İSTANBUL
ÜNİVERSİTESİ
FEN BİLİMLERİ
ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

BİLGİSAYAR DESTEKLİ EL YAZISI KARAKTERLERİNİ TANIMA SİSTEMİ TASARIMI

Bilgisayar Müh. Pelin ARAS
Bilgisayar Mühendisliği Anabilim Dalı

Danışman
Yrd. Doç. Dr. Oğuzhan ÖZTAŞ

Haziran, 2006

İSTANBUL

Bu çalışma 17/07/2006 tarihinde aşağıdaki jüri tarafından Bilgisayar Mühendisliği Anabilim Dalı Bilgisayar Mühendisliği programında Yüksek Lisans Tezi olarak kabul edilmiştir.

Tez Jürisi

Yrd. Doç. Dr. Oğuzhan ÖZTAŞ (Danışman)
İstanbul Üniversitesi
Mühendislik Fakültesi
Fakültesi

Doç.Dr. Sabri ARIK
İstanbul Üniversitesi
Mühendislik

Prof. Dr. Ahmet SERTBAŞ
İstanbul Üniversitesi
Mühendislik Fakültesi

Prof. Dr. Mahmut ÜN
İstanbul Üniversitesi
Mühendislik Fakültesi

Doç. Dr. Hakan Ali ÇIRPAN
İstanbul Üniversitesi
Mühendislik Fakültesi

ÖNSÖZ

Bu çalışma, İstanbul Üniversitesi Fen bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı Yüksek Lisans Tezi olarak hazırlanan "Bilgisayar Destekli El Yazısı Karakterlerini Tanıma Sistemi Tasarımı" isimli tezi içermektedir.

El yazısı karakterleri tanıma, günümüzde yaygınlaşarak gelişen çalışma konularındandır. Güvenlik, bankacılık, gazetecelik, dokümantasyon gibi pek çok alanda karakter tanıyan sistemlere ihtiyaç duyulmaktadır.

Karakter tanımada kullanılan birçok yöntem olup, bu çalışmada yapay sinir ağları tekniği kullanılmıştır. YSA'lar gerçek dünyaya ait ilişkileri tanıyabilir, sınıflandırma, kestirim ve işlev uydurma gibi görevleri yerine getirebilirler. Örüntü tanıma (Pattern recognition) tekniğinin gerekliliği, gerçek dünya ile bilgisayar ilişkisinin başlaması ile ortaya çıkmıştır. Bu durum YSA'nın çok güçlü örnek tanıma tekniği olarak ortaya çıkmasına ve gelişmesine neden olmuştur.

Yapay sinir ağlarında kullanılmak üzere birçok öğrenme algoritması geliştirilmiştir. En yaygın olarak kullanılanı geriye yayılım (backpropagation) öğrenme algoritmasıdır. Bu tezde, bilinen öğrenme algoritmalarına alternatif olarak Shashank tarafından geliştirilen öğrenme algoritması ve geriye yayılım öğrenme algoritması incelenmiştir. Geriye yayılım ağının eğitim kalitesi artırılmış ve el yazısı karakterleri tanıma performansı açısından Shashank' ın algoritmasından üstün hale getirilmiştir.

Yüksek Lisans ders ve tez aşamasındaki çalışmalarım süresince her türlü destek ve yardımlarını esirgemeyen değerli hocalarım Doç. Dr. Sabri Arık ve Yrd. Doç. Dr. Oğuzhan ÖZTAŞ teşekkürü bir borç bilirim.

Hayatımın her döneminde bana kararlı, azimli ve çalışkan olmayı aşılayan, daima ve her koşulda yanımda olan aileme de teşekkür ederim.

Haziran, 2006 **Pelin ARAS**

İÇİNDEKİLER

ŞEKİL LİSTESİ

TABLO LİSTESİ

SEMBOL LİSTESİ

δ : Nöron Hatası
w : Bağlantı Ağırlığı
β : Öğrenme Katsayısı
Y : Nöron Çıkışı
Ψ : Adaylık Skoru (Candidate Score)
μ : İdeal Ağırlık Modeli Skoru (Ideal Weight Model Score)
Q : Tanıma Bölüm Değeri (Recognition Quotient)

KISALTMALAR

OCR : Optical Character Recognition (Optik Karakter Tanıma)
YSA : Yapay Sinir Ağları (Artificial Neural Networks)
ÇKA : Çok Katmanlı Algılayıcılar (Multilayer Perceptrons)
AI : Artificial Intelligence (Yapay Zeka)

ÖZET

BİLGİSAYAR DESTEKLİ EL YAZISI KARAKTERLERİNİ TANIMA SİSTEMİ TASARIMI

Örüntü tanıma (Pattern recognition) bilim disiplininin amacı nesneleri bir kategoriye koymak veya sınıflamaktır. Bu nesneler, uygulamaya göre görüntü, ses ya da sınıflandırılması istenen başka bir işaret olabilir ve genel olarak örüntü (pattern) diye adlandırılır.

Bu çalışma örüntü tanımanın dallarından biri olan karakter (harf veya sayı) tanıma uygulamasıdır. Karakterlerin tanınmasında birkaç metodoloji kullanılır. Bunlardan biri yapay sinir ağlarına dayanan tanıma işlemidir. YSA (Yapay Sinir Ağları), biyolojik sinir ağlarını taklit eden bilgisayar programlarıdır [1]. Bir çok YSA tipi bulunmakla birlikte, en çok kullanılan sinir ağı yapısı, İleri Beslemeli Geri Yayılımlı YSA olarak bilinendir. Belirsiz, gürültülü ve eksik bilgilerin işlenmesinde yapay sinir ağları başarıyla kullanılmaktadır. Klasik bilgisayarlar bilgiyi belleğinde belirli bir yerde saklar, sinir ağları ise bilgiyi tüm ağ boyunca dağıtır. Bu durum dağıtılmış bellek olarak bilinir [2].

Yapay sinir ağları, algoritmaları çıkarılamayan problemler için, çözüm sağlayan yeni ve güvenli bir bilgi işleme sistemi olabilmektedir. Yapılan çalışma bu durum için uygun bir örnek niteliği taşımaktadır.

Bu çalışmada öncelikle geriye yayılım (backpropagation) öğrenme algoritması kullanılarak el yazısı karakterlerini tanıma sistemi geliştirilmiştir. Oluşturulan geriye yayılım yapay sinir ağının parametre performansları grafiksel olarak incelenmiştir. Ve ağ performansını yükseltecek parametre değerleri bulunmuştur. İkinci olarak ise Shashank'ın öğrenme algoritması kullanılarak karakter tanıma sistemi tasarlanmıştır. Test aşamasında Shashank' ın geliştirdiği adaylık skoru (candidate score), ideal ağırlık modeli skoru (ideal weight model score) ve tanıma bölüm (recognition quotient) değerleri kullanılmıştır.

Sonuç olarak ise bu iki algoritmanın performans ve eğitim süreleri karşılaştırılmıştır. İyi eğitilmiş geriye yayılım ağının Shashank ağına göre üstünlüğü gösterilmiştir. Geriye yayılım ağının eğitim süresi ise Shashank ağının eğitim süresine oranla çok daha fazladır.

SUMMARY

COMPUTER AIDED HANDWRITING CHARACTER RECOGNITION SYSTEM DESIGN

The target of pattern recognition science is to categorize or classify of objects. These objects can be image, voice, speech or another sign according to the application. And generally they are named as pattern.

This thesis is an application of character recognition that is one of the branches of pattern recognition. In character recognition a lot of methods are used. One of them is artificial neural networks. ANN is a computer program that imitates biological neural network [1]. There are a lot ANN types but the most used is feed forward backpropagation neural networks. Artificial neural networks perform with ambigious, noisy and defected data well. Traditional computer systems keep data in a particular place in memory. But neural networks distribute data all over the network. This is called distributed memory [2].

ANN is a new and safety data operating system that produces solutions for problems difficult to obtain algorithm. This study is a sample of that.

In this work, a handwritten characters recognition system has been developed using backpropagation learning algorithm first. Parameter performance of backpropagation neural network was analyzed with graphics. And the parameter values that increase system performance were determined. Secondary, the system has been developed again using Shashank's learning algorithm. In test step, Shashank's candidate score, ideal weight model score and recognition quotient values were used.

As a result, performance and training time of two algorithms are compared. And it is shown that a well trained neural network with backpropagation learning algorithm is more superior than Shashank network. On the other hand the training of network using backpropagation algorithm takes much more time than Shashank algorithm.

1. GİRİŞ

İnsanlar varoluşlarından itibaren, birbirleriyle anlaşabilmek için çeşitli yollar denemişlerdir. Önce konuşmayı öğrenmişler, diller ortaya çıkmıştır. Bir müddet sonra, yalnızca karşısındaki insanlarla konuşmak yeterli olmamış, mesaj bırakma ihtiyacı duymuşlardır. Bunun için mağara duvarlarına resim yapmışlardır. Resimler zamanla sabitleşerek, özel işaretlerden oluşan bir alfabeye dönüşmüş, dolayısıyla yazının temelleri atılmıştır. Her yazı ifadesinin, dilin alfabesinde farklı sayıda karakterleri, biçimleri ve kuralları vardır. Dilin sesli göstergelerini karşılamayı amaçlayan, görüntüsel öğelerden veya harflerden oluşan göstergeler dizisi olan yazı, tartışmasız insanlığın en önemli keşfidir.

1.1. DOKÜMAN İŞLEME

Tez çalışmasının konusu görüntü tanıma alanına girmektedir. Görüntü tanımanın yapay zeka uygulamalarında geniş yeri vardır. Bunlara örnek olarak bankacılıkta ve kriminolojide kullanılan el yazısı tanıma sistemleri, optik karakter tanıma, (zarflarda posta kodunun tanınması), biyoloji, tıp ve uzaybilim sistemlerindeki görüntülerin yorumlanması gösterilebilir.

Bu çalışmada tanımanın önemli dallarından biri olan, yazılı dokümanların içerdiği metinlerin tanınması için karakter tanıma üzerinde durulmaktadır. Üzeri satırlarca yazı dolu bir kağıt, okuma bilmeyen bir kişi için bir şey ifade etmediği gibi, bilgisayarlar için de değersiz bir resimden ibarettir. Bilgisayarın, bu resmin içindeki bilgiyi kullanabilmesi için yorumlaması gerekmektedir. Başka bir deyişle, bu resimdeki yazıların bilgisayarın anlayacağı, daha kolay ve verimli saklayıp, bulabileceği sembollere çevrilmesi gerekmektedir. Bunun için de kağıt üzerindeki bir metnin doküman işleme denilen bir dizi işlemden geçmesi gerekir. Bu işlemi gerçekleştiren sistemlere optik karakter tanıma (Optical Character Recognition-OCR) sistemleri denilir ve birçok ticari OCR sistemi bulunmaktadır.

1.2. OCR SİSTEMLERİ

Modern OCR teknolojisi 1951 yılında M. Sheppard'ın icadı olan GISMO Okur-Yazar Robot ile birlikte doğmuştur [1]. 1960'ların sonlarında teknoloji çok gelişmesine rağmen, OCR sistemleri yalnız devlet kuruluşları ve çok büyük şirketler tarafından kullanılan, egzotik ve geleceğe ait gibi düşünülen sistemlerdi. Bu sistemlerin maddi değerleri milyon dolarlar düzeyindeydi.

Günümüzde artık OCR sistemleri daha ucuz, daha hızlı ve daha güvenilir olmuşlardır. Bugünkü OCR sistemleri ile daha fazla yazı tipi tanınabilmektedir. OCR sistemleri üzerine yapılan oldukça fazla sayıda araştırmalar ve daha ucuz elektronik bileşenler, bu sistemlerin yolunu açmışlardır ve bunlara olan talep hala devam etmektedir. Bu sebepledir ki, bu sistemleri daha da hızlandıracak ve hata oranlarını düşürecek yeni yöntemlere ihtiyaç vardır.

OCR sistemleri, kabaca iki kategoride toplanmaktadır: işe özel sistemler ve genel amaçlı sistemler. İşe özel sistemler özel doküman tiplennde kullanılmaktadır. Sıkça rastlanan, işe özel okuma sistemleri banka çekleri, posta adresleri ve kredi kartı dökümlerini okumakta kullanılmaktadır. Bu sistemlerde özel donanımların kullanımı ile dokümanın belirli kısımları taranarak doküman sayısallaştırılmaktadır. Bu tip sistemler, düşük hata oranları ile hızlı bir şekilde sonuç üretmek zorundadırlar.

Birçok ülkede kullanılan pasaport kontrol sistemleri de, otomatik bir pasaport okuyucu yardımı ile çalışmaktadır. Okuyucu, yolcunun adını, doğum tarihini ve pasaport numarasını okuyarak, kaçak suçluların veri tabanı ilc karşılaştırmaktadır. Böylece bir insanın okuma ve yazma süresinde kaybedeceği zaman kazanılmıştır.

Bunların dışında banka çeki okuyucuları, hesap işleme sistemleri ve havayolu bilet okuyucuları gibi işe özel OCR sistemleri kullanımdadır. Bu sistemler sayesinde, insanlar için çok uzun zaman alacak işler otomatikleştirilmektedir. Ama yine de hata oranlarına karşın insan gözetiminde çalışmaktadırlar.

Genel amaclı okuyucular, iş mektupları, teknik raporlar ve gazeteler gibi geniş bir yelpazedeki dokümanları işlemektedirler. Bu sistemler, doküman sayfasının görüntüsünü alarak metin bloklarına ve metin dışı bloklara ayırırlar. Grafikler ve çizimler gibi metin dışı bloklar, yazılardan ayrı olarak işlenip, saklanmaktadırlar. Metin blokları satırlara, kelimelere ve karakterlere parçalanıp karakter tanıma kısmına geçirilirler. Bu tez çalışmasında karakter tanıma işlemi üzerine çalışılmıştır. Bu da OCR teknolojisinin temelidir.

1.3. KARAKTER TANIMA

Karakter tanıma örüntü tanıma disiplininin bir uygulamasıdır. On yıllardır bilgisayarların da, insanlar gıbi örüntüleri ayırt edebilmeleri için çok miktarda çalışma yapılmıştır. Üzerinde çalışılan örüntülerden bazıları, karakterler, semboller, resimler, üç boyutlu fiziksel objeler, ses dalgaları, elektro kardiyogramlar, sismik dalgalardır.

Örüntü tanımanın en basit yolu şablon işlemedir. Bu durumda her bir örüntü sınıfı için bir şablon olmak üzere, şablonlar kümesi bellekte saklanır. Bilinmeyen sınıf her sınıfın şablonu ile karşılaştırılır. Sınıflama, daha önceden belirlenmiş bir eşleme kriterine veya benzerlik kriterine göre yapılır.

Karakter tanıma bir metnin içindeki karakterlerin taranmış sayısal görüntüsünü bunlara karşılık gelen sembolik ifadesine dönüştürme işlemidir. Karakter tanıma basılı (printed) ve elyazısı (handwriting) olmak üzere iki ayrı kategoride araştırılmaktadır. Kullanılan yazı tipi ve stili bilinen basılı karakterler için tanıma yüzdesi %99' lar düzeyindedir. Ama bu çalışmada olduğu gibi alışılmadık yazı tipleri veya düşük kalitede baskılar ile karşılaşıldığında başarı düşmektedir.

1.4. KARAKTER TANIMA ÖN İŞLEMLERİ

Karakter tanıma işlemi genel olarak ön işlemleri gerektirmektedir. Ön işlemlerden bazıları eşikleme, inceltme, onarma ve normalizasyondur.

1.4.1. Eşikleme

Eşikleme giriş karakterinin ikili görüntüsünü elde etmek için kullanılır. Bilgisayar ekranında karakterleri oluşturan her piksel için mevcut bir parlaklık derecesi vardır. Bu piksellerin parlaklık derecelerinden oluşan bir örüntü, matris şeklinde tutulur. Çoğu karakter tanıma sistemleri gri seviyelerini (grayscale) siyah ve beyaz olmak üzere iki seviyeye indirgerler. Bir eşik değerine göre bu indirgemeye ikili hale getirme (binarization) adı verilir.

1.4.2. İnceltme

İnceltme işlemi tanıma işlemini daha kolay ve hızlı bir hale getirir. Ayrıca oluşabilecek hataları en aza indirir. Çerçevelenmiş karakterdeki bilgi taşıyan siyah piksel sayısını en aza indirdiği için, işleme sokulacak nokta sayısı azalmış olur. Bu da program hızını artırır.

1.4.3. Onarım

Onarım işleminde ise kopuk çizgilerin birleştirilmesi, eğri ve doğruların biçimlerinin düzenlenmesi işlemleri yapılmaktadır. Parçaları elde ederken, iskelete ait anahtar noktalar (köşe noktaları ve kesişen noktalar gibi) tanımlanır. İki anahtar nokta arasındaki parça böylece elde edilir.

1.4.4. Normalizasyon

Normalizasyon genellikle karakter geometrisindeki değişim olarak tanımlanabilir. Değişimler, pozisyon, boyut ve konumda olabilir. Örneğin 12x7 ve 14x8 boyutlarında bir karakterin 25x25 bir alana taşınması bir normalizasyon (ölçekleme) işlemidir.

1.5. KARAKTER TANIMA TEKNİKLERİ

Karakter tanıma uygulamaları her alanda kullanılıp yaygınlaştığı için yıllar boyunca üzerinde çalışıldığından dolayı farklı teknikler geliştirilmiştir.

1.5.1. Ağırlık Merkezine Dayalı Tanıma

Ağırlık merkezine dayalı tanıma, karakterlerin ağırlık merkezlerinin farklı noktalarda olduğu düşüncesinden ortaya çıkmıştır. Şekil 1.1 de bazı karakterlerin ağırlık merkezlerine (G(x, y)) işaret edilmiştir. Bu şekil piksel bazında incelendiğinde ağırlık

merkezlerinin aynı noktada olduğu söylenebilir. Oysa ki oransal temelde bakıldığında farklı olacaktır [1].

Şekil 1.1 : Bazı karakterlerin ağırlık merkezleri [1]

Ağırlık merkezine dayalı sınıflandırma sabit karakter boyutlarında (bazı çakışmalar göz ardı edilirse) iyi sonuçlar vermektedir. Ağırlık merkezi kriteri her ne kadar ayırt edici özellik taşıyorsa da, karakterlerin büyüklükleri değiştiğinde ağırlık merkezleri de kaymaktadır. Bu kaymanın sebebi, büyütme küçültme işlemlerinin piksel bazında yapılmasıdır. Aynı yazı tipinin farklı büyüklüklerinde ağırlık merkezi kaymaktadır. Bu nedenle farklı özellik vektörlerine ihtiyaç vardır.

1.5.2. Korelasyon

Karakter tanıma alanında ilk yaklaşımlardan biri basit optik şablon eşlemedir. Korelasyon, karakter tanıma literatüründe sıkça kullanılan bir terimdir. Korelasyon en basit anlamda iki ikili örüntü (pattern) arasındaki ıızaklığın belirlenmesi olarak açıklanabilir. Bu uzaklık ikili değişimler sayısını ifade etmektedir [3].

1.5.3 Kontür Çıkarma

Bu algoritma çerçevelenmiş karakterin 4 yönlü kesit (sol, üst, sağ, alt) eğrilerinin yapılarının incelenmesine dayanmaktadır. Karakter tanıma işlemleri bu 4 yön izdüşümü bilgilerinin karşılaştırılması ile gerçekleştirilmektedir.

Aramayı hızlandırmak amacı ile her yönde elde edilen uzaklık bilgileri önce toplanarak toplam izdüşüm değerleri elde edilir. Aynı değerli birkaç karaktere rastlandığında ise oluşturulmuş yön bilgilerini içeren 4 vektörün farkları değerlendirilir. Her yönde

normalize edilmiş karakterlerin izdüşüm farklarının toplamı alınarak minimum olanı muhtemel sonuç olarak verilir.

1.5.4. Yapay Sinir Ağları İle Öğrenme

Yapay Sinir Ağları (YSA), giriş verilerinin yetersiz olduğu, mevcut verilerden hareketle bilinmeyen ilişkilerin ortaya çıkarılması ve algoritması veya kuralları tam olarak bilinmeyen durumlar için geliştirilmiş bir bilgi işleme sistemidir. Bu tez çalışmasında, yapay sinir ağlarının önemli bir uygulaması olan karakter tanıma işlemi ele alınmıştır. elyazısı karakterler, giriş vektörü olarak ağa sunulmuş ve YSA'nın eğitim işlemi bu karakterler üzerinden gerçekleştirilmiştir.

Bu çalışmada çok katmanlı ileri beslemeli bir YSA mimarisi ve geriye yayılım (backpropagation) öğrenme metodu kullanılmıştır. İkinci bölümde genel anlamda yapay zeka ve öğrenme üzerinde durulmuş, yapay sinir ağları ve çok katmanlı algılayıcıların (multilayer perceptrons) yapısı ve çalışma teknikleri anlatılmıştır.

Üçüncü bölümde, yapay sinir ağlarında kullanılan geriye yayılım algoritmasında her adımda gerçekleştirilen hesaplamalar, görüntüyü ağa sunmak için yapılan ön işlemler ve dijitalleştirme anlatılmıştır. Ayrıca çok katmanlı geriye yayılım ağının çalışması, eğitimi, eğitimin durdurulması ve test edilmesi üzerinde durulmuştur. Bunun yanı sıra karşılaştırılmak üzere Shashank' ın algoritması incelenmiş, aynı şekilde Shashank ağı eğitimi ve test edilmesi anlatılmıştır.

Dördüncü bölümde, geriye yayılım algoritması ile ilgili elde edilen bulgulardan söz edilmiştir. Geriye yayılım algoritmasının performansını etkileyen parametreler tek tek incelenmiş, performansı yükselten veya düşüren değerler grafiksel olarak elde edilmiştir.

Beşinci bölümde geriye yayılım ağının genel olarak el yazısı karakterlerini tanıma performansı, ardından Shashank ağının performansı incelenmiştir. Son olarak iki algoritmanın performans ve kaynak kullanımları grafiksel olarak karşılaştırılmıştır. Ve ileriye yönelik geliştirilebilecek çalışmalar düşünülmüştür.

