T.C.
MANİSA CELAL BAYAR ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ
YAZILIM MÜHENDİSLİĞİ ANA BİLİM DALI
YAZILIM MÜHENDİSLİĞİ BİLİM DALI

TRANSFER ÖĞRENMESİ TEKNİĞİ TABANLI DERİN ÖĞRENME YÖNTEMİYLE ÜRÜN TANIMA

Kübra AKGÖZLÜOĞLU

Danışman
Doç. Dr. Akın ÖZÇİFT

MANİSA-2021

Kübra
AKGÖZLÜOĞLU

TRANSFER ÖĞRENMESİ TEKNİĞİ TABANLI DERİN ÖĞRENME YÖNTEMİYLE ÜRÜN TANIMA

2021

TAAHHÜTNAME

Bu tezin Manisa Celal Bayar Üniversitesi Hasan Ferdi Turgutlu Teknoloji Fakültesi Yazılım Mühendisliği Bölümünde, akademik ve etik kurallara uygun olarak yazıldığını ve kullanılan tüm literatür bilgilerinin referans gösterilerek tezde yer aldığını beyan ederim.

Kübra AKGÖZLÜOĞLU

İÇİNDEKİLER

SİMGELER VE KISALTMALAR DİZİNİ

Acc	Başarım (Accuracy)
b	Bias
CNN	Convolutional Neural Network (Evrişimli Sinir Ağı)
Dk	Dakika
Loss	Kayıp
SGD	Stochastic Gradient Descent
W	Ağırlık (Weight)
YSA	Yapay Sinir Ağları

ŞEKİLLER DİZİNİ

Sayfa

TABLO DİZİNİ

Sayfa

TEŞEKKÜR

Çalışmamın her aşamasında bana destek olan, bilgi ve deneyimleri ile yol gösteren danışman hocam Sayın Doç. Dr. Akın Özçift'e, lisansüstü öğrenimin bana kazandırdığı birlikte çalışıp her daim birbirimize destek olduğumuz sevgili arkadaşlarım Batuhan, Sevgi ve Yeşim'e, yorulup zorlandığım olumsuzluğa kapıldığım anlarda hep yanımda olup her an beni cesaretlendiren her konuda desteğini esirgemeyen değerli eşim Serkan Akgözlüoğlu'na ve öğrenim hayatım boyunca beni maddi ve manevi olarak destekleyen her zaman arkamda duran bir ömür sevgisini esirgemeyen ve hep yanımda olan canım aileme yürekten teşekkür ederim.

Kübra AKGÖZLÜOĞLU
Manisa, 2021

ÖZET

Yüksek Lisans Tezi

TRANSFER ÖĞRENMESİ TEKNİĞİ TABANLI DERİN ÖĞRENME YÖNTEMİYLE ÜRÜN TANIMA

Kübra AKGÖZLÜOĞLU

Manisa Celal Bayar Üniversitesi
Fen Bilimleri Enstitüsü
Yazılım Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Akın ÖZÇİFT

Üretim yapan fabrikalarda üretim esnasında makine ve robotların payı oldukça fazla olmasına rağmen, üretimde insan faktörü olmazsa olmaz sektörler mevcuttur. Üretimin herhangi bir aşamasında insan eliyle üretim söz konusu ise hatalı ürün üretilmesi kaçınılmazdır. Yine kalite testlerinin insan eliyle olması üretilen bu hatalı ürünlerin müşteriye gitmeden elenme olasılığını düşürmektedir. Hatalı ürün üretiminin minimuma indirilmesi için daha önce hatalı ve doğru ürünler ile eğitilmiş bir modelin, üretim bandı sonunda ürünlerin kontrolü için uygulanması kalite departmanında çalışan insan yükünün azalmasını ve hatalı ürünlerin tespitinin kolaylaşmasını sağlayacaktır.

Yüksek başarı oranına sahip iyi bir sınıflandırma modeli oluşturmak için bu modelin birçok örnek ile eğitilmesi gerekmektedir. Yeterli sayıda etiketlenmiş veri örneği edinmek her zaman mümkün olmayabilir. Model başarımını arttırmak için yeterli sayıda veri olmadığı durumlarda transfer öğrenme modeli başarım oranını arttırdığı birçok çalışmayla literatürde gözlemlenmektedir. Bu çalışmada elektrik ve su sayacı üreten bir fabrikada üretim esnasında hatalı ürünlerin tespiti için tek bir sayaç modeli üzerinden elde edilen hatalı ve doğru ürünleri içeren veri seti oluşturulmuştur. Elde edilen veri setinde ürün sayısı düşük olduğu için transfer öğrenme yaklaşımıyla deneysel çalışma yürütülmüştür.

Bu çalışmada, transfer öğrenme tekniğinden önce veri seti CNN algoritması ile eğitilmiş ve modelin başarı oranı %88 olarak tespit edilmiştir. Transfer öğrenme algoritmalarından VGG16 algoritması ile oluşturulan model %99.5 başarı oranı gösterirken ResNet50 ile %99.4 ve Xception ile %100 başarı oranları elde edilmiştir. Sonuç olarak kullanılan veri setinde transfer öğrenme algoritmalarının başarı oranı CNN algoritmasına oranla daha yüksek olduğu gözlemlenmiş ve en yüksek başarı oranına ortalama bir sürede eğitimi gerçekleştiren Xception transfer öğrenme algoritması olmuştur.

Anahtar Kelimeler: Derin Öğrenme, Transfer Öğrenme, Ürün Tespiti, VGG16, ResNet50, Xception, CNN

2021, 51 sayfa

ABSTRACT

M.Sc. Thesis

PRODUCT RECOGNITION WITH DEEP LEARNING METHOD BASED ON TRANSFER LEARNING TECHNIQUE

Kübra AKGÖZLÜOĞLU

Manisa Celal Bayar University
Graduate School of Applied and Natural Sciences
Department of Software Engineering

Supervisor: Assoc. Prof. Dr. Akın ÖZÇİFT

Although the share of machines and robots during production is quite high in manufacturing factories, there are sectors in which the human factor is indispensable in production. If there is man-made production at any stage of production, it is inevitable to produce faulty products. Again, human-made quality tests reduce the possibility of eliminating these faulty products before they go to the customer. In order to minimize the production of faulty products, the application of a previously trained model with faulty and correct products for the control of the products at the end of the production line will reduce the human burden working in the quality department and facilitate the detection of faulty products.

In order to create a good classification model with a high success rate, this model needs to be trained with many examples. It may not always be possible to obtain a sufficient number of labeled data samples. In cases where there is not enough data to increase the model performance, it is observed in the literature with many studies that the transfer learning model increases the performance rate. In this study, a data set containing faulty and correct products obtained from a counter model was created to detect faulty products during production in a factory that produces electricity and water meters. Since the number of products in the obtained data set was low, an experimental study was carried out with the transfer learning approach.

In this study, before the transfer learning technique, the data set was trained with the CNN algorithm and the success rate of the model was determined as 88%. The model created with the VGG16 algorithm, one of the transfer learning algorithms, showed a 99.5% success rate, while 99.4% with ResNet50 and 100% with Xception. As a result, it was observed that the success rate of the transfer learning algorithms in the data set used was higher than the CNN algorithm, and the Xception transfer learning algorithm, which performed the training in an average time, had the highest success rate.

Keywords: Deep Learning, Transfer Learning, Product Recognation, VGG16, ResNet50, Xception, CNN

2021, 51 pages

1. GİRİŞ

Üretim kuşkusuz bir ülkenin gelişiminin bel kemiğidir. Üretmeyen toplum, gelişmiş ülkelerden geri kalmaya ve ihtiyaçları için başka ülkelere bağımlı olmaya mahkumdur. Üretim bu denli önemliyken, üretim yapan fabrikaların ürün kalitesinin ölçümü, daha hızlı ve daha fazla ürün üretebilmeleri için teknolojik gelişmelere adapte olması gerekmektedir. Birçok fabrika günümüzde robotik çözümlerle otomatik üretime geçmiş olsa da üretim aşamasında hala insan elinin değmek zorunda olduğu ürünlerin üretimi devam etmektedir. İnsan faktörünün etkinliğinin bulunduğu ürünlerde hatalı üretim ister istemez söz konusu olacak ve binlerce ürün arasında kalite testleri için birkaç örnek ile yapılan testler, hatalı ürün tespitinde doğru sonuç vermeyecektir. Gözden kaçan hatalı ürünler müşteri memnuniyetsizliği, ürünün türüne göre yaşam kalitesini düşürmesi ve üretim yapan fabrika için maddi-manevi yükümlülüklere neden olacaktır. Bu durum göz önüne alındığında daha önce doğru ve yanlış örneklerle eğitilen bir derin öğrenme modeli üretim sonunda hatalı olan ürünlerin tespitini gerçekleştirebilir ve müşteriye gönderim öncesinde önlem alınıp ciddi yükümlülüklerin önüne geçilebilir.

Elektrik ve su sayaçları üretimi yapan bir fabrikada ana kart üretiminin temel küçük parçaların dizilimi bir cihazda gerçekleştirildikten sonra, ürünün modeline göre üzerine eklenecek pil, direnç, transistör gibi elementlerin kart üzerine entegrasyonu bant üzerinde çalışan personel ile gerçekleştirilmektedir Teslimat öncesi uygun sayıda kolilerden rastgele seçilen sayaçların kalite kontrolleri yapılıp müşteriye teslim edilmek üzerine lojistik işlemleri başlatılmaktadır. Dağıtım şirketleri tarafından son kullanıcı olan bizlerin kullanımına sunulan testler sırasında gözden kaçan hatalı sayaçlarda oluşan sorunlar mağduriyet yaratmaktadır. Bu mağduriyeti minimuma indirmek adına üretim bandının sonuna yerleştirilecek bir kamera derin öğrenme modeli ile hatalı ürünleri tespit ederek iş akışında önlemler alınmasına yardımcı olup hatalı ürünlerin müşteriye gönderilmesinin engellemesini sağlayabilecektir.

Tek bir sayaç modeli üzerinde oluşabilecek hatalı ürünlerin kombinasyonları oluşturulup eşit sayıda hatalı ve doğru ürün görseli ile bir veri seti elde edilmiştir. Yeterli başarı oranına sahip iyi bir sınıflandırma yapan model üretmek için birçok

eğitim verisine ihtiyaç duyulmaktadır. Elde edilen veri setinin eğitim için yeterli sayıda olmaması üzerine transfer öğrenmesi algoritmaları kullanılması kararlaştırılıp bu alanda en uygun algoritmanın tespiti ile deneysel çalışma yürütülmüştür. Bu çalışma ile üretim sırasında hatalı ürünlerin tahminlemesi yapılıp, hatalı ürünlerin bant üzerinde başka bir alana taşınmasıyla müşteriye gönderilmeden önce hatalı ürünün tespit edilmesi amaçlanmıştır.

2. LİTERATÜR TARAMASI

Begüm K. yüksek lisans tez çalışmasında MOBESE kameralarından elde edilen 169 farklı araç modelinin belirlenmesinde transfer öğrenmesi yöntemini kullanmıştır. Transfer öğrenmesi için AlexNet, GoogleNet ve ResNet-50 modellerini seçerek gerçekleştirdiği çalışmada sınıflandırıcı olarak Softmax ve Dektek Vektör Makinesi kullanmıştır. Çalışma sonucunda GoogleNet+Softmax modeli ile %98.73 başarı oranı ile 169 farklı araç modelini tanımlayabilmiştir [1].

Nizam Abdulghani S. ve ark. bu çalışmalarında, meme kanserini sınıflandırmak için önceden eğitilmiş CNN modeli ile derin öznitelik çıkarım yöntemi kullanılmıştır. Çalışma, dört farklı büyütme faktörüne sahip iyi huylu ve kötü huylu olmak üzere iki sınıf içeren halka açık yayınlanan BreakHis veri seti ile gerçekleştirilmiştir. Yama stratejisi ile AlexNet modeli ele alınmış ve sistemin ince ayarı için önceden eğitilmiş AlexNet kullanılmıştır. Destek Vektör Makinesi (SVM) kullanılarak elde edilen değerlendirme sonuçları SVM sınıflandırması ve yama stratejisi ile önceden eğitilmiş Alexnet'in en iyi doğruluğu verdiğini göstermektedir. Farklı büyütme faktörleri için beş kat çapraz doğrulama tekniği kullanılarak %92 ile %96 arasında doğruluk elde edilmiştir[2].

Fırıldak K. ve Talu M. F. makalelerinde Evrişimli Sinir Ağlarında (ESA) kullanılan transfer öğrenme yaklaşımlarını incelemişlerdir. Cifar, Caltech, Mnist veri kümeleri için AlexNet'den transfer edilen ağırlıklarla sınıflama başarılarını değerlendirmişlerdir. AlexNet'in farklı veri setlerinde sınıflama başarısının yüksek olduğu ve yeni veri setinde sınıflarının AlexNet'in eğitim veri seti ile benzerliği arttıkça transfer öğrenmede sınıflandırma başarısının arttığını gözlemlemişlerdir. Transfer öğrenme ile farklı veri setleri için %80-%90 ortalama sınıflama başarısına ulaştığını makalelerinde açıklamışlardır [3].

Karahan T. ve Nabiyev V. konvolüsyonel yapay sinir ağlarından ve öğrenme transferinden yararlanarak otomatik bitki tanıma çalışması gerçekleştirmişlerdir. Diğer veri tabanlarından ve web ortamından toplayarak elde ettikleri veri seti toplam 76 cinse ait 5345 bitkiden oluşmaktadır. 65 cins çeşitli çiçek türlerinden 11 cins ise diğer bitki çeşitlerinden oluşmaktadır. Çeşitli veri çoğaltma ve normalizasyon

işlemlerinden geçirerek geliştirdikleri model eğitim veri seti için 0.9971 , test veri seti için 0.9897 isabet oranı elde etmişlerdir [4].

Kaya, A. ve arkadaşları çalışmalarında otomatik çiçek tanımlama üzerine transfer öğrenme yöntemlerini ve farklı transfer öğrenme senaryolarını analiz etmişlerdir. Beş sınıflandırma modeli tasarlamış ve modellerin değerlendirilmesi için dört farklı veri seti kullanmışlardır. Performansı artırmak için derin öğrenme modellerinde transfer öğrenmeyi kullanmanın olumlu etkileri bu çalışmada göstermişlerdir. [5].

Mario Lasseck, Lifeclef 2017 bitki tanımlama yarışmasında sunduğu çalışma ile 10.000 türü sınıflandırmak için Derin Evrişimsel Ağlar ve transfer öğrenmeyi ince ayar (fine tuning) yoluyla kullanmış ve farklı veri kümeleri üzerinde birkaç modeli eğitmiştir. Her bir görüntüden rastgele konumlarda, kare yamaları kırpma, yatay çevirme, döndürme, rastgele doygunluk değişimi ve uygulanan rastgele hafiflik değişimi gibi yöntemlerle veri artırma tekniklerini kullanmıştır. Sistemi, resmi PlantCLEF test setinde %92'lik bir ortalama karşılıklı sıralama (MRR) ve %96'lık bir ilk 5 doğruluk elde etmiştir [6].

