YAPAY SİNİR AĞLARI İLE YÜZ İFADESİ TANIMA

Fatih Ateşoğlu
161402201

YÜKSEK LİSANS TEZİ
Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Yüksek Lisans Programı
Danışman: Dr. Öğr. Üyesi Ayşe Salman

İstanbul
T.C. Maltepe Üniversitesi
Lisansüstü Eğitim Enstitüsü
Haziran 2020

YAPAY SİNİR AĞLARI İLE YÜZ İFADESİ TANIMA

Fatih Ateşoğlu
161402201
Orcid: 0000-0001-6174-5549

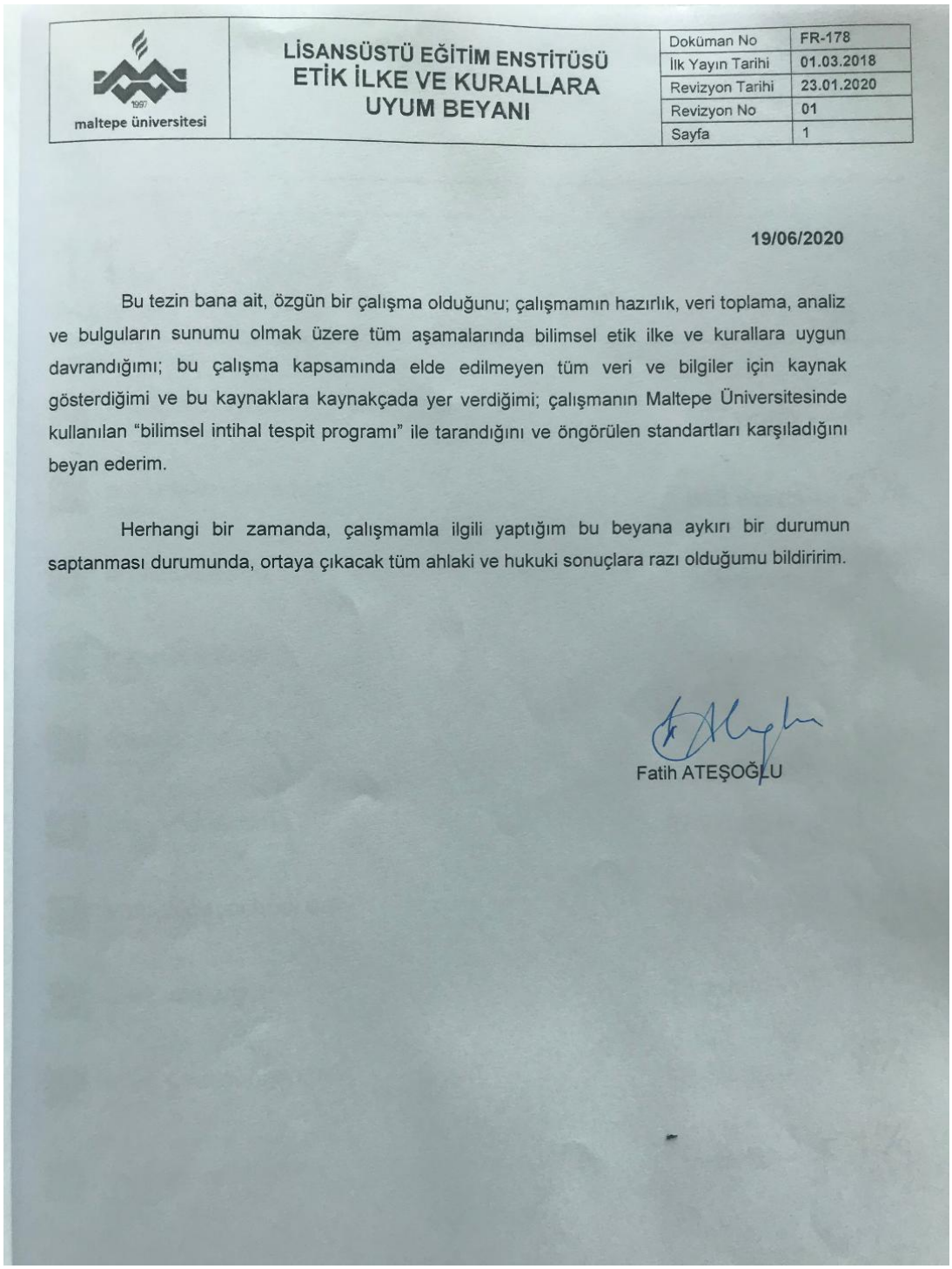
YÜKSEK LİSANS TEZİ
Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Yüksek Lisans Programı
Danışman: Dr. Öğr. Üyesi Ayşe Salman

İstanbul
T.C. Maltepe Üniversitesi
Lisansüstü Eğitim Enstitüsü
Haziran 2020



JÜRİ VE ENSTİTÜ ONAYI

ETİK İLKE VE KURALLARA UYUM BEYANI



maltepe üniversitesi

LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
ETİK İLKE VE KURALLARA UYUM BEYANI

Doküman No	FR-178
İlk Yayın Tarihi	01.03.2018
Revizyon Tarihi	23.01.2020
Revizyon No	01
Sayfa	1

19/06/2020

Bu tezin bana ait, özgün bir çalışma olduğunu; çalışmamın hazırlık, veri toplama, analiz ve bulguların sunumu olmak üzere tüm aşamalarında bilimsel etik ilke ve kurallara uygun davrandığımı; bu çalışma kapsamında elde edilmeyen tüm veri ve bilgiler için kaynak gösterdiğimi ve bu kaynaklara kaynakçada yer verdiğimi; çalışmanın Maltepe Üniversitesinde kullanılan "bilimsel intihal tespit programı" ile tarandığını ve öngörülen standartları karşıladığını beyan ederim.

Herhangi bir zamanda, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun saptanması durumunda, ortaya çıkacak tüm ahlaki ve hukuki sonuçlara razı olduğumu bildiririm.

Fatih ATEŞOĞLU

TEŞEKKÜR

Bu tez çalışmasının seçimi ve yürütülmesinde ve sonuçlandırılmasında destek ve yardımlarını esirgemeyen danışman hocam sayın Dr. Öğretim Üyesi Ayşe Salman'a ve Bilgisayar Mühendisliği Anabilim Dalı Başkanı sayın Prof. Dr. Ahmet Mesut Razbonyalı'ya teşekkür ederim.

Fatih Ateşoğlu

Haziran 2020

ÖZ

YAPAY SİNİR AĞLARI İLE YÜZ İFADESİ TANIMA

Fatih Ateşoğlu
Yüksek Lisans Tezi
Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Yüksek Lisans Programı
Danışman: Dr. Öğr. Üyesi Ayşe Salman
Maltepe Üniversitesi Lisansüstü Eğitim Enstitüsü, 2020

En basit tanımıyla Yapay zekâyı, insanı taklit etmeye çalışan makineler olarak da açıklayabiliriz. Yapay zekânın alt dallarından birisi olan yapay sinir ağları, problem çözmede insanın yerini almaya en yakın aday sistemlerdendir. Yapay sinir ağları, insan beyni gibi davranarak bir takım karmaşık hesaplamalar yapmak suretiyle verilen probleme çözüm bulabilirler.

Yapılan bu çalışmada yapay sinir ağları, bu karmaşık problemleri çözebilme gücü sayesinde, yüzden acı ifadesi okuma gibi bir problemi çözmek amacıyla kullanılmıştır.

Anahtar Sözcükler: Yapay Zekâ, Makine Öğrenmesi, Yapay Sinir Ağları, YSA, Yüz İfadesi Tanıma.

ABSTRACT

FACIAL EXPRESSION RECOGNITION WITH ARTIFICIAL NEURAL NETWORKS

Fatih Ateşoğlu
Master Thesis
Department of Computer Engineering
Computer Engineering Programme
Advisor: Asst. Prof. Ayşe Salman
Maltepe University Graduate Education Institute, 2020

In its simplest definition, artificial intelligence could be explained as machines trying to simulate human beings. Artificial neural networks, which is the sub-branche of artificial intelligence, are one of the closest candidate systems to replace with human in problem solving. Artificial neural networks could be solve the problems via performing complex calculations by acting as human brain.

In this study, because of the power of artificial neural network to solve the complex problems, we aimed the pain detection from expressions at human face as a problem needs to solve.

Keywords: Artificial Intelligence, Machine Learning, Artificial Neural Networks, ANN, Facial Expression Recognition.

İÇİNDEKİLER

JÜRİ VE ENSTİTÜ ONAYI ii
ETİK İLKE VE KURALLARA UYUM BEYANI iii
TEŞEKKÜR iv
ÖZ v
ABSTRACT vi
TABLOLAR LİSTESİ ix
ŞEKİLLER LİSTESİ x
KISALTMALAR xi
BÖLÜM 1. GİRİŞ 1
BÖLÜM 2. ÖN BİLGİ VE TEMEL KAVRAMLAR 3
2.1 Yapay Zekâ 3
2.1.1 Yapay Zekânın Tarihçesi 5
2.1.2 Yapay Zekâ Yaklaşımları 9
2.1.3 Yapay Zekâ Türleri 11
2.2 Makine Öğrenmesi 14
2.2.1 Öğrenme Stratejileri 15
2.2.2 Kullanım Alanları 15
2.3 Bilgisayarlı Görme 17
2.3.1 Tarihçesi: 18
2.3.2 Bilgisayarlı Görmenin Basamakları 18
2.3.3 Bilgisayarlı Görme Alt Dalları 19
2.3.4 Kullanım Alanları: 19
2.4 Yapay Sinir Ağları 21
2.4.1 Biyolojik Sinir Ağı 21
2.4.2 Yapay Sinir Ağı (YSA) 22
2.4.3 Yapay Sinir Ağının Tarihçesi 24
2.4.4 Bir Yapay Sinir Hücresinin Temel Elemanları 27
2.4.5 Yapay Sinir Ağının Yapısı: 29
2.4.6 Öğrenme Stratejileri 31
2.4.7 Öğrenme Kuralları 33
2.4.8 Yapay Sinir Ağlarının Kullanım Alanları 35
2.4.9 Endüstriyel Uygulamalar 36
2.4.10 Yapay Sinir Ağlarının Oluşturulması 37

2.4.11 Yapay Sinir Ağı Kullanarak Örüntü Tanıma....45
2.5 Ağrı....47
2.5.1 Ağrının Tanımı:....47
2.5.2 Ağrının Tespit Edilmesi....48
BÖLÜM 3. YÖNTEM....51
3.1 Veri Toplama....51
3.2 Veri Ön İşleme....51
3.3 Yapay Sinir Ağını Oluşturma ve Ağın Eğitilmesi....52
3.4 Ağrı İfadesinin Tanınması....54
BÖLÜM 4. SONUÇ VE TARTIŞMA....58
BÖLÜM 5. EKLER....59
5.1 EK-1 Eğitim ve Test Aşamasında Kullanılan Fotoğraflar....59
5.2 EK-2 Kaynak Kodlar....62
KAYNAKÇA....77

TABLOLAR LİSTESİ

Tablo 2. 1 Örneklerle zekâ türleri. 4
Tablo 2. 4 Biyolojik ve yapay sinir ağlarının karşılaştırılması (Elmas, 2018). 23
Tablo 2. 5 Toplam Fonksiyonları (Çayıroğlu, 2019). 28
Tablo 2. 6 Aktivasyon Fonksiyonları (Çayıroğlu, 2019). 29
Tablo 2. 7 Yapay sinir ağları ve uygulama alanları (Stuart et.al, 1995). 36
Tablo 3. 1 Ağrı tahmini test sonuçları. 57

ŞEKİLLER LİSTESİ

Şekil 2. 1 Yapay zekâ tipleri. 11
Şekil 2. 15 Bilgisayarlı görme işleminin blok diyagramı. 18
Şekil 2. 16 Optik karakter tanıma şeması. 20
Şekil 2. 17 Biyolojik ve yapay sinir hücreleri. 23
Şekil 2. 19 Algılayıcı (Perceptron). 25
Şekil 2. 20 Bir yapay sinir hücresi. 27
Şekil 2. 21 Yapay sinir ağının yapısı. 30
Şekil 2. 22 Öğretmenli öğrenme blok diyagramı. 31
Şekil 2. 23 Öğretmensiz öğrenme blok diyagramı. 32
Şekil 2. 24 Takviyeli öğrenme blok diyagramı. 33
Şekil 2. 25 Tek Katmanlı Algılayıcı Modeli (Öztemel, 2003). 37
Şekil 2. 26 Basit Algılayıcı (Perceptron). 38
Şekil 2. 27 İleri Beslemeli Yapay Sinir Ağı. 40
Şekil 2. 28 ÇKA modeli. 41
Şekil 2. 29 Düzgün bir A karakterinin 7x5'lik bir matrisle ikili gösterimi. 46
Şekil 2. 30 Elyazısı karakter tanımada YSA kullanımı. 47
Şekil 2. 31 Ağrı tespit ölçeği örnekleri. 49
Şekil 2. 32 Ağrı Tespit Cetvelleri (URL-8, 2019). 50
Şekil 3. 1 Adım adım ön işlem aşamaları. 51
Şekil 3. 2 Ağırlık setine göre YSA'nın çıkış hatası değişimi. 53
Şekil 3. 3 Performans arttırmak için kullanılan yöntem (Martella, 2005). 54
Şekil 3. 4 Yüzdeki ağrı ifadesi tahmini uygulamasının algoritma şeması. 56
Şekil 5. 1 Ağın eğitiminde kullanılan normal yüz ifadeleri. 59
Şekil 5. 2 Ağın eğitiminde kullanılan az ağrılı yüz ifadeleri. 60
Şekil 5. 3 Ağın eğitiminde kullanılan çok ağrılı yüz ifadeleri. 60
Şekil 5. 4 Ağı test etmekte kullanılan fotoğraflar-1 61
Şekil 5. 5 Ağı test etmekte kullanılan fotoğraflar-2 61
Şekil 5. 6 Ağı test etmekte kullanılan fotoğraflar-3 62

KISALTMALAR

YSA	Yapay Sinir Ağı	Artificial Neural Network
ÇKA	Çok Katmanlı Algılayıcı	Multi Layer Perceptron
CPU	Merkezi İşlem Birimi	Central Processing Unit
GPU	Grafik İşleme Birimi	Graphics Processing Unit
YZ	Yapay Zekâ	Artificial İntelligence
GOFAI	Sembolik Yapay Zekâ Yaklaşımı	Good Old-Fashioned Artificial Intelligence
BVLC	Berkeley Görme ve Öğrenme Merkezi	Berkeley Vision and Learning Center
ADN	Aktif Derin Ağ	Active Depth Network
DBN	Derin İnanç Ağı	Depth Bayes Network
GPS	Genel Problem Çözücü	General Problem Solver
CV	Bilgisayarlı Görme	Computer Vision
NLP	Doğal Dil İşleme	Natural Language Processing
LVQ	Doğrusal Vektör Niceleme	Linear Vector Quantization
LMS	En Küçük Ortalamalar Karesi	Least Mean Square
SVM	Destek Vektör Makineleri	Support Vector Machines
ASCII	Bilgi Değişimi İçin Amerikan Standart Kodlama Sistemi	American Standard Code for Information Interchange
IASP	Uluslararası Ağrı Araştırmaları Teşkilatı	International Association of Science Parks and Areas of Innovation
CNN	Konvolüsyonel Sinir Ağları	Convolutional Neural Network

LSTM	Uzun-Kısa vadeli HafızaAğları	Long-Short Term Memory
DAE	Derin Oto-kodlayıcılar	Deep Autoencoders

ÖZGEÇMİŞ

Fatih ATEŞOĞLU

Bilgisayar Mühendisliği Anabilim Dalı

Eğitim

Derece	**Yıl**	**Üniversite, Enstitü, Anabilim/ Anasanat Dalı**
Ls.	2006	Fırat Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü
Lise	2000	Elazığ Balakgazi Lisesi

İş/İstihdam

Yıl	**Görev**
2007 - 2007	Uzman Yazılım Mühendisi, Tepum Sigma Bilişim Danışmanlık A.Ş.
2008 - devam	Mühendis, İstanbul Büyükşehir Belediyesi

Kişisel Bilgiler

Doğum Yeri ve Yılı : Elazığ 1983 Cinsiyeti: E

Yabancı Diller : İngilizce (Orta Seviye)

e-posta : atesoglufatih@yahoo.com

BÖLÜM 1. GİRİŞ

Her şeyin dijitale doğru gittiği çağımızda, sağlık sektörü de bu dönüşüme kayıtsız kalmamış olup, neredeyse bütün alanlarda aynı dönüşüme ayak uydurmuştur. Bu alanlardan bir tanesi de ağrı tespitidir. Farklı hasta gruplarında görülen ağrının niceliği ve niteliği, günümüzde önemli bir konu haline gelmiş olup, bu alanda birçok bilimsel çalışma yapılmış ve çok sayıda akademik yazılar yazılmıştır.

Çeşitli nedenlerle kendini ifade etmekte yetersiz kalan hastaların (zekâ özürlü hastalar, bebekler, sağlık çalışanları ile aynı dili kullanmayan hastalar, çok yaşlı hastalar vb.) kendisini sağlık uzmanına tam anlamıyla ifade edememesinden, bazen de hastanın bilincinin kapalı olmasından dolayı, sağlık çalışanları tarafından ağrının tespitinin ve şiddetinin anlaşılması kolay olmamaktadır.

Ağrının varlığının ve şiddetinin tespiti, gerek hasta yakınları gerekse sağlık çalışanları için her zaman anlaşılmak istenmen bir konu olmuş olup, bu konuda birçok akademik çalışma yapılmıştır. Ağrının varlığının ve şiddetinin tespiti konusunda geleneksel yöntem, ya hastaya sorarak, ya hastaya gösterilen bir kâğıdın işaretlenmesiyle ya da şekillerin seçilmesiyle olmaktaydı. Fakat bu tespitin, daha önce bahsedilen nedenlerden dolayı her zaman bu kadar kolay olamaması konusunda, gelişen bilişim dünyasının da yardımıyla yapılan çalışmalar, sağlık çalışanlarının işini önemli ölçüde kolaylaştırmaktadır.

Son zamanlarda yapay zekânın artan gücüyle birlikte, diğer birçok alanda olduğu gibi tıp alanında da kullanımı ile ilgili olarak gerekli olan hastane, akademi ve işletmeler arasındaki işbirliğinin yetersizliği ve bunun geliştirilmesi konusunda çeşitli çalışmalar yapılmıştır. Bununla birlikte yapay zekânın tıp alanında geçmişten günümüze nerelerde ve ne amaçlarla kullanıldığı, kullanım avantaj ve dezavantajları gibi konularda da yapılan çalışmalar mevcuttur (Gardner, 1993).

Monwar yaptığı çalışmada, biri yüz özelliklerine dayalı diğeri YSA ile olmak üzere iki farklı ağrı tespit yöntemini karşılaştırmış olup, eğitim süresi ve tanıma süresi olarak YSA'nın daha başarılı olduğunu belirtilmiştir. Doğruluk karşılaştırmasında ise,

çok küçük bir farkla YSA üstün olmakla birlikte iki yöntemin de yaklaşık doğruluk oranlarına sahip olduğuna değinilmektedir (Monwar, 2006).

Zhu çalışmasında, video dizisinden yüzdeki ağrı ifadesini tanıyıp 4 ayrı kategoriye ayıran bir uygulama geliştirmiştir. Bu çalışmasında Zhu, Olasılıksal Gizli Semantik Analiz (PLSA) adını verdiği bu ağrı tespit yöntemi ile Yüz Özelliklerine Dayalı (Eigenface) Ağrı Tespiti, yüz özelliklerini çıkarmak için Aktif Görünüm Modellerini (AAM) ve ağrıyı sınıflandırmak için Destek Vektör Makineleri(SVM)'ni kullanan "AAM + SVM" modeli ve "SLPP + MKSVM" yöntemlerini kıyaslamış olup kendi yöteminin diğer üç yöntemden daha başarılı olduğunu savunmaktadır (Zhu, 2014).

Acının varlığının ve şiddetinin kolay, hızlı ve doğru şekilde anlaşılması, sağlık uzmanının hastanın vücudunun doğru yerine, doğru zamanda müdahale etmesi noktasında hayati önem arz edebilmektedir. Örneğin kırık yüzünden hastanede yatan yaşlı hastalarda en yaygın sözsüz tepkinin **yüz buruşturma** olduğu, bu tür hastalarda hareket ağrıya neden olduğu için hareketsiz yatma eğiliminin fazla olduğu belirlenmiştir (URL-1, 2019). Bu durum bize insan yüzündeki ifadelerin sağlık açısından ne kadar önemli ipuçları verdiğini göstermektedir. Bu nedenle çalışmamızda, insan yüzündeki ağrı ifadesinin bilgisayarlar yardımıyla yüz ifadesinden okunarak tahmin edilmesi sağlanarak, sağlık çalışanlarının işlerini kolaylaştırmaya yardımcı olmak amaçlanmıştır.

BÖLÜM 2. ÖN BİLGİ VE TEMEL KAVRAMLAR

2.1 Yapay Zekâ

İnsan beynine özgü bazı özelliklerin, makinelere kazandırılması işi olan yapay zekâ kavramını daha iyi anlamak için, öncelikle zekâ kavramının ne olduğunu kavramak gerekmektedir.

a) Zekâ Kavramı:

İlk kez karşılaşılan ya da anlık gelişen bir olaya uyum sağlayabilme, öğrenme, anlama, analiz yeteneği, duyuların, düşüncenin ve dikkatin o noktaya yoğunlaştırılması ancak zekâ ile mümkün olmaktadır (Elmas, 2018).

Zekâ; gerçek hayatta karşılaşılan problemlere etkili ve verimli çözümler üretebilme kabiliyeti, başka bir deyişle bir ürün veya bir değer ortaya koyabilme kapasitesidir (Gardner, 1993).

Gardner'a göre; "Toplumumuzda zekâ ile karıştırılan diğer bir kavram olan **akıl** ise; düşünme anlama, kavrama, idrak, muhakeme gibi yeteneklerin bütünüdür. Akıl, genetik yolla kişiye geçen sevgi, kıskançlık, korku gibi doğal güdülerle birlikte, yaşadığımız toplum ve çevreden etkilenerek zamanla gelişebilmektedir" (Elmas, 2018). Gardner'ın bahsettiği 6 zekâ türü ve bu türlere ait örnekler Tablo 2.1'de özetlenmiştir.

Gardner'ın teorisine ek olarak farklı kaynaklarda;

Toplumda diğer insanlarla çok iyi ilişkiler kurmada özgün yeteneklere sahip olma, sözlü ya da sözsüz efektif bir ilişki kurabilme ve sosyal ilişkilerde farklı bakış açıları geliştirebilme yeteneği olan **Sosyal Zekâdan**, hayvanları, bitkileri ve doğada bulunan diğer varlıkları ayırabilme yeteneği olan **Doğacı Zekâdan** (Nabiyev, 2016) ve kendi varlık sebeplerini düşünme, insanların neden doğduğu, hayata nasıl geldiği ve neden öldükleri, hayatın ve ölümün ötesinde neler olduğu konusunda akıl yürütebilecek kapasiteye ve duyarlılığa sahip olma yeteneği olan **Varoluşsal Zekâdan** bahsedilmektedir.

Tablo 2. 1 Örneklerle zekâ türleri.

Zekâ Türü	Açıklama	Örnek
Sözel (Dilsel) Zekâ	Fonoloji (konuşma sesleri), sözdizimi (dilbilgisi) ve semantik (anlam) mekanizmalarını konuşma, tanıma ve kullanma becerisi.	Anlatıcılar, Hatipler
Müzikal Zekâ	Ses, perde, ritim anlayışından oluşan anlamları yaratma, iletişim kurma ve anlama becerisi.	Müzisyenler, Şarkıcılar, Besteciler
Mantıksal-Sayısal Zekâ	Eylem veya nesnelerin yokluğunda ilişkileri kullanma ve anlama yeteneği. Karmaşık ve soyut fikirleri anlama.	Matematikçiler, Bilim Adamları
Görsel (Mekansal) Zekâ	Görsel veya mekansal bilgileri algılama, değiştirme ve nesnelere atıfta bulunmadan görsel görüntüler yeniden oluşturma, 3D görüntüler oluşturma ve bunları taşıma ve döndürme yeteneği.	Harita okuyucular, Astronotlar, Fizikçiler
Bedensel-Devinimsel Zekâ	Sorunları veya moda ürünlerini çözmek, ince ve kaba motor becerileri kontrol etmek ve nesneleri manipüle etmek için vücudun tamamını veya bir kısmını kullanma yeteneği.	Oyuncular, Dansçılar
Kişisel (İçsel) Zekâ	Kişinin kendi duygularını, niyetlerini ve motivasyonlarını ayırt etme yeteneği.	Yazarlar, Şairler, Dini Liderler

b) Yapay Zekâ Kavramı:

Yapay zekâ kısaca; insanlara özgü davranışlar olan öğrenme ve problem çözme becerilerinin bilgisayarlar tarafından taklit edilmesi işlemidir. Yapay zekânın farklı birçok tanımı olsa da, bazı kaynaklarda bu konuya ait; insan gibi düşünen, insan gibi davranan, rasyonel düşünen ve rasyonel davranan sistemler olarak dört farklı tanıma rastlamak mümkündür (Stuart ve ark., 1995).

Yapay zekaın günümüzde birçok alt dalı bulunmaktadır. Bunlardan bazıları şunlardır:

- Robotik
- Uzman Sistemler
- Doğal Dil İşleme
- Bulanık Mantık
- Bilgisayarlı Görme
- Yapay Sinir Ağları

2.1.1 Yapay Zekânın Tarihçesi

Fransız matematikçi ve filozof Descartes, 1637'de insan gibi düşünebilen bir makinenin asla yapılamayacağını söylemiştir. Buna karşılık, İngiliz matematikçi, kriptolog ve bilgisayar biliminin kurucusu sayılan Alan Mathison Turing, 1950'de ilk defa bilgisayarların insan gibi davranmayı öğrenebileceğini ve bilgisayarla uzaktan iletişim kuran bir insanın, kendisiyle konuşan makineyi insan sanmasının mümkün olacağını öne sürmüştür.

John McCarthy, ilk olarak 1955'te Amerikan ordusunda askerlik yaptığı sırada, Almanların gönderdiği mesajları çözmek için uğraşırken, yazılan uygun programlar sayesinde bilgisayarların da insanlar gibi düşünebileceği fikrini geliştirmiştir. "Yapay Zekâ" terimi ise ilk kez, yapay zekâ program dili Lisp'i, bilgisayarlarda zaman paylaşımı (time-sharing) ve e-ticaret kavramlarını da geliştirmiş olan John McCarthy tarafından 1956'da Dartmouth Konferansı'nda kullanılmıştır.

Yapay zekâ tarihçesinin daha iyi anlaşılabilmesi için, çeşitli tarihsel dönemlerdeki gelişmeler aşağıdaki şekilde özetlenebilmektedir.

