüntü pikselleri dizisi üzerine çeşitli filtreler kullanarak, görüntünün içerdiği yapıyı ve ilişkiyi anlamaya çalışır. Evrişimsel Sinir Ağları, girdi katmanı, gizli katmanlar, evrişim / filtreleme katmanları, havuzlama katmanları ve tam bağlı katmanlarından oluşmaktadır. Bunlarla beraber normalizasyon, düzleştirme ve iletim sönümleme gibi farklı işlemler de uygulanabilmektedir.



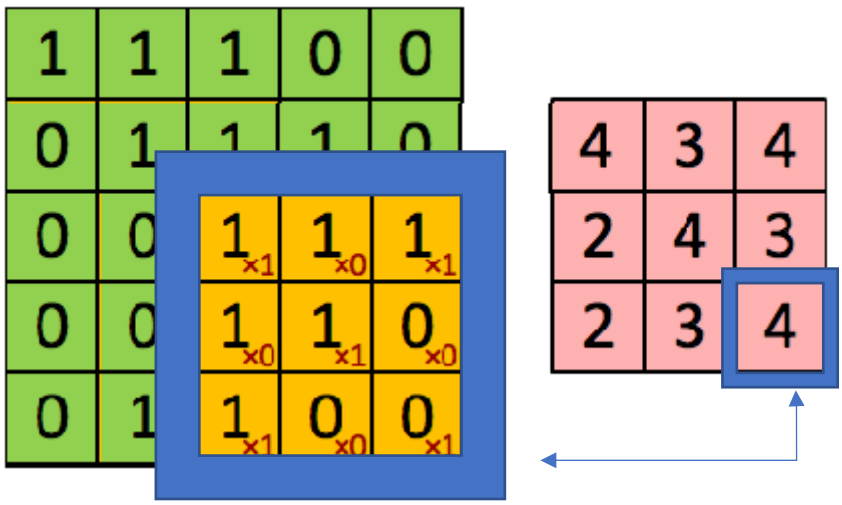
Şekil 2.9 Üç renk kanalı

Evrişim katmanı, ilk LeCun [15] tarafından belirtilmiş ve matris işlemleri yerine evrişim işlemlerinin gerçekleştirildiği katmandır. Evrişim katmanları üç adımdan oluşmaktadır. Bunlardan ilki, evrişim işleminin uygulandığı katmandır. İkinci katman ise doğrusal olmayan verilere aktivasyon fonksiyonunun gerçekleştirildiği ya da havuzlama gibi işlemlerin yapıldığı katmandır. Son katmanlarda ise tam bağlantı ve düzleştirme işlemleri uygulanarak çıkış katmanına iletilmektedir. Akabinde sınıflandırma işlemi yapılmaktadır.



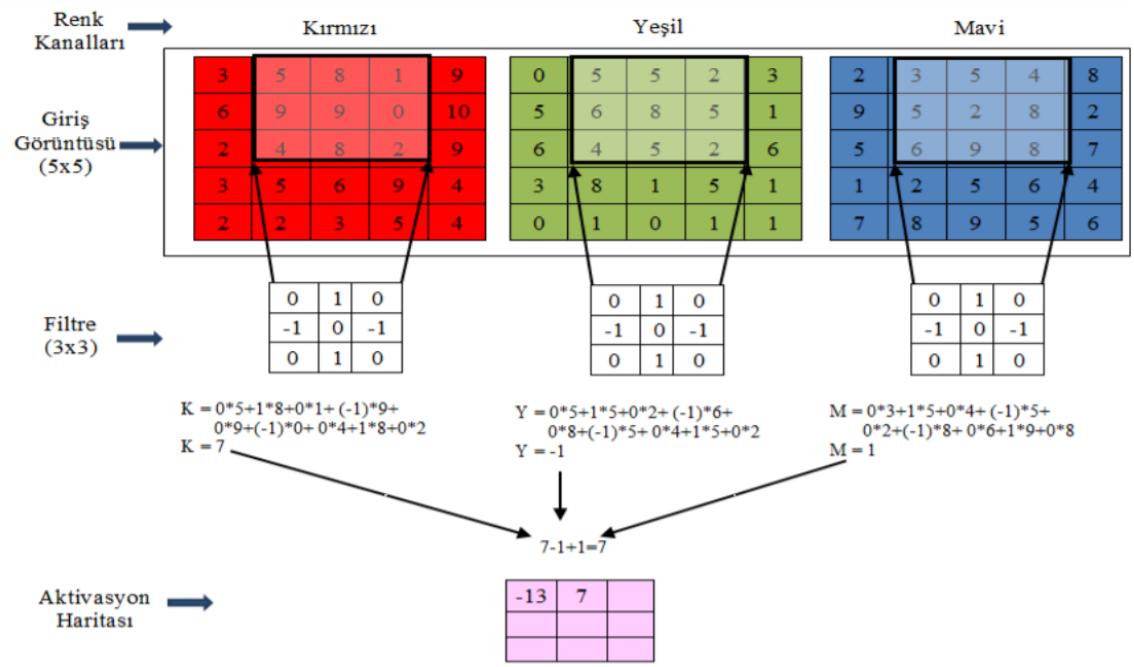
Şekil 2.10 Esa yapısı [32]

Evrişimsel Sinir Ağları, görüntüleri çeşitli katmanlarla işler. Evrişim katmanı, Evrişimsel Sinir Ağları modellerinin en önemli adımlarından biridir. Çünkü bu katmanda, görüntü verisi içeren pikseller çeşitli filtrelemeler ile veriyi matris olarak kullanır ve özellik haritası oluşturur. Özellik haritasını oluşturmak için evrişim işlemleri uygulanır. Evrişim katmanlarında filtrelerin oluşturulmasında 3 x 3, 5 x 5 ve 7 x 7 boyutlarının kullanımı yaygındır. Evrişim katmanında, görüntü verisindeki özellikler tespit edilir ve öznitelikler çıkartılır. Bu çıkarılan öznitelikler, renkler, eğrilikler, şekiller ve kenarlar gibi özellikler olabilir.



Şekil 2.11 3x3 boyutunda filtre katmanı

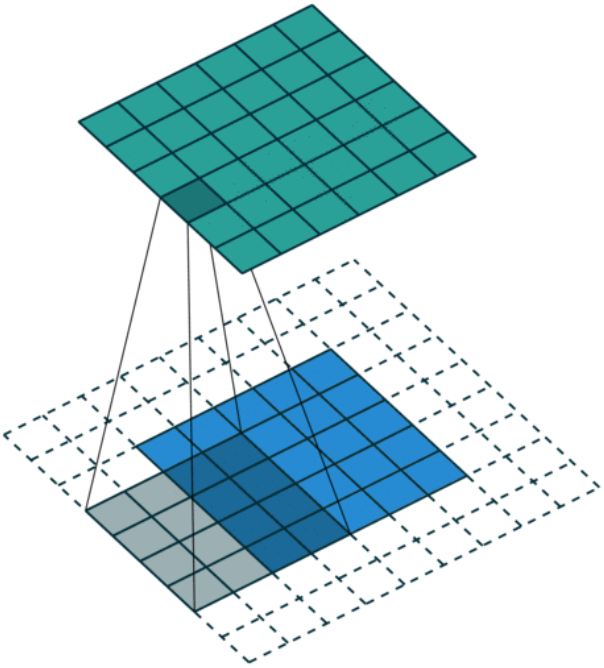
Evrişim işlemlerinde, görüntü verisi matrislere dönüştürülerek filtre veya diğer adıyla çekirdek ile aynı boyuta sahip, yerel bir bölgenin iç çarpım işlemleri gerçekleştirilir ve çıkış değeri hesaplanır. Bu çıkış değeri tam sayı olmalıdır. Filtreleme işlemleri, adım uzunluğu (stride) üzerinden yapılır. Stride kelimesi genellikle padding / piksel ekleme / dolgu terimi ile kullanılır. Stride kısaca, görüntü verisinden alınmış ve filtre katmanı uygulanmış bir giriş görüntüsünün, nasıl evriştiğini denetler ve kaydırma adımını belirler ve tüm bu işlemler tekrarlanır. Tüm bu kaydırma işlemleri, görüntü matrisindeki değerler için yapılır. İç çarpım değerlerinin sonucu negatif olabilir. Bu sebeple evrişim katmanında ReLU aktivasyon fonksiyonu kullanılabilmektedir [36].



Şekil 2.12 3 x 3 boyutunda filtrenin görüntü verisine uyguladığı konvolüsyon işlemi [27]

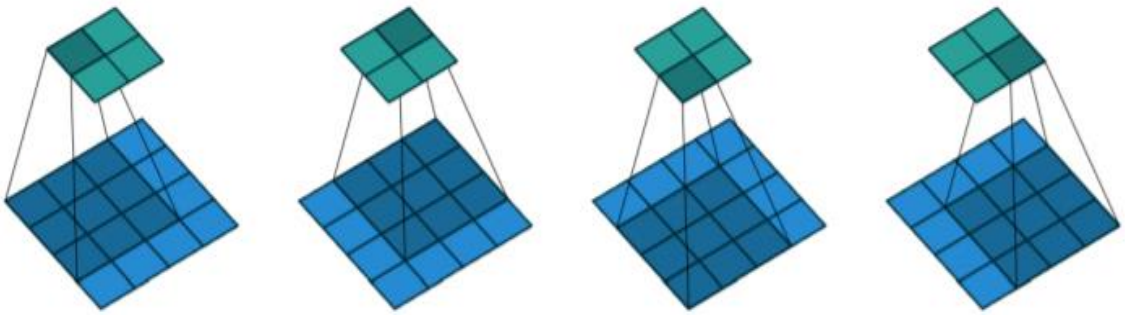
Evrişim katmanında ilk filtre katmanından sonra Evrişimsel Sinir Ağları bir özelliklik haritası (feature map) oluşturur. Özellik haritası ile görüntünün nasıl bir özellik türüne sahip olduğu belirlenir. İkinci filtre katmanı ile özellikler başka özellik türlerini algılar ve ikinci özellik haritasını oluşturur. Stride değeri, özellik haritasının boyutunu etkiler. Örneğin adım sayısı 2 seçildiği durumda, evrişim işlemi sonrasında filtre 2 piksel kayacaktır.

Evrişim katmanlarından sonra verinin giriş ve çıkış boyutlarını yönetmek, araştırmacının elinde olan bir durumdur. Verinin giriş ve çıkış boyutunun aynı olmasının istendiği durumlarda padding işlemi uygulanır. Giriş matrisine fazladan piksel eklenerek, çıkış katmanı ile boyutlar eşitlenir.



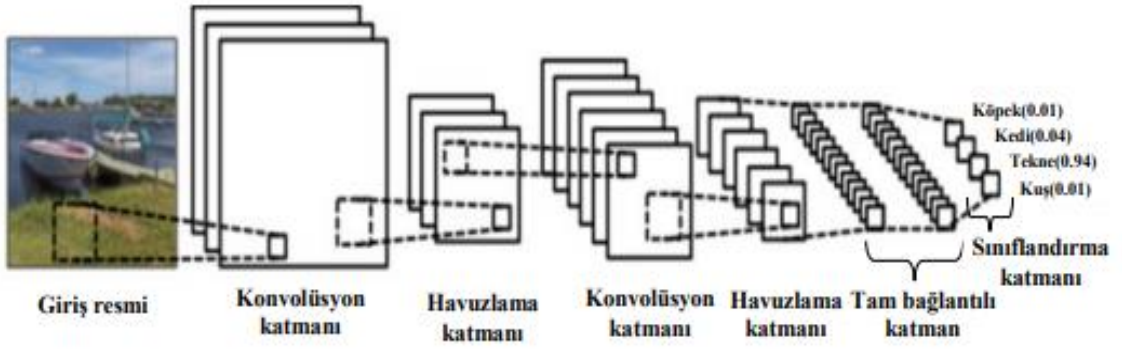
Şekil 2.13 Örnek dolgulu evrişim işlemi [37]

Padding işlemi, görsel verinin boyutlarını korumak için yapılmaktadır. Evrişim işlemi 2'ye ayrılmaktadır. Bunlar; geçerli ve aynı evrişimdir ve padding kullanımına göre ayrılır. Geçerli Evrişim, görüntü verisinin dönüştürülmediği anlamına gelir. Ayrıca, geçerli evrişim uygulandığında her evrişim adımıyla birlikte görüntü küçülecektir ve bilgi kaybına neden olacaktır. Bu sebeple sınırlı sayıda yapılmaktadır. Şekil 2.13'te dolgu kullanılan evrişim işlemi örnek gösterilmektedir.



Şekil 2.14 Örnek padding kullanılmayan evrişim işlemi [37]

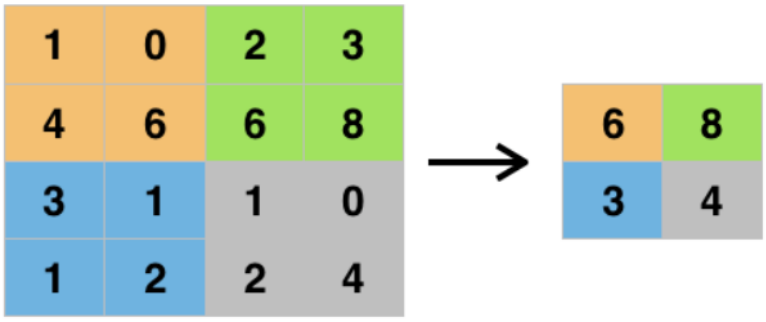
Aynı evrişim / yarım dolgu ise, padding uygulandığında girdi boyutunun dolgulandığı yani girdi ve çıktı boyutunun aynı olduğu anlamına gelmektedir. Görüntü matrisine bir ya da birkaç piksel eklenerek, sıfır değerleri içeren, aynı evrişim işlemi gerçekleştirilebilir. Şekil 2.14'te dolgu kullanılmayan evrişim işlemi örnek gösterilmektedir.



Şekil 2.15 Esa katman sıralaması ve örnek mimarisi [27]

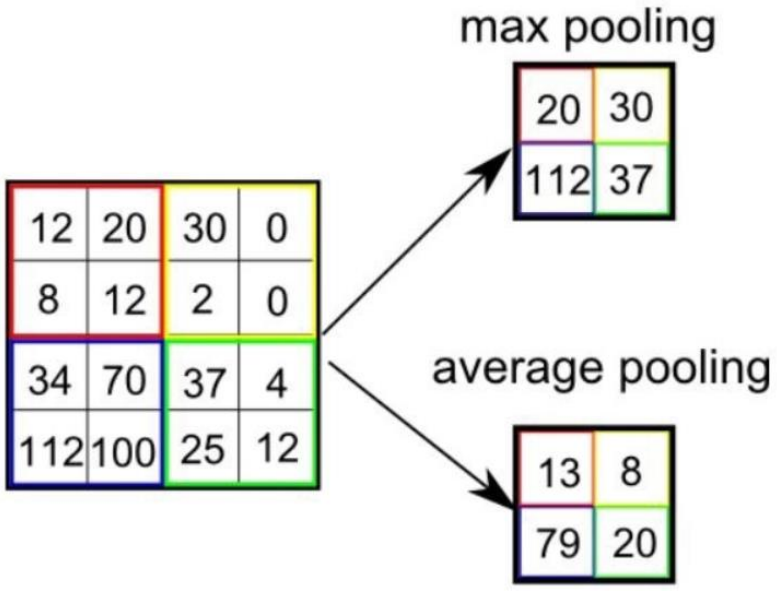
Evrişimsel Sinir Ağları'nda evrişim katmanlarından sonra havuz katmanı bulunmaktadır. Havuz katmanında havuzlama / ortaklama işlemi yapılmasıyla beraber, filtre katmanlarından gelen bilgiler özetlenecek şekilde temsil edilir ve bu bilgiler, bir sonraki katmana girdi olarak girer. Ortaklama katmanının amacı, alt-örnekleme (downsampling) işleminin gerçekleştirilmesidir ve ortaklama katmanıyla görüntü verisinin boyutu azaltılır ve işlemleri hızlandırmak için kullanılır.