2. GENEL KISIMLAR

2.1. YAPAY ZEKA VE ÖĞRENME

2.1.1. Yapay Zeka Teknolojisi

Yapay Zeka (Artificial Intelligence, Kısa: AI) terimi Stanford Üniversitesinde Profesör olan John McCarthy tarafından ortaya atılmıştır. Yapay zekanın tanımını yapmak önemli olduğu kadar da zordur. Çünkü yapay zeka çok kapsamlı bir konu olup pek çok alt alan içermektedir ve alt alanı olarak düşünülemeycek pek çok alanla da bir biçimde ilintilidir. Yapay zekanın içerdiği ve ilintili olduğu tüm alanları kapsayacak şekilde yapılmaya çalışılacak tek bir tanım ya çok uzun olacaktır, ya çok soyut olacaktır ya da eksik olacaktır. Dolayısıyla yapay zekanın tanımını yapmaya çalışan kişiler genellikle kendi uzmanlık alanlarını ön plana çıkaran tanımlar yapmışlardır.

Bu nedenle tek bir tanım yapmak yerine, birden çok tanıma yer vererek zihnimizde yapay zeka kavramının daha eksiksiz oluşmasını sağlamak en uygunu olacaktır:

- Yapay zeka yapay bir varlığın (genellikle bir bilgisayar) sergilediği zekadır.
- Yapay zeka bir makine ya da insan eliyle üretilmiş otonom bir sistem kullanarak insan zekasının benzetimini yapmaya çalışan bir araştırma alanıdır.
- Yapay zeka, insanın düşünme yöntemlerini analiz ederek bunların benzeri yapay yönergeleri geliştirmeye çalışan araştırma alanıdır.
- Yapay zeka, canlılarda (özellikle insanlarda) bulunan algılama, öğrenme, çoğul kavramları bağlama, düşünme, fikir yürütme, sorun çözme, iletişim kurma, çıkarım yapma ve karar verme gibi yüksek bilişsel fonksiyonları ve otonom davranışları sergilemesi beklenen yapay bir sistemdir (donanım+yazılım).

Matematiksel olarak formulasyonu kurulamayan ve çözülmesi çok zor ya da mümkün olmayan problemler sezgisel yöntemlerle bilgisayarlar ile çözülebilmektedir.

Bilgisayarların bu özelliklerle çalışmasını sağlayan ve bu yöndeki yeteneklerinin gelişmesini sağlayan çalışmalar "yapay zeka" çalışmaları olarak bilinmektedir.
Yapay zeka sistemlerinin en önemli özellikleri olaylara çözümler üretirken bilgiye dayalı karar mekanizmalarını çalıştırabilmeleri ve eldeki bilgilerle olayları öğrenerek, sonraki olaylar hakkında karar verebilmeleridir.

2.1.2. Yapay Zeka'nın Alt Alanları

Yapay zeka çalışmaları değişik teknolojilerin doğmasına neden olmuştur. Çünkü günlük olaylar ve problemler sürekli değişmektedir. Ve bir olay farklı insanlar tarafından farklı şekilde algılanabilmektedir. Bilgisayarların insanların karar verme mekanizmalarını taklit etmelerinin sağlanması da farklı teknolojilerin ortaya çıkmasını doğurmuştur.

Yapay Zeka bilimi içerisinde yer alan alt alanlardan bazıları aşağıda listelenmiştir:

- Bulanık Mantık (Fuzzy Logic)
- Dağıtık Zeka (Distributed Intelligence)
- Doğal Dil İşleme (Natural Language Processing)
- Genetik Algoritmalar (Genetic Algorithms)
- Görüntü İşleme (Image Processing)
- Konuşma İşleme (Speech Processing)
- Makine Öğrenmesi (Machine Learning)
- Örüntü Tanıma (Pattern Recognition)
- Robotik (Robotics)
- Uzman sistemler (Expert Systems)
- Veri Madenciliği (Data Mining)
- Yapay Sinir Ağları (Artificial Neural Networks)

2.1.3. Makine Öğrenmesi

Öğrenme kavramı değişik şekillerde tanımlanmıştır. Simon öğrenmeyi, "zaman içinde yeni bilgilerin keşfedilmesi yoluyla davranışların iyileştirilmesi süreci" olarak tanımlamaktadır [3]. Makine öğrenmesi ise öğrenme işinin bilgisayarlar yardımıyla gerçekleştirilmesidir. Bilgisayar da insan gibi zaman içinde tecrübe kazanarak öğrenir. Bilgisayarlar bir olayla ilgili bilgi ve tecrübe edindikten sonra karar verip gelecekte oluşacak benzer olaylar hakkında çözümler üretebilirler.

2.1.4. Öğrenme Çeşitleri

Yapay sinir ağları gibi öğrenme yöntemleri, örneklerden öğrenmeye dayalıdır. Örneklerden öğrenme, bir olay ile ilgili gerçekleşmiş örnekleri kullanarak olayların girdi ve çıktıları arasındaki ilişkileri öğrenip bunlara göre yeni örneklerin çıktılarını belirlemektir. Örneklerden öğrenen sistemlerde değişik öğrenme çeşitleri geliştirilmiştir [4].

2.1.4.1. Öğretmenli- Eğiticili (Supervised) Öğrenme

Bu sistemde öğrenen sistemin olayı öğrenmesine bir öğretmen yardımcı olmaktadır. Öğretmen sisteme öğrenilmesi istenen olay ile ilgili örnekleri giriş/çıkış seti olarak verir. Yani, her örnek için hem girdiler hem de o girdiler karşılığında oluşturulması gereken çıktılar sisteme gösterilirler. Sistemin görevi girişleri öğretmenin belirlediği çıkışlara göre düzenlemektir. Bu sayede ağ, olayın girişleri ile çıkışları arasındaki ilişkileri öğrenmektedir.

2.1.4.2. Destekleyici (Reinforcement) Öğrenme

Bu tür öğrenmede öğrenen sisteme bir öğretmen yardımcı olur. Fakat öğretmen her giriş seti için olması gereken (üretilmesi gereken) çıkış setini sisteme göstermek yerine sistemin kendisine gösterilen girişlerine karşılık çıkışlarını üretmesini bekler ve üretilen çıkışın doğru veya yanlış olduğunu gösteren bir sinyal üretir. Sistem, öğretmenden gelen bu sinyali dikkate alarak öğrenme sürecini devam ettirir. LVQ ağı bu tip öğrenmeyi kullanan sistemlere örnek olarak verilebilir.

2.1.4.3. Öğretmensiz (Unsupervised) Öğrenme

Bu tür öğrenmede sistemin öğrenmesine yardımcı olan herhangi bir öğretmen yoktur. Sisteme sadece giriş değerleri gösterilir. Örneklerdeki parametreler arasındaki ilişkileri sistemin kendi kendisine öğrenmesi beklenir. Bu, daha çok sınıflandırma problemleri

için kullanılan bir stratejidir. Yalnız sistemin öğrenmesi bittikten sonra çıkışların ne anlama geldiğini gösteren etiketlendirmenin kullanıcı tarafından yapılması gerekmektedir. ART ağları bu stratejiyi kullanan sistemlere örnek olarak verilebilir.

2.1.5. Öğrenme Kuralları

Yapay sinir ağları gibi öğrenen sistemlerde öğrenme, yukarıda anlatılan türlerden hangisi uygulanırsa uygulansın bazı kurallara göre gerçekleştirilmektedir. Bu kuralların bazıları çevrimiçi (on-line) bazıları ise çevrimdışı (of-line) çalışmaktadır.

2.1.5.1. Çevrimiçi (On-Line) Öğrenme Kuralları

Bu kurallar gerçek zamanlı çalışabilmektedir. Bu kurallara göre öğrenen sistemler gerçek zamanda çalışırken bir taraftan fonksiyonlarını yerine getirmekte diğer taraftan ise öğrenmeye devam etmektedir. ART ağının öğrenme kuralı ve *Kohonen* öğrenme kuralı bu sınıfta bulunan öğrenme kurallarına örnek olarak verilebilir.

2.1.5.2. Çevrimdışı (Off-Line) Öğrenme Kuralları

Çevrimdışı öğrenme kurallarına dayalı öğrenen sistemler kullanıma alınmadan önce örnekler üzerinde eğitilirler. Bu kuralları kullanan sistemler eğitildikten sonra gerçek hayatta kullanıma alındığında artık öğrenme olmamaktadır. Sistemin öğrenmesi gereken yeni bilgiler söz konusu olduğunda sistem kullanımdan çıkarılmakta ve çevrimdışı olarak yeniden eğitilmektedir. Eğitim tamamlanınca sistem tekrar kullanıma alınmaktadır. Yapay sinir ağlarında yaygın olarak kullanılan Delta Öğrenme Kuralı bu tür öğrenmeye örnek olarak verilebilir.

2.1.6. Öğrenme Kurallarından Bazıları

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. Bunların yanında, farklı öğrenme kuralı fonksiyonları da vardır ve bu konuda çalışmalar sürmektedir. 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 öğrenme, kullanılmakta olan öğrenme kurallarının ima ettiğinden çok daha komplikedir. Kullanılmakta olan bazı önemli öğrenme kuralları arasında Hebb Kuralı, Hopfield Kuralı, Delta Kuralı, Eğimli Değişim (Gradient Descent) Kuralı ve Kohonen Öğrenme

Kuralı sayılabilir. Yapay sinir ağlarında öğrenme kurallannın çoğu Hebb kuralına dayanmaktadır.

- Hebb Kuralı : Bilinen en eski öğrenme kuralıdır. Diğer öğrenme kurallarının temelini oluşturmaktadır. 1949 yılında geliştirilen bu kurala göre, bir hücre (yapay sinir ağı elemanı) diğer bir hücreden bilgi alırsa ve her iki hücrede aktif ise (matematik olarak aynı işareti taşıyorsa) her iki hücrenin arasındaki bağlantı kuvvetlendirilmelidir. Diğer bir deyişle bu kural şu şekilde özetlenebilir. Bir hücre kendisi aktif ise bağlı olduğu hücreyi aktif yapmaya pasif ise pasif yapmaya çalışmaktadır. Diğer öğrenme kurallarının çoğu bu felsefeyi baz alarak geliştirilmiştir.
- Hopfield Kuralı : Bu kural Hebb kuralına benzemektedir. Yapay sinir ağı elemanlarının bağlantılarının ne kadar kuvvetlendirilmesi veya zayıflatılması gerektiği belirlenir. Eğer beklenen çıkış ve girişler ikisi de aktif /pasif ise öğrenme katsayısı kadar ağırlık değerlerini kuvvetlendir/zayıflat denmektedir. Yani ağırlıkların kuvvetlendirilmesi veya zayıflatılması öğrenme katsayısı yardımıyla gerçekleştirilmektedir. Öğrenme katsayısı genel olarak 0-1 arasında kullanıcı tarafından atanan sabit ve pozitif bir değerdir.
- Kohonen Kuralı : Kohonen (1982) tarafından geliştirilen bu kural biyolojik sistemlerdeki öğrenme sisteminden esinlenilmiştir. Bu yöntemde, işlem elemanlarının, ağırlıkları ayarlamak (öğrenmek) için rekabet ettikleri düşünülmektedir. En uygun çıkışa sahip işlem elemanı diğerlerine göre baskın olur ve sadece bu nöronun çıkış sağlaması söz konusudur. Kendi kendine öğrenme (self-organising) olarak da bilinen bu kural özellikle dağılımlara yönelik çalışmalarda kullanılmaktadır fakat teorik alt yapısının tam gelişmemiş olmasından dolayı uygulamada henüz yaygınlık kazanmamıştır.
- Delta Kuralı : Bu kural Hebb kuralının biraz daha geliştirilmiş şeklidir. Bu kurala göre beklenen çıkış ile gerçekleşen çıkış arasındaki farklılığı azaltmak için yapay sinir ağının elemanlarının bağlantılarının ağırlık değerlerinin sürekli değiştirilmesi ilkesine dayanarak geliştirilmiştir. Ağın ürettiği çıkış ile üretilmesi gereken (beklenen) çıkış arasındaki hatanın karelerinin ortalamasını en aza indirgemek hedeflenmektedir.

2.2. YAPAY SİNİR AĞLARI

Yapay zeka çalışmaları kapsamında ortaya çıkan ve bir noktada yapay zeka çalışmalarına destek sağlamakta olan farklı alanlardan bir tanesi de Yapay Sinir Ağları teknolojisidir. Dolayısıyla, yapay zeka alanının bir alt dalını oluşturan YSA teknolojisi öğrenebilen sistemlerin temelini oluşturmaktadır. İnsan beyninin temel işlem elemanı olan nöronu (neuron) şekilsel ve işlevsel olarak basit bir şekilde taklit eden YSA'lar, bu yolla biyolojik sinir sisteminin basit bir simülasyonu için oluşturulan programlardır. Bu şekilde, insanoğluna özgü, deneyerek (yaşayarak) öğrenme yeteneğini bilgisayar ortamına taşıyabildiği düşünülen YSA teknolojisi bir bilgisayar sistemine inanılmaz bir giriş verisinde öğrenme kapasitesi sağlamaktadır ve bir çok avantajlar sunmaktadır.

2.2.1. Yapay Sinir Ağı (YSA) Nedir?

İnsanlığın doğayı araştırma ve taklit etme çabalarının en son ürünlerinden bir tanesi Yapay Sinir Ağları teknolojisidir. Yapay Sinir Ağları, basit biyolojik sinir sisteminin çalışma şeklini simüle etmek için tasarlanan programlardır. Simüle edilen sinir hücreleri (nöronlar) içerirler ve bu nöronlar çeşitli şekillerde birbirlerine bağlanarak ağı oluştururlar. Bu ağlar öğrenme, hafızaya alma ve veriler arasındaki ilişkiyi ortaya çıkarma kapasitesine sahiptirler. Diğer bir ifadeyle, YSA'lar, normalde bir insanın düşünme ve gözlemlemeye yönelik doğal yeteneklerini gerektiren problemlere çözüm üretmektedir. Bir insanın, düşünme ve gözlemleme yeteneklerini gerektiren problemlere yönelik çözümler üretebilmesinin temel sebebi ise insan beyninin ve dolayısıyla insanın sahip olduğu yaşayarak veya deneyerek öğrenme yeteneğidir.

Yapay sinir ağları, insan beyninin özelliklerinden olan öğrenme yolu ile yeni bilgiler türetebilme, yeni bilgiler oluşturabilme ve keşfedebilme gibi yetenekleri herhangi bir yardım almadan otomatik olarak gerçekleştirmek amacı ile geliştirilen bilgisayar sistemleridir. Bu yetenekleri geleneksel programlama yöntemleri ile gerçekleştirmek oldukça zor veya mümkün değildir. O nedenle, yapay sinir ağlarının, programlanması çok zor veya mümkün olmayan olaylar için geliştirilmiş adaptif bilgi işleme ile ilgilenen bir bilgisayar bilim dalı olduğu söylenebilir.

Biyolojik sistemlerde öğrenme, nöronlar arasındaki sinaptik (synaptic) bağlantıların ayarlanması ile olur. Yani, insanlar doğumlarından itibaren bir yaşayarak öğrenme

süreci içerisine girerler. Bu süreç içinde beyin sürekli bir gelişme göstermektedir. Yaşayıp tecrübe ettikçe sinaptik bağlantılar ayarlanır ve hatta yeni bağlantılar oluşur. Bu sayede öğrenme gerçekleşir. Bu durum YSA için de geçerlidir. Öğrenme, eğitme yoluyla örnekler kullanarak olur; başka bir deyişle, gerçekleşme giriş/çıkış verilerinin işlenmesiyle, yani eğitme algoritmasının bu verileri kullanarak bağlantı ağırlıklarını (weights of the synapses) bir yakınsama sağlanana kadar, tekrar tekrar ayarlamasıyla olur. YSA'lar, ağırlıklandırılmış şekilde birbirlerine bağlanmış bir çok işlem elemanlarından (nöronlar) oluşan matematiksel sistemlerdir. Bir işlem elemanı, aslında sık sık transfer fonksiyonu olarak anılan bir denklemdir. Bu işlem elemanı, diğer nöronlardan sinyalleri alır; bunları birleştirir, dönüştürür ve sayısal bir sonuç ortaya çıkartır.

Tanım 1:

Yapay Sinir Ağları, en kısa ve basit şekilde, bir örnekler kümesi yardımıyla parametrelerin uyarlanabilmesini sağlayacak bir matematiksel formül için yazılan bilgisayar programı olarak tanımlanabilir. Bu tanım, YSA'yı en basit şekilde ve teknik detaya girilmeksizin ifade etmektedir [5].

Tanım 2:

Yine basit ama daha teknik ikinci bir tanım ise şu şekildedir: YSA, ilgili bağlantı ağırlıklarıyla (synaptic wheights) bir ağa bağlanmış basit işlem elemanlarından (nöron) oluşan bir sistemdir.

Tanım 3:

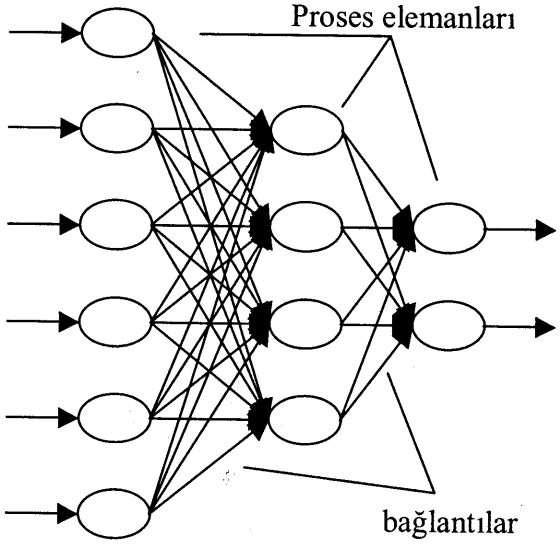
DARPA Neural Network Study (1988) [6] isimli yayında ise biraz daha açıklayıcı bir tanım kullanılmaktadır: "Bir YSA, birbirlerine paralel olarak çalışan bir çok basit işlem elemanından oluşan ve fonksiyonu, ağın yapısı, bağlantı ağırlıkları ve elemanlarda gerçekleştirilen işlemler tarafından belirlenen bir sistemdir."

Tanım 4:

Daha kapsamlı ve genel kabul gören bir tanım ise, "Bir sinir ağı, basit işlem birimlerinden oluşan, deneyimsel bilgileri biriktirmeye yönelik doğal bir eğilimi olan ve

bunların kullanılmasını sağlayan yoğun bir şekilde paralel dağıtılmış bir işlemcidir. Bu işlemci iki şekilde beyin ile benzerlik göstermektedir.” şeklindedir.

Yapay sinir ağları günümüzde bir çok probleme çözüm üretebilecek yeteneğe sahiptir. Birbirinden farklı yapay sinir ağı tanımlarının ortak birkaç noktası vardır. Bunların en başında yapay sinir ağlarının birbirine hiyerarşik olarak bağlı ve paralel olarak çalışabilen yapay hücrelerden oluşmaları gelmektedir. Proses elemanları da denilen bu hücrelerin birbirlerine bağlandıkları ve her bağlantının bir değerinin olduğu kabul edilmektedir. Bilginin öğrenme yolu ile elde edildiği ve proses elamanlarının bağlantı değerlerinde saklandığı, dolayısıyla dağıtık bir hafızanın söz konusu olduğu da ortak noktaları oluşturmaktadır. Proses elemanlarının birbirleri ile bağlanmaları sonucu oluşan ağa yapay sinir ağı denmektedir (Şekil 2.1). Yapay sinir ağları aynı zamanda, bağlantılı ağlar, paralel dağıtılmış ağlar (parallel distributed networks), nöromorfik sistemler (neuromorfic systems) olarak da adlandırılmaktadır. Bu ağın oluşturulması biyolojik sinir sistemi hakkındaki bulgulara dayanmaktadır.



Şekil 2.1: Bir yapay sinir ağı örneği [1]

2.2.2. Yapay Sinir Ağlarının Görevleri

Yapay sinir ağları, insanlar tarafından gerçekleştirilmiş örnekleri (gerçek beyin fonksiyonlarının ürünü olan örnekleri) kullanarak olayları öğrenebilen, çevreden gelen olaylara karşı nasıl tepkiler üretileceğini belirleyebilen bilgisayar sistemleridir. İnsan beyninin fonksiyonel özelliklerine benzer şekilde,

- Öğrenme
- İlişkilendirme
- Sınıflandırma
- Genelleme
- Özellik belirleme ve
- Optimizasyon

gibi konularda başarılı bir şekilde uygulanmaktadırlar [7]. Örneklerden elde ettikleri bilgiler ile kendi deneyimlerini oluşturur; ve daha sonra, benzer konularda benzer kararlan verirler.

Teknik olarak , bir yapay sinir ağının en temel görevi, kendisine gösterilen bir giriş setine karşılık gelebilecek bir çıkış seti belirlemektir. Bunu yapabilmesi için ağ, ilgili olayın örnekleri ile eğitilerek (öğrenme) genelleme yapabilecek yeteneğe kavuşturulur. Bu genelleme ile benzer olaylara karşılık gelen çıktı setleri belirlenir. Ağı oluşturan proses elemanları, bunların bilgileri işleme yetenekleri, birbirleri ile bağlantılarının şekilleri değişik modelleri oluşturmaktadır.

2.2.3. Yapay Sinir Ağlarının Temel Özellikleri Ve Avantajları

Bütün yapay sinir ağı modelleri için geçerli olan genel karakteristik özellikler vardır. Bunlar aşağıdaki gibi sıralanabilir [8] :

- Doğrusal olmama :

  YSA' nın temel işlem elemanı olan hücre doğrusal değildir. Dolayısıyla hücrelerin birleşmesinden meydana gelen YSA da doğrusal değildir ve bu özellik bütün ağa yayılmış durumdadır. Bu özelliği ile YSA, doğrusal olmayan karmaşık problemlerin çözümünde en önemli araç olmuştur.