- **Tarih Öncesi Dönem:**

Tarih öncesinde bile, insanınkine benzer yapay bir zekâ oluşturma fikri olmuştur. Mitolojide rüzgâr tanrısı olarak bilinen Daedelus'un, bir "yapay-insan" yapma teşebbüsü buna örnek olarak gösterilebilir (Yılmaz, 2018).

Yapay zekâ alanındaki diğer tarihi gelişmeleri şu şekilde sıralamak mümkündür:

Yapay zekâ için 1884 yılı, en önemli dönüm noktası olarak kabul edilmektedir. Charles Babbage adlı bilim adamı, **fark makinesi** adını verdiği ve bazı zeki davranışlar göstermesini istediği bir takım mekanik makineler üzerinde deneyler yapmıştır. O tarihte ortaya atılan bu fikir, daha sonra gelen bilim adamları tarafından geliştirilerek günümüzdeki halini almıştır.

- **Yapay zekânın olgunlaşması (1943-1952)**

Yıl 1943: Günümüzde kullanılan Yapay Zekâ alanında bilinen ilk çalışma, Warren McCulloch ve Walter Pits tarafından Yapay nöron modelinin geliştirilmesiyle yapıldı.

Yıl 1949: Donald Hebb nöronlar arasındaki bağlantı gücünü değiştirmek için **Hebbian Öğrenme** adının verdiği bir güncelleme kuralı geliştirdi.

Yıl 1950: Kendisi bir İngiliz matematikçi olan ve makine öğrenmesine öncülük eden Alan Turing**, "Bilgisayar Makineleri ve İstihbarat"** adlı bir test yayınladı. Turing testi olarak adlandırılan bu test, makinenin insan zekâsına eşdeğer zeki davranış sergileme yeteneğini kontrol etmektedir.

Yine bu yıllarda Shannon adındaki bir bilim adamı, bilgisayarların satranç oynayabileceğini savunmuştur (Yılmaz, 2018).

- **Yapay zekânın doğuşu (1952-1956)**

Yıl 1955: Allen Newell ve Herbert A. Simon "Mantık Teoristi" olarak adlandırılan ilk yapay zekâ programını yaptı. Bu program, 52 Matematik teoreminin 38'ini kanıtlamış ve bunun dışında bazı teoremler için yeni ve daha etkili çözümler bulmuştur.

Yıl 1956: Dartmouth Konferansı'nda John McCarthy tarafından ilk kez "Yapay Zekâ" deyimi ortaya atıldı. Bu, aynı zamanda yapay zekânın akademik bir alan olarak da icadı sayılmaktadır (Yılmaz, 2018).

Daha sonra FORTRAN, LISP veya COBOL gibi üst düzey bilgisayar dillerinin icat edilmesiyle yapay zekâ alanındaki gelişmeler artan bir ivme yakalamıştır.

- **Altın yıllar - erken coşku (1956-1974)**

Yıl 1965: Matematiksel problemleri çözebilecek algoritmalar geliştirmeyi amaçlayan yapay zekâ araştırmacısı Joseph Weizenbaum, sohbet programı ELİZA'yı ilk defa geliştirdi.

Yıl 1966: Stanford Üniversitesinde ilk hareketli robot olan "Shakey" üretildi.

Yıl 1972: Japonya'da ilk zeki insansı robot olan wabot-1 adlı robot yapıldı.

- **İlk Yapay zekâ kışı (1974-1980)**

1974-1980 yılları arasındaki Yapay zekâ kışı olarak adlandırılan yıllar, bilgisayar bilimcisinin yapay zekâ araştırmaları için resmi kurumlardan finansman desteği alamadığı süreyi ifade etmektedir. Bu süreç, aynı zamanda yapay zekâya olan ilginin azaldığı yılları ifade eder.

- **Yapay zekâ patlaması (1980-1987)**

Yıl 1980: Yapay zekâ kışından sonra, Uzman Sistemler (Expert Systems) ile yapay zekâ tekrar geri geldi. Bir insan uzmanının karar verme yeteneğini taklit eden uzman sistem adındaki programlar yapıldı.

Stanford Üniversitesi'nde, Amerika yapay zekâ derneğinin ilk ulusal Konferansı yapıldı.

- **İkinci yapay zekâ kışı (1987-1993)**

1987 ile 1993 yılları arasındaki süre ikinci yapay zekâ kışıydı. Yatırımcılar ve resmi kurumlar, yüksek maliyet ve verimli olmayan sonuçlar nedeniyle yapay zekâ araştırması için verilen fonları kestiler.

- **Zeki ajanların ortaya çıkışı (1993-2011)**

1997 yılı: IBM'in ürettiği Deep Blue adlı bilgisayar, dünya satranç şampiyonu Rus Gary Kasparov'u yendi ve dünya satranç şampiyonu olan ilk bilgisayar oldu [57].

Yıl 2002: Yapay zekâ ilk kez bir elektrikli süpürge olan Roomba adıyla evlere girdi.

Yıl 2006: Yapay zekâ iş dünyasına kadar girdi. Facebook, Twitter ve Netflix gibi şirketler de yapay zekâ kullanmaya başladı.

- **Derin Öğrenme (Deep Learning), Büyük Veri (Big Data) ve Yapay Zekâ (2011-2019)**

Yıl 2011: IBM'in bilgisayarı Watson, Jeopardy adlı karmaşık soruların yanı sıra bilmeceler de çözmek zorunda olunan bir yarışmada, bu yarışmayı arka arkaya 74 kez kazanan Ken Jennings ile yarışmadan büyük para ödülü kazanan yarışmanın rekortmeni Brad Rutter'a karşı yarıştı ve kazandı. Watson, bu yarışmada doğal dili anlayabileceğini ve zor soruları hızlı bir şekilde çözebileceğini kanıtladı.

Yine aynı yıl içinde Apple şirketi, bir yapay zekâ uygulaması olan sesli asistan özellikli Siri'yi piyasaya sundu.

Yıl 2012: Google şirketi, yapay zekâ uygulaması olan kişisel sesli asistan Google Asistan'ı piyasaya sürdü.

Yıl 2014: Reading Üniversitesi'nden Prof. Kevin Warwick, Rus bir programcı grubu tarafından 2001 yılında geliştirilmeye başlanan Eugene Goostman adlı sohbet programının, Turing testini geçtiğini iddia etti. Fakat Quebec Üniversitesi'nden Prof. Stevan Harnad, Turing testinin geçilmediğini, buna yakın bile olunamadığını ifade etti (URL-2, 2019).

Microsoft şirketi Cortana isimli yapay zekâlı sesli asistanı piyasaya sürdü.

Amazon şirketi Alexa adlı yapay zekâlı sesli asistanı piyasaya sürdü.

Yıl 2017: Samsung Bixby isimli yapay zekâlı sesli kişisel asistanı piyasaya sürdü.

Google ve Fiat Chrysler şirketlerinin ortak ürünü olan ve kendi kendini sürebilen arabalar ilk kez tanıtıldı (URL-3, 2019).

Yıl 2018: IBM şirketinin geliştirdiği Proje Tartışıcısı adlı yapay zekâ uygulaması, iki uzman tartışmacı ile karmaşık konularda tartıştı ve son derece iyi performans gösterdi (URL-4, 2019).

Google, sanal rezervasyon asistanı Duplex adlı yapay zekâ programını, kuaför ve restoran randevusu alma amacıyla denedi. Kuaför ve restoran çalışanları, bir bilgisayar programı ile konuştuğunu fark etmedi (URL-5, 2019).

Yapay zekâ günümüzde olağanüstü bir seviyeye geldi. Derin Öğrenme (Deep Learning), Büyük Veri (Big Data) ve Veri Bilimi (Data Science) kavramı şimdilerde gelişmekte olan güncel konuların başında gelmektedir. Günümüzde birçok büyük teknoloji şirketi yapay zekâ ile çalışan cihazlar yapmaya devam etmektedir.

2.1.2 Yapay Zekâ Yaklaşımları

Yapay zeka kavramı, zaman içinde yapısal olarak değişime uğramıştır. Bu sebeple yapay zeka denince eski (sembolik) ve yeni (ilişkisel) olmak üzere iki farklı yaklaşım söz konusudur.

Sembolik Yapay Zekâ Yaklaşımı:

Yapay zekâ tarihinde, araştırmacılar akıllı bir sistem oluşturmak için Sembolik ve Sembolik Olmayan yapay zekâ kullananlar olmak üzere iki gruba ayrılır. Eski moda da denen **sembolik yaklaşım**, kurallara ve bilgiye dayanan ve eylemleri yorumlanabilir olan akıllı bir sistem geliştirmeye çalışmışken, sembolik olmayan yaklaşım ise insan beynini taklit eden bir hesaplama sistemi oluşturmaya çalışmıştır.

Geleneksel sembolik yaklaşım, 1976'da Newell & Simon tarafından tanıtılan yapay zekâyı sembolik hile kullanan modellerin gelişimi olarak tanımlamaktadır. Bu yaklaşımda, yapay zekâ uygulamaları, sayılar veya harfler yerine, gerçek dünyadaki varlıkları veya kavramları temsil eden ve "sembol" denen karakter dizelerini işler. Arama ve temsil sembolik (GOFAI) yapay zekâ gelişiminde önemli bir rol oynamaktadır. GOFAI yaklaşımı, statik problemleri çözmede başarılıyken, Gerçek zamanlı dinamik problemleri çözmeye ise uyumlu değildir.

Sembolik yapay zekânın temel avantajlarından biri, akıl yürütme sürecinin kolayca anlaşılabilmesidir. Yani sembolik bir yapay zekâ programı, bir problemin çözümünde, sonuca nasıl ulaşıldığını ve akıl yürütme adımlarını kolayca açıklayabilir.

Sembolik yapay zekânın önemli bir dezavantajı ise, öğrenme sürecinde kuralların ve bilginin elle kodlanmasıdır ki, bu durum büyük çaplı problemlerde oldukça zor bir iş teşkil etmektedir. Örneğin, dili ayrıştırmak için gramer kullanan doğal dil işleme sistemleri, sembolik yapay zekâ sistemlerine dayanır. IBM'in 1997'de satranç şampiyonu Kasparov'u mağlup eden Deep Blue sistemi, sembolik yapay zekâ yaklaşımına örnek olarak verilebilir.

Sembolik Olmayan (İlişkisel) Yapay Zekâ Yaklaşımı:

İlişkisel Yapay Zekâ olarak da bilinen bu yaklaşım, günümüzde ağırlıklı olarak kullanılan yaklaşımdır. Bu yaklaşımda problemlere çözüm bulmak için sembol kullanmak yerine, çözüme nasıl ulaştığını açıklamadan, verilen problemi çözebildiğini gösteren bazı ilkelere göre hesaplamalar yapılır. Sembolik olmayan yapay zekânın temeli, insan beynini ve birbirine karmaşık biçimde bağlı nöronları taklit etmeye dayanmaktadır.

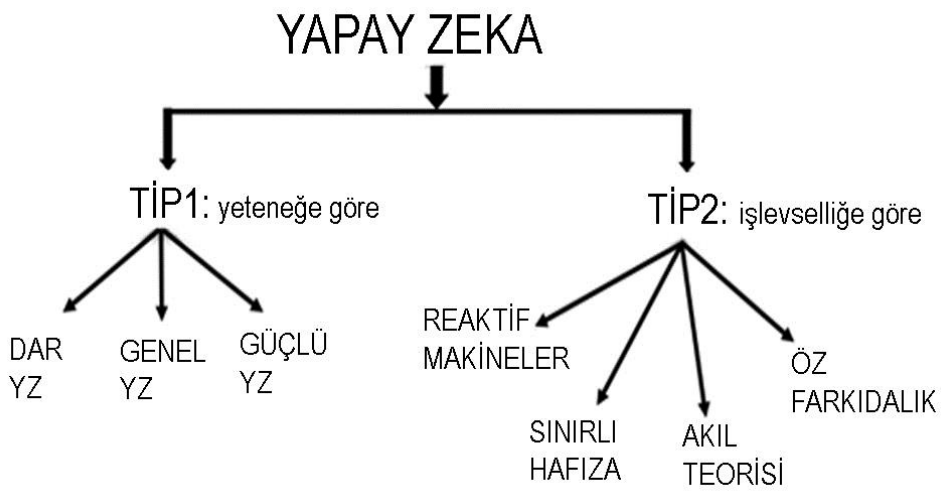
Sembolik olmayan yapay zekânın önemli bir **dezavantajı**, bir problemin çözümü sırasında, sistemin sonuca nasıl vardığını açıklayamamasıdır. Bu durum, özellikle kendini süren arabalar ve tıbbi teşhis koyan uzman sistemler gibi yüksek riskli uygulamalarda sembolik olmayan yapay zekâ kullanılmasında tereddüt yaşanmasına neden olabilmektedir.

Sembolik olmayan yapay zekâya örnek olarak, genetik algoritmalar, yapay sinir ağları ve derin öğrenme verilebilir. Günümüzde IBM'in Watson'ı, Facebook'un Yüz Tanıma algoritması, kendi kendini süren otomobil teknolojisi gibi birçok farklı alanda

sembolik olmayan yapay zekâ yaklaşımının kullanıldığı canlı sistemlere rastlamak mümkündür.

2.1.3 Yapay Zekâ Türleri

Yeteneklerine ve işlevselliğine göre yapay zekâ Tip1 ve Tip2 olmak üzere ikiye ayrılır (Şekil 2.1).



Şekil 2. 1 Yapay zekâ tipleri.

2.1.3.1 Yapay Zekâ Tip-1: Yeteneklerine Göre
Dar (Zayıf) Yapay Zekâ:

Dar Yapay Zekâ, istihbarat ile özel bir görevi yerine getirebilen bir yapay zekâ türüdür. En yaygın yapay zekâ tipidir. Dar yapay zekâ, yalnızca belirli bir görev için eğitildiği için alanının veya sınırlamalarının ötesinde performans gösteremez. Bu nedenle **Zayıf veya Dar Yapay Zekâ** olarak da adlandırılır.

Dar yapay zekâ, belirlenen sınırların ötesine geçerse öngörülemeyen şekilde başarısız olabilir. Apple Siri bir dar yapay zekâ örneğidir, önceden tanımlanmış sınırlı bir fonksiyon aralığı ile çalışır. IBM'in Watson süper bilgisayarı da, makine öğrenimi ve doğal dil işleme ile birlikte uzman bir sistem yaklaşımı kullandığı için dar yapay zekâ türüne örnek gösterilebilir. Dar yapay zekânın, satranç oynamak ya da e-ticaret sitesinde satın alma önerileri sunmak gibi diğer bazı örnekleri mevcuttur. Bunların dışında kendi kendini süren arabalar, konuşma tanıma ve görüntü tanıma gibi birçok alanda da dar yapay zekâ örneklerine rastlamak mümkündür.

Genel Yapay Zekâ:

Genel yapay zekâ, tıpkı bir insan gibi herhangi bir görevi yerine getirebilecek bir zekâ türüdür. Genel yapay zekânın arkasında, bir insan gibi kendi başına düşünebilen daha zeki bir sistem yapma fikri yatmaktadır.

Günümüzde bir görevi bir insan kadar mükemmel şekilde yerine getirebilecek böyle bir sistem halen mevcut değildir. Şu an dünya çapında birçok araştırmacı genel yapay zekâ makineleri geliştirmeye odaklanmıştır. Genel yapay zekâlı sistemler henüz araştırma aşamasında olduğundan, bu tür sistemleri geliştirmek için çok fazla çaba ve zaman gerekmektedir.

Süper(Güçlü) Yapay Zekâ:

Süper yapay zekâ, makinelerin insan zekâsını aşabileceği ve herhangi bir görevi bilişsel özelliklere sahip insandan daha iyi gerçekleştirebileceği bir sistem zekâsıdır. Bu fikir genel yapay zekânın doğal bir sonucudur.

Süper yapay zekâ bazı temel özellikler olarak, düşünme, akıl yürütme, bulmaca çözme, muhakeme, planlama, öğrenme ve kendi başına iletişim kurma gibi yetenekleri içermelidir. Süper yapay zekalar, halen varsayımsal bir kavram olmakla birlikte bu tür sistemlerin gerçekleştirilmesi, hızla değişen ve gelişen çağımızda temel bir görevdir.

2.1.3.2 Yapay Zekâ Tip-2: İşlevselliğine göre

Reaktif Makineler:

Tamamen reaktif makineler, yapay zekânın en temel türleridir. Bu tür sistemler, gelecekteki eylemler için anı ya da geçmiş deneyimleri biriktirmezler.

Bu makineler sadece mevcut olasılıklara odaklanır ve olası en iyi eyleme göre hareket ederler. Google'ın AlphaGo ve IBM'in Deep Blue sistemi bu tip makinelere bir örnektir.

Sınırlı Bellek:

Sınırlı bellek makineleri, geçmiş deneyimleri ya da bazı verileri kısa süreliğine saklayabilir. Bu makineler, sadece sınırlı bir süre depolanan verileri kullanabilir.

Kendi kendini süren arabalar, bu sistemlerin en iyi örneklerinden biridir. Bu arabalar, yakındaki araçların hızını, mesafesini, hız sınırını ve diğer bazı bilgileri saklayabilir.

Akıl Teorisi:

Bu tür sistemler insan duyguları, inançları gibi konuları anlamalı ve insanlar gibi sosyal olarak etkileşime girebilmelidir. Bu tür yapay zekâ makineleri hala geliştirilmemiş olup, araştırılmaya devam etmektedir.

Öz Farkındalık:

Öz farkındalık yapay zekâ, gerçekte mevcut olmayıp halen varsayımsal bir kavramdır ve yapay zekânın geleceği olarak da nitelendirilmektedir. Süper zeki olacak ve kendi bilincine, duygularına ve öz farkındalığına sahip olacak bu sistemler, insandan daha zeki olacak şekilde tasarlanmaya çalışılmaktadır.

2.2 Makine Öğrenmesi

Makine öğrenmesi (öğrenimi) en kısa anlatımla bilgisayarların olayları öğrenmesini sağlayan teknolojidir (Öztemel, 2003). Sistemleri açıkça programlamadan otomatik olarak öğrenmesini ve daha önce edindiği deneyimlere bakarak sonradan karşılaştığı durumlarda tahmin yeteneğini sağlayan yapay zekânın bir alt koludur.

Bilgisayar bilimcisi ve makine öğrenmesinin öncüsü Tom M.Mitchell, kitabında, makine öğrenmesinin yapay zekânın bir dalı olduğunu belirtmektedir (Mitchell, 1997).

Makine öğrenmesi yöntemleri, geçmişteki veriyi kullanarak yeni veri için en uygun modeli bulmaya çalışır (Diri, 2014). Makine öğrenmesine, insana özgü olan öğrenme yeteneğinin makinelere uyarlanmaya çalışılma çabası da denebilir.

1-2 Ekim 2015 tarihlerindeki "Siber Fiziksel Sistemler için Makine Öğrenmesi - Machine Learning for Cyber Physical Systems", makine öğrenmesi üzerine düzenlenen ilk konferans olma özelliğini taşımaktadır. Bu konferansta hakkında karar alınan konu başlıkları şöyledir (URL-6, 2019);

- Siber fiziksel sistemler için makine öğrenmesi algoritmaları
- Gerçek zamanlı etkin algoritmalar, sınıflandırılmış öğrenme yaklaşımları, çevrimiçi öğrenme ve ön işleme
- Siber fiziksel sistemler için öğrenilebilir biçimsel modelleme
- Sensör tümleştirme metotları, dağınık sistemlerde veri toplama ve veri bütünleşmesi

- Durum görüntüleme, pratik bakım, görüntü işleme, teşhis, optimizasyon ve kendini tamir uygulamaları.

2.2.1 Öğrenme Stratejileri

Makine öğrenmesi stratejileri temelde, Öğretmenli Öğrenme (supervised learning) ve Öğretmensiz Öğrenme (unsupervised learning) ve Takviyeli (Yarı Öğretmenli) Öğrenme olarak üç başlık altına toplanabilmekte olup bunlar Bölüm 2.9'da detaylı biçimde anlatılmaktadır.

Aşırı Eğitim (Overfitting):

Veri ile iterasyonal eğitim yapan modellerin hepsinde öğrenme sürecinin zamanı gelince durdurulması gerekir. Eğitim durdurulmazsa öğrenilmesi gereken veri içerisindeki tüm örnekler sistem tarafından ezberlenir ve bilinmeyen örnekleri tahmin yeteneği azalır. Aşırı eğitim (overfitting) denilen bu istenmeyen durum makine öğrenmesinin temel amacı olan genelleştirme kavramına zarar verir (Orhan, 2018).

2.2.2 Kullanım Alanları

Her geçen gün kendini yenileyen teknolojiyle birlikte, üretim sektörü de bu gelişime ayak uydurarak hem daha fazla pazar bulup hem de sektördeki rakiplerinin önüne geçmeye çalışmak durumundadır.

Büyük miktarda veriyle çalışan birçok endüstri, bu verilerden gelen bilgileri toplayarak, daha verimli çalışabilir veya rakiplere göre bir avantaj kazanabilir. Bunu da makinelerin gücünü kullanarak daha efektif şekilde yapabilmektedirler. Bu teknolojiyi kullanan sektörlerin en bilinenleri şunlardır:

- **Otomotiv**

Halen geliştirilmeye devam eden kendini süren arabalarda makine öğrenmesi kullanılmaktadır.

- **Finansal hizmetler**

Finansal sektördeki bankalar ve diğer işletmeler, makine öğrenimi teknolojisini iki temel amaç için kullanırlar;

- Verideki önemli bilgileri tanımlamak
- Sahtekârlığı önlemek

Makine öğrenmesinin sağladığı öngörüyle, yatırım fırsatlarını belirleyebilir veya yatırımcıların ne zaman yatırım yapacağını bilmelerine yardımcı olabilir. Veri madenciliği, yüksek riskli profillere sahip müşterileri de tanımlayabilir veya siber gözetimi kullanarak muhtemel bir sahtekârlığın önüne geçebilir.

- **Petrol ve Gaz Sektörü**

  Makine öğrenmesinin bu sektördeki kullanımının örnekleri şunlardır;
  - Yeni enerji kaynakları bulmak
  - Yerdeki minerallerin analizi
  - Rafineri sensör arızasını tahmin etmek
  - Daha verimli ve düşük maliyetli hale getirmek için petrol dağıtımını kolaylaştırmak
- **Perakende Pazarlama Sektörü**

Önceki satın alımlara dayanarak beğenebileceğiniz ürünleri öneren elektronik ticaret siteleri, satın alma geçmişinizi analiz etmek için makine öğrenmesini kullanmaktadır. Perakendeciler veri toplamak, analiz etmek ve alışveriş deneyimini kişiselleştirip, kişiye özel pazarlama kampanyası uygulamak, fiyat optimizasyonu, ürün tedarik planlaması ve müşteri görüşleri için hâlihazırda makine öğrenmesine güvenmektedirler.

- **Taşımacılık/Lojistik Sektörü**

Makine öğrenmesi, modelleri ve eğilimleri belirlemek için verileri analiz etmek, rotaları daha verimli hale getirmek ve karlılığı artırmak için olası sorunları öngörmeye dayanan ulaştırma sektörünün çok önemli bir parçası halindedir. Makine öğrenmesinin veri analizi ve modellenmesi, lojistik şirketleri, toplu taşıma ve diğer ulaşım firmaları için şu an çok önemli bir pozisyondadır. Gönderilerin üzerindeki el yazısını okuyup gönderi tasnifi yapan sistemler buna örnek olarak gösterilebilir.

- **Resmi Kurumlar**

Kamu güvenliği ve kamu hizmetleri gibi faaliyetleri gören devlet kurumları, öngörü için kullanılabilecek çok sayıda veri kaynağına sahip olduklarından, makine öğrenmesine özel bir ihtiyaç duymaktadır. Örneğin sensör verilerini analiz ederek,

verimliliği artırmanın ve maliyetten tasarruf etmenin yollarını ararlar. Makine öğrenmesi ayrıca dolandırıcılığı tespit etmeye ve kimlik hırsızlığını en aza indirmeye de yardımcı olabilir.

- **Sağlık Sektörü**

Makine öğrenmesi, bir hastanın sağlığını gerçek zamanlı olarak değerlendirmek için verileri kullanabilen giyilebilir cihazlar, sensörler ve kameralar sayesinde sağlık sektöründe hızlı büyüyen bir ivme göstermektedir. Makine öğrenmesi aynı zamanda doktorların verileri analiz etmesine yardımcı olarak, daha gelişmiş teşhis ve tedavi imkânı sunmaktadır. Yüz ifadesinden ağrı tespiti de makine öğrenmesinin bu bahsedilen faydalarından biri olarak gösterilebilir.

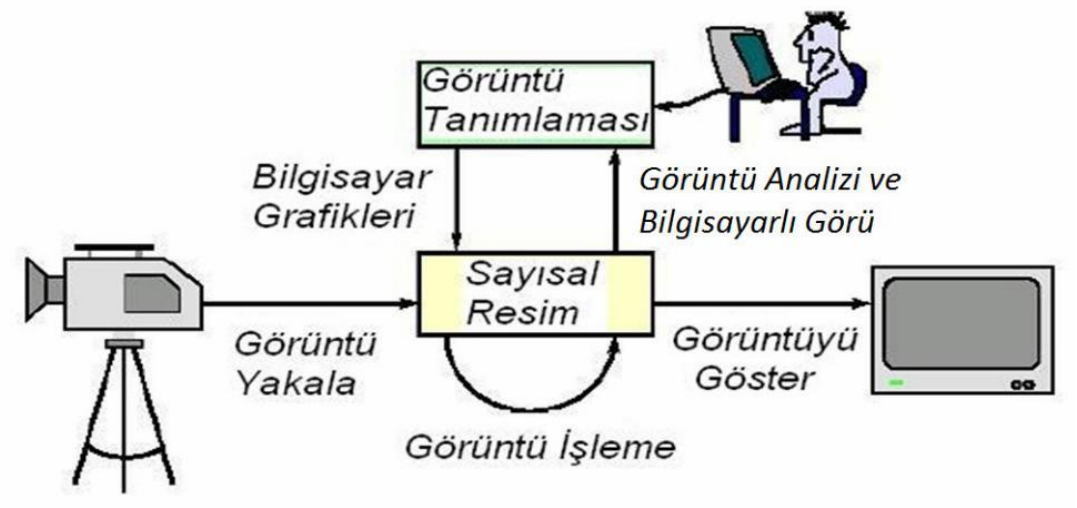
2.3 Bilgisayarlı Görme

Bilgisayarlı görme (Computer Vision - CV), bilgisayarların fotoğraf ve video gibi dijital görüntülerin içeriğini görmesine ve anlamasına yardımcı olacak teknikler geliştirmeye çalışan yapay zekânın bir alt dalı olarak tanımlanabilmektedir.

Şekil 2.15'de blok diyagramı görülen bilgisayarlı görmenin diğer bir tanımı şu şekildedir; Bilgisayarlı görme, çevreyle herhangi bir fiziksel etkileşim olmaksızın, optik araçlar kullanarak cisimleri algılama ve buna göre bilgi toplama ya da çeşitli süreçleri yönetmektir (Güvenoğlu, 2017).