Abanoz, H. çalışmasında öncelikle transfer öğrenme için kullanılacak bir temel model bulmaya çalışmaktadır. Bu amaçla, farklı veri kümeleri kullanılarak eğitilmiş farklı CNN mimarileri ve modelleri incelenmiştir. ImageNET veri seti kullanılarak eğitilen popüler VGG16, ResNet50, InceptionV3 modelleri denenmiştir. Oxford Face veri seti üzerinde de eğitilmiş VGGFace modeli denenmiştir. VGG16 ve VGGFace modelleri en umut verici sonuçları gösterdiği belirtilmiş ve VGGFace modeli, ön eğitim için kullanılan veri setinin duygu tanıma veri setine benzemesi nedeniyle transfer öğrenmesi için en uygun temel model olarak bulunmuştur. Deneyler, transfer öğrenmenin iyi bir sınıflandırıcı oluşturmaya yardımcı olduğunu göstermektedir. Çalışma sonucunda en iyi sınıflandırıcı, FER13 doğrulama setinde %69.49 doğruluk göstermiştir [7].

Şengür, D. transfer öğrenme ve önceden eğitilmiş CNN AlexNet ve VGG16 modelleri öznitelik çıkarımı için ele alınmış ve destek vektör makineleri (SVM) ile

sınıflandırılmıştır. Görüntü seviyelerinde %90,5 ile %91,4 arasında bir doğruluk elde edilmiştir [23].

Kassani ve arkadaşları çalışmasında; InceptionV3, Xception ve VGG Net modelleri de dahil olmak üzere, içinde beş adet önceden eğitilmiş Derin Evrişimli Sinir Ağı mimarisini öznitelik çıkarıcı olarak kullanmışlardır. Yazarlar, sınıflandırmanın performansını iyileştirmek için veri arttırma yöntemi uygulamışlar ve deney sonuçlarında önceden eğitilmiş eğitilmiş Xception modeli, diğer modeller arasında %92,50 ile en iyi ortalama sınıflandırma doğruluğunu vermiştir [27].

3. MAKİNE ÖĞRENMESİ (MACHINE LEARNING)

3.1. Makine Öğrenmesi Nedir?

Makine öğrenmesi; belirli bir problemin, bu probleme özgü veri ile modellenmesini sağlayan algoritmaların genel adıdır. Veri seti ve uygun algoritma ile tasarlanan model, performansı en yüksek olacak şekilde kurgulanmaktadır. [10]

Makine öğrenmesinde amaç, belirli girdilerle eğitilen bir model oluşturulup makinenin öğrenmesi ve insana özgü yetkinlikler olan analiz etme, çıkarımlarda bulunma gibi veriden öğrenip yeni değerler için tahminler yapmasını sağlamaktır. Belirli bir görev seti üzerinde aşamalı olarak gelişmek için bir algoritma öğretmeye odaklanır. Özetle makine öğrenimi bir sürecin nasıl çalıştığına dair bir model tasarlar ve tekrarlı geliştirme sunan uygulamaları pratik olarak oluşturmak için modeli uygular. Yeni bir örnek geldiğinde geçmiş verilerden bilgileri işleyerek hangi yolu izleyeceğini belirler ve her yeni örnek için gelişimini sürdür. Makine öğrenmesi temelde öğrenme ve kendini geliştirme işlevi görür.

Makine öğrenmesi, yapay zeka çalışmalarının bir alt dalıdır. Tüm makine öğrenme problemleri bir yapay zeka problemi olarak incelenirken tüm yapay zeka problemleri makine öğrenimi problemi olarak değerlendirilemez. Makine öğrenimi uygulamalı bir yapay zeka şeklidir. Veri toplama, depolama, ayrıştırma, veriyi analiz etme gibi yöntemlerle makinenin kendi başına karar verme yeteneği geliştirmesi sağlanarak insanlar gibi öğrenmesi amaçlanmıştır. Öngörülebilir örüntü veya anlamlı verileri ortaya çıkartmak için eldeki tüm verilerin analiz edilmesi gerekmektedir. Bundan sonrasında istatistiksel analiz kullanılarak geçmiş verilere göre çıkarım yapma yeteneğine sahip yeni verilere karşı tahminler yapmak için bir model oluşturulur [8]. Oluşturulan model çeşitli metrikler ile test edilerek başarı ve hata oranları tespit edilir ve her yeni veri ile öğrenme süreci geliştirilerek hata oranı azaltılıp başarı oranı arttırmak amaçlanır.

3.2. Makine Öğrenmesi Temel Kavramlar

Makine öğrenmesinde kendi kendine öğrenen dinamik bir model oluşturmak için belirli algoritmalar kullanılmakta ve yöntemler izlenmektedir. Bu algoritma ve

yöntemler uygulanırken bilinmesi gereken temel terminoloji aşağıdaki maddelerle özetlenmiştir:

Örnekler (Samples): Makinenin öğrenmesi ve değerlendirmesi için kullanılan her bir veri parçasına örnek ya da gözlem olarak ifade edilir.

Öz nitelikler (Features): Bir gözlemi temsil eden verilerdir. Bu bir kiralık ev fiyatı tahminleme probleminde evin konumu, oda sayısı, yaşı gibi özellikler olabilirken spam e-posta sınıflandırma probleminde e posta içeriğindeki kelimeler, e-posta uzunluğu gibi özellikler olabilmektedir.

Etiketler (Labels): Bir sınıflandırma probleminde gözlemdeki kategoriler. E-posta örneğinde spam/spam değil ya da hayvan resimlerini tanıma probleminde kedi/köpek/kuş vb. Sınıflar örnek gösterilebilir.

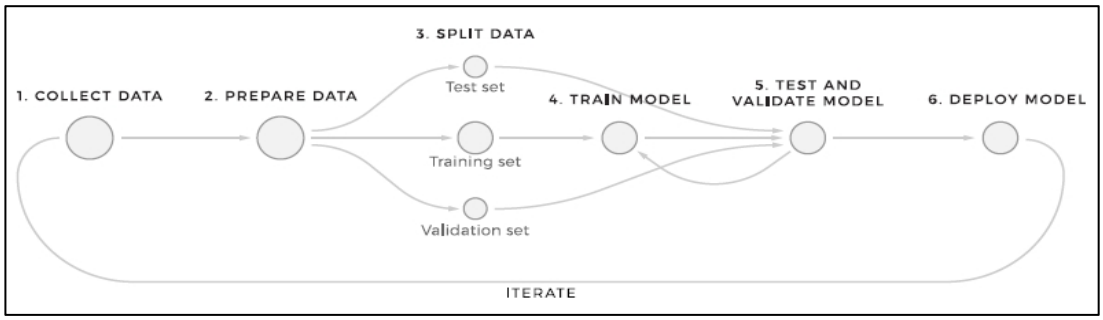
Eğitim verisi: Oluşturulmak istenen makine öğrenmesi modelinin eğitilmesi için kullanılan analiz edilmiş, ön hazırlık sürecinden geçirilmiş örneklerden oluşan veri setidir.

Test verisi: Eğitim verisi ile eğitilmiş makine öğrenmesi modelinin başarı oranını arttırmak ve hata oranını azaltmak için test edilecek test verisidir.

Aşırı öğrenme (Overfitting): Oluşturulan modelin veri seti üzerinde ezber yapmaya başlaması durumudur. Eğitim seti tek düze oluşturulmuş ve etiket ağırlıkları doğru dağıtılmamış ise modelin aşırı öğrenme olasılığı artacaktır. Eğitim setinde yüksek skor alınıp test veri setinde düşük skorlar elde ediliyorsa model eğitim setindeki değerleri ezberlemiş demektir. Bu durum genellikle modelde gözlem sayısına oranla çok fazla değişken varsa gerçekleşir.

Eksik öğrenme (Underfitting): Aşırı öğrenme oluşumunun aksi bir durumudur. Bir modelde öğrenme yetersiz kaldıysa modelin eğitim verilerine uygun ilerlemediği ve yeni verileri genelleştiremediği anlamına gelir. Eksik öğrenme sorunu olan modellerde eğitim ve test veri setlerinin ikisinde de hata oranı yüksek olmaktadır.

Makine öğrenmesi belirli bir iş akışında ilerlemektedir. Bu iş akışında öncelikle veri toplama gelmektedir. Makine öğrenmesinde oluşturulacak model için ne kadar çok örnek o kadar başarı oranını arttıracaktır. Bu nedenle farklı kaynaklardan olabildiğince çok örnek toplamak önemlidir. Toplanan veri ikinci adımda hazırlanma sürecinden geçecektir ki bu süreç anlamlı veri oluşturabilmek adına çok önemlidir. Farklı kaynaklardan toplanan veri aynı amaca hizmet etmek adına ve de kullanılacak algoritmaların anlamlandırabilmesi için ayıklama, normalleştirme, eksik verilerin çeşitli yöntemlerle doldurulması gibi adımlardan geçerek ön hazırlık sürecini tamamlamaktadır. Bir sonraki aşamada elde edilen normalleştirilmiş temiz veri model eğitimi ve test aşamasında kullanılmak üzere çeşitli yöntemlerle eğitim ve test verisi olarak ikiye ayrılmalıdır. Bazı çalışmalarda problemin çeşidine göre ihtiyaçlar dahilinde doğrulama çalışmaları için bir de doğrulama verisine ihtiyaç duyulur. Bu gibi durumlarda veri seti belli oranlarda üçe bölünerek eğitim, doğrulama ve test veri setleri elde edilmektedir. Bölünen veriler öncelikle eğitim verisi ile modelin eğitiminde kullanılır. Sonrasında test ve doğrulama veri setleri ile model test edilip değerlendirme metrikleri ile kontrolü sağlanır. Oluşturulan modelin kullanılması sağlanarak zaman içerisinde toplanan yeni verilerle modelin sürekli iyileştirilmesi sağlanır.



Şekil 3.1. Makine Öğrenmesi Akışı

3.3. Makine Öğrenmesi Türleri

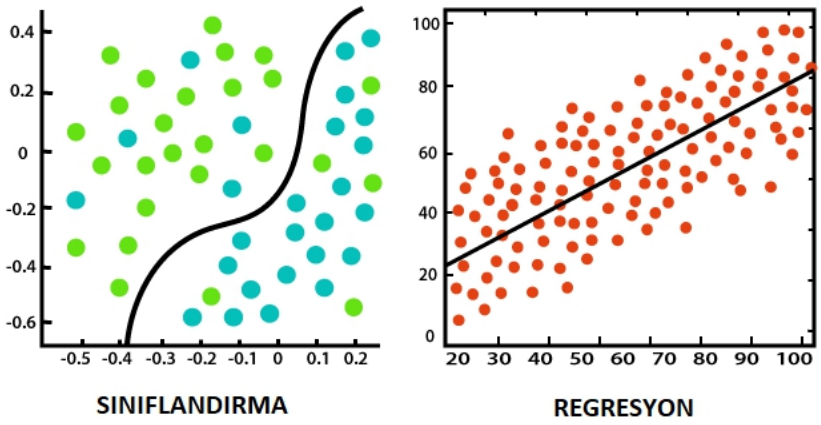
3.3.1. Denetimli Öğrenme (Supervised Learning)

Denetimli öğrenme, bir modelin etiketli veriler üzerinde eğitildiği bir makine öğrenmesi türüdür. Bu öğrenme türünde veri setinin ne olduğu ve bu verilerden istenilen çıktının ne olması gerektiği bilinir. Denetimli öğrenmede veri seti hem girdi

hem de çıktılardan oluşur ve amaç algoritmaya girdilerle birlikte çıktılar da verilerek algoritmanın bir model oluşturmasının sağlanmasıdır. Algoritmanın var olan verilerden öğrenip kendine bir fonksiyon oluşturması ve yeni gelen veriyi bu fonksiyona tabi tutarak tahminde bulunması istenir. Eğitim veri setindeki tüm özellik ve çıktılardan, örüntüler öğrenerek bir işlev ortaya çıkaran algoritma tahminlerde bulunmaya başlar ve kabul edilebilir bir başarı seviyesine ulaştığında öğrenme durur. Denetimli öğrenme sıklıkla kullanılan bir makine öğrenmesi türüdür ve problemlerin çıktısına bağlı olarak regresyon (regression) ve sınıflandırma (classification) problemleri şeklinde gruplandırılır [9].

Sınıflandırma, bir verinin sahip olduğu özelliklere göre hangi sınıfa ait olduğunun belirlenmesidir. Sınıflandırma algoritmalarında var olan verilerden örüntüler elde edilerek yeni gelecek verilerin hangi sınıfta yer alacağı tahmin edilir. Elde edilen veriler etiketlenebilir, kategorilere ayrılabilir bir veri seti şeklinde ise sınıflandırma algoritmaları kullanılabilir. En yaygın sınıflandırma algoritmaları; karar ağaçları, Naive Bayes sınıflandırma algoritması, k en yakın komşu (K-NN) ve destek vektör makinaları (SVN) örnek verilebilir.

Regresyon, bağımlı bir değişken ile bağımsız değişkenler arasındaki ilişkinin gücünü belirlemeye çalışır ve bu güce göre tahminler oluşturan istatistiksel bir ölçümdür. Regresyon algoritmaları ise sürekli bir değeri tahmin etmek için kullanılır. Girdi değerleri sürekli bir fonksiyon ile ifade edilmeye çalışılır. İkinci el araç fiyatı tahminleme için araç veri seti içerisinde aracın marka, model, km, yıl vb. özelliklerine (girdilere) göre bir fonksiyon elde edilip tahminleme sonucu fiyat çıktısı edinme bir regresyon örneğidir. Lineer regresyon, multiple lineer regresyon, polinominal lineer regresyon, karar ağacı regresyonu, Random Forest regresyonu regresyon algoritma örneklerindendir.