- Programları çalışma stili bilinen programlama yöntemlerine benzememektedir :
  Geleneksel programlama ve yapay zeka yöntemlerinin uygulandığı bilgi işleme yöntemlerinden tamamen farklı bir bilgi işleme yöntemi vardır.
- Yapay sinir ağları makine öğrenmesi gerçekleştirirler :
  Yapay sinir ağlarının temel işlevi bilgisayarların öğrenmesini sağlamaktır. Olayları öğrenerek benzer olaylar karşısında benzer kararlar vermeye çalışırlar.
- Bilginin saklanması :
  Yapay sinir ağlarında bilgi ağın bağlantılarının değerleri ile ölçülmekte ve bağlantılarda saklanmaktadır. Diğer programlarda olduğu gibi veriler bir veri tabanında veya programın içinde gömülü değildir. Bilgiler ağın üzerinde saklı olup ortaya çıkartılması ve yorumlanması zordur.
- Yapay sinir ağları örnekler ile öğrenirler :
  Yapay sinir ağlarının olayları öğrenebilmesi için o olay ile ilgili örneklerin belirlenmesi gerekmektedir. Örnekleri kullanarak ilgili olay hakkında genelleme yapabilecek yeteneğe kavuşturulurlar (adaptif öğrenme). Örnek bulunamıyorsa yapay sinir ağının eğitilmesi mümkün değildir. Örnekler ise gerçekleşmiş olan olaylardır. Elde edilen örneklerin olayı tamamı ile gösterebilmesi çok önemlidir. Ağa olay bütün yönleri ile gösterilemez ve ilgili örnekler sunulmaz ise başarılı sonuçlar elde edilemez.. O nedenle örneklerin oluşturulması ve toplanması yapay sinir ağı biliminde özel bir öneme sahiptir.
- Önce eğitilmeleri daha sonra performanslarının test edilmesi gerekmektedir :
  Yapay sinir ağlarının eğitilmesi demek, mevcut örneklerin tek tek ağa gösterilmesi ve ağın kendi mekanizmalarını çalıştırarak örnekteki olaylar arasındaki ilişkileri belirlemesidir. Ağı eğitmek için bulunan örnekler test grubu ve eğitim gurubu olmak üzere ikiye bölünür. Her ağ önce eğitim seti ile eğitilir. Ağ bütün örneklere doğru cevaplar vermeye başlayınca eğitim işi tamamlanmış kabul edilir. Daha sonra ağın hiç görmediği test setindeki örnekler ağa gösterilerek ağın verdiği cevaplara bakılır. Eğer ağ hiç görmediği örneklere kabul edilebilir bir doğrulukta cevap veriyor ise o zaman ağın performansı iyi kabul edilir ve ağ kullanıma alınarak gerekirse çevrimiçi (on-line) kullanılır. Eğer ağın performansı yetersiz olursa o zaman yeniden eğitmek veya yeni örnekler ile eğitmek gibi bir çözüme gidilir.

- Eksik bilgi ile çalışabilmektedir :

  Geleneksel sistemler bilgi eksik ise çalışmazlar fakat yapay sinir ağları eğitildikten sonra eksik bilgiyle çalışabilir. Yapay sinir ağlarının eksik bilgiler ile çalışması performanslarının düşük olması anlamına gelmez.. Performansın düşmesi eksik olan bilginin önemine bağlıdır. Hangi bilginin önemli olduğuna kullanıcı değil, ağ eğitim sırasında öğrenerek karar verir. Ağ performansı düşerse, kayıp olan bilginin önemli olduğu kararına varılır.
- Görülmemiş örnekler ile ilgili bilgi üretirler :

  Ağ kendisine gösterilen örneklerden genellemeler yaparak daha önce görmediği örnekler hakkında bilgi üretebilir.
- Hata toleransına sahiptirler :

  Yapay sinir ağları elemanlarının bozulması ve çalışamaz gelmesi halinde ağ çalışmaya devam eder. Ağın bozuk olan hücrelerinin işlevlerinin önemine göre ağın performansında düşmeler görülebilir. Hangi hücrelerin işlevlerinin önemli olduğuna ağ eğitim sırasında kendisi karar verir. Bunu kullanıcı bilmemektedir. Ağın bilgisinin yorumlanamamasının sebebi de budur. Yapay sinir ağları dereceli bozulma (graceful degradation) gösterirler. Hatalara karşı toleranslı olmaları bozulmalarının da dereceli olmasına neden olmaktadır. Bir ağ eğitilirken yavaş yavaş bozulur. Bu eksik olan bilgiden veya hücrelerin bozulmasından kaynaklanır. Ağlar, herhangi bir problem ortaya çıktığında hemen anında bozulmazlar.
- Dağıtık belleğe sahiptirler :

  Yapay sinir ağlarında bilgi ağa yayılmış durumdadır. Hücrelerin birbirleri ile bağlantılarının değerleri ağın bilgisini gösterir. Tek bir bağlantının bir anlamı yoktur. Ağın bilgilerinin açıklanamamasının sebeplerinden birisi de budur. Bu ağlarda, ağın tamamı öğrendiği olayın bütününü karakterize etmektedir. O nedenle bilgiler ağa dağıtılmış durumdadır. Bu ise dağıtık bir belleğin doğmasına neden olmaktadır.
- Sadece nümerik bilgiler ile çalışabilmektedirler :

  Yapay sinir ağları sadece nümerik bilgiler ile çalışırlar. Sembolik ifadeler ile gösterilen bilgilerin nümerik gösterime çevrilmeleri gerekmektedir. Fakat sembolik bilgilerin nümerik değerler ile ifade edilmesinde bilgilerin yorumlanmasını ve kararların açıklanmasını zorlaştırmaktadır.

- Yerel İşlem ve Esneklik Olanağı :

  YSA'lar, geleneksel işlemcilerden farklı şekilde işlem yapmaktadırlar. Geleneksel işlemcilerde, tek bir merkezi işlem elemanı her hareketi sırasıyla gerçekleştirir. YSA modelleri, her biri büyük bir problemin bir parçası ile ilgilenen çok sayıda basit işlem elemanlarından oluşma ve bağlantı ağırlıklarının ayarlanabilmesi gibi özelliklerinden dolayı önemli derecede esnek bir yapıya sahiptirler. Bu esnek yapı sayesinde ağın bir kısmının zarar görmesi modelde sadece performans düşüklüğü yaratır. Modelin işlevini tamamen yitirmesi söz konusu olmaz. Ayrıca, toplam işlem yükünü paylaşan işlem elemanlarının birbirleri arasındaki yoğun bağlantı yapısı sinirsel hesaplamanın temel güç kaynağıdır. Bu yerel işlem yapısı sayesinde, YSA yöntemi en karmaşık problemlere bile uygulanabilmekte ve tatminkar çözümler sağlayabilmektedir.
- Sınırsız Sayıda Değişken ve Parametre Kullanılabilir :

  YSA modelleri sınırsız sayıda değişken ve parametre ile çalışabilmektedir. Bu sayede mükemmel bir öngörü doğruluğu ile genel çözümler sağlanabilmektedir.

2.2.4. Yapay Sinir Ağlarının Kullanım Alanları

YSA'lar gerçek hayatta karşılaşılan problemlerde oldukça geniş bir uygulama alanı kazanmışlardır. Bugün, bir çok endüstride başarılı şekilde kullanılmaktadırlar. Uygulama alanları için bir sınır yoktur fakat, öngörü, modelleme ve sınıflandırma gibi bazı alanlarda ağırlıklı olarak kullanılmaktadır. YSA'lar 1950'li yıllarda ortaya çıkmalarına rağmen, ancak 1980'li yılların ortalarında genel amaçlı kullanım için yeterli seviyeye gelmişlerdir. Bugün, YSA'lar bir çok ciddi problem üzerinde uygulanmaktadır ve bu problemlerin sayısı giderek artmaktadır. Verideki yapıyı (pattern) en iyi tanımlayan yöntem olmaları dolayısıyla, tahmin (prediction) ve öngörü işlemleri için çok uygundurlar. YSA'ların gerçek hayattaki yaygın uygulama alanlarına şu örnekler verilebilir [9]:

- Optik karakter tanıma
- Robot hareket mekanizmalarının kontrol edilmesi
- Veri madenciliği
- Güvenlik sistemlerinde konuşma ve parmak izi tanıma
- İletişim kanallarındaki trafik yoğunluğu kontrol etme ve anahtarlama

- Kalite Kontrol
- Kredi kartı hilelerini saptama
- Kanserin saptanması ve kalp krizlerinin tedavisi
- Zeki araçlar ve robotlar için optimum rota belirleme
- Üretim planlama ve çizelgeleme

2.2.5. Yapay Sinir Ağlarının Dezavantajları

YSA' ların bazı dezavantajları bulunmaktadır. Bu dezavantajlarına rağmen yapay sinir ağları tarafından her problem için değişik şekillerde çözümler üretilebilmekte ve başarılı uygulamalar oluşturmak mümkün olabilmektedir. Bu nedenle, dezavantajları yapay sinir ağlarına olan ilgiyi düşürmek için görmemek gerekir. Bu dezavantajları şu şekilde sayabiliriz :

- Problemin öğrenilecek ağa gösterimi çok önemlidir :
  Yapay sinir ağları sadece nümerik bilgiler ile çakışmaktadırlar. Problemin nümerik gösterime dönüştürülmesi kullanıcının tecrübesine bağlıdır. Uygun bir gösterim mekanizmasının kurulamamış olması problemin çözümünü engelleyebilir veya düşük performanslı bir öğrenme (çözüm) elde edilebilir. Problemin nümerik gösterimi mümkün olsa bile bunun ağa gösteriliş şekli problemin başarılı bir şekilde çözülmesini yakından etkiler. Bu, günümüzde bazı olayın yapay sinir ağları ile çözülememesinin en önemli nedenlerinden birisidir.
- Tüm giriş ve çıkışlar [0-1] aralığında olmalıdır :
  Yapay sinir ağlarında tüm giriş ve çıkışlar genellikle 0 ile 1 arasındadır. Bu durum tek tek dönüştürme ve işlem oluşturur. Bu oluşumdan CPU gücü ve disk alanı etkilenir. Dönüştürme işleminin seçimine göre ağ çıkışı değişebilir.
- Katsayılar ile ilgili belirli bir kural yoktur :
  Bazı ağlarda ağın parametre değerlerinin (mesela öğrenme katsayısı, her katmanda olması gereken proses elemanı sayısı, katman sayısı vb.) belirlenmesinde bir kural olmaması diğer bir problemdir. Bu, iyi çözümler bulmayı zor durumda bırakan bir etken olarak görülebilir. Her problem için ayrı faktörleri dikkate almak gerekebilir. Bu parametre değerleri için belirli standartların oluşturulması çok zor olduğundan her problem için ayrı ayrı

değerlendirmeler yapılması gerektirmektedir. Bu da önemli bir dezavantaj olarak görülebilir.

- Yapay sinir ağları donanım bağımlı çalışırlar :

  Yapay sinir ağlarının donanım bağımlı çalışmaları önemli bir sorun olarak görülebilir. Ağlar paralel işlemciler üzerinde çalışabilmektedirler. Ağların özellikle, gerçek zamanlı bilgi işleyebilmeleri paralel çalışabilen işlemcilerin varlığına bağlıdır. Günümüzdeki makinelerin çoğu seri şekilde çalışabilmekte ve aynı zamanda sadece tek bir bilgiyi işleyebilmektedir. Paralel işlemleri seri makinelerde yapmak ise zaman kaybına yol açar. Bir ağın nasıl oluşturulması gerektiğini belirleyecek kuralların olmaması da başka bir dezavantajdır.

- Eğitimin bitimine karar verilmesi :

  YSA eğitiminin sonlandırılması kararını vermek için geliştirilmiş bir yöntem yoktur. Eğitimin tamamlanması için ağın örnekler üzerindeki hatasının belirli bir değerin altına inmesi yeterli olmasına rağmen, optimum öğrenmenin gerçekleştiği söylenemez. Yalnızca iyi çözümler üretebilen bir ağın oluştuğu söylenebilir. Optimum sonuçları veren bir mekanizma henüz geliştirilmemiştir fakat buna yönelik çalışmalar sürmektedir.

2.2.6. Tarihsel Gelişim

1950'li yılların sonlarında, büyük ölçekli işlemcilerin geliştirilmesiyle, beynin yaptığı işlemleri yapabilecek sinir ağlarının oluşturulabilmesi mümkün hale gelmiştir. Gerçekten de, YSA'lar dijital işlemcilerin geliştirilmesinden sonra işlem yöntemi olarak önemli bir yeni yaklaşım olarak görülmektedir.

YSA simülasyonları yeni bir gelişme olarak görülmektedir. Bununla beraber, bu alan bilgisayarın çıkışından önce ortaya çıkmıştır ve bir bocalama devresi geçirdikten sonra yoluna devam etmiştir.

Bilgisayarların yaygın bir şekilde kullanılmaya başlanmasıyla birlikte, YSA alanında oldukça önemli gelişmeler olmuştur. Bu alandaki araştırmalar ve çalışmalar büyük bir ilgi ile başlamış fakat beklenen gelişmelerin gerçekleşmemesi sonucunda ilgi azalmış ve bir suskunluk dönemi başlamıştır. Profesyonel ve maddi katkının minimum olduğu bu dönemde, sadece birkaç araştırmacı tarafından katkı sağlanmıştır. Bu araştırmacılar,

Minsky ve Papert [10] tarafından tanımlanan sınırlamaları etkisiz kılan bir teknoloji geliştirmişlerdir. Minsky ve Papert, 1969 yılında bir kitap yayınlamışlardır ve bu kitapta, araştırmacılar arasında ön plana çıkan ve ekstra analiz yapılmadan kabul gören YSA'lara karşı bazı olumsuzlukları toplamışlardır. Son yıllarda ise, YSA alanı ilgi ve katkı olarak yeniden canlanmaktadır. YSA tarihi, dönemler itibariyle incelenebilir.

- İlk Girişimler : Bu dönemde, genel mantığı kullanan başlangıç simülasyonları yapılmıştır. McCulloch ve Pitts, kendi nöroloji anlayışları çerçevesinde YSA modelleri geliştirmişlerdir. Bu modeller, nöronların çalışma şekilleri hakkında bazı varsayımlarda bulunmuştur. Oluşturdukları ağlar, sabit eşiklere sahip ikili (binary) aletler olarak görülen basit nöronları baz almıştır. Modellerinin sonuçları, "a veya b" ve "a ve b" gibi basit mantıksal fonksiyonlardı. Diğer bir girişim, bilgisayar simülasyonları kullanılarak yapılmıştır. Bu noktadaki katkılar iki araştırmacı grubu tarafından yapılmıştır: Farley ve Clark (1954) ve Rochester, Holland, Haibit ve Duda (1956). Özellikle ilk grup, ki bunlar IBM araştırmacılarıdır, modellerini çalıştıramamışlar ve McGill Üniversitesinden nörobilimcilerle ortak bir çalışma yapmışlardır. Bu etkileşim, günümüze kadar süren, çok disiplinli bir trend oluşturmuştur.
- Umut Verici Gelişmeler : YSA'ların gelişmesinde tek etkisi olan nörobilim değildir, psikologlar ve mühendisler de YSA simülasyonundaki ilerlemye katkı sağlamıştır. Rosenblatt (1958) [11] Perceptron'u tasarlayıp geliştirdikten sonra, bu alandaki ilgi ve etkinlik canlanmaya başlamıştır. Perceptron üç tabaka içermekteydi ve orta tabaka birleştirme tabakası olarak adlandırılmaktaydı. Bu sistem, bir veri giriş kümesinin bir rassal çıkışa bağlanma veya birleşme şeklini öğrenebilmekteydi. Burada öğrenme kelimesi bağlantı ağırlıklarının ilişkiye göre ayarlanması anlamında kullanılmaktadır. Diğer bir sistem (ADALINE – Adaptive Linear Element) ise Stanford Üniversitesinden Widrow ve Hoff tarafından 1960 yılında geliştirilmiştir. Basit bileşenlerden oluşan bir analog elektronik alet olan ADALINE, kullanılan öğrenme yöntemi ile perceptrondan farklılaşmıştır. Bu sistemde En Küçük Ortalama Kareler (LMS – Least Mean Squares) öğrenme kuralı kullanılmıştır.

- Olumsuz Gelişmeler : 1969 yılında, Minsky ve Papert bir kitap yazmış ve bu kitapta çok tabakalı sistemlere göre tek tabakalı Perceptronların sahip olduğu sınırlamaları ortaya koymuşlardır. Kitabın ana fikri şu şekilde özetlenebilir: "…bizim sezgisel görüşümüz çok tabakalı sistemlere genişlemenin verimsiz olduğudur.". Kitapta ortaya konulan bu önemli sonuç sonrasında YSA simülasyonlarına yönelik araştırmalar hem ilgi hem de kaynak kaybına uğramıştır. Sonuç olarak, bu alana yönelik önemli bir önyargı oluşmuştur. Minsky ve Papert tarafından altı çizilen sorun YSA literatüründe XOR Problemi olarak bilinmektedir.

- Yenilikler : İlgi ve kaynağın minimum düzeyde olmasına rağmen bazı araştırmacılar yapı tanımlama (pattern recognition) gibi problemlerin çözümüne yönelik çalışmalarını sürdürmüşlerdir. Bu dönem süresince bazı paradigmalar ortaya çıkmıştır. Grossberg ve Carpenter (1995) tarafından yapılan çalışmalar, yankı (resonating) algoritmaları araştıran bir düşünce okulunun temellerini atmıştır. Bu araştırmacılar, temeli biyolojik olarak teorik gelişmelerle ilgilenmiştir. Adaptif yapı (pattern) sınıflandırması konusu üzerine bir makale yayınlamıştır ve bu makalede bir öğrenme temeli (error-correction method – hata düzeltme metodu) için bir matematiksel teori oluşturmuştur. Fukushima ise el yazısı karakterleri yorumlamak için bir adım adım (step wise) eğitilmiş çok tabakalı YSA oluşturmuştur. Cognitron olarak adlandırılan bu model 1975 yılında yayınlanmıştır. makul modellere dayanan ART (Adaptive Resonance Theory – Adaptif Rezonans Teorisi) ağlarını geliştirmişlerdir. Anderson ve Kohonen ise birbirlerinden bağımsız olarak benzer teknikler geliştirmişlerdir. Klopf, 1972 yılında, yapay nöronlarda öğrenme işlemi için, "heterostasis" olarak adlandırılan ve nöronsal öğrenmenin biyolojik prensiplerine dayanan bir temel oluşturmuştur. Werbos (1974) geri-besleme öğrenme metodunu geliştirmiş ve kullanmıştır ve bir kaç yıl sonrasında bu metot oldukça popülarite kazanmıştır. Geri-besleme ağlar, bugün, en çok bilinen ve kullanılan yapay sinir ağlarıdır. Geri-besleme ağ aslında, yapay nöronunda farklı bir eşik fonksiyonuna sahip ve daha sağlam (robust) ve yetenekli öğrenme kuralı olan bir çok tabakalı perceptrondur.

- Yeniden Canlanma : 1970'li yılların sonlarında ve 1980'li yılların başlarındaki ilerleme, yapay sinir ağları alanına ilginin yeniden canlanması bakımından önemlidir. Bu hareketi bir kaç faktör etkilemiştir. Örneğin, ayrıntılı kitaplar ve konferanslar çok farklı alanlarda uzmanlaşmış insanlara bir forum imkanı ve dolayısı ile bir etkileşim sağlamıştır. Akademik programlar oluşturulmuş ve en önemli üniversitelerde dersler açılmıştır. Artan ilgi ile beraber bu alandaki araştırmalara yönelik fonlar da artmış ve enstitüler ortaya çıkmıştır.

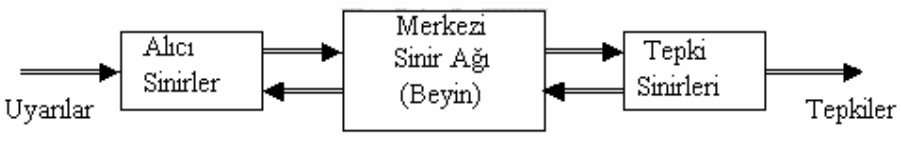
- Bugün: Sağlanan önemli ilerleme yapay sinir ağları alanında daha ileri araştırmalar için gerekli ilgi ve bilgi birikimini sağlamıştır. Sinir sistemi tabanlı işlemciler oluşturulmakta ve komplike problemlerin çözümüne yönelik uygulamalar gelişmektedir. Kısacası, bu alan günümüzde bir geçiş dönemi içinde görülmektedir.

YSA'lar 1950'li yıllarda ortaya çıkmalarına rağmen, ancak 1980'li yılların ortalarında genel amaçlı kullanım için yeterli seviyeye gelmişlerdir.

2.3. YSA' LARIN YAPISI VE ÇALIŞMA TEKNİGİ

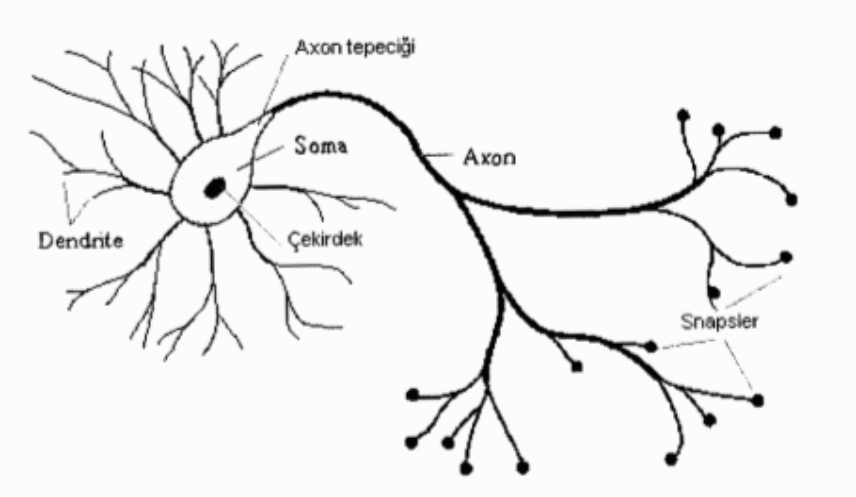
2.3.1. Biyolojik Sinir Sistemi

Biyolojik sinir sistemi, merkezinde sürekli olarak bilgiyi alan, yorumlayan ve uygun bir karar üreten beynin (merkezi sinir ağı) bulunduğu 3 katmanlı bir sistem olarak açıklanır. Alıcı sinirler (receptor) organizma içerisinden ya da dış ortamlardan algıladıkları uyarıları, beyine bilgi ileten elektriksel sinyallere dönüştürür. Tepki sinirleri (effector) ise, beyinin ürettiği elektriksel darbeleri organizma çıktısı olarak uygun tepkilere dönüştürür. Şekil 2.2 de bir sinir sisteminin blok gösterimi verilmiştir.



Şekil 2.2: Biyolojik sinir sisteminin blok gösterimi [9]

Merkezi sinir ağında bilgiler, alıcı ve tepki sinirleri arasında ileri ve geri besleme yönünde değerlendirilerek uygun tepkiler üretilir. Bu yönüyle biyolojik sinir sistemi, kapalı çevrim denetim sisteminin karakteristiklerini taşır. Merkezi sinir sisteminin temel işlem elemanı, sinir hücresidir (nöron) ve insan beyninde yaklaşık 10 milyar sinir hücresi olduğu tahmin edilmektedir. Sinir hücresi; hücre gövdesi, dendriteler ve axonlar olmak üzere 3 bileşenden meydana gelir. Dendriteler, diğer hücrelerden aldığı bilgileri hücre gövdesine bir ağaç yapısı şeklinde ince yollarla iletir. Axonlar ise elektriksel darbeler şeklindeki bilgiyi hücreden dışarı taşıyan daha uzun bir yoldur. Axonların bitimi, ince yollara ayrılabilir ve bu yollar, diğer hücreler için dendriteleri oluşturur. Şekil 2.3 de görüldüğü gibi axon-dendrite bağlantı elemanı synapse olarak söylenir.