Adrian Kaehler ve Gary Brandski'ye göre ise "hareketsiz veya video kameradan alınan verinin bir karar veya yeni bir temsile dönüştürülmesidir." (Szeliski, 2010). Ayrıca şu tanımlara rastlamak mümkündür;

- Bir veya daha fazla dijital görüntüden üç boyutlu dünyanın özelliklerini hesaplamaktır.
- Görüntülerden fiziksel nesnelerin kapalı ve anlamlı tanımlarını oluşturmaktır.
- Resimlerden veya resim dizilerinden dünyanın betimlemelerini çıkartmaktır.
- Szeliski'ye göre; bilgisayarlı görme ile bir veya daha fazla görüntüde görülen dünyayı tanımlamaya ve şekil, aydınlatma ve renk dağılımları gibi özelliklerini yeniden oluşturmaya çalışılmaktadır (Szeliski, 2010).



Şekil 2. 2 Bilgisayarlı görme işleminin blok diyagramı.

2.3.1 Tarihçesi:

Bilgisayarlı görme ilk olarak 1950'lerde, bir nesnenin kenarlarını tespit etmek ve daireler ve kareler gibi basit nesneleri kategorilere ayırmak için ilk sinir ağlarından bazılarını kullanarak gerçekleşmiştir. Bilgisayarlı görmenin babası kabul edilen Larry Roberts, 1963'te yayınladığı "Machine Perception Of Three-Dimensional Solids (Üç Boyutlu Katıların Makine Algısı)", adlı makalesinde ilk defa 2-boyutlu görüntülerden yola çıkarak 3-boyutlu cisimleri bulmaktan bahsetmiştir (Roberts, 1963).

1970'lerde, bilgisayarlı görmenin ilk ticari kullanımı ile optik karakter tanıma kullanarak, yazılan yazılar veya el yazısı metinler yorumlanmıştır. Bu gelişme daha sonra, görme engelliler için yazılı metni yorumlamak amacıyla kullanılmıştır.

İnternet 1990'larda olgunlaştıkça, analiz için çevrimiçi olarak büyük görüntü setleri hazırlarken, Yüz Tanıma programları da gelişmiştir. Bu büyüyen veri setleri, makinelerin fotoğraf ve videolardaki belirli kişileri tanımlamasını mümkün kılmıştır.

2.3.2 Bilgisayarlı Görmenin Basamakları

Düşük seviye: Genel olarak üç ayrı basamakta görüntülerden görsel bilgi edinilmektedir. İlk basamakta çeşitli görüntü işleme teknikleriyle, resimden bilgileri daha kolay çıkarmamızı sağlayacak başka bir resim elde edilir. Keskinleştirme, dağıtma,

köşelerini elde etme, eşik değer uygulama, histogramını dağıtma, vb. işlemler uygulanarak bu resimler ikinci evreye hazırlanır.

Orta seviye: Bu aşamada resmin karakteristik özellikleri çıkarılır. Farklı bölgelerin renkleri, çembersel yaylar ve çizgiler, birleşik elemanlar gibi resmin öğeleri çıktı olarak verilirken, bir yandan da bu karakteristikler veri yapılarında tutulur.

Yüksek seviye: Bu seviyede elde edilen özelliklerden görüntünün içindeki nesneler bulunabilir ve başta belirtildiği gibi bilgisayarın verdiği "kullanışlı kararlar" çıktılanabilir.

2.3.3 Bilgisayarlı Görme Alt Dalları

- Indexleme
- Görüntü Restorasyonu (Image Reconstruction)
- Olay Algılamala (Event Detection)
- İzleme (Detection)
- Nesne Tanımlama (Object Recognition)
- Öğrenme (Learning)
- Görüntü İşleme (Image Processing)

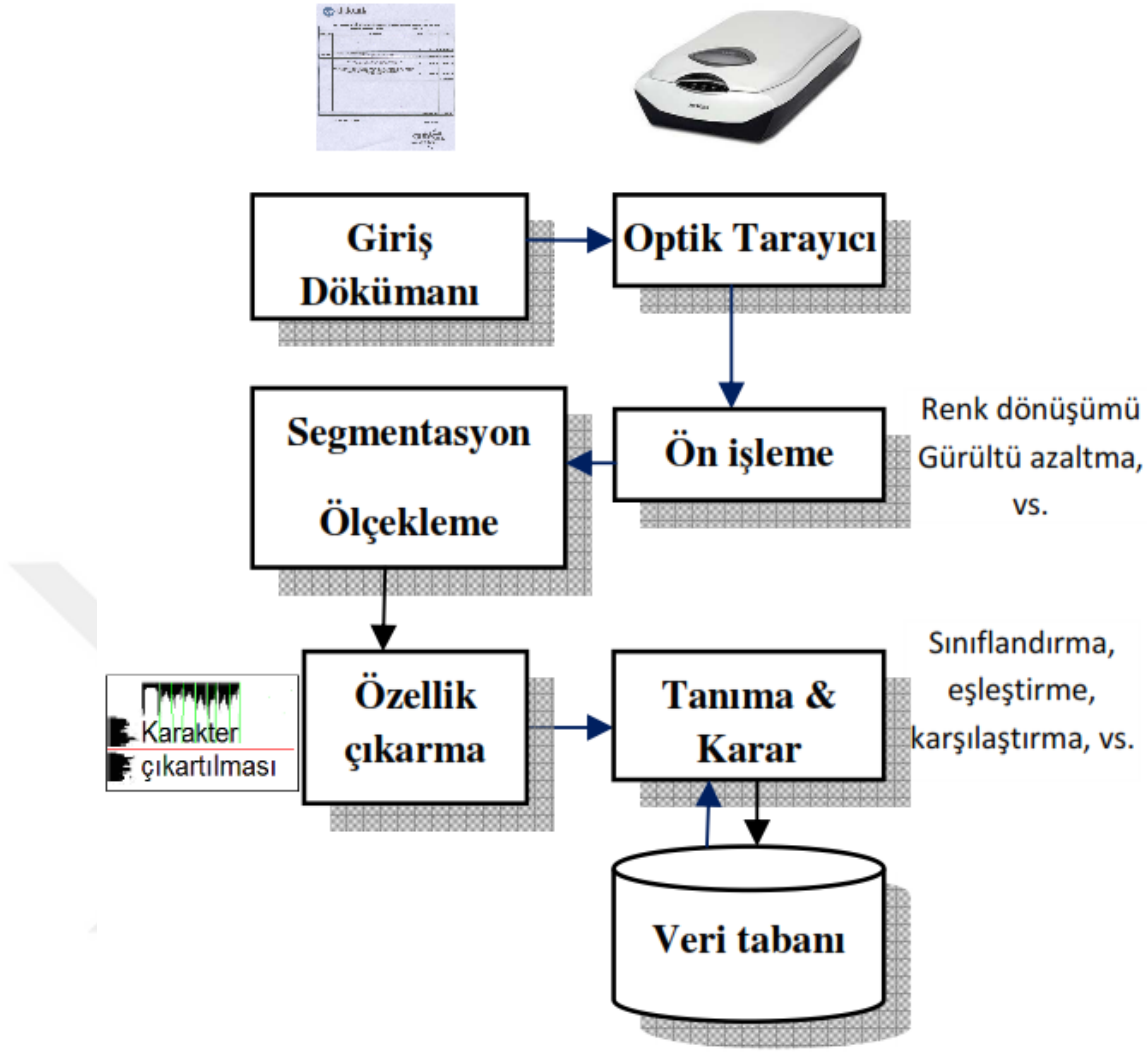
2.3.4 Kullanım Alanları:

Bilgisayarlı görmenin günümüzde yaygın olarak kullanıldığı pek çok alan vardır. Bunlar;

Optik Karakter Tanıma Sistemleri: Elle ya da elektronik olarak yazılmış görüntüler üzerindeki metin bilgilerinin okunarak ASCII koduna dönüştürülmesi işlemidir. Yani, bilgisayar ortamında bulunmayan yazılı dokümanların özel tarayıcılar veya normal olarak taranmış resimlerin, özel programlar arayıcılığıyla bilgisayar ortamında düzenlenebilecek durumda aktarılması işlemidir. Bu alanda, 1950'de David Shepard tarafından geliştirilen, optik karakter tanıma sistemi olan GISMO, Amerika'da geliştirilmiş ilk çalışma sayılmaktadır (şeil 2.16).

Perakende satış: Otomatik ödeme sistemleri için nesne tanıma işlemlerinde bilgisayarlı görme kullanılmaktadır.

Makine kontrolü: İnsan hatalarını ortadan kaldırmak ve üretimde performansı artırmak için, endüstride kalite kontrol aşamasında sayısal görüntüleme sistemleri kullanılmaktadır.



Şekil 2. 3 Optik karakter tanıma şeması.

Medikal Görüntüleme: Görüntüleme aygıtları ile elde edilen tıbbi görüntü veya video verilerinin görüntü işleme teknikleri kullanılarak işlenmesi ve bilgi elde edilmesi işlemidir. Medikal görüntüleme sistemlerinden elde edilen görüntüler ve görüntü işleme teknikleri kullanılarak çıkarılan bilgiler, tıbbi uzmanların konu hakkında karar vermesine yardımcı olmaktadır.

3-Boyutlu model oluşturma (Fotogrametri): Google ve Yandex haritalar gibi sistemlerde, kullanılan hava fotoğraflarından 3B modellerin tamamen otomatik olarak oluşturulmasıdır.

Güvenlik: Güvenlikte kullanılan makine ile görme sistemlerinin standart güvenlik kamerası sistemlerinden farkı, kayıtların arşivsel bir tespit aracı olmasından

ziyade görüntünün sürekli bir algoritma tarafından izlenerek, anında gerekli müdahaleyi mümkün kılmasıdır. Örneğin, girilmesi yasak bir alanda hareket algılanması durumunda ihlalin bildirilmesi (örneğin alarm çalınması) durumu. Standart kamera ile, bu bahsedilen ihlal, yetkili kimsenin olay anında ihlali gözlemlemesi ya da daha sonra arşivi incelemesiyle anlaşılabilir. Fakat bilgisayarlı görme algoritması ile uyarı sisteminin otomatik olarak harekete geçirilmesi mümkün olabilmektedir.

Hareket Algılama: Güvenlik konusunda da bahsedildiği gibi hareket algılamada, bilgisayarlı görme algoritmaları kullanılarak tespit etmek mümkün olmaktadır.

Parmak İzi Tanıma: Yüz tanıma gibi parmak izi tanıma da güvenlik noktasında, telefon şifresinden, işyeri giriş çıkış kontrolüne kadar birçok yerde karşımıza çıkmaktadır.

Yüz Tanıma: Son zamanlarda telefon, tablet ve bilgisayarlarımızda da sıkça görmüş olduğumuz yüz tanıma sistemi, güvenlik amacıyla başta hava alanları olmak üzere toplu taşımada güvenlik amaçlı olarak kullanılmaya başlanmış bulunmaktadır. Bazı ülkelerde (örneğin Almanya) yurda girişlerde pasaport kontrol noktalarında da yine yüz tanıma kullanılmakta olup, ülkeye girişlerde büyük kolaylık sağlanmaktadır.

2.4 Yapay Sinir Ağları

Yapay Sinir Ağları, insan beyninden esinlenerek geliştirilmiş, ağırlıklı bağlantılar aracılığıyla birbirine bağlanan ve her biri kendi belleğine sahip işlem elemanlarından oluşan paralel ve dağıtılmış bilgi işleme yapılarıdır. İnsan beyninin özelliklerinden olan öğrenme yolu ile yeni bilgiler türetebilme ve keşfedebilme gibi yetenekleri, herhangi bir yardım almadan otomatik olarak gerçekleştirmek amacı ile geliştirilen bilgisayar sistemleridir (Öztemel, 2003).

Yapay sinir ağlarının çalışmasını daha iyi anlamak için biyolojik sinir hücresinin çalışma sistemini bilmek gerekmektedir.

2.4.1 Biyolojik Sinir Ağı

Biyolojik sinir ağları milyarlarca sinir hücresinin bir araya gelmesi ile oluşmuş bir sinir hücresi topluluğudur. Bir insanın beynindeki sinir hücresi (nöron) sayısı birçok kaynakta farklı söylenmekle birlikte Carter'a göre bu sayı 50 milyon ile 500 milyon

olarak belirtilmiştir (URL-7). Öztemel'e göre aynı sayı 10^{10} adettir ve bunların da $6x10^{13}$'ten fazla sayıda bağlantısının olduğunu savunmaktadır (Öztemel, 2003). Elmas ise bu sayının yaklaşık 10^{11} olduğunu, her bir sinir hücresinin de 10^4 bağlantısının olduğunu savunur (Elmas, 2018).

Bir sinir hücresi sinaps, soma, akson ve dendrit olmak üzere dört temel yapıdan oluşur. Sinir hücreleri bir kaynaktan aldığı girişi dendritten aksona doğru, oradan da sinaps denilen boşluklar yoluyla diğer bir sinir hücresine gönderir. Dendrit uçlarındaki kimyasal değişim elektriksel uyarıya dönüşür, oluşan elektriksel uyarı belli bir eşiği aşınca akson üzerinden sinapslara doğru iletilir. Sinapslara gelen elektriksel potansiyel, kimyasal uyarıya dönüşerek bir sonraki sinir hücresinin akson uçlarının uyarılmasını sağlar. Bu olay bütün iletim süreci boyunca aynı şekilde devam eder.

Nöron: Biyolojik sinir hücreleri olan nöronlar, beyin ve sinir sisteminin temel birimleridir. Dendritler yoluyla girdiyi alır, işler ve aksonlar aracılığıyla çıktı verirler.

Hücre gövdesi (Soma): Nöron hücresinin gövdesi çekirdeği içerir ve nöronların yaşamı için gerekli biyokimyasal dönüşümü gerçekleştirir.

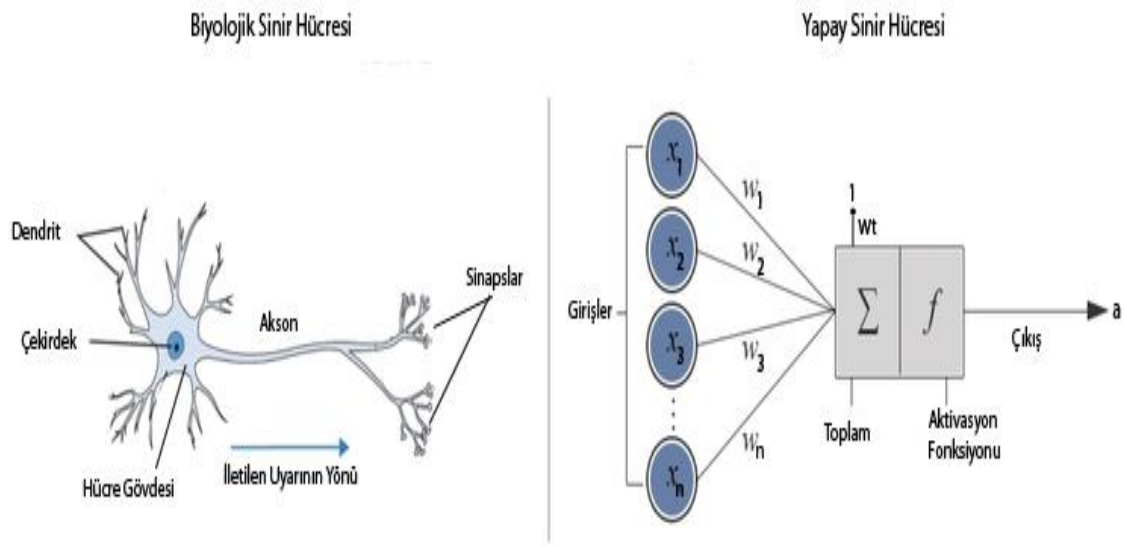
Dendrit: Her nöronun etrafında ince, saç benzeri uzantılar vardır. Bunlar hücre gövdesinin etrafındaki bir ağaç gibi dallanır ve gelen sinyalleri kabul ederler.

Akson: Bir elektrik iletim hattı gibi çalışan uzun, ince, boru şeklindeki bir yapıdır.

Sinaps: Aksonların sonunda iki nöron arasındaki bağlantının gerçekleştiği son derece karmaşık ve özel yapılardır.

2.4.2 Yapay Sinir Ağı (YSA)

Şekil 2.17'de görüldüğü gibi yapay sinir ağları, biyolojik sinir ağlarını taklit eden bilgisayar programlarıdır [10]. İnsan beyninin yaptığı bazı işlemlere benzer olarak öğrenme, ilişkilendirme, sınıflandırma, genelleme, özellik belirleme ve optimizasyon gibi konularda başarılı performans sergilemektedirler. Tablo 2.4'de yapay sinir ağı ile biyolojik sinir ağı karşılaştırılmıştır.



Şekil 2. 4 Biyolojik ve yapay sinir hücreleri.

Tablo 2. 2 Biyolojik ve yapay sinir ağlarının karşılaştırılması (Elmas, 2018).

Biyolojik Sinir Ağı	**Yapay Sinir Ağı**
Sinir Sistemi	Sinirsel Hesaplama Sistemi
Sinir	Düğüm (yapay sinir)
Sinaps	Sinirler arası bağlantı ağırlıkları
Dendrit	Toplama İşlevi
Hücre Gövdesi	Etkinlik (Aktivasyon) İşlevi
Akson	Sinir Çıkışı

Yapay sinir ağlarının genel özelliklerini şu şekilde sıralamak mümkündür;

- Öğrenebilen yapılardır.
- Genelleme yapabilirler.

- Bilgiyi bir veri tabanında saklamak yerine ağın bütününe dağıtılmış şekilde saklarlar.
- Hata tolere etme yönleri gelişmiştir.
- Eksik bilgiyle çalışabilirler.
- Yapay Sinir ağında bir bilginin kaybolması birdenbire değil, kademeli bozulma gösterir.
- Görmedikleri örnekler hakkında çözüm önerisinde bulunabilirler.
- Örüntü tanıma ve sınıflandırma yapabilirler.

Yapay sinir ağlarının avantajları yanında bazı dezavantajları da mevcuttur. Bunları şu şekilde sıralamak mümkündür;

- Sadece sayısal bilgiyle çalıştıkları için girişin her zaman sayısal forma dönüştürülmesi gerekmektedir.
- Her problem için ayrı bir ağ tasarlanması, parametrelerin probleme göre belirlenmesi gerekmektedir.
- Donanım bağımlıdırlar.
- Klasik programlara göre daha yavaş çalışırlar.
- Ağın performansı, eğitim setine bağlıdır.
- Üretilen çözümün nasıl bulunduğuna dair soruları yanıtsız bırakır.
- Eğitim aşamasının durdurulması konusunda net bir kural yoktur.
- Ağı tasarlayan kişinin tecrübeli olmasının ağın performansına etkisi büyüktür.

2.4.3 Yapay Sinir Ağının Tarihçesi

İlk yapay sinir ağ modeli 1943 yılında, bir tıp doktoru olan Warren McCulloch ile matematikçi Walter Pitts tarafından geçekleştirilmiştir. McCulloch ve Pitts, insan beyninin hesaplama yeteneğinden esinlenerek elektrik devreleriyle basit bir sinir ağını modellemişlerdir.

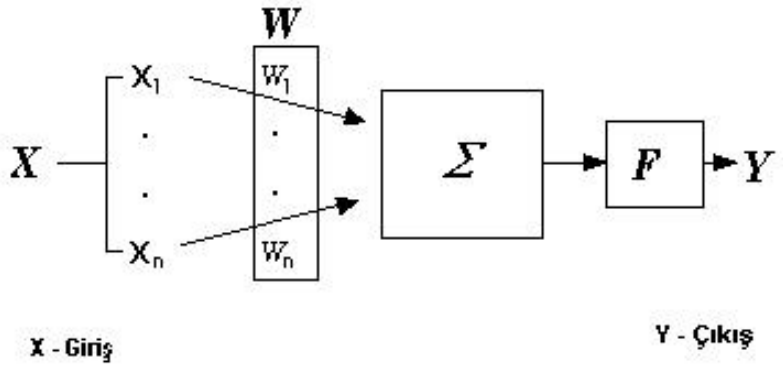
1948 yılında Wienre “Cybernetics” isimli kitabında, sinirlerin çalışması ve davranış özelliklerine değinmiş, 1949’da ise Hebb “Organization of Behavior” isimli kitabında, öğrenme ile ilgili temel teoriyi ele almıştır.

1949 yılında Donald Hebb kitabında sinir ağları modeli için temel olacak Hebbian Öğrenme Kuralı’nı ortaya koymuştur. Hebbian Öğrenme Kuralı, sinir ağının

bağlantı sayısı değiştirilebilirse, öğrenebileceğini öngörmekteydi. 1950'li yıllardan sonra birçok araştırmacı Hebb Kuralı'ndan esinlenerek yapay sinir ağlarının hesaplama gücünü artırıcı yönde çalışmalar yapmıştır. IBM araştırma laboratuvarlarında yapılan bir sinir ağı benzetimi (simülasyon) çalışmalarının başarısızlıkla sonuçlanmasına karşın, sonraki girişimlerde başarı sağlanmıştır. 1951 yılında ilk kez nörobilgisayar üretilmiştir.

1954 yılında Farley and Clark tarafından rassal ağlar (random networks) ile adaptif tepki üretme kavramı ortaya atılmıştır.

1957 yılında Frank Rosentblatt'ın yapay sinir ağlarında devrim sayılabilecek Algılayıcının (perceptron) gerçekleştirilmesinden sonra yapay sinir ağı alanındaki gelişmeler hızlanmıştır. Şekil 2.18'de görüldüğü gibi algılayıcı, beyin işlevlerini modelleyebilmek amacıyla yapılan çalışmalar neticesinde ortaya çıkan tek katmanlı eğitilebilen ve tek çıkışa sahip olan yapay sinir ağıdır.



Şekil 2. 5 Algılayıcı (Perceptron).

Bu çalışmalara yardımcı olmak için, John Von Neuman vakum tüplerini ve telgraf rölelerini kullanarak basit sinir işlevlerini taklit etmeyi önermiştir. Frank Rosentblatt ise anlama ve kavrama hakkında çalışmalara başlamış ve bir sineğin göz işlemleriyle (çalışmasıyla) ilgilenmiştir. Bir sineğe kaçmasını söyleyen işlemlerin çoğunun sineğin beyni yerine gözünün içinde gerçekleşmesi, **kavrama düğümü** diye adlandırılan ağ yapısının kurulmasını sağlamıştır. Tek katmanlı bir kavrama düğümünde, giriş değerleri ağırlıklı toplamı hesaplanarak, bir eşik değeri çıkarılmakta ve iki olası değerden biri sonuç olarak alınmaktadır. Kavrama düğümü bugün kullanımda olan en eski sinir ağıdır.

1890'da insan beyninin yapısı ve fonksiyonları ile ilgili ilk yayının yazılması, yapay sinir ağları ile ilgili yapılan çalışmalara temel teşkil etmesi açısından önemlidir. Bu çalışmayı 1911'de insan beyninin sinir hücrelerinden oluştuğu fikrinin kabul edilmesi izlemiştir (Öztemel, 2003).

1959 yılında, Bernand Widrow ve Marcian Hoff (Stanford Üniversitesinde) ADALINE ve MADALINE diye adlandırdıkları ağ modellerini geliştirdiler. MADALINE, telefon hatlarında oluşan Yankıları yok eden bir uyarlanabilir süzgeç olarak kullanılmış, gerçek dünya sorunlarına uygulanmış olan ilk sinir ağıdır ve hala kullanımda bulunmaktadır. Bu dönemde en dikkat çekici çalışma Widrow'un 1963 yılında yaptığı 'ters sarkaç' denetleyicisidir.

1969 yılında Minsky ve Papert Algılayıcının yetersizliğini görmüşler ve XOR problemini çözemediğini ispatlamışlardır. Bunun için iki katmanlı ileri beslemeli ağların kullanılabileceğini ileri sürmüşler ve tek katmanlı ağlardaki birçok sınırlamayı ortadan kaldırdığını göstermişlerdir. Fakat gizli katmanların ağırlıklarının nasıl değiştirileceği konusunda bir yöntem önerememişlerdir. Bu soruna daha sonra Rumelhart ve arkadaşları, geri yayılım yöntemi ile çözüm getirmişlerdir.

Bu erken başarılar, insanların sinir ağlarının potansiyelini abartması sonucunu doğurmuştur. Abartılı beklentilerin gerçekleştirilmemesi bu beklentileri hayal kırıklığına dönüştürmüş, aynı zamanda bazı bilim kurgu yazarlarının, 'düşünen makinelerin' insanları nasıl etkileyeceğini ele almalarına yol açmıştır. Asimov robotlar hakkındaki kitaplarında, düşünen makinelerin, özellikle insanoğlunun ahlaki değerlerine olan etkilerini konu almıştır. Diğer bazı yazarlar ise çok daha korkunç bilgisayarların yaratılacağı hakkında abartılı kitaplar yazmışlardır. Bu yersiz ve abartılı iddialar, yapay sinir ağları araştırmalarının kesilmesine yol açmakla birlikte bu ilginç gelişme 1981 yılında sonra ermiştir.

Tüm bunlara rağmen, Anderson, Wilshaw, Kanerva ve Kohonen gibi az sayıdaki sinir ağının savunucuları sessiz araştırmalarına devam etmişlerdir. Son yıllarda yapılan çalışmalarla, yapay sinir ağları her alanda yaygın olarak kullanılmaya ve başarılı sonuçlar alınmaya başlanmıştır.

Yapay sinir ağı çalışmaları 80'li yıllarda hızlı bir yükselişe geçmiştir. Hopfield 1982 yılında ağların önemli sınıflarının matematik temellerini üretmiştir. John Hopfield beyne benzeyen basit bir model değil, kullanışlı bir alet yaratma yaklaşımıyla Ulusal

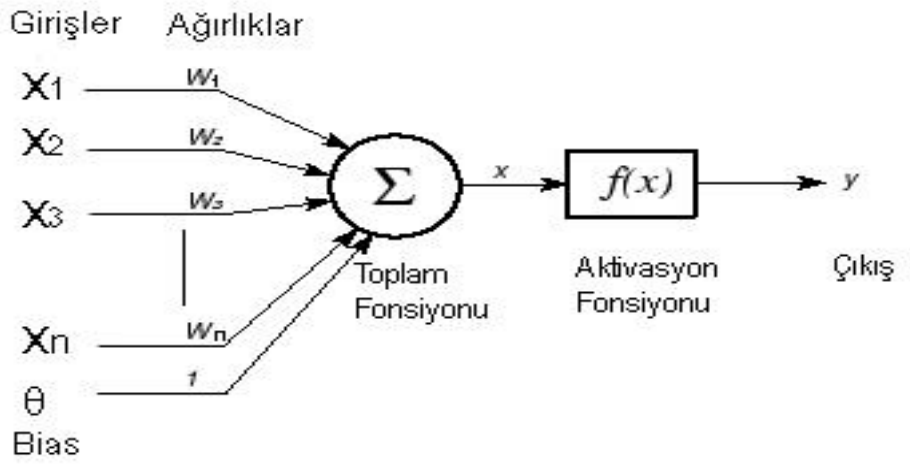
Bilimler Akademisine bir makale sunmuş, açık ve matematiksel analizleri kullanarak, böyle bir aletin nasıl çalışabileceğini ve ne yapabileceğini göstermiştir.

