YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

STANDART VE HİBRİD YAPILAR KULLANARAK YAPAY SİNİR AĞLARI İLE İMZA TANIMA

Elektrik-Elektronik Mühendisi Canan ŞENOL

FBE Elektronik ve Haberleşme Anabilim Dalı Elektronik Programında Hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı : Doç. Dr. Tülay YILDIRIM

Yrd. Doç. Dr. Songül Albayrak

Yrd. Doç. Dr. Lale Özyılmaz

İSTANBUL, 2004

İÇİNDEKİLER

Sayfa

SİMGE LİSTESİ iv

KISALTMA LİSTESİ v

ŞEKİL LİSTESİ vi

ÇİZELGE LİSTESİ vii

ÖNSÖZ viii

ÖZET ix

ABSTRACT x

1. GİRİŞ 11

2. İMZA TANIMA VE DOĞRULAMA İŞLEMLERİ 14

2.1 İmza Tanıma ve İmza Doğrulama Kavramları 14
2.2 Online ve Offline İmza Tanıma ve Doğrulama 14
2.3 Önişlemler 14
2.3.1 Görüntü İşleme 14
2.3.2 Gürültü İndirgeme 14
2.3.3 Boyut Normalizasyonu 15
2.3.4 İskelet Çıkarma 15
2.4 Özellik Çıkarma 15
2.4.1 Genel Özellikler 15
2.4.1.1 Resim Alanı 16
2.4.1.2 Tam Genişlik ve Tam Yükseklik 16
2.4.1.3 Maksimum Yatay ve Düşey İzdüşüm, Yatay ve Düşey İzdüşüm Tepeleri 16
2.4.1.4 Taban Çizgisi Kayması 16
2.4.1.5 Köşe Noktası Sayısı 17
2.4.1.6 Kapalı Çevrim Adedi 17
2.4.1.7 Kesişme Noktası Sayısı 18
2.4.1.8 Genel ve Yerel Meyil Açıları 19
2.4.2 Grid Bilgi Özellikleri 19

3. YAPAY SİNİR AĞLARI 20

3.1 Biyolojik Nöron 20
3.2 Yapay Sinir Ağlarının Tarihsel Gelişimi 21
3.3 Yapay Sinir Ağlarının Özellikleri ve Kullanıldığı Yerler 21
3.4 Yapay Sinir Modeli 22
3.5 Kullanılan Ağ Yapıları 23
3.5.1 Çok Katmanlı Algılayıcılar (Multilayer Perceptron - MLP) 23
3.5.2 Radyal Temelli Fonksiyonlar (Radial Basis Function – RBF) 23

3.5.3 Konik Kesit Fonksiyonlu Ağlar (Conic Section Function Neural Network-CSFNN) 25

4. İMZA TANIMA VE DOĞRULAMA PROBLEMİ 28

4.1 Önişlemler 28
4.2 Temel Bileşen Analizi (PCA) 32
4.3 Simülasyon Sonuçları 37
4.3.1 İmza Tanıma 38
4.3.1.1 Normal SR 38
4.3.1.2 PCA Kullanılarak SR 39
4.3.2 İmza Doğrulama 40
4.3.2.1 Normal SV 40
4.3.2.2 PCA Yöntemiyle SV 41

5. SONUÇLAR 42

KAYNAKLAR 43

EKLER 44

Ek 1 SRVS eğitme için MLP ile yazılmış program 45
Ek 2 SRVS eğitme için RBF ile yazılmış program 47
Ek 3 MLP ve RBF ile eğitilmiş ağın test edilmesi için yazılmış program 48
Ek 4 SRVS için CSFNN ile yazılmış program 49

ÖZGEÇMİŞ 59

SİMGE LİSTESİ

A	Herhangi bir parçanın alan değeri
A_{max}	Maksimum alan değeri
A_{min}	Minimum alan değeri
A_N	A'nın normalize değeri
a	Koninin ekseni
c_i	Merkez değeri
CL	İmza resmindeki kapalı çevrim adedi
EL	İmza resmindeki ayrılma adedi
EP	İmza resmindeki köşe noktası sayısı
$\varphi_i(x)$	Aktivasyon fonksiyonu
σ_i	Genişlik değeri
ω	Açılma açısı
ω_{ij}	Gizli katmandaki i. nöron ile j. nöron arasındaki ağırlık
x	Giriş vektörü
v	Koninin tepe noktası

KISALTMA LİSTESİ

MLP	Çok Katmanlı Algılayıcı (Multilayer Perceptron)
PCA	Temel Bileşen Analizi (Principal Component Analysis)
RBF	Radyal Temelli Fonksiyon (Radial Basis Function)
SR	İmza Tanıma (Signature Recognition)
SV	İmza Doğrulama (Signature Verification)
SRVS	İmza Tanıma ve Doğrulama Sistemleri (Signature Recognition and Verification Systems)
YSA	Yapay Sinir Ağı

ŞEKİL LİSTESİ

Şekil 2.1 Merkez piksel ve 8-komşuluğu 15
Şekil 2.2 Tam genişlik ve tam yükseklik 16
Şekil 2.3 Bir köşe noktası 17
Şekil 2.4 Kapalı çevrim örneği 18
Şekil 2.5 Bir kesişme noktası 18
Şekil 3.1 Biyolojik nöron modeli 20
Şekil 3.2 İşlem elemanının modeli 22
Şekil 3.3 İki katmanlı MLP yapısı 23
Şekil 3.4 RBF modeli 25
Şekil 3.5 Konik Kesit Fonksiyonlu Ağ modeli 26
Şekil 4.1a Gerçek imzalar 29
Şekil 4.1b Gerçek imzalar 30
Şekil 4.2 Sahte imzalar 30
Şekil 4.3 Bir imza örneği ve bu imzanın iskeleti 31
Şekil 4.4 Bir imza örneği ve bu imzanın gridi 31
Şekil 4.5a Giriş vektörü oluşturmaya ait işaret akış diyagramı 33
Şekil 4.5b Eğitmeye ait işaret akış diyagramı 34
Şekil 4.5c Tanımaya ait işaret akış diyagramı 35
Şekil 4.6a CSFNN ile eğitmeye ait işaret akış diyagramı 36
Şekil 4.6b Geriye yayılım algoritmasına ait işaret akış diyagramı 37