Şekil 2.3: Biyolojik sinir hücresinin yapısı [4]

Synapse gelen ve dendriteler tarafından alınan bilgiler genellikle elektriksel darbelerdir ancak, synapsedeki kimyasal ileticilerden etkilenir. Belirli bir sürede bir hücreye gelen girişlerin değeri, belirli bir eşik değerine ulaştığında hücre bir tepki üretir. Hücrenin tepkisini artırıcı yöndeki girişler uyarıcı, azaltıcı yöndeki girişler ise önleyici girişler olarak söylenir ve bu etkiyi synapse belirler.

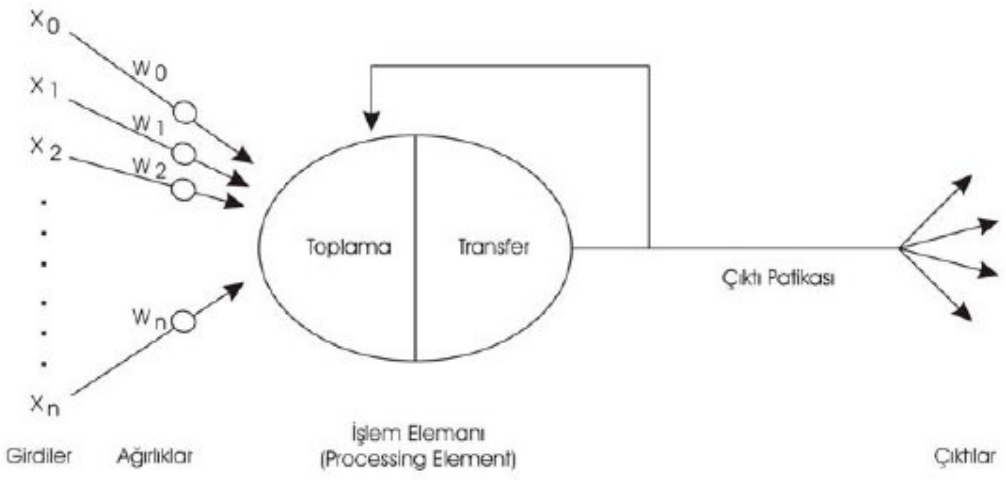
İnsan beyninin 10 milyar sinir hücresinden ve 60 trilyon synapse bağlantısından oluştuğu düşünülürse son derece karmaşık ve etkin bir yapı olduğu anlaşılır. Diğer taraftan bir sinir hücresinin tepki hızı, günümüz bilgisayarlarına göre oldukça yavaş

olmakla birlikte duyusal bilgileri çok hızlı değerlendirebilmektedir. Bu nedenle insan beyni; öğrenme, birleştirme, uyarlama ve genelleştirme yeteneği nedeniyle son derece karmaşık, doğrusal olmayan ve paralel dağılmış bir bilgi işleme sistemi olarak tanımlanabilir.

2.3.2. Yapay Sinir Hücresi Ve Elemanları

Beynin üstün özellikleri, bilim adamlarını üzerinde çalışmaya zorlamış ve beynin nörofiziksel yapısından esinlenerek matematiksel modeli çıkarılmaya çalışılmıştır. Beynin bütün davranışlarını tam olarak modelleyebilmek için fiziksel bileşenlerinin doğru olarak modellenmesi gerektiği düşüncesi ile çeşitli yapay hücre ve ağ modelleri geliştirilmiştir.

Biyolojik sinir ağlarının sinir hücreleri olduğu gibi yapay sinir ağlarının da yapay sinir hücreleri vardır. Yapay sinir hücrelerine mühendislik biliminde proses elemanları da denmektedir. Yapay sinir hücreleri, YSA' nın çalışmasına esas teşkil eden en küçük bilgi işleme birimidir. Geliştirilen hücre modellerinde bazı farklılıklar olmakla birlikte genel özellikleri ile bir yapay hücre modeli, girişler, ağırlıklar, birleştirme fonksiyonu, aktivasyon (etkinleştirme) fonksiyonu ve çıkışlar olmak üzere 5 bileşenden meydana gelir (Şekil 2.4). Bunlar:



Şekil 2.4: Yapay sinir hücresinin yapısı

- Girişler : Yapay sinir hücresinin girişleridir. Girişler dışarıdan gelen bilgilerdir. Bunlar ağın öğrenmesi istenen örnekler tarafından belirlenir.
- Ağırlıklar : Ağırlıklar yapay sinir hücresine gelen bilginin önemini ve hücre üzerindeki etkisini gösterir. Şekil 2.4 de Ağırlık 1, Giriş 1'in hücre üzerindeki etkisini göstermektedir. Ağırlık değerlerinin pozitif, negatif veya 0 olması o ağırlığın önemsiz olması anlamına gelmez. Ağırlıklar değişken veya sabit değerler olabilir.
- Toplam fonksiyonu : Bu fonksiyon yapay sinir hücresine gelen net girişi hesaplar. Bunun için değişik fonksiyonlar kullanılmaktadır. En yaygın olarak kullanılanı ağırlıklı toplamdır. Burada her gelen girdi değeri kendi ağırlığı ile çarpılarak toplanır. Böylece ağa gelen net girdi bulunmuş olur. Bu şu şekilde formülize edilir:

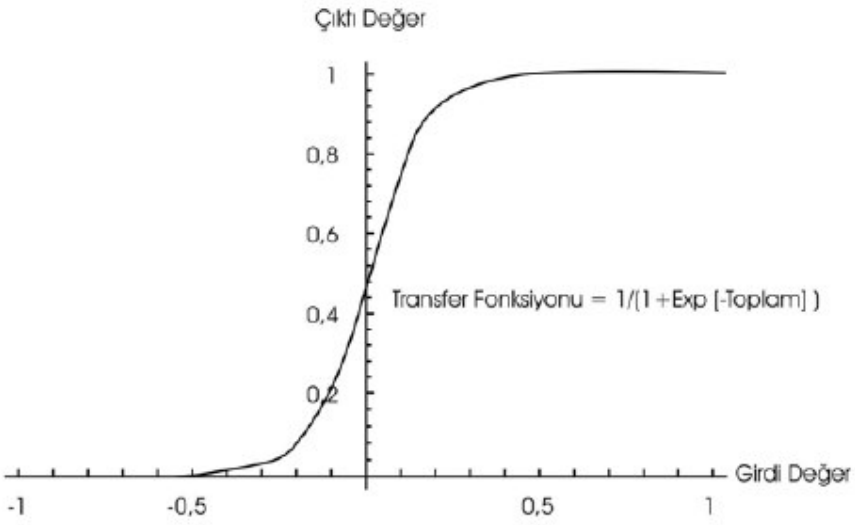
$$NET = \sum_{i}^{n} G_i A_i \tag{2.1}$$

Formülde G girişleri, A ağırlıkları n ise hücreye gelen toplam giriş sayısını göstermektedir.

- Ativasyon fonksiyonu: Bu fonksiyon, hücreye gelen net girişi işler ve hücrenin girişe karşılık üreteceği çıkışı belirler. Aktivasyon fonksiyonunda ağın proses elemanlarının hepsinin aynı fonksiyonu kullanması gerekmez. Aktivasyon fonksiyonu probleme uygun olarak seçilir. Uygun fonksiyonu gösteren formül bulunmuş değildir. Aktivasyon fonksiyonları sabit parametreli yada uyarlanabilir parametreli seçilebilir. Günümüzde en çok kullanılan çok katmanlı algılayıcı modellerde genel olarak sigmoid aktivasyon fonksiyonu kullanılır (Şekil 2.5). Sürekli ve doğrusal olmayan bir fonksiyon olması nedeniyle doğrusal olmayan problemlerin çözümünde kullanılan YSA'larında tercih edilir. Bu fonksiyon şu formül ile gösterilmektedir:

$$F(NET) = \frac{1}{1 + e^{-NET}} \tag{2.2}$$

Burada NET, proses elemanına gelen net giriş değerini gösterir.



Şekil 2.5: Transfer fonksiyonunun çalışma yapısı

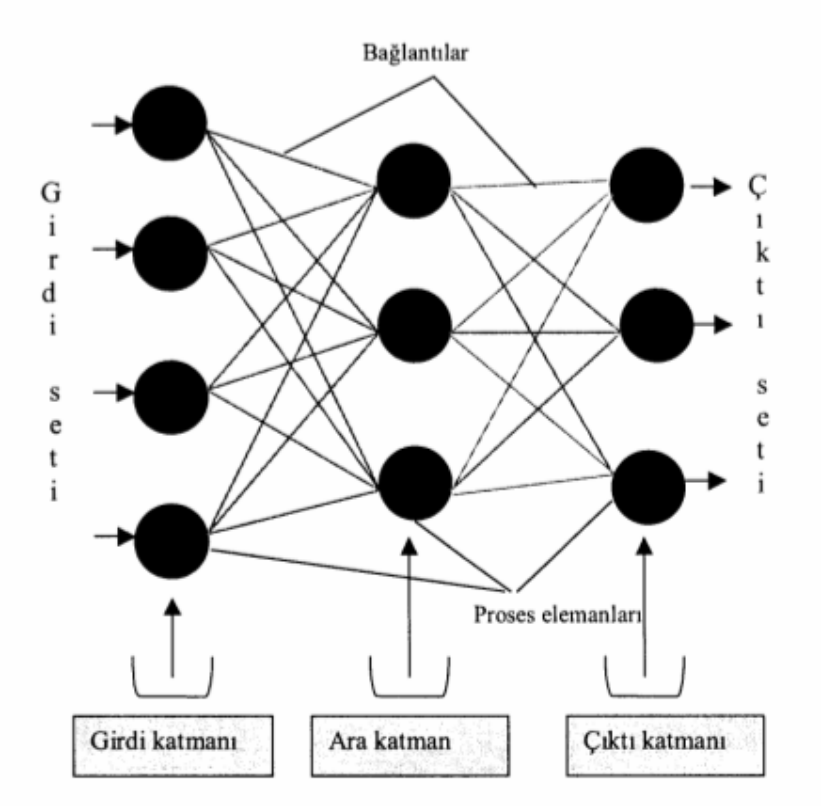
- Hücrenin çıktısı : Aktivasyon fonksiyonu ile belirlenen çıktı değeridir. Üretilen çıktı başka bir hücreye gönderilir. Bir hücrenin birden fazla girdisi olmasına rağmen tek çıktısı vardır.

2.3.3. Yapay Sinir Ağının Yapısı

Yapay sinir hücreleri biraraya gelerek yapay sinir ağını oluştururlar. Hücrelerin biraraya gelmesi rasgele olmaz. Bir yapıyı dizayn etmenin en kolay yolu elemanları tabakalandırmaktır. Burada tabakalandırmanın üç bölümü vardır. Bunlar, nöronları tabakalar halinde gruplandırmak, tabakalar arasındaki bağlantıları gruplandırmak ve son olarak ise toplama ve transfer fonksiyonlarını gruplandırmaktır. Tek tabaka ya da tek eleman içeren bazı başarılı ağlar oluşturulabilmesine rağmen çoğu uygulamalar en az üç tabaka içeren ağlara ihtiyaç duymaktadır (Şekil 2.6). Bu katmanlar:

- Girdi Katmanı (Input Layer): Bu katmandaki hücreler dışarıdan aldıkları girişleri işleyerek ara katmanlara iletirler.
- Ara Katmanlar (Hidden Layer): Ara katmanlarda, girdi katmanından gelen bilgiler işlenerek çıktı katmanına gönderilir. Bir ağda birden çok ara katman olabilir.

- Çıktı Katmanı (Output Layer): Bu katmandaki proses elemanları ara katmandan gelen bilgileri işleyerek ağın girdi seti için üretmesi gereken çıktıyı üretirler.



Şekil 2.6: Bir yapay sinir ağı örneği [12]

Şekil 2.6 daki yuvarlak şekiller proses elemanlarını göstermektedir. Her katmanda birbirine paralel elemanlar bulunmaktadır. Proses elemanlarını birbirine bağlayan çizgiler ise ağ bağlantılarını göstermektedir. Proses elemanları ve bağlantıları bir yapay sinir ağını oluşturur. Çoğu ağ türünde, gizli tabakadaki bir nöron sadece bir önceki tabakanın tüm nöronlarından sinyal alır. Nöron işlemini yaptıktan sonra ise çıktısını bir sonraki tabakanın tüm nöronlarına gönderir. Bu yapı ağın çıktısı için bir ileri besleme patikası oluşturur. Bu bir nörondan diğerine olan iletişim hattı, sinir ağları için önemli bir parçadır. Diğer bir bağlantı şekli ise geriye yayılımdır (feedback). Geriye yayılım bağlantı, bir tabakanın çıktısının önceki tabakaya gönderilmesidir.

2.3.4 Yapay Sinir Ağlarında Öğrenme Ve Test Etme

Yapay sinir hücreleri bağlantılarının ağırlık değerlerinin belirlenmesine ağın eğitilmesi denir. Başlangıç ağırlık değerleri rasgele verilir. Örnekler teker teker ağa gösterildikçe bu ağırlık değerleri değişir. En doğru ağırlık değerleri bulunmaya çalışılır. Ağın doğru

ağırlık değerlerine ulaşması örneklerin temsil ettiği olay hakkında genellemeler yapabilme yeteneğine kavuşması demektir. Bu genelleştirme özelliğine kavuşması işlemine ağın öğrenmesi denir.

Yapay sinir ağlarında öğrenme iki aşamalıdır. İlk aşamada gösterilen örnekler için ağın üreteceği çıktı belirlenir. Bu çıktı değerinin doğruluk derecesine göre ikinci aşamada ağın bağlantılarının sahip olduğu ağırlıklar değiştirilir.

Ağın eğitimi tamamlandıktan sonra öğrenip öğrenmediğini ölçmek, performans değerlendirmek için yapılan denemelere ise ağın test edilmesi denir. Test aşamasında ağın öğrenme sırasında görmediği örnekler kullanılır. Bu örneklere test seti, eğitilirken ağa gösterilen örneklere ise eğitim seti denir. Test etme sırasında ağın ağırlık değerleri değiştirilmez. Ağ eğitim sırasında belirlenen bağlantı ağırlıklarını kullanarak görmediği test seti örnekleri için çıktılar üretir. Sonuçlar ne kadar iyi olursa eğitimin performansı da o kadar iyi demektir. Yapay ağlarının bu şekilde bilinen örneklerden belirli bilgilere ulaşarak bilinmeyen örnekler hakkında yorum yapabilme (genelleme yapabilme) yeteneğine "adaptif öğrenme" denir. Bu çalışmada el yazısı karakterlerini tanımada geriye yayılım (backpropagation) öğrenme algoritması kullanılmıştır.

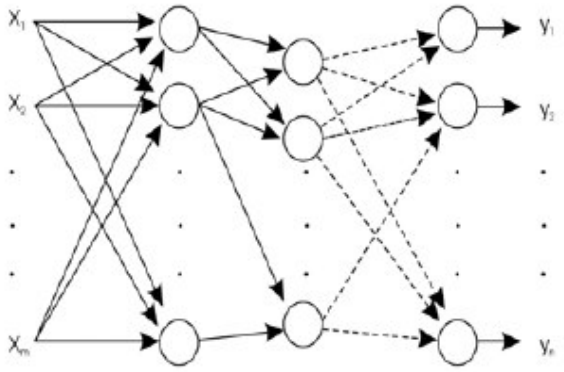
2.4. ÇOK KATMANLI ALGILAYICILAR (MULTILAYER PERCEPTRONS)

2.4.1. Çok Katmanlı Algılayıcı

Yapay sinir ağlarına aynı zamanda birden çok katmandan oluştuğu için çok katmanlı algılayıcı da denmektedir (ÇKA). Çok katmanlı algılayıcı ileriye doğru bağlantılı ve üç katmandan oluşur. ÇKA ağında bilgiler giriş katmanından ağa sunulur ve ara katmanlardan geçerek çıktı katmanına gider ve ağa sunulan girdilere karşılık gelen ağın cevabı dışarıya iletilir.

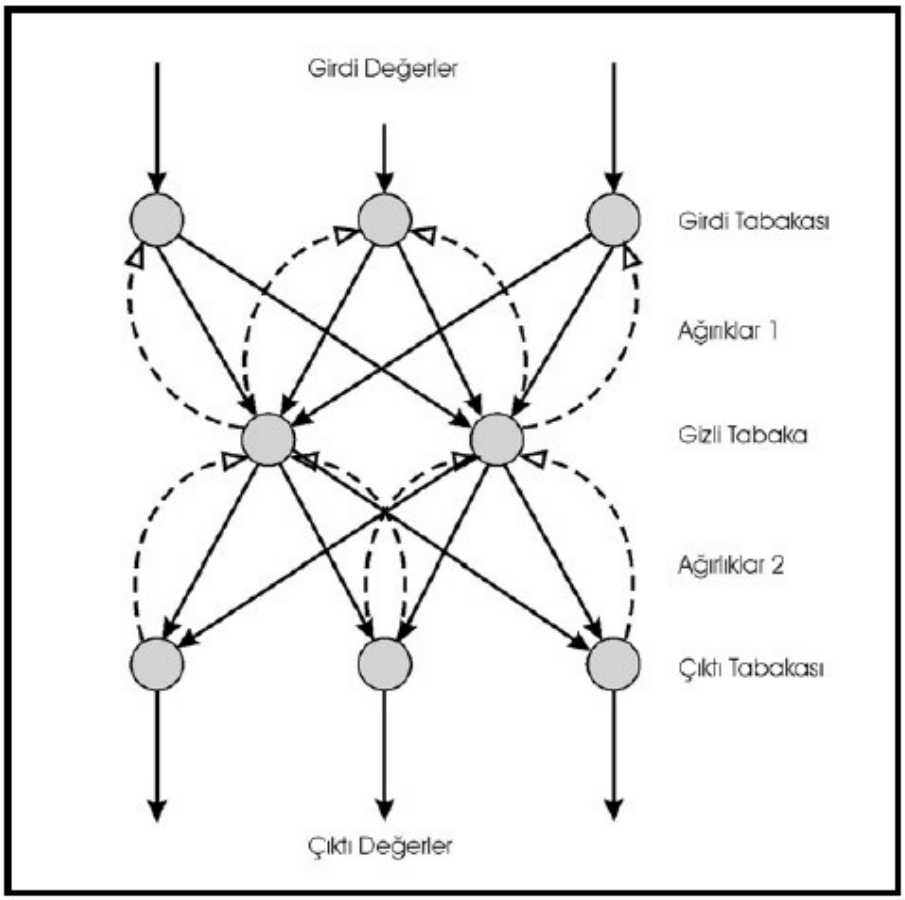
ÇKA ağı öğretmenli öğrenme stratejisini kullanır. Ağa, hem örnekler hem de örneklerden elde edilmesi gereken çıktılar verilmektedir. Ağ kendisine gösterilen örneklerden genellemeler yaparak problem uzayını temsil eden bir çözüm uzayı üretmektedir. Daha sonra gösterilen benzer örnekler için bu çözüm uzayı sonuçlar ve çözümler üretebilmektedir.

Bu çalışmada kullanılan ÇKA ağ yapısı ileri besleme şeklinde yapılandırılmıştır. İleri besleme sinir ağlarında, işlem elemanları arasındaki bağlantılar bir döngü oluşturmazlar ve bu ağlar giriş verisine genellikle hızlı bir şekilde karşılık üretirler. Şekil 2.7 de çok tabakalı ileri besleme ağ yapısı gösterilmiştir.



Şekil 2.7 : Çok katmanlı ileri beslemeli ağ

Bu tezde, bir Geriye Yayılım Yapay Sinir Ağı kullanılmaktadır. Geriye yayılım ağları, çok tabakalı algılayıcı (perceptron) ile aynı yapıya sahiptirler ve öğrenme yöntemi olarak geri yayılma algoritması kullanırlar (Şekil 2.8).



Şekil 2.8 : İleri beslemeli geriye yayılım ağlarının genel yapısı

3. MALZEME VE YÖNTEM

Bu çalışmada el yazısı karakterlerini tanıma, ileri beslemeli yapay sinir ağları ile gerçekleştirilmiştir. Öğrenme işlemi öğretmenli öğrenmedir (supervised learning). Öğrenme algoritması olarak geriye yayılım algoritması (backpropogation algorithm) ve Shashank' ın algoritması kullanılmıştır. Daha sonra bu iki algoritma performans olarak karşılaştırılmıştır.

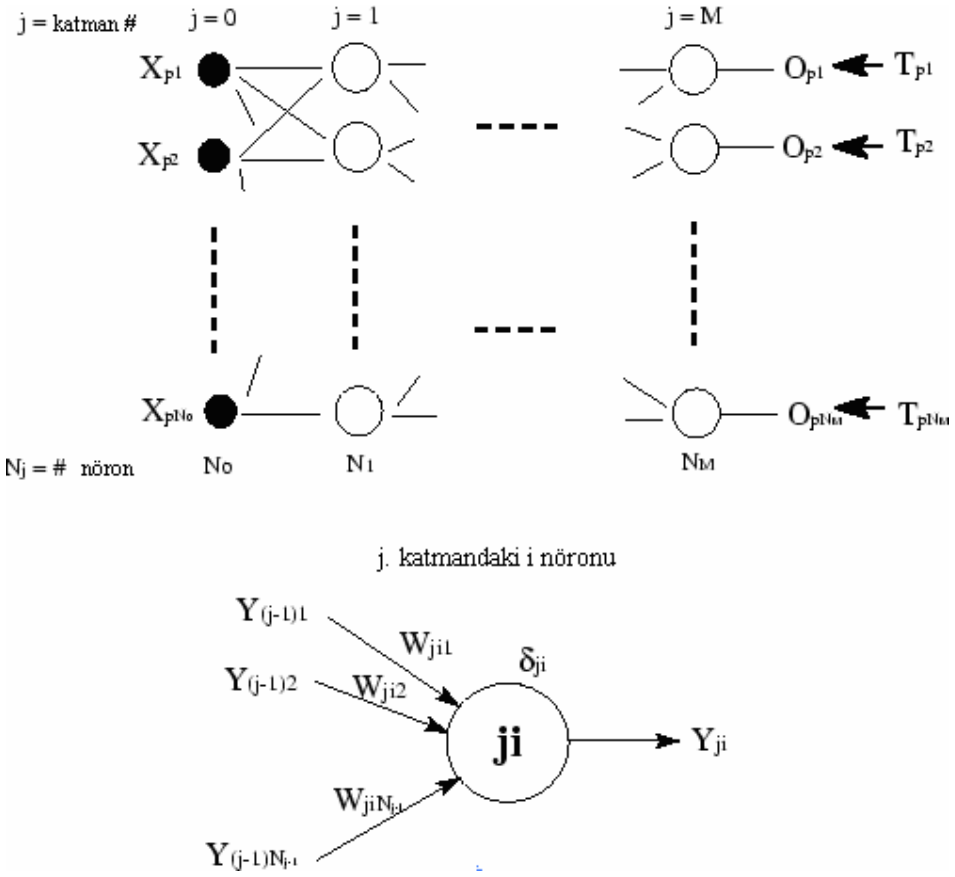
3.1. GERİYE YAYILIM (BACKPROPAGATION) ALGORİTMASI VE ÇALIŞMA PROSEDÜRÜ

Geriye yayılma algoritması, kolay anlaşılabilir olması ve uygulamadaki görüş açısı gibi başarılarından dolayı ağ eğitimi için en popüler algoritmalardan biridir. Bir geriye yayılım ağı modelinde giriş, gizli ve çıkış olmak üzere 3 katman bulunmakla birlikte, problemin özelliklerine göre gizli katman sayısını artırabilmek mümkündür.