Şekil 3.2. Sınıflandırma ve Regresyon Denetimli Öğrenme

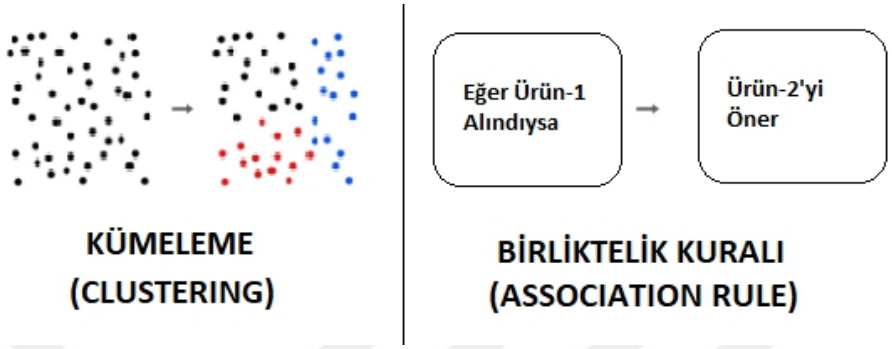
3.3.2. Denetimsiz Öğrenme (Unsupervised Learning)

Denetimsiz öğrenme sisteminde eğitim verisinde etiketli veri yoktur, sistem öğretmensiz çalışır. Etiketlenmemiş ham verileri, organize verilere dönüştürmek üzere çalışan bir makine öğrenim türüdür. Denetimsiz öğrenmede hedef veriler hakkında bolca bilgi toplayıp bu bilgilerin veriler üzerindeki dağılımını modellemektir. Denetimsiz öğrenme algoritmaları veride ne aradığımızı bilmediğimiz durumlarda yararlıdır. Hayatımızdaki birkaç denetimsiz öğrenme uygulamaları: Spotify, Netflix, YouTube gibi müzik/dizi/film/video öneri sistemleri, Facebook arkadaş/grup/sayfa öneri sistemi, E-Ticaret siteleri ürün öneri sistemi gibi. [9]

Denetimsiz öğrenme sitemleri kümeleme ve birliktelik şeklinde gruplandırılabilir:

Kümeleme (Clustering), veriler arasındaki doğal gruplamayı yapabilmek adına uygulanan algoritmalardır. Verilerin dağılımlarına göre bir gruplama yapılabilineceği gibi belirli bir sayıda küme belirterek de algoritmanın istenen sayıda gruba ayrılması sağlanabilir. K-kümeleme, hiyerarşik kümeleme ve olasılıksal kümeleme kümeleme çeşitleri arasındadır.

Birliktelik kuralı (Association Rule), A ürününü satın alan kişiler B ürününü de satın alma eğilimindedirler şeklindedir gibi veriler arasında ilişkilendirme kuralı öğrenmesidir. Özellikle marketlerde satışları arttırmak için işletilen bir denetimsiz öğrenme türüdür.



Şekil 3.3. Kümeleme ve Birliktelik Kuralı Denetimsiz Öğrenme

3.3.3. Yarı Denetimli Öğrenme (Semi-Supervised Learning)

Yarı denetimli öğrenme, denetimli ve denetimsiz öğrenme arasında bir makine öğrenimi türüdür. Eğitimde etiketlenmemiş verilerin kullanımını sağlayan denetlenen öğrenme şeklidir de denebilir. Bu öğrenme türünde genelde etiketlenmemiş veri sayısı, etiketli veriden çok daha fazladır. Etiketlenmemiş verilerin küçük miktarda etiketlenmiş veriler ile birlikte kullanıldığında öğrenme doğruluğunda önemli ölçüde iyileşme sağlayabildiği gözlemlenmiştir [11].

Bir öğrenme probleminde veri etiketlemek çoğu zaman uzman kişi veya fiziksel bir deneyi gerektirebilir. Tamamen etiketlenmiş bir veri setinin maliyeti bir kısmı etiketlenmiş veri setinden daha yüksek olabilecek durumlarda yarı denetimli öğrenme tercih edilebilir.

Kendi kendine eğitim (Self Training), sıklıkla kullanılan bir yarı denetimli öğrenmede tekniğidir. Bu teknikte kullanılan sınıflandırıcı öncelikle küçük bir etiketlenmiş veri setinde eğitilir. Daha sonra etiketlenmemiş bir veri seti üzerinde sınıflandırma denenerek en güvenli etiketlenmiş noktalar ön görülen etiketleriyle birlikte eğitim setine eklenir. Bu işlemler tekrar edilerek model eğitilmeye devam edilir.

3.3.4. Takviyeli Öğrenme (Reinforcement Learning)

Takviyeli öğrenme, bulunduğu ortamı algılayan ve bulunduğu ortamda kendi başına kararlar alabilen bir sistemin, amacını gerçekleştirmek için doğru kararlar alabilmeyi nasıl öğreneceğini gösterir [12]. Takviyeli öğrenmede denetimli öğrenmedeki gibi bir eğitmen bulunur ancak çok detay veremez. Detay vermek yerine öğrenen sistem bir karar verdiğinde bu karar doğru bir karar ise sistemi ödüllendirir yanlış bir kararda da cezalandırarak yönlendirir. Buradaki amaç öğrenen sistemin denediği olası durumların ulaşmak istediği hedef olup olmadığının kontrolü ve denenen doğru veya yanlış tüm durumların hatırlanmasının sağlanmasıdır. Takviyeli öğrenme Robotik, oyun programlama, fabrika otomasyonu gibi alanlarla sıklıkla kullanılır.

Q öğrenme en sık kullanılan takviyeli öğrenme algoritmalarından biridir. Rastgele bir ortam içinde sisteme, optimal politikayı nasıl öğrenebileceği gösterilir. Sistemin eğitim verisi bulunmadığı için optimal politikayı doğrudan öğrenmesi zordur. Sistemin kullanılabilir eğitim verisi sadece anlık ödüllerdir. Bu eğitim verilerini kullanarak sayısal bir değerlendirme fonksiyonunu öğrenmek ve sonrasında bu fonksiyon yardımıyla en uygun politikayı belirlemek çok daha kolaydır [13]. Q öğrenme algoritmasında amaç bir sonraki hareketleri inceleyip gerçekleşecek hareketlere göre kazanacağı ödülü görmek ve bu ödülü çoğaltacak şekilde hareket etmektir. Algoritmada bir anlamda, ajanın geleceğe dair plan kurması beklenmektedir. Bu algoritmanın labirent, arama vb. problemlerde sıklıkla uygulandığı görülmektedir.

4. DERİN ÖĞRENME (DEEP LEARNING)

Yapay zeka araştırmacıları her zaman daha zeki sistemler tasarlamak amacıyla çalışmaktadır. Bu amaçla insan düşünce yapısını ve karar verme yetisini modellemek en önemli hedeflerden birisi haline gelmiştir. McCulloch-Pitts [14] ilk defa insana özgü sinir sisteminden esinlenip beyin fonksiyonları işleyişini mantıksal olarak hesaplayan bir model ortaya koymuşlardır ve bu çalışma ile insan sinir sisteminin bir benzeri olan yapay sinir ağlarının da temelini oluşturmuştur.

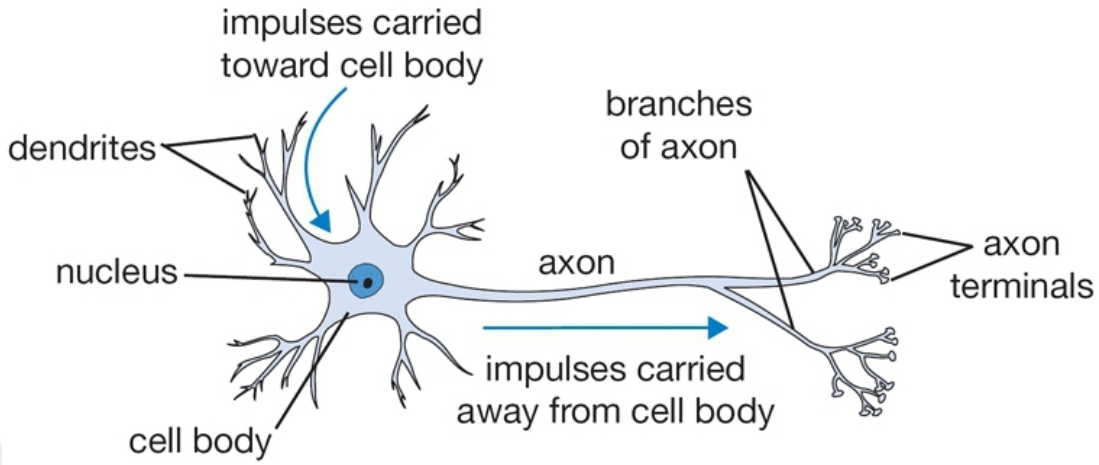
Derin öğrenme insan beynindeki nöronlardan ilham alarak oluşturulan yapay sinir ağlarının veriden öğrendiği bir makine öğrenmesi alt dalıdır. İlk defa 2012 yılında Büyük Ölçekli Görsel Tanıma Yarışması (ImageNet - Competition 2012) derin öğrenmenin temel mimarisi olarak bilinen evrişimli sinir ağı algoritmasıyla kazanılmış ve bu yarışma ile derin öğrenme hızlı bir yükseliş sağlamıştır. Derin öğrenmenin ilk çalışmaları çok eskilere dayanmasına rağmen gelişen teknoloji ile yeterli verinin edinilmesi ve bu verilerin güçlü bilgisayarlarla modellenebilmesi sayesinde günümüzde başarılı bir şekilde kullanılmaya başlanmıştır. [15]

4.1. Yapay Sinir Ağları (Artificial Neural Networks)

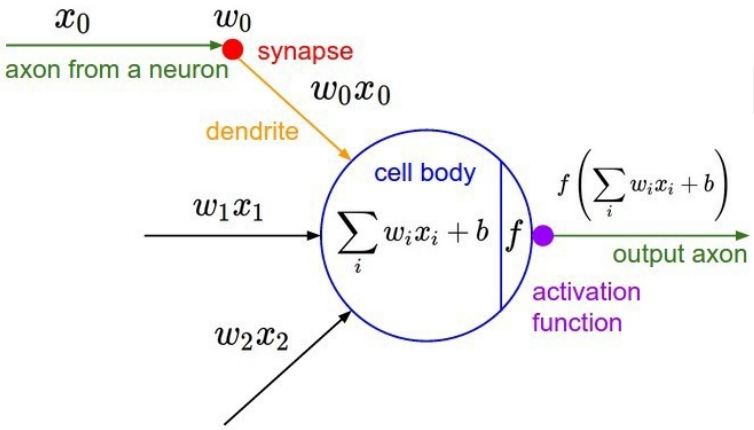
Yapay sinir ağı canlıların sinir sistemi ile öğrenme şeklinden, bilgi işleme tekniğinden esinlenilmiş bir yapay öğrenme teknolojisidir. Sinir sistemindeki sinir yapısı hiyerarşik şekilde oluşan bir elektriksel akım yardımıyla algılama işlemleri gerçekleştirir. Almaçlar aracılığı ile alınan elektriksel dürtü insanoğlunun doğumundan günümüzdeki hayatına değin duyduğu, gördüğü, hissettiği hatta düşündüğü her şeyi öğrenmeyi, hatırlayıp ezberlemeyi sağlar.

YSA modelinin esinlendiği sinir hücresi (nöron), sinir sisteminin temel fonksiyon birimidir. Bir nöronun yapısı çekirdek, akson, dentritler, miyelin tabaka ve sinapslardan oluşur. Çıkış darbelerinin üretildiği aktif elektriksel gövde olan Akson üzerinde tek yönlü iletim söz konusudur. Dentritler, diğer hücrelerden ulaşan uyarıları toplayan sistemin girişi, elektriksel anlamda pasif kollarıdır. Hücrelerin aksonları ile dentritler arasındaki bağlantı sinapslar ile sağlanır. Miyelin tabaka yayılma hızına etki eden yalıtım malzemesi iken akson boyunca uyarıların sürekli bir şekilde yeniden üretilmesini çekirdek sağlar. Akson üzerinde taşınan uyarı sinapslara

kimyasal taşıyıcılar yardımıyla iletilir ve belirli bir eşik gerilimin üstündeyken hücre uyarılır. Bu durumlara göre üretilen çıkış işareti sinirsel hesaplama olarak adlandırılır.



Şekil 4.1. Bir Nöronun Biyolojik Gösterimi

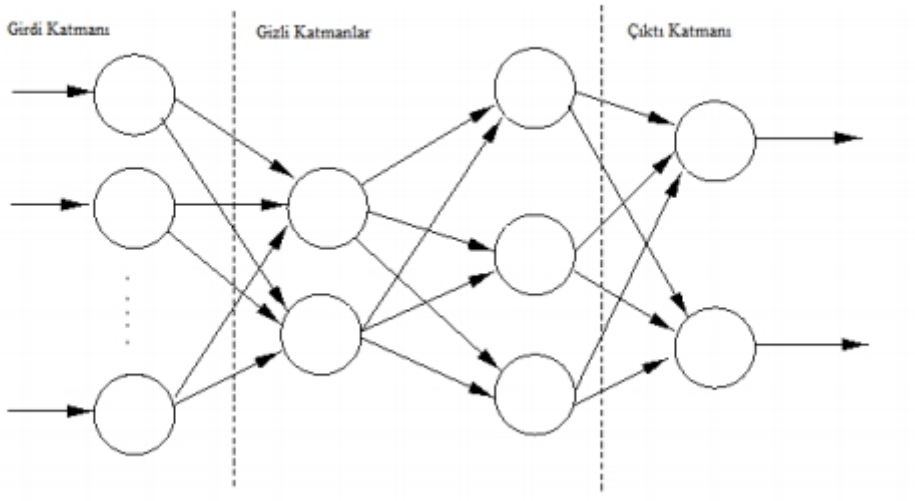


Şekil 4.2. Bir Nöronun Matematiksel Gösterimi

Perseptron, YSA'nın en küçük parçası olarak bilinir. Perseptron ilk defa 1957 yılında Frank Rosenblatt tarafından tanımlan $y = W * x + b$ şeklinde ifade edilen lineer bir fonksiyon olarak belirtilir. Y, X'in değerine bağlı bir değişken olup girdiye ait çıktıyı verir. X bağımsız bir değişken yani girdidir. W, ağırlık parametresi ve b, bias değeri olarak tanımlanır. YSA modelinde planlanan asıl amaç modelin en iyi sonucu vereceği ağırlık (W) ve bias (b) parametrelerinin hesabını yapmaktır. [16]

YSA yapısı yapay sinir hücrelerinin birbirlerine bağlanmasından oluşur. Bu bağlantılar başka hücrelere ya da kendi hücresine ağırlıklar üzerinden girdi şeklinde de olabilir. Hücreler arasındaki bağlantı şekilleri, aktivasyon fonksiyonları ya da öğrenme kurallarına göre YSA modelleri tasarlanmıştır. Basit bir YSA modelinde tek katman, birden çok girdi ve bir çıktı bulunmaktadır. Bu tip modeller 1 ya 0 şeklinde çıktı üretip genellikle nesneleri iki ayrı sınıfta kategorilendirmek için kullanılır. Tek katmanlı YSA modelleri sadece girdi ve çıktılardan oluştuğu için karmaşık işlevleri hesaplama yeteneklerinden yoksun kalmışlardır.