Veri özetlenirken, görüntü matrisi bölgelere bölünerek her parçanın maksimum veya ortalama değeri alınır ve yeni bir matrise eklenir. Bu işleme Max Pooling ya da Average Pooling denmektedir. En popüler ortaklama yöntemi ise bu bölgelerdeki değerlerin maksimumunu alma olan max pooling yöntemidir. 4 x 4 boyutundaki veri 2 x 2'lik kısımlara bölünerek toplamda 16 değeri olan bir görsel ortaklama işlemiyle 4 değere dönüştürülür.



Şekil 2.16 Maksimum ortaklama işlemi

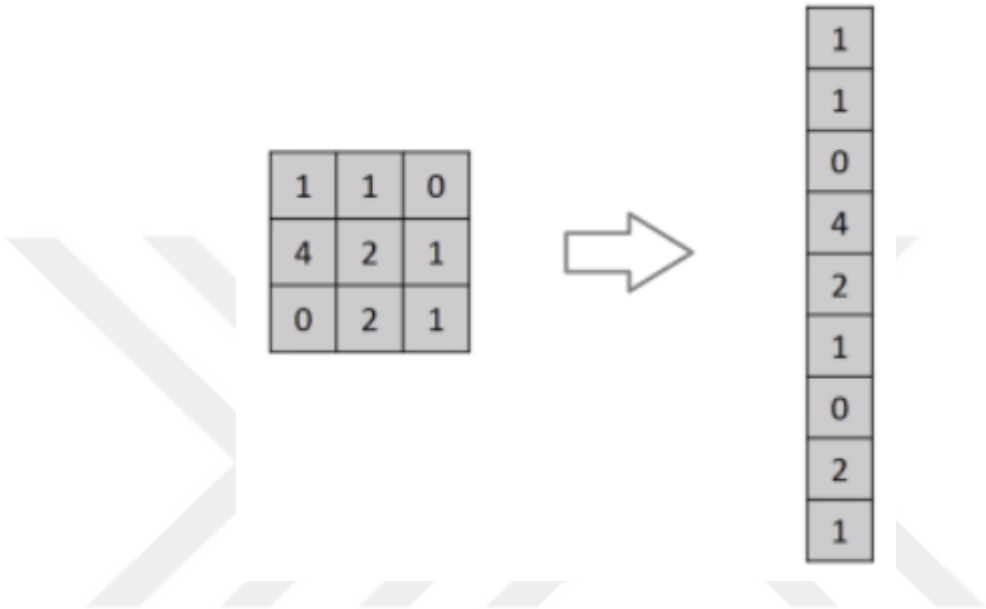
Şekil 2.16'da gösterildiği gibi, parçalara bölünen her bölgenin maksimum değeri alınarak veri küçültülür ve özetlenir. n x n boyutundaki veri matrisi, ortaklama işlemiyle m x m boyutuna indirilir. Böylelikle, Evrişimsel Sinir Ağları'nın hesaplama maliyeti azalır ve veri yeterli bilgiyi içermeye devam etmesiyle birlikte daha küçük matrislerle işlemler yapar ve bir sonraki katmana daha küçük bir girdi gönderir.



Şekil 2.17 Maksimum ve ortalama ortaklama işlemi [38]

Evrişimsel Sinir Ağları'nda ortaklama katmanında öğrenme gerçekleştirilmez. Evrişim katmanında olduğu gibi belirlenen büyüklükteki filtre, tüm görüntü matrisinde dolaşır. Kanal sayısı sabit tutularak, matris verisinin boyutu (Genişlik x Yükseklik) azaltılır. Kapsül Teorisine göre, veri özetlendiği için bilgi kaybına sebep olmaktadır.

Evrişimsel Sinir Ağları'nda evrişim ve ortaklama katmanlarından sonra tam bağlantı katmanı (fully connected layer) gelmektedir. Evrişimsel Sinir Ağları mimarilerinin çoğu tam bağlantı katmanı ile sona erer ve ardından aktivasyon fonksiyonu ile görüntü verisi sınıflandırılır. Tam bağlantı katmanında, daha önceden çıkarılmış özellik haritaları işlenir ve üst düzey özellik haritaları kullanılarak görüntü sınıflandırma gerçekleştirilir.



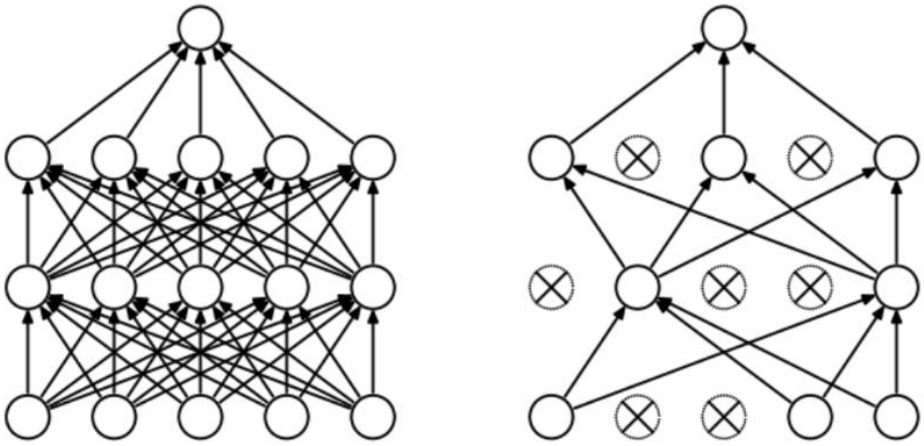
Şekil 2.18 Düzleştirme (flattening)

Tam bağlantı katmanına girdi olarak gelen veri matrisi ise düzleştirme (flattening) adımından gelmektedir. Ortaklama ve filtreleme katmanlarında çeşitli işlemlerden geçen çok boyutlu veri matrisi, düzleştirme işlemi uygulandıktan sonra tam bağlantı katmanına girer. Tam bağlantı katmanı, farklı derin öğrenme mimarilerine göre bağlantı sayısı değişse bile bu katmanda, kendisinden önceki tüm katmanlara bağlanır. Ayrıca tam bağlantı katmanında her giriş, tüm nöronlara bağlandığı katmandır. Evrişimsel Sinir Ağları'nın özet çalışma prensibi bu şekildedir.

2.7.2 İletim Sönümleme (Dropout)

Derin öğrenmede asıl amaç, görüntü verisinin sahip olduğu ilişkileri ve yapıyı öğrenerek, veriyi genelleştirmektir. Evrişimsel Sinir Ağları'nda ya da diğer ağ türlerinde de amaç bu şekildedir. Öğrenme işlemi yapıldığında veriyi temsil edecek bilgi öğrenilir. Bazı durumlarda model ya da ağ, veriyi öğrenmekten ziyade ezberler.

Derin öğrenmenin amacı ise genelleştirme yapmaktır. Modelin ezberleme yapmaması için iletim sönümleme / seyreltme adımı kullanılır. Kısaca iletim sönümleme değeri 0-1 arasında değer alan ve bu orana göre bazı düğümlerin etkisini yitirmesidir. Her tekrarda (epoch) rassal olarak farklı düğümlerden gelen bilgiler sıfıra eşitlenir. Böylelikle ağ, veriyi ezberlemek yerine öğrenir.



Şekil 2.19 İletim sönümleme (dropout)

Derin katmanlarda öğrenmenin gerçekleştiği derin öğrenmede, çeşitli katmanlarca matris haline çevrilen görüntü verisinin, modelleme yapabilmek için nöron adı verilen birimlerin yapıyı öğrenmesi gerekmektedir. Nöronlar çeşitli ağırlık değerleri ile birlikte daha iyi tahmin yapabilme yeteneği kazanır. Daha iyi tahmin yapmak için ileri beslemeler ya da geri beslemeler ile aldıkları ağırlık değerleri güncellenir ve öğrenme devam ettikçe daha başarılı tahminler üretir.

İleri ve geri besleme ile güncellenen ağırlıklar zamanla yapıyı genelleştirmekten ziyade ezberlemeye başlar. Bu duruma aşırı uyum / ezberleme denir. Makine öğrenmesinin en önemli konularından biri olan ezberleme durumundan kaçınmak için derin öğrenme katmanları arasında iletim sönümleme işlemi uygulanır. İletim Sönümleme 0 ile 1 arasında değer alır ve belirlenen oran arasında nöronlardan gelen ağırlıklar etkisiz hale gelir yani sıfır ağırlığı verilir. Diğer bir ifade ile belirtilen oran kadar rastgele nöron söndürülür. Derin öğrenmede, ezberlemeyi engellemek için sıklıkla iletim sönümleme kullanılır.

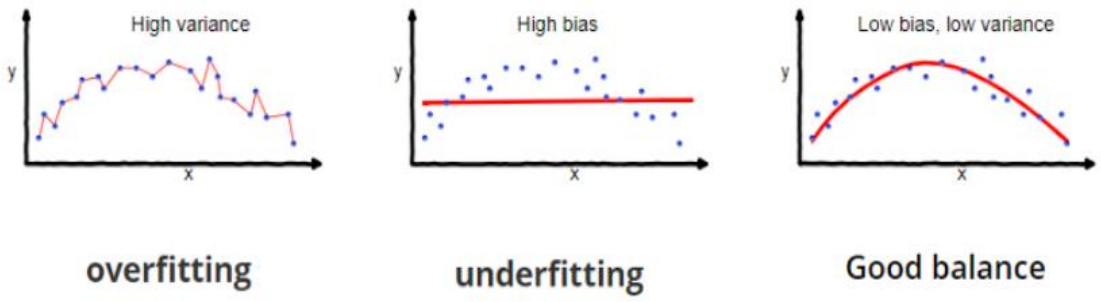
2.7.3 Az Öğrenme ve Ezberleme (Underfit ve Overfit)

Makine öğrenmesinde model geliştirirken araştırmacının en sık kontrol ettiği durum, modelin az öğrenmesi ya da veriyi ezberlemesidir. Araştırmacının geliştirdiği model, eğitim için kullandığı veri seti üzerinde çok çalışırsa, eğitimde kullanılan görüntülerdeki yapıyı ezberler ve onun dışındaki bir yapıyı bilemez bunun sonucunda modelin çalışma performansı düşmüş olur. Bu nedenle, eğitimde kullanılan verilerin sayısı bol ve tek düze olmamalıdır. Ezberlemeden kaçınmak için genellikle modele daha fazla veri eklenmesi literatürce önerilmektedir. Bununla birlikte iletim sönümleme, düzenlemeler, çapraz doğrulama ve modelin hiper paramatreleri değiştirilerek ezberleme ve az öğrenme durumu kontrol altında tutulmaya çalışılır.

Örneğin, görüntü sınıflandırma üzerine derin öğrenme algoritmalarından Evrişimsel Sinir Ağları kullanılarak bir model geliştirmek istenildiği durumda, modelin eğitim sürecindeki başarıları ile test verisindeki başarısı arasında ciddi bir fark varsa ezberleme ya da az öğrenme (underfit) durumu gerçekleşmiş olabilir. Eğitim başarısı çok iyi ve test başarısı eğitim başarısına göre daha düşükse, eğitimde modele gösterilen veri ezberlenmiş demektir.

Az öğrenme ise ezberleme durumunun tam tersidir. Geliştirilmiş bir model test verisini daha iyi tahminliyorsa, eğitim sırasında modele gösterilen verilerdeki özellikleri ve yapıyı yeterince öğrenemediğini gösterir. Daha fazla veri ve daha karmaşık bir model ile bu problem giderilebilir.

Makine öğrenmesinde yanlılık (bias) ile varyans arasında hep bir denge vardır. Bu dengeye yanlılık – varyans ikilemi denmektedir.

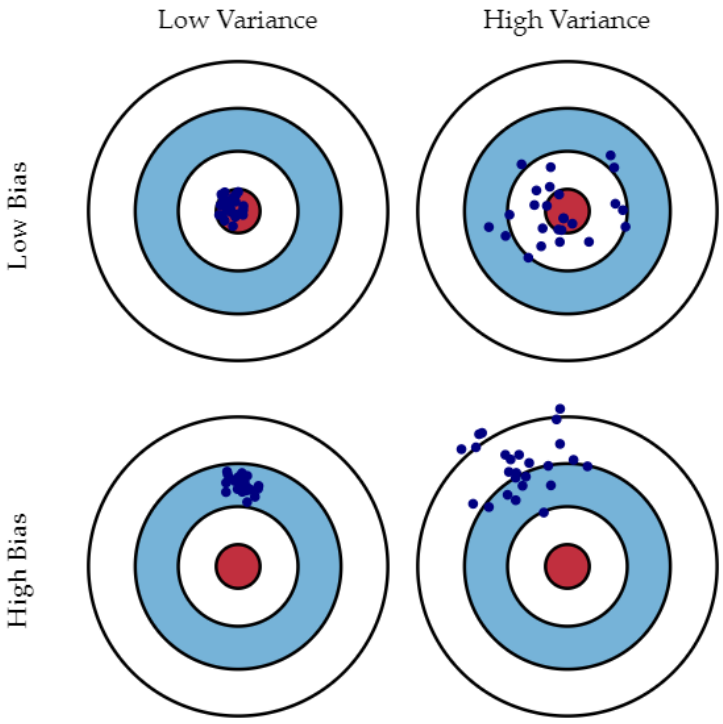


Şekil 2.20 Az öğrenme - aşırı uyum ile yanlılık - varyans ilişkisi [39]

Şekil 2.20'de gösterildiği gibi, ezberleme ile az öğrenme arasında yanlılık ve varyans dengesi vardır. Yüksek yanlılığın olduğu durum, az öğrenmedir. Geliştirilen model yeterince karmaşık olmadığı için, tahmin edilen değerlerin önyargılı bir şekilde yapıldığı anlamı çıkartılabilir.

Yüksek varyans ise aşırı uyum yani ezberleme durumudur. Makine öğrenmesi modeli geliştiren bir araştırmacının, en çok dikkat etmesi gereken konuların başında gelmektedir. Model geliştirme sürecinde, modele gösterilen veriyi yani eğitim setini çok iyi öğrendiğini gösterir. Model, test verisi ise karşılaştığında genelleme yapamayacağı için yanlış tahmin edebileceği durumdur.

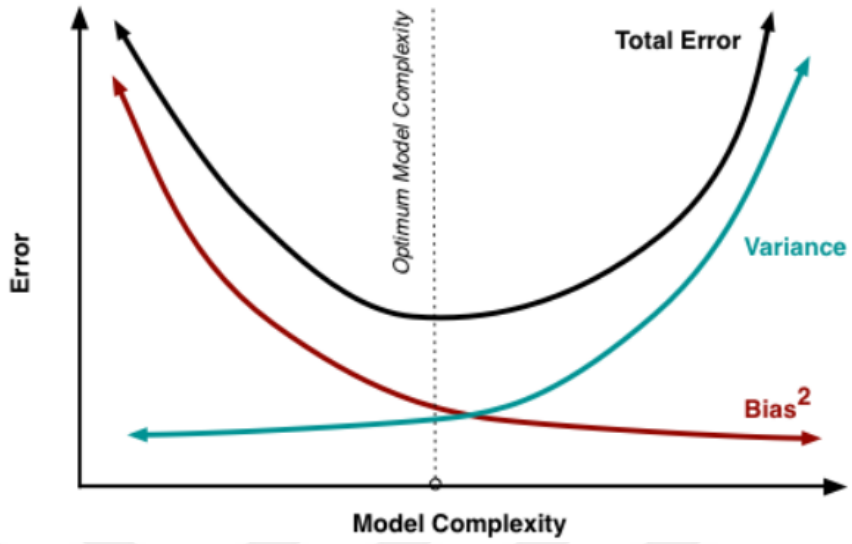
Sonuç olarak, yanlılık (bias) ile varyans arasında bir denge vardır. Makine öğrenmesinde kurulan modellerin karmaşıklığı ve verinin tekdüze ya da yetersiz olması bu dengeyi değiştirmektedir. Geliştirilen model basit, yetersiz veri ve tam öğrenmeden, yapıyı ve ilişkileri, sonlandırılırsa yüksek yanlılık ve düşük varyans ile az öğrenme durumu gerçekleşebilir.