ÇİZELGE LİSTESİ

Çizelge 4.1 SR için sınıflanan eğitme imzaları sayısı 38
Çizelge 4.2 SR için sınıflanan test imzaları sayısı 39
Çizelge 4.3 SR için PCA kullanılarak sınıflanan eğitme imzaları sayısı 39
Çizelge 4.4 SR için PCA kullanılarak sınıflanan test imzaları sayısı 40
Çizelge 4.5 SV için tanınamayan taklit imza sayısı 40
Çizelge 4.6 SV için PCA kullanılarak tanınamayan taklit imza sayısı 41

ÖNSÖZ

Bu tezde Yapay Sinir Ağları kullanılarak imza tanıma ve doğrulama problemi incelenmiştir. Simülasyonlarda Matlab 6.5 Neural Network Toolbox kullanılarak tanıma ve doğrulama yapılmıştır. Kullanılan sınıflama yöntemi offline imza tanıma ve doğrulamadır. Tanıma ve doğrulama için Çok Katmanlı Algılayıcı, Radyal Temelli Fonksiyonlar ve Konik Kesit Fonksiyonlar olmak üzere üç farklı yöntem kullanılmış ve bunların karşılaştırmalı sonuçları incelenmiştir.

Gerek tez çalışmam sırasında, gerekse de yüksek lisans eğitimim boyunca bana her türlü olanağı ve desteği sağlayan danışman hocam Sayın Doç. Dr. Tülay Yıldırım'a ve eşim Öğr. Gör. Habib Şenol'a teşekkürü bir borç bilirim.

Ayrıca; tez çalışmam sırasında küçücük yaşına rağmen bana en azından uslu durarak yardımcı olan oğlum Mert'e bu tezimi hediye ediyorum.

ÖZET

Bu tezin amacı yapay sinir ağları kullanarak offline imza tanıma ve doğrulama işlemi yapmaktır. Son zamanlarda yapay sinir ağları sınıflama, örnek tanıma ve doğrulama problemlerinde iyi sonuç vermelerinden dolayı çok tercih edilir olmuşlardır. Tezde, imza tanıma ve doğrulama işlemi önce çok bilinen ve yaygın olarak kullanılan yapay sinir ağı yapılarından olan Çok Katmanlı Algılayıcı ve Radyal Temelli Fonksiyonlar, daha sonra hibrid bir yapı olan Konik Kesit Fonksiyonlu Ağ kullanılarak yapılmış ve sonuçları verilmiştir. 8 farklı kişiden imza tanıma için alınan 256 imzanın 200 tanesi (bir kişiden 25 tane) eğitmede kullanılmış, kalan 56 imza (bir kişiden 7 tane) ile de ağ test edilmiştir. Bu veri kümesine imza doğrulama için 2'şer tane (toplam 16 tane) taklit imza eklenmiştir. Simülasyonlarda Matlab 6.5 Neural Network Toolbox kullanılmıştır. Konik Kesit Fonksiyonlu Ağ kullanılarak yapılan imza tanıma başarı yüzdelerinin diğer iki yönteme göre biraz daha düşük olmasına rağmen, eğitme ve test işlemlerinin daha hızlı olması ve hedef vektörü ile neredeyse aynı olan test sonuçlarına ulaşması bakımından oldukça dikkat çekicidir. Ayrıca Konik Kesit Fonksiyonlu Ağlarda kullanılan nöron sayısı diğer yöntemlere göre çok azdır ki bu da pratikte uygulanması açısından diğer yöntemlere göre üstünlük sağlar.

Anahtar kelimeler: İmza tanıma ve doğrulama, çok katmanlı algılayıcı, radyal temelli fonksiyonlar, konik kesit fonksiyonlu ağlar.

ABSTRACT

This research presents a new approach for offline signature recognition and verification based on artificial neural networks (ANNs). ANNs have recently become a very important method for classification and recognition problems. In this work, two well known neural network architectures (Multilayer Perceptron and Radial Basis Function Networks) and a hybrid neural network (Conic Section Function Neural Networks) are proposed for the signature recognition. The proposed system was trained and tested on a signature database consisting of a total of 256 signature images taken from 8 different persons. A total of 200 samples (25 samples for each person) for training and 56 samples (7 samples for each person) for testing are used. 16 samples (2 samples for each person) added this signature database for verification. The results and comparisons are presented. Although Conic Section Function Neural Networks gives results slightly worse than other two networks for signature recognition, the size of Conic Section Function Neural Network structure is quite small compared to those standard ones. It needs only 20 (12+8) neurons while Multilayer Perceptron needs 92 (60+24+8) and Radial Basis Function needs 183 (175+8) neurons for training. This result gives superiority to Conic Section Function Neural network for neural network hardware implementations in practice.

Keywords: Signature recognition and verification, Multilayer Perceptron, Radial Basis Function, Conic Section Function Neural Networks

1. GİRİŞ

Bu tez ile ilgili olarak bugüne kadar yapılmış çalışmalar imza tanıma ve/veya imza doğrulamayı içerir. İmza tanımada, farklı imzalar ağa tanıtılır ve bu imzalar arasından rastgele seçilen bir imzanın kime ait olduğu bulunur. İmza doğrulamada ise imzalar ağa tanıtıldıktan sonra taklit edilen bir imzanın ağ tarafından tanınıp tanınmadığına bakılır. İmza tanıma, imza doğrulamaya göre daha uzun ve zor bir işlem olduğundan literatürdeki uygulamalar daha çok imza doğrulama üzerinedir.