1984'te Kohonen sinirlerin düzenli sıralanışına eşleme özelliği için gözetimsiz öğrenme ağlarını geliştirmiştir. 1985 yılına kadar, Amerikan Ulusal Fizik Akademisi (National Academy of Sciences of the USA), yapay sinir ağları ile ilgili gelişmeleri izlemiş ve desteklemiştir. 1986'da Rumelhart ve McClelland karmaşık ve çok katmanlı ağlar için Geriye Yayılmalı Öğrenme Algoritmasını ortaya koymuştur.

1987 yılında Elektrik Elektronik Mühendisliği Enstitüsü (Institute of Electrical Electronic Engineering (IEEE)) tarafından sinir ağlarını konu alan ilk uluslararası konferans 1800'ü aşkın katılımcıyla gerçekleştirilmiştir (Elmas, 2018). Tüm bu gelişmelerden sonra yapay sinir ağları ile yapılan çalışmalar günümüze kadar artan bir ivmeyle devam etmiş olup, derin arama uygulamalarının kullanımının yaygınlaşmasıyla birlikte artarak da devam etmektedir.

2.4.4 Bir Yapay Sinir Hücresinin Temel Elemanları

Bir yapay sinir ağı, paralel işlem yapan, birbirine bağlı çok sayıda düğüm ya da yapay sinir de denebilen işlem elemanlarından oluşmaktadır. Şekil 2.19'da yapay bir sinir hücresi gösterilmiştir.



Şekil 2. 6 Bir yapay sinir hücresi.

Şekil 2.19'da gösterilen bazı kavramların açıklaması şu şekildedir;

Girişler (X1, X2, …,Xn): Yapay sinir hücresine (düğüm de denmektedir) dışarıdan ya da kendisinden önceki düğümden gelen sayısal bilgilerdir.

Bias (θ): Bir düğümün yerel optimum noktalarına takılıp yanlış sonuç üretmemesi için girişlerle birlikte toplam fonksiyonuna girilen sayısal bir değerdir.

Ağırlıklar (W1, W2,…,Wn): Bir düğüme giren değerlerin ağdaki etkinliğini sağlayan değerdir. Başlangıçta rastgele değerler atanan bu değişkenler, ağın öğrenmesi sırasında negatif ya da pozitif değer alabilirler.

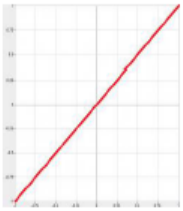
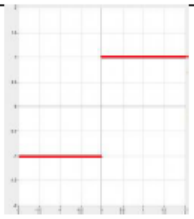
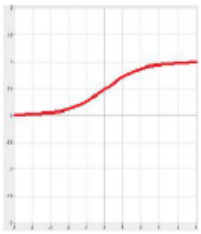
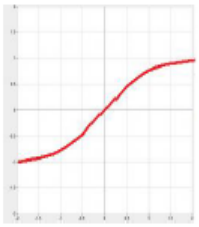
Toplam Fonksiyonu (∑): Bu fonksiyon, bir düğümdeki net girdiyi hesaplar. Bunu hesaplarken birçok farklı fonksiyon kullanılmakla birlikte yaygın olan girdiler ile ilgili ağırlık değerlerini çarparak, çıkan sonuçların eşik değeriyle toplanmasıyla bulunur. Yapay sinir ağlarında kullanılan toplam fonksiyonları Tablo 2.5'de gösterilmektedir

Tablo 2. 3 Toplam Fonksiyonları (Çayıroğlu, 2019).

Toplam $\text{Net} = \sum_{i=1}^{N} X_i * W_i$	Ağırlık değerleri girdiler ile çarpılır ve bulunan değerler birbirleriyle toplanarak Net girdi hesaplanır.
Çarpım $\text{Net} = \prod_{i=1}^{N} X_i * W_i$	Ağırlık değerleri girdiler ile çarpılır ve daha sonra bulunan değerler birbirleriyle çarpılarak Net Girdi Hesaplanır.
Maksimum $\text{Net} = \text{Max}(X_i * W_i)$	n adet girdi içinden ağırlıklar girdilerle çarpıldıktan sonra içlerinden en büyüğü Net girdi olarak kabul edilir.
Minimum $\text{Net} = \text{Min}(X_i * W_i)$	n adet girdi içinden ağırlıklar girdilerle çarpıldıktan sonra içlerinden en küçüğü Net girdi olarak kabul edilir.
Çoğunluk $\text{Net} = \sum_{i=1}^{N} Sgn(X_i * W_i)$	n adet girdi içinden girdilerle ağırlıklar çarpıldıktan sonra pozitif ile negatif olanların sayısı bulunur. Büyük olan sayı hücrenin net girdisi olarak kabul edilir.
Kumilatif Toplam $\text{Net} = \text{Net(eski)} + \sum_{i=1}^{N} X_i * W_i$	Hücreye gelen bilgiler ağırlıklı olarak toplanır. Daha önce hücreye gelen bilgilere yeni hesaplanan girdi değerleri eklenerek hücrenin net girdisi hesaplanır.

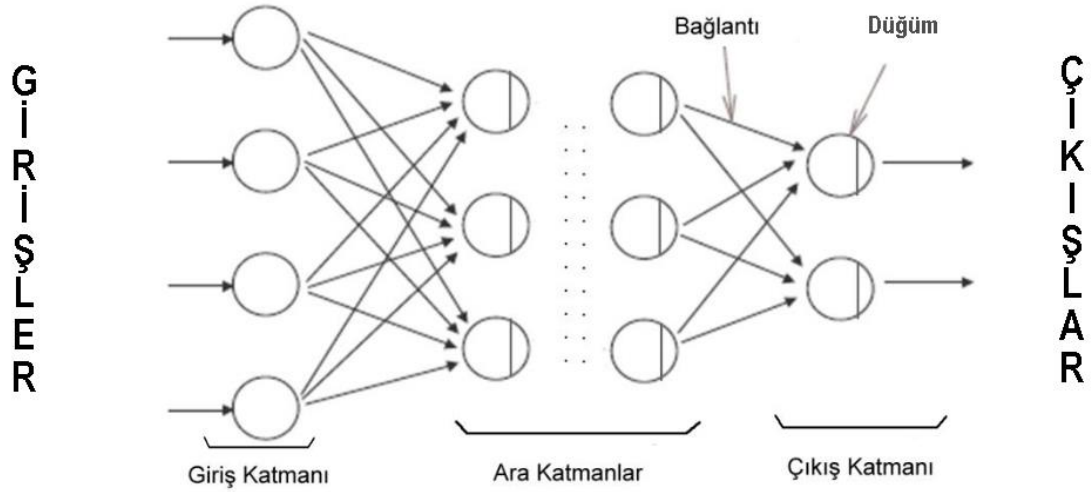
Aktivasyon Fonksiyonu (f(x)): Toplam fonksiyonundan gelen girdiyi işleyerek bir çıkış üretir. Bu çıktı, düğümün bulunduğu konum itibariyle giriş katmanı, ara katman ya da çıkış katmanında olmasına göre, ya bir sonraki düğüme veya direk çıkışa (y) gönderilir. Tablo 2.6'da görüldüğü gibi aktivasyon fonksiyonu olarak birçok seçenek mevcuttur. Bunlardan en yaygın olarak kullanılan sigmoid fonksiyonudur.

Tablo 2. 4 Aktivasyon Fonksiyonları (Çayıroğlu, 2019).

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.
Eşik Değer Fonksiyonu		$F(Net)=\begin{cases}0 \text{ if Net<=0}\\Net \text{ if } 0<Net<1\\1 \text{ if Net>= 1}\end{cases}$	Gelen bilgilerin 0 dan küçük-eşit olduğunda 0 çıktısı, 1 den büyük-eşit olduğunda 1 çıktısı, 0 ile 1 arasında olduğunda ise yine kendisini veren çıktılar üretilebilir.
Sinüs Aktivasyon Fonksiyonu		F(Net) = Sin(Net)	Öğrenilmesi düşünülen olayların sinüs fonksiyonuna uygun dağılım gösterdiği durumlarda kullanılır.

2.4.5 Yapay Sinir Ağının Yapısı:

Yapay sinir ağları, çok sayıdaki düğüm ve bu düğümleri birbirine bağlayan ağırlıklardan meydana gelir (Şekil 2.20). Yapay sinir ağına girilen sayısal bir bilgi, düğümler arası bağlantılar vasıtasıyla çıkışa kadar iletilir.



Şekil 2. 7 Yapay sinir ağının yapısı.

İlk yapay sinir ağının bulunmasından günümüze dek birçok yapay sinir ağı modeli ve bu modellerde kullanılacak birçok öğrenme stratejisi geliştirilmiştir. Düğümler arasındaki bilgi iletim yönüne göre; İleri Beslemeli (Feed Forward) ve Geri Beslemeli (Feed Backward) ya da Özyinelemeli (Feedback ya da Recurrent) iki temel ağ yapısından söz etmek mümkündür.

Girişler ağa giriş katmanından girer, ağın çıktısının verildiği katman olan çıkış katmanına doğru ileri yönlü yayılır. Geri yayılımlı ağlarda en az bir düğüm kendisinden önceki katmana bilgi gönderir.

Yapay sinir ağlarının bir diğer özelliği öğrenebilmeleridir. Bunu da belirlenen öğrenme stratejisine göre düğümler arasındaki bağlantı değerlerini değiştirerek yaparlar. Temelde, Öğretmenli Öğrenme (Supervised Learning), Öğretmensiz Öğrenme (Unsupervised Learning) ve Takviyeli Öğrenme (Reinforcement Learning) olmak üzere üç çeşit öğrenme stratejisi mevcuttur.

Öğretmenli öğrenmede, giriş setiyle birlikte çıkış seti ağa birlikte verilir ve ağın istenen çıkışı üretmesi beklenir. Bu tip ağlara örnek olarak Çok Katmanlı Algılayıcı veya ADALİN/MADALİN ağlarını vermek mümkündür.

Öğretmensiz öğrenmede, girişler ağa verilir, ağdan herhangi bir çıkış istenmez, ağ bu girilen nesneleri ortak özelliklerine göre gruplara ayırır. Çıkışta bu grupların etiketlenmesi gerekmektedir. Bu tip ağlara Hebbian Öğrenme, Yarışmacı Öğrenme (Competitive Learning) ya da Özörgütlemeli Harita Ağları (Self-Organizing Map - SOM) örnek verilebilir.

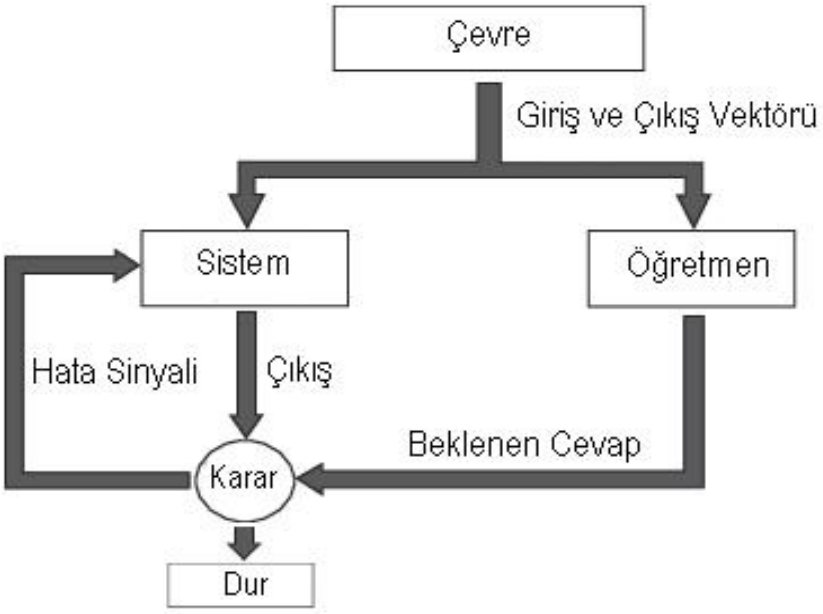
Takviyeli öğrenmede ise, giriş seti ağa verilir, ağın verdiği çıkış her seferinde öğretmene sorulur, öğretmen ağın doğru ya da yanlış yaptığını ağa bildirir. Burada ağ, bir nevi ödül-ceza sistemine göre çalışır. Bu tip ağlara LVQ (Linear Vector Quantization) ağları örnek verilebilir.

2.4.6 Öğrenme Stratejileri

Öğrenme stratejileri temelde 3 başlık altında toplanmıştır. Bunlar;

2.4.6.1 Öğretmenli Öğrenme (Supervised Learning):

Bu tip öğrenmede, ağın eğitimi esnasında, bir öğretmen tarafından sisteme bir girdi seti verilir ve bu girdiye karşılık çıktının ne olması gerektiği ağa gösterilir. Ağ girdi/çıktı setine göre, sistemdeki ağırlık değerlerini güncelleyerek öğrenme işlemini gerçekleştirir. Bu öğrenme türüne Çok Katmanlı Algılayıcı (Multilayer Perceptron), Geri Besleme (Backpropagation Algoritması), Delta Kuralı, Genelleştirilmiş Delta Kuralı veya En Küçük Karelerin Ortalaması (Least Mean Square) da denen Widrow-Hoff ve ADALIN 'i örnek vermek mümkündür. Şekil 2.21'de öğretmenli öğrenme stratejisinin blok diyagramı görülmektedir.



Şekil 2. 8 Öğretmenli öğrenme blok diyagramı.

2.4.6.2 Öğretmensiz Öğrenme (Unsupervised Learning):

Şekil 2.22'de de görüldüğü gibi bu öğrenme türünde sisteme verilen girdilere karşılık çıktının ne olacağını öğretecek bir öğretmen yoktur. Sistem girdilerdeki benzerliklere bakarak kendisi gruplar oluşturur. Bu işlem sonunda sistemin ürettiği çıktıların kullanıcı tarafından etiketlenmesi gerekmektedir. Bu öğrenme türüne Yarışmacı Öğrenme (Competitive Learning), Kohonen'in Özörgütlemeli Harita Ağları (Self-Organizing), Hebbian Öğrenme ve Grossberg Öğrenme'yi örnek vermek mümkündür.

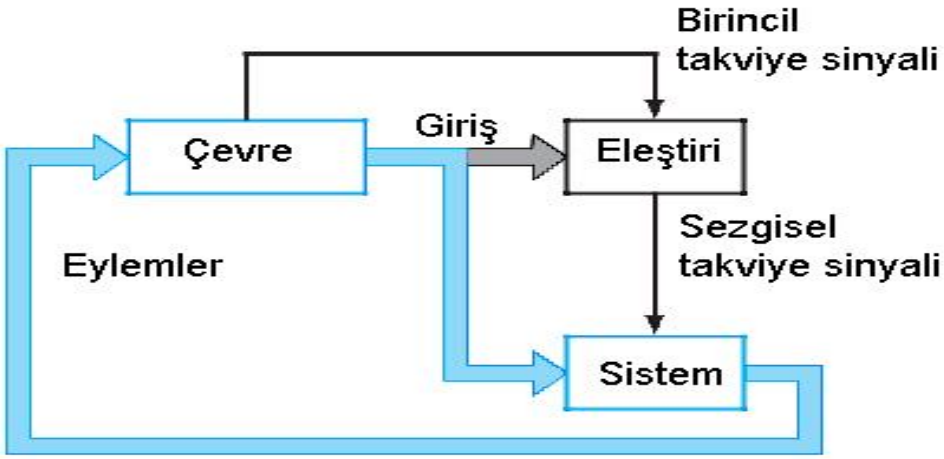


Şekil 2. 9 Öğretmensiz öğrenme blok diyagramı.

2.4.6.3 Takviyeli Öğrenme (Reinforcement Learning):

Şekil 2.23'de görüldüğü gibi takviyeli öğrenme türünde, ağa verilen girdilere karşılık ağ çıktı üretir. Burada öğretmen, öğretmenli öğrenmeden farklı olarak, beklenen çıktı olup olmamasına göre sisteme doğru ya da yanlış olduğunu gösteren sinyali gönderir (doğruysa ödül, yanlışsa ceza gibi). Kohonen tarafından geliştirilmiş olan LVQ (Linear Vector Quantization) ağları, takviyeli öğrenmeye örnek verilebilir. Genel olarak sınıflandırma problemlerinin çözümünde kullanılır (Elmas, 2018).

Bunların dışında, Radyal Tabanlı Yapay Sinir Ağları ve Olasılık Tabanlı Sinir Ağları gibi, yukarıda anlatılan öğrenme türlerini birlikte kullanan ağlar da mevcuttur.



Şekil 2. 10 Takviyeli öğrenme blok diyagramı.

2.4.7 Öğrenme Kuralları

Yapay Sinir Ağlarında çok sayıda öğrenme kuralı kullanılmaktadır. Bu öğrenme kurallarının büyük çoğunluğu en eski ve en çok bilinen öğrenme kuralı olan Hebb Kuralının bir versiyonudur. Bazı araştırmacılar temel amaç olarak biyolojik öğrenmeyi modellemenin peşindedirler. Bazıları ise eldeki öğrenme hakkındaki bilgilerin bir şekilde uyarlanmasına çalışmaktadır. Her iki noktada da, sinirsel işlemin gerçekte nasıl olduğuna yönelik bilgiler oldukça kısıtlıdır. Dolayısıyla sinirsel öğrenme, kullanılmakta olan öğrenme kurallarının ima ettiğinden çok daha karmaşıktır. Kullanılmakta olan bazı önemli öğrenme kuralları arasında Hebb Kuralı, Hopfield Kuralı, Delta Kuralı, Eğimli Değişim (Gradient Descent) Kuralı ve Kohonen Öğrenme Kuralı sayılabilir.

2.4.7.1 Hebb Kuralı

Öğretmensiz öğrenme örneği olan Hebb Kuralı, Donald Hebb tarafından 1949'da tanımlanmış olan, ilk ve en çok bilinen öğrenme kuralıdır. Temel kural; bir nöron başka bir nörondan girdi alırsa ve iki nöron da yüksek derecede aktifse (yani matematiksel olarak aynı işarete sahipse) nöronlar arasındaki bağlantının ağırlığı artırılmalıdır.

2.4.7.2 Hopfield Kuralı

Bir kural, küçük bir farklılık dışında Hebb Kuralına benzemektedir. O farklılık da şudur; girdi ve istenilen çıktının ikisi de aktifse veya ikisi de aktif değilse, ilgili

bağlantının ağırlığı öğrenme oranı kadar artırılır, aksi durumda ise öğrenme oranı kadar azaltılır.

2.4.7.3 Delta Kuralı

Hebb kuralının biraz daha geliştirilmişi olan delta kuralı, en yaygın kullanılan kural sayılabilir. Widrow-Hoff öğrenme kuralı ya da En Küçük Ortalamalar Karesi (LMS) kuralı olarak da adlandırılan bu kural, ilk olarak Bernard Widrow ve Ted Hoff tarafından 1959 yılında geliştirilmiştir. Buna göre, bir sinir hücresinin çıkışı ile istenen çıkış arasındaki farkı azaltmak için giriş ağırlık değerleri sürekli değiştirilir. Ağ hatasının karesini en aza indirmek (minimize etmek) için giriş katmanına ulaşıncaya kadar hata geri çoğaltılır. Bu tip ağlar ileri beslemeli geri yayılımlı ağlar olarak da bilinir (Elmas, 2018).

2.4.7.4 Eğimli İniş (Gradient Descent) Kuralı

Delta kuralından geliştirilmiş olan bu kurala göre; Delta Kuralında olduğu gibi, hatanın modifiye edilmesi için transfer fonksiyonunun türevi kullanılmaktadır. Eğimli Değişim Kuralının farklılığı, öğrenme oranına orantılı bir ek sabit değerin, son değiştirme faktörüne eklenmesidir. Bağlantı ağırlığı, bu ek değerli faktöre göre değiştirilir. Bu kural, yavaş bir yakınsama sağlamasına rağmen yaygın olarak kullanılmaktadır.

2.4.7.5 Kohonen Öğrenme Kuralı

Kohonen (1982) tarafından geliştirilen bu kural, biyolojik sistemlerdeki öğrenme sisteminden esinlenmiştir. Bu yöntemde, düğümlerin ağırlıkları ayarlamak (öğrenmek) için rekabet ettikleri düşünülmektedir. En uygun çıktıya sahip düğüm diğerlerine göre baskın olur ve sadece bu düğümün çıktı sağlaması söz konusudur. Kendi kendine öğrenme (self-organizing) olarak da bilinen bu kural, özellikle dağılımlara yönelik çalışmalarda kullanılsa da teorik alt yapısının tam gelişmemiş olmasından dolayı uygulamada henüz yaygınlık kazanmamıştır. Burada sunulan temel kurallara ilave olarak öğrenme kuralları hakkında daha detaylı bilgi için (Haykin, 1999)'dan yararlanılabilir.

2.4.8 Yapay Sinir Ağlarının Kullanım Alanları

Yapay sinir ağları bilgi sınıflama ve bilgi yorumlamanın da içinde bulunduğu çok değişik problemlerin çözümünde kullanılmaktadır. Yapay sinir ağlarının kullanıldığı alanlar özet olarak şu şekilde sıralanabilir;

- Denetim
- Sistem modelleme
- Ses tanıma
- El yazısı tanıma
- Parmak izi tanıma
- Elektrik işareti tanıma
- Meteorolojik yorumlama
- Otomatik araç denetimi
- Fizyolojik işaretleri (Kalp, beyin fonksiyonları vb.) izleme, tanıma ve yorumlama

Yukarıda verilen başlıklara ilave olarak, yapay sinir ağları, her türlü bilgiyi işlemek ya da analiz etmek amacıyla kullanılırlar. İş hayatı, finans, endüstri, eğitim ve karışık problemli bilim alanlarında, bulanık veya var olan basit yöntemlerle çözülemeyen problemlerin çözümünde ve doğrusal olmayan sistemlerde başarıyla uygulanmaktadırlar.

Temel olarak, yapay sinir ağları uygulamalarının çoğu aşağıdaki sınıflardan birine girmektedir:

- Öngörü
- Sınıflandırma
- Veri birleştirilmesi
- Veri kavramlaştırılması
- Veri süzmesi
- Resim veya görüntü işleme

Uygulama ile öbeklenmiş bu ağlardan bazıları farklı türdeki sorunları çözmek için kullanılmaktadır. Bu ağ türlerinden ileri beslemeli geri yayılımlı ağ en sık

kullanılandır. Uygulama alanına göre kullanılan yapay sinir ağları Tablo 2.7'deki gibi sınıflandırılabilirler.

2.4.9 Endüstriyel Uygulamalar

Sinir ağları sanayideki kalite denetiminde, kalitenin geliştirilmesinde yaygın olarak kullanılmaktadır. Pavilion Technologies şirketi tarafından geliştirilen bir sinir ağı işlem denetim sistemi, ürün kalitesi geliştirme ve ürün kalitesinin yükseltilmesinde kullanılmaktadır.

İngiliz Columbia Hidroelektrik santralında bulunan dört büyük senkron kompanzatör sinir ağları, güç dağıtım şebekelerinde başarıyla kullanılmaktadır.

Neural Application Corp. tarafından geliştirilen ark fırın denetim sistemi ile elektrik tüketiminde büyük tasarruf sağlandığı rapor edilmiştir. Bu denetim dünya çapında bir fırında kullanılmaktadır.

Tablo 2. 5 Yapay sinir ağları ve uygulama alanları (Stuart ve ark., 1995).

Uygulama	**Ağ Türü**
Öngörü	○ Geri-Yayılım Ağı. ○ Delta Bar Delta. ○ Genişletilmiş Delta Bar Delta. ○ Yüksek Seviyeli Ağlar. ○ Özörgütlemeli Harita Ağı. ○ Algılayıcı.
Sınıflandırma	• Geri-Yayılım Ağı. • Öğrenme Vektörü Nicelemesi. • Algılayıcı. • Olasılıksal Sinir Ağları. • Kohonen Ağı. • Boltzmann Makinesi.
Veri Birleştirilmesi	➢ Hopfield. ➢ Boltzmann Makinesi. ➢ Hamming Ağı. ➢ İki Yönlü Çağrışım Belleği. ➢ Yığın Ağları. ➢ Vektör Nicelemesi Ağı.
Veri Kavramlaştırılması	❖ Uyarlanır Rezonans Ağı. ❖ Özörgütlemeli Harita Ağı.

Veri Süzmesi	○ Yeniden Dolaşım.
Resim veya Görüntü İşleme	▪ Yeri-Yayılım Ağı. ▪ Algılayıcı.

2.4.10 Yapay Sinir Ağlarının Oluşturulması

Günümüzde kullanılan çok çeşitli yapıda yapay sinir ağı mevcut olduğundan, bunlardan hangisiyle çalışılacağını, yapılacak işin niteliğine göre belirlemek gerekmektedir. Burada, bir yapay sinir ağı oluştururken bilinmesi gereken bazı kavramlar açıklanmıştır.

2.4.10.1 İlk Yapay Sinir Ağları

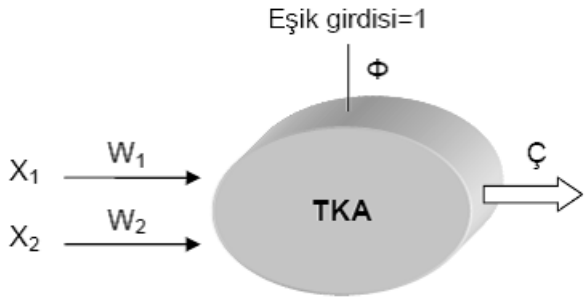
Tek Katmanlı Algılayıcılar:

Bu tip sinir ağı sadece iki girdi (X1,X2) ve bir adet çıktıdan (Ç) oluşur. Girişlerle düğüm arasında W1 ve W2 ağırlıkları ile bu ağın çıktısının sıfıra eşit olmamasını sağlayan ve değeri 1'e eşit olan eşik değeri (Φ) mevcuttur. Şekil 2.24'de tek katmanlı algılayıcı modeli görülmektedir.