En az bir katmandan oluşan katmanlı YSA modeli birbiriyle bağlantılı sinirlerin yer aldığı girdi katmanı (input layer), gizli katman (hidden layer) ve çıktı katmanı (output layer) olmak üzere üç katmandan oluşmaktadır. Gizli katmanların arttırılmasıyla çoğalarak çok katmanlı yapılar oluşturulabilir. Girdi katmanı, dışarıdan gelen verilerin yapay sinir ağına aktarılmasını sağlayan bağımsız değişken olarak ifade edilen girdi değişkenlerinden oluşan ilk katmandır. Son katman bağımlı değişken olarak ifade edilen çıktı değişkenlerinden oluşan çıktı katmanıdır. Modeldeki diğer katmanlar, girdi ve çıktı katmanları arasında yer alarak gizli katman olarak adlandırılır. Gizli katmanlar girdi katmanlarından alınan sinyallerin çıktı katmanlarına iletilmesinden sorumludur ve dış ortamlar ile bağlantısı yoktur. [17]



Şekil 4.3. Katmanlı YSA Modeli

YSA mimarileri sinirler arasındaki bağlantılara, uyarıların akış yönüne göre ileri beslemeli(feed forward) ve geri beslemeli(back forward) olarak ikiye ayrılırlar. İleri beslemeli YSA modelinde bir katmandaki hücrelerin çıkışları bir sonraki katmana ağırlıklar üzerinden giriş olarak verilir. Giriş katmanı dış ortamlardan aldığı bilgileri hiçbir işleme sokmadan gizli katmanlardaki hücrelere iletir. Gelen bilgi gizli ve çıkış katmanlarında işlenerek ağ çıkışı belirlenir. Geri beslemeli YSA modelinde bir veya daha fazla hücrenin çıkışı, kendisine veya diğer hücrelere giriş olarak verilir. Geri besleme bir katmandaki hücreler arasında olduğu gibi katmanlar arasındaki hücreler arasında da gerçekleşebilir. Geri beslemeli YSA modelleri doğrusal olmayan dinamik bir davranış göstermektedir.[18]

4.2. Derin Öğrenmede Hiper-Parametreler

Derin öğrenme modeli tasarlanırken modelde kullanılan algoritmanın belirlenmesinde tasarımcının karar vermesi gereken bazı parametreler bulunmaktadır. Belirlenecek bu parametreler model eğitiminde başarı oranında önemli rol oynamaktadır. Genellile başlangıçta hangi hiper-parametreler kullanılacağı kesin bir şekilde belli olmamakla birlikte; probleme, veri setine vb. etkenlere bağlı olarak değişkenlik göstermektedir. Modelin yüksek başarım sağladığı birden fazla hiper parametre grubu olabilmektedir. Ancak uygun hiper parametre seçimi tasarımcının sezgilerine ve konu hakkındaki deneyimlerine, problemin farklı alanlarda uygulamalarına göre değişiklik gösterebilmektedir ki bu aşamada hiper parametre seçimi model tasarımında baş edilmesi gereken en önemli problemlerden biridir.

En sık kullanılan hiper parametreler; veri seti boyutu, batch boyutu, epoch sayısı, öğrenme hızı (learning rate), ağırlık başlangıç değerlerinin belirlenmesi, optimizasyon algoritması seçimi, aktivasyon fonksiyonu seçimi olarak bahsedilebilir.

Veri seti boyutu ve çeşitliliği, derin öğrenme modellerinin başarı oranında etkisinin yüksek olduğu sıklıkla gözlemlenmektedir. Veri seti ne kadar büyük olursa öğrenme de o oranda artacaktır ancak sadece büyüklük değil veri setindeki çeşitlilik de modelin başarımını arttıran bir özelliktir. Veri seti boyutu arttıkça başarım sürekli artacak diye bir durum yoktur. Bir süre sonra artış oranları azalacak ve model eğitilirken bu oranlar göz önüne alınarak sınır noktası belirlenmesi sağlanacaktır.

Batch boyutu, veri setindeki birden fazla girdinin parçalar halinde işlenmesi için belirlenmesi gereken her bir parça boyutu batch olarak adlandırılır. Veri setinde bulunan tüm verilerin özellikle çok büyük veri setlerinde aynı anda işlenmesi zaman ve bellek açısından oldukça maliyetli bir işlem olur. Derin öğrenme modelinin her iterasyonunda geri yayılım işlemi ile ağ üzerinde geriye dönük gradyan hesaplaması yapılır ve ağırlık değerleri bu şekilde güncellenmektedir. Bu hesaplama işleminde veri sayısı arttıkça hesaplama da o oranda zaman alacaktır. Hesaplama zamanı ve bellek maliyeti problemini çözmek için; veri seti küçük gruplara ayrılmakta ve öğrenme işlemi seçilen bu küçük gruplar üzerinde yapılmaktadır. Belirlenen batch boyutunun GPU belleğe sığması gerektiği için batch boyutu 2,4 ,8, 16, 32.. şeklinde ikinin katları oranında belirlenmelidir. Bu şekilde belirlenmezse başarım da ani düşüşler yaşanabilir.

Epoch, bir modelin eğitim kapsamında model ağırlıklarını güncellemek için tüm eğitim setini kullanarak bir iterasyon oluşturma terimidir. Model eğitilirken verinin tamamı değil batch olarak bahsettiğimiz parçalar halinde eğitilir. İlk eğitim sırasında modelin başarımı test edilip geri yayılım ile ağırlıklar güncellenir. Yeni parça ile tekrar eğitilip ağırlıklar yeniden güncellenir. Bu iterasyon her bir eğitim adımında tekrarlanarak model için en optimum ağırlık değerleri bu şekilde hesaplanır. Bu eğitim adımlarının her biri bir epoch olarak isimlendirilir. Eğitim sırasında en uygun ağırlık değerleri tek tek her bir adımda hesaplandığı için ilk epochlarda başarım oranı düşük olacak ancak epoch sayısının artmasıyla başarı oranı da artacaktır. Epoch sayısının büyüklüğü probleme bağlı olarak değişkenlik göstermektedir. Başarım belli bir epoch sayısından sonra çok az artmaya başlayacağı için belirlenen bu az artış noktalarında eğitim sonlandırılır.[19]

Öğrenme hızı, derin öğrenmede modelin öğrenme performansını ölçmek için kullanılan önemli bir parametredir. Derin öğrenmede parametrelerin güncellenmesi geri yayılım işlemi ile sağlanmaktadır. Bu güncelleme işlemi geriye doğru türev alarak farkın bulunması ve bulunan fark değerinin öğrenme oranı parametresiyle çarpılmasıyla çıkan sonucun ağırlık değerlerinden çıkarılarak yeni ağırlık değerinin hesaplanmasıyla yapılmaktadır. Bu işlem esnasında kullanılan öğrenme oranı parametresi sabit değer ya da artarak ilerleyen bir değer olarak da belirlenebilir. Genellikle öğrenme oranı için başlangıçta büyük değerler ancak sonlara doğru daha

küçük değerler seçilmektedir. İlk başlarda büyük öğrenme oranlarının seçilmesi eğitim adımlarını hızlandırırken hedefe doğru yaklaşıldıkça öğrenme oranının azaltılması sınıflandırma başarısını arttıracaktır. [20]

Modelin başlangıç ağırlık değerlerinin belirlenmesi, modelin öğrenmesine ve hızına etki etmektedir. Farklı ağırlık belirleme yöntemleri olmakla birlikte 0.5 gibi standart sapmaya ya da 0.5 ile 0.9 arasında uniform dağılıma sahip olacak şekilde tanımlamalar yapılabilir. Modelin ağırlık değerleri başlangıçta sıfır olarak başlatılırsa hesaplamada matris çarpımı bir toplam olması sebebiyle girdiler aynı şekilde çıktı olacağı için başlangıçta ağırlıklar sıfır olarak verilmemelidir.

Derin öğrenme modellerinde optimizasyon yöntemleri doğrusal olmayan problemlerin çözümünde en optimum değeri bulmak için önemlidir. Yaygın olarak kullanılan optimizasyon algoritmaları; stochastic gradient descent (SDG), adam, adamax, RMS-Prop, adagrad, olarak bahsedilebilir. Derin öğrenme modellerinde genellikle varsayılan olarak kullanılan optimizasyon algoritması SGD olmasına rağmen diğer yöntemlere göre daha yavaş çalıştığı söylenebilir. Aynı zamanda özellikle resim tanıma problemleri olmak üzere bazı problemlerde çok kötü sonuçlar verebilmektedir. Adaptive algoritmalar öğrenme hızını kendisi öğrenmektedir. Birkaç optimizasyon algoritmasından bahsedersek; AdaGrad seyrek parametreler için büyük güncellemeler yaparken sık parametreler için daha küçük güncellemeler yapar ve bu nedenle NLP ve resim tanıma gibi seyrek veriler için daha uygun olduğu söylenebilir. Adagrad'da her parametreye ait öğrenme hızı bulunur ve güncelleme işleminde öğrenme oranı süresince büyümeye devam etmesiyle öğrenme katsayısı aşırı küçülmektedir. Rmsprop ise Adagrad'daki tam olarak bu aşırı küçülme sorununa çözüm olarak geliştirilmiştir. Adagrad algoritmasındaki geçmiş bütün eğimlerin karelerinden elde edilen değerlerin tamamını kullanmak yerine, değer miktarını belli bir çerçeve boyutu ile kısıtlamıştır [21]. Diğer bir optimizasyon algoritması olan Adam, parametrelerin her birinin öğrenme oranlarının yanı sıra momentum değişikliklerini de ön bellekte saklar. [19]

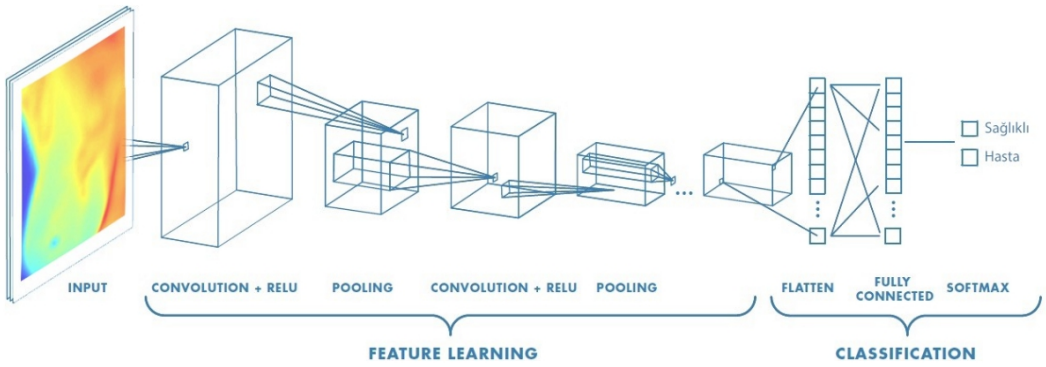
Aktivasyon fonksiyonları, çok katmanlı yapay sinir ağlarında doğrusal işlemlerin doğrusal olmayan işlemlere dönüşümü için kullanılan fonksiyonlardır. Derin öğrenmede gizli katmanlarda gerçekleşen matris işlemleri sonucunda doğrusal

bir yapı oluşmaktadır. Derin öğrenme yöntemleri doğrusal olmayan problemlerin çözümünde daha etkili olduğu için aktivasyon fonksiyonları ile model doğrusal olmayan bir yapıya dönüştürülür [20]. Kullanılan aktivasyon fonksiyonlarının bazıları Sigmoid, Tanjant, Relu ve Leaky Relu olarak örneklendirilebilir. Sigmoid sıklıkla kullanılan aktivasyon fonksiyonu olmakla birlikte parametreler daha hızlı bir şekilde öğrenildiği için en kullanışlı aktivasyon fonksiyonu Relu olarak gösterilebilir. Eğer problemimizde negatif değerler önemli ise Leaky Relu, Relu'nun kaçırdığı negatif değerleri yakaladığı için daha kullanışlı olacaktır [19]. Yine aktivasyon fonksiyonu seçimi de probleme bağlı olarak değişkenlik göstermektedir.

4.3. Evrişimli Sinir Ağları (Convolutional Neural Networks - CNN)

Özellik çıkarımı için filtrelerin girdi üzerinde dolaşması işlemine konvolüsyon (evrişim) denmektedir. CNN, veriden öğrenen doğrusal olmayan dönüşüm fonksiyonları yığınıdır. Başlangıçta 1980'lerde rakam tanıma için kullanılmış olsa da şimdilerde büyük ölçekli verilerde sınıflandırma problemlerinde ve bilgisayarlı görüntü tanıma işlemlerinde kullanılmaktadır.

Klasik YSA'dan farklı olarak CNN en az bir katmanda matris çarpımı yerine konvolüsyon işlemi kullanan katmanlar içermektedir. CNN temelde konvolüsyon, aktivasyon fonksiyonu (relu) ve havuzlama aşamalarını içermekte ve büyük boyutlu veriden düşük boyutlu özniteliklerin çıkarılmasını sağlamaktadır. [3]



Şekil 4.4. CNN Mimarisi

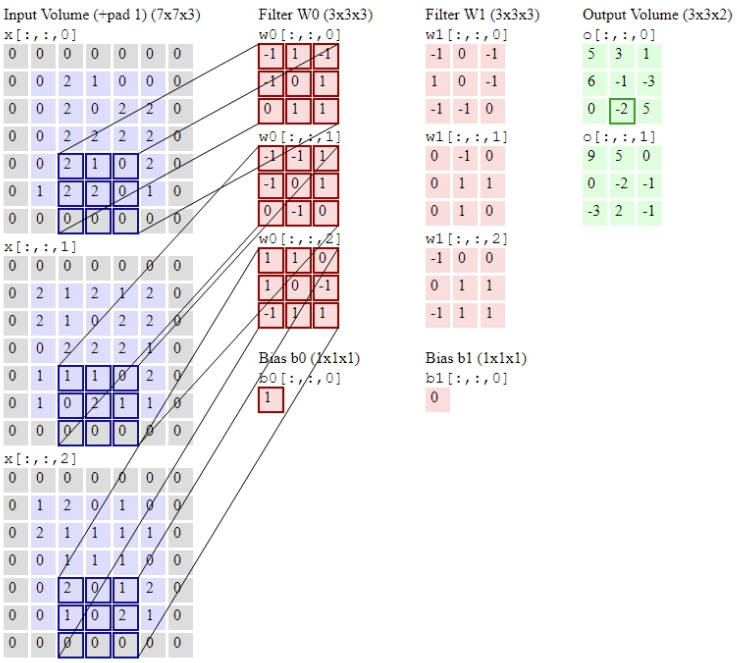
İlk katmanda girdiler üzerinde konvolüsyon işlemi ile özellik çıkarımı gerçekleştirilir. Hesaplama maliyetini düşürmek ve girdiden öğrenilen özelliklerin bir

sonraki katmana aktarmak için boyut düşürme fonksiyonları kullanılır. En sonda elde edilen bu özellikler tek boyutlu dizi haline getirilip tam bağlantılı katmanlara girdi olarak verilir ve sınıflandırma işlemi gerçekleştirilir. [20]

4.3.1. Konvolüsyon Katmanı (Convolution Layer)

CNN modelinde, girdi matrisi üzerinde belirli filtreler uygulanarak öznitelik çıkarma işlemi konvolüsyon olarak adlandırılmaktadır. Bu katman bir CNN modelinin ana yapı taşıdır. Girdi üzerine bazı filtreler uygulayarak yüksek ve düşük seviyeli özelliklerini çıkarmak için kullanılır.