İmza tanıma ve/veya imza doğrulama probleminin çözümünde iki temel sınıflama yöntemi vardır:

1. Online imza tanıma ve doğrulama sistemleri (Online SRVS)
2. Offline imza tanıma ve doğrulama sistemleri (Offline SRVS)

İmza tanıma ve/veya imza doğrulama problemi için öncelikle imzanın çeşitli özellikleri çıkartılarak bu özelliklere göre eğitme yapılmalıdır. İyi bir özellik çıkarma, eğitme ve test sonucunda başarı yüzdelerinin artması açısından çok önemlidir. İmzanın karakteristik özelliklerine göre çıkartılabilecek bazı genel özellikler arasında; yükseklik-genişlik oranı, maksimum yatay izdüşüm, maksimum düşey izdüşüm, yatay izdüşüm tepeleri, düşey izdüşüm tepeleri, imza alanı, imzanın yatay merkezi, imza yüksekliği, genel ve yerel meyil açıları, köşe noktası sayısı, kesişme noktası sayısı ve kapalı çevrim adedi sayılabilir. Bunlardan başka özellikleri çıkarmak kullanıcıya bağlıdır.

Online SRVS imza atarken ortaya çıkan hız, yön ve basınç bilgilerini ölçüp kaydetmeye yarayan bir takım özel donanımlar (dijital kalem, dijital ped, özel baskılı kağıtlar v.b.) gerektirir. Bir imzanın gerçek mi sahte mi olduğunu anlamak online SRVS'de daha kolaydır. Online SRVS, imza atarken insan elinin hızına, imzanın atılış yönüne ve kağıda uyguladığı basınca bağlı olarak özel ölçümleri hafızasında tuttuğu için taklit edilen imzayı daha kolay anlar. Online SRVS üzerine yapılmış çalışmalara bir örnek olarak Keit vd., (2001)'yi verebiliriz. Bu çalışmada internet ve e-ticaret uygulamalarında kullanılmak üzere kalemin kağıda uyguladığı basıncı ölçerek online SV yapılmıştır. 20 kişiden toplam 1000 tane imza toplanmıştır. Bu imzaların her biri için dijital kalemin ucunun kağıt üzerinde uyguladığı basınç ölçülmüş ve geriye yayılım algoritması kullanılarak ağ eğitilmiştir. Sahte imzaların %3.40'lık hata ile kabul edilme oranı ve gerçek imzaların %2.13'lük hata ile reddedilme oranıyla araştırmalarında tatmin edici sonuçlar bulmuşlardır.

Offline SRVS'de ise görüntü işleme ve özellik çıkarma teknikleri kullanılarak sınıflama yapılabilir. Murshed vd., (1995) tarafından 5 kişiden 40'ar tane imza toplanmış ve bu imzaların 18'er tanesi eğitmede 22'şer tanesi testte kullanılarak imza doğrulama yapılmıştır. Eğitmede kullanılan imzalar gerçek imzalardır. Fuzzy ARTMAP Neural Network kullanılarak imza doğrulama yapılmıştır. Bu yapıya göre imzalar sözcük sayısı*2 bölgeye ayrılmış ve her bölge de bir kenarı 16-piksel olan karelere bölünerek grid bilgisi çıkarılmıştır. Grid bilgisine göre geriye yayılım algoritmasıyla eğitme yapılarak imzanın gerçek mi sahte mi olduğu anlaşılmıştır. En yüksek başarı %91.64 olarak elde edilmiştir.

Bajaj ve Chaudhury (1996) ise imza doğrulama için alt ve üst zarf eğrileri ile yatay ve düşey izdüşüm momentlerini çıkartarak özellik vektörlerini oluşturmuş ve MLP-ADALINE hibrid yapısını kullanarak eğitme yapmışlardır. 10 kişiden 15'er tane imza toplanmış ve toplanan imzalar arasından rastgele seçilen 5'er tane ile eğitme yapılmıştır. Test için ise kalan imzalara toplam 100 tane taklit imza eklenmiştir. Ortalama %10.3 hata ile sonuç elde etmişlerdir.

Huang ve Yan (1996) ise imzanın iskeletini çıkartmış, imzayı çevreleyen dış çerçevesi, iyi ve kötü mürekkep dağılımı, imzanın genişlik ve uzunluk bilgilerine göre her imzaya ait özellik vektörü oluşturmuş ve bunu MLP ile eğiterek doğrulama yapmışlardır. 504 gerçek ve 3024 taklit imza ile gerçekleştirilen bu çalışmada doğru sınıflama oranı ortalama %90 bulunmuştur.

Offline imza doğrulama alanındaki çalışmalardan biri de Zhou ve Quek (1996) tarafından yapılmıştır. Fuzzy Neural Network kullanılarak yapılan bu çalışmada taban çizgisi, basınç ve eğim özellikleri kullanılmıştır. Hem gerçek hem taklit imzalar kullanılarak bir eğitme yapıldıktan sonra, sadece gerçek imzalar kullanılarak da bir eğitme yapılmış ve sonuçları karşılaştırılmıştır. Bu karşılaştırmanın sonucunda sadece gerçek imzalar kullanılarak yapılan eğitmede, ustalıkla taklit edilmiş bir imzanın bile tanınabildiği görülmüştür.

Baltzakis ve Papamarkos (2001) ise imza yüksekliği, resim alanı, tam yükseklik ve genişlik, imzanın yatay ve düşey merkezi, maksimum yatay ve düşey izdüşüm, yatay ve düşey izdüşüm tepeleri, genel ve yerel meyil açıları, kapalı çevrim adedi, köşe ve kesişme noktası sayıları gibi genel özelliklerin yanısıra grid bilgisi ve doku özelliklerini de kullanarak imza tanıma ve doğrulama yapmışlardır. Bir kişiden 15-25 imza olmak üzere 115 kişiden 2000 imza toplanarak yaptıkları bu eğitmede MLP-RBF hibrid yapısını kullanmışlardır. İmza tanımada doğru sınıflama oranı %80.81 bulunmuştur. İmza doğrulamada ise gerçek imzanın sahibini %97 doğru %3 yanlış sınıflarken, imzanın taklit olduğunu %90.019 oranla doğru sınıflarken %9.81 oranında ayırt edememiştir.