Geriye yayılım ağları örneklerle öğrenirler. Ağa istenen örnekler verilir ve ağ çıkışı hesaplar. Hesaplanan çıkış istenen çıkışla karşılaştırılır ve hata hesaplanır. Bu algoritma; hataları geriye doğru çıkıştan girişe doğru azaltmaya çalışmasından dolayı geriye yayılım ismini almıştır. Geriye yayılım öğrenme kuralı ağ çıkışındaki mevcut hata düzeyine göre her bir tabakadaki ağırlıkları yeniden hesaplamak için kullanılmaktadır.

Geriye yayılım çok katmanlı ağlarda kullanılan delta kuralı için genelleştirilmiştir bir algoritmadır. Bu algoritma çok katlı ağlarda hesap işlerini öğrenmede kullanılabilmektedir. Geriye yayılım ağında hatalar, ileri besleme aktarım işlevinin türevi tarafından, ileri besleme mekanizması içinde kullanılan aynı bağlantılar aracılığıyla, geriye doğru yayılmaktadır. Öğrenme işlemi, bu ağda basit çift yönlü hafıza birleştirmeye dayanmaktadır.

Şekil 3.1 M katmanlı bir çok katmanlı geriye yayılım ağını (backpropagation multilayer network) göstermektedir. N_j, j. katmandaki nöron sayısını, p ise eğitim örnek seti p. örneğini temsil eder. X_{p1}, X_{p2}, …, X_{pN_0} girişleri, T_{p1}, T_{p2}, …, T_{pN_M} istenen çıkışları diğer deyişle hedef çıkışları, $O_{p1,}$ O_{p2}, …, O_{pN_M} ise ağın çıkışlarını göstermektedir. Y_{ji}, p örneği için j katmanındaki i. nöronun çıkışıdır. $W_{jik,}$ (j-1) katmanındaki k. nörondan j katmanındaki i. nörona giden bağlantı ağırlığıdır. δ_{ji} ise j katmanındaki i. nörona ait hatadır.



Şekil 3.1: Geriye yayılım sinir ağı

3.1.1. Geriye Yayılım Algoritması Hesaplamaları

Geriye yayılım algoritmasında çeşitli formülasyonlar kullanılır[13]. Bunlar :

Başlangıç ağırlıklarının belirlenmesi :
Başlangıç bağlantı ağırlıkları küçük aralıklarda rasgele sayılardan seçilir.

Giriş ve hedef vektörlerinin belirlenmesi :
p. örneğin giriş vektörü $X_p = (X_{p1}, X_{p2}, \ldots, X_{pN_0})$ ağa verilir. Verilen giriş için hedef çıkış vektörü $T_p = (T_{p1}, T_{p2}, \ldots, T_{pN_M})$ saptanır.

İlk girişlerin çıkış kabul edilmesi :
Girişler ilk katmandan geçirilerek 0 katmanındaki her i düğümü için şu eşitlik sağlanır :

$$Y_{0i} = X_{pi}. \tag{3.1}$$

Katmanlardaki nöron çıkışlarının hesaplanması :
Giriş katmanından çıkış katmanına kadar olan her j = 1,2, ..., M katmanındaki her i nöronu için sigmoid aktivasyon fonksiyonu da kullanılarak çıkış hesaplanır [14].

$$Y_{ji} = f\left(\sum_{k=1}^{N_{j-1}} Y_{(j-1)k} W_{jik}\right) \tag{3.2}$$

$$f(x) = \frac{1}{1 + exp(-x)}. \tag{3.3}$$

Ağın çıkış değerlerinin belirlenmesi :
Ağın çıkış değerleri bulunur. M katmanındaki her i düğümü için şu eşitlik sağlanır :

$$O_{pi} = Y_{Mi} \tag{3.4}$$

Gizli katmanlar ve çıkış katman nöronları için hata değerlerinin hesaplanması :
Son katmandan başlayarak ilk katmana doğru her j = M, M-1, ..., 1 katmanındaki i nöronu için δji hata değeri hesaplanır. Çıkış katmanı i nöronu için hata değeri :

$$\delta_{Mi} = Y_{Mi}(1 - Y_{Mi})(T_{pi} - Y_{Mi}) \tag{3.5}$$

Gizli katman nöronları için hata değeri :

$$\delta_{ji} = Y_{ji}(1 - Y_{ji}) \sum_{k=1}^{N_{j+1}} \delta_{(j+1)k} W_{(j+1)ki} \tag{3.6}$$

Gizli katman ve çıkış katmanı ağırlıklarının değiştirilmesi :
(i-1) katmanındaki k nöronundan, i katmanındaki her i nöronuna olan bağlantı ağırlıkları değiştirilir [15].

$$W_{jik}^{+} = W_{jik} + \beta \delta_{ji} Y_{ji} \tag{3.7}$$

β, 0 ile 1 değerleri arasında olup öğrenme oranını (learning rate) gösterir.

2. adımdan 6. adıma kadar olan işlemler her eğitim seti örneği p için tekrarlanır. Ve bu eğitim seti, ortalama karesel hata (Mean Square Error) değeri minimuma indirilene dek ağa sunulur. MSE, p. örneğin çıkış katmanı için aşağıdaki formülle ifade edilir [16].

$$E_p = \frac{1}{2} \sum_{j=1}^{N_M} (T_{pj} - O_{pj})^2 \tag{3.8}$$

Genelleştirilmiş delta kuralında (Generalized Delta Rule), j katmanındaki i nöronu hatası δji, ortalama karesel hata (Mean Square Error) değişiminin nöron toplam ürün değerine oranıdır.

$$\delta_{ji} = -\frac{\partial E_p}{\partial net_{ji}} \tag{3.9}$$

net_{ji} toplam ürün değerini gösterir.

Zincir kuralıyla E_p değerindeki değişimin ağırlık değişimine oranı elde edilebilir.

$$\begin{aligned}\frac{\partial E_p}{\partial W_{jik}} &= \frac{\partial E_p}{\partial net_{ji}}\frac{\partial net_{ji}}{\partial W_{jik}} \\ &= -\delta_{ji}\frac{\partial}{\partial W_{jik}}\left[Y_{(j-1)0}W_{(j-1)i0} + \ldots + Y_{(j-1)k}W_{(j-1)ik} + \ldots\right] \\ &= -\delta_{ji}\frac{\partial}{\partial W_{jik}}Y_{(j-1)k}W_{(j-1)ik} \\ &= -\delta_{ji}Y_{(j-1)k}\end{aligned} \tag{3.10}$$

Ağırlık değişimi aşağıdaki gibi yazılabilir.

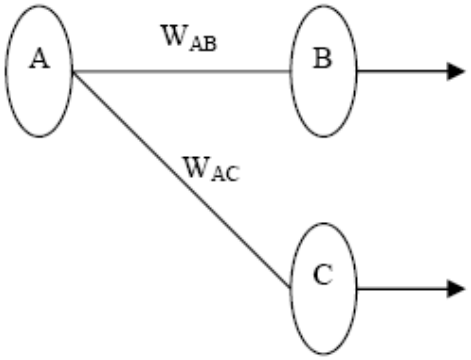
$$\Delta W_{jik} = \beta\delta_{ji}Y_{(j-1)k} \tag{3.11}$$

Burada β sabit olduğundan ağırlık değişimi şu şekilde yazılabilir.

$$W^{+}_{jik} = W_{jik} + \Delta W_{jik} \tag{3.12}$$

3.1.2. Geri Yayılım Algoritması Adımları

Şekil 3.2 de gizli katman, çıkış katmanı, ve çıkış katmanı ağırlıkları gösterilmiştir. Bu çalışmadaki algoritma adımları sembolik olarak bu şekil üzerinden gösterilecektir [17].



Şekil 3.2: Çıkış katmanı ağırlıkları gösterilen geri yayılımlı ağ

- 1. Adım :

  Girişler ağa verilerek çıkışlar hesaplanır. Başlangıç ağırlıkları rasgele seçildiğinden (Örn. [-0.5 , +0.5]), başlangıç çıkış değerleri de rasgele değerler olur.

- 2. Adım :

  B nöronunun hatası hesaplanır. Hata, istenen çıkışın elde edilen çıkıştan farkıdır. Diğer bir deyişle;

  $Hata_B = Çıkış_B (1 – Çıkış_B) (Hedef\_Çıkış_B – Çıkış_B)$

  Çıkış (1 - Çıkış) ifadesi sigmoid fonksiyon kullanıldığı için gereklidir.

- 3. Adım :

  Çıkış katmanı ağırlıkları değiştirtilir. W^+_{AB}, eğitilmiş ağırlık W_{AB} ise başlangıçtaki ağırlık olsun. Ağırlık güncelleme ifadesi aşağıdaki gibi olur.

  $W^+_{AB} = W_{AB} + (Hata_B * Çıkış_A)$

  Çıkış katmanındaki tüm bağlantı ağırlıkları bu şekilde tekrar düzenlenir.

- 4. Adım :

  Gizli katman nöronlarının hataları hesaplanır. Çıkış katmanındaki nöronların hatalarını hesaplamada kullanılan yöntemden farklı bir yöntem kullanılır. Çünkü bu katmanda hedef çıkış yoktur. Bundan dolayı çıkış katmanı hataları geriye iletilir [18]. Algoritma ismini bu adımdan dolayı almıştır. Çıkış katmanı hata değerleri geriye olan bağlantılar boyunca gizli katmana yayılır. Örneğin Şekil 3.2 deki A nöronunun hatasını elde etmek için çıkış katmanındaki B ve C nöronlarının hataları kullanılır.

  $Hata_A = Çıkış_A (1 – Çıkış_A) (Hata_B \ W_{AB} + Hata_C \ W_{AC})$

- 5. Adım :

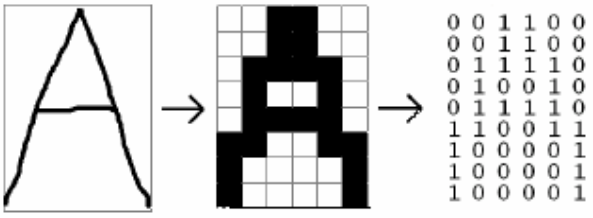
  Gizli katman hataları hesaplandıktan sonra, 3. adımdaki gibi gizli katman ağırlıkları değiştirilir. Bu işlemler ağ eğitilene kadar devam ettirilir.

3.1.3. Görüntü Dijitalleştirme (Image Digitization)

Örneklerin (giriş/çıkış çiftleri) ağa nasıl gösterileceği örneklerin belirlenmesi kadar önemlidir. Görüntüyü dijital hale getirme yapay sinir ağları için temel adımdır. Görüntü, bilgisayar sisteminin ve sinir ağının işleyebileceği hale getirilmelidir. Yapay sinir ağları yalnızca rakamlar ile çalışabilmektedir. Eğer problem uzayında sayısal (nümerik) olmayan faktörleri dikkate almak gerekiyor ise o zaman bunların rakamlar ile temsil edilebilmesi gerekmektedir. Bu dönüştürme çeşitli şekillerde olabilmekte ve bu da ağın performansını etkilemektedir. Hem giriş değerlerinin hem de beklenen çıkış değerlerinin

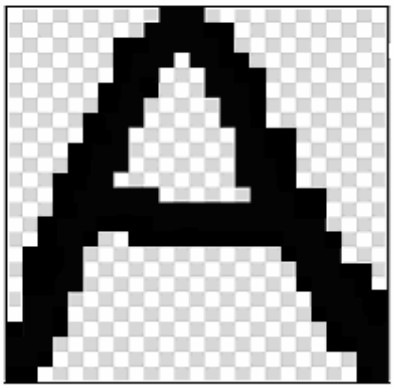
nümerik olarak gösterilmesi gerekmektedir. Dijital bir resim elektronik olarak bit ya da baytlarla ifade edilir. Bu dönüşümden ikili bir matris elde edilerek, resim orjinal boyutundan bağımsız hale getirilir. Böylelikle herhangi bir boyuttaki bir resim, önceden tanımlanmış sabit boyutlu bir ikili matris şeklinde ifade edilebilir.

Resmi rakamlarla ifade etmek için resmin bilgisayar ortamındaki gri tonları kullanılmaktadır. Bir resmin 256x256 pixelden oluşan bir çerçevede çekilmesi sonucu resmin her noktasının gri tonunu gösteren 0-256 arasında bir sayı vardır. Görüntü dijitalleştirmede giriş görüntüsü ikili bir pencere formatına dönüştürülür. Yani resim özel dönüşümlerden geçirilerek bu 0-256 arası sayılar, tonlarına göre 0 veya 1 olur. Bu ise siyah-beyaz resmi ifade eder. Beyaz renk pikselleri 0 ile, siyah renk pikselleri ise 1 ile işaretlenir. Şekil 3.3 de A harfi 6X8 = 48 hücre ile dijital hale getirilmiştir. Elde edilen matris 0 ve 1' lerden oluşur.



Şekil 3.3: Dijital "A" harfi

Bu çalışmada, karakterler daha büyük boyutlu alınmıştır. Çünkü boyut küçüldükçe bilgi kaybı oluşmakta, resimlerin bazı ince detayları kaybolmakta ve ağın görmesi mümkün olamamaktadır. Bundan dolayı karakterler için 25x25 boyutu seçilmiştir. Şekil 3.4 de eğitim seti örneklerinden bir "A" harfi, Şekil 3.5 de ise oluşturulan 25x25 boyutlu ikili A matrisi görülmektedir.



Şekil 3.4: Dijital hale getirilmiş bir “A” harfi

0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0
0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0
0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0
0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Şekil 3.5: İkili (binary) A matrisi

3.1.4. Çok Katmanlı Geriye Yayılım Ağının Çalışması

Yapılan el yazısı karakterlerini tanıma çalışmasında geriye yayılımlı öğrenme algoritmasının uygulanması şu şekilde ilerler :

- Örneklerin toplanması :

Ağa öğretilmek istenen örneklerin bulunması adımıdır. Ağın eğitilmesi için örnekler toplandığı gibi (eğitim seti) ağın test edilmesi için de örneklerin (test seti) toplanması gerekmektedir. Ağın eğitilmesi sırasında test seti ağa hiç gösterilmez. Eğitim setindeki örnekler tek tek gösterilerek ağın olayı öğrenmesi sağlanır. Ağ olayı öğrendikten sonra test setindeki örnekler gösterilerek ağın performansı ölçülür. Hiç görmediği örnekler karşısındaki başarısı ağın iyi öğrenip öğrenmediğini ortaya koymaktadır. Bu çalışmada ağa sunulan eğitim seti 1500 elemandan oluşmaktadır. Bu 1500 örnek el yazısı karakteri, Latin alfabesindeki A' dan Z' ye kadar olan tüm harfleri yaklaşık olarak eşit sayıda kapsamaktadır.

- Ağın topolojik yapısının belirlenmesi :

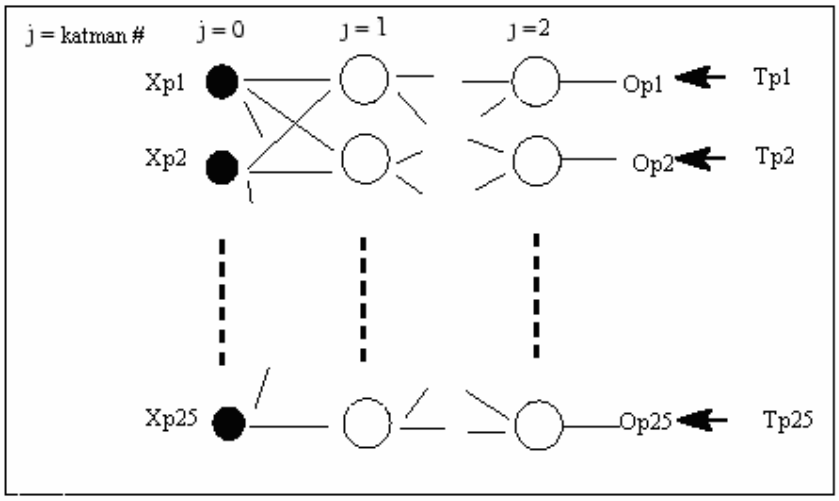
Öğrenilmesi istenen örnekler için oluşturulacak olan ağın topolojik yapısı belirlenir. Kaç tane giriş ünitesi, kaç tane ara katman, her ara katmanda kaç tane proses elemanı ve kaç tane çıkış elemanı olması gerektiği bu adımda belirlenmektedir. Ancak ÇKA modelinde herhangi bir problem için kaç tane ara katman ve her ara katmanda kaç tane proses elemanı kullanılması gerektiğini belirten bir yöntem şu ana kadar bulunmuş değildir. Ara katman sayısı ve proses elemanı sayıları ağın performansını yakından ilgilendirmektedirler.

Ağı oluşturmak için giriş katmanı, bir gizli katman ve çıkış katmanı kullanılmıştır. Giriş katmanı 625 nörondan oluşmaktadır (Şekil 3.6). Sebebi ise, giriş matrislerini boyutlarının 25X25 = 625 olmasıdır. Matrisin her elemanı bir nöron ile temsil edilmektedir.

Gizli katmanda ise başlangıçta 50 nöron ile başlanmıştır. Fakat bir problem herhangi bir ağ ile kabul edilebilir hata altında çözüm üretse bile, daha iyi bir ağ oluşturmak için farklı sayıdaki ara katman ve her ara katmanda farklı sayıda proses elemanları ile denemeler yapılmıştır. ÇKA ağlarında problemin çözümü için gerekli en iyi topolojiyi belirlemek mümkün olmadığından deneme yanılma yöntemi kullanılmakta, bazen daha eksik sayıda bazen de daha fazla sayıda proses elemanına

ihtiyaç duyulmaktadır. Bu nedenle ağ budanarak gizli katman nöron sayısı 35' e düşürüldüğünde ağ performansının daha iyi olduğu gözlenmiştir. Büyük bir ağdan başlayıp küçük bir ağa doğru eğitim sırasında sürekli ağı küçültmek ve proses elemanlarını teker teker ağdan çıkartmaya ağın budanması denmektedir.

Çıkış katmanı nöronları ise 26 tanedir. Her nöron takımı alfabedeki bir harfi göstermektedir [19]. Örneğin A harfi için çıkış katmanı nöronlarından sadece ilk nöron değeri 1' dir. Diğer nöron değerleri 0' dır. B nöronu içinse sadece ikinci nöron 1, diğerleri 0' dır. Tüm karakterler için bu şekilde bir tanımlama yapılmıştır (Tablo 3.1). Bu değerlerin resim veya karakterlerle bağlantısı yoktur. Ağ performansı açısından uygun bulunduğu için çıkışlar bu şekilde tasarlanmıştır.



Şekil 3.6 : Kullanılan yapay sinir ağı topolojisi

Tablo 3.1 : Tüm karakterler için beklenen çıkış nöron değerleri

Çıkış Katmanı Nöron Değerleri	Karakter
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	A
0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	B
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	D
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	E
0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	F
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	G
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	H
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	I
0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	J
0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	K
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0	L
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0	M
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0	N
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0	O
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0	P
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0	Q
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0	R
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0	S
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0	T
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0	U
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0	V
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0	W
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0	X
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0	Y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1	Z

- Öğrenme parametrelerinin belirlenmesi :

Ağın öğrenme katsayısı, proses elemanlarının toplama ve aktivasyon fonksiyonları, gibi parametreler bu adımda belirlenmektedir. Başlangıç değerleri kadar öğrenme katsayısının belirlenmesi de ağın öğrenme performansı ile yakından ilgilidir. Öğrenme katsayısı ağırlıkların değişim miktarını belirlemektedir. Bu çalışmada öğrenme katsayısı 0.2 olarak seçilmiştir. Bu sayı tamamen ilgili probleme bağlıdır.

- Ağırlıkların başlangıç değerlerinin atanması :

Proses elemanlarını birbirlerine bağlayan bağlantıların ağırlıklarının başlangıç değerlerinin atanması yapılır. Başlangıçta genellikle rasgele değerler atanır. Daha sonra ağ uygun değerleri öğrenme sırasında kendisi belirler. Başlangıç ağırlık değerleri W_{ji} [-0.5 , +0.5] arasında alınmıştır.

- Öğrenme setinden örneklerin seçilmesi ve ağa gösterilmesi :

Ağın öğrenmeye başlaması ve yukarıda anlatılan öğrenme kuralına uygun olarak ağırlıkları değiştirmesi için örnekler (giriş/çıkış değerleri) ağa teker teker ve belirli bir düzeneğe göre sıralı veya rasgele olarak gösterilir.