Şekil 2.21 Yanlılık – varyans ilişkisi [40]

Eğitim veri setini çok iyi öğrenip, test verisinde yeterince performans sergileyemiyorsa model, burada aşırı uyum söz konusudur ve modelin

karmaşıklığını azaltıp, daha çok veri ekleyip ve modelin parametreleri değiştirilerek, ezberlemeden ziyade genelleme yapabilme yeteneği artırılabilir.



Şekil 2.22 Toplam hata – varyans ve yanlılık ilişkisi [40]

Şekil 2.22'de gösterildiği gibi araştırmacı, makine öğrenmesi modeli geliştirirken yanlılık, varyans ve toplam hatayı göz önünde bulundurmalıdır. Yanlılık ve varyansı anlayabilmek için modelin tahmin davranışını anlamak kritik öneme sahiptir. Fakat burada araştırmacının amacı, toplam hatayı en aza indirmektir. En uygun noktayı bulabilmek için varyans ve sapmanın artış ve azalışı incelenerek, modelin karmaşıklığı ya da basitliği belirlenmelidir.

2.7.4 Görüntü İşleme Modellerinde Kullanılan Hiper Parametreler

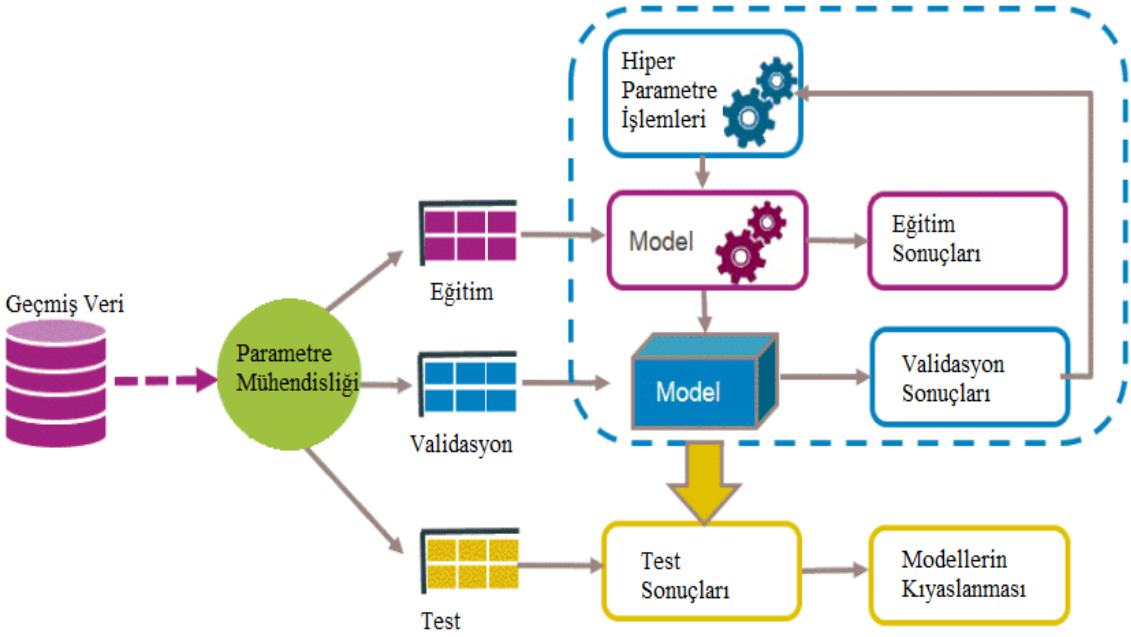
Makine öğrenmesinde araştırmacının, model performansını artırabilmesi ve maliyetleri azaltabilması için alması gereken bazı kararlar / tercihler vardır. Hiper parametre olarak adlandırılan bu tercihler, modelin çalışmasını ve performansını doğrudan etkiler. Bu tercihler, araştırmacıya büyük kolaylık sağlamasıyla birlikte modelini geliştirme imkanı da sağlar. Aşırı uyumu ya da az öğrenmeyi buradan belirleyebiliceği gibi model ayarlamaları (model tununing) işlemlerini de ayarlayabilir. Hiper parametreler iki türde incelenebilir. Bunlardan ilki model hiper parametreleri, ikincisi ise iyileştirici (optimizer) parametreleridir.

Model hiper parametrelerinden bazıları, aktivasyon fonksiyonları, iletim sönümleme oranı ve katman sayılarıdır. İyileştirici olan parametreler, model

mimarisinden ziyade modelin optimizasyonuna odaklanır. Batch boyutu, epoch, momentum ve öğrenme oranı gibi parametrelerdir.

Bu hiper parametrelerin değerleri önceden belli değildir. Algoritmaya, veriye ve probleme göre değişmektedir. Derin öğrenme algoritmalarında, kullanılacak olan gizli katman sayısı, her katmanda kaç nöron olacağı ya da filtrelerin boyutu önceden belli değildir. Araştırmacı deneme yanılma yoluyla ve tecrübesiyle bu değerler üzerinde çalışmalar gerçekleştirir ve en iyi (optimum) değerleri bulmaya çalışır. Bu sebeple, araştırmacının hiper parametre işlemleri özgürlük alanıdır.

Makine öğrenmesi süreçlerinde hiper parametre ayarlamaları, model kurulumundan sonra yapılır. Araştırmacı, birçok modeli karşılaştırıp en uygununu belirledikten sonra model ince ayarlamalarını yapar.



Şekil 2.23 Makine öğrenmesi model geliştirme süreci

Validasyon verisi üzerinden hiper parametrelerini deneyen araştırmacı, test verisi ile modelin başarını test ederek, nihai modelini oluşturabilir. Hiper parametreler, veri setine, probleme ve amaca göre değişebilir.

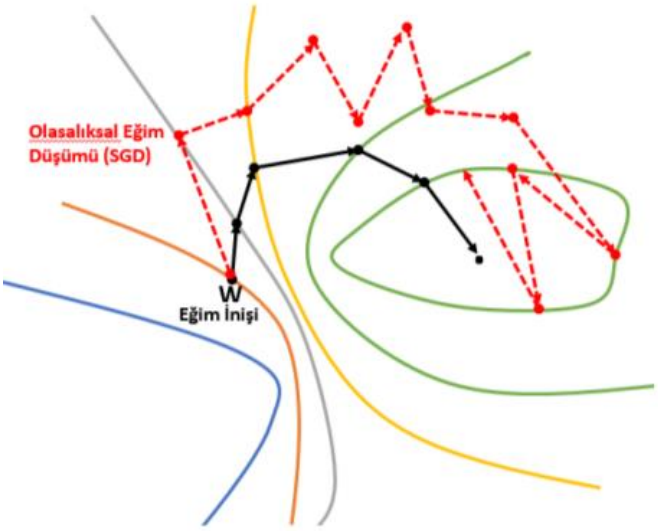
Hiper parametrelerin amacı, modelin daha iyi çalışmasını sağlamaktır. Verinin dağılımı ve araştırmacının istatistik - bilgisayar bilgisi ve ekipman gücüne bağlı olarak belirlenebilir. Kısaca hiper parametre, model performansını doğrudan

etkileyen, modelin optimum değerlerine göre farklılık gösteren, modelin çalışma süresini büyük ölçüde etkileyen, problemdeki amaca göre hassasiyetler seçilebilen ve parametre değerlerinin belirlenmesi için araştırmacıya özgürlük tanıyan modelin alabileceği değerlerdir. Hiper parametreler, veri bilimi süreçlerinde ve makine öğrenmesi modellerinde kritik rol oynamaktadır. Derin öğrenmede en sık kullanılan hiper parametreler; verinin boyutu, mini-batch boyutu, öğrenme hızı, momentum katsayısı, optimizasyon algoritmasının seçimi, tekrar (epoch) sayısı, aktivasyon fonksiyonu, iletim sönümleme değeri, ortaklama yöntemi, filtre katmanının boyutu, gizli katman ve katmanlardaki nöron sayısıdır.

Derin öğrenme modellerinde, verinin sahip olduğu yapıyı anlamak ve anlamlandırmak öğrenme işlemiyle olur. Veri setinin tek parça halinde öğrenmek, model için zaman ve bellek açısından maliyet oluşturur. Her iterasyonda geriye yayılım işlemleri gerçekleştirilerek geriye dönük gradyan (gradient descent) hesapları yapılmaktadır. Bu hesaplar ve iterasyonlar doğrultusunda nöronlardaki bilgiler yani ağırlık değeri değişip, güncellenmektedir.

Mini Batch işlemleri yapılarak, veri seti parçalara bölünür ve hesaplama maliyeti azaltılır. Veri seti parçalara ayrılarak öğrenme işlemi bu parçalar üzerinden yapılır. Veri sayısının parçalara bölünmesi ve bu parçalardan öğrenme işlemlerinin gerçekleştirilmesine mini batch denmektedir. Derin öğrenmede batch sayısı, aynı anda kaç veriyi inceleyip, öğrenmeye çalışacağı anlamı taşımaktadır.

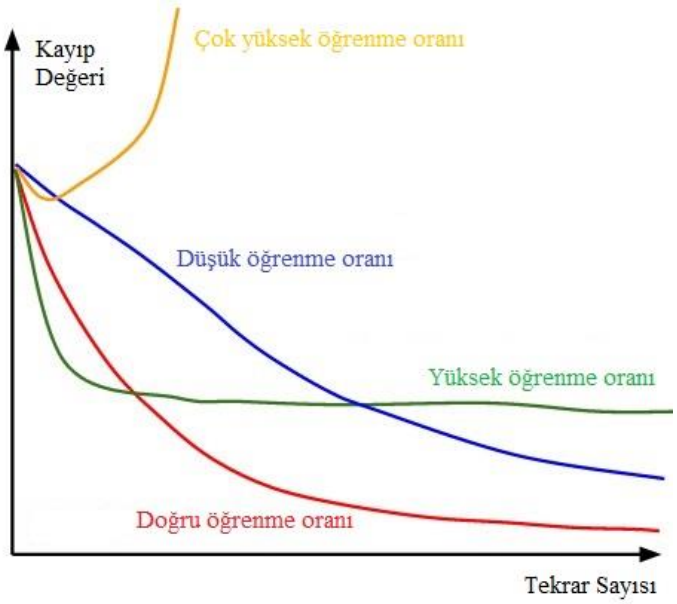
Batch sayısı kullanılarak yapılan eğitimlerde, modelin kayıp değerinin arttığı fakat eğitim süresinin kısaldığı literatürde belirtilmiştir. Probleme ve veri setine bağlı olarak 16, 32, 64, 128 ve 256 batch değerlerinin kullanımı yaygındır. Eğer batch sayısı 1 olarak algoritmaya belirtilmişse, Olasılıksal Eğim Düşümü (SGD) ile aynı iş yapılmış olur. Bu sebeple, en uygun batch değeri için ne küçük ne de büyük bir batch değeri, araştırmacı tarafından belirlenmelidir. Eğer küçük bir sayı seçilirse, gürültü çok fazla olacaktır. Büyük bir sayı seçilirse, modelin performansı düşecek ve yapıyı öğrenmekte zorlanacaktır.



Şekil 2.24 Olasılıksal eğim düşümü [32]

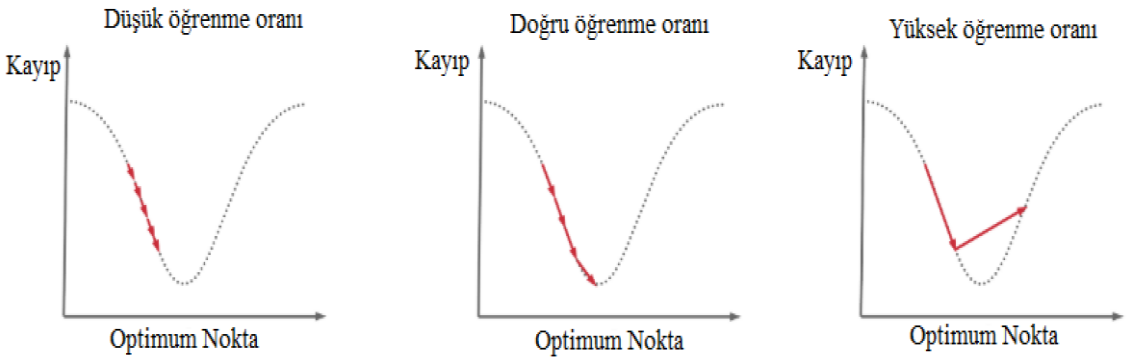
Derin öğrenmede taşınan bilgiler, geri yayılım yoluyla hesaplanmaktadır. Güncellenen değerler geriye yönelik türevi alınarak fark hesaplanır ve bu farklar öğrenme oranı (learning rate) değeri ile çarpılır. Öğrenme hızı, makine öğrenmesi modellerinde en önemli hiper parametrelerin başında gelmektedir. Eğer öğrenme hızı büyük bir sayı seçilirse, model optimum noktaları öğrenemez ve yapıyı genelleştiremez. Küçük bir değer seçilirse, model çok küçük miktarda ilerleme yapacağı için eğitim süresi çok maliyetli olacaktır. Öğrenme hızının yüksek olması, salınımlara neden olmaktadır. Küçük olması ise modelin global optimum noktasını bulmaktan ziyade local optimum değerlerine takılmasına neden olur.

Öğrenme oranının 0.1 ile 0.000001 arasında seçilmesi, derin öğrenme uygulamalarında popülerdir. Öğrenme oranı, az öğrenme ile ezberleme durumlarını doğrudan etkileyen bir hiper parametredir. Bu sebeple derin öğrenme algoritmalarında ilk oluşturulan modelde öğrenme oranı yüksek belirtilir. İstenilen sonuçlara ve hata değerlerine yaklaştıkça, öğrenme oranı küçük oranlarla düşürülmektedir. Adaptif yöntemlerle öğrenme oranı, eğitim esnasında algoritmalar tarafından da değiştirilebilir.



Şekil 2.25 Öğrenme oranı ve tekrar sayısı

Derin öğrenmede, SGD süreçlerde çok fazla sayıda salınım oluşturulmaktadır. Buna bağlı olarak gürültü değerleri üretilmektedir. Bu gürültü değerleri momentum katsayılarıyla normalize edilerek yani salınımlar azaltılarak, hesaplamanın doğru yapılmasını sağlamaktadır. Diğer bir ifade ile güncellenen değer, eskisi ile değiştirilmez, katsayı oranı da işe katılarak hesaplanır. Derin öğrenme uygulama alanlarında, momentum katsayısının 0.8 ile 0.99 arasında seçilmesi yaygın bir kullanımdır.



Şekil 2.26 Öğrenme oranı ve lokal minimum

Yüksek momentum ve öğrenme oranı kullanılması durumunda, algoritma global minimum noktasını kaçırabilir. Bu sebeple araştırmacı, deneme yanılma yoluyla en

uygun değerleri belirlemeli ya da genetik algoritmalar kullanarak bu oran algoritmalarca değiştirilmelidir.

Yüksek öğrenme oranları, önceden öğrenilmiş bilgileri kaybetme riskini artırır. Önceden eğitilmiş modellerin çok iyi eğitildiği varsayımı yapılırsa, öğrenme oranının düşük tutulması ESA ağırlıklarının kısa sürede çok fazla bozulmasına engel olacaktır. Aksi durumda, çok kısa sürede aşırı bir bozulma gerçekleşeceğinden önceki bilgilerin kaybedilmesi riski de artmış olacaktır [36].

Derin öğrenmede model geliştirilirken veri seti, mini batch'lere ayrılarak model eğitilir. Parçalar eğitilirken epoch olarak adlandırılan tekrarlamalarla ileriye ya da geriye doğru ağırlıklar güncellenir. Epoch sayısı kısaca, tekrar / iterasyon / tur sayısıdır. Epoch sayısı az seçilir ise az öğrenme gerçekleşebilir. Genelde epoch sayısı artırılarak model aşırı uyuma yakınlaştırılır. Bu sebeple epoch ile beraber *earlystopping* değeri kullanılır. Erken durdurma ile aşırı uyum engellenebilir. Bu değer, belirtilen değerden daha fazla ilerlemiyorsa tekrarı durdur gibi bir anlam ifade eder.