Tek katmanlı algılayıcılarda çıkış doğrusaldır. Ağa gösterilen örnekler arasındaki doğrusal sınır çizgisi tespit edilmeye çalışılır. O yüzden çıkış fonksiyonu olarak eşik değer fonksiyonu kullanılır. Ağın çıkışı şu formül ile hesaplanır;

$Ç = f(\sum_{i=1}^{n} X_i W_i + \Phi)$

(Formül 2.1)

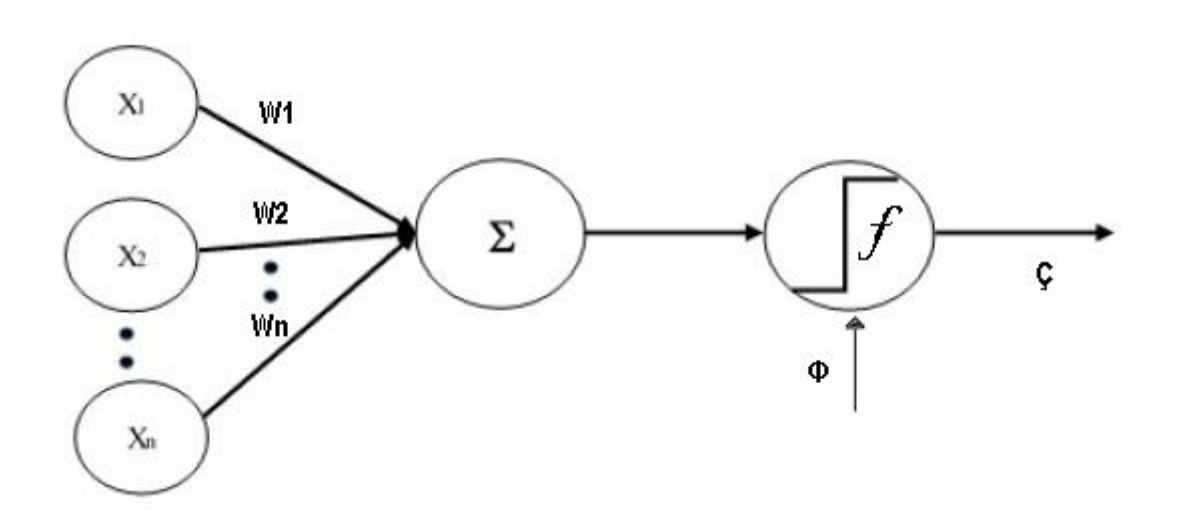


Şekil 2. 11 Tek Katmanlı Algılayıcı Modeli (Öztemel, 2003).

Algılayıcı (Perceptron)

Frank Rosenblatt 1958 yılında McCulloch–Pitts modelini geliştirerek "Perceptron" isimli modeli örüntü sınıflandırma amacıyla geliştirilmiştir. Algılayıcı da denen Perceptron modelinde; McCulloch–Pitts nöronuna öğrenme kuralı ilave edilmiştir. Öğrenme kuralı, 1949 yılında Donald Olding Hebb tarafından yayımlanan "The Organization of Behaviour" isimli çalışmada ortaya konulan Hebbian öğrenme kuralını temel almaktadır. Öğrenme kuralı ile ağrılık değerlerinin güncellenmesi sağlanmaktadır (Hebb, 1949).

Şekil 2.25'de basit bir perceptron görülmektedir. Perceptronlar, eğitilebilen yapay sinir hücreleridir.



Şekil 2. 12 Basit Algılayıcı (Perceptron).

Buradaki algılayıcıya ait net girdi şu şekilde hesaplanır:

NET = $\sum_{i=1}^{n} X_i W_i$

(Formül 2.2)

Burada X giriş değerlerini, W ise ağırlık değerlerini ifade etmektedir. Çıkış fonksiyonu olarak eşik değer fonksiyonu kullanılmıştır. Yani çıkış 0 ya da 1 olacaktır.

Ç = { 1 Eğer Net > Φ
{ 0 Eğer Net <= Φ

(Formül 2.3)

Net girdinin eşik değerden (Φ) küçük eşit ya da büyük olması durumlarına göre ağ bir çıkış üretecektir. Burada beklenen değer ile ağın ürettiği çıkış aynıysa, ağırlık değerlerinde herhangi bir değişiklik olmayacaktır. Fakat beklenen değer ile ağın çıkışı arasında fark olması durumunda ağın ağırlık değerlerinin şu şekilde güncellenmesi gerekmektedir.

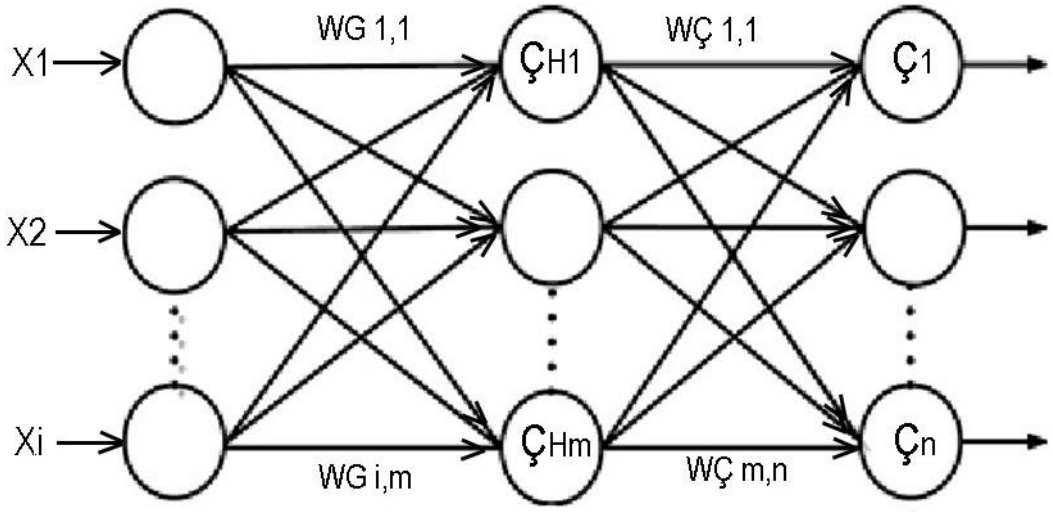
$W_n = W_o (\pm) \ \lambda X$

(Formül 2.4)

Burada λ değeri öğrenme katsayısı sabitidir. Ağın beklenen değerinin 0, çıkışın ise 1 olduğu durumda (Formül 2.4)'teki ağırlık değerlerinin küçülmesi gerektiğinden formüldeki işaret negatif, ağın beklenen değerinin 1 olduğu ve çıkışın 0 olduğu durumda, ağırlık değerlerinin büyümesi gerektiğinden formüldeki işaret pozitif (+) olur ve bu şekilde ağırlık değerleri güncellenir. Bu güncelleme işlemi, beklenen değer ile ağın değeri eşit olana kadar sürer.

2.4.10.2 İleri Beslemeli (Feed Forward) Ağlar

İleri beslemeli ağlar, sayısal girişin düğümler arasındaki ağırlıklarla çarpımlarının toplanarak bir aktivasyon fonksiyonu yardımıyla bir sonraki düğüme girdi oluşturacak şekilde tasarlandığı yapay sinir ağı çeşididir. Bilgi akışı giriş çıkış yönlüdür. Şekil 2.26'da görüldüğü gibi her düğüm, kendisindeki bilgiyi ağırlıklar vasıtasıyla bir sonraki katmanda bağlı bulunduğu düğüme/düğümlere girdi oluşturacak şekilde ileri aktarmaktadır.



Şekil 2. 13 İleri Beslemeli Yapay Sinir Ağı.

2.4.10.3 Geri Yayılım (Back Propagation) Algoritması

Yapay sinir ağlarını eğitmenin çeşitli yollarından biri olan Geriye Yayılım Algoritması, aslında ağın çıkış hatasının geriye doğru ağın bütününe yayılması işlemi olup, bu algoritma George Hinton ve James McClelland tarafından geliştirilmiştir. Beklenen çıkış ile ağın çıktısı arasındaki sayısal fark, ağın hatası olarak kabul edilir ve bu hata, çıkış katmanından giriş katmanına doğru bütün ağırlıklara dağıtılır.

Burada ağ en az bir ara katmandan oluşur ve ara katman sayısı ile ara katmandaki düğüm sayılarını tasarımcı belirler. Ara katman ve düğüm sayısının artması, ağın hatırlama yeteneğini arttırır fakat öğrenme süresinin uzamasına neden olur (Elmas, 2018). Geri yayılım algoritmasında hata, ileri besleme aşamasında kullanılan aktivasyon fonksiyonunun türevi alınarak ağda geriye doğru dağıtılır.

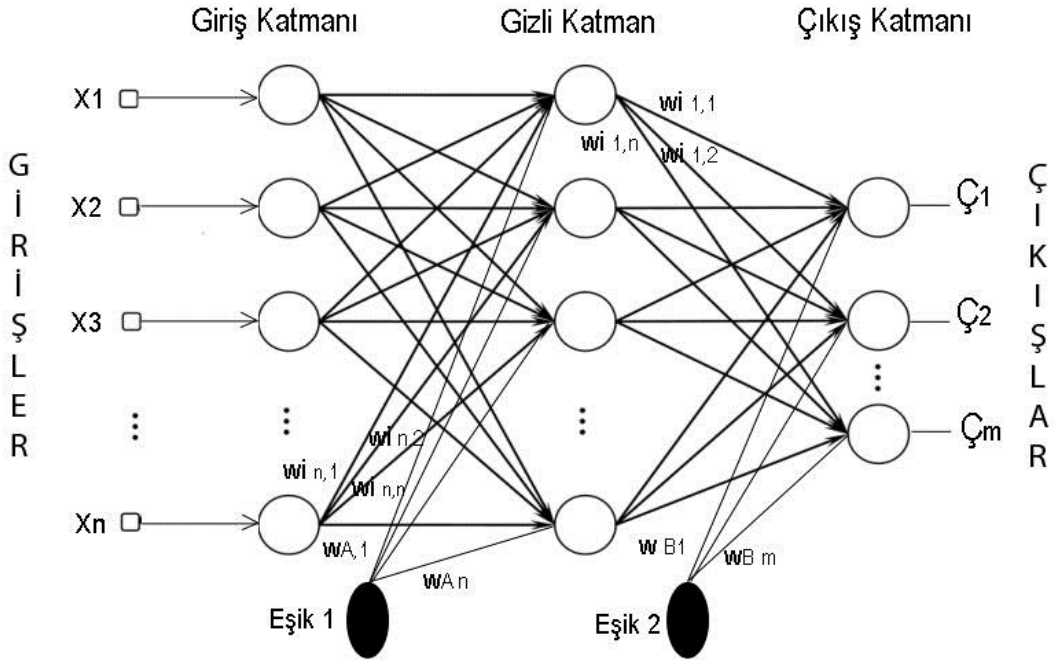
Başlangıçta tüm ağırlıklar rastgele atanır. Eğitim veri setindeki her girdi için ağ etkinleştirilir ve çıktı gözlemlenir. Bu çıktı beklediğimiz çıktı ile karşılaştırılır ve hata önceki katmana geri-yayılır. Bu hata kaydedilir ve ağırlıklar buna göre değiştirilir. Bu ağırlık güncelleme işlemi, ağın hatası istenen eşiğin altına düşene kadar devam eder.

2.4.10.4 Çok Katmanlı Algılayıcı Yapay Sinir Ağları

İlk yapay sinir ağı modellerinin sadece doğrusal problemleri çözebilmesi, doğrusal olmayan problemleri çözebilecek yeni yapay sinir ağı modellerinin geliştirilmesini zorlamıştır. XOR gibi doğrusal olmayan problemleri çözmek için geliştirilmiş ağ modellerinden biri de bu "Çok Katmanlı Algılayıcı Modeli"dir.

Minsky'nin basit algılayıcı modelinin XOR problemine çözüm üretemediğini göstermesiyle yapay sinir ağı çalışmaları bir süre durma noktasına gelmiştir. Bu tür doğrusal olmayan problemleri çözmek için Rumelhart ve arkadaşları tarafından "Hata Yayma Modeli" ya da "Geri Yayım (Backpropagation) Modeli" de denen Çok Katmanlı Algılayıcı Modelini (ÇKA) geliştirdiklerinden daha önce bahsedilmişti. Bu modelin gelişmesiyle birlikte yapay sinir ağlarına olan ilgi tekrar artmış ve bu alandaki çalışmalar kaldığı yerden devam etmiştir.

Öğrenme yöntemi olarak Delta Öğrenme Kuralını kullanan ÇKA, mühendislik uygulamalarındaki doğrusal olmayan sınıflandırma, tanıma ve genelleme gibi çoğu probleme çözüm üretebilmektedir. Bir ÇKA, basit bir algılayıcıdan farklı olarak şekil 2.27'de görüldüğü gibi Giriş Katmanı, Ara Katman ve Çıkış Katmanı olmak üzere en az üç katmandan oluşmaktadır.



Şekil 2. 14 ÇKA modeli.

Çok Katmalı Algılayıcı Ağında Öğrenme:

Bir ÇKA ağının öğrenmesi, eğitim esnasında bir girdi setine karşılık üretmesi gereken çıktının belli olmasından dolayı öğretmenli öğrenme stratejisine göre çalışmaktadır. Kullandığı öğrenme kuralı olarak, En Küçük Kareler Yönteminin uyarlanmış hali olan Delta Kuralının bir sürümü denebilecek genelleştirilmiş şekli Genelleştirilmiş Delta Kuralını kullanmaktadır.

Bir ÇKA ağında ileri doğru ve geri doğru olmak üzere iki farklı yönde hesaplama yapılmaktadır.

A. İleri Doğru Hesaplama:

Bu aşamada sayısallaştırılmış öğrenme seti ÇKA ağına gösterilir. Bu katmanda herhangi bir işlem yapılmaz ve girişler (X_1,X_2,…,X_n) olduğu gibi ara katmana iletilir. İleri doğru hesaplama ağın hem eğitim hem de test aşamasında uygulanır.

Ara katmanda bulunan her bir düğümdeki net girdi şu şekilde hesaplanır;

$$\mathrm{NET}_j^i = \sum_{k=1}^{n} X_k^i \, Ç_k^i \qquad \text{(Formül 2.5)}$$

Aktivasyon fonksiyonu olarak sigmoid fonksiyonu kullanılan bir düğümün çıktısı ise şu formülle hesaplanır;

$$Ç_j^i = \frac{1}{1+ e^{-(\mathrm{NET}_j^i + \beta_j^i)}}$$

(Formül 2.6)

Ara katmanın çıkışları, çıkış düğümlerine giriş olarak, aynı şekilde bütün düğümler için çıkış (Ç) hesaplanır. Burada Beta (β) değeri, gizli katman ve çıkış katmanına bağlanan eşik değerini göstermektedir.

B. Geriye Doğru Hesaplama:

Bu hesaplama, sadece ağ eğitilirken yapılır. Çıkış düğümünün/ düğümlerinin çıktısıyla olması gereken çıkış arasındaki fark şu şekilde hesaplanır;

$E_m = B_m - Ç_m$

(Formül 2.7)

Burada E_m bir tek düğümün hatasını, B_m beklenen çıkış değerini, $Ç_m$ ise ağın anlık çıkış değerini ifade etmektedir. Ağın toplam hatası şu şekilde hesaplanır;

$$H = \frac{1}{2}\sum_m E_m^2$$

(Formül 2.8)

Ağın eğitimi süresince ağın toplam hatası (H), kullanıcının belirlemiş olduğu kabul edilebilir hatadan düşük olana kadar, ağırlıkların güncellenmesi işlemi devam eder. Dolayısıyla ağırlıklar güncellenirken iki tür ağırlık güncelleme durumu söz konusudur.

1. Çıkış katmanı ile ara katman arasındaki ağırlıkların güncellenmesi şu şekilde yapılır;

$$\Delta W_{jm}^o = \lambda \delta_m Ç_j^o + \alpha \Delta W_{jm}^o(t-1)$$

(Formül 2.9)

Bu hesaplamada (Formül 2.9) λ, ağın eğitimi sırasında yerel optimuma takılmaması için ağırlık değişim değerinin ne oranda bir sonraki değere ekleneceğini gösteren "Öğrenme katsayısı"nı; α ise, ağırlıkların değişim adım büyüklüğü olan "Eğitim Katsayısı"nı ifade eder. Denklemdeki diğer bir değişken olan δ_m ise çıkış düğümünün hatasını göstermektedir.

Çıkış düğümünün hatası şu şekilde hesaplanır;

δ_m = f '(NET).Em (Formül 2.10)

Formülden de görüldüğü üzere (2.10), hata hesaplanırken NET fonksiyonunun türevi kullanılır (f '(NET)). Dolayısıyla bu formülün nihai hali şu şekilde olacaktır;

δm = Çm (1 - Çm).Em (Formül 2.11)

ÇKA ağının herhangi t. adımdaki ağırlığın yeni değeri şu şekilde hesaplanır;

$$W_{jm}^{o}(t) = \Delta W_{jm}^{o}(t) + W_{jm}^{o}(t-1) \quad \text{(Formül 2.12)}$$

Benzer şekilde eşik değer bağlantılarının ağırlık güncellemesi şu formüle göre yapılır;

$$\Delta\beta_{jm}^{o} = \lambda\delta_{m} + \alpha\Delta\beta_{jm}^{ç}(t-1) \quad \text{(Formül 2.13)}$$

Ağın herhangi bir t. adımdaki eşik değer ağırlık hesaplaması şu şekilde olur;

$$B_{jm}^{o} = \Delta\beta_{jm}^{o}(t) + \beta_{jm}^{o}(t-1) \quad \text{(Formül 2.14)}$$

2. Ara katman ile giriş katmanı arasındaki ağırlıkların güncellenmesi şu şekilde yapılır;

$$\Delta W_{kj}^{i} = \lambda\delta_{j}ç_{k}^{i} + \alpha\Delta W_{kj}^{i}(t-1) \quad \text{(Formül 2.15)}$$

Bu formüldeki (2.15) hata değişim miktarı δ_mşu şekilde hesaplanır;

$$\delta_{j}^{i} = f'(NET).\sum_{m}\delta_{m}\ W_{jm}^{i} \quad \text{(Formül 2.16)}$$

Burada (2.16) NET hesaplanırken aktivasyon fonksiyonunu **sigmoid** olarak düşünülürse formülün son hali şu şekilde olacaktır;

$$\delta_{j}^{i} = ç_{j}^{i}(1-ç_{j}^{i}).\sum_{m}\delta_{m}\ W_{jm}^{i} \quad \text{(Formül 2.17)}$$

Giriş ile ara katman arasındaki ağırlıklar şu formüle göre güncellenir;

$$W_{kj}^{i}(t) = \Delta W_{kj}^{i}(t) + W_{kj}^{i}(t-1) \quad \text{(Formül 2.18)}$$

Benzer yolla ara katman eşik değer ağırlıkları şu şekilde güncellenir;

$$\Delta\beta_j^i(t) = \lambda\delta_j^i + \alpha\Delta\beta_j^i(t-1) \quad \text{(Formül 2.19)}$$

Ağın herhangi bir t. adımdaki eşik değer ağırlık hesaplaması şu şekilde olur;

$$\beta_{jm}^i(t) = \Delta\beta_j^i(t) + \beta_j^i(t-1) \quad \text{(Formül 2.20)}$$

Buraya kadar yapılan işlem, ağa gösterilen bir tek örnek için birinci iterasyondur. İkinci iterasyon için ağa ikinci örnek verilerek bütün adımlar öğrenme tamamlanıncaya kadar devam eder.

2.4.11 Yapay Sinir Ağı Kullanarak Örüntü Tanıma

Örüntü tanıma konusunda yapay sinir ağlarının oldukça yaygın bir kullanımı söz konusudur.

Örnek:

Elyazısı tanıma, el ile yazılan harf, rakam ve sembollerin bilgisayar sistemleri tarafından tanınmasıdır. İnsanlar için oldukça kolay olmasına rağmen, bir zemin üzerindeki çizgi ve eğrilerin otomatik olarak harf ve rakamlar, daha ileri aşamada da anlamlı sözcükler olarak algılanması oldukça zor bir problemdir.

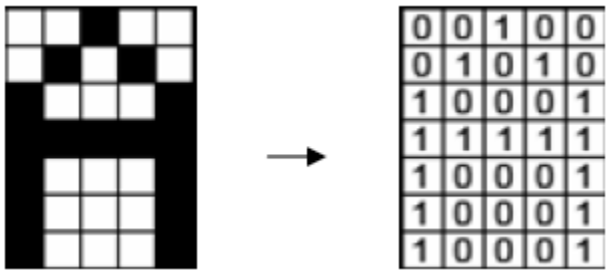
Elyazısı Tanıma Yöntemleri:

Elyazısı tanıma yöntemleri; Etkileşimli (çevrimiçi) ve etkileşimsiz (çevrimdışı) yöntemler olmak üzere iki gurupta toplanabilir:

1. **Etkileşimli sistemler**, yazının yazıldığı sırada tanıyan sistemlerdir. Tabletlerde ya da akıllı cep telefonlarında sıklıkla kullanılır.
2. **Etkileşimsiz sistemler** ise genelde kâğıt üzerine daha önceden yazılmış bilgilerin bir tarayıcı vasıtasıyla ya da fotoğrafı çekilmek suretiyle sayısallaştırılarak, sonradan bu yazı karakterlerinin tanınmaya çalışılması işlemidir.

El yazısı harf tanıma işlemi, kâğıt üzerinden okunan metnin bölütleme sonrasında önce kelimelere parçalanması, daha sonra da bu kelimelerin bölütleme işlemiyle harflere ayrılması, sonra bu harflerin tahmin edilmesi ve daha sonra bu

harflerin birleştirilmesiyle anlamlı kelimeler oluşturma şeklinde yapılmaktadır. Şekil 2.28'de düzgün bir büyük "A" harfinin yapay sinir ağına giriş olarak verilebilmesi için sayısallaştırması görülmektedir.

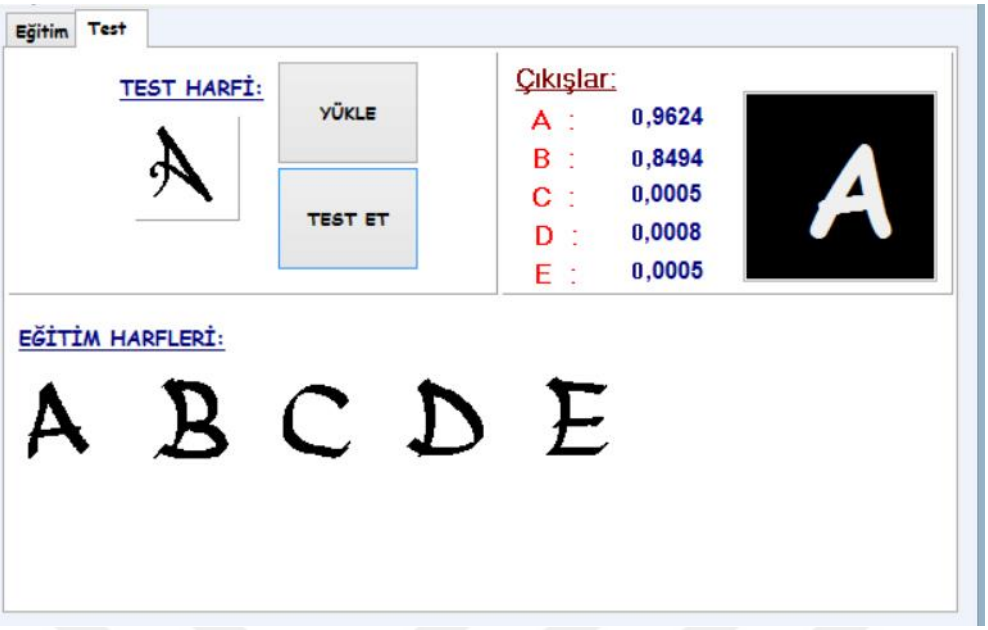


Şekil 2. 15 Düzgün bir A karakterinin 7x5'lik bir matrisle ikili gösterimi.

YSA'ya gösterilecek harfler ilk önce 50x50 formatında fotoğraflara çevrilip, bu fotoğraflar bir takım ön işlemlerden geçirilerek şekil 2.28'deki gibi 0 ve 1'lerden oluşan bir matrise dönüştürülmüş olup, oluşturulan bu matris İleri Beslemeli Geri Yayılımlı YSA'ya gösterilmek suretiyle YSA eğitilmiştir.

El yazısı karakter tanıma esnasında 100 girişli, 100 ara katmanlı ve 5 çıkışlı İleri Beslemeli Geri Yayılımlı yapay sinir ağı kullanılmıştır.

Eğitim aşamasında sinir ağını eğitmek için ilk önce el yazısı ile A, B, D ve E harfleri gösterilerek YSA'nın eğitilmesi sağlanmıştır. Daha sonra yine el yazısıyla yazılmış karakterler gösterilerek test işlemi Şekil 2.29'da görüldüğü gibi yapılmıştır. YSA, kendisine ilk kez gösterilmiş olan bu el yazısı karakterlerini tanımayı başarmıştır. Burada YSA'nın güçlü örüntü tanıma özelliğinden yararlanılmıştır.



Şekil 2. 16 Elyazısı karakter tanımada YSA kullanımı.

2.5 Ağrı

2.5.1 Ağrının Tanımı:

Ağrı, hastayı yardım arayışı içine sokan ve varlığı çağlar öncesine dayanan bir sağlık sorunudur. Tarihi insanlığın var oluşu kadar eski olan bu durumun en yaygın kullanılan tanımını Uluslararası Ağrı Araştırmaları Teşkilatı (IASP) "Varolan veya olası doku hasarına eşlik eden veya bu hasar ile tanımlanabilen, hoşa gitmeyen duyusal ve emosyonel bir deneyimdir" olarak yapmıştır [56, 57] (Eti-Aslan, 2002)(Çöçelli ve ark., 2008).

Algı derecesi ve şiddetinin yaş, cinsiyet, kültür ve daha önceki deneyimlere göre kişiden kişiye değişiklik gösterdiği belirlenmiş ve ağrının var oluşu her ne kadar istenmeyen bir durum olsa da, bazı kesimler tarafından izlenmesi gereken beşinci yaşam bulgusu olarak da nitelenmiştir. Rus yazar Lev Nikolayeviç Tolstoy şöyle der: "Acıyı duyabiliyorsan canlısın, başkalarının acılarını duyabiliyorsan insansın." Yani, bir kimsenin ağrıyı hissediyor olması, hayatta olduğunu göstermektedir.

Kişinin tanımlayabildiği en düşük ağrı miktarı, ağrı eşiği olarak tanımlanmaktadır. Bu eşik, farklı bireylerde ve/veya aynı bireyde zaman içerisinde değişiklik gösterebilmektedir.

Ağrı sınıflandırmaları aşağıdaki şekillerde görülebilmektedir (Çöçelli ve ark., 2008).