İmza doğrulama için yapılmış bir başka çalışma da Hanmandlu vd., (2003)'ye aittir. 4 kişiden 100'er imza olmak üzere 400 imzanın eğitmede 221 imzanın da testte kullanıldığı bu çalışmada Yapay Sinir Ağı ve Bulanık Mantık birlikte kullanılmıştır. Ağın başarısı 1. kişi için %95, 2. kişi için %88.89, 3.kişi için %90 ve 4. kişi için %86 bulunmuştur.

İmza tanıma ve doğrulamanın birlikte yapıldığı bir çalışma da Öz vd., (2003) tarafından yapılmıştır. 5 kişiden 6'şar imza toplanarak eğitme yapılmıştır. Diğer bütün çalışmalarda da ortak olan görüntü işleme, renkli görüntüyü siyah-beyaz hale çevirme, gürültü indirgeme, arka planın yok edilmesi ve boyut normalizasyonu önişlemlerinden sonra, bu çalışmada cebirsel moment hesabı yapılmış ve geriye yayılım algoritması kullanılmıştır. Oluşturulan ağ yapısı imza tanıma için 14-18-30, imza doğrulama için ise 14-10-2'dir. İmza tanımada ağın başarısı nöron çıkışları 0.5'den küçükse 0, 0.5-1 arasındaysa 1 alınarak %100 bulunmuştur. İmza doğrulamada ise 5 gerçek ve 5 taklit imza ile ağ test edilmiştir. Eğitilen ağ 4 gerçek ve 5 taklit imzayı doru sınıflamıştır.

Şimdiye kadar yapılmış çalışmaların çoğu offline imza tanıma ve/veya doğrulama üzerinedir ve genellikle imza doğrulama yapılmıştır. Bu tezde literatürde çok yapılmamış olan imza tanıma işlemi gerçekleştirilmiştir. İmza doğrulama üzerine de bazı çalışmalar yapılmıştır.

İmza tanıma ve/veya imza doğrulama probleminde kullanılan yöntem ne olursa olsun temelde yapılan önişlemler benzerdir Bu tezde genel özellik olarak imzanın yatay merkezi, imzanın düşey merkezi ve taban çizgisi kayması özelliklerinin yanısıra grid bilgi özellikleri de kullanılmıştır.

Bu tezde Matlab Neural Network Toolbox kullanılmış ve bu program Ekler kısmında verilmiştir. 2. bölümde imza tanıma ve doğrulama işlemlerinde kullanılan temel kavramlar, yapılan önişlemler ve imzadan çıkarılabilecek özellikler anlatılmıştır. 3. bölüm yapay sinir ağları ve tezde kullanılan ağ yapılarını içerir. 4. bölümde ise imza tanıma ve doğrulama problemi incelenmiş ve simülasyon sonuçları tablolar halinde gösterilmiştir. 5. ve son bölümde de sonuçlar sunulmuştur.

2. İMZA TANIMA VE DOĞRULAMA İŞLEMLERİ

2.1 İmza Tanıma ve İmza Doğrulama Kavramları

İmza tanımada amaç; eğitme imzaları ile yapay sinir ağını eğittikten sonra rastgele seçilen bir test imzasının sahibini bulmaktır. Bunun için imzalar Bölüm 2.3'de belirtilen önişlemlerden geçirildikten sonra eğitme işlemi gerçekleştirilir ve test imzaları arasından rastgele bir imza seçilerek bu imzanın sahibi araştırılır.

-İmza doğrulamada ise eğitme imzaları, eğitmeden önce yukarıda belirtilen önişlemlerden geçirilir. Test için bu kullanıcılara ait taklit imzalar kullanılır. Amaç ağın taklit imzaları ayırabilmesidir.

2.2 Online ve Offline İmza Tanıma ve Doğrulama

Eğitmede yöntem ne olursa olsun sınıflama yapmak için belli başlı 2 yöntem vardır. Bunlar Online SRVS ve Offline SRVS'dir. Eğitme işlemine geçilmeden önce sınıflama yöntemi belirlenmelidir. Online SRVS imza atarken ortaya çıkan hız, yön ve basınç bilgilerini ölçüp kaydetmeye yarayan bir takım özel donanımlar (dijital kalem, dijital ped, özel baskılı kağıtlar v.b.) gerektirir ki bu extra maliyetten dolayı çok tercih edilen bir yöntem değildir. Offline SRVS'de ise görüntü işleme ve özellik çıkarma teknikleri kullanılarak sınıflama yapılabilir.

2.3 Önişlemler

İmza tanıma ve doğrulama probleminde eğitme ve test işlemlerinden önce imzanın birtakım önişlemlerden geçirilmesi gerekmektedir.

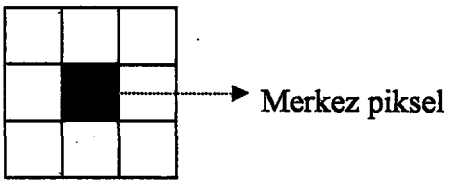
2.3.1 Görüntü İşleme

Bu aşamada kişilerden toplanan imzalar bir tarayıcı yardımıyla bilgisayar ortamında kullanıma hazır hale getirilir. Tarayıcının çözünürlüğünü ayarlamak kullanıcıya bağlıdır. Düşük çözünürlükte imzalar çok iyi ayırt edilemeyebilir. Literatürdeki uygulamalarda çözünürlük 100-400 arası seçilmiştir.