- Öğrenme sırasında ileri hesaplamaların yapılması :

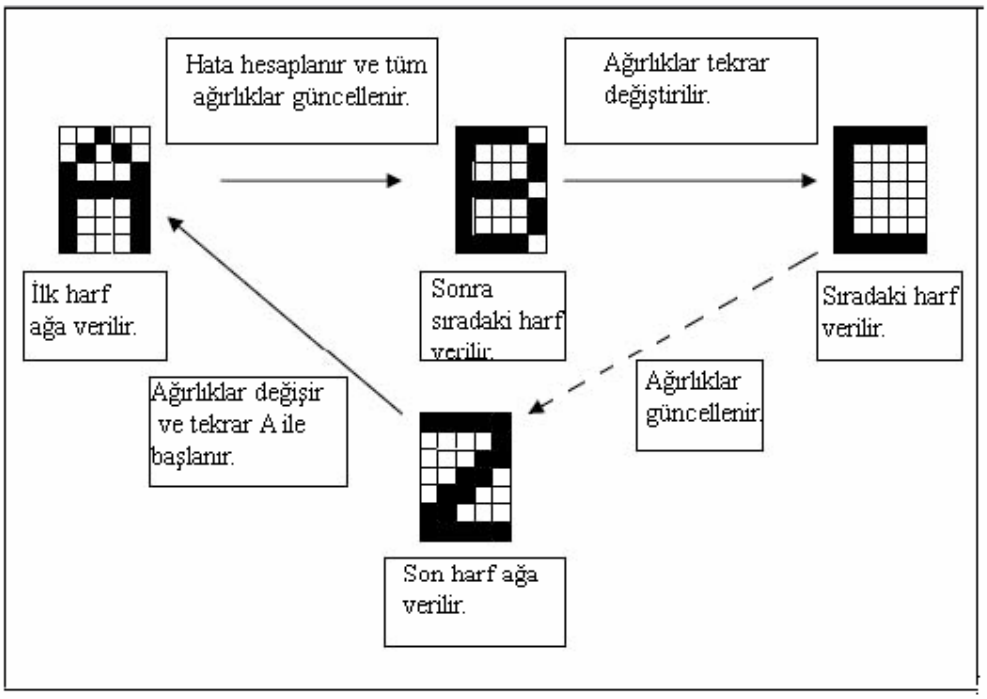
Sunulan giriş için ağın çıkış değerleri hesaplanır. Bu aşamada bilgi işleme, eğitim setindeki bir örneğin giriş katmanından ağa gösterilmesi ile başlar. Daha önce de belirtildiği gibi giriş katmanında herhangi bir bilgi işleme olmaz. Gelen girişler hiçbir değişiklik olmadan ara katmana gönderilir. Önce ara katmanın çıkışları hesaplanarak sigmoid fonksiyonundan geçirilir, çıkış katmanının net girişleri bulunduktan sonra çıkışları hesaplanıp tekrar sigmoid fonksiyonundan geçirilerek sonuç belirlenir. Çıkış katmanındaki çıkış değerleri elde edildiğinde ileri hesaplama tamamlanmış olur. Bu adımda kullanılan denklemler 3.1, 3.2, 3.3 ve 3.4 dür.

- Öğrenme sırasında geri hesaplamaların yapılması :

Bulunan çıktının beklenen çıktı ile karşılaştırılması yapılarak ağın ürettiği hata değerleri bu adımda hesaplanır. Amaç bu hatayı düşürmektir. O nedenle geriye hesaplamada bu hata ağın ağırlık değerlerine dağıtılarak bir sonraki iterasyonda hatanın azaltılması sağlanır. Ağırlık değerleri hatayı azaltacak şekilde tekrar düzenlenir. Bu adımda kullanılan denklemler ise 3.5, 3.6, 3.7 ve 3.8 dir.

3.1.5. Çok Katmanlı Geriye Yayılım Ağının Eğitilmesi

Çok katmanlı geriye yayılım ağının eğitilmesi yukarıdaki adımların tekrarlanmasıyla gerçeklenir. Ağ eğitilirken ağa geniş bir örnek seti sunulur. Ağı eğitmenin etkin yolu, önce ilk karakteri ağa vermek ve ağırlıkları değiştirmektir. Daha sonra ikinci ve üçüncü karakterler ağa sunularak tüm eğitim seti bitirilir. Eğitim seti tamamen ağa verilip hatalar geriye yayılıp ağırlıklar değiştirildikten sonra eğitim seti tekrar ağa sunulur (Şekil 3.7). Bu işlem, elde edilen çıktılar ile beklenen çıktılar arasındaki hatalar kabul edilir düzeye ininceye kadar devam eder.

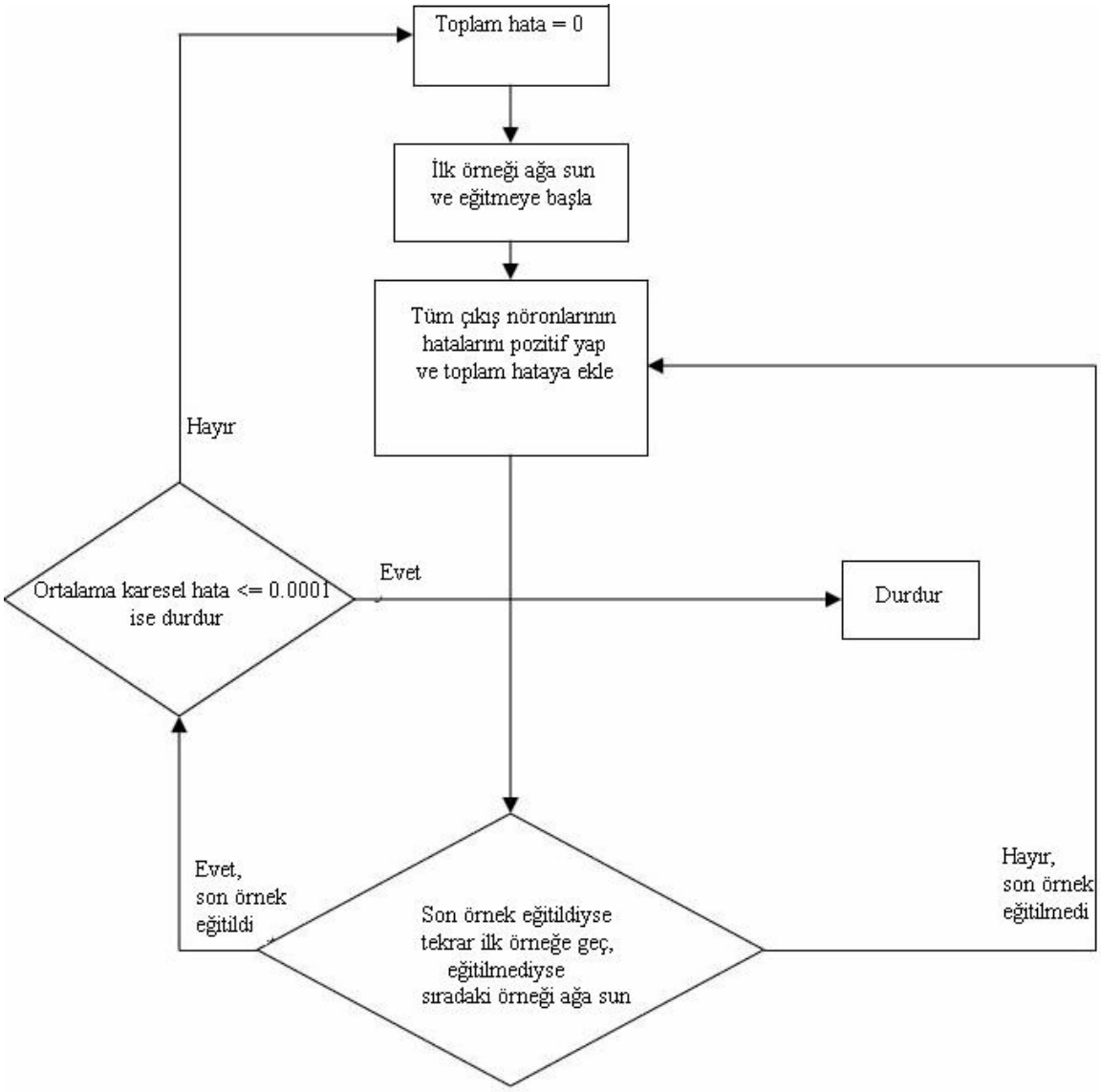


Şekil 3.7 : Çok katmanlı ağın eğitim akışı

3.1.6. Ağın Eğitiminin Durdurulması

Teorik olarak; ağ tüm karakterleri tanıdığında eğitim sonlandırılır. Fakat pratikte ağın öğrenmesi için bir durdurma kriterinin olması gerekmektedir. Bu ise genellikle üretilen hatanın belirli bir düzeyin altına düşmesi olarak alınmaktadır. Bu çalışmada eğitilen ağın ortalama karesel hata değeri 0,0001 olarak ölçülmüştür. Yani ortalama karesel hata 0.0001 değerine düşene kadar eğitim seti örnekleri ağa verilmiştir. Bu aşamadan sonra ağ örnekleri tanıyacak duruma gelmiştir. Çalışmadaki adımlar Şekil 3.8 deki akış şemasında gösterilmiştir [20].

Toplam hata, tüm örneklerin tüm çıkış nöron hatalarının toplamıdır. Ortalama karesel hata ise örnek başına düşen ortalama hatadır (toplam hata / örnek sayısı). Nöron hatalarını üstüste toplarken pozitife çevirmek gerekir. Çünkü hatanın -0.5 veya 0.5 olması, gereken değerden sapması açısından farklı değildir. Ayrıca hata değerleri toplam hata değerini düşürmemelidir. Bu nedenle negatif hatalar pozitife çevrilir (Denklem 3.8).



Şekil 3.8 : Algoritmanın akış şeması

3.1.7 Ağın Test Edilmesi

Ağ bağlantı ağırlıkları son halini eğitimden sonra alır. Ağın eğitimi tamamlandıktan sonra test setindeki tüm veriler ağa sunulur. Tablo 3.1 de tüm karakterler için olması gereken nöron çıkışları listelenmiştir. Giriş örneğinin çıkışları alınır. Çıkış nöron değerleri her karaktere karşılık gelen çıkış kümesine yakın olmalıdır. Örneğin A karakteri için beklenen çıkış 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 olduğu

için sadece ilk nöron çıkışı 1 veya 1' e yakın olmalıdır. Diğer nöron çıkışlarının ise 0 veya 0' a yakın olması gerekir.

Ağın performans ölçümü, öğrenme yeteneğinin ölçülmesidir. Ağın, test setindeki örnekler karşısında ürettiği doğru cevaplar oranı ile ölçülür.

$$P = \frac{D}{T} \times 100 \tag{3.13}$$

Burada D test setinden doğru olarak cevaplandırılan örnek sayısını, T test setinde bulunan toplam örnek sayısını, P ise performans yüzdesini göstermektedir.

3.2. SHASHANK' IN ALGORITMASI VE ÇALIŞMA PROSEDÜRÜ

Shashank, el yazısı karakterlerini tanıma sistemi için geriye yayılım (backpropagation) gibi öğrenme algoritmalarına alternatif olarak yeni bir öğrenme algoritması ve test etme tekniği geliştirmiştir.

3.2.1. Shashank Ağının Eğitilmesi

Burada geriye yayılım algoritmasından farklı olarak, sürekli güncellenen tek bir ağırlık matrisi yerine, tüm karakterler için ayrı bir ağırlık matrisi oluşur. Eğitim aşamasındaki örnek girişi olan M matrisi aşağıdaki şekilde tanımlanır :

$$M(i,j) = \begin{cases} 1 & \text{Eğer I (i,j)} = 1 \\ -1 & \text{Eğer I (i,j)} = 0 \end{cases} \tag{3.14}$$

Burada ixj boyutlu I (i,j) matrisi eğitim setindeki karakter matrisidir. Kullanılan boyut 25x25' tir. M (i,j) matrisi ise ağa verilen giriş matrisidir.

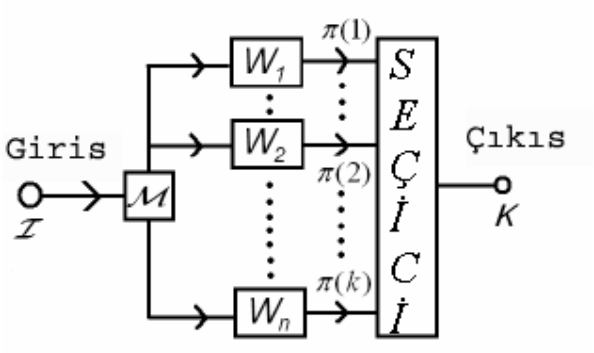
Aynı karakterin eğitim setindeki farklı örnekleri birer birer ağa öğretilir. Her karakterin ağırlık matrisi ayrıdır. k. karakter için $W_k(i,j)$ matrisi başlangıçta 0' lardan oluşur. Sonrasında ise aynı karakterlerin giriş matrisleriyle toplanarak güncellenir.

```
i = 1,x
 {
     j = 1,y
        {
            Wk ( i,j ) = Wk ( i,j ) + M ( i,j )                    (3.15)
        }
 }
```

Bu işlem tüm k. karakterler için tekrarlanarak $W_k(i,j)$ matrisi son haline getirilir. Eğitim seti örneklerinden A harfi 50 taneyse, $W_A(i,j)$ 50 defa güncellenir. Tüm karakterler için toplam 26 tane ağırlık matrisi oluşturulduğunda test aşamasına geçilir.

3.2.2. Shashank Ağının Test Edilmesi

Test aşamasında giriş matrisi I (i,j) karakter matrisidir. M bloğu, her k için W_k ağırlık bloklarına M giriş matrisini üretir. Ağırlık blokları 26 tanedir (Şekil 3.9).



Şekil 3.9 : Shanshank karakter tanıma sistemi

Shashank' ın tekniğinde adaylık skoru, ideal ağırlık modeli skoru ve tanıma bölüm değeri gibi üç değer tanımlanmıştır [21]. Bu çalışmada da tanıma işlemi bu değerlere göre yapılmıştır.

3.2.2.1. Adaylık Skoru (Candidate Score) (ψ)

Adaylık skoru değeri; I (i,j) karakter matrisi elemanlarının, öğrenilen k. karakterinin W_k (i,j) ağırlık matrisi elemanlarıyla çarpımları toplamıdır. Burada I (i,j), eğitim aşamasından farklı olarak, 0 ve 1' lerden oluşur.

$$\psi(k) = \sum_{i=1}^{x} \sum_{j=1}^{y} W_k(i,j) * I(i,j) \qquad (3.16)$$

3.2.2.2. İdeal Ağırlık Modeli Skoru (Ideal Weight – Model Score)(μ)

İdeal ağırlık modeli skoru, öğretilen örneğin ağırlık matrisindeki pozitif değerlerin toplamıdır. Bu skor için aşağıdaki yöntem kullanılır. μ(k) başlangıç değeri 0' dır.

```
i = 1,x
 {
     j = 1,y
        {
              Eğer W_k (i,j) > 0
              μ(k) = μ(k) + W_k ( i,j )                    (3.17)
        }
 }
```

3.2.2.3 Tanıma Bölüm Değeri (Recognition Quotient)

Bu değer bir test örneğinin eğtim setindeki hangi karakterle eşleşebileceğinin ölçümüdür.

$$Q(k) = \frac{\psi(k)}{\mu(k)} \tag{3.18}$$

Q değeri tüm karakterler için bulunur. Test örneği, en büyük Q değerini bulduran karakterin sınıfındandır. Eğer sistem yetersizse Q değeri kabul edilemeyecek bir değer olarak (Q<0.5) karşımıza çıkar. Böyle bir durumla karşılaşıldığında eğitim seti değiştirilerek sistem iyileştirilmiştir.

4. BULGULAR

Yapılan testler ile geriye yayılım ağının performansını etkileyen birçok parametre tespit edilmiştir. Parametrelere çeşitli değerler verilerek her durum için hata düşüşü gözlemlenmiştir. Elde edilen grafiklere göre performansı yükseltecek değerler belirlenmiştir.

4.1. GERİYE YAYILIM AĞININ PERFORMANSINI ETKİLEYEN PARAMETRELER

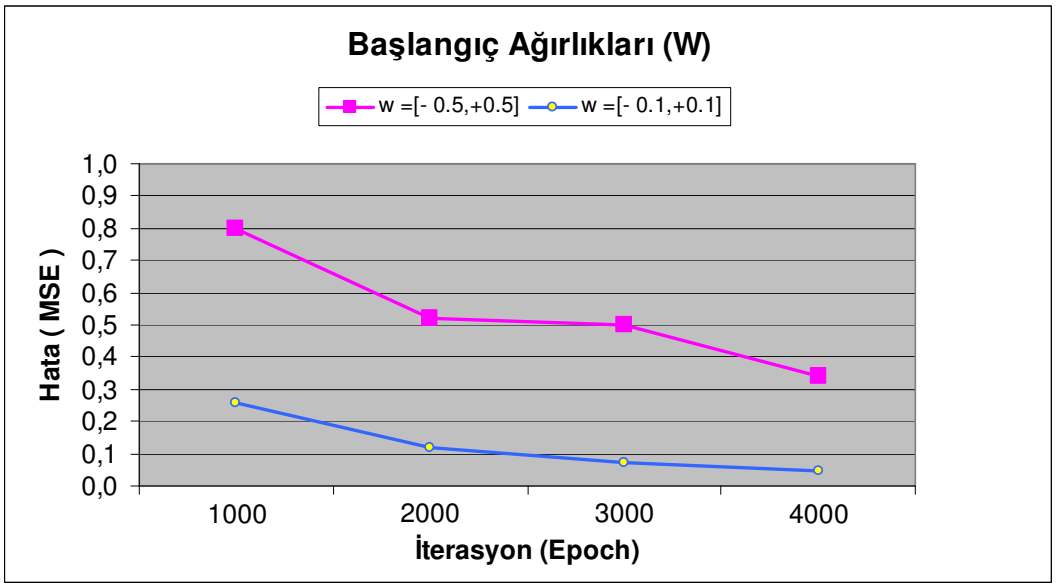
Değiştirildiklerinde ağ performansını etkileyen parametreler tespit edilmiştir. Bunlar :

- Başlangıç değerlerinin atanması
- Ara katmanların ve her katmandaki proses elemanlarının sayısının belirlenmesi
- Öğrenme katsayısının belirlenmesi
- Durdurma kriterinin belirlenmesi
- Örneklerin seçilmesi
- Örneklerin ağa sunulması

4.1.1. Başlangıç Değerlerinin Atanması

ÇKA ağının proses elemanlarını birbirine bağlayan bağlantı ağırlıklarının başlangıç değerlerinin atanması da ağın performansı ile yakından ilgilidir. Genel olarak ağırlıklar belirli aralıklarda atanmıştır. Bu değerlerin atanması için henüz belirlenmiş standart bir yöntem yoktur. Ağırlıkların başlangıç değerlerinin rasgele atanmaları istenmektedir. Bu aralık eğer büyük tutulursa ağın yerel çözümler arasında sürekli dolaştığı küçük olması durumunda ise öğrenmenin geç gerçekleştiği görülmüştür. Bu çalışmada ağırlıklar küçük tutulduğunda ağın ortalama karesel hatasının (MSE) sıfıra yaklaşmasının güç olduğu ve zaman aldığı gözlenmiştir. Bundan dolayı ağırlıklar [-0.5, +0.5] aralığında seçilmiştir. Bu değerler verildiğinde başlangıçta hata değerlerinin yüksek olduğu

sonraki adımlarda ise hızlı bir şekilde düştüğü görülmüştür. Şekil 4.1 de, bağlantı ağırlıkları [-0.5, +0.5] aralığında ve [-0.1, +0.1] iken iterasyon (etoch) arttıkça hatanın düşme oranları verilmiştir.



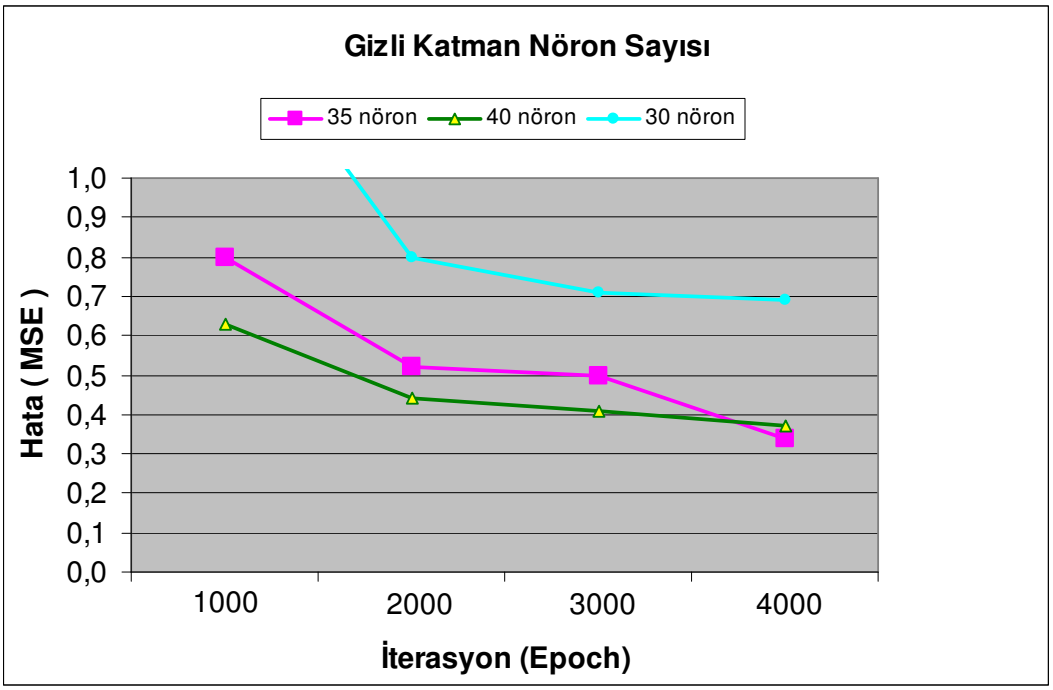
Şekil 4.1 : Farklı başlangıç ağırlık değerlerinde hata – iterasyon değişim grafiği

4.1.2 Ara Katmanların Ve Her Katmandaki Proses Elemanlarının Sayısının Belirlenmesi

ÇKA modelinde herhangi bir problem için kaç tane ara katman ve her ara katmanda kaç tane proses elemanı kullanılması gerektiğini belirten bir yöntem şu ana kadar bulunmuş değildir. Ara katman sayısı ve proses elemanı sayıları da ağın performansını yakından ilgilendirmektedirler. O nedenle bu çalışmada oluşturulan ağ ile kabul edilebilir hata altında çözüm üretse bile performansı daha iyi bir ağın oluşabilme ihtimali için farklı sayıdaki ara katman ve her ara katmanda farklı sayıda proses elemanları ile denemeler yapılmıştır. Böylece performansı daha yüksek bir ağ bulmak mümkün olmuştur. Başlangıçta bir ağ oluşturulup zaman içinde yapılan testler ışığında ağ büyütülerek veya küçültülerek istenene ulaşılmıştır.