- Süresine göre; akut ya da kronik,
- Kaynaklandığı bölgeye göre; somatik, visseral, sempatik;
- Mekanizmasına göre; nösiseptif, nöropatik, deafferentasyon, reaktif, psikosomatik;
- Duyu şekillerine göre; sızlama şeklinde, yanıcı, batıcı ve kolik

2.5.2 Ağrının Tespit Edilmesi

Ağrının tespit edilmesi kolay bir iş değildir. Geleneksel yollarla ağrı tespiti, ya sağlık uzmanının hastayı gözlemi ile ya da kâğıt üzerine yapılan ölçeği hastanın işaretlemesi yoluyla interaktif şekilde yapılır. Wong Baker Yüzler ölçeği (WBF), görsel analog ölçekleme (visual analog scale-VAS), Sayısal Ağrı Ölçeği (SAÖ), Sözel Değerlendirme Ölçeği (SDÖ), FLACC (yüz, bacaklar, aktivite, ağlama, avunma) gibi bir ve iki boyutlu ya da İlk Ağrı Değerlendirme Ölçeği, McGill Ağrı Anketi (MPQ), Kısa Ağrı Envanteri (BPI) gibi çok boyutlu ağrı ölçekleme yöntemleri de dahil olmak üzere birçok ağrı tespit yöntemi mevcuttur (Ayaz, 2019).

Bu yöntemlerde hasta, duyduğu acıyı sıfır acıdan, dayanılmaza doğru kendisi işaretleyerek gösterir. Seçilen yönteme göre ölçekteki kademe sayısı farklılık göstermektedir. Şekil 2.30 ve 2.31'de farklı türden ağrı tespit yöntemleri ve buna bağlı olarak farklı türden ağrı tespit cetvelleri mevcuttur.

Ağrı değerlendirilmesinde ölçek kullanımı, her ne kadar hekimler ve hemşireler arasındaki görüş farkını ortadan kaldırmaya imkân sağlamasıyla avantaj olarak kullanışlı gibi gözükse de çocuklar, akıl sağlığı bozuk hastalar, bunamış hastalar, felçli hastalar gibi bazı tür hastalara uygulanamaması bu yöntemlerin dezavantajı olarak bildirilmektedir. Ayrıca hastanın ilk değerlendirmesinde hissettiği acıyı maksimum olarak belirtmesinden sonra, ağrılarının artması durumunda bunu düzeltme yoluna başvurulması, ağrı tespitinin diğer bir dezavantajı olarak karşımıza çıkmaktadır. Bu tür durumlar, bilgisayar destekli otomatik sistemlerin gerekliliğini ortaya çıkarmaktadır.

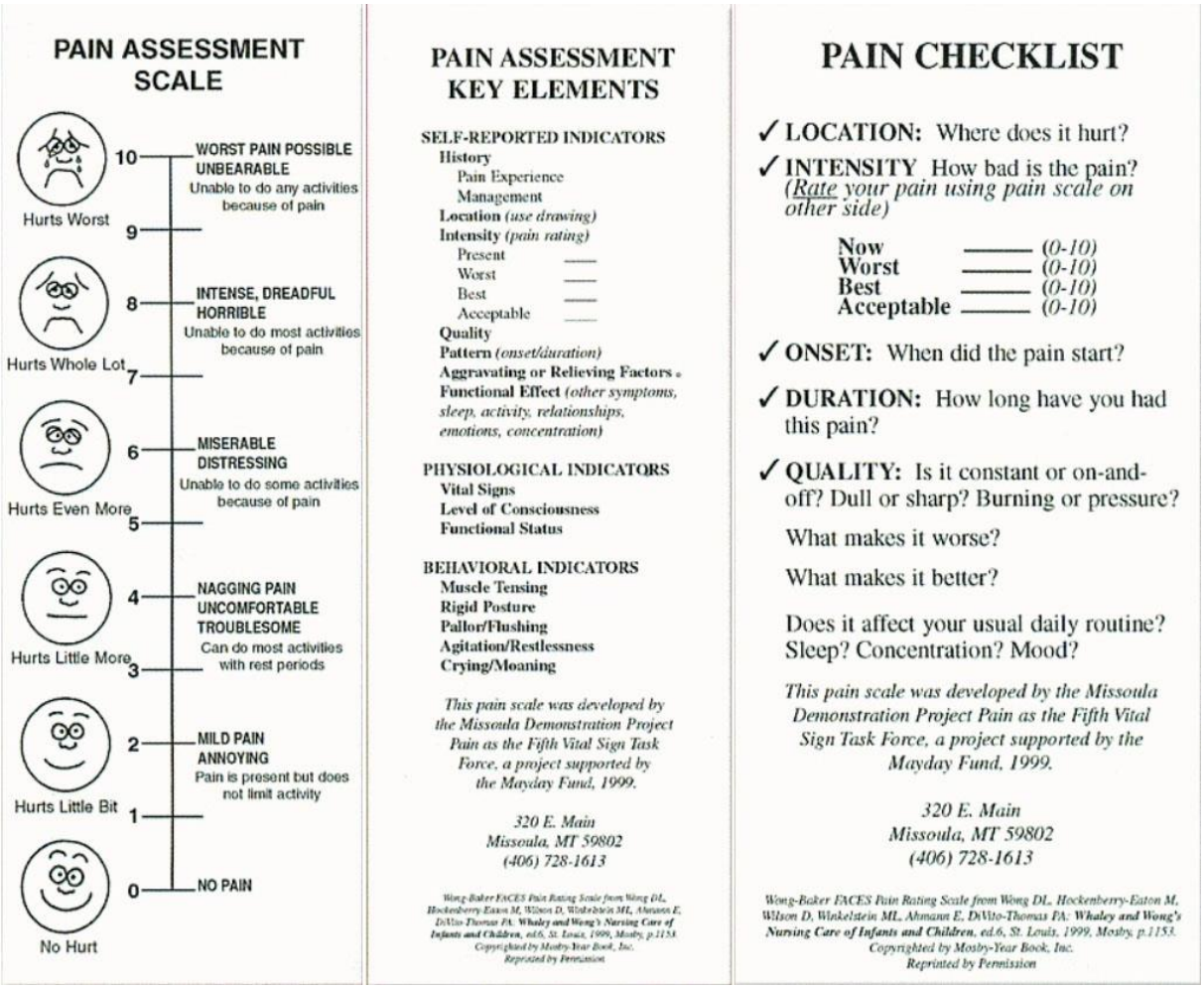




FLACC AĞRI SKALASI

Yüz	0 Özel bir ifade veya gülümseme yok	1 Arasıra yüz ekşitme,kaş çatma, çekilme, rahatsızlık	2 Sıklıkla kaş çatma, çeneyi sıkma, yanaklarda kasılma Titreme
Bacaklar	0 Normal pozisyon veya gevşeklik	1 Rahatsız - amaçsız kasılma	2 Tekme atma veya bacakları yukarı çekme
Aktivite	0 Sessizce uzanıyor Normal pozisyon Kolaylıkla hareket ediyor	1 Kıvranıp arkaya/öne dönüyor Dirençli	2 Açılı duruyor Rijid veya tetik şeklinde
Ağlama	0 Ağlama yok (Uyanık veya uyuyor)	1 Sızlanıyor veya inliyor	2 Hızla ağlıyor Çığlık atıyor, iç çekiyor
Teselli Hali	0 Hayatından memnun, rahat	1 Arasıra dokunmakla, konuşmakla ve kucaklamakla teselli oluyor Dikkati dağıtılabiliyor	2 Teselli etmek veya rahat ettirmek güç

Şekil 2. 17 Ağrı tespit ölçeği örnekleri.



Şekil 2. 18 Ağrı Tespit Cetvelleri (URL-8, 2019).

.

BÖLÜM 3. YÖNTEM

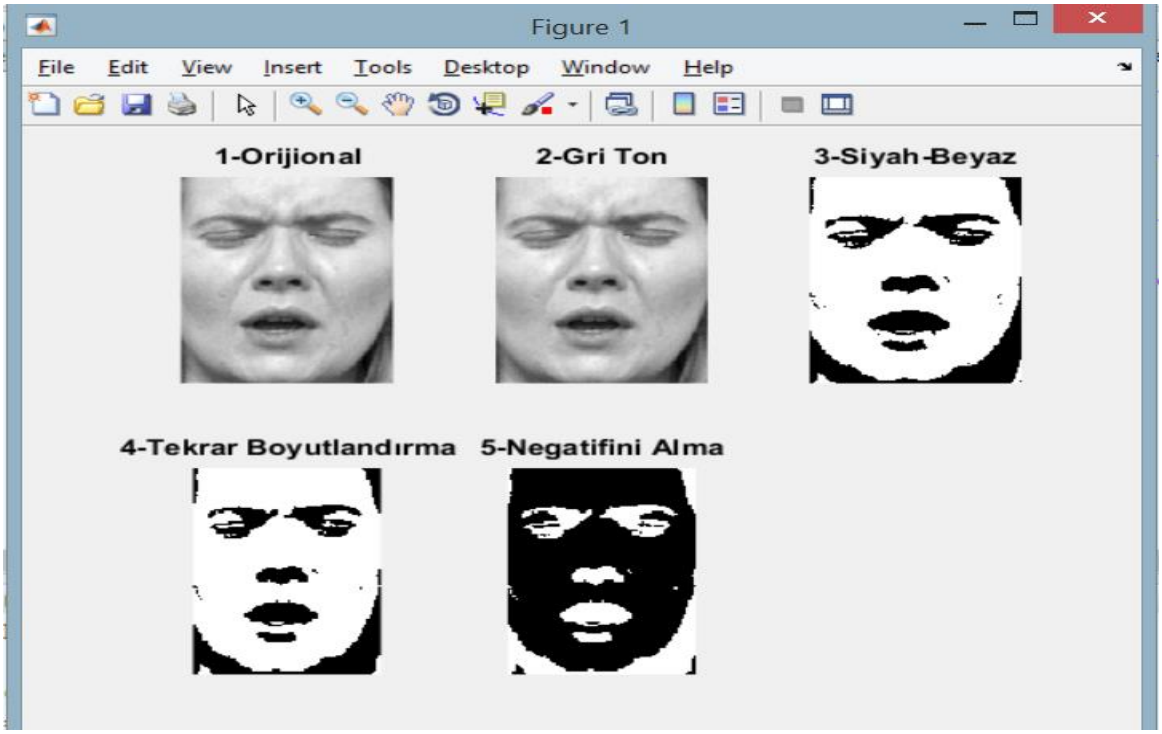
3.1 Veri Toplama

Uygulamada kullanılacak veri setinde bulunan 250 adet fotoğrafın, 69 tanesi eğitim için (toplam verinin yaklaşık %27'i) ve geri kalan 181 tanesi ise test seti olarak (toplam verinin yaklaşık %73'u) bölündükten sonra eğitim setine ait fotoğraflara, sinir ağımızın işleyebileceği şekilde ön işlem uygulanmıştır. Uygulamada kullanılan veri setinin internet erişim adresi kaynaklar bölümünde verilmiştir (URL-9, 2019).

3.2 Veri Ön İşleme

Yapılan bu çalışmada uygulama geliştirme aracı olarak Matlab R2014b kullanılmıştır. Yapay sinir ağları, giriş olarak sayısal verilerle çalıştığından, öncelikli olarak mevcut fotoğrafı ağa sunmadan önce bir takım sayısallaştırma ve diğer ön işleme özellikleri için bilgisayarlı görme tekniklerinden faydalanılmıştır.

Yapılan bu ön işlem sonunda, renkli ya da gri tondaki bir fotoğraftan, 120x90 boyutunda ikili (binary) formatında fotoğraflar oluşturulmuştur. Şekil 3.1'de yapılan bu ön işlemler aşama aşama görülebilmektedir.



Şekil 3. 1 Adım adım ön işlem aşamaları.

3.3 Yapay Sinir Ağını Oluşturma ve Ağın Eğitilmesi

Yapay sinir ağları, sınıflandırma konusunda oldukça başarılı olduğundan, bu uygulamayı gerçekleştirmede İleri Beslemeli Geri Yayılımlı (Feed Forward Back Propagation) yapay sinir ağı tipi seçilmiştir.

Bu uygulamadan beklentimiz, ağımıza verdiğimiz yüz ifadesinde, eğitim setindeki yüzlerde ağrı olup olmadığını öğrenip, daha sonra ilk kez gördüğü test setindeki yüzlerde ağrı olgusunun olup olmadığını tahmin etmesi olarak özetlenebilir.

Ağımızın bunu yapması için öncelikle, ağrı ifadesi olan ve olmayan fotoğrafları ağımıza göstermiş bulunmaktayız. Bunun için kullanmış olduğumuz veri setimizdeki resimler, acı derecesine göre gruplandırılıp ağımıza tanıtılmıştır (URL-9, 2019).

Fotoğraflarımızı ağa tanıtırken 3 farklı yüz ifadesine sahip fotoğraf ağa sunulmuştur. Bunlar;

- Ağrı hissetmeyen yüz,
- Az ağrı hisseden yüz,
- Çok ağrı hisseden yüz

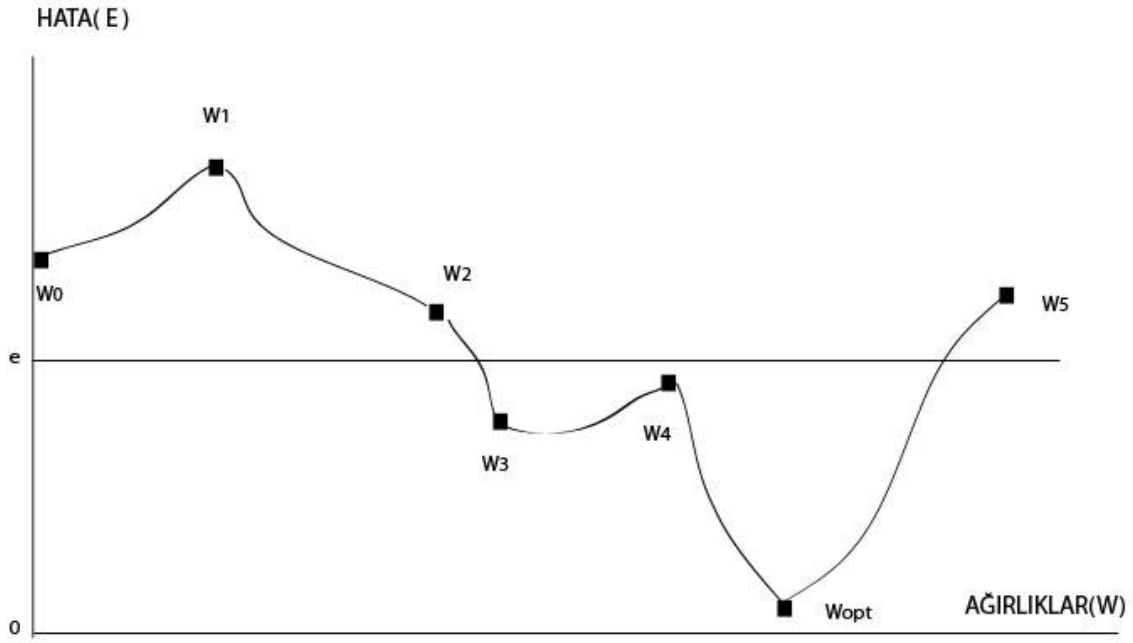
Yapay sinir ağı, eğitim setinden giriş olarak aldığı her bir fotoğraftaki yüz ifadesini, çıkış hatası kabul edilebilir hata oranının altına düşene kadar, bu 3 gruba göre gruplandırma işlemini gerçekleştirmiştir. Daha sonra test için ayrılan ve ağın daha önce görmediği diğer grup fotoğraf ağa tek tek gösterilip, bu fotoğraflardaki ifadelerin hangi ağrı seviyesine ait olduğunu bilmesi istenmiştir.

Uygulamamızda kullandığımız yapay sinir ağının bazı özellikleri şu şekildedir;

- Giriş sayısı = 210;
- Ara katman sayısı = 210;
- Çıkış sayısı = 3;
- Eğitim katsayısı(α) = 0.3;
- Momentum katsayısı(λ) = 0.7;
- Giriş eşik değeri(βi) = - 0.1;
- Ara katman eşik değeri(βo) = - 0.8;

Görüldüğü üzere 210 girişli, 210 düğümden oluşan 1 ara katmanlı ve 3 çıkışlı olarak, Şekil 2.27'deki gibi İleri Beslemeli Geri Yayılımlı bir YSA oluşturulmuştur. Ayrıca eğitim (α) ve momentum (λ) sabitleri ile giriş(βi) ve ara katman eşik değerleri(βo) belirlenmiştir.

Düğümler arası ağırlıklar (W), başlangıç aşamasında 0 – 1 aralığında ve "random" olarak belirlenir. Ağ çalıştığı sürece bu ağırlıklar, ağın doğru sonucu verebilmesi için güncellenir.



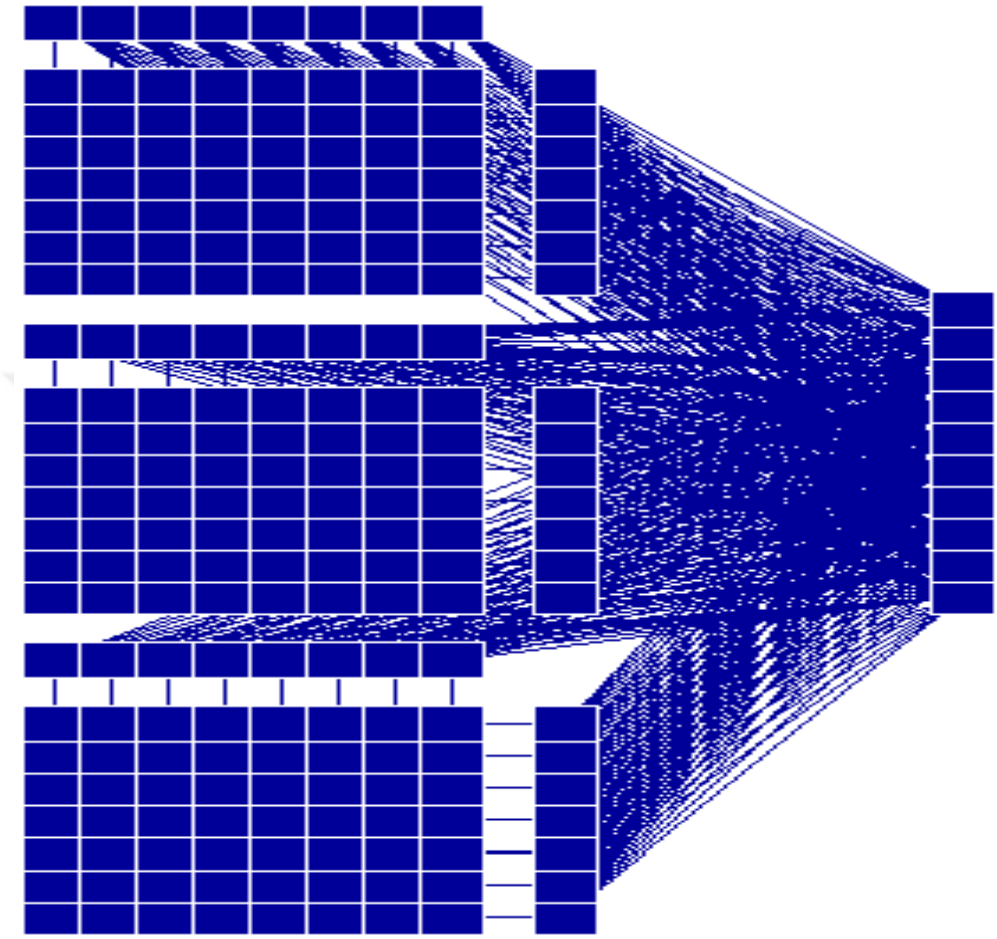
Şekil 3. 2 Ağırlık setine göre YSA'nın çıkış hatası değişimi.

Şekil 3.2'de W0,… W5 ve Wopt, ağda kullanılan ağırlık oranlarını; e ise, ağın çıkışında ürettiği kabul edilebilir hata oranını göstermektedir. Wopt, ağın en ideal çözüm vermesini sağlayan ağırlık değerleri olmak üzere, W0, W1,W2 ve W5 ağırlıkları, ağın kabul edilebilir hata oranını vermesini sağlamada yeterli değildir. Fakat W3 ve W4 kabul edilebilir hata oranının altında çıkışa yardımcı olduğu için kabul edilebilir ağırlıklardır.

Buradan da anlaşılacağı üzere, yapay sinir ağları, hiçbir zaman en doğru çözümü ürettiğini iddia etmez. Amaç, çözüm kümesindeki çözümlerden en idealine ulaşabilmektir.

Ağı oluşturduktan ve parametreler belirlendikten sonra ilk olarak işlenecek olan fotoğraf, ön işlem aşamasından sonra ağa sunulur.

Uygulamada kullanılan YSA'yı Şekil 2.27'deki gibi oluşturduktan sonra, ağın işlem süresini kısaltıp performansını arttırabilmek üzere giriş ağa sunulmadan önce Şekil 3.3'teki yöntem uygulanmıştır (Martella, 2005).



Şekil 3. 3 Performans arttırmak için kullanılan yöntem (Martella, 2005).

3.4 Ağrı İfadesinin Tanınması

Yapay sinir ağına eğitim setinin gösterilmesiyle ağ, gösterilen her fotoğrafa karşılık bir çıktı vermektedir. Ağın ikilik sisteme göre (100) çıktısını vermesi ağrısız yüz, ikilik sisteme göre (010) çıktısını vermesi az ağrılı yüz, yine ikilik sisteme göre (001) çıktısını vermesi ise çok ağrılı yüz olduğunu göstermektedir.

Oluşturduğumuz YSA'nın yaptığı işlemler sırasıyla şöyledir:

1. Ağın oluşturulması.
2. Ağda kullanılacak parametrelerin belirlenmesi (Öğrenme katsayısı, momentum katsayısı vb.).

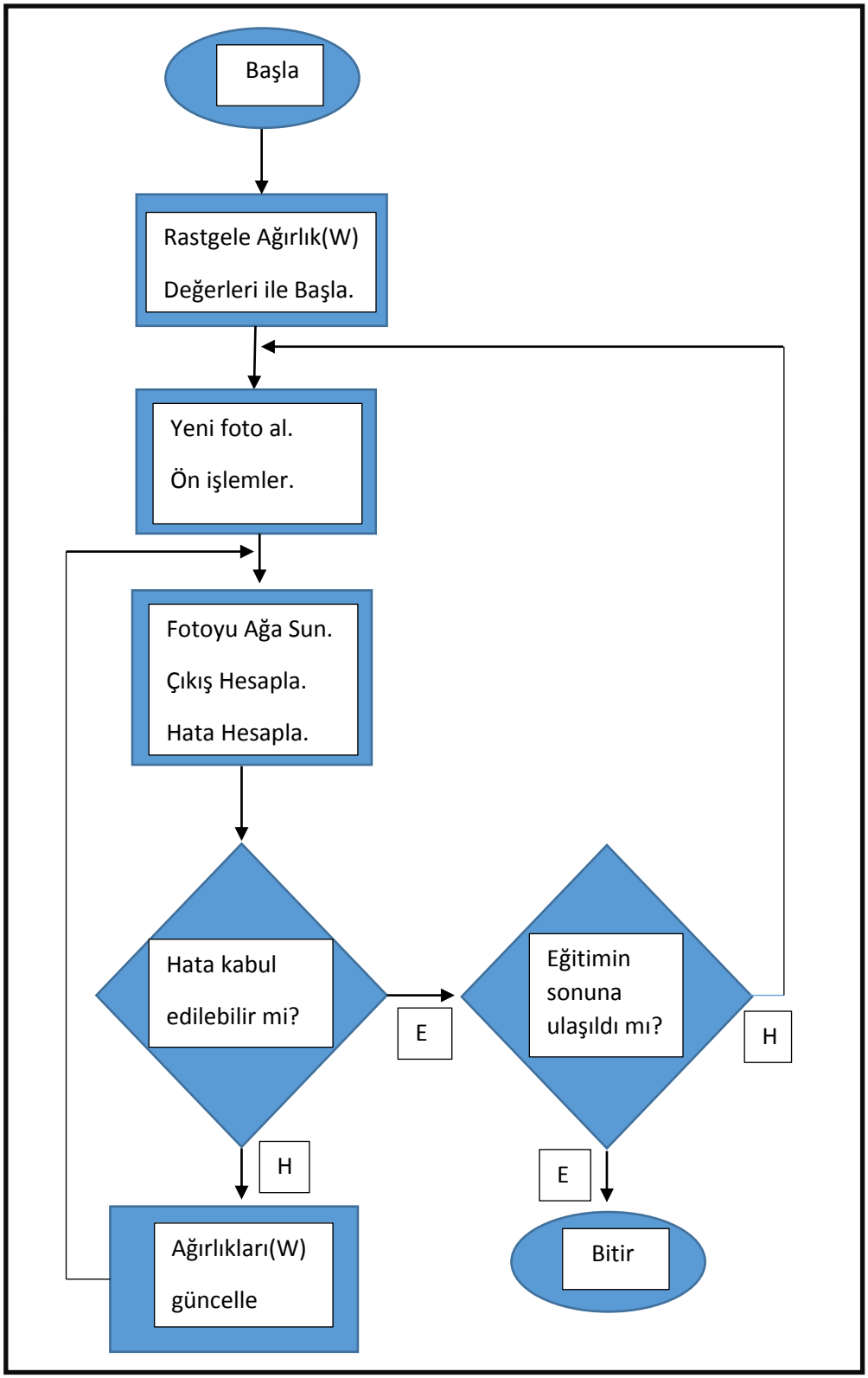
3. Başlangıç ağırlık değerlerinin verilmesi.
4. Ağa gösterilecek fotoğrafın seçilmesi, ön işlemden geçirilmesi.
5. Oluşturulan sayısal verinin ağa gösterilmesi, ileri besleme fonksiyonunun işletilmesi ve çıkış hesaplanması.
6. Hata kontrolü;
   i. Hata kabul edilemezse geri-yayılım fonksiyonunun işletilmesi ve ağırlıkların güncellenmesinden sonra 5. adımdan devam edilmesi.
   ii. Hata kabul edilebilirse bir sonraki eğitim fotoğrafını alıp 4. adımdan devam edilmesi.
7. Eğitim setinin sonunun kontrolü ve öğrenme sürecinin tamamlanması.

Bu işlemler ağın öğrenme süreci boyunca, gösterilen şekilde devam eder. Şekil 3.4, bu adımları şekilsel olarak da göstermektedir.