2.3.2 Gürültü İndirgeme

Bu aşamada gürültü indirgeme filtresi siyah beyaz taranmış resme uygulanır. Bunu yapmaktaki amaç; arka plan beyaz ise üzerindeki siyah pikselleri, siyah ise üzerindeki beyaz

pikselleri bularak yok etmektir. Bunu yapabilmek için resme, basit bir karar kuralı olan 3x3'lük bir maske uygulanır. Merkez piksel, çevresindeki birbiriyle aynı renkteki 8 pikselin renginden farklı ise merkez pikselin rengi değiştirilir.



Şekil 2.1 Merkez piksel ve 8-komşuluğu

İmza resmindeki gürültülerin çıkarılmasından sonra yapılacak işlem imza alanının kesilip alınmasıdır. Yatay ve düşey izdüşümlü parçalama metodu kullanılarak imza alanı arka plandan kesilip alınır. Böylece imzayı çerçeveleyen beyaz boşluk gözden çıkarılmış olunur.

2.3.3 Boyut Normalizasyonu

Resmin genişliğinin kabul edilen değere ulaşması için, genişlik-yükseklik oranının değişmemesi şartıyla resmin boyutu ayarlanır.

2.3.4 İskelet Çıkarma

Önişlemlerin son aşamasıdır. İmzanın karakteristiğini bozmadan gereksiz piksellerin atılması mantığına dayanır. Daha sonra yapılacak olan bütün işlemlerde kullanılacak olan imza iskelet hale getirilmiş imzadır.

2.4 Özellik Çıkarma

İmza tanıma ve doğrulama işlemi için öncelikle imzanın çeşitli özellikleri çıkarılarak bu özelliklere göre eğitme yapılmalıdır. İyi bir özellik çıkarma, eğitme ve test sonucunda başarı yüzdelerinin artması açısından çok önemlidir. Bir imzaya ait özellikler genel özellikler ve grid bilgi özellikleri olmak üzere 2 ana başlıkta toplanabilir.

2.4.1 Genel Özellikler

İmzanın genel özellikleri arasında imza yüksekliği, resim alanı, tam genişlik ve tam yükseklik, maksimum yatay ve düşey izdüşüm, yatay ve düşey izdüşüm tepeleri, imzanın

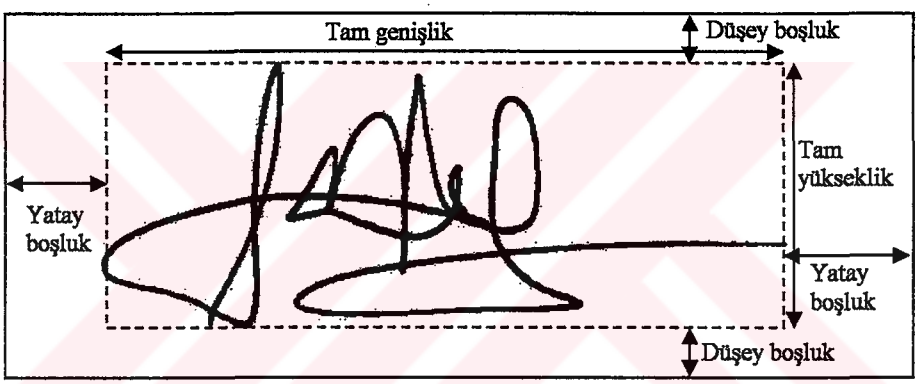
yatay ve düşey merkezi, genel ve yerel meyil açıları, taban çizgisi kayması, köşe noktası sayısı, kesişme noktası sayısı ve kapalı çevrim adedi sayılabilir. (Baltzakis ve Papamarkos, 2001)

2.4.1.1 Resim Alanı

İmza resminde ön planda kalan siyah piksellerin sayısıdır. İskelet hale getirilmiş imza resminde bu, imza izinin yoğunluk ölçüsünü gösterir. İmza alanı adıyla da kullanılabilir.

2.4.1.2 Tam Genişlik ve Tam Yükseklik

Tam genişlik yatay boşluklar, tam yükseklik ise düşey boşluklar çıkartıldıktan sonra resmin genişlik ve yüksekliğidir. Bu tanım Şekil 2.2'nin incelenmesiyle daha iyi anlaşılır.



Şekil 2.2 Tam genişlik ve tam yükseklik

2.4.1.3 Maksimum Yatay ve Düşey İzdüşüm, Yatay ve Düşey İzdüşüm Tepeleri

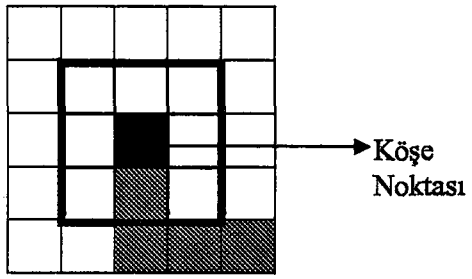
İskeleti çıkarılmış imza resminin düşey izdüşümü alınır. Bu izdüşümler arasındaki en büyük değer maksimum düşey izdüşümüdür. Düşey izdüşüm tepeleri de düşey izdüşümlerdeki yerel maksimum adedidir. Maksimum yatay izdüşüm ise, imza resminin yatay izdüşümleri arasındaki en büyük değerdir. Yatay izdüşüm tepeleri de yatay izdüşümlerdeki yerel maksimum adedidir.

2.4.1.4 Taban Çizgisi Kayması

İskelet haline getirilmiş imza resmi dikey olarak ikiye bölünür. Resmin sol yarısı ile sağ yarısının ağırlık merkezlerinin düşey bileşenleri arasındaki fark hesaplanır. Bu farka taban çizgisi kayması denir.

2.4.1.5 Köşe Noktası Sayısı

Bir köşe noktası, 8 komşuluğunda sadece bir komşusu ile aynı renkte (siyah) olan bir imza noktasıdır. Şekil 2.3 bir köşe noktasını göstermektedir. Köşe noktası sayısı da imza resmindeki köşe noktalarının adedidir.