CNN modelinde her bir katman üzerinde işlem yapan filtreler bulunur ve bu filtrelerin boyutunun öğrenme üzerinde etkisi oldukça fazladır. Her bir filtre farklı bir özelliği ortaya çıkarmaya çalışır. Genellikle 3x3,5x5 şeklinde filtreler kullanılır. Büyük boyutlu filtrelerde konvolüsyon uygulandıktan sonra oluşacak resim küçük olacaktır bu da bilgi kaybına neden olur. Bu nedenle genellikle 3x3 gibi küçük boyutlu filtreler kullanılır. Girdi üzerinde uygulanan filtreler tüm pikselleri dolaşarak girdi ile filtre noktasal çarpılır.



Şekil 4.5. Konvolüsyon Gösterimi

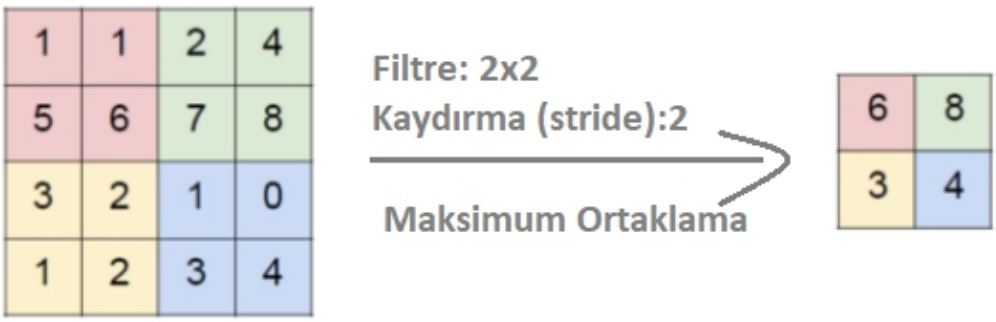
Kenar bilgileri görüntü girdisinin yüksek frekanslı bölgelerini temsil eden en çok ihtiyaç duyulan özniteliklerdir. Bu öznitelikler girdiye uygulanan yatay ve de

dikey filtrelerin çıkışıyla elde edilir. Genellikle kenarlar, CNN modellerinin ilk katmanlarında hesaplanmaktadır. Bu hesaplamalar sırasında giriş boyutu ile çıkış boyutu arasında farklılıklar meydana gelir. Konvolüsyon işlemi sonrası bu farklılıkların oluşmaması istenirse giriş matrisine piksel ekleme yöntemi uygulanabilir. [22]

Konvolüsyon işlemi sırasında yapılan işlemler doğrusal işlemler olduğu için Relu aktivasyon fonksiyonu ile model doğrusal olmayan yapıya dönüştürülür. Bu adıma aktifleştirme de denilmektedir.

4.3.2. Ortaklama - Havuzlama Katmanı (Pooling Layer)

CNN modellerinde havuzlama katmanı ardışık konvolüsyon katmanları arasına eklenir. Ağdaki parametre ve hesaplama karmaşıklığını azaltmak için kullanılır ve bu sayede ağdaki uyumsuzluk kontrol edilmiş olur. Boyuttaki azalma bilgi kaybına neden olacaktır ancak sinir ağının eğitimi hızlandırdığı için havuzlama CNN modellerinde yaygın olarak kullanılan bir adımdır. Bu adımda öğrenilecek bir parametre yoktur. Havuzlama katmanında ağırlıklı olarak maksimum (max pooling) ve ortalama (average pooling) yöntemleri kullanılmaktadır. Maksimum havuzlamada seçilen havuzlama boyutu içindeki maksimum değer çıkışa aktarılır. Ortalamada ise seçilen havuzlama boyutundaki değerlerin ortalaması çıkışa aktarılır.



Şekil 4.6. Maksimum Ortaklama

4.3.3. Düzleştirme Katmanı (Flattening Layer)

Genellikle, sinir ağları girdileri tek boyutlu bir diziden alır. Konvolüsyon ve havuzlama işlemlerinden elde edilen matris tam bağlantılı katmanda kullanılabilmesi için düzleştirilmesi gerekir. Bu katmanda elde edilen matris tek boyutlu diziye çevrilir.

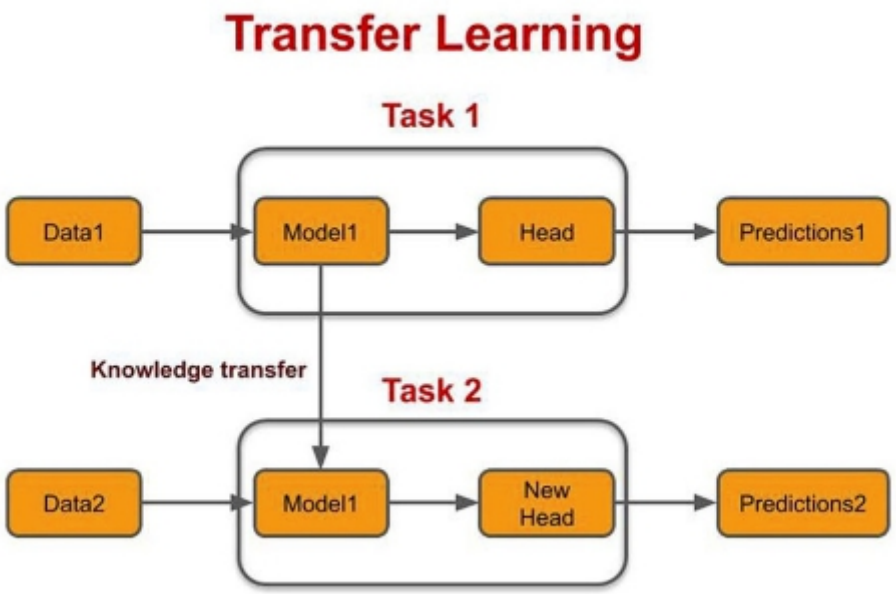
4.3.4. Tam Bağlantılı Katman (Fully Connected Layer)

Tam bağlantılı katman, CNN mimarisinde son katmanı oluşturur. Bu katmana gelene kadar giriş verisinin özellik seçimi, boyut azaltma, normalizasyon işlemleri gerçekleştirilerek elde edilen çıktıya göre hata hesaplanır. Ağırlıklar hesaplanan bu hata değerine göre tekrar güncellenir. Bu katmanda genellikle softmax aktivasyon fonksiyonu kullanılır ve sınıflandırma bu katmanda tamamlanır.

5. TRANSFER ÖĞRENME (TRANSFER LEARNING)

5.1. Transfer Öğrenmesi Nedir?

Transfer öğrenmesi, bir problem için benzer bir görevde büyük veri seti ile daha önce eğitilmiş bir model kullanan derin öğrenme yöntemidir. Bir modeli sıfırdan eğitmek yerine, transfer öğrenmeyi kullanarak modele ince ayar (fine tuning) yapmak çok daha hızlı ve kolay bir öğrenme yöntemidir [2]. Örneğin insan hayvan sınıflandırmasını yapan bir model eğitilirken kazanılan bilgi kadın erkek sınıflandırmasında transfer edilerek kullanılabilir.



Şekil 5.1. Transfer Öğrenmesi Mimarisi

Farklı görüntü işleme problemlerinde verilerin elde edilmesi ve modellerin tasarlanması oldukça zor olabilmektedir.Probleme uygun transfer öğrenmesi yöntemi daha az veri ile daha yüksek performans elde edilmesini mümkün kılmakla birlikte model tasarımcısına zamandan kazanç sağlamaktadır [1]. Transfer öğrenme ile yapay öğrenme modelinin öğrendiklerinin tamamı ya da bir kısmı transfer edilerek benzer bir problemin çözümü daha hızlı ve kolay olabilmektedir.

Öğrenme aktarımı sürecinde ne aktarılacak, ne zaman ve nasıl aktarılmalı soruları önemlidir. Kaynak veriden hedef görevin başarımını arttırmak için hangi bilginin aktarılması gerektiğinin belirlenmesi ilk ve en önemli adımdır. Burada bazı

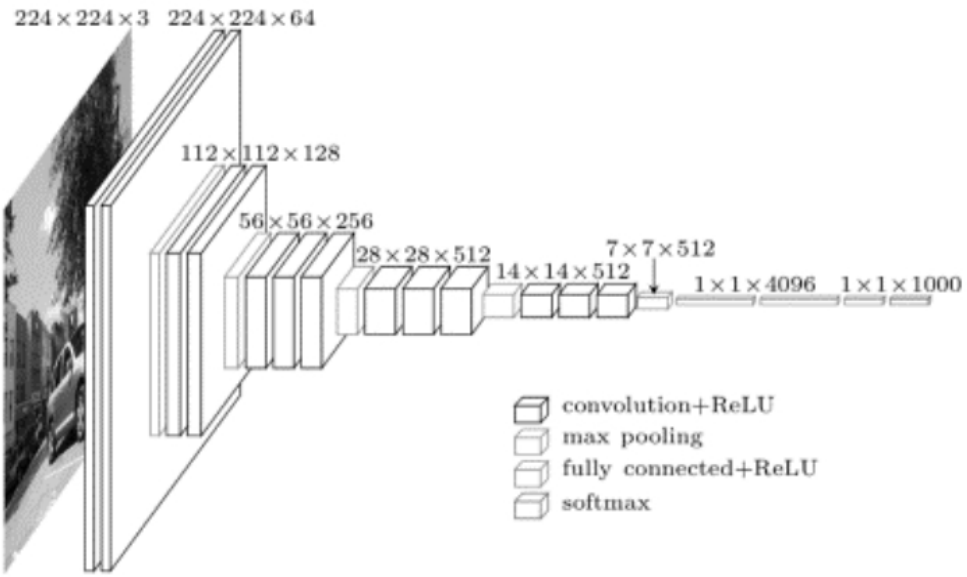
bilgilerin kullanılan veriye özgü olabileceği dikkate alınarak kullanılan alan ne olursa olsun ortak bilginin aktarılacak bilgi olarak seçilmesi gerekmektedir. İkinci adımda ise verinin hangi aşamada aktarılması gerektiğinin belirlenmesidir. Kaynak veri ile hedef veri birbirinden çok farklı ise aktarımın temel adımlar işlendikten sonra ilk katmanlarda aktarım yapılması önemlidir. Aksi takdirde hedef veri üzerinde başarım düşebilir hatta negatif transfer söz konusu olabilir. Son olarak bilginin aktarımı için hangi yöntemler kullanılmalı kararı gelir ki burada var olan standart yöntemler kullanılabilir ya a bu yöntemlerin değiştirilmesi yaklaşımı uygulanabilir.

5.2. Transfer Öğrenmesi Modelleri

Transfer öğrenmesi modellerinde birden fazla örnek modelle karşılaşılabilmektedir. Tez çalışmamızda VGG16, ResNet50 ve Xception modelleri kullanılmış ve bu modeller hakkında bilgilendirme alt başlıklarda yapılmıştır.

5.2.1. VGG16 Modeli

VGG16 modeli Oxford üniversitesi tarafından ILSVRC2014 yarışmasında 13 konvolüsyon 3 tam bağlı katmanı ile daha iyi sonuçlar elde etmek için oluşturulan bir ağdır. Mimaride toplamda 41 katman bulunurken bu katmanlar; Maxpool, Fullconnected, Relu, Dropout ve Softmax katmanlarından oluşmaktadır. Giriş katmanında 224x224x3 boyutunda[25] görüntü yer alır ve tam bağlantı katmanında 7x7x512=4096 nöronlu bir öznitelik vektörüne dönüştürülür. İki tam bağlantı katmanı çıkışında 1000 sınıflı softmax ile başarım oranı hesaplanır. Modelde girişten çıkışa doğru matrislerin yükseklik ve genişlik boyutları azalırken derinlik artmaktadır. Her konvolüsyon katmanı çıkışında farklı ağırlıklara sahip filtreler hesaplanır [22]. VGG16 modeli ImageNet veri setinde %89 doğruluk oranı elde etmiş bir derin öğrenme algoritmasıdır [24].

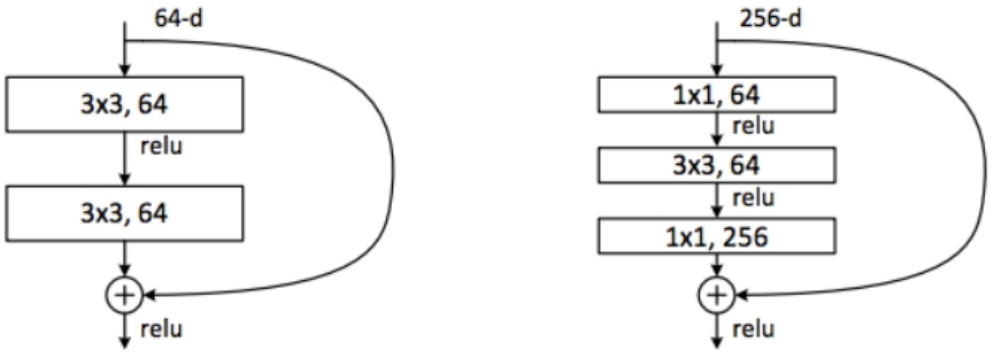


Şekil 5.2. VGG16 Modeli Mimari Gösterimi

5.2.2. ResNet50 Modeli

ResNet50 (artık değerli nöral ağlar - residual neural network), VGG16 gibi ImageNet veri seti ile eğitilmiş ancak geleneksel ağ mimarisinden farklı olarak mikro mimari yapısı ile diğer modellerden ayrışmaktadır. ResNet50 mimarisinde bazı katmanlardaki değişim görmezden gelinerek alt katmanlara geçiş yapılmasına imkan verilmesi başarım oranlarını arttırmıştır. Resnet50 mimarisinde 177 katmanlı bir ağ yapısı ve katmanlar arasındaki bağlantılar hakkında bilgi mevcuttur. [25]

ResNET kendi makalesinde, CNN ağlarında yakınsama başladığında bozulma problemlerinin ortaya çıkmasından bahsetmiş ve ağ derinliği arttıkça verimlilik doygunluğa ulaşır ancak daha sonra hızlı bir düşüş gösterme eğilimindedir şeklinde açıkladığı CNN ağlarındaki performans düşümü problemine çözüm üretmeyi hedeflemiştir. ResNET bu problemi çözmek için katmanlar arasına kısa yollar ekler ve bu kısa yollar ağ derinleştikçe oluşan bozulmaları önler.