YSA'nın öğrenme aşamasında dikkat edilmesi gerekenler şunlardır:

1. Ağın eğitimi sırasında dikkat edilmesi gereken noktalardan biri, **eğitim setinin** doğru seçilmesidir. Yapılan araştırmalarda, ağrı hisseden insan yüzünün belirli bir karakteristiği bulunmaktadır. Kaşların şeklinin değişmesi, alnın kırışması, elmacıkların tümsekleşmesi vs. Eğitim aşamasında, az ağrı derecelerini gösteren fotoğrafların doğru seçilmemesinin, ağın yanlış öğrenmesine ve dolayısıyla yanlış sonuçlar üretmesine sebep olabileceğini unutmamak gerekmektedir.
2. Uygun **parametrelerin** seçilememesi, bazen ağın performansında düşmeye bazen ise ağın yanlış çıkış üretmesine sebep olabilmektedir.
3. Ağa gösterilecek fotoğrafta yapılacak yanlış bir **ön işlem**, yüz ifadesinde kayba neden olur. Bu durum da ağın fotoğrafı tanımasına engel olabilmektedir.
4. Ağa verilecek **görüntünün kalitesi** ve **ışıklandırmasının** doğru yapılması, ağın gösterilen fotoğraftaki ifadeyi doğru tanıması açısından çok önemlidir. Çok az ya da aşırı ışıklandırılmış bir fotoğrafta, yüzün detaylarına ulaşmak mümkün olmayabilmektedir.
5. Ağa sunulacak fotoğrafta **yüzün konumlandırılması** doğru yapılmalıdır. Fotoğraf çekildiği sıradaki objektifin açısı, eğitim ve test fotoğraflarında her zaman aynı olmalıdır.
6. Ağa sunulan fotoğraftaki **yüzü kapatacak** herhangi bir cisim (örneğin şapka), yüze dökülen kâküller veya uzun sakal-bıyık, ağın doğru tanıma yapmasına engel olabilmektedir.

7. Yüzün herhangi bir yerinin **boyanmış olması** da, YSA'nın yüzdeki ifadeyi tanımasını zorlaştıran etkenlerden biri olarak gösterilebilmektedir.



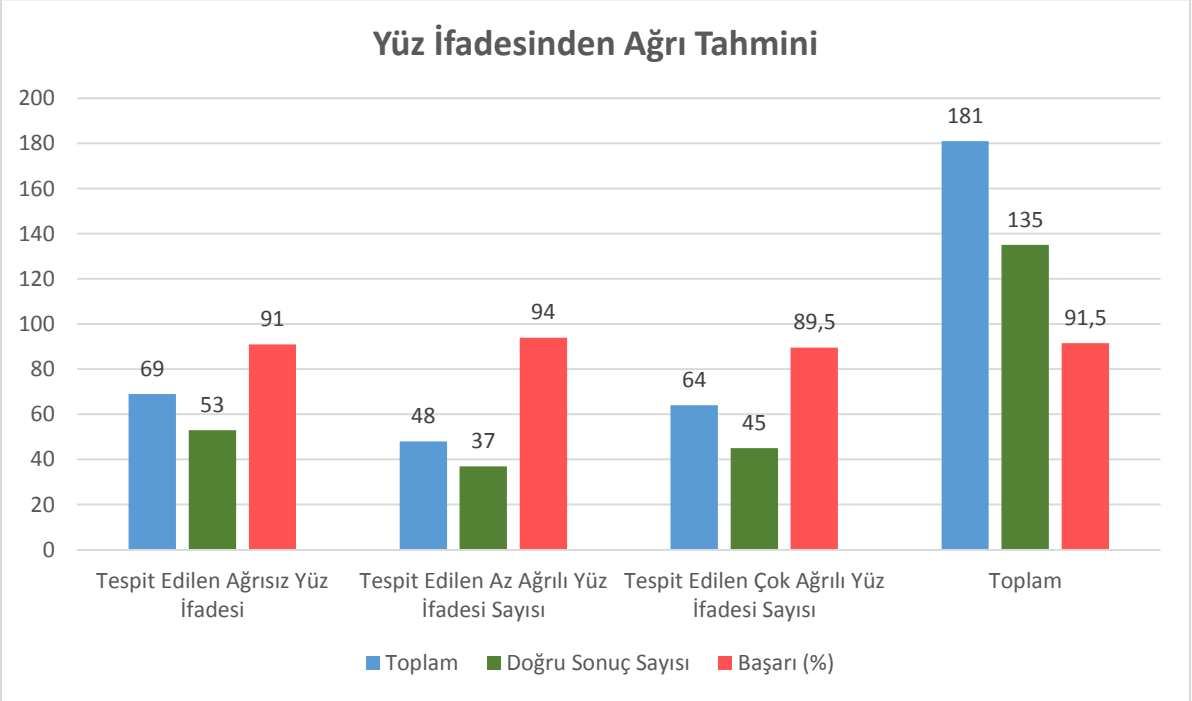
Şekil 3. 4 Yüzdeki ağrı ifadesi tahmini uygulamasının algoritma şeması.

Eğitim setindeki tüm fotoğraflar, ağın çıkışı belirlenen hata oranına ulaşıncaya kadar ağa gösterildikten sonra eğitim aşaması bitmiş ve sıra test aşamasında gelmiştir.

Test için seçilen fotoğrafların her biri teker teker ağa gösterilmiş ve ağın çıkışları değerlendirmek üzere akademisyen ve çeşitli kurumlarda çalışan 5 farklı kişiye sorularak gelen sonuçların aritmetik ortalaması alınmış olup şu şekildedir:

Tablo 3. 1 Ağrı tahmini test sonuçları.

	Toplam Tahmin Sayısı	**Yaklaşık Doğru Bilinen Sonuç Sayısı**	**Yaklaşık Başarı (%)**
Tespit Edilen Ağrısız Yüz İfadesi	69	53	91
Tespit Edilen Az Ağrılı Yüz İfadesi	48	37	94
Tespit Edilen Çok Ağrılı Yüz İfadesi	64	45	89,5
Toplam	**181**	**135**	**91,5**



BÖLÜM 4. SONUÇ VE TARTIŞMA

Teknolojinin artan gücüyle birlikte, günümüzde birçok sektörde bilgisayarların kullanımı çok yaygın hale gelmiştir. Sağlık sektöründe de teknoloji, eskiden beridir olmazsa olmaz denecek bir konumdadır. Yapılan bu çalışma da yapay zekânın gücünün sağlık alanında kullanımına ilişkindir.

Yüz ifadesinden ağrı tespitinin bilgisayarlar yardımıyla yapılmasıyla, hem sağlık çalışanlarının zor hastalarla iletişimini kolaylaştırmaya, hem hasta bakımı noktasında hasta yakınlarının maddi yükünü azaltmaya hem de kendini ifade etmekte zorluk çeken hastaların yaşam kalitesini arttırmaya çalışılmıştır.

Tablo 3.1 ve Şekil 3.5 incelendiğinde, ağrısız yüzlerde %91, az ağrılı yüzlerde %94, çok ağrılı yüzlerde %89,5 ve toplamda yaklaşık %91,5'luk bir başarı elde edilmiştir.

Sonuçta elde edilen başarı oranı kötü olmamakla birlikte, çıkan hatalı sonuçların neden **kaynaklanabileceği** hakkında şunlar söylenebilir:

1. Eğitimde kullanılan veri setinin yeterli boyutta olamamış olması
2. Eğitim için seçilen veri setinin içeriğinin her bir yüz ifadesi için doğru seçimler olamamış olması
3. Eğitim ve test setlerinde kullanılan fotoğrafların çözünürlük kalitesinin, ışık miktarının ve fotoğraf açısının yeterli olamamış olması
4. Kullanılan ağ türünün, sınıflandırmada yeterince başarılı olamaması
5. Ağın eğitimi aşamasında kullanılan parametrelerin, en doğru sonuçları verecek nitelikte olmayışı veya ağ eğitilirken iterasyon sayısının fazla ya da eksik oluşu
6. Ön işlem aşamasında yapılan uygulamaların veri kaybına neden olması

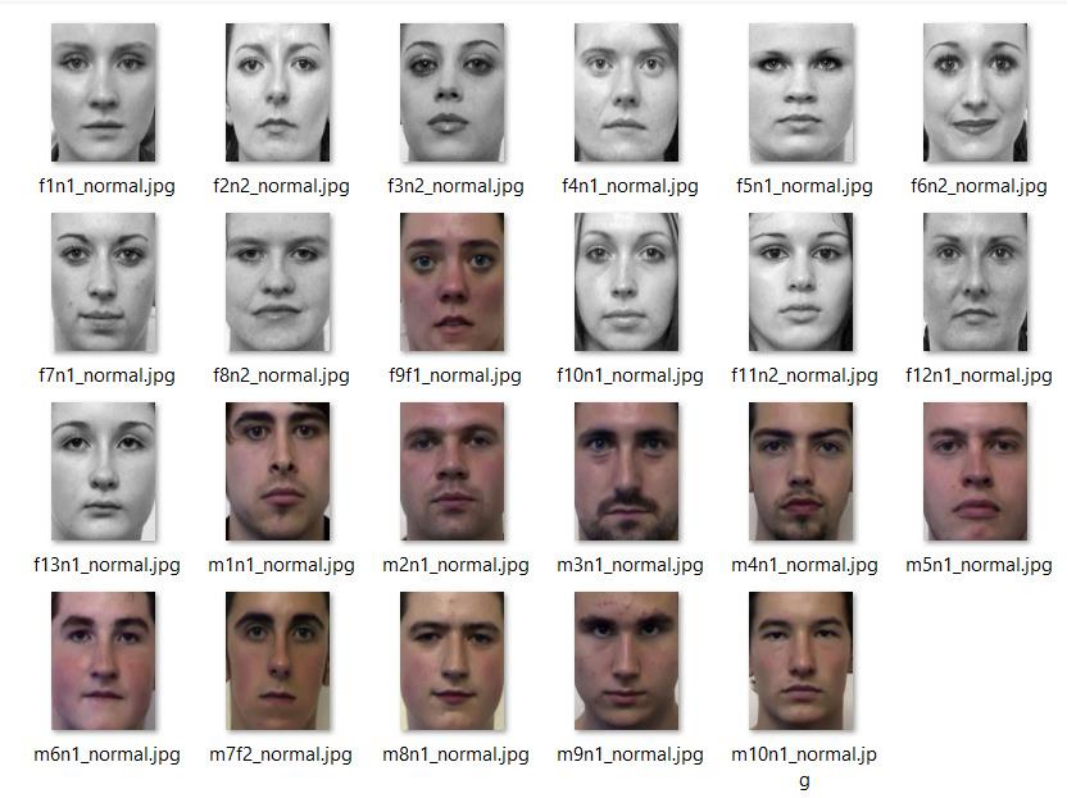
Bunlarla birlikte sonuçlar değerlendirilirken, ağrı ifadesinin yüze yansımasının her insanda farklı şekillerde olabileceği ve her insanın ağrı eşiğinin ve ağrıyı tolere edebilme seviyelerinin farklı olabileceğini de hatırlamak gerekmektedir.

Sonraki yapılacak çalışmada, sınıflandırma konusunda daha başarılı olabilecek başka bir teknik (Örneğin Deep Learning) ve fotoğraf yerine canlı kamera görüntüsü kullanımıyla birlikte, bu uygulamanın daha başarılı olabileceği düşünülmektedir.

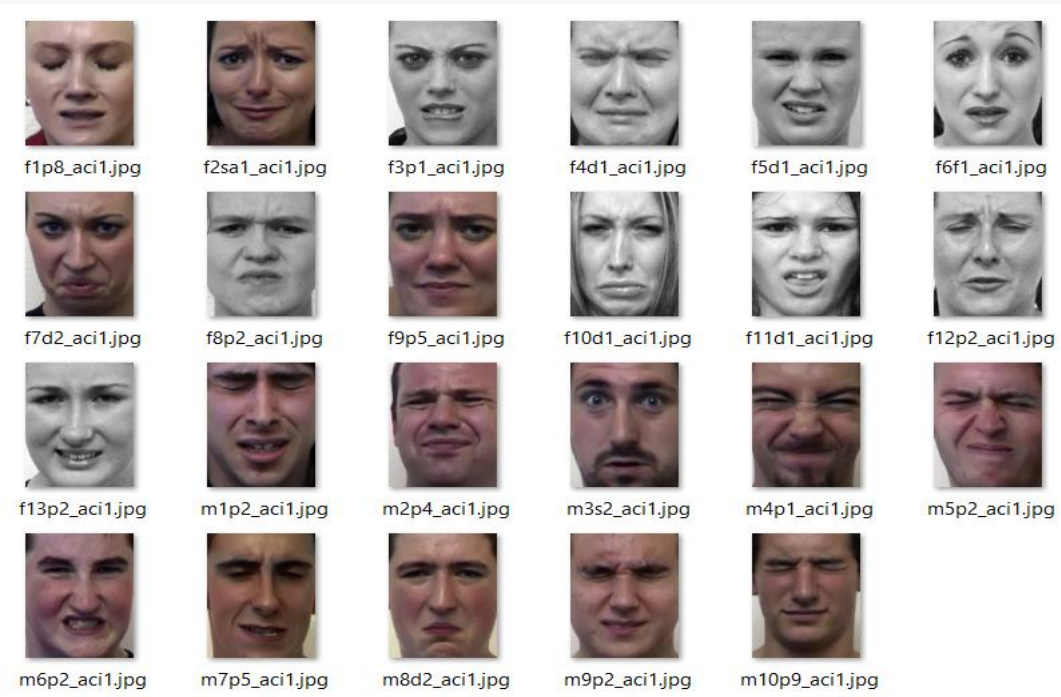
BÖLÜM 5. EKLER

5.1 EK-1 Eğitim ve Test Aşamasında Kullanılan Fotoğraflar

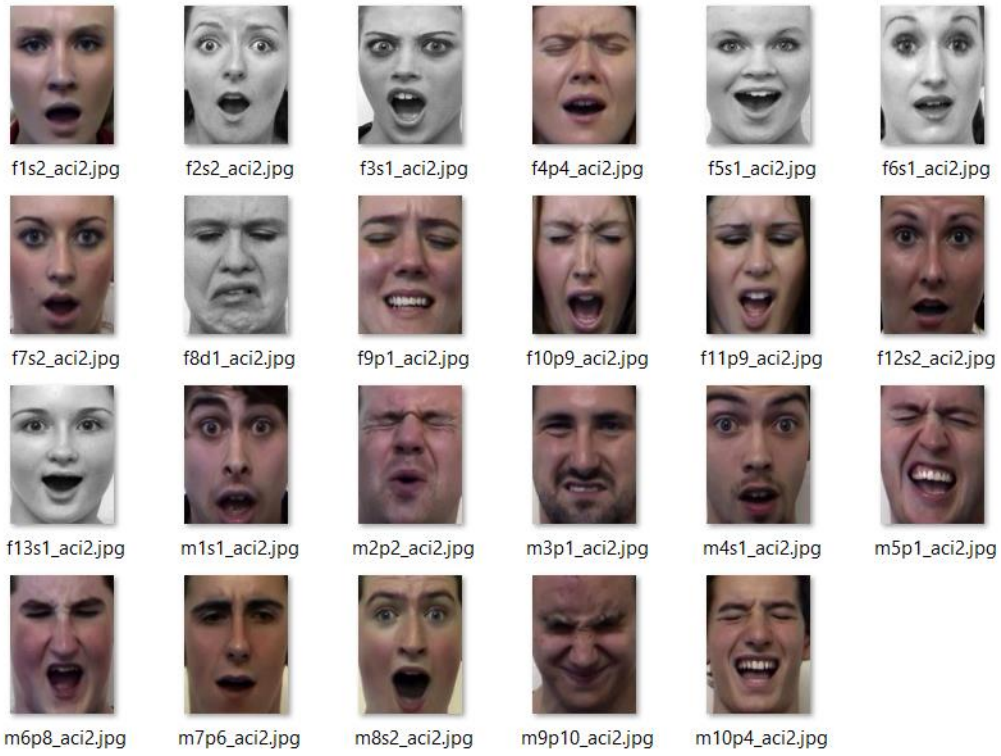
Yapay sinir ağının eğtimi ve test aşamasında kullanılan fotoğraflar şu şekildedir (URL-9, 2019).



Şekil 5. 1 Ağın eğitiminde kullanılan normal yüz ifadeleri.



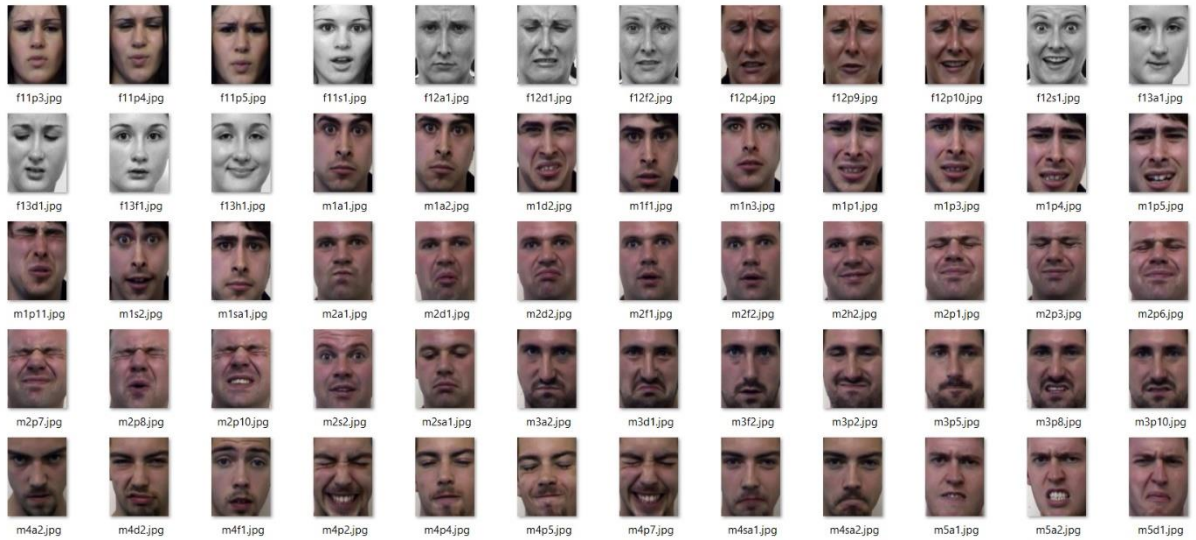
Şekil 5. 2 Ağın eğitiminde kullanılan az ağrılı yüz ifadeleri.



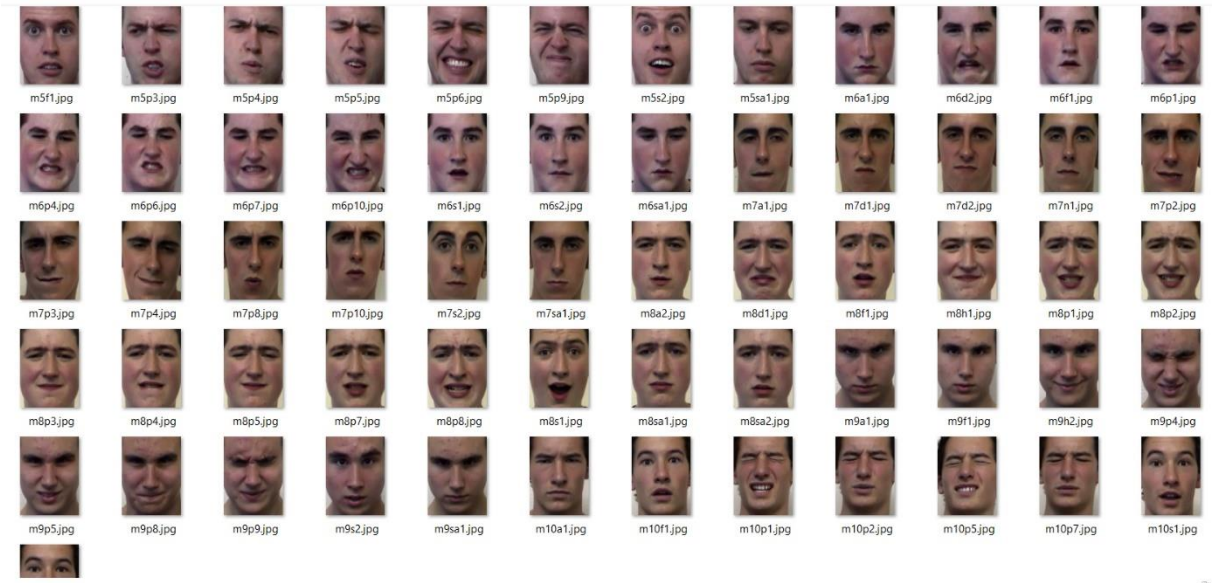
Şekil 5. 3 Ağın eğitiminde kullanılan çok ağrılı yüz ifadeleri.



Şekil 5. 4 Ağı test etmekte kullanılan fotoğraflar-1



Şekil 5. 5 Ağı test etmekte kullanılan fotoğraflar-2



Şekil 5. 6 Ağı test etmekte kullanılan fotoğraflar-3

5.2 EK-2 Kaynak Kodlar

Uygulamada kullanılan kaynak kodlar şu şekildedir;

1 - egit.m
2 - create_network.m
3 - feed_forward.m
4 - back_propagation.m
5 - foto_resize.m
6 - ortalama.m
7 - test.m
8 – koselerikaldir.m

1. egit.m Fonksiyonu Kodları:

```
% belleği ve komut penceresini temizleme

clc;
clear all;

% Network oluşturma fonksiyonu
er_create_network;
maxhata = 0.1;
fprintf('Eğitim işlemi başladı....');
fprintf('%s\n', datestr(now,'HH:MM:SS.FFF'));
toplamiterasyon=0;
dongutekrarsayisi = 1;
max_iterasyonsayisi = 1;
```

```
for aa =1:dongutekrarsayisi % ağın öğreneceği bütün yüzleri
tekrar etme sayısı
    fprintf('__________%d .nci tekrar_____________\n', aa);
    for ii=1:69 % 3x23= 69 tane eğitim verisi var
        fotopath = paths{ii,1};

        iterasyon = 0;

        Beklenen(:) = 0;

%ağın hangi ifade için eğitildiğine göre beklenen çıktı değeri
1 yapılır
        if mod(ii,3) == 1 Beklenen(1)= 1;
        elseif  mod(ii,3) == 2 Beklenen(2)= 1;
        else Beklenen(3)= 1;
        end % if ... elseif sonu

        %gelen fotoğraf 120x90 moduna sokuluyor

        Giris = func_er_foto_resize(fotopath);

        %gelen 120x90 lik matris 210x1 lik vektore cevriliyor
        Girisf = func_er_ortalama(Giris);


       while 1

            %feed forward ile ağı ileri doğru çalıştırıp
sonuçlar alındı
            [Cif,Cof] =
func_er_feed_forward(Girisf,Wi,Wib,Wo,Wob,Bi,Bo,t1,t2,arakatma
nsayisi,cikissayisi);
            Ci = Cif;
            Co = Cof;


     %back propagation fonksiyonu çalıştırıp sonuçlar alındı
            [Wif,Wibf,Wof,Wobf,Ef] =
func_er_back_propagation(Girisf,Ci,Bi,Co,Bo,Wi,Wib,Wo,Wob,Bekl
enen,egitimks,momentumks);
            Wi =  Wif;
            Wib = Wibf;
            Wo = Wof;
            Wob = Wobf;
            E = Ef;


            E_1 = [E_1 Beklenen' Co' E'];

     %çıkış hatası kontrolü
```

```
        if (abs(E(1)) < maxhata && abs(E(2)) < maxhata &&
abs(E(3)) < maxhata) break; end;
        if (iterasyon > max_iterasyonsayisi)
            fprintf('%d tekrarda öğrenilemeyen foto_________%s
\n',max_iterasyonsayisi,fotopath);
            break;
        end;
        iterasyon =  iterasyon +1;
        end % while  sonu
        toplamiterasyon = toplamiterasyon + iterasyon ;
        fprintf('%d iterasyon,toplam __: %d  \n',iterasyon,
toplamiterasyon);

    end % for sonu
end %for aa sonu
fprintf('Egitim islemi %d iterasyonda tamamlandı :
',toplamiterasyon);
fprintf(' %s\n', datestr(now,'HH:MM:SS.FFF'));
fprintf('Her 3 Çıkışın Hatası:%d , %d , %d
\n',E(1),E(2),E(3));
```

2. create_network.m Fonksiyonu

```
%eğtim için kullanılacak fotoğrafların
 belirlenmesi.

paths ={'imdata/er_photos/f1n1_normal.jpg' ;
'imdata/er_photos/f1p8_aci1.jpg' ;
    'imdata/er_photos/f1s2_aci2.jpg' ;

    'imdata/er_photos/f2n2_normal.jpg' ;
'imdata/er_photos/f2sa1_aci1.jpg' ;
    'imdata/er_photos/f2s2_aci2.jpg' ;

    'imdata/er_photos/f3n2_normal.jpg' ;
'imdata/er_photos/f3p1_aci1.jpg' ;
    'imdata/er_photos/f3s1_aci2.jpg' ;

    'imdata/er_photos/f4n1_normal.jpg' ;
'imdata/er_photos/f4d1_aci1.jpg' ;
    'imdata/er_photos/f4p4_aci2.jpg' ;

    'imdata/er_photos/f5n1_normal.jpg' ;
'imdata/er_photos/f5d1_aci1.jpg' ;
    'imdata/er_photos/f5s1_aci2.jpg' ;

    'imdata/er_photos/f6n2_normal.jpg' ;
'imdata/er_photos/f6f1_aci1.jpg' ;
    'imdata/er_photos/f6s1_aci2.jpg' ;
```

```
    'imdata/er_photos/f7n1_normal.jpg' ;
'imdata/er_photos/f7d2_aci1.jpg' ;
    'imdata/er_photos/f7s2_aci2.jpg' ;

    'imdata/er_photos/f8n2_normal.jpg' ;
'imdata/er_photos/f8p2_aci1.jpg' ;
    'imdata/er_photos/f8d1_aci2.jpg' ;

    'imdata/er_photos/f9f1_normal.jpg' ;
'imdata/er_photos/f9p5_aci1.jpg' ;
    'imdata/er_photos/f9p1_aci2.jpg' ;

    'imdata/er_photos/f10n1_normal.jpg' ;
'imdata/er_photos/f10d1_aci1.jpg' ;
    'imdata/er_photos/f10p9_aci2.jpg' ;

    'imdata/er_photos/f11n2_normal.jpg' ;
'imdata/er_photos/f11d1_aci1.jpg' ;
    'imdata/er_photos/f11p9_aci2.jpg' ;

    'imdata/er_photos/f12n1_normal.jpg' ;
'imdata/er_photos/f12p2_aci1.jpg' ;
    'imdata/er_photos/f12s2_aci2.jpg' ;

    'imdata/er_photos/f13n1_normal.jpg' ;
'imdata/er_photos/f13f1_aci1.jpg' ;
    'imdata/er_photos/f13s1_aci2.jpg';

    'imdata/er_photos/m1n1_normal.jpg' ;
'imdata/er_photos/m1p2_aci1.jpg' ;
    'imdata/er_photos/m1s1_aci2.jpg';

    'imdata/er_photos/m2n1_normal.jpg' ;
'imdata/er_photos/m2p4_aci1.jpg' ;
    'imdata/er_photos/m2p2_aci2.jpg';

    'imdata/er_photos/m3n1_normal.jpg' ;
'imdata/er_photos/m3s2_aci1.jpg' ;
    'imdata/er_photos/m3p1_aci2.jpg';

    'imdata/er_photos/m4n1_normal.jpg' ;
'imdata/er_photos/m4p1_aci1.jpg' ;
    'imdata/er_photos/m4s1_aci2.jpg';

    'imdata/er_photos/m5n1_normal.jpg' ;
'imdata/er_photos/m5p2_aci1.jpg' ;
    'imdata/er_photos/m5p1_aci2.jpg';

    'imdata/er_photos/m6n1_normal.jpg' ;
'imdata/er_photos/m6p2_aci1.jpg' ;
```

```
    'imdata/er_photos/m6p8_aci2.jpg';

    'imdata/er_photos/m7f2_normal.jpg' ;
'imdata/er_photos/m7p5_aci1.jpg' ;
    'imdata/er_photos/m7p6_aci2.jpg';

    'imdata/er_photos/m8n1_normal.jpg' ;
'imdata/er_photos/m8d2_aci1.jpg' ;
    'imdata/er_photos/m8s2_aci2.jpg';

    'imdata/er_photos/m9n1_normal.jpg' ;
'imdata/er_photos/m9p2_aci1.jpg' ;
    'imdata/er_photos/m9p10_aci2.jpg';

    'imdata/er_photos/m10n1_normal.jpg' ;
'imdata/er_photos/m10p9_aci1.jpg' ;
    'imdata/er_photos/m10p4_aci2.jpg'    };


girissayisi=210;
arakatmansayisi= 210;
cikissayisi =3;
egitimks = 0.3;
momentumks = 0.7;
t1 = 5;
t2 = 10;
Bi = -0.1;
Bo = -0.8;

%ara katman ile çıkış katmanı arasındaki Wo ve Wob rastgele
ağırlıkların belirlenmesi
Wo = rand(arakatmansayisi,cikissayisi);
Wob = rand(arakatmansayisi,cikissayisi);

%giriş ile ara katman arasıdaki Wi ve Wib rastgele
ağırlıkların belirlenmesi
Wi = rand(girissayisi,arakatmansayisi);
Wib = rand(girissayisi,arakatmansayisi);


%giris katmani olusturma
Giris = zeros(girissayisi,1);

%giris ile ara katman arası degerler, ara katman nöron sayısı
m =210
Ni = linspace(0,0,arakatmansayisi); % linspace(x1,x2,N)komutu
[x1,x2] aralığını N noktaya böler
Ni_1 = linspace(0,0,arakatmansayisi);
Ci = linspace(0,0,arakatmansayisi);
Ci_1 = linspace(0,0,arakatmansayisi);
Deltai = linspace(0,0,arakatmansayisi);
```

```
DeltaWi = zeros(girissayisi,arakatmansayisi);

%çıkış katmanı
No = linspace(0,0,cikissayisi);
No_1 = linspace(0,0,cikissayisi);
Co = linspace(0,0,cikissayisi);
Co_1 = linspace(0,0,cikissayisi);
Deltao = linspace(0,0,cikissayisi);
Beklenen = linspace(0,0,cikissayisi);
E = linspace(0,0,cikissayisi);
E_1 = zeros(cikissayisi,1);

DeltaWo = zeros(arakatmansayisi,cikissayisi); % çıktı katmanı
ile ara katman arasındakı hata degisim miktarı
fprintf('Network olusturma basarılı \n');
```