Şekil 2.3 Bir köşe noktası

2.4.1.6 Kapalı Çevrim Adedi

EP imza resmindeki köşe noktası sayısı, EL de imza resmindeki ekstra ayrılma adedini göstermek üzere bir imzadaki kapalı çevrim adedi (2.2) denkleminde olduğu gibidir.

$$EL = \sum_{\text{kesişme noktaların da}} (\text{kesişme noktası komşu adedi} - 2) \tag{2.1}$$

$$CL = 1 + \frac{EL - EP}{2} \tag{2.2}$$

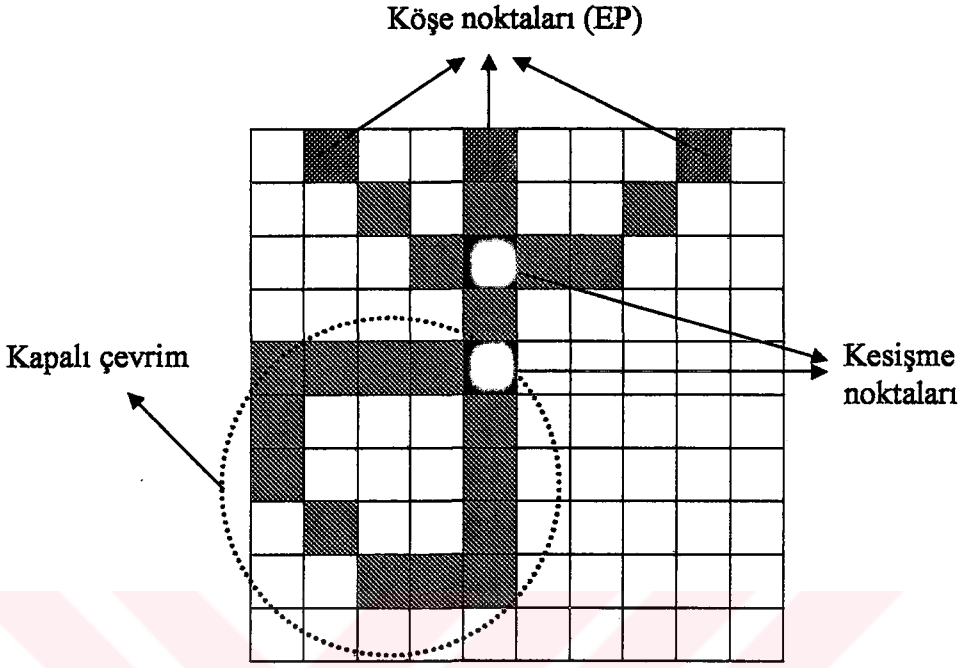
Şekil 2.4 de bir kapalı çevrim görülmektedir.

Burada;

EP=3

$$EL = \sum_{\text{kesişme noktaların da}} (\text{kesişme noktası komşu adedi} - 2) = (4\text{-}2) + (3\text{-}2) = 3$$

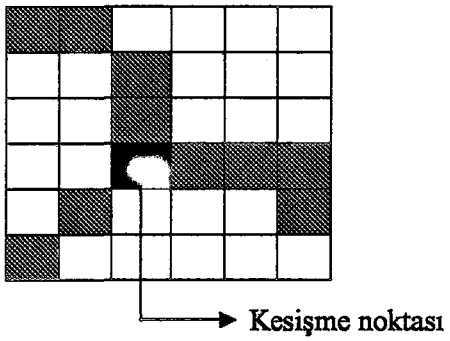
$CL = 1 + \frac{EL - EP}{2} = 1 + \frac{3-3}{2} = 1$ adet kapalı çevrim vardır.



Şekil 2.4 Kapalı çevrim örneği

2.4.1.7 Kesişme Noktası Sayısı

Bir kesişme noktası, 8 komşuluğunda birbiriyle bitişik olmayan en az 3 komşusu ile aynı renkte (siyah) olan bir imza noktasıdır. Şekil 2.5 bir kesişme noktasını göstermektedir. Kesişme noktası sayısı da imza resmindeki kesişme noktalarının adedidir.



Şekil 2.5 Bir kesişme noktası

2.4.1.8 Genel ve Yerel Meyil Açıları

Resim, -30 dereceden +40 dereceye kadar 1 derecelik adımlarla döndürülür. Her adımda düşey 3 piksel bağlantılarının adedi hesaplanır. Düşey 3 piksel bağlantılarının en fazla olduğu açı "Genel Meyil Açısı"dır.

Yerel meyil açısını bulmak içinde resim yukarıdaki yöntemle döndürülür. Düşey izdüşüm sayıları hesaplanır. Ve bu sayıların en büyük 70 değeri toplanır. Toplamın en büyük olduğu açı "Yerel Meyil Açısı"dır.

2.4.2 Grid Bilgi Özellikleri

Grid bilgisini elde etmek için ise iskelet hale getirilmiş resim 96 dikdörtgen parçaya (8 x 12) bölünür. Her parça için alan yani ön plandaki piksellerin (siyah pikseller) adedi hesaplanır. Bulunan sonuç, en düşük alan değerine sahip parçanın alanı 0, en yüksek alan değerine sahip parçanın alanı 1 olacak şekilde normalize edilir.

A herhangi bir parçanın alan değeri, A_{max} maksimum alan değeri, A_{min} minimum alan değeri olmak üzere A'nın normalize değeri A_N (2.3) denklemindeki gibi hesaplanır.

$$A_N = \frac{A - A_{min}}{A_{max} - A_{min}} \tag{2.3}$$

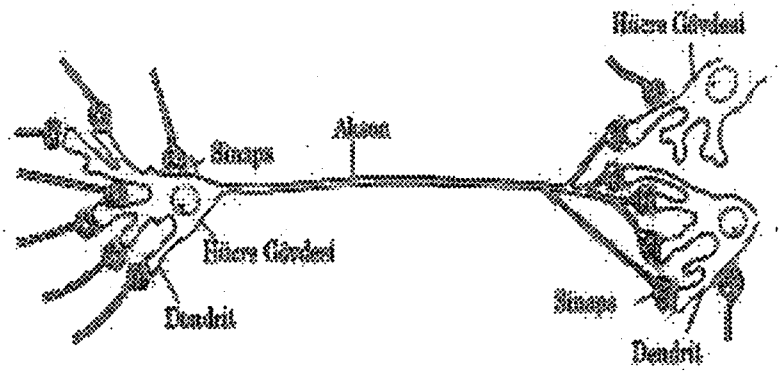
Sonuçta; 12 x 8 elemanlı ve her bir elemanı 0 ile 1 arasında değerler alan bir grid matrisi elde edilir. Bu da parçalardaki siyah piksellerin yoğunluğunu gösterir.