Derin öğrenme algoritmaları temelinde optimizasyona dayanmaktadır. Tüm makine öğrenmelerindeki ortak hedef, tahmin hatasını düşürmektir. Derin öğrenmede, derin katmanlarda öğrenme işlemleri gerçekleştirilirken, global minimum nokta aranır ve kayıp fonksiyonunun değeri minimize edilmeye çalışılır. Derin öğrenmede doğrusal olmayan problemlerin çözümünde çeşitli optimizasyon algoritmaları kullanılmaktadır. En yaygın kullanılan yöntemler SGD, Adadelta, Adam ve adamax gibi algoritmalardır. Problem, veri seti ve hiper parametrelere göre bu optimizasyon algoritmaları başarılı değişkenlik gösterebilir. Adaptif optimizasyonlar, daha önceden de belirtildiği gibi, öğrenme hızını kendisi belirlemektedir ve süreç içinde değişkenlik göstermektedir. Derin öğrenmede en yaygın kullanılan optimizasyon türlerinin başında adam optimizer gelmektedir.

2.7.5 Aktivasyon Fonksiyonları

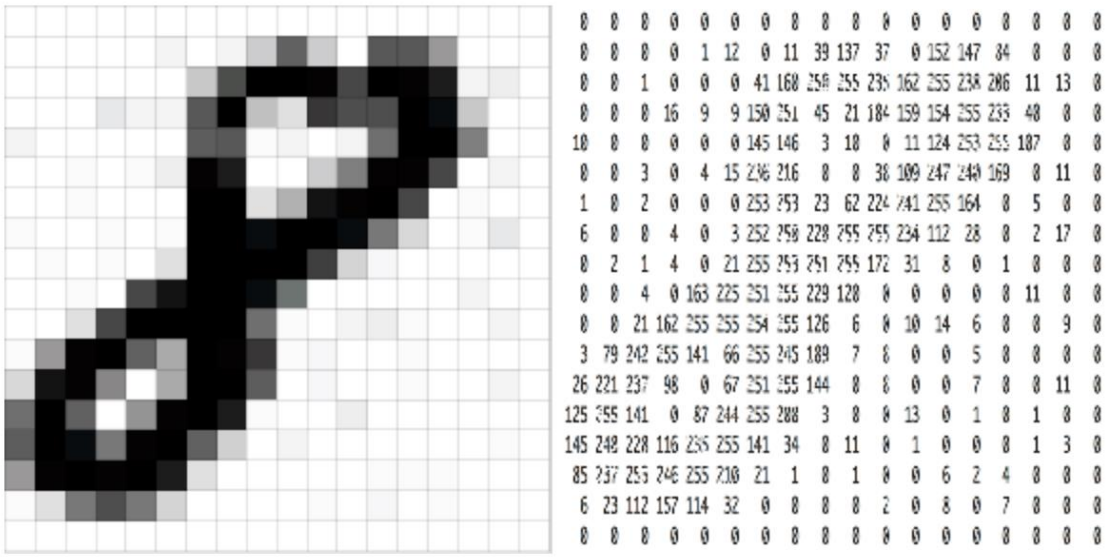
Makine öğrenmesinde aktivasyon fonksiyonları kritik bir rol oynar. Öğrenmenin gerçekleşebilmesi için gereken bu fonksiyonda amaç, doğrusal olmayan verilerin özelliklerini tanıyabilmek için kullanılır ve model performansını doğrudan etkiler.

Aktivasyon fonksiyonu bulunmayan bir fonksiyonun çıkışı, basit bir doğrusal fonksiyondur. Gerçek dünyada ise doğrusal olmayan verileri modelleyebilmek ve modelin öğrenebilmesi için doğrusal olmayan aktivasyon fonksiyonları kullanılır. Bu sebeple, çıkış sinyalinin sadece doğrusal fonksiyon olmaması için aktivasyon fonksiyonları ile doğrusal olmayan fonksiyonlar ve tahminler oluşturulur. Aktivasyon fonksiyonu kullanılmayan sinir ağı modelinde, sınırlı bir öğrenme gücüne sahiptir ve nöronun aktif olması için aktivasyon fonksiyonları kullanılır.

Aktivasyon fonksiyonları, girdi bilgilerini işleyerek bu girdiye karşılık gelen çıktıyı belirler ve bu süreçte eşik değerlerini kullanır. Derin öğrenmede kullanılan aktivasyon fonksiyonlarında, doğrusal olmayan yapıları anlamak için doğrusal olmayan bir fonksiyon seçilir. Doğrusal olmayan yapıları anlayabilmek için aktivasyon fonksiyonları, makine öğrenmesinde önemli rol oynar.

Aktivasyon fonksiyonlarının çalışma mantığı detaylı incelendiğinde, türev işlemlerinin yapıldığı görülmektedir. Sinir düğümlerinden çıktı alabilmek için aktivasyon fonksiyonuna ihtiyaç duyulmaktadır. Makine öğrenmesinde en sık kullanılan aktivasyon fonksiyonlarının başında doğrusal, adım / basamak, sigmoid, hiperbolik tanjant (tanh), doğrultulmuş doğrusal birim (relu), sızıntı doğrultulmuş doğrusal birim (leaky relu) ve softmax fonksiyonları gelmektedir.

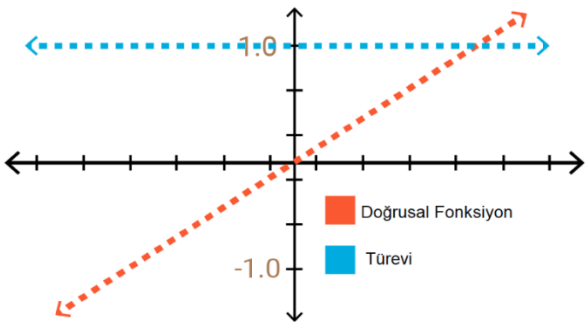
Bilgisayarlı görü projelerinde, görsel veriyi / çok boyutlu matrisi bilgisayara işleyip, derin ağlar kurabilmek için daha önceden de belirtildiği gibi görseli matris haline çevirmek gerekmektedir. Sayısal değerlere dönüştürülen görüntü verisi, çeşitli katmanlarca işlenerek faydalı bilgiler üretilir. Oluşturulan özellik haritaları ve öznitelikler ile algoritmalarca belirlenen yapılar, aktivasyon fonksiyonu ile son adımda tahmin değerlerine dönüştürülür. Aktivasyon fonksiyonu, çıktıdan gelen değerlerin aralıklarının belirlenmesinde kullanılır ve girdi çıktı değerlerinin eğrisel eşleşmesini yapar.



Şekil 2.27 Görüntünün dijitalleştirilmiş bilgisayar görüsündeki karşılığı

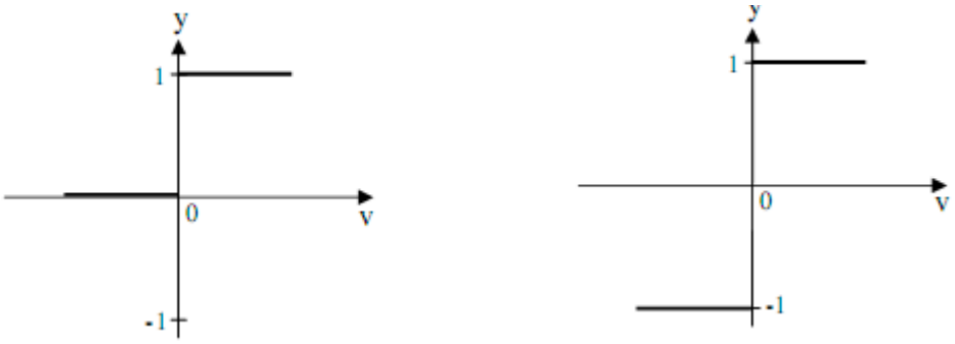
Doğrusal aktivasyon fonksiyonunda, gelen girdiler dönüştürülmeden çıktı katmanına gönderilir. Doğrusal aktivasyon fonksiyonu genellikle regresyon analizlerinde kullanılmaktadır. Doğrusal olmayan verilerde yeterli öğrenme işlemi gerçekleştirilemez ve türevi sabit bir sayıdır.

Geriye yayılım algoritmasında öğrenme, türev değerleri üzerinden gerçekleştirilir. Doğrusal aktivasyon fonksiyonlarında, türev değerleri sabit olacağı için karmaşık veriler üzerine geliştirilen modellerin öğrenmesi yetersiz kalacaktır.



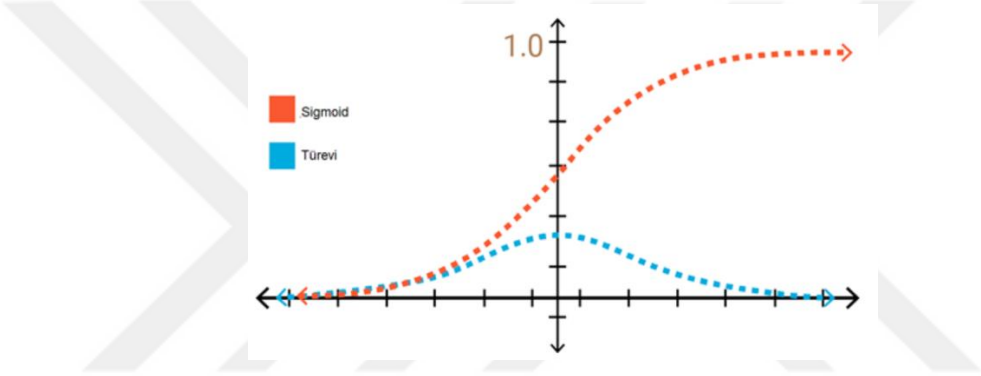
Şekil 2.28 Doğrusal aktivasyon fonksiyonu ve türevi [41]

Adım / basamak aktivasyon fonksiyonları, tek ya da çift yönlü olabilir. Kısaca, ikili sınıflandırma için kullanılan aktivasyon fonksiyonudur. Gelen girdinin, bir eşik değerinin üstünde ya da altında olmasına göre 1 ya da 0 değerini alır.



Şekil 2.29 Tek kutuplu ve çift kutuplu adım aktivasyon fonksiyonu

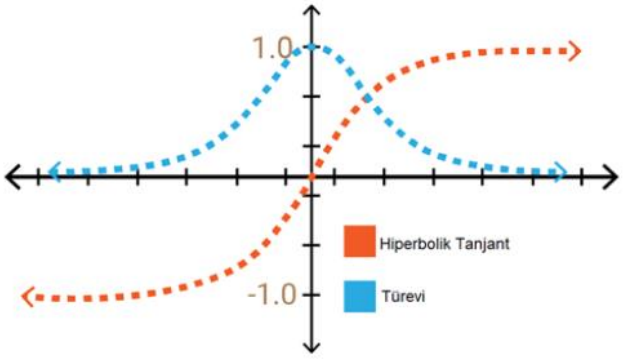
Sigmoid / lojistik aktivasyon fonksiyonu, çıktı değerlerini 0 ile 1 arasına sıkıştırır. Sigmoid aktivasyon fonksiyonları, türevlenebilir olduğu için öğrenme işlemi gerçekleştirebilir. Bu sebeple, ikili sınıflandırmada yaygın olarak kullanılmaktadır.



Şekil 2.30 Sigmoid aktivasyon fonksiyonu [41]

Sigmoid aktivasyon fonksiyonlarında, y değeri x değerlerine karşı belli bir seviyeden sonra tepkisizleşir. Az tepki vererek sıfıra yakınsamasına, gradyanların ölmesi denmektedir (vanishing gradient). Gradyanların öldüğü durumda, öğrenme olayı en az seviyede gerçekleşir ya da tamamen biter. Bu sebeple, algoritma lokal minimum seviyelerinde gezineceği için, global minimum noktasını bulamaz ve karmaşık problemlerde istenilen performansı sergileyemez. Ayrıca Sigmoid aktivasyon fonksiyonu, makine öğrenmesinde en sık kullanılan aktivasyon fonksiyonlarının başında gelmektedir.

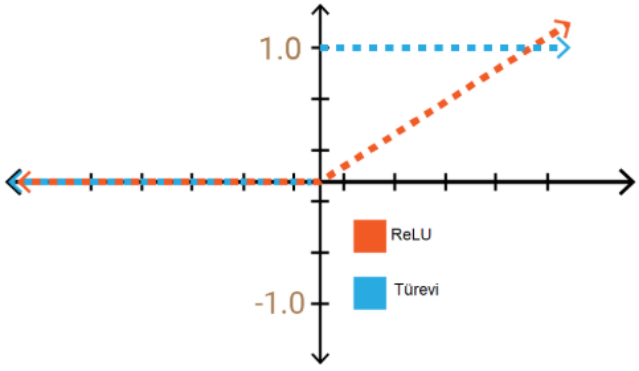
Sigmoid aktivasyon fonksiyonunun çalışma mantığına benzeyen Hiperbolik Tanjant aktivasyon fonksiyonu, Sigmoid aktivasyon fonksiyonundan farklı olarak aralığı -1 ile +1 arasına sıkıştırır. Derin öğrenmede en sık kullanılan aktivasyon fonksiyonlarından biridir.



Şekil 2.31 Hiperbolik tanjant aktivasyon fonksiyonu [41]

Şekil 2.31'de görüldüğü üzere Hiperbolik tanjant aktivasyon fonksiyonu, Sigmoid aktivasyon fonksiyonuna göre daha fazla değer alabilmektedir. Daha geniş bir aralığa sahiptir. Girdi olarak negatif bir değer gelmişse negatif, pozitif bir değer gelmiş ise pozitif değer döndürmesi yapmaktadır. Doğrusal olmayan verilerde, nöronların aktive edilmesi için sıklıkla kullanılan Hiperbolik Tanjant aktivasyon fonksiyonu, -1 ile +1 arasında değişim göstererek derin öğrenme modellerine hızlı bir eğitim imkanı sağlar.

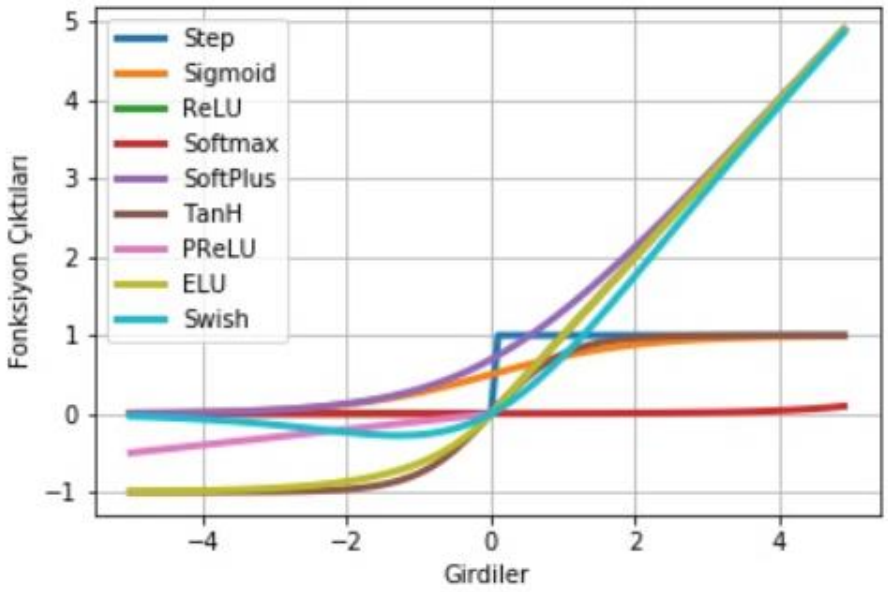
Doğrusal, Sigmoid ve Hiperbolik tanjant aktivasyon fonksiyonları kadar sık kullanılan diğer bir aktivasyon fonksiyonu ise Doğrultulmuş Doğrusal Birim fonksiyonudur. Evrişimsel modellerde ya da derin öğrenmede en sık kullanılan algoritmaların başında gelmektedir.