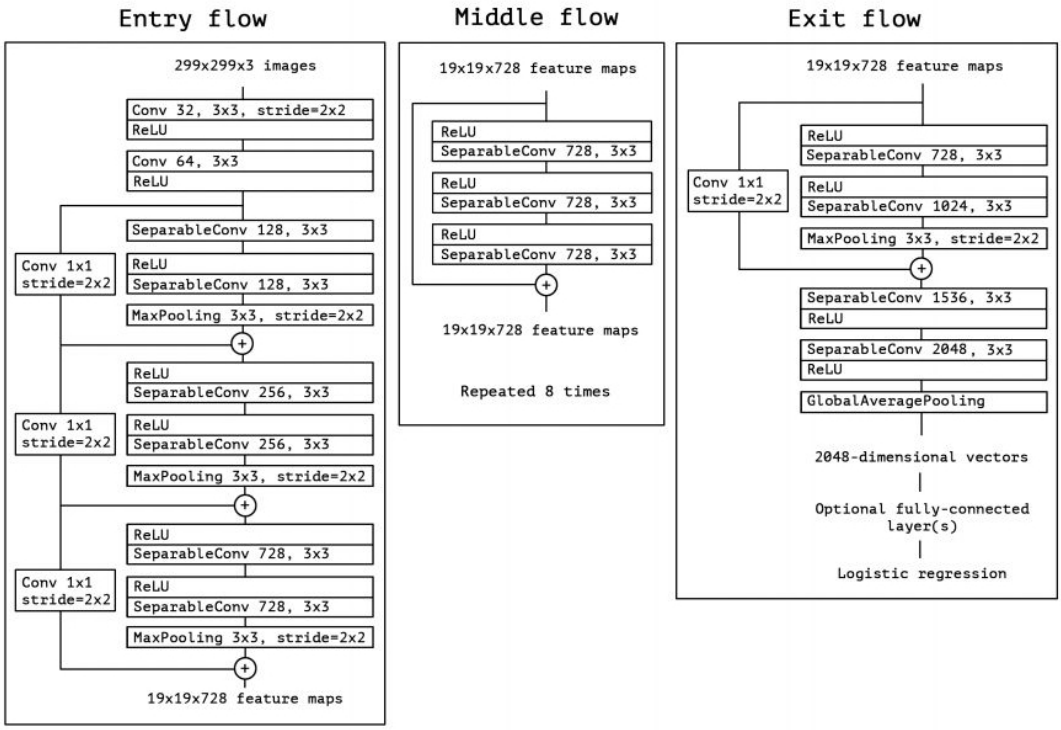


Şekil 5.3. ResNet50 Mimarisinde Kullanılan Bağlantı Örneği

5.2.3. Xception Modeli

Xception derinlemesine ayrılabilir konvolüsyonlar içeren Google araştırmacıları tarafından geliştirilmiş bir CNN mimarisidir. Google, GoogLeNet ismini verdiği mimari ile 2015 yılında yayınladığı "Going Deeper with Convolutions" makalesinde konvolüsyonlar ile daha derine gitme fikrinden bahsetmiştir. Bu yayın CNN modelleri içerisinde genişlik terimini kullanan ilk yayın olarak gözlemlenmektedir. Xception modeli diğer modellere göre daha derin ve daha geniş bir model olarak kurgulanmıştır. Google CNN'deki Inception modüllerinin bir yorumunu düzenli evrişim ile derinlemesine ayrılabilir evrişim işlemi arasında bir ara adım olarak sunmuştur. Bununla beraber derinlemesine ayrılabilir bir evrişim, en fazla sayıda kuleye sahip bir Inception modülü olarak anlaşılabilir.

Xception mimarisinde veriler önce giriş akışından, ardından sekiz kez tekrarlanan orta akıştan ve son olarak da çıkış akışından geçer. Tüm evrişim ve ayrılabilir evrişim katmanları toplu normalizasyon adımından geçer. Xception mimarisi, klasik sınıflandırma zorluklarının çoğunda VGG-16, ResNet ve Inception V3'ten fazla performans göstermiştir. [26]



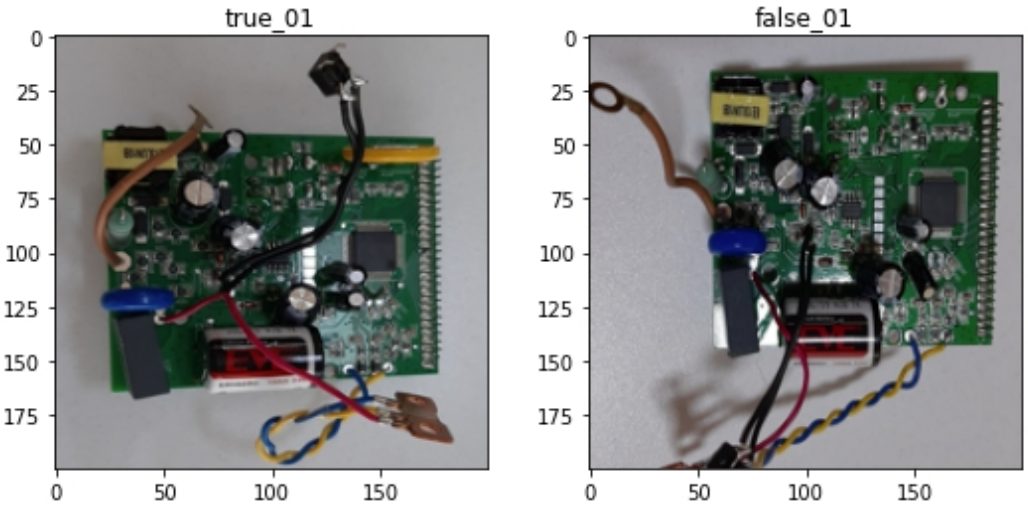
Şekil 5.4. Xception Mimarisi

6. MATERYAL VE YÖNTEMLER

6.1. Veri Setinin Hazırlanması

6.1.1. Veri Setinin Toplanması

Akıllı sayaç üretimi yapan bir firmada üretim, assembly kart diziliminin bir cihaz üzerinde gerçekleştirilmesinden sonra kart üzerindeki pil, transistör gibi parçaların bant üzerinde insan eliyle eklenmesiyle gerçekleşmektedir. Hatalı ürünler bu aşamada bant üzerinde tespit edilip kalite departmanında çalışanlar tarafından ayrılmasıyla elde edilmiştir. Ek olarak çeşitlilik yaratabilmek için doğru ürün üzerinde oluşabilecek hata durumları belirlenip manuel müdahale ile hatalı ürünler üretilmiştir. Doğru ve hatalı ürünlerin bant üzerinde aynı zamanda resimleri çekilerek tek bir modele ait 13 hatalı ve 13 doğru kart dizilimi fotoğraflanarak veri setinin ilk ürünleri elde edilmiştir.

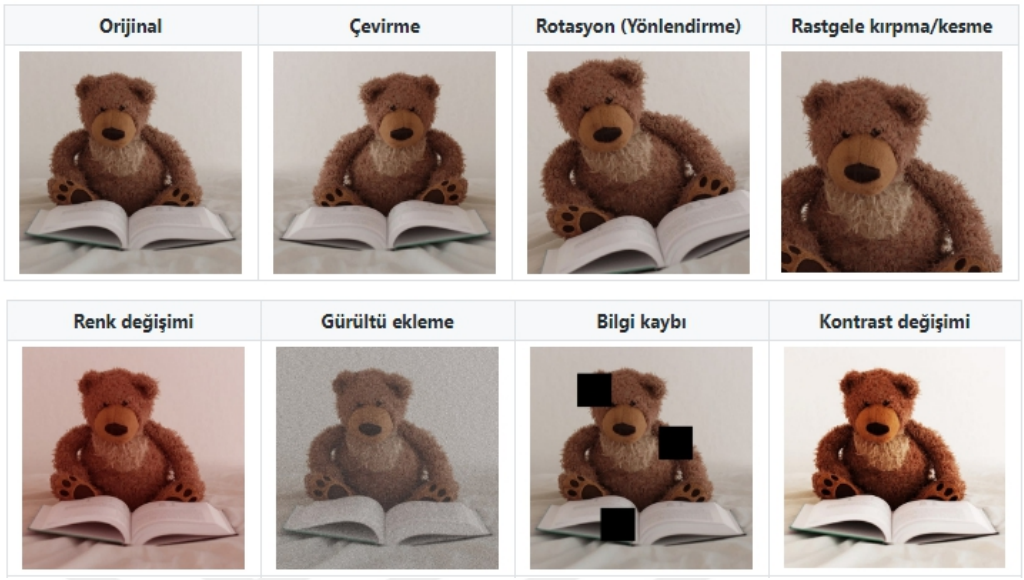


Şekil 6.1. Veri Setinden Doğru ve Hatalı Örnek Kart Gösterimi

6.1.2. Veri Arttırma (Data Augmentation)

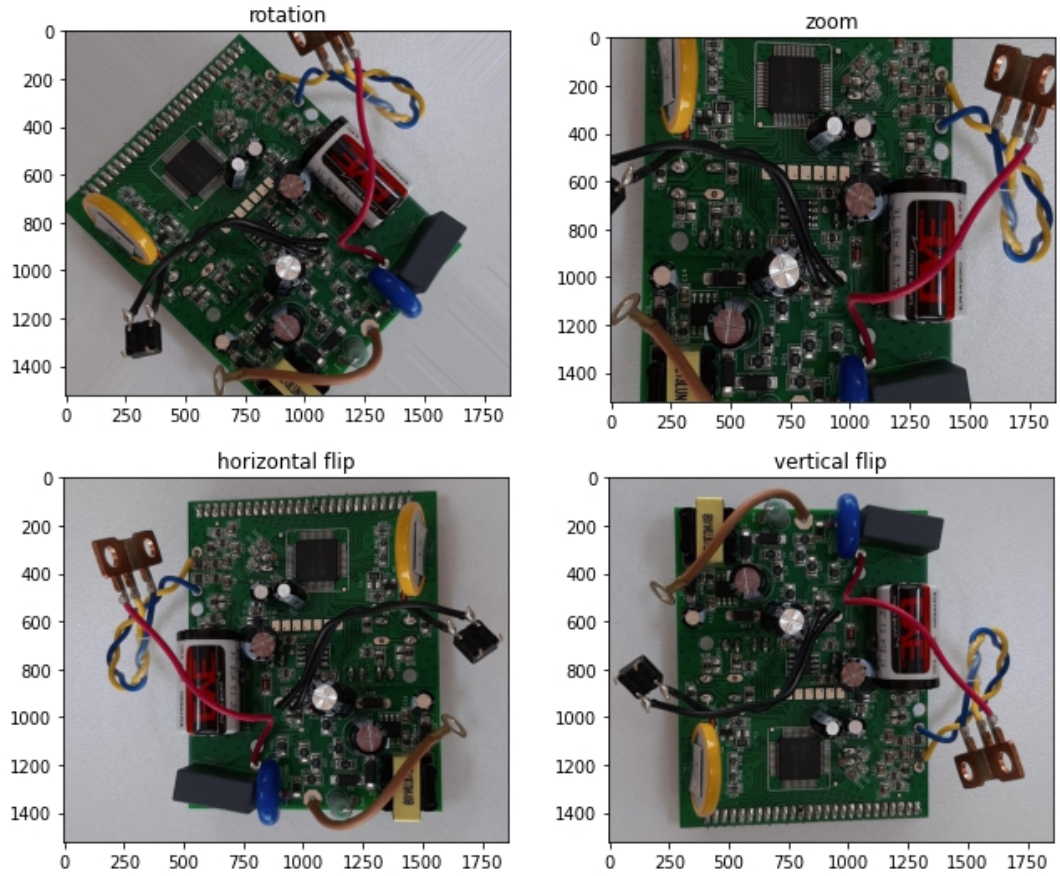
Derin öğrenme modellerini optimum şekilde eğitmek için genellikle çok fazla veriye ihtiyaç vardır. Elde edilen veri yeterli miktarda olmadığında veri arttırma teknikleri kullanılarak daha fazla veri üretmek model eğitiminde genellikle yararlı olmaktadır. Büyüklüğü daha az olan veri kümelerinin başarım oranlarını arttırmak amacıyla veri üzerinde farklı bozulma seçenekleri uygulanarak veri çoğaltılır ve böylece modelin daha çeşitli koşulları da öğrenebilmesi sağlanır. Data Augmentation ile girdi görüntüsüne uygulanabilen temel işlemler görseli çevirme, yönlendirme,

kırpma, renk değiştirme görüntü ekleme,bilgi kaybı ve kontrast değişimi gibi örneklendirilebilir.



Şekil 6.2. Örnek Data Augmentation Yöntemleri

Tez çalışmamızda elde edilen veriler yetersiz olduğu için veri arttırma yöntemi ile eğitim ve validasyon veri setleri oluşturulmuştur. Ham veriler üzerinde rotasyon, rastgele kırpma, çevirme gibi veri arttırma işlemleri uygulanarak hatalı ve doğru verilerden oluşan 910 adet eğitim veri seti ve 468 adet validasyon veri seti oluşturulmuştur.