3. YAPAY SİNİR AĞLARI

Son yıllarda sınıflama, doğrulama ve tanıma problemlerinde sıkça kullanılan yöntemlerden biri olan YSA'lar insan beyninin çalışma prensibi üzerine oluşturulmuştur. YSA'lar giriş ve çıkış verilerini kullanarak algoritmalar geliştirir ve bu şekilde sistemin davranışını öğrenir. Bu davranıştan bir genelleştirme çıkararak test örneklerine çözüm üretir. Donanımla (dijital, analog, optik) ve yazılımla (Matlab, C, Pspice, ...) gerçeklenebilirler. Bir bilgisayarla beyni karşılaştırarak sinir ağlarının önemini daha iyi kavrayabiliriz. Bilgisayarlar seri çalışır ve hataya tolerans yoktur. Giriş verilerine duyarlıdır yani bir adımdaki yanlış hesaplama sonucu çok değiştirir. Ayrıca tam tanımlı komut dizini gereklidir. Dijital olarak çalışır ve hızlıdır (10^6 işlem/sn). Beyin ise paralel çalışır, adaptiftir ve hataya toleranslıdır. Giriş verilerine duyarsızdır yani birimlerin yapılan ana işe katkıları azdır. Analog olarak çalışır ve her işlem için geçerli olmasa da yavaştır (10 işlem/sn). (Yıldırım, 2002)

3.1 Biyolojik Nöron

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. Şekil 3.1'de bir biyolojik nöron modeli görülmektedir.



Şekil 3.1 Biyolojik nöron modeli (Yıldırım, 2002)

Merkezi sinir sisteminin temel işlem elemanı, sinir hücresidir (nöron) ve insan beyninde yaklaşık 10 milyar sinir hücresi olduğu tahmin edilmektedir. Sinir hücresi; hücre gövdesi, dendritler ve aksonlar olmak üzere 3 bileşenden meydana gelir. Dendritler, diğer hücrelerden

aldığı bilgileri hücre gövdesine bir ağaç yapısı şeklinde ince yollarla iletir. Aksonlar ise elektriksel darbeler şeklindeki bilgiyi hücreden dışarı taşıyan daha uzun bir yoldur. Aksonların bitimi, ince yollara ayrılabilir ve bu yollar, diğer hücreler için dendritleri oluşturur. (http://www.batitrakya.dostweb.com)

3.2 Yapay Sinir Ağlarının Tarihsel Gelişimi

Yapay sinir ağlarının esas gelişimi 1940'lı yıllardan itibaren başlamıştır. 1943'de McCulloch-Pitts modeli geliştirilmiştir. En temel ve hala geçerli olan bir modeldir. 1949 yılında Hebb sinaps kavramını gerçeklemiş, fizyolojik modellemeyi incelemiş ve kendine yönelik bir öğrenme geliştirmiştir. 1954'de Minsky tarafından ilk nöro bilgisayarlar gerçekleştirilmiştir. 1958'de Rosenblatt tarafından ilk ağ yapısı gerçeklenmiş, 1969 yılında Minsky ve Papert tek katmanlı ağ yapısıyla XOR gerçeklenemediğini bularak çok katmanlı ağları geliştirmesi için bir adım atmışlardır. 1961'de Steinbuch tarafından çağrışımlı bellek geliştirilmiş. 1973'de Vander Malsberg kendi kendini örgütleyen bir gerçekleme yapmıştır. 1974-1982 arasında Grossberg ve Carpenter yarışmalı öğrenme metodunu geliştirerek ART Ağları ve FuzzyART Ağlarını bulmuşlardır. 1982'de Hopfield enerji fonksiyonu kavramından yararlanarak öğrenmeyi sağlamış. 1984-1989 yılları arasında Kohonen tarafından SOM algoritması geliştirilmiş. 1986'da McClelland-Rumelhart hatanın geriye yayılımını bulmuş ve geliştirmiş. (Yıldırım, 2002) Bu tarihten günümüze kadar geliştirilen algoritmalarla bir çok sınıflama ve örnek tanıma problemi çözümlenmiştir. Ayrıca son zamanlarda bu algoritmaların birlikte kullanıldığı hibrid yapılar geliştirilmiş ve uygulamalarda sıklıkla kullanılmaya başlanmıştır.