3. feed_forward.m

```
% ileri besleme fonksiyonunu
function  [Cii,Coo]=
func_er_feed_forward(GirisVektoru,Wii,Wib,Woo,Wob,Bi,Bo,t1,t2,
arakatmansay,cikissay)

vektorboyu = length(GirisVektoru);
Cii = linspace(0,0,arakatmansay);
Nii = linspace(0,0,arakatmansay);
Noo = linspace(0,0,cikissay);
Coo = linspace(0,0,cikissay);


for i=1:arakatmansay % 210
    for j=1:vektorboyu % 210
        Nii(i) = Nii(i)+GirisVektoru(j)*Wii(j,i)+Bi*Wib(j,i);
    end

        Cii(i)=1/(1+exp(-Nii(i)/t1));%her bulunan Ni net
değeri için bir Ci ara katman düğümü cikiş değeri hesaplanır
%net değerleri büyük çıktığından, bir t1 sabitine bölünür.

end

%çıkış net değerlerinin hesaplanması
for i=1:cikissay
    for j=1:arakatmansay
        Noo(i) = Noo(i)+Cii(j)*Woo(j,i)+Bo*Wob(j,i);
    end % for j sonu
```

```
    %bulunan her No değeri için bir Co cikiş değeri hesaplanır
%net değerleri büyük çıktığından, bir t2 sabitine bölünür.
        Coo(i)=1/(1+exp(-Noo(i)/t2));

end % for i sonu
end %fonksiyon sonu
```

4. back_propagation.m Fonksiyonu

```
function  [Wii,Wiib,Woo,Woob,E_]=
func_er_back_propagation(Giriss,Cii,Bii,Coo,Boo,Wii,Wiib,Woo,W
oob,Beklenenn,egitimks,momentumks)
%_____________________________________________________________
%Ağırlıklardaki Delta değişim miktarları
%Back Propagation yöntemiyle bulunur

%çıktı katmanı ile ara katman(hidden layer) arasındaki
%ağırlıklara ait delta değişim oranları bulunur

girissayi = length(Giriss);
arakatmansayi = length(Cii);
cikissayi = length(Coo);

Ni_ = linspace(0,0,arakatmansayi);
Deltai_ = linspace(0,0,arakatmansayi);
DeltaWi_ = zeros(girissayi,arakatmansayi);

%çıkış katmanı
No_ = linspace(0,0,cikissayi);
Deltao_ = linspace(0,0,cikissayi);
DeltaWo_ = zeros(arakatmansayi,cikissayi);

E_ = linspace(0,0,cikissayi);

%Çıkış Hataları hesaplanır
for i=1:cikissayi
    E_(i)=Beklenenn(i) - Coo(i);
    Deltao_(i) = Coo(i)*(1-Coo(i))*E_(i);
end

 %orta katmandaki Deltai, Wo ve Wob değerleri hesaplanıyor
for i = 1:cikissayi
    for j = 1:arakatmansayi
        DeltaWo_(j,i)= egitimks*Deltao_(i)*Cii(j) +
momentumks*DeltaWo_(j,i);
        Woo(j,i)=Woo(j,i)+DeltaWo_(j,i);
        Woob(j,i)=Woob(j,i)+DeltaWo_(j,i);
```

```
    end
end
%hücrelerden gelen hataları ağırlıklarla çarpmak için t
değişkeni kullanıldı
t = 0;
%Gizli katman Delta değişim oranları bulunur
for i = 1:arakatmansayi % 210
    for j = 1:cikissayi % 3
        t = t+Deltao_(j)*Woo(i,j);
    end
    Deltai_(i) = Cii(i)*(1-Cii(i))*t*10;
    t =0;
end



%ara katman ile giriş katmanı arasındaki DeltaWi ağırlık
değişimi bulunup Wi ve Wib ağırlıklar güncellenir
for i=1:arakatmansayi % 210 adet
    for j=1:girissayi % 210 adet
        DeltaWi_(j,i)=egitimks*Deltai_(i)*Giriss(j) +
momentumks*DeltaWi_(j,i);
        Wii(j,i) = Wii(j,i) + DeltaWi_(j,i);
        Wiib(j,i) = Wiib(j,i) + DeltaWi_(j,i);
    end %end inner-for
end %end outer-for
end %fonksiyon sonu
```

5. foto_resize.m Fonksiyonu

```
function girisdizi = func_er_foto_resize(fotopath)

imageorj=imread(fotopath); % Diskten fotoğraf okunuyor.
imageorj=rgb2gray(imageorj); % Fotoğraf gri tona çevrilir. (2-
Boyut)

level=graythresh(imageorj); % fotoğrafa göre esik değeri
belirlenir.
bw=im2bw(imageorj,level); % Fotoğraf, eşik değerine göre siyah
beyaza çevrilir.

% Gelen herhangi bir boyuttaki fotoğrafı 120x90 boyutuna sokar
bw = imresize(bw, [120 90]);

imageorj = imresize(imageorj, 0.5); %Resmimizin ebatlarını %50
oranında sıkıştırılıyor
```

```
bw = func_er_koselerikaldir(bw); % köşelerdeki gereksiz
kısımlar temizleniyor
bw=imcomplement(bw); % Fotoğraf negatife çevriliyor.
bw = im2double(bw); % mantıksal diziyi double değerlere
donuşturuyoruz

girisdizi = bw;

end
```

6. ortalama.m Fonksiyonu

```
function vektor = func_er_ortalama(dizi)
%bu fonksiyona parametre gelen gelen diziyi 1/10 oranında
sadeleştirip, bu dizinin satır ve sütun sayısının toplamı
kadar (x1 vektörü ve y1 vektörü) bir adet vektör geri
döndürür. yani x1*y1/10 boyutunda bir dizi geri döndürür.

    [satirsay,sutunsay]=size(dizi);
    for i =1:satirsay
        altmatris= dizi(i:i,1:sutunsay); % matrisinin satır
sayısı
        yvektor(i)= mean(altmatris);
         altmatris = 0;
    end % buraya kadar bütün satırların tek tek ortalamasını
alır
        %[toplamsütundeğeri/sütunsayısı] orn: 15/20 =0,75 gibi
    y1 = yvektor;
    altmatris = 0;
    for i =1:sutunsay
        altmatris= dizi(1:satirsay,i:i); % matrisinin sütun
sayısı
        xvektor(i)= mean(altmatris);
         altmatris = 0;
    end % buraya kadar bütün sütunların teker teker
ortalamasını alır
    d1 = xvektor;
    vektor = [y1 d1];
end
```

7. test.m Fonksiyonu

```
pathss ={'imdata/er_test/f1a2.jpg'      ;
        'imdata/er_test/f1d2.jpg'       ;
        'imdata/er_test/f1d3.jpg'       ;
        'imdata/er_test/f1f2.jpg'       ;
        'imdata/er_test/f1p2.jpg'       ;
        'imdata/er_test/f1p5.jpg'       ;
        'imdata/er_test/f1s1.jpg'       ;
        'imdata/er_test/f2a2.jpg'       ;
        'imdata/er_test/f2d2.jpg'       ;
```

```
'imdata/er_test/f2f1.jpg'       ;
'imdata/er_test/f2p3.jpg'       ;
'imdata/er_test/f2p10.jpg'      ;
'imdata/er_test/f3a2.jpg'       ;
'imdata/er_test/f3d1.jpg'       ;
'imdata/er_test/f3f1.jpg'       ;
'imdata/er_test/f3p9.jpg'       ;
'imdata/er_test/f3p10.jpg'      ;
'imdata/er_test/f3sa2.jpg'      ;
'imdata/er_test/f4a1.jpg'       ;
'imdata/er_test/f4d2.jpg'       ;
'imdata/er_test/f4f1.jpg'       ;
'imdata/er_test/f4p3.jpg'       ;
'imdata/er_test/f4p5.jpg'       ;
'imdata/er_test/f4p6.jpg'       ;
'imdata/er_test/f4p8.jpg'       ;
'imdata/er_test/f4p10.jpg'      ;
'imdata/er_test/f4s1.jpg'       ;
'imdata/er_test/f5a1.jpg'       ;
'imdata/er_test/f5p5.jpg'       ;
'imdata/er_test/f6a1.jpg'       ;
'imdata/er_test/f6h2.jpg'       ;
'imdata/er_test/f6p2.jpg'       ;
'imdata/er_test/f7a1.jpg'       ;
'imdata/er_test/f7d1.jpg'       ;
'imdata/er_test/f7f1.jpg'       ;
'imdata/er_test/f7p1.jpg'       ;
'imdata/er_test/f7p10.jpg'      ;
'imdata/er_test/f7s1.jpg'       ;
'imdata/er_test/f8a1.jpg'       ;
'imdata/er_test/f8f1.jpg'       ;
'imdata/er_test/f8s1.jpg'       ;
'imdata/er_test/f9a1.jpg'       ;
'imdata/er_test/f9p2.jpg'       ;
'imdata/er_test/f9p3.jpg'       ;
'imdata/er_test/f9p7.jpg'       ;
'imdata/er_test/f9p8.jpg'       ;
'imdata/er_test/f9p10.jpg'      ;
'imdata/er_test/f9s2.jpg'       ;
'imdata/er_test/f9sa1.jpg'      ;
'imdata/er_test/f10a2.jpg'      ;
'imdata/er_test/f10d2.jpg'      ;
'imdata/er_test/f10f1.jpg'      ;
'imdata/er_test/f10h1.jpg'      ;
'imdata/er_test/f10p4.jpg'      ;
'imdata/er_test/f10s2.jpg'      ;
'imdata/er_test/f11a2.jpg'      ;
'imdata/er_test/f11f1.jpg'      ;
'imdata/er_test/f11n1.jpg'      ;
'imdata/er_test/f11p1.jpg'      ;
'imdata/er_test/f11p3.jpg'      ;
```

```
'imdata/er_test/f11p4.jpg'      ;
'imdata/er_test/f11p5.jpg'      ;
'imdata/er_test/f11s1.jpg'      ;
'imdata/er_test/f12a1.jpg'      ;
'imdata/er_test/f12d1.jpg'      ;
'imdata/er_test/f12f2.jpg'      ;
'imdata/er_test/f12p4.jpg'      ;
'imdata/er_test/f12p9.jpg'      ;
'imdata/er_test/f12p10.jpg'     ;
'imdata/er_test/f12s1.jpg'      ;
'imdata/er_test/f13a1.jpg'      ;
'imdata/er_test/f13d1.jpg'      ;
'imdata/er_test/f13h1.jpg'      ;
'imdata/er_test/f13p2.jpg'      ;
'imdata/er_test/m1a1.jpg'       ;
'imdata/er_test/m1a2.jpg'       ;
'imdata/er_test/m1d2.jpg'       ;
'imdata/er_test/m1f1.jpg'       ;
'imdata/er_test/m1n3.jpg'       ;
'imdata/er_test/m1p1.jpg'       ;
'imdata/er_test/m1p3.jpg'       ;
'imdata/er_test/m1p4.jpg'       ;
'imdata/er_test/m1p5.jpg'       ;
'imdata/er_test/m1p11.jpg'      ;
'imdata/er_test/m1s2.jpg'       ;
'imdata/er_test/m1sa1.jpg'      ;
'imdata/er_test/m2a1.jpg'       ;
'imdata/er_test/m2d1.jpg'       ;
'imdata/er_test/m2d2.jpg'       ;
'imdata/er_test/m2f1.jpg'       ;
'imdata/er_test/m2f2.jpg'       ;
'imdata/er_test/m2h2.jpg'       ;
'imdata/er_test/m2p1.jpg'       ;
'imdata/er_test/m2p3.jpg'       ;
'imdata/er_test/m2p6.jpg'       ;
'imdata/er_test/m2p7.jpg'       ;
'imdata/er_test/m2p8.jpg'       ;
'imdata/er_test/m2p10.jpg'      ;
'imdata/er_test/m2s2.jpg'       ;
'imdata/er_test/m2sa1.jpg'      ;
'imdata/er_test/m3a2.jpg'       ;
'imdata/er_test/m3d1.jpg'       ;
'imdata/er_test/m3f2.jpg'       ;
'imdata/er_test/m3p2.jpg'       ;
'imdata/er_test/m3p5.jpg'       ;
'imdata/er_test/m3p8.jpg'       ;
'imdata/er_test/m3p10.jpg'      ;
'imdata/er_test/m4a2.jpg'       ;
'imdata/er_test/m4d2.jpg'       ;
'imdata/er_test/m4f1.jpg'       ;
'imdata/er_test/m4p2.jpg'       ;
```

```
'imdata/er_test/m4p4.jpg'       ;
'imdata/er_test/m4p5.jpg'       ;
'imdata/er_test/m4p7.jpg'       ;
'imdata/er_test/m4sa1.jpg'      ;
'imdata/er_test/m4sa2.jpg'      ;
'imdata/er_test/m5a1.jpg'       ;
'imdata/er_test/m5a2.jpg'       ;
'imdata/er_test/m5d1.jpg'       ;
'imdata/er_test/m5f1.jpg'       ;
'imdata/er_test/m5p3.jpg'       ;
'imdata/er_test/m5p4.jpg'       ;
'imdata/er_test/m5p5.jpg'       ;
'imdata/er_test/m5p6.jpg'       ;
'imdata/er_test/m5p9.jpg'       ;
'imdata/er_test/m5s2.jpg'       ;
'imdata/er_test/m5sa1.jpg'      ;
'imdata/er_test/m6a1.jpg'       ;
'imdata/er_test/m6d2.jpg'       ;
'imdata/er_test/m6f1.jpg'       ;
'imdata/er_test/m6p1.jpg'       ;
'imdata/er_test/m6p4.jpg'       ;
'imdata/er_test/m6p6.jpg'       ;
'imdata/er_test/m6p7.jpg'       ;
'imdata/er_test/m6p10.jpg'      ;
'imdata/er_test/m6s1.jpg'       ;
'imdata/er_test/m6s2.jpg'       ;
'imdata/er_test/m6sa1.jpg'      ;
'imdata/er_test/m7a1.jpg'       ;
'imdata/er_test/m7d1.jpg'       ;
'imdata/er_test/m7d2.jpg'       ;
'imdata/er_test/m7n1.jpg'       ;
'imdata/er_test/m7p2.jpg'       ;
'imdata/er_test/m7p3.jpg'       ;
'imdata/er_test/m7p4.jpg'       ;
'imdata/er_test/m7p8.jpg'       ;
'imdata/er_test/m7p10.jpg'      ;
'imdata/er_test/m7s2.jpg'       ;
'imdata/er_test/m7sa1.jpg'      ;
'imdata/er_test/m8a2.jpg'       ;
'imdata/er_test/m8d1.jpg'       ;
'imdata/er_test/m8f1.jpg'       ;
'imdata/er_test/m8h1.jpg'       ;
'imdata/er_test/m8p1.jpg'       ;
'imdata/er_test/m8p2.jpg'       ;
'imdata/er_test/m8p3.jpg'       ;
'imdata/er_test/m8p4.jpg'       ;
'imdata/er_test/m8p5.jpg'       ;
'imdata/er_test/m8p7.jpg'       ;
'imdata/er_test/m8p8.jpg'       ;
'imdata/er_test/m8s1.jpg'       ;
'imdata/er_test/m8sa1.jpg'      ;
```

```
        'imdata/er_test/m8sa2.jpg'      ;
        'imdata/er_test/m9a1.jpg'       ;
        'imdata/er_test/m9f1.jpg'       ;
        'imdata/er_test/m9h2.jpg'       ;
        'imdata/er_test/m9p4.jpg'       ;
        'imdata/er_test/m9p5.jpg'       ;
        'imdata/er_test/m9p8.jpg'       ;
        'imdata/er_test/m9p9.jpg'       ;
        'imdata/er_test/m9s2.jpg'       ;
        'imdata/er_test/m9sa1.jpg'      ;
        'imdata/er_test/m10a1.jpg'      ;
        'imdata/er_test/m10f1.jpg'      ;
        'imdata/er_test/m10p1.jpg'      ;
        'imdata/er_test/m10p2.jpg'      ;
        'imdata/er_test/m10p5.jpg'      ;
        'imdata/er_test/m10p7.jpg'      ;
        'imdata/er_test/m10s1.jpg'      ;
        'imdata/er_test/m10s2.jpg'      ;
        'imdata/er_test/m10sa1.jpg'    };



fprintf('Otomatik Test işlemi başladı');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ij=1:size(pathss) % 3x23= 69 tane eğitim verisi var
        fotopathh = pathss{ij,1};
fprintf('_____________________________\n' );
      fprintf('Fotoğraf : %s\n', pathss{ij,1});

        %gelen foto 120x90 moduna sokuluyor

        Girish = func_er_foto_resize(fotopathh);

        %gelen 120x90 lik matrisi 210x1 lik vektore
ceviriyoruz
        Girisff = func_er_ortalama(Girish);

        %feed forward ile ağı ileri doğru çalıştırılıp çıkış
gözlemleniyor.
       [Cif,Cof] =
func_er_feed_forward(Girisff,Wi,Wib,Wo,Wob,Bi,Bo,t1,t2,girissa
yisi,cikissayisi);
        Ci = Cif;
        Co = Cof;


enBuyuk=Co(1);
hangiifade =1;
for i=2:length(Co) %3 elemanlı dizi
    if Co(i)>enBuyuk
        enBuyuk=Co(i);
```

```
        hangiifade=i;
    end
end

 % test sonucuna göre ağa gösterilen fotoğrafın, ağın
çıktısına bakılarak hangi kategoriye girdiğinin tespiti
yapılır ve ilgili dosyaya yazılıp ilgili klasöre kopyalanır
switch (hangiifade)
     case 1
          disp('AGRI YOK ')
          yaz = fopen('agriyok.txt','a');
          fprintf(yaz,fotopathh);
          fprintf(yaz,',\n');
          fclose(yaz);

copyfile(fotopathh,'C:\Users\fatıh\Documents\MATLAB\test_aci0_
');

     case 2
          disp('AZ AGRI')
          yaz = fopen('agri1.txt','a');
          fprintf(yaz,fotopathh);
          fprintf(yaz,',\n');
          fclose(yaz);

copyfile(fotopathh,'C:\Users\fatıh\Documents\MATLAB\test_aci1_
');
     case 3
          disp('COK AGRI! ')
          yaz = fopen('agri2.txt','a');
          fprintf(yaz,fotopathh);
          fprintf(yaz,',\n');
          fclose(yaz);

copyfile(fotopathh,'C:\Users\fatıh\Documents\MATLAB\test_aci2_
');
     otherwise
          disp('OGRENILMEMIS IFADE\n')
end
fprintf(' CikişDegeri_ 1:%f,\n CikişDegeri_2:%f,\n
CikişDegeri_3:%f\n', Co(1),Co(2),Co(3));
end %for sonu
fprintf('Otomatik Test işlemi bitti _________\n');
```

8.koselerikaldir.m fonksiyonu

```
function temizdizi = func_er_koselerikaldir(dizi)
% Bu filtre, fotoğrafın etrafındaki gereksiz kısımları siler

    [x,y] = size (dizi); % [satirxsutun]
```

```
    %fotoğrafın 4 köşesindeki 20x20 kısmı siler
    dizi(1:20,1:20)=           1 ; %sol ust
    dizi(1:20,y-20+1:y)=       1 ; %sag ust
    dizi(x-20+1:x,1:20)=       1 ; %sol alt
    dizi(x-20+1:x,y-20+1:y)=   1 ;%sag alt

    %fotoğrafı 10 piksel çerçeve şeklinde çevresini temizler
    dizi(1:5,1:y)=             1 ; %ustten 5 piksel temizle
    dizi(x-10:x,1:y)=          1 ; %alttan 10 piksel temizle
    dizi(:,1:5)=               1 ; %sol
    dizi(:,y-5:y)=             1 ; %sag


    temizdizi = dizi;
end
```

KAYNAKÇA

Ayaz, G., *Karın Ağrısı Nedeniyle Acil Servise Başvuran Hastaların Ağrı Şiddetlrinin Ölçümü ve Acil Hekimlerinin Ağrı Yönetimindeki Tutumlarının Değerlendirilmesi*, Yayımlanmamış Tus Tezi, 2019.

Çayıroğlu, İ., Karabük Üniversitesi İleri Algoritma Analizi Ders Notları, 2019.

Çöçelli ve ark, "Ağrı Tedavisinde Hemşirenin Rolü", **Gaziantep Tıp Dergisi,** 2008 14:53-58.

Diri, Banu, Makine Öğrenmesine Giriş Ders Notları, Yıldız Teknik Ünv Bilgisayar Mühendisliği Bölümü, 2014.

Elmas, Çetin, **Yapay Zeka Uygulamaları**, Seçkin Yayıncılık 4.Baskı Ankara 2018 s:25-104.

Eti-Aslan, Fatma, "Ağrı Değerlendirme Yöntemleri", **C.Ü. Hemşirelik Yüksekokulu Dergisi**, 6(1), 2002.

Gardner, H., **Multiple Intelligences: The Theory in Practice**, Basic Books, 1993. p:63-292.

Güvenoğlu Erdal, Bil 632 Kodlu Bilgisayarlı Görme Ders Notları, Maltepe Üniversitesi, 2017.

Haykin, S., **Neural Network and Learning Machines**, Pearson Education Ltd., 1999 p:34-44.

Hebb, D.O., The Organization Of Behavior, 1949 s:17-25.

Martella, C., Jimbo Project: Handwriting recognition using an Artificial Neural Network, 2005 P:5.

Mitchell, Tom M.,**Machine Learning**, McGraw-Hill Science,1997.

Monwar, M., *Machine Learning Techniques in Pain Recognition*, The University of Northern British Columbia December, 2006.

Nabiyev, V.V., **Yapay Zeka:İnsan - Bilgisayar Etkileşimi**, Seçkin Yayınevi, 5.Baskı, Ankara 2016 s:27.

Orhan, Umut, Makine Öğrenmesine Giriş Ders Notları, Çukurova Üniversitesi, 2018.

Öztemel, E., **Yapay Sinir Ağları**, Papatya Yayıncılık, Ağustos 2003, 1.Basım s:2-47.

Roberts, Lawrence G. Machine perception of three-dimensional solids. Diss. Massachusetts Institute of Technology, 1963.

Stuart, J. Russell and Norvig P., **Artificial Intelligence A Modern Approach**, Prentice-Hall Inc. 1995 p:5.

Szeliski, R., Computer vision: algorithms and applications. Springer Science & Business Media s:5, 2010.

Yılmaz, A., **Yapay Zeka**, İnkılap Yayınevi, İstanbul 2018, 4.Baskı, s:5 s:13.

Zhu, S., "Pain Expression Recognition Based on PLSA Model", **ScientificWorldJournal** 2014; 2014: 736106., Published online 2014 Mar 27.

İnternet Kaynakları

URL-1, **https://web.itu.edu.tr/~sonmez/lisans/ai/yapay_zeka_icerik1_1.6.pdf**, Erişim Tarihi:12.07.2019 14:10.

URL-2, **https://www.theguardian.com/technology/2014/jun/09/scientists-disagree-over-whether-turing-test-has-been-passed**, Erişim Tarihi 12.07.2019 18:40.

URL-3, **https://www.youtube.com/watch?v=qtApzKnGU94**, Erişim Tarihi: 12.07.2019 19:40.

URL-4, **https://www.youtube.com/watch?v=GhGiyuEfQVY**, Erişim tarihi: 09.08.2019 18:55.

URL-5, **https://www.youtube.com/watch?v=D5VN56jQMWM**, Erişim Tarihi: 12.07.2019 19:30.

URL-6, **http://catalog.tedu.edu.tr**, Erişim Tarihi 01.08.2019.

URL-7, **https://web.itu.edu.tr/kcankocak/docs/ek_okumalar/Carter_Beyin_Kitabi-68-69.pdf**, Erişim Tarihi 27.08.2019.

URL-8, **https://journals.lww.com/ajnonline/Fulltext/2001/02000/Speaking_the_Language_of_Pain.47.aspx**, Erişim Tarihi: 10.09.2019.

URL-9, **http://pics.stir.ac.uk/2D_face_sets.htm**, Erişim Tarihi : 07.12.2019.