Şekil 2.32 Doğrusal birim aktivasyon fonksiyonu [41]

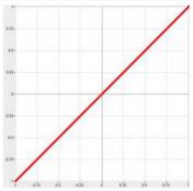
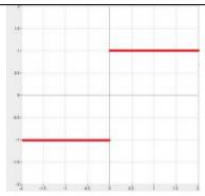
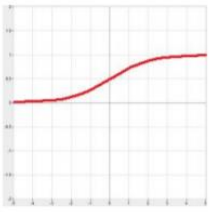
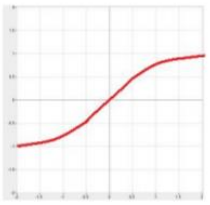
İyi bir tahmin edici olan Doğrusal birim aktivasyon fonksiyonu, 0'dan pozitif sonsuza kadar değer alabilmektedir. Negatif rakamlar için sıfır değerini veren bu fonksiyonda, derin mimarilerde daha hızlı çalışmaktadır. Ayrıca işlem yükü

bakımından Sigmoid ve Hiperbolik tanjant aktivasyon fonksiyonlarına göre daha hızlıdır. Bu sebeple kullanımı yaygındır. Diğer yandan negatif değerlere karşı türevinin olmamasından, öğrenme işlemi negatif değerler için gerçekleşmez. Bu sebeple negatif değerlerde de öğrenmenin gerçekleşebilmesi için sızıntı doğrusal birim aktivasyon fonksiyonu geliştirilmiştir.



Şekil 2.33 Aktivasyon fonksiyonlarının düzlem üzerinde gösterimi [42]

Şekil 2.33'te görüldüğü üzere, derin öğrenme algoritmalarında sıklıkla kullanılan aktivasyon fonksiyonları düzlem üzerinde gösterilmiştir. Her bir aktivasyon fonksiyonun ayrı ayrı gösterimi ve matematiksel fonksiyonu Şekil 2.34'te gösterilmiştir.

Doğrusal (Lineer) Aktivasyon Fonksiyonu		F(NET)=A* NET (A sabit bir sayı)	Doğrusal problemler çözmek amacıyla aktivasyon fonksiyonu doğrusal bir fonksiyon olarak seçilebilir. Toplama fonksiyonundan çıkan sonuç, belli bir katsayı ile çarpılarak hücrenin çıktısı olarak hesaplanır.
Adım (Step) Aktivasyon Fonksiyonu		$F(Net)=\begin{cases}1 & \text{if Net>Eşik Değer}\\ 0 & \text{if Net<=Eşik Değer}\end{cases}$	Gelen Net girdinin belirlenen bir eşik değerin altında veya üstünde olmasına göre hücrenin çıktısı 1 veya 0 değerini alır.
Sigmoid Aktivasyon Fonksiyonu		$F(Net)=\frac{1}{1+e^{-Net}}$	Sigmoid aktivasyon fonksiyonu sürekli ve türevi alınabilir bir fonksiyondur. Doğrusal olmayışı dolayısıyla yapay sinir ağı uygulamalarında en sık kullanılan fonksiyondur. Bu fonksiyon girdi değerlerinin her biri için 0 ile 1 arasında bir değer üretir.
Tanjant Hiperbolik Aktivasyon Fonksiyonu		$F(Net)=\frac{e^{Net}+e^{-Net}}{e^{Net}-e^{-Net}}$	Tanjant hiperbolik fonksiyonu, sigmoid fonksiyonuna benzer bir fonksiyondur. Sigmoid fonksiyonunda çıkış değerleri 0 ile 1 arasında değişirken hiperbolik tanjant fonksiyonunun çıkış değerleri -1 ile 1 arasında değişmektedir.

Şekil 2.34 Aktivasyon fonksiyonları [43]

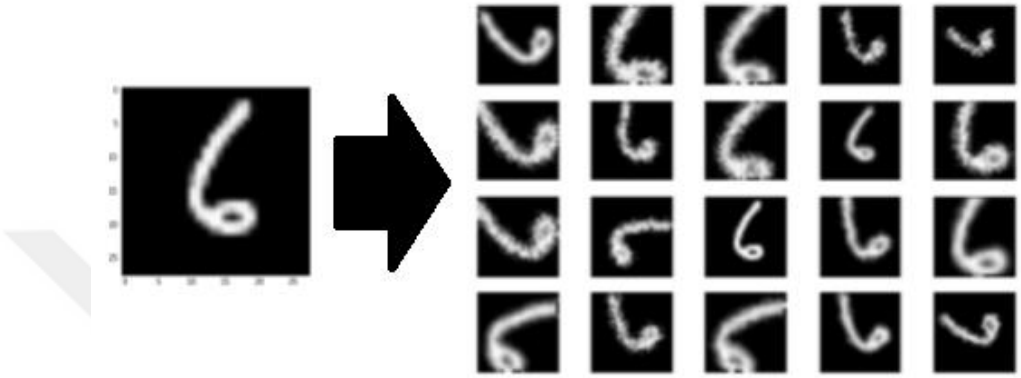
2.7.6 Veri Artırma

Derin öğrenme modellerinde sıklıkla kullanılan ve model başarısını artıran diğer bir yöntem ise yapay veri üretmedir (data augmentation). Yapay veri üretme işleminde, eğitim veri setindeki her farklı görsel algoritmaya girmeden önce sentetik veri üretilir. Bu sentetik veri üretme işlemlerinde, her bir görsel verinin üzerinde oynanır ve yeni bir görselmiş gibi modele girer. Veri artırma yöntemleri, az verinin olduğu durumlarda ya da model başarısının artırılmasının istendiği durumlarda, araştırmacının ilk başvurduğu yöntemlerden biridir.

Derin öğrenme algoritmaları, ne kadar çok veri olursa model performansının yükseldiği belirtilmiştir. Her bir görüntü verisinin içerdiği ilişkiyi ve yapıyı anlayabilmek için derin öğrenme modelleri çok fazla sayıda öznitelik çıkartır. Veri artırma ile beraber bir görsel, araştırmacının belirlediği kısıtlar altında modele farklı bir görüntü verisiymiş gibi girerek, modelin yapıyı anlamasına yardımcı olur. Az verinin olduğu durumlarda araştırmacılar, yapay veri üreterek ellerindeki

mevcut veri sayısını artırmaya çalışır. Sentetik veri üretme işlemi ile model performansı büyük ölçüde yükselir.

Veri artırımı yapılırken araştırmacılar en sık döndürme (rotation) işlemini uygular. Bunun sebeplerinden biri ise test verisinde her zaman görüntü verisinin eğitim verisinde bulunduğu açı ile gelemeyeceğidir.



Şekil 2.35 Döndürme işlemi yaparak sentetik veri üretme

Şekil 2.35'te görüldüğü üzere, 6 rakamının döndürme işlemi gerçekleştirilerek 1 görsel veri, 20 farklı görüntü verisine dönüştürülmüştür. Bu işlem ile Evrişimsel Sinir Ağı modelleri, 6 rakamının karakteristik özelliklerini ve özniteliklerini anlama konusunda daha başarılı olacaktır.

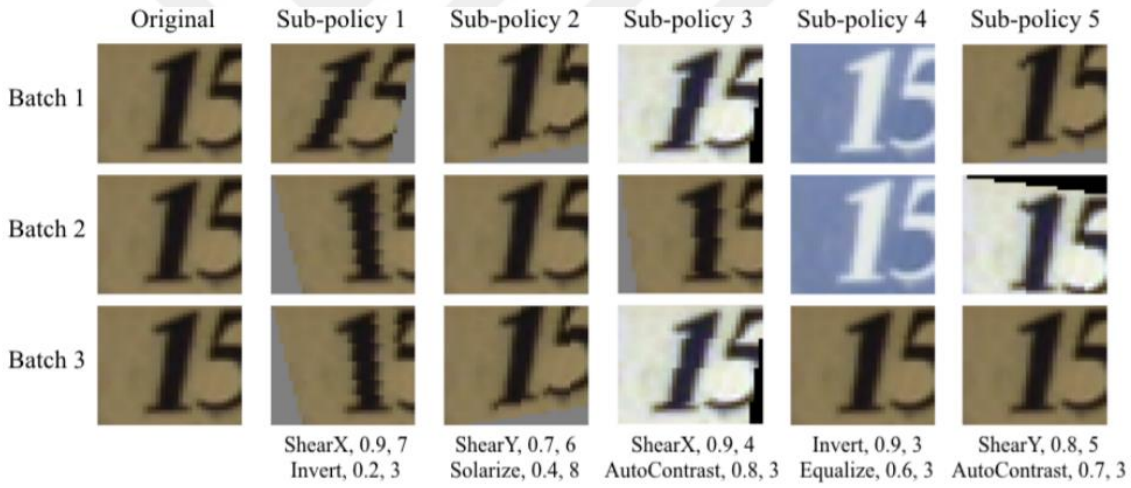
Veri artırmanın avantajı ise kısıtlı veri kümesinin olduğu durumlarda, aşırı uyum ihtimalini düşürmektedir. Tekdüze ve az çeşitli verinin olması aşırı uyumun gerçekleşme ihtimalini yükseltir. Sentetik veri üreterek, veri çeşitlendirilir.

Sentetik veri üretirken araştırmacılar genelde eğitim verisindeki görseli döndürerek, ölçekleyerek, gürültü ekleyerek, belli bir bölgeyi kırparak, görüntüyü biraz daha karartarak / parlatarak ya da merkeze odaklayarak çoğaltmaya çalışır. Bu gibi teknikler gerçek hayat problemlerine benzediği için modelin test ve gerçek hayattaki performansını artırır.

Sentetik veri üretmenin diğer bir avantajı ise derin öğrenme modelini, gerçek hayat problemlerine karşı hazırlamaktır. Nesne tanıma, görüntü işleme ya da görüntü sınıflandırma problemlerinde genellikle araştırmacı çok fazla sayıda eğitim verisine

ulaşmaya çalışır. Fakat gerçek hayatta karşılaşılan görsel veriler, model eğitiminde kullanılan görsel verilere benzemeyebilir. Bu sebeple araştırmacı modelini eğitirken, çok sayıda durumu ve koşulu düşünerek, yakınlık / uzaklık, aydınlık / karanlık gibi durumlar, eğitim verisi hazırlar ve bu veriler üzerinden derin öğrenme modelini geliştirir. Yapay veri üretilirken araştırmacının belirlediği kısıtlar ve özellikler doğrultusunda her görsel değiştirilir ve yeni bir veriymiş gibi modele girmiş olur.

Derin öğrenmede çeşitli algoritmalarda daha başarılı sentetik veri üretimi yapılmaktadır. AutoAugment: Larning Augmentation Strategies From Data (CVPR) [44], otomatik veri artırımı için geliştirilen bu yöntemde, arama algoritmaları kullanılmaktadır. Eğitim verisindeki veriler modele girmeden önce arama algoritmaları verinin üzerinde çalışarak veriyi artırır. Arama algoritmalarında ise Pekiştirmeli Öğrenme (RF) kullanmaktadır [45].



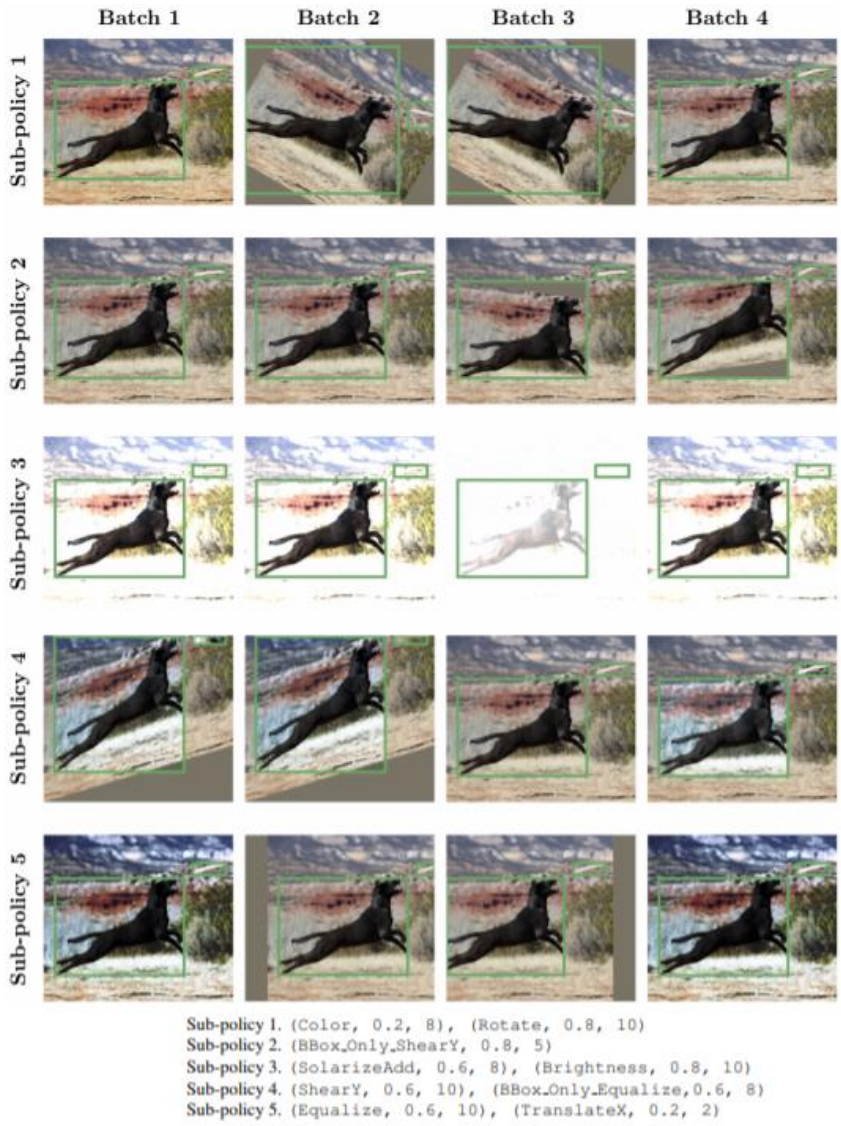
Şekil 2.36 Autoaugment yöntemi ile sentetik veri üretme [44]

Şekil 2.36'da farklı yöntem ve batch ile üretilen veriler gösterilmiştir. Orjinal görsel verinin birinci yöntemde 0.9 ihtimal ile 7 büyüklüğünde kaydırıldığı ve 0.2 ihtimal ile 3 büyüklüğünde döndürüldüğü belirtilmiştir.

Sentetik veri üretilirken diğer bir sık kullanılan veri üretme algoritması ise Fast AutoAugment'tir [46]. CVPR'de büyük oranda GPU kullanılmaktadır. Bu da hız anlamında algoritmayı etkilemektedir. Fast AutoAugment'te bu problem giderebilmek için Population Based Augmentation (PBA) tekniği kullanılmaktadır. Arama algoritmalarından biri olan Population Based Augmentation kullanılmasıyla

algoritma daha hızlı çalışmaktadır ve tek modelde en fazla faydayı veren yöntemleri bulur.

Çok boyutlu veri matrislerinde, görsel üzerinden nesne tespiti derin öğrenmede sıklıkla kullanılmaktadır. Sentetik veri üretilirken, nesne tespiti yapan ve bu nesne üzerinden veri üretimi yapan diğer bir algoritma ise Learning Data Augmentation Strategies for Object Detection'dur [47].