Ağ gizli katmanı önce 30 nöron (proses elemanı) ile başlatılmıştır. Ağ giderek büyütülüp performansı gözlenmiştir. Başlangıçta 30 ara katman elemanı ile ağın eğitimine başlanmış ve ağın öğrenme hatası belirli bir değere kadar düşürülmüştür.

Yalnız bu öğrenmede iterasyonlar çabuk ilerlemekle birlikte hata düşme oranı küçük olduğundan tercih edilmemiştir. Daha sonra eleman sayısı 40'a yükseltilmiştir. Bu öğrenmede ise hata düşme oranı başlangıçta büyük olmakla birlikte sonraki adımlarda hata istenmeyen bir değerde sabit ilerlemeye başlamıştır. Ayrıca iterasyon içi işlemler çok yavaş gerçekleşmiş, ağ eğitiminde zorlanılmıştır (Şekil 4.2). Bundan dolayı ağ, gizli katman 35 nöron ile eğitilmiştir. Bu durumda hatanın iterasyon sayısına bağlı olarak düzenli şekilde düştüğü ve istenen hatada sabit ilerlediği görülmüştür. Ara katman (gizli katman) sayısı bir olarak seçilmiştir. Yani ağ üç katmandan oluşturulmuştur. Bu, yapay sinir ağlarında genel olarak kullanılan yapıdır. Aksi taktirde eğitim maliyeti yüksek olmaktadır.

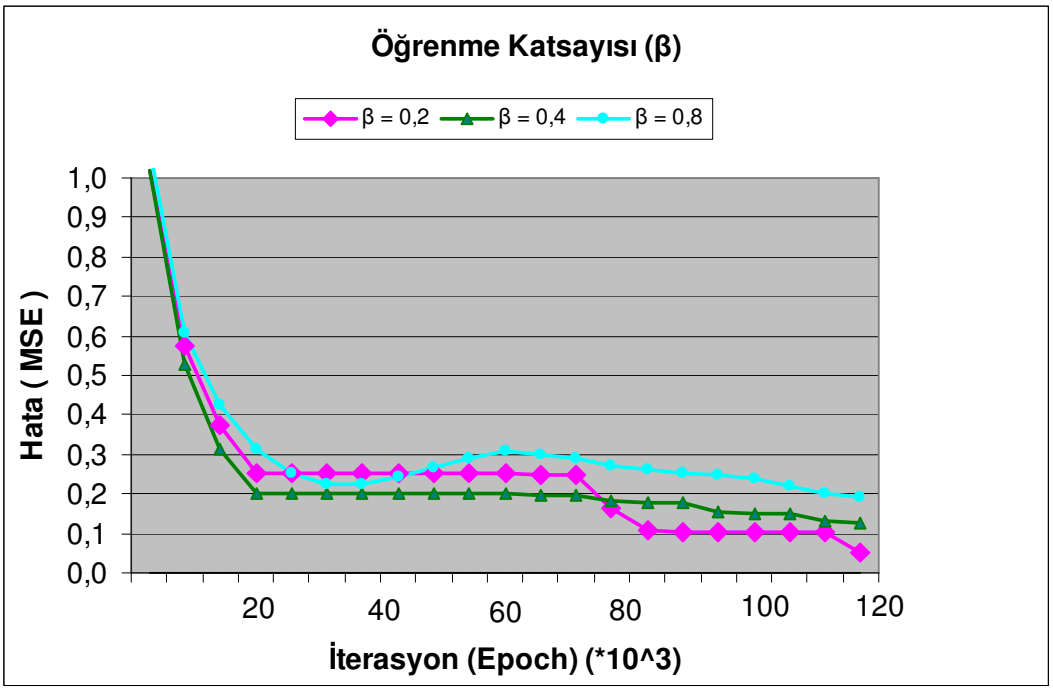


Şekil 4.2 : Farklı gizli katman nöron sayılarında hata – iterasyon değişim grafiği

4.1.3. Öğrenme Katsayısının Belirlenmesi

Öğrenme katsayısının belirlenmesi de ağın öğrenme performansı ile yakından ilgilidir. Öğrenme katsayısı ağırlıkların değişim miktarını belirlemektedir. Eğer büyük değerler seçilirse o zaman yerel çözümler arasında ağın dolaşması ve osilasyon yaşaması söz konusu olmaktadır. Küçük değerler şeçilmesi ise iterasyon geçiş zamanını artırmaktadır.

Şekil 4.3 de, öğrenme katsayısı 0.2, 0.4 ve 0.8 iken iterasyon (etoch) arttıkça hatanın düşme oranları verilmiştir.



Şekil 4.3 : Farklı öğrenme katsayılarında hata – iterasyon değişim grafiği

Öğrenme katsayısı 0.8 iken, hata değerlerinde sıkça salınım oluşmuştur. Ayrıca hatanın istenen değere düşmesi için 120.000 iterasyondan daha fazlaya ihtiyaç vardır. Bu da eğitim için oldukça fazla zaman harcanması demektir. Bundan dolayı tercih edilmemiştir.

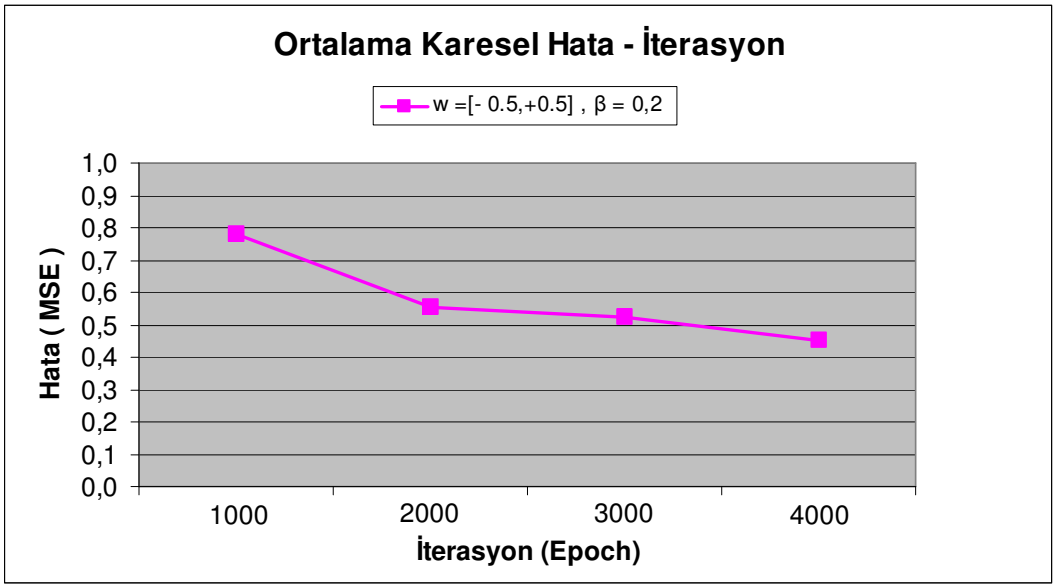
Öğrenme katsayısı 0.4 iken, osilasyon oluşmamış, fakat hatanın sıfıra yaklaşmasının çok da basit olmadığı görülmüştür.

Öğrenme katsayısı 0.2 iken ise, hata belli adımda istenen değere yaklaşmıştır. Zaman maliyeti fazla değildir. Bundan dolayı bu çalışmada öğrenme katsayısı 0.2 olarak seçilmiş ve ağ eğitilmiştir.

4.1.4. Durdurma Kriterinin Belirlenmesi

Çalışmada kullanılan ağın eğitilmesinin belirli bir aşamadan sonra durdurulması gerekir. Çünkü ağın çözüm üretecek ağırlıklarını bulduktan sonra eğitime devam edildiğinde ağırlıkların ve sistemin istenmeyen şekilde değiştiği görülmüştür. Bu şekilde, en iyi çözüm üreten bir ağ tekrar daha performansı düşük ağlara veya öğrenemeyen ağlara dönüşmektedir. O nedenle ağın eğitiminin ne zaman durdurulması gerektiği konusunda karar verilmesi de ağın başarısını ve performansını etkilemektedir. Ağ eğitildikçe hatası iterasyon sayısına bağlı olarak azalmıştır (Şekil 4.4). Bu çalışmada iki türlü durdurma kriteri kullanılmıştır.

- Hatanın belirli bir değerin altına düşmesi halinde eğitimi durdurma :
  Bu durumda hatanın bütün eğitim seti için kabul edilebilir bir değerin altına düşmesi kriter olarak alınır. Bazı durumlarda eğitim seti küçük olduğunda hata değerleri çok küçük seviyelere düşmekte büyük olduğunda ise kabul edilemez düzeyde kalmaktadır. Bu durumda eğitime devam edilmiştir. Bu çalışmada bütün örneklerin hatalarının ortalama değerinin belirli bir değerin altına düşmesi sağlanmıştır. Yani oratalama karesel hata minimize edilerek 0.0001' e kadar düşürülmüş ve ağ eğitilerek son haline getirilmiştir.
- Ağın belirli bir iterasyon sayısını tamamlaması sonucu eğtimi durdurma :
  Bu durumda hatanın hangi değere kadar düşeceğini belirlemek yerine belli iterasyon sayısı ile ağın kaç adımda eğitileceği belirlenir. Ağ eğitim performansı ile ilgili eğitim denemeleri yapmak ve hata grafiklerini pratik yoldan, zaman tüketimi fazla olmadan incelemek için ağın eğitileceği iterasyon sayısı saptanır. Özellikle hatanın hangi değere kadar düşebileceğinin kestirilemediği (yani kabul edilebilir hatanın belirlenemediği) durumlarda bu durdurma kriteri uygulanmıştır.

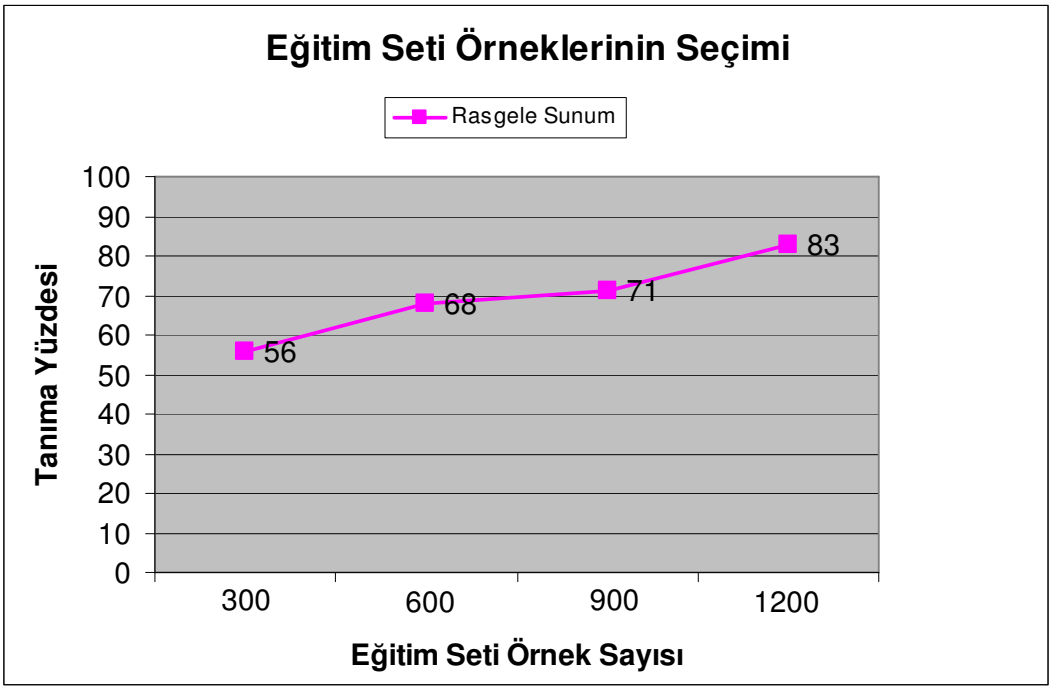


Şekil 4.4 : İterasyon sayısı arttığında MSE değerinin değişim grafiği

4.1.5. Örneklerin Seçilmesi

Örneklerin seçilmesi ağın performansını yakından ilgilendiren bir konudur. Çünkü ağ, bu örnekleri dikkate alarak ağırlıklarını değiştirmektedir. Seçilen örneklerin problem uzayını temsil edebilecek nitelikte olması çok önemlidir. Çünkü ağa ne gösterilirse ağ ancak o konularda yorumlar yapabilir ve ancak o konuda görmediği örneklere çözümler üretebilir.

Eğitim seti sadece uç veya çok düzgün örneklerden (el yazısı karakterleri) oluşmamıştır. Ağ oluşturulurken problem uzayının her bölgesinden ve uzayı temsil eden örnekler seçilmiştir. Ve bütün örnekler belirlenip eğitim ve test seti olarak ikiye ayrılmıştır. Ağ test setine başarılı sonuçlar üretmediğinde seçilen örnekler tekrar gözden geçirilmiştir. Ağ test setindeki örnekleri öğrenirken görmemektedir. Ağın tanımada başarısız olduğu örnekler belirlenip toplanarak eğitim setine katılmış ve ağ yeniden eğitilerek performansı artırılmıştır. Böylece, zaman içinde problemi çözen eğitim setine ve ağa kavuşmak mümkün olmuştur. Ayrıca eğitim seti örnekleri arttıkça ve çeşitlendikçe ağ performansının ve tanıma yüzdesinin de arttığı gözlenmiştir (Şekil 4.5).



Şekil 4.5 : Farklı eğitim seti örnek sayılarında hata – iterasyon değişim grafiği

4.1.6. Örneklerin Ağa Sunulması

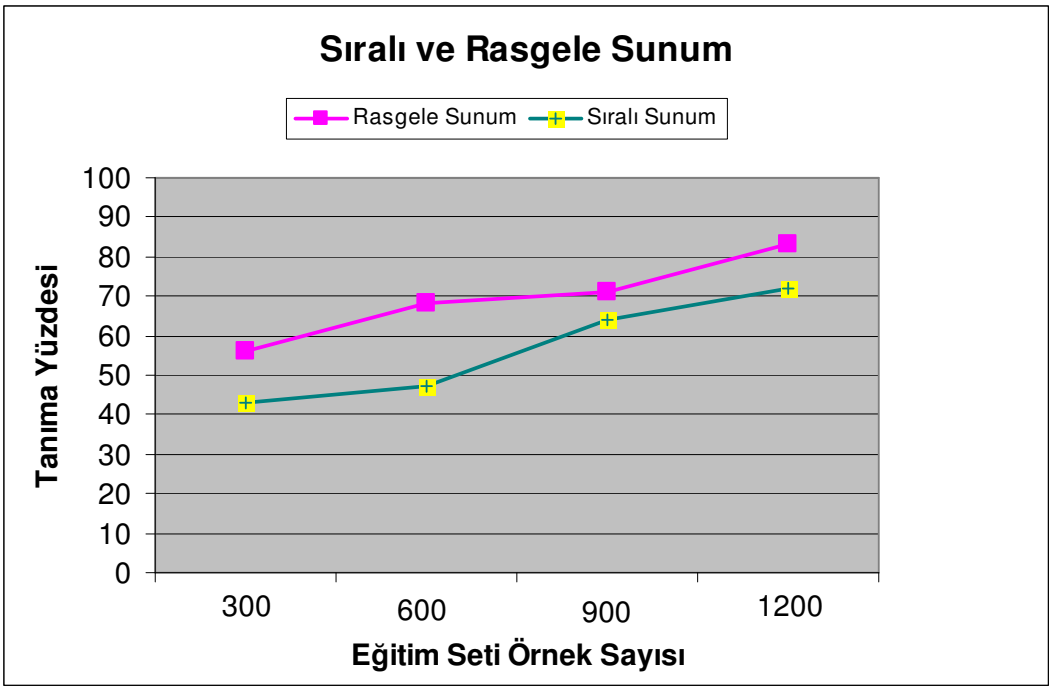
Örneklerin ağa sunulma şekli de öğrenme performansını etkilemektedir. Bu çalışmada örnekler ağa önce sıralı, ardından da rasgele olarak sunulmuştur.

- Sıralı Sunum
- Rasgele Sunum

Sıralı sunumda önce örnek setindeki ilk karakter tipindeki tüm örnekler sırayla ağa sunulmuş, sonraki iterasyonlarda ise ikinci, üçüncü ve sırasıyla en sonuncu karakter tipindeki örneklerin tamamı ağa sunulmuştur. Sonra tekrar başa dönülüp örnek setindeki örnekler aynı sıra ile ağa tekrar sunularak bu işlem öğrenme sağlanıncaya kadar devam ettirilmiştir. Bu tür bir sunuşta örnek setindeki bütün örneklerin ağa gösterilme şansları eşittir.

Rasgele sunumda ise karakter örnekleri eğitim seti içinden rasgele seçilmiş, rasgele seçilen örnek eğitim seti içine tekrar atılmamıştır. Kalanlar arasından rasgele tekrar yeni

örnek seçilerek ağa sunulmuştur. Bütün örnekler ağa gösterilince, eğitim seti tekrar rasgele seçilerek ağa gösterilmiştir. Bir defa gösterilen örnek bütün set ağa gösterilinceye kadar bekler. Öğrenme sağlanıncaya kadar bu işlem aynı şekilde tekrar eder. Örneklerin ağa gösterilme şansları bu durumda da eşittir. Böylece eğitim seti kümesi değişmemiş olur. Sıralı sunuma göre performansı daha yüksektir (Şekil 4.6). Çünkü aynı karakter örnekleri ardarda ağa sunulduğunda ağ o karakteri iyi bir şekilde öğrenmekte fakat diğer karaktere geçildiğinde, önceki karakteri unutabilmektedir. Bundan dolayı karakter örneklerini karışık ve rasgele olarak ağa sunmak gerekir. Sıralı sunumda ağ performansı % 72'de kalırken, rasgele sunumda bu %83'e çıkmıştır.



Şekil 4.6 : Ağa eğitim seti farklı sunulduğunda hata – iterasyon değişim grafiği

4.2. GERİYE YAYILIM AĞI PERFORMANS İNCELEMESİ

4.2.1. Optimum Parametreleri Bulma

Elde edilen grafiksel sonuçlara göre parametre değerleri değiştirilerek ağ performansı iyileştirilmiştir. Parametreler değiştirilirken sistemin eğitim süresi – performans değişimi göz önünde tutulmuştur. Bu parametrelere aşağıda değinilmiştir.

- Başlangıç Değerleri :
  Ağırlıkların başlangıç değerleri öncelikle rasgele atanmıştır. Yapılan deneme ve testlerden faydalanılarak başlangıç ağırlık değerleri [-0.5, +0.5] aralığında rasgele seçilmiştir.
- Ara Katmanların ve Her Katmandaki Proses Elemanlarının Sayısı :
  Oluşturulan ilk ağ yapısı ile kabul edilebilir hata değerlerine yaklaşılsa bile, ağ performansını en iyi hale getirebilmek için farklı ağ yapıları denenmiştir. Ağa tek gizli katmanın yeterli geldiği görülmüştür. Bu katmandaki nöron sayısı 35 olarak seçilmiştir. Daha iyi sonuçlar veren nöron sayısı tespit edilmesine rağman, sadece performansı yükseltmek değil kaynakları da ölçülü kullanmak adına nöron sayısı yükseltilmemiştir. Giriş katmanında ise 625 adet proses elemanı (nöron) kullanılmıştır. Bunun nedeni her karakter resmi boyutunun 25x25 olmasıdır. Kare resimlerde sistem daha iyi sonuç vermiştir. Resmi fazla küçültmek veri kaybına ve tanıma oranının azalmasına, fazla büyütmek de ağ yapısının karmaşıklaşmasına, eğitimin zorlaşmasına ve zaman kaybına yol açmıştır.
- Öğrenme Katsayısı (β) :
  Öğrenme katsayısı olarak genelde [0.2, 1] arası değerler kullanılır. Optimum sonuç veren öğrenme katsayısı için ağ eğitim testleri yapılmıştır. $\beta = 0.8$ gibi büyük bir değer verildiğinde yerel çözümler arasında ağın dolaştığı ve osilasyon yaşadığı görülmüştür. Bu durumda ayrıca hata değerinin belli bir değerin altına düşmesi için çok fazla iterasyonla ağı eğitmek gerekir. Bu da zaman açısından oldukça maliyetlidir. $\beta = 0.2$ gibi küçük bir değer verildiğinde ise iterasyon işlem zamanını artmaktadır. Yani tüm örnek setinin her eğitim adımından diğer adıma geçiş yavaşlamaktadır. Fakat bu zaman kaybı, fazla iterasyon olduğunda

meydana gelecek zaman kaybından çok daha azdır. Bundan dolayı bu çalışmada öğrenme katsayısı 0.2 olarak seçilmiş ve ağ eğitilmiştir.

- Durdurma Kriteri :

  Durdurma kriteri olarak hatanın belirli bir değerin altına düşmesi kriteri kullanılmıştır. Hata belli değerin altına düştüğünde eğitim tamamlanmış sayılarak durdurulmuştur. Ağ eğitildikçe hatası iterasyon sayısına bağlı olarak azalır. Bu çalışmada hata 0.0001' e kadar indirilmiştir. Böylece ağ performansı iyiye yaklaştırılmıştır. Kabul edilebilir hata oranı %17' ye düşürülmüştür.

- Eğitim Seti :

  Ağ eğitimi için seçilen örnekler problem uzayını temsil edebilecek niteliktedir. Eğitim seti düzgün ve gürültülü örneklerden oluşmaktadır. Bazen bütün örnekleri ağın öğrenmesi mümkün değildir. Böyle bir durumda bu örnekler eğitim setinden çıkartılmış veya onların öğrenememesi kabul edilmiştir. Yani bu çalışma için öğrenme performansı %99-100 olmaz da %83-85 gibi bir düzeyde kalır. Bu kabul edilebilir olduğundan sorun olmayabilir. Kabul edilebilir hatanın miktarı ise problemden probleme değişebilir. Bazı örnekler için %2 bile kabul edilemez bir hata oranı (özellikle insan hayatını ilgilendiren konularda) olabilir.