Şekil 6.3. Veri Arttırma Yöntemi ile Arttırılmış Veri Setinden Örnekler

6.2. CNN Algoritması ile Çalışma

Bir modele transfer learning uygulamadan önce modelin standart CNN modeli ile başarım oranlarını görmek aradaki farkı çok daha iyi özetleyecektir. Bu nedenle tez çalışmamızda elde ettiğimiz veri seti öncelikle CNN modelinde eğitilmiştir. 910 adet eşit sayıda doğru ve hatalı ürün içeren eğitim veri setimiz 10 epochta eğitilmiştir. Optimizer olarak Adam optimizasyon yöntemi ve kayıp fonksiyonu olarak binary cross entropy kullanılmıştır.

```
Epoch 1/10
14/14 [==============================] - 129s 9s/step - loss: 1.0185 - accuracy: 0.5384 - val_loss: 0.8275 - val_accuracy: 0.5214
Epoch 2/10
14/14 [==============================] - 85s 6s/step - loss: 0.8697 - accuracy: 0.4871 - val_loss: 0.6901 - val_accuracy: 0.5429
Epoch 3/10
14/14 [==============================] - 65s 5s/step - loss: 0.6944 - accuracy: 0.5505 - val_loss: 0.6911 - val_accuracy: 0.5357
Epoch 4/10
14/14 [==============================] - 51s 4s/step - loss: 0.6957 - accuracy: 0.4674 - val_loss: 0.6932 - val_accuracy: 0.4929
Epoch 5/10
14/14 [==============================] - 41s 3s/step - loss: 0.6933 - accuracy: 0.5389 - val_loss: 0.6935 - val_accuracy: 0.4786
Epoch 6/10
14/14 [==============================] - 32s 2s/step - loss: 0.6875 - accuracy: 0.5544 - val_loss: 0.6731 - val_accuracy: 0.6643
Epoch 7/10
14/14 [==============================] - 29s 2s/step - loss: 0.6591 - accuracy: 0.6157 - val_loss: 0.6153 - val_accuracy: 0.7000
Epoch 8/10
14/14 [==============================] - 27s 2s/step - loss: 0.6000 - accuracy: 0.6904 - val_loss: 0.5581 - val_accuracy: 0.7429
Epoch 9/10
14/14 [==============================] - 22s 2s/step - loss: 0.5491 - accuracy: 0.7784 - val_loss: 0.4613 - val_accuracy: 0.8071
Epoch 10/10
14/14 [==============================] - 20s 1s/step - loss: 0.3927 - accuracy: 0.8359 - val_loss: 0.3502 - val_accuracy: 0.8857
```

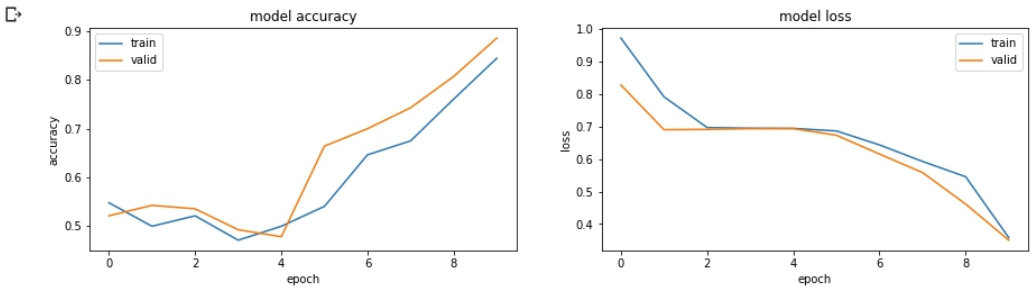
Şekil 6.4. CNN Modeli ile Eğitim

```
## Checking model's accuracy
print('Training accuracy: {:.3f}'.format(history.history['accuracy'][-1]))
print('Validation accuracy: {:.3f}'.format(history.history['val_accuracy'][-1]))
```

```
Training accuracy: 0.844
Validation accuracy: 0.886
```

Şekil 6.5. CNN Modeli ACC

Eğitim sonrası elde edilen eğitim veri seti ile başarım 0.844, validasyon veri seti ile başarım 0.886 olarak gözlemlenmiştir. Ek olarak aşağıdaki grafikte modelin acc ve loss değerlerinin epoch ile değişimi gösterilmiştir.



Şekil 6.6. CNN Modeli ACC ve LOSS Grafiği

Standart CNN modeli ile eğitim 910 adet eğitim ve 468 adet validasyon veri seti ile 11 dakikalık bir sürede gerçekleşmiştir.

6.3. Transfer Learning ile Çalışma

Tez çalışmamızda CNN modeli ile elde ettiğimiz %80'lerdeki başarı oranı elde edilmesi transfer öğrenme yaklaşımını deneyimlemeye teşvik etmiştir. Başarım

oranını arttırmak amacıyla birden fazla transfer öğrenme modeli ile çalışmanın başarı oranları gözlemlenmek istenmiş ve VGG16, ResNet50 ve Xception transfer öğrenme modelleri aynı veri setine uygulanarak sonuçları aşağıdaki başlıklarda paylaşılmıştır.

6.3.1. VGG16 Modeli ile Çalışma

VGG16 Modelinin ImageNet ile eğitilmiş yüksek başarı oranına sahip bir model olduğundan transfer öğrenmesi başlığı altında bahsedilmişti. Bu modelin elimizdeki veri seti üzerindeki başarı oranını incelemek için veri seti transfer öğrenmesi ile 10 epochta eğitilmiştir. RMSProp optimizerı ile binary cross entropy kayıp fonksiyonu kullanılmıştır. Bu modelde ince ayar (fine tuning) uygulanmış ve fine tuning öncesi sonrası başarı oranları karşılaştırılmıştır.

```
Found 910 images belonging to 2 classes.
Found 468 images belonging to 2 classes.
Epoch 1/10
10/10 - 21s - loss: 0.4454 - acc: 0.8600 - val_loss: 0.4184 - val_acc: 0.8850
Epoch 2/10
10/10 - 20s - loss: 0.4253 - acc: 0.8900 - val_loss: 0.3570 - val_acc: 0.9600
Epoch 3/10
10/10 - 20s - loss: 0.3677 - acc: 0.9350 - val_loss: 0.3328 - val_acc: 0.9600
Epoch 4/10
10/10 - 19s - loss: 0.3952 - acc: 0.8632 - val_loss: 0.3185 - val_acc: 0.9700
Epoch 5/10
10/10 - 19s - loss: 0.3537 - acc: 0.9050 - val_loss: 0.3083 - val_acc: 0.9650
Epoch 6/10
10/10 - 20s - loss: 0.3164 - acc: 0.9450 - val_loss: 0.2884 - val_acc: 0.9850
Epoch 7/10
10/10 - 20s - loss: 0.3191 - acc: 0.9300 - val_loss: 0.2810 - val_acc: 0.9950
Epoch 8/10
10/10 - 20s - loss: 0.2953 - acc: 0.9450 - val_loss: 0.2554 - val_acc: 0.9950
Epoch 9/10
10/10 - 19s - loss: 0.2661 - acc: 0.9842 - val_loss: 0.2375 - val_acc: 0.9900
Epoch 10/10
10/10 - 20s - loss: 0.2787 - acc: 0.9500 - val_loss: 0.2226 - val_acc: 0.9800
```

Şekil 6.7. VGG16 Modeli ile Eğitim - Fine Tuning Öncesi

```
Epoch 1/10
10/10 - 22s - loss: 0.2228 - acc: 0.9250 - val_loss: 0.1329 - val_acc: 0.9800
Epoch 2/10
10/10 - 20s - loss: 0.1646 - acc: 0.9600 - val_loss: 0.1340 - val_acc: 0.9900
Epoch 3/10
10/10 - 20s - loss: 0.1164 - acc: 0.9650 - val_loss: 0.0751 - val_acc: 0.9850
Epoch 4/10
10/10 - 20s - loss: 0.0930 - acc: 0.9750 - val_loss: 0.0575 - val_acc: 0.9900
Epoch 5/10
10/10 - 19s - loss: 0.0799 - acc: 0.9842 - val_loss: 0.0461 - val_acc: 0.9950
Epoch 6/10
10/10 - 19s - loss: 0.0513 - acc: 0.9895 - val_loss: 0.0358 - val_acc: 0.9950
Epoch 7/10
10/10 - 19s - loss: 0.0701 - acc: 0.9850 - val_loss: 0.0344 - val_acc: 0.9950
Epoch 8/10
10/10 - 19s - loss: 0.0515 - acc: 0.9900 - val_loss: 0.0229 - val_acc: 0.9950
Epoch 9/10
10/10 - 19s - loss: 0.0302 - acc: 1.0000 - val_loss: 0.0210 - val_acc: 1.0000
Epoch 10/10
10/10 - 19s - loss: 0.0317 - acc: 0.9950 - val_loss: 0.0235 - val_acc: 0.9950
```

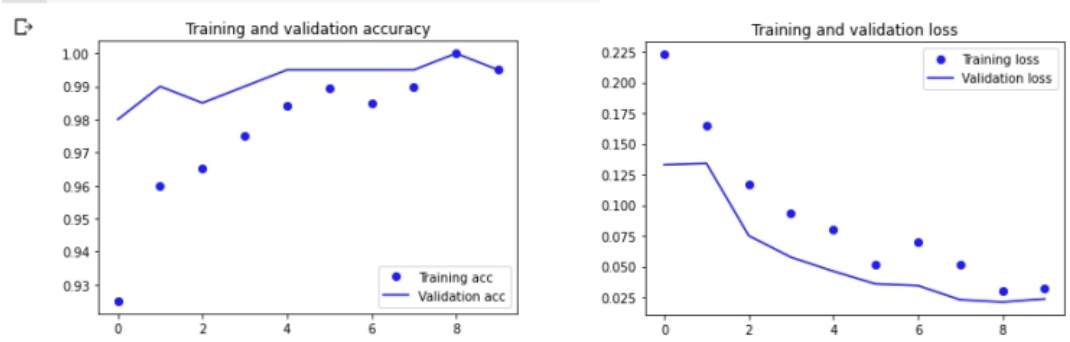
Şekil 6.8. VGG16 Modeli ile Eğitim - Fine Tuning Sonrası

```
## Checking model's accuracy
print('Training accuracy: {:.3f}'.format(history.history['acc'][-1]))
print('Validation accuracy: {:.3f}'.format(history.history['val_acc'][-1]))
```

```
Training accuracy: 0.995
Validation accuracy: 0.995
```

Şekil 6.9. VGG16 Modeli ACC

Eğitim sonrası elde edilen eğitim ve validasyon veri seti ile başarı 0.995 olarak gözlemlenmiştir. Ek olarak aşağıdaki grafikte modelin acc ve loss değerlerinin epoch ile değişimi gösterilmiştir.



Şekil 6.10. VGG16 Modeli ACC ve LOSS Grafiği

VGG16 modeli ile eğitim 910 adet eğitim ve 468 adet validasyon veri seti ile 25 dakikalık bir sürede gerçekleşmiştir.

6.3.2. ResNet50 Modeli ile Çalışma

ResNet50 mimarisinden Transfer Öğrenmesi Modelleri başlığı altında bahsedildiği gibi standart CNN modellerinden farklı olarak mikro mimari yapıdadır. Birçok görüntü sınıflandırma probleminde yüksek başarı oranlarına sahip olan ResNet50 modelinin veri setimiz üzerindeki başarı oranı deneyimlenmiştir. Tez çalışmamızda kullanılan veri seti ResNet50 modeli ile transfer öğrenmesi gerçekleştirilmiş, 10 epoch ile eğitilen modelde catgorical cross entropy kayıp fonksiyonu ve SGD optimizer kullanılmıştır.

```
Epoch 1/10
10/10 [==============================] - 245s 25s/step - loss: 6.3967 - accuracy: 0.5118 - val_loss: 0.2580 - val_accuracy: 0.9338
Epoch 2/10
10/10 [==============================] - 75s 8s/step - loss: 0.1597 - accuracy: 0.9645 - val_loss: 0.0520 - val_accuracy: 0.9786
Epoch 3/10
10/10 [==============================] - 59s 6s/step - loss: 0.0694 - accuracy: 0.9891 - val_loss: 0.0478 - val_accuracy: 0.9872
Epoch 4/10
10/10 [==============================] - 48s 5s/step - loss: 0.0096 - accuracy: 0.9962 - val_loss: 0.0509 - val_accuracy: 0.9872
Epoch 5/10
10/10 [==============================] - 46s 5s/step - loss: 0.0142 - accuracy: 0.9979 - val_loss: 0.0429 - val_accuracy: 0.9893
Epoch 6/10
10/10 [==============================] - 44s 5s/step - loss: 0.0215 - accuracy: 0.9900 - val_loss: 0.0260 - val_accuracy: 0.9915
Epoch 7/10
10/10 [==============================] - 42s 4s/step - loss: 0.0031 - accuracy: 0.9988 - val_loss: 0.0241 - val_accuracy: 0.9936
Epoch 8/10
10/10 [==============================] - 42s 4s/step - loss: 0.0032 - accuracy: 0.9994 - val_loss: 0.0296 - val_accuracy: 0.9936
Epoch 9/10
10/10 [==============================] - 42s 4s/step - loss: 0.0069 - accuracy: 0.9977 - val_loss: 0.0206 - val_accuracy: 0.9893
Epoch 10/10
10/10 [==============================] - 43s 5s/step - loss: 0.0044 - accuracy: 0.9986 - val_loss: 0.0182 - val_accuracy: 0.9936
```

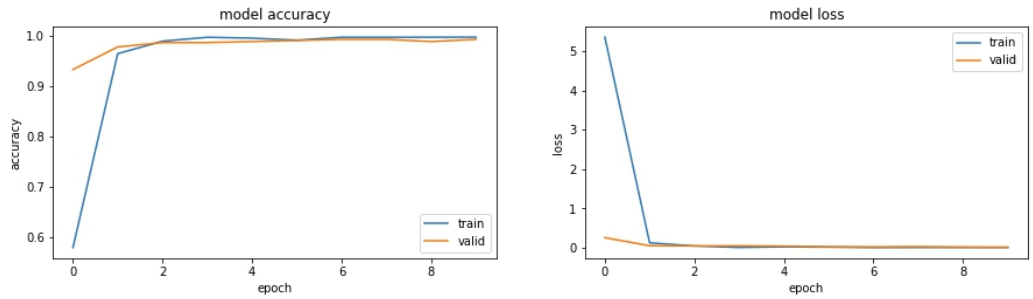
Şekil 6.11. ResNet50 Modeli ile Eğitim

```
## Checking model's accuracy
print('Training accuracy: {:.3f}'.format(fit_history.history['accuracy'][-1]))
print('Validation accuracy: {:.3f}'.format(fit_history.history['val_accuracy'][-1]))
```

```
Training accuracy: 0.998
Validation accuracy: 0.994
```

Şekil 6.12. ResNet50 Modeli ACC

Eğitim sonrası elde edilen eğitim veri seti ile başarı 0.998, validasyon veri seti ile başarı 0.994 olarak gözlemlenmiştir. Ek olarak aşağıdaki grafikte modelin acc ve loss değerlerinin epoch ile değişimi gösterilmiştir.



Şekil 6.13. ResNet50 Modeli ACC ve LOSS Grafiği

ResNet50 modeli ile eğitim 910 adet eğitim ve 468 adet validasyon veri seti ile 14 dakikalık bir sürede gerçekleşmiştir.