Şekil 2.37 Nesne belirleme yaparak sentetik veri üretme [47]

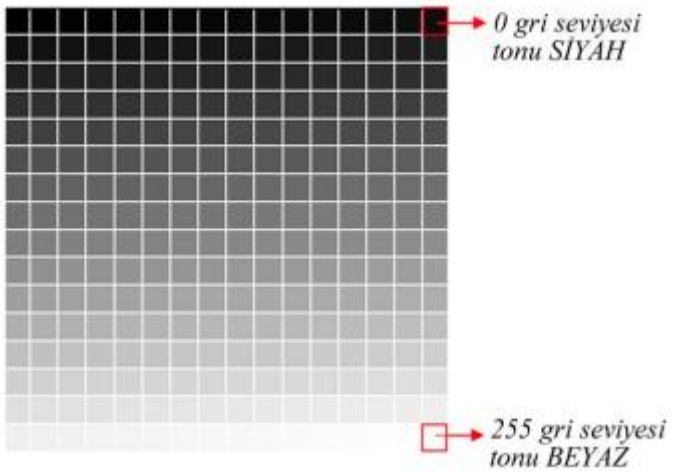
Bu yöntemde ilk önce görsel üzerinden nesne belirleme yapılır ve ardından çeşitli yöntemler kullanılarak yapay veri üretilir.

2.7.7 Derin Öğrenmede Görüntü Ön İşleme

Makine öğrenmesinde model kurulmadan önce, verinin model kurulumuna hazır hale getirilmesi gerekir. Bu işleme veri ön işleme ya da veri manipülasyonu denir. Metin / ses analiz edilecekse ya da görüntü işleme yapılacaksa, bu verileri modele sokmadan önce modelin istediği formatta ayarlanması gerekmektedir.

Derin öğrenme modellerinin kurulabilmesi için görsel verinin, algoritmalar tarafından istenilen şekilde olması gerekmektedir. Piksel boyutu veya renk derinliği gibi değerler, algoritmaların ya da mimarilerin türlerine göre değişkenlik gösterebilir. Modelin kurulabilmesi için bu gibi değerlere dikkat edilmelidir.

Görüntü verisi algoritmaya girdikten sonra daha önceden de belirtildiği gibi bilgisayarlı görü için dijitalleştirilerek 0'dan 255'e kadar değer alır ve böylelikle çok boyutlu matris verisine ulaşılmış olur.



Şekil 2.38 Gri renk tonları ve bilgisayar görüsündeki karşılıkları [48]

Modele girmeden önce görüntülerin iyileştirilmesi, onarılması ve sıkıştırılması amacıyla, görsel veri ön işlemeden geçmektedir. Bunların amacı, hem model başarısını artırmak hem de kullanılan yöntemin koşulunu sağlamaktır. Bazı transfer öğrenme mimarileri, örneğin VGG16 mimarisi 224 x 224 x 3 boyutunda, belli boyutta ve derinlikte görsel veri kabul etmektedir.

Renk	R	G	B	Görünüm
Kırmızı	255	0	0	
Yeşil	0	255	0	
Mavi	0	0	255	
Beyaz	255	255	255	
Siyah	0	0	0	
Açık Gri	200	200	200	
Koyu Gri	100	100	100	
Sarı	255	255	0	
Turkuaz	0	255	255	
Eflatun	255	0	255	

Şekil 2.39 Bazı renklerin rgb değerleri [48]

Modeller öznitelik çıkarımı yaparken renk desenleri üzerinden bir ilişki / yapı aramaktadır. Renk desenleri ile beraber görüntü verileri model tarafından öğrenilirken, görüntü boyutu değiştirilir ve piksel boyutları düşürülerek evrişim katmanına gönderilir. Çeşitli filtreler ve ortaklama işlemleri ile çok boyutlu veri iyice küçültülür ve düzleştirme (flatten) denilen katmanda n x 1 boyutuna dönüştürülür. Bu çalışmada elde edilen tüm görüntü verileri, 224 x 224 x 3 boyutuna indirilmiştir.

3
UYGULAMA

3.1 Veri Seti

Evrişimsel Sinir Ağları, derin öğrenme modellerinin görüntü ve ses verileri üzerinde otomatik olarak öznitelik çıkarmasında kullanılan yaygın bir algoritmadır. Filtre ve havuz katmanlarında görüntü verisi, çok boyutlu matris formatına dönüştürülür ve veriler işlenir. Derin öğrenmede öğrenme işlemi, derin katmanlarda gerçekleştiği için adına derin öğrenme denmektedir. Katmanlar sıralı bir şekilde oluşturulur ve farklı mimari desenler kullanılarak, çeşitli hiper parametreler ile model çıktıları tahmin edilmeye çalışılır.

Görüntü işlemede model performansını artırabilmek için çeşitli yöntemler vardır. Bu çalışmada, çeşitli hiper parametreler ile beraber geliştirilen 2. modelde sentetik veri üretilmiş ve transfer öğrenme yöntemleri uygulanmıştır. Veri artırma işlemlerinde her görselin boyutları değiştirilerek görüntü verisi çeşitli noktalara yakınlaştırılır, döndürülür ya da görsel verinin parlaklıkları değiştirilerek modele yeni bir görüntü verisiymiş gibi girer. Böylelikle veri zenginleştirilmiş olur ve daha az çaba harcanarak, model daha fazla verinin yapısını inceleme fırsatı bulmuş olur.

Bu çalışmada, geliştirilen tüm modellerde her bir görüntü verisi, 224 x 224 x 3 boyutuna indirgenmiştir. Bu indirgeme işleminden sonra evrişim katmanına gönderilerek öznitelikler çıkartılıp, çeşitli filtrelerden geçirilmiştir. Filtreleme katmanlarından sonra 2 x 2 boyutunda en büyük ortaklama - havuzlama (Max Pooling) katmanına gönderilerek görüntü matrisi 224 x 224 boyutundan 112 x 112 boyutuna indirgenmiştir. Ardından filtre ve evrişim katmanlarına gönderilerek yeni öznitelikler çıkartılmıştır. Ardından bir kez daha en büyük ortaklama – havuzlama katmanına girerek çok boyutlu görüntü matrisi 56 x 56 boyutuna indirilmiştir. Bu işlem 2 kez daha uygulanarak düzleştirme (flatten) katmanına 7 x 7 x 512 boyutunda girmiştir. Son olarak, tam bağlantı katmanıyla tüm nöronlar birbirine bağlanıp, aktivasyon fonksiyonu yardımıyla 10 farklı sınıfa ayrıştırılmıştır.

Çalışmada VGG16 mimarisi, transfer öğrenme, sentetik fotoğrafların üretilmesi ve Tensorflow Lite kullanılarak Efficientnet gibi yeni yöntem ve modeller kullanılarak yeni sınıflandırma modelleri geliştirilmiştir. Nöron sayıları, iletim sönümleme oranı, öğrenme hızı gibi farklı hiper parametreler değiştirilerek modellerin başarıları kıyaslanmış ve eniyilemeye çalışılmıştır. Bu kıyaslama, doğru sınıflandırma başarı yüzdesi üzerinden hesaplanmıştır. Çalışmada, Python programlama dili tercih edilmiştir.

Bu çalışmada, uygulama verisi olarak toplamda 10 farklı sınıftan, 2500 adet görsel veri kullanılmış ve bu sınıflar görüntü işleme teknikleri ile sınıflandırılmaya çalışılmıştır. Bu görüntü verileri, her ailenin evinde rahatlıkla bulunabilecek ilaç kutularından seçilmiştir. İlaç kutuları seçilirken, yaygın bulunabilirlik şartı aranmıştır. Elde edilen 10 farklı ilaç kutusundan, ayrı ayrı 250 adet fotoğraf çekilmiştir. Çalışmada, 10 farklı sınıf için toplamda 500 adet test verisi mevcutken, 2000 adet eğitim verisi bulunmaktadır.

Çalışmada, veriler elde edilirken aynı standartlarda / şartlarda bulunmasına özen gösterilmiştir. Seçilen her ilaç kutusunun farklı arka planlarda, farklı açılardan, farklı fotoğraf makinelerinden ve farklı uzaklıklardan çekilerek veriler üretilmiştir. Farklı uzaklıklar belirlenirken yakın, çok yakın ve uzak şeklinde uzaklık anlayışı ölçülebilir hale getirilmiştir.

İlaç kutularının her yüzünden, eşit sayıda, aynı arka planda ve eşit uzaklıklarda, sadece açıları değiştirilerek veriler üretilmiştir. Bunun sebebi ise modele yanlılık ve dengesizlik katmamaktır. Derin öğrenme modellerinde, modelin yapıyı ezberlemeyip genelleştirebilmesi önemlidir. Bu sebeple, aynı arka planda, aynı ortamda, aynı görüntü kalitesiyle ve aynı açıdan fotoğraflar çekilmemiştir.

3.2 Yöntem ve Bulgular

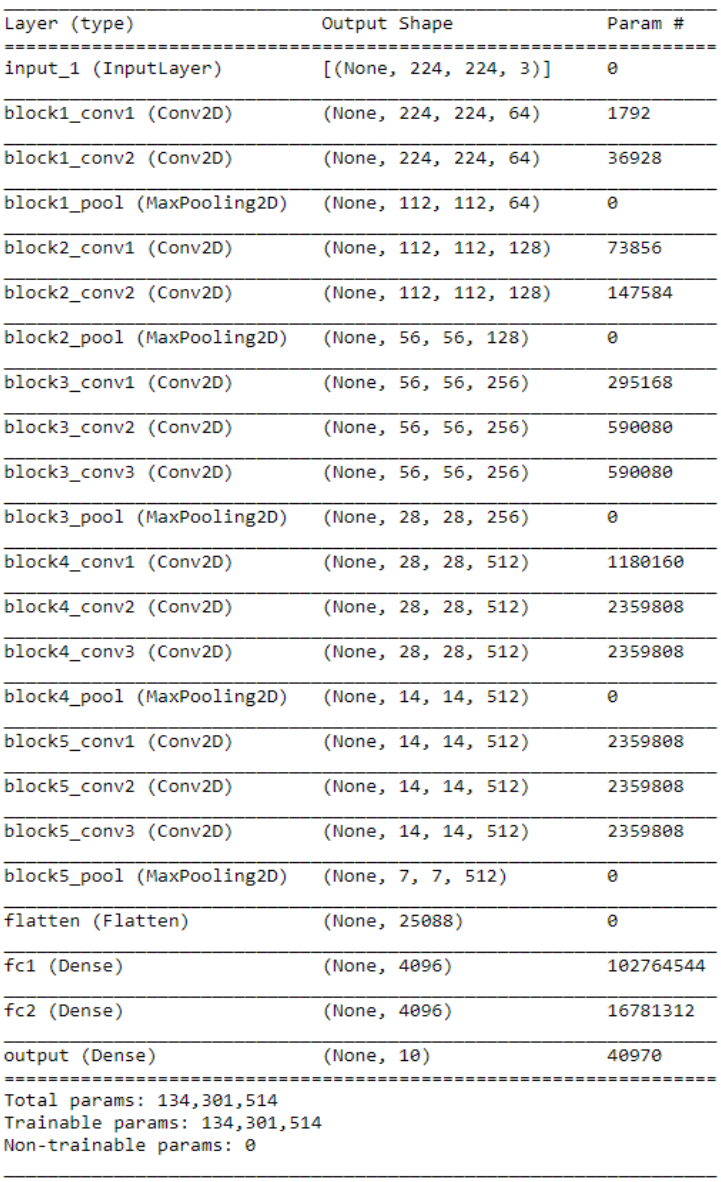
Derin öğrenme algoritmaları, bilinen girdi ve beklenen çıktılara bakarak anlamlı ve faydalı bilgiler üretmeyi öğrenir. Tahmin edilen çıktı ile beklenen çıktı arasındaki fark, minimize edilmeye çalışılarak derin katmanlarda verinin sahip olduğu özellikler öğrenilir. Derin öğrenme ve makine öğrenmesi modelleri, beklenen çıktılara en yakın tahmin değerlerini üretebilmek için çeşitli dönüşümler yapar.

Örneğin, derin öğrenme algoritmaları kırmızı görüntü noktalarını bulabilmek için RGB renk uzayında basit dönüşümler yaparken, görüntü doygunluğunu azaltabilmek için HSV renk uzayında dönüşümler yapmaktadır. Bu sebeple ESA algoritmaları, görüntü verisini işlerken filtreler ve çeşitli yöntemler kullanarak veriden anlamlı ve çeşitli bilgiler türetmeye çalışır.

Literatür araştırmasında görülmüştür ki, veri setinin az olması durumunda, derin öğrenme algoritmalarının etkili bir şekilde kullanılabilmesi için modelin hiper parametreleriyle oynanmalı, transfer öğrenme algoritmaları kullanılmalı ve sentetik veri üretilmelidir. Simonyan ve Zisserman [49] tarafından önerilen VGG16 mimarisinin, çalışmanın konusu olan görüntü sınıflandırma problemleri için uygun olduğu belirlenmiştir. Bu sebeple uygulamada, VGG16 mimarisi tercih edilmiştir. Bu çalışmada, ESA mimarisi baştan oluşturulmamış, bunun yerine VGG16 mimarisi ile transfer öğrenmeden faydalanılmıştır. ImageNet verisinden daha önce eğitilmiş Evrişimsel Sinir Ağları modelinin ağırlıkları hazır olarak alınmış ve çalışma verilerine uygulanarak, modelin sahip olduğu ağırlıklar güncellenmiştir.

Bu çalışmada, kamera yardımıyla elde edilen görüntü verilerinin sınıflandırılması amaçlanmıştır. Bu sınıflandırmayı yapabilmek ve yüksek model performansı sağlayabilmek için çeşitli ESA yöntemleri kullanılmıştır. Model eğitim süresinin kısaltılması ve model performansının artırılabilmesi için de öğrenme aktarımı kullanılmıştır. Çalışma kapsamında, elde edilen veriler öncelikle veri ön işleme adımlarından geçerek yeniden boyutlandırılmış ve dijital olarak sayısallaştırılmıştır. Sayısallaştırılan bu veriler ESA yöntemleri ile öznitelikleri ve özellik haritaları çıkartılarak, modele girmiş ve tahmin değerleri alınmıştır.

Klasik bir VGG16 mimarisinde yaklaşık 134 milyon parametre vardır. Bu uygulama kapsamında, 3 farklı model kurulmuştur. Bunlardan biri 134 milyon parametreli klasik VGG16 mimarisine sahip ESA modeli iken diğer model, 14 milyon parametreye sahiptir. Bu iki model arasında, 14 milyon parametreye sahip modelin daha başarılı olduğu görülmüştür.