- Örneklerin Ağa Sunulması :

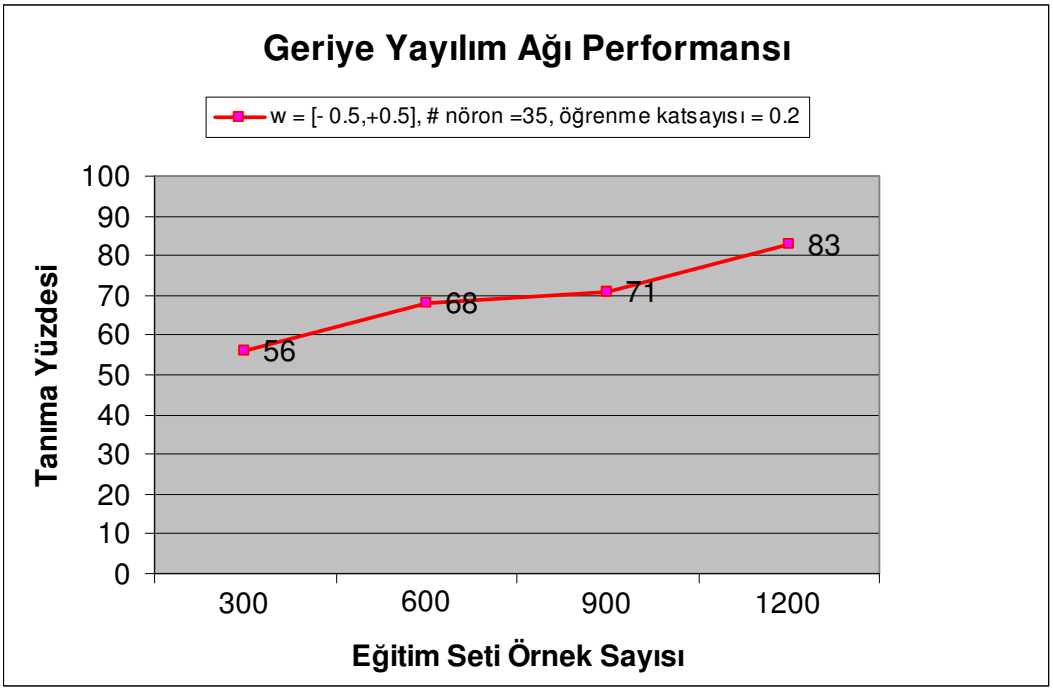
  Eğitim setindeki örnekler ağa sıralı veya rasgele sunulabilirler. Sıralı durumda bir karakterin tüm örnekleri bitmeden diğer karaktere geçilmez. Bu sunum şeklinin herhangi bir avantajı görülmemiştir. Çünkü aynı karakter örnekleri ardarda ağa sunulduğunda ağ o karakteri iyi bir şekilde öğrenmekte fakat diğer karaktere geçildiğinde, önceki karakteri unutma durumu meydana gelmiştir. Bundan dolayı ağ eğitimi için rasgele sunum seçilmiştir. Yani karakter örnekleri karışık olarak ağa verilmiş, öğrenme daha sağlıklı ve doğru hale getirilmiştir.

4.2.2. Ağ Performansı

Geriye yayılım öğrenme algortiması ve uyarlanan parametreler ile ağın karakter tanıma oranı %83' ü bulmuştur (Şekil 4.7). Ağ bu oranı, temsil gücü yeterli fazla sayıda örnekten oluşmuş eğitim seti ile yakalamıştır. Eğitim seti örnek sayısı arttıkça ağın tanıma oranının artışı Tablo 4.1 de görülmektedir.

Tablo 4.1 : Geriye Yayılım Ağında Tanıma Yüzdesi – Örnek Sayısı Değişimi

Tanıma Yüzdesi	Eğitim Seti Örnek Sayısı
56	300
68	600
71	900
83	1200



Şekil 4.7 : Geriye yayılım ağı performans grafiği

4.3. SHASHANK AĞI PERFORMANS İNCELEMESİ

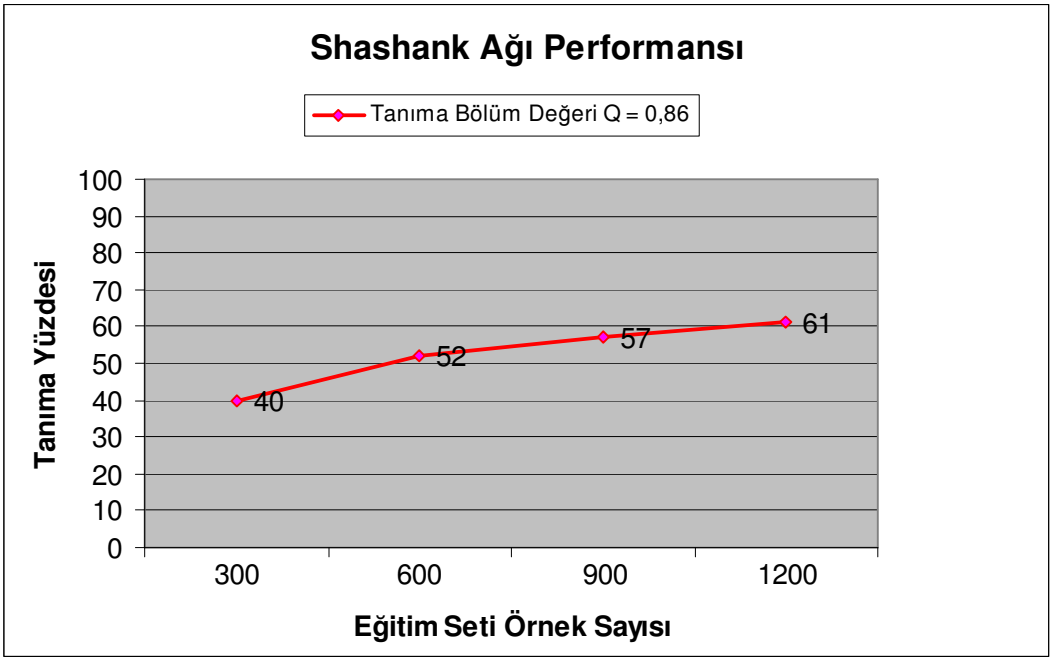
Shashank' ın geliştirdiği öğrenme algoritmasında hata ve hatayı geriye döndürerek ağa yayma kavramları yoktur. Öğrenme işlemi kendine özgü bir şekilde gerçekleşmektedir. Test aşamasında Shashank' ın Adaylık Skoru (Candidate Score) (ψ), İdeal Ağırlık Modeli Skoru (Ideal Weight – Model Score) (μ) ve Tanıma Bölüm Değeri (Recognition Quotient) gibi tanımlamaları kullanılır. Bir test örneğinin bir sınıfa ait olduğunu söyleyebilmek için tanıma bölüm değerinin (Q) en az 0.5 olması gerekir. Aksi halde

eğitim setini tekrar düzenilmesi gerektiği veya test örneği sınıfından olan karakterlerin hiç eğitilmediği yorumu yapılabilir.

Eldeki eğitim örnek seti ile Shashank ağının el yazısı karakterlerini tanıma oranı %61 olarak tespit edilmiştir (Şekil 4.8). Geriye yayılım ağında olduğu gibi oluşan bu ağda da, eğitim seti örnek sayısı arttıkça ağın tanıma yüzdesi artmıştır (Tablo 4.2).

Tablo 4.2 : Shashank Ağında Tanıma Yüzdesi – Örnek Sayısı Değişimi

Tanıma Yüzdesi	Eğitim Seti Örnek Sayısı
40	300
52	600
57	900
61	1200



Şekil 4.8 : Shashank ağı performans grafiği

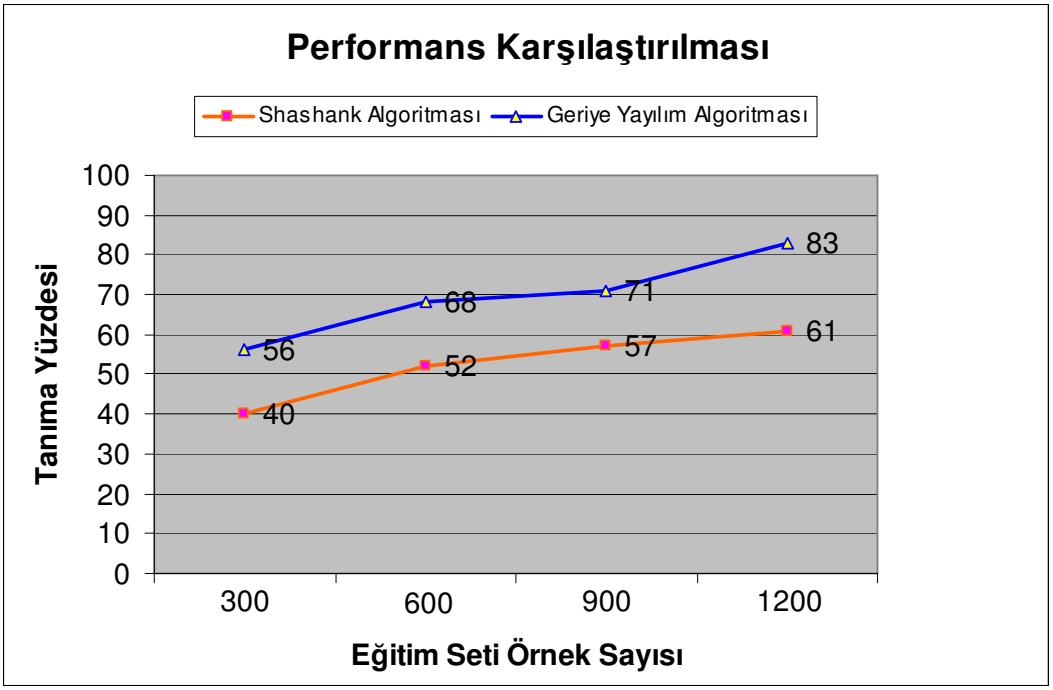
4.4. SHASHANK VE GERİYE YAYILIM ALGORİTMASI PERFORMANS VE EĞİTİM SÜRELERİ KARŞILAŞTIRMASI

4.4.1. Shashank Ve Geriye Yayılım Algoritması Performans Karşılaştırılması

Yapılan çalışmada Shashank ağının karakter tanıma yüzdesi %61 iken, geriye yayılım ağının tanıma yüzdesi %83' e kadar çıkmıştır. Bu sonuçlara göre geriye yayılım algoritması daha yüksek tanıma oranı ile daha iyi sonuç vermiştir (Şekil 4.9). Eğitim seti örnek sayısına göre tanıma yüzdeleri Tablo 4.3 de verilmiştir.

Tablo 4.3 : Shashank ve Geriye Yayılım Ağlarında Tanıma Yüzdesi – Örnek Sayısı Değişimi

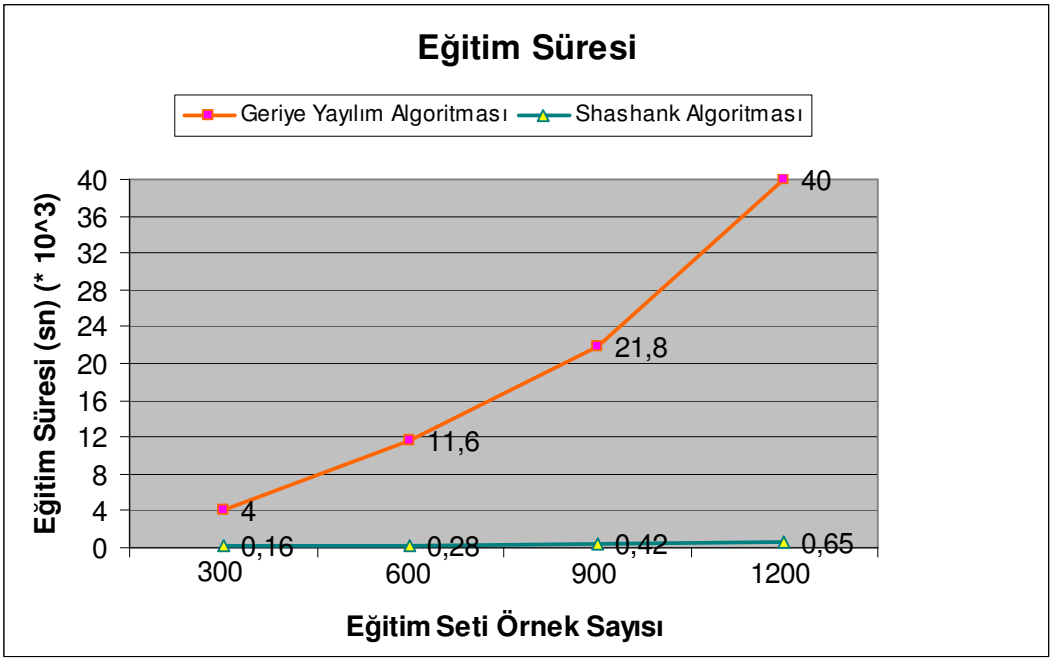
Geri Yayılım Alg. Tanıma Yüzdesi	**Shashank Alg. Tanıma Yüzdesi**	**Eğitim Seti Örnek Sayısı**
56	40	300
68	52	600
71	57	900
83	61	1200



Şekil 4.9 : Shashank ve geriye yayılım algoritmaları performans karşılaştırılması

4.4.2 Shashank Ve Geriye Yayılım Algoritması Eğitim Süresi Karşılaştırılması

Tanıma oranlarına bakıldığında geriye yayılım algoritmasının performansı çok daha yüksektir. Zaman tüketimi olarak incelendiğinde ise geriye yayılım algoritmasının eğitim süresi Shashank' ın algoritmasına göre oldukça fazladır (Şekil 4.10). Bunun nedeni geriye yayılım ağı proses elemanları ve bağlantılarıyla karmaşık bir yapıya sahiptir. Sistem eğitimi süresince her bağlantıda çeşitli işlemler gerçekleştirilir. İleri ve geri beslemedeki işlem sayısı zaman maliyetini artırmaktadır.



Şekil 4.10 : Shashank ve geriye yayılım algoritmaları eğitim süreleri karşılaştırılması

4.5. KARAKTER TANIMA SİMÜLASYONUNDAN ÖRNEK SONUÇLAR

	**A**
	**A**
	**B**
	**B**
	**C**
	**D**
	**O**
	**E**
	**R**
	**F**
	**G**
	**H**
	**T**
	**J**
	**K**
	**K**
	**L**
	**L**
	**M**

	N
	**M**
	**U**
	**N**
	**O**
	**O**
	**P**
	**Y**
	**D**
	**Q**
	**B**
	**R**
	**R**
	**K**
	**S**
	**S**
	**P**
	**T**
	**T**

	**U**
	**U**
	V
	V
	V
	V
	W
	W
	W
	X
	Y
	X
	Y
	Y
	T
	Z
	Z
	Z
	E

5.TARTIŞMA VE SONUÇ

Oluşturulan el yazısı karakterlerini tanıma sisteminde farklı kişilerden birçok el yazısı karakterler toplanmıştır. Karakterleri tanımada yapay sinir ağları kullanılmıştır. Oluşturulan eğitim seti ağa sunularak, farklı iki öğrenme algortimasıyla ağ eğitilmiştir. Bunlardan biri karakter tanıma sistemlerinde iyi sonuçlar veren geriye yayılım (backpropagation) algoritması diğeri ise Shashank' ın geliştirdiği bir öğrenme algoritmasıdır. Karakterleri tanımada MATLAB teknik programlama dili kullanılmıştır.

Geliştirilen sistemde geriye yayılım ağının performansı %83 iken Shashank ağının performansı %61 'de kalmıştır. Geriye yayılım algoritmasının performansının yüksek olmasının en önemli nedenlerinden biri, istenen çıkışa göre ağın çıkışlarının kontrol edilip hatanın bulunması ve bu hatanın bağlantılardan geriye yayılarak her seferinde düşürülmesidir. Diğer bir neden ise, eğitim sırasındaki hata – iterasyon değişimlerinin incelenip ağın parametrelerinin değiştirilebilme olanağıdır. Bu durum, toplanan eğitim seti örneklerine bağımlılığı bir derece azaltır.

Shashank' ın algoritmasında ise ağın öğrenmesi tamamıyla seçilen örneklere bağımlıdır. Çünkü hatayı geriye doğru yayıp azaltma veya ağın topolojisini değiştirme imkanı yoktur. Örneklerin sayısı az ya da temsil yeteneği yeterli seviyede değilse (gürültülü karakter örneği sayısı fazlaysa) ağın performansı bundan direk olarak etkilenir.

Sonuç olarak, geriye yayılım algoritması kullanılarak gerçekleştirilen öğrenmede performansı artırmak için eğitim setine doğal el yazısı karakteri olan çeşitli hatalı ve düzgün rasgele örnekler eklenebilir. Ağın topolojisi değiştirilebilir. Veya hata değerini (MSE) sıfıra daha yakın bir değere düşürmek gibi yöntemler uygulanabilir. Parametrelerin denenmemiş değerleri için daha fazla sayıda test yapılarak daha iyi sonuçlara yaklaşmak da ağ performansını artırır. Karakterlerin boyutu artırıldığında da ağın tanıma yüzdesi artacaktır.

Shashank'ın algoritmasının performansını artırmak içinse eğitim seti birbirine benzer el yazısı karakterlerden ya da makine karakterlerinden oluşturulabilir. Test örnekleri eğitim seti örneklerine benzer nitelikte tutulabilir. Karakterlerin boyutları ve sayıları artırılabilir.

Bu çalışmalar ışığında el yazısının kime ait olduğunu belirleme gibi çalışmalar da yapılabilir. El yazısının tanınması bilgisayar biliminin başlangıcından bu yana araştırmacıları meşgul eden zor problemlerdendir. Bu konuda çok sayıda çalışma yapılmasına rağmen halen problem devam etmektedir. Yazı içerisinde farklı karakterlerin benzerliği, harflerin birleştirilmesi, bilgilerin bulanıklığı ve çok çeşitliliği bu problemi zor kılan unsurlardandır. El yazısı yazı işaretlerinin yapılmasında gerekli olan ardışık el hareketidir. El yazısının kime ait olduğunu tanıyan sistemler, dokümanın yazarını belirleyebilirler. Bu tespit yazının sabit ve kişisel özellikler taşımasına dayanmaktadır. Bu açıdan kriminolojide de kişilik tespitinde kullanılır.

KAYNAKLAR

1. NABİYEV, V. V., 2003, *Yapay zeka*, Seçkin Yayınevi, Ankara, 975-347-68-76.

2. SIMON, H. A., 1983, *Why should machines learn? In: Machine learning – An artificial intelligence approach,* California.

3. VAN DER SMAGT, P. P., 1990, A Comparative study of neural network algorithms applied to optical character recognition, *IEA/AIE,* 2(1), 1037-1044.

4. ÖZTEMEL, E., 2003, *Yapay sinir ağları,* Papatya Yayıncılık, İstanbul, 975-67-97-39-8.

5. YURTOĞLU, H., 2005, *Yapay sinir ağları metodolojisi ile öngörü modellemesi : Bazı makroekonomik değişkenler için Türkiye örneği* [online], Devlet Planlama Teşkilatı, http://ekutup.dpt.gov.tr/ekonomi/tahmin/yurtoglh/ysa.pdf [Ziyaret Tarihi: 14 Aralık 2005].

6. DARPA, 1988, Neural Network Study*, AFCEA International Press*, 60.

7. MAKHFI, 2004, Introduction *to ANN* [online], http://makhfi.com/introduction.htm [Ziyaret Tarihi: 3 Ekim 2005].

8. CHIANG, C. C. ve FU, H.C., 1991, Using neural nets to recognize handwritten printed characters, *IEEE Signal Processing*, 492-495.

9. XATZH, 2000, Yapay sinir ağları teknolojileri [online], http://batitrakya.dostweb.com/yapaysiniraglari/yapaysiniraglari.htm [Ziyaret Tarihi: 29 Ocak 2006].

10. MINSKY, M. ve PAPERT, S., 1969, Perceptrons, *The MIT Press.*

11. ROSENBLATT, F., 1958, The perceptron: A probabilistic model for information storage and organization in the brain, Psychoanalytic Review, 65, 386-408.

12. CROCHAT, P. ve FRANKLIN, D., 2002, *Backpropagation neural network tutorial* [online], http://pcrochat.online.fr/webus/tutorial/ [Ziyaret Tarihi: 3 Ekim 2005].

13. KAWAGUCHI, K., 2000, *A multithreaded software model for backpropagation neural network applications*, Thesis (MD), The University Of Texas At El Paso.

14. BANKS, S. P., 1990, Signal processing, image processing and pattern recognition, Prentice Hall, 0-13-812587-2.

15. RIPLEY, B. D., 1996, *Pattern recognition and neural networks*, Cambridge University Press, Cambridge, 0-521-46086-7.

16. BATSAIKHAN, O. ve SINGH, Y. P., 2002, Mongolian character recognition using multilayer perceptron (MLP), *Proceedings of the 9th International Conference on Neural Information Processing (ICONIP'OP),* 18-22 November 2002, Hawaii, Soo Young Lee: Xin Yao, 621 – 625.

17. SALAMEH, W. A., OTAIR M. A., 2005, Online handwritten character recognition using an optical backpropagation neural network, *Issues in Informing Science and Information Technology,* 787-795.

18. KAMRUZZAMAN, J., 2001, Comparison of feed-forward neural net algorithms in application to character recognition, *Proc. of IEEE TENCON2001*, 165-169.

19. FAABORG, A. J., 2002, Using Neural Networks to Create an Adaptive Character Recognition System, *Modeling Perception and Cognition, 5-8.*

20. ERDEM, O. A. ve UZUN, E., 2004, Yapay sinir ağları ile türkçe times new roman, arial ve elyazısı karakterleri tanıma, *Gazi Üniv. Müh. Mim. Fak. Der.*, 20(1), 13-19.

21. ARAOKAR, S., 2005, Visual character recognition using artificial neural networks, *MGM's College of Engineering andTechnology,* 0505016.

22. LEE, H. J. ve LIN Y. C., 1998, Using confusing characters to improve character recognition rate, In proceedings of IEEE conf. system, man and cybernetics, 4195-4200.

ÖZGEÇMİŞ

Pelin ARAS, 28.05.1981 tarihinde İstanbul'da doğmuştur. İlkokul eğitimini Yeşilköy Halil Vedat Fıratlı İlkokulu' nda, ortaokul ve lise eğitimini Büyükşehir Hüseyin Yıldız Anadolu Lisesi' nde tamamlamıştır. 1999 yılında İstanbul Üniversitesi Bilgisayar Mühendisliği bölümünü kazanarak 2003 yılında mezun olmuştur. Aynı sene İstanbul Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'ndaYüksek Lisans eğitimine başlamıştır.

Ağustos 2004'te İstanbul Üniversitesi Bilgisayar Mühendisliği Bölümü'nde Araştırma Görevlisi olarak çalışmaya başlamış ve halen aynı görevi sürdürmektedir.