6.3.3. Xception Modeli ile Çalışma

VGG16 ve ResNet50 mimarisine oranla farklı bir yaklaşım ile modellenen derinlemesine ayrılabilir konvolüsyonlardan oluşan Xception modeli ile tez çalışmamızda kullanılan veri seti eğitilmiştir. Modelde NAdam optimizer ve categorical cross entropy kayıp fonksiyonu kullanılmış. 10 Epochta eğitilen modele fine tuning uygulanıp sonuçlar aşağıdaki görsellerle paylaşılmıştır.

```
Epoch 1/10
10/10 [==============================] - 183s 18s/step - loss: 0.6425 - accuracy: 0.6814 - val_loss: 0.5078 - val_accuracy: 0.9031
Epoch 2/10
10/10 [==============================] - 91s 9s/step - loss: 0.4775 - accuracy: 0.9281 - val_loss: 0.3703 - val_accuracy: 0.9750
Epoch 3/10
10/10 [==============================] - 63s 7s/step - loss: 0.3518 - accuracy: 0.9873 - val_loss: 0.2993 - val_accuracy: 0.9906
Epoch 4/10
10/10 [==============================] - 49s 5s/step - loss: 0.2741 - accuracy: 0.9889 - val_loss: 0.2273 - val_accuracy: 0.9844
Epoch 5/10
10/10 [==============================] - 43s 4s/step - loss: 0.2183 - accuracy: 0.9726 - val_loss: 0.1946 - val_accuracy: 0.9969
Epoch 6/10
10/10 [==============================] - 35s 4s/step - loss: 0.1856 - accuracy: 0.9885 - val_loss: 0.1644 - val_accuracy: 0.9937
Epoch 7/10
10/10 [==============================] - 34s 4s/step - loss: 0.1526 - accuracy: 0.9887 - val_loss: 0.1434 - val_accuracy: 0.9906
Epoch 8/10
10/10 [==============================] - 33s 3s/step - loss: 0.1358 - accuracy: 0.9881 - val_loss: 0.1269 - val_accuracy: 0.9969
Epoch 9/10
10/10 [==============================] - 32s 3s/step - loss: 0.1220 - accuracy: 0.9908 - val_loss: 0.1120 - val_accuracy: 0.9969
Epoch 10/10
10/10 [==============================] - 32s 3s/step - loss: 0.1115 - accuracy: 0.9794 - val_loss: 0.1022 - val_accuracy: 0.9969
```

Şekil 6.14. Xception Modeli ile Eğitim - Fine tuning Öncesi

```
Epoch 1/10
10/10 [==============================] - 36s 3s/step - loss: 0.3436 - accuracy: 0.8219 - val_loss: 0.0603 - val_accuracy: 0.9844
Epoch 2/10
10/10 [==============================] - 31s 3s/step - loss: 0.0557 - accuracy: 0.9670 - val_loss: 0.0557 - val_accuracy: 0.9688
Epoch 3/10
10/10 [==============================] - 32s 3s/step - loss: 8.5373e-04 - accuracy: 1.0000 - val_loss: 0.0558 - val_accuracy: 0.9719
Epoch 4/10
10/10 [==============================] - 31s 3s/step - loss: 4.5081e-04 - accuracy: 1.0000 - val_loss: 0.0189 - val_accuracy: 0.9937
Epoch 5/10
10/10 [==============================] - 31s 3s/step - loss: 6.7590e-04 - accuracy: 1.0000 - val_loss: 0.0358 - val_accuracy: 0.9906
Epoch 6/10
10/10 [==============================] - 31s 3s/step - loss: 3.6236e-04 - accuracy: 1.0000 - val_loss: 0.0263 - val_accuracy: 0.9937
Epoch 7/10
10/10 [==============================] - 31s 3s/step - loss: 3.6335e-04 - accuracy: 1.0000 - val_loss: 0.0056 - val_accuracy: 1.0000
Epoch 8/10
10/10 [==============================] - 31s 3s/step - loss: 1.3742e-04 - accuracy: 1.0000 - val_loss: 0.0024 - val_accuracy: 1.0000
Epoch 9/10
10/10 [==============================] - 30s 3s/step - loss: 1.1290e-04 - accuracy: 1.0000 - val_loss: 0.0013 - val_accuracy: 1.0000
Epoch 10/10
10/10 [==============================] - 31s 3s/step - loss: 8.3365e-05 - accuracy: 1.0000 - val_loss: 0.0012 - val_accuracy: 1.0000
```

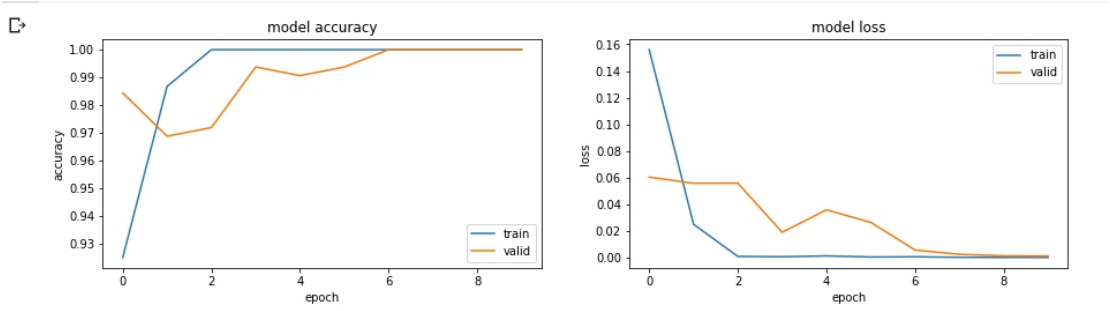
Şekil 6.15. Xception Modeli ile Eğitim - Fine tuning Sonrası

```
## Checking model's accuracy
print('Training accuracy: {:.3f}'.format(fit_history.history['accuracy'][-1]))
print('Validation accuracy: {:.3f}'.format(fit_history.history['val_accuracy'][-1]))
```

```
Training accuracy: 1.000
Validation accuracy: 1.000
```

Şekil 6.16. Xception Modeli ACC

Eğitim sonrası elde edilen eğitim ve validasyon veri seti ile %100 başarı sağlanmıştır. Ek olarak aşağıdaki grafikte modelin acc ve loss değerlerinin epoch ile değişimi gösterilmiştir.



Şekil 6.17. Xception Modeli ACC ve LOSS Grafiği

Xception modeli ile eğitim 910 adet eğitim ve 468 adet validasyon veri seti ile 17 dakikalık bir sürede gerçekleşmiştir.

7. SONUÇLAR

Bu tez çalışmasında, insan elinin üretimde müdahil olduğu bir fabrikanın tek bir modeline ait ürününün hatalı ya da doğru ürün olarak ayrıştırılması deneysel çalışması gerçekleştirilmiştir. Fabrikanın üretim aşamasında banttan alınan ham verilere veri arttırma yöntemleri uygulanarak 455 hatalı 455 doğru veri içeren 910 adet veriden oluşan eğitim veri seti ve 234 hatalı 234 doğru veri içeren 468 adet validasyon veri seti oluşturulmuştur.

Standart CNN modeli ve transfer öğrenmesi modelleri incelenmiş farklı mimaride 3 transfer öğrenmesi modeli seçilerek tez çalışması yürütülmüştür. Çalışma sonuçlarında transfer öğrenmesi modellerinin standart CNN modeline oranla daha yüksek başarı oranı elde ettiği gözlemlenmiş ancak tez çalışmamızda kullanılan veri ile en kısa sürede eğitim standart CNN modeli ile gerçekleştiği gözlemlenmiştir. CNN modeli ile eğitim veri setinde %84 başarı oranı yakalanmıştır.

Transfer öğrenmesi modellerinden VGG16 ile yapılan çalışmada eğitim veri setinde %99'luk başarı oranı sağlanmasına rağmen eğitim 25 dk gibi bir sürede gerçekleşmiş ve bu süre diğer modeller arasında en yüksek süre olarak kaydedilmiştir. ResNet50 modeli ile transfer öğrenmesi uygulanan veri setinin eğitim başarı oranı %99 ve eğitim süresi 14 dk olarak gözlemlenmiştir. Seçilen transfer öğrenmesi modelleri içerisinde Xception modeli tez çalışmamızdaki veri seti üzerinde %100 sağlayıp 17 dk gibi ortalama bir sürede eğitimi tamamlamıştır. Aşağıdaki görselde özet tablo ile başarım oranları ve eğitim süreleri karşılaştırması gösterilmiştir.

Tablo 7.1. Sonuçların Karşılaştırılması

	VGG16	**ResNet50**	**Xception**	**CNN**
Train ACC	0.995	0.998	1.000	0.844
Validation ACC	0.995	0.994	1.000	0.886
Tahmini Süre (dk)	25	14	17	11

Transfer öğrenmesinin az miktarda veride standart CNN modeline göre ne kadar başarılı olduğu bu tez çalışmasında da gözlemlenmiştir. Geliştirilen her modelde genellikle var olan sorunlara çözümler üretilmiş bu nedenle başarı oranları yükseltilmiştir. Bu tez çalışmasında da transfer öğrenmesi ile model eğitimi başarı oranı yüksek olduğu gözlemlenmiş VGG16, ResNet50 ve Xception modelleri arasında hem başarım oranı hem de süre bakımından çok az farkla sonuçlar elde edilmiş yine de bu veri seti üzerinde en başarılı transfer öğrenmesi modeli Xception olarak gözlemlenmiştir.

KAYNAKLAR

1. Kılıç, B. Konvolüsyon Sinir Ağlarında Transfer Öğrenmesi Yöntemi ile Araç Modellerinin Sınıflandırılması. Mersin Üniversitesi, Elektrik Elektronik Mühendisliği Anabilim Dalı, Mersin, 2019, 60 s. (Yüksek Lisans Tezi).
2. Abdulghani, N., Fadhil, A. F., Gültekin, S. S. Transfer learning using Alexnet with Support Vector Machine for Breast Cancer Detection. European Journal of Science and Technology. 2020, (Special Issue), 423-430.
3. Fırıldak, K., Talu, M. F. Evrişimli Sinir Ağlarında Kullanılan Transfer Öğrenme Yaklaşımlarının İncelenmesi. Anatolian Journal of Computer Science. 2019, 4(2), 88-95.
4. Karahan, T., Nabiyev, V. Konvolüsyonel Yapay Sinir Ağları ve Öğrenme Transferi ile Bitki Tanıma. Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi. 2020, DOI: 10.5505/2020.84042.
5. Kaya, A., Keçeli, A.S., Çatal, Ç., Yalic, H.Y., Huseyin, T., Tekinerdoğan, B. Analysis of transfer learning for deep neural network based plant classification models. Computers and Electronics in Agriculture, 2019, 158, 20-29.
6. Lasseck, M. Image-based Plant Species Identification with Deep Convolutional Neural Networks. CLEF, Dublin, Ireland, 11-14 Eylül 2017.
7. Abanoz, H. Emotion Recognition On Static Images Using Transfer Learning And Ensembling. Istanbul Technical University, Graduate School Of Science Engineering And Technology, İstanbul, 2018, 67 s. (Yüksek Lisans Tezi)
8. Yaraş, N. Vehicle Type Classification With Deep Learning. İzmir Institute of Technology, Computer Engineering, İzmir, 2020, 77 s. (Yüksek Lisans Tezi)
9. Internet, Tekeş, M. Denetimli (Supervised) Öğrenme ve Denetimsiz (Unsupervised) Öğrenme. https://mihribantekes.medium.com/makine-%C3%B6%C4%9Frenmesi-denetimli-supervised-e%C4%9Fitim-ve-denetimsiz-unsupervised-e%C4%9Fitim-5e3730354a5b
10. Atalay, M., Çelik, E. Büyük Veri Analizinde Yapay Zekâ Ve Makine Öğrenmesi Uygulamaları. Mehmet Akif Ersoy Üniversitesi Sosyal Bilimler Enstitüsü Dergisi. 2017, 9(22), 155-172.
11. Internet, DevHunter. Yarı Denetimli Öğrenme (Semi-Supervised Learning). https://devhunteryz.wordpress.com/2018/08/11/yari-denetimli-ogrenme-semi-supervised-learning
12. Kaelbling, L. P., Littman, M. L., and Moore, A. P. Reinforcement learning: A Survey. Journal of Artificial Intelligence Research. 1996, 237–285.
13. Hacıbeyoğlu, M. Çoklu Etmen Mimarisi ve Takviyeli Öğrenme. Selçuk Üniversitesi, Bilgisayar Mühendisliği Anabilim Dalı, Konya, 2006, 70 s. (Yüksek Lisans Tezi)
14. McCulloch, W. S., W. Pitts. A logical calculus of the ideas immanent in nervous activity. The Bulletin of Mathematical Biophysics.1943, 5(4), 115–133.
15. İnik, Ö., Ülker, E. Derin Öğrenme ve Görüntü Analizinde Kullanılan Derin Öğrenme Modelleri. Gaziosmanpaşa Bilimsel Araştırma Dergisi. 2017, 6(3), 85-104.
16. Internet, Kızrak, M. A. Şu Kara Kutuyu Açalım: Yapay Sinir Ağları. https://ayyucekizrak.medium.com/%C5%9Fu-kara-kutuyu-a%C3%A7alim-yapay-sinir-a%C4%9Flar%C4%B1-7b65c6a5264a
17. Budak, H., Erpolat, S. Kredi Riski Tahmininde Yapay Sinir Ağları ve Lojistik Regresyon Analizi Karşılaştırılması. AJIT-e: Online Academic Journal of Information Technology. 2012, DOI: 10.5824/1309-1581.2012.4.002.x

18. Ataseven, B. Yapay Sinir Ağlari İle Öngörü Modellemesi. Dergipark, 2013, 10(39), 101-115.
19. Internet, Çarkacı, N. Derin Öğrenme Uygulamalarında En Sık kullanılan Hiper-parametreler. https://medium.com/deep-learning-turkiye/derin-ogrenme-uygulamalarinda-en-sik-kullanilan-hiper-parametreler-ece8e9125c4
20. Gülcü, A., Kuş, Zeki. Konvolüsyonel Sinir Ağlarında Hiper-Parametre Optimizasyonu Yöntemlerinin İncelenmesi. Gazi Üniversitesi Fen Bilimleri Dergisi. 2019, 7(2), 503-522.
21. Ruder, S. An Overview of Gradient Descent Optimization Algorithms. 2016, DOI:2016arXiv160904747R.
22. Internet, Kızrak, M. A. Derine Daha Derine: Evrişimli Sinir Ağları. https://ayyucekizrak.medium.com/deri%CC%87ne-daha-deri%CC%87ne-evri%C5%9Fimli-sinir-a%C4%9Flar%C4%B1-2813a2c8b2a9
23. Deniz, E., Şengür, A., Kadiroğlu, Z., Guo, Y., Bajaj, V., Budak, Ü. Transfer learning based histopathologic image classification for breast cancer detection. Health Inf Sci Syst. 2018, DOI:10.1007/s13755-018-0057-x
24. Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille A. L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. 2016, DOI:arXiv1606.00915.
25. Doğan, F., Tükoğlu, İ. Derin Öğrenme Algoritmalarının Yaprak Sınıflandırma Başarımlarının Karşılaştırılması. Sakarya University Journal Of Computer And Information Sciences. 2018, VOL. 1, ID. SAUCIS-1-2018
26. Internet, Fabien, M. XCeption Model and Depthwise Separable Convolutions. https://maelfabien.github.io/deeplearning/xception/
27. Kassani, S.H. Breast Cancer Diagnosis with Transfer Learning and Global Pooling. International Conference on Information and Communication Technology Convergence (ICTC). 2019, DOI: 10.1109/ICTC46691.2019.8939878

ÖZGEÇMİŞ