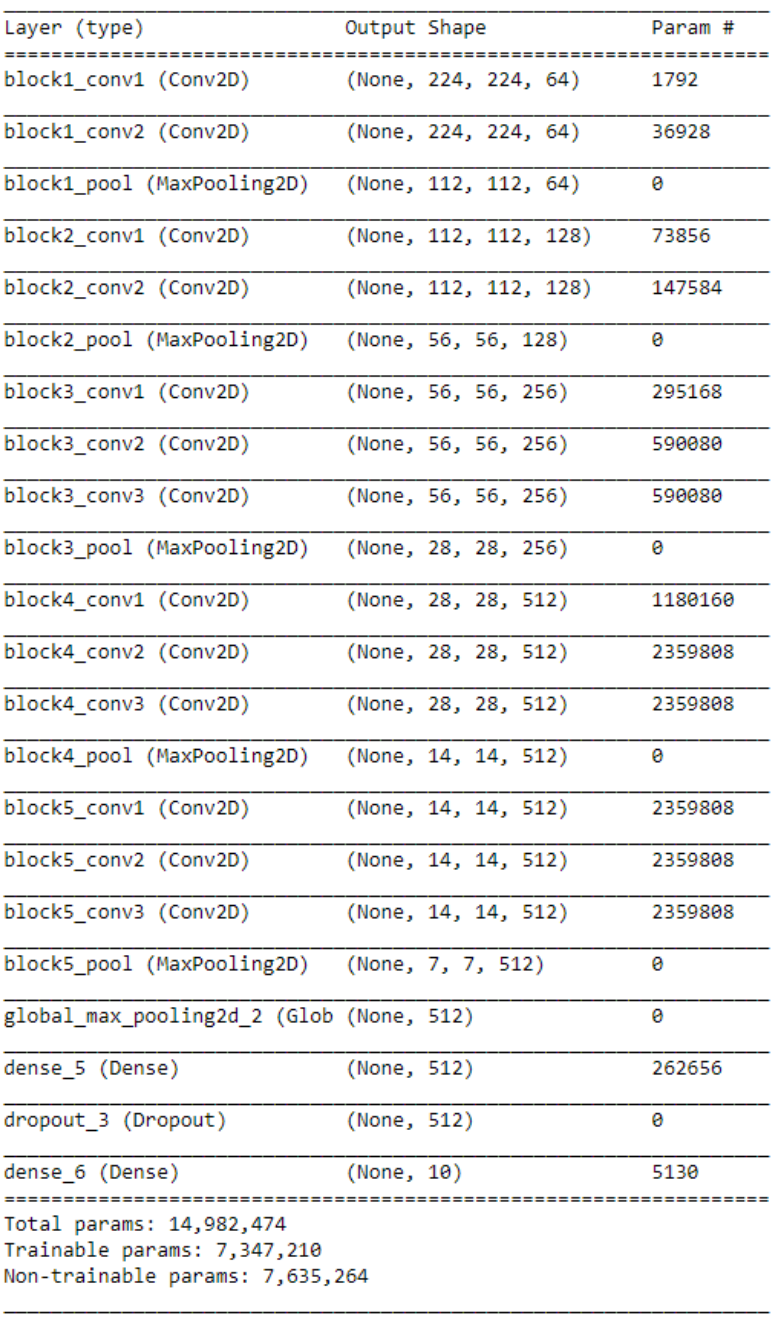


Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
fc1 (Dense)	(None, 4096)	102764544
fc2 (Dense)	(None, 4096)	16781312
output (Dense)	(None, 10)	40970

Total params: 134,301,514
Trainable params: 134,301,514
Non-trainable params: 0

Şekil 3.1 Kurulan birinci modelin mimarisi

Unutulmamalıdır ki, makine öğrenmesindeki asıl amaç, eniyileme ile genellenebilirlik arasında kalan kısımda bir model geliştirmektir. Occam's Razor prensibinde, bir problemi daha basit bir şekilde açıklayan ve daha az varsayımı bulunan bir açıklamanın, daha başarılı olduğu söylenir. Derin öğrenmede deson yıllarda geliştirilen mimarilerde de durum bu şekildedir. Son yıllarda Imagenet verisinde başarılara bakıldığında mimari derinlikten ziyade yeni yaklaşımların olduğu görülmektedir. Karmaşık ve yüksek parametreli modeller, aşırı öğrenmeye daha yatkındır [4]. Geliştirilen ikinci modelde 0.5 oranında iletim sönümleme kullanılmasıyla beraber, mimariye birkaç katman daha eklenmiştir.



Layer (type)	Output Shape	Param #
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
global_max_pooling2d_2 (Glob	(None, 512)	0
dense_5 (Dense)	(None, 512)	262656
dropout_3 (Dropout)	(None, 512)	0
dense_6 (Dense)	(None, 10)	5130

Total params: 14,982,474
Trainable params: 7,347,210
Non-trainable params: 7,635,264

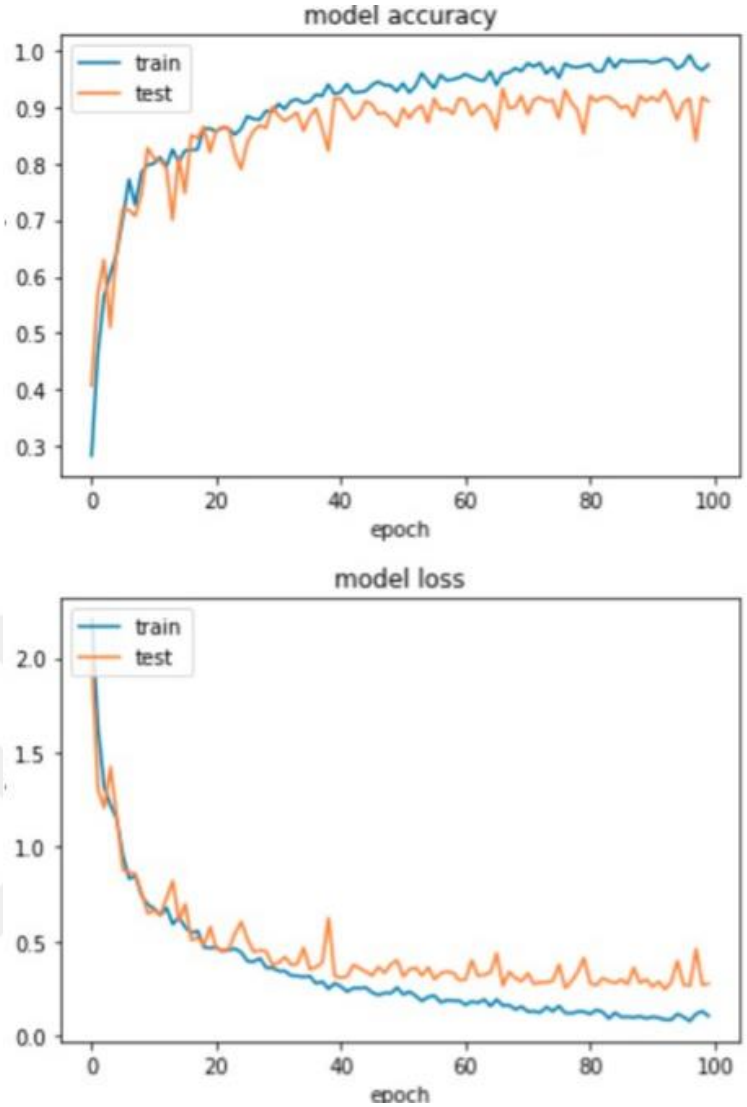
Şekil 3.2 Kurulan ikinci modelin mimarisi

VGG16 mimarisine ek olarak, Tensorflow Lite kullanılarak (Efficientnet model ailesi) üçüncü bir ESA modeli geliştirilmiştir. Geliştirilen bu modelde, toplam 3.413.024 parametre vardır ve en başarılı sonuç da bu modelde gözlemlenmiştir.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
hub_keras_layer_v1v2_1 (HubK (None, 1280)              3413024
_________________________________________________________________
dropout_1 (Dropout)          (None, 1280)              0
_________________________________________________________________
dense_1 (Dense)              (None, 10)                12810
=================================================================
Total params: 3,425,834
Trainable params: 12,810
Non-trainable params: 3,413,024
```

Şekil 3.3 Kurulan üçüncü modelin mimarisi

İlk VGG16 mimarisiyle geliştirilmiş modelde 2 bin görsel %20 validasyon, %80 eğitim olarak bölünmüştür. Validasyon tahmin başarısına göre hiper parametreler güncellenmiştir. Geliştirilen ikinci modelde ise 500 adet test verisi kullanılmıştır. Tensorflow Lite kullanılarak geliştirilen üçüncü modelde ise %60 eğitim, %20 validasyon ve %20 test verisi kullanılmıştır. Validasyon ve test verisine göre hiper parametreler test edilmiştir ve tüm modellerin kıyaslandığı 500 adet görsel verinin bulunduğu test verisi ile kıyaslanmıştır. Bu modellerin aktivasyon fonksiyonu Doğrultulmuş Doğrusal Birim (relu), ağlar için uygun eniyileme algoritma türü olarak Adam ve son katmanlardaki aktivasyon fonksiyonu ise Softmax tercih edilmiştir.



Şekil 3.4 İlk kurulan modelin başarı - eğitim ve validasyon kayıp değerleri

Şekil 3.4'te görüldüğü üzere geliştirilen birinci modelin eğitim ve validasyon verilerinin kayıp ve başarı değerleri görülmektedir. Görsele bakarak, modelin genelleme yapabilmesinden daha çok, ezberleme yaptığı görülmüştür. VGG16 mimarisiyle geliştirilen ilk modelde, aşırı uyum olduğu fark edildikten sonra geliştirilen ikinci modele dropout denilen iletim sönümleme uygulanmıştır. Geoffrey Hinton ve arkadaşları tarafından geliştirilen bu yöntem, katmanlar arasında öğrenilen bilgilerin rastgele olarak sıfır yapılmasıdır. İletim sönümleme, görüntü işleme ve derin öğrenmede aşırı uyumdan kaçınmak için kullanılan en yaygın yöntemlerin başında gelmektedir. İletim sönümleme oranı genellikle 0.1 ile 0.5 arasında değer almaktadır.

Tablo 3.1 İlk kurulan modelin çapraz doğrulama tablosu

48	0	0	0	0	0	1	0	1	0
0	47	0	0	0	0	0	0	2	0
0	0	50	0	0	0	0	0	0	0
0	0	0	47	0	0	0	1	1	1
0	0	1	0	38	1	1	2	6	1
0	0	0	0	0	50	0	0	0	0
0	1	2	0	0	1	40	2	4	0
0	0	3	0	0	0	0	46	1	0
0	0	2	0	0	0	0	0	48	0
0	0	3	4	0	1	1	0	1	40

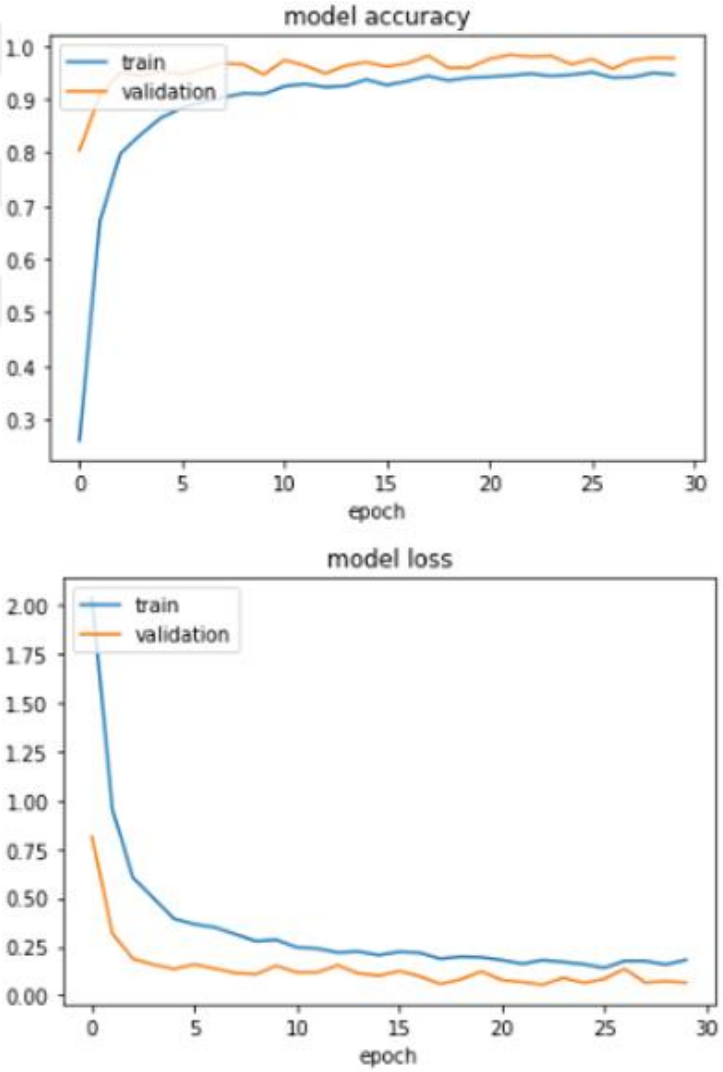
Kurulan ilk modelden sonra modelin tahmin ve test hatalarının kontrolü Tablo 3.1'de yapılmıştır. Çapraz doğrulama tablosundan modelin, hangi ilaç kutularını birbirine karıştırdığı ve öğrendiği yapı gereği birbirlerine benzediği görülmektedir. 5, 7 ve 10. sıralarındaki ilaç kutularını sınıflandırırken, model genel olarak diğerlerine göre daha fazla tahmin hatası yapmıştır. Model, ilaç kutuları arasındaki farkı tam olarak ayırt edememiş ve karıştırmıştır.

Derin katmanlar arasında gerçekleşen öğrenmede, en büyük ortaklama işlemleri sıklıkla kullanılmaktadır. En büyük ortaklama işlemleri ile filtrelenen ve üretilen değişkenlerin boyutları küçültülür. Böylelikle veri seti giderek saflaşan ve gereksiz bilgiler içermeyen bir matris olmuş olur. Bu işlem, matrislerden oluşan her görseli belli bir oranda küçültür. Matris değerleri bir açıdan özetlenmiş olur. Böylece öğrenme işlemi daha az ve saf veri üzerine gerçekleştirilir. Max Pooling katmanı, veri matrisinde belli bir kısmı seçer ve seçilen kısımdaki en büyük değeri alarak veriyi farklı bir şekilde ifade eder. Böylece veri matrisi küçülmüş ve farklı boyutlarla veri temsil edilmiş olur. Bu katmanda öğrenme işlemi gerçekleşmez. Öğrenme katmanına girecek matrisin kanal sayısı sabit tutulur, yükseklik ve genişlik bilgisi azaltılır. Şekil 3.1 ve Şekil 3.2'de görüldüğü üzere, özniteliklerin oluşturulduğu her iki-üç Conv2D katmanda bir Max Pooling katmanı bulunmaktadır ve bu katman, geliştirilen 3 modelde de kullanılmıştır.

Geliştirilen ikinci modelde, 2.000 görsel eğitim verisi ve 500 görsel test verisi kullanılmıştır ayrıca diğer geliştirilen modellerden farklı olarak sınıflandırma

başarısını artırmak için veri artırma işlemi yapılmıştır. Bu işlem sonucunda, geliştirilen ilk modelden daha başarılı sonuçlar elde edildiği görülmüştür.

Veri artırma işlemi, belirlenmiş kriterler üzerinden gerçekleştirilmiştir. İkinci model için sentetik veri üretilmiştir. Her bir görsel modele girmeden önce kendisinden üretilmiş yeni görsellerle modele girer. Veri artırılırken her görsel 90 derece dikey olarak döndürülmüştür. Bunlarla beraber çeşitli yakınlaştırma, görüntüyü kırma oranları ve gürültü ekleme gibi farklı adımlar uygulanmıştır. Böylece veri sayısı artırılarak modelin sınıflandırma başarısı yükseltilmeye çalışılmıştır. 30 tekrarlı çalıştırılan modelin kayıp değeri Şekil 3.5'te gösterilmiştir. Doğru sınıflandırma başarısı ise %96.9'dur.



Şekil 3.5 İkinci kurulan modelin başarı - eğitim ve validasyon kayıp değerleri

Uygulamada kullanılan 3. Model ise Tensorflow Lite kullanılarak geliştirilmiş bir modeldir. Tensorflow Lite kullanılarak eğitilmiş modeller, optimizasyona ağırlık veren, mobil telefonlar üzerinde derin öğrenme modelleri çalıştırabilmek için Google ekibi tarafından geliştirilmiştir. Optimizasyona önem veren Lite yöntemleri, hızlı çalışmakla beraber başarılı bir teknolojidir [50]. Mobil telefonlarda hızlı bir şekilde kullanılabilmesi için geliştirilen Tensorflow Lite kullanılarak geliştirilen modeller nesnelerin interneti gibi çeşitli projelerde de kullanılmaktadır. Ayrıca Tensorflow Lite kullanılarak geliştirilen modeller öğrenme aktarımına imkan sağlamaktadır.

Model name	Paper and model	Model size	Top-1 accuracy	Top-5 accuracy	CPU, 4 threads	GPU	NNAPI
DenseNet	paper, tflite&pb	43.6 Mb	64.2%	85.6%	195 ms	60 ms	1656 ms
SqueezeNet	paper, tflite&pb	5.0 Mb	49.0%	72.9%	36 ms	9.5 ms	18.5 ms
NASNet mobile	paper, tflite&pb	21.4 Mb	73.9%	91.5%	56 ms	---	102 ms
NASNet large	paper, tflite&pb	355.3 Mb	82.6%	96.1%	1170 ms	---	648 ms
ResNet_V2_101	paper, tflite&pb	178.3 Mb	76.8%	93.6%	526 ms	92 ms	1572 ms
Inception_V3	paper, tflite&pb	95.3 Mb	77.9%	93.8%	249 ms	56 ms	148 ms
Inception_V4	paper, tflite&pb	170.7 Mb	80.1%	95.1%	486 ms	93 ms	291 ms
Inception_ResNet_V2	paper, tflite&pb	121.0 Mb	77.5%	94.0%	422 ms	100 ms	201 ms
Mobilenet_V1_0.25_128	paper, tflite&pb	1.9 Mb	41.4%	66.2%	1.2 ms	1.6 ms	3 ms
Mobilenet_V1_0.25_160	paper, tflite&pb	1.9 Mb	45.4%	70.2%	1.7 ms	1.7 ms	3.2 ms
Mobilenet_V1_0.25_192	paper, tflite&pb	1.9 Mb	47.1%	72.0%	2.4 ms	1.8 ms	3.0 ms
Mobilenet_V1_0.25_224	paper, tflite&pb	1.9 Mb	49.7%	74.1%	3.3 ms	1.8 ms	3.6 ms
Mobilenet_V1_0.50_128	paper, tflite&pb	5.3 Mb	56.2%	79.3%	3.0 ms	1.7 ms	3.2 ms
Mobilenet_V1_0.50_160	paper, tflite&pb	5.3 Mb	59.0%	81.8%	4.4 ms	2.0 ms	4.0 ms
Mobilenet_V1_0.50_192	paper, tflite&pb	5.3 Mb	61.7%	83.5%	6.0 ms	2.5 ms	4.8 ms

Şekil 3.6 Tensorflow lite ile geliştirilebilecek mimarilerin boyutları - başarıları ve tahmin süresi [51]

Şekil 3.6'da görüldüğü üzere, Tensorflow Lite kullanılarak geliştirilen modeller çeşitli mimarilerde ve çeşitli verilerle eğitilmiş, ImageNet gibi, modelleri öğrenme aktarımı kullanmaya imkan sağlamaktadır. Tensorflow Lite kullanılarak geliştirilen modelde veri setinin %20'si test, %20'si validasyon ve %60'ı eğitim olarak kullanılmıştır. Validasyon verileri ile hiper parametreler güncellenip, 500 adet görsel verinin bulunduğu test verisi ile test edilmiştir. Aşırı öğrenmeden kaçınmak için modelde iletim sönümleme tercih edilmiştir. Öğrenmenin gerçekleştirildiği

katmanlardaki bilgiler rastgele 0.4 oranında değerleri sıfır yapılmıştır. Bu yöntem ile aşırı öğrenmeden kaçınılmıştır ve modelin ezberlemeden genelleme yapabilme yeteneği artırılmıştır. Bu modelde öznitelik çıkarmak ve veriyi öğrenmesi için bulunan katmanlarda toplam 3.413.024 parametre bulunmaktadır. Modelin sınıflandırma başarısı ise %97.6 olarak görülmüştür.

Çalışmada Tenforflow Lite kullanılarak geliştirilen 3. modelde EfficientNet model ailesi kullanılmıştır. EfficientNet modelleri, diğer modellerden farklı olarak mimarileri daha küçük ve hızlıdır. Bu modeller, tahmin başarılarını yükseltebilmek ve iyileştirebilmek adına modelde bazı değişiklikler yapar. Katman derinliği, çözünürlüğü ve genişliği dengeleyerek performansı artırır. ImageNet yarışmasında son yıllarda iyi sonuçlar alan model mimarileri ve yöntemleri incelendiğinde, farklı yaklaşımlar kullanan modellerin başarılarında yükselmelerin olduğu görülmektedir. EfficientNet modelleri birleşik bir katsayı kullanarak modelin genişliğini, çözünürlüğünü ve tüm derinliğini eşit olarak ölçeklendiren yeni bir ölçeklenme yöntemi kullanmaktadır. Bu sebeple EfficientNet modellerinin yarattığı en büyük fark ise genişlik, derinlik ve çözünürlüğü ölçeklendirmesidir. Tensorflow Lite arka planda AutoML gibi çalışmaktadır. EfficientNet model aileleri, katman sayılarını, nöron sayılarını ve çözünürlüğünü kendi algoritmasıyla belirlemektedir.

ESA modellerinde daha iyi bir doğruluk oranı ve verimlilik sağlayan EfficientNet modelleri, sinir ağı mimarisi aramasını kullanır. Bu nedenle, her veride farklı bir mimari ve derinlik kullanabilir. Bu optimizasyonu ve dengelemeyi kullanmak için grid search kullanmaktadır.

Modelin derinliğini dengeleyerek gradyanların ölmesi problemini engellemeye yardımcı olur. Çok derin mimarilerde gradyanların ölmesi durumu yaşanmaktadır. Katman sayılarını dengeleyerek model karmaşıklığı engellenir. Genişlik parametrelerini dengeleyerek, nöron sayılarında ayarlamalar yapar. Böylece gereksiz detayları öğrenerek modelin eğitim süresini optimize eder. Çözünürlük parametresinde de buna benzer işlemler yapmaktadır. Ayrıca atlama bağlantıları ile model eğitim süresinde iyilemeler yapar.

4
SONUÇ VE ÖNERİLER

4.1 Tartışma ve Sonuç

Çalışma kapsamında 10 farklı sınıftan toplam 2.500 adet görsel veri bulunmaktadır ve 10 farklı ilaç kutusunun çeşitli derin öğrenme yöntemleri ile sınıflandırılması amaçlanmıştır. Çalışmada kullanılan ilaç kutuları belirlenirken her evde bulunabilecek en yaygın ilaç kutuları seçilerek, üretilen görsel veriler algoritmalarca sınıflandırılmıştır. Sınıflandırma işlemleri yapılırken 3 farklı model geliştirilmiştir. Bu kapsamda veri ön işleme, sentetik veri üretme, transfer öğrenme, VGG16 mimarileri ve Tensorflow Lite kullanılarak geliştirilmiş Evrişimsel Sinir Ağları uygulanmıştır.

VGG16 mimarili Evrişimsel Sinir Ağları ile geliştirilen modellerin başarısı, Tensorflow Lite kullanılarak geliştiriken modelin başarısını geçememiştir. Transfer öğrenme kullanılan modellerde aşırı uyumu engelleyebilmek adına iletim sönümleme kullanılsa da modellerde aşırı uyum ve az öğrenmenin mevcut olduğu görülmüştür. Aşırı uyumdan kaçınmak için ayrıca daha düşük mimari karmaşıklık ve parametre sayıları kullanılmıştır. Daha az parametre kullanılarak aşırı uyumdan kaçınıldığı görülmüştür ve bulunan bu sonuç literatürü destekler niteliktedir.

Tablo 4.1 Model kıyaslama tablosu

Model Adı	Parametre Sayısı	Eğitim Süresi(saniye)	Tekrar Sayısı	Başarı Yüzdesi
1. Model	134.301.514	1.105	100	% 89
2. Model	14.982.474	16.116	30	% 96.9
3. Model	3,413,024	4.150	50	% 97.6

4.2 Öneriler

Çalışma kapsamında geliştirilen modeller bilgisayarlı görü kullanarak görüntü sınıflandırma yapmaktadır. Modellerin sınıflandırma performansını yükseltebilmek için görüntüden metine (image to text) yöntemleri ile desteklenmeye açıktır. Görüntü verisinden metinler alınarak, görüntü sınıflandırma modelinden çıkan sonuç ile eşleşmesi durumunda daha doğru sınıflandırma yapılabileceği öngörülmektedir. Tensorflow Lite kullanılarak geliştirilen modeller gelişime açık, mobil uygulamalar / telefonlar için uyumlu, hızlı ve yeni bir model türü olduğu için gelecekte literatürde önemli bir yere sahip olacaktır.

KAYNAKÇA

[1] I. N. Aizenberg, N. N. Aizenberg, and J. Vandewalle, "Multiple – Values Threshold Logic and Multi – Values Neurons," In Multi – Values and Universal Binary Neurons, Springer, 2000, pp. 25-80.

[2] G. Doğan, "Yapay Sinir Ağları Kullanılarak Türkiye'deki Özel Bir Sigorta Şirketinde Portföy Değerlendirmesi," Yüksek Lisans Tezi, Hacettepe Üniversitesi, 2010.

[3] E. Cengil and A. Çınar, "A New Approach for Image Classification: Convolutional Neural Network," European Journal of Technic, vol. 6, no. 2, pp. 96-103, 2016.

[4] F. Chollet, "Deep Learning with Python," Manning Publications, 2017.

[5] MNIST database, http://yann.lecun.com/exdb/mnist/, 02 Kasım 2019.

[6] L. Deng and D. Yu, "Deep Learning: Methods and Applications," Foundations and Trends Signal Processing, vol. 7, no. 3, pp. 197-387, 2014.

[7] A. Şeker, B. Diri, and H. H. Balık, "A Review about Deep Learning Methods and Applications," Journal of Engineering Sciences, vol. 3, no. 3, pp. 47-64, 2017.

[8] W. S. McCulloch, and W. Pitts, "A Logical Calculus of the Ideas Immanent in Nervous Activity," The Bulletin of Mathematical Biophysics, vol. 5, no. 4, pp. 115-133, 1943.

[9] History of Deep Learning, https://maelfabien.github.io/deeplearning/Perceptron/, 12 Mart 2020.

[10] T. Liu, S. Fang, Y. Zhao, P. Wang, and J. Zhang, "Implementation of Training Convolutional Neural Networks," 2015.

[11] A. G. Ivakhnenko and V. G. Lapa, "Cybernetic Predicting Devices," Joint Publications Research Service, 1966.

[12] F. Doğan and I. Türkoglu, "The Comparison of Leaf Classification Performance of Deep Learning Algorithms," Sakarya University Journal of Computer and Information Sciences, vol. 1, no.1, 2018.

[13] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors," 2012.

[14] K. Sabanci, C. Aydin, and M. F. Ünlerşen, "Determination of Classification Parameters of Potatoes with the Help of Image Processing and Artificial Neural Network," Journal of the Institute of Science and Technology, vol. 2, no. 2, pp. 59-62, 2012.

[15] Y. Lecun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition," Neural Computation, vol. 1, no. 4, pp. 541–551, 1989.

[16] A. Krizhevsky, I. Sutskever, and H. Geoffrey, “ImageNet Classification with Deep Convolutional Neural Networks,” Advances in Neural Information Processing Systems, vol. 25, no. 2, 2012.

[17] R. Can, S. Kocaman, and C. Gokceoglu, “A Convolutional Neural Network Architecture for Auto-Detection of Landslide Photographs to Assess Citizen Science and Volunteered Geographic Information Data Quality,” ISPRS International Journal Geographic Information, vol. 8, no. 7, 2019.

[18] A. E. Çağlayan, “A New Horizon in Econometrics: Big Data and Machine Learning,” Social Sciences Research Journal, vol. 7, no. 2, pp. 41-53, 2018.

[19] Big Data in Zettabytes, https://www.slideshare.net/mciobo/big-data-and-the-creative-destruction-of-todays-business-models, 29 Aralık 2019.

[20] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative Adversarial Nets,” Advances in Neural Information Processing Systems, 2014, pp. 2672 – 2680.

[21] L. Torrey and J. Shavlik, “Transfer Learning,” 2009.

[22] Deep Learning Inception V3 Architecture in Tensorflow, https://github.com/tensorflow/models/tree/master/research/inception, 8 Ocak 2020.

[23] J. Devlin, M. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” 2018.

[24] Bert Algorithm, https://github.com/google-research/bert, 27 Şubat 2020.

[25] ImageNet Dataset Official Website, http://www.image-net.org/, 27 Şubat 2020.

[26] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[27] Ö. İnik and E. Ülker, “Deep Learning and Deep Learning Models Used in Image Analysis,” Gaziosmanpasa Journal of Scientific Research, vol. 6, no. 3, pp. 85-104, 2017.

[28] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," Advances in Neural Information Processing Systems, 2012, pp. 1097 – 1105.

[29] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks," European Conference on Computer Vision, 2014, pp. 818-833.

[30] C. Szegedy, W. Liu, Y. Q. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions." 2015 IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1-9.

[31] K. M. He, X. Y. Zhang, S. Q. Ren, and J. Sun, "Deep Residual Learning for Image Recognition." IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770-778.

[32] F. Kurt, “Analysis of the Effects of Hyperparameters in Convolutional Neural Networks,” Yüksek Lisans Tezi, Hacettepe Üniversitesi, 2018.

[33] Keras Official Website, https://keras.io/, 2 Ocak 2020.

[34] Tensorflow Official Website, https://www.tensorflow.org/, 2 Ocak 2020.

[35] J. W. Tukey, "The Future of Data Analysis," The Annals of Mathematical Statistics, vol. 33, no. 1, pp.1-67, 1962.

[36] R. B. Mete, "Derin Öğrenme ile Görüntü Sınıflandırma," Yüksek Lisans Tezi, İstanbul Üniversitesi, 2019.

[37] V. Dumoulin and F. Vision, "A Guide to Convolution Arithmetic for Deep Learning," 2016, Available: https://arxiv.org/abs/1603.07285.

[38] Maksimum ve Ortalama Ortaklama İşlemi, https://medium.com/@r.guven887/evri%C5%9Fimli-sinir-a%C4%9Flar%C4%B1nda-cnn-shifted-problem-kayma-problemi-6221b49e61c, 18 Şubat 2020.

[39] Az Öğrenme - Aşırı Uyum ile Yanlılık - Varyans İlişkisi, https://towardsdatascience.com/is-overfitting-really-bad-ddaabfc2f6b8, 20 Şubat 2020.

[40] Toplam Hata ve Varyans, http://scott.fortmann-roe.com/docs/BiasVariance.html, 21 Mart 2020.

[41] Aktivasyon Fonksiyonları, https://medium.com/@ayyucekizrak/derin-%C3%B6%C4%9Frenme-i%C3%A7in-aktivasyon-fonksiyonlar%C4%B1n%C4%B1n-kar%C5%9F%C4%B1la%C5%9Ft%C4%B1r%C4%B1lmas%C4%B1-cee17fd1d9cd, 19 Mart 2020.

[42] Aktivasyon Fonksiyonları, http://buyukveri.firat.edu.tr/2018/04/17/derin-sinir-aglari-icin-aktivasyon-fonksiyonlari/, 21 Mart 2020.

[43] Aktivasyon Fonksiyonları, http://www.ibrahimcayiroglu.com/Dokumanlar/IleriAlgoritmaAnalizi/IleriAlgoritmaAnalizi-5.Hafta-YapaySinirAglari.pdf, 20 Mart 2020.

[44] E. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. Le, "AutoAugment: Learning Augmentation Strategies From Data," IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 113-123.

[45] Sentetik Veri Üretimi, https://medium.com/novaresearchlab/veri-art%C4%B1r%C4%B1m%C4%B1-data-augmentation-part-2-5b8f3b4a2803, 18 Mart 2020.

[46] S. Lim, I. Kim, T. Kim, C. Kim, and S. Kim, "Fast AutoAugment," 2019. Available: https://arxiv.org/abs/1905.00397.

[47] B. Zoph, E. D. Cubuk, G. Ghiasi, T. Lin, J. Shlens, and Q. V. Le, "Learning Data Augmentation Strategies for Object Detection," 2019.

[48] Görüntü İşleme, http://www.ibrahimcayiroglu.com/Dokumanlar/GoruntuIsleme/Goruntu_Isleme_Ders_Notlari-1.Hafta.pdf, 20 Mart 2020.

[49] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large - Scale Image Recognition,” 2014. Available: http://arxiv.org/abs/1409.1556.

[50] Tensorflow Lite, https://www.veribilimiokulu.com/tensorflow-lite-derin-ogrenme/, 14 Ocak 2020.

[51] Tensorflow Lite Host, https://www.tensorflow.org/lite/guide/hosted_models, 20 Şubat 2020.

TEZDEN ÜRETİLMİŞ YAYINLAR

İletişim Bilgisi: utkukubilaycinar@gmail.com

Sempozyum Bildirileri

1. U. K. Çınar, and G. G. Şimşek, "Image Classification Using Deep Learning Algorithms", 20th International Symposium on Econometrics, Operations Research and Statistics, 2020, pp. 272.