3.3 Yapay Sinir Ağlarının Özellikleri ve Kullanıldığı Yerler

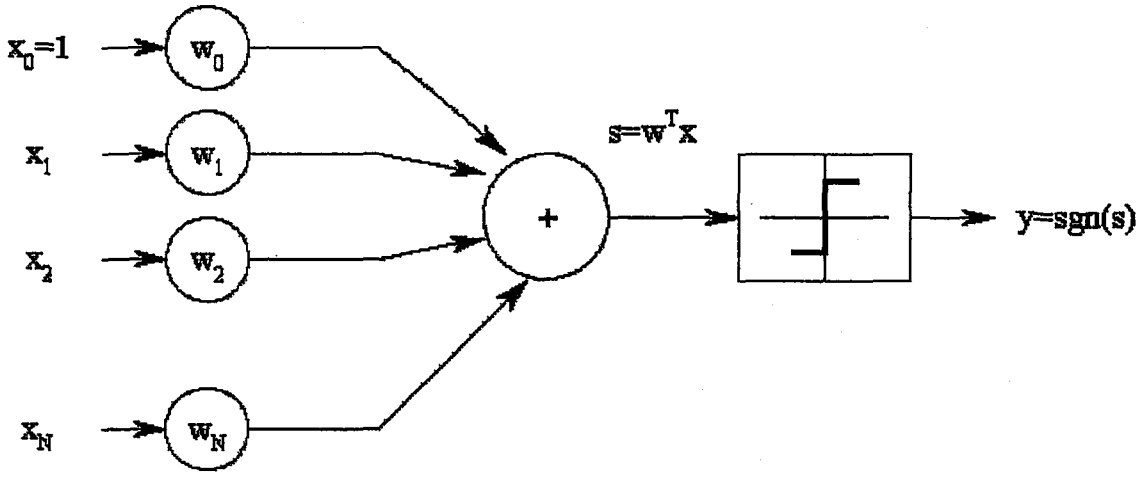
Nöral sistemler eğitilebilir sistemlerdir. Sistem değişen koşullara uyar yani adaptiftir ayrıca lineer değildir. Nöral Sistemlerin en genel uygulama alanı örnek tanıma üzerinedir. Bunun yanısıra Görüntü İşleme ve İşaret İşleme alanlarında da kullanılır. Askeri Sistemlerde radar işaretleri ve denizaltında mayın bulmak, silahların otomasyonu ve hedef izleme, nesneleri/görüntüleri ayırma ve tanıma, yeni algılayıcı tasarımı ve gürültü önleme v.s gibi olaylarda kullanılmaktadır. Tıpta hastalık teşhisinde, işaretlerin incelenmesinde (EKG-kalbin elektriksel işareti, EMG-kasların elektriksel işareti, EEG-beynin elektriksel işareti v.b. işaretler) ve tıbbi görüntülerin incelenmesinde (tomografi, MR, ultrason v.b. görüntüler) kullanılır. Diğer kullanım yerleri: Finansal Sistemler, Kontrol (Uçaklarda otomatik pilot sistemi otomasyonu, ulaşım araçlarında otomatik yol bulma/gösterme, robot sistemlerin

kontrolü, doğrusal olmayan sistem modelleme ve kontrolü, elektrikli sürücü sistemlerin kontrolü v.s.), Güç Sistemleri ve Güvenlik alanlarıdır.

3.4 Yapay Sinir Modeli

İnsan beyninin 10 milyar sinir hücresinden ve 60 trilyon sinaps 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 son derecede 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. (http://www.batitrakya.dostweb.com) Bu yapısından dolayı beynin üstün özellikleri üzerinde çalışılmış ve nörofiziksel yapısından esinlenerek matematiksel modeli çıkarılmaya çalışılmıştır.

Bu modele değinmeden önce bazı temel kavramları açıklamalıyız. İşlem elemanı "nöronlar"dır. Her bir giriş için bir "aktivasyon durumu" ve birimler arasında W_i ile gösterilen "bağlantılar" var. Bir birimin dış girişleri arasında hangisinin etkili olduğunu gösteren bir propogasyon kuralı vardır. Bu kural ile etkili giriş belirlenir. y ile gösterilen bir aktivasyon fonksiyonu ile o andaki etkili giriş ve propogasyona göre nöron cevap verir ya da vermez. Buna göre bir nöron Şekil 3.2'deki gibi modellenebilir.



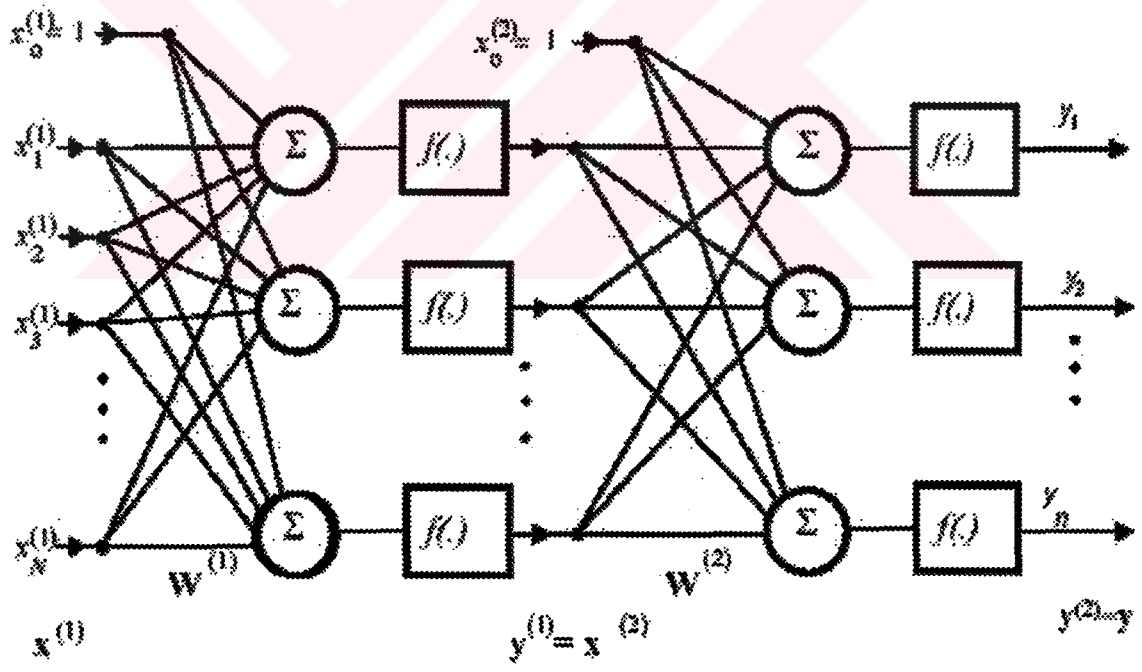
Şekil 3.2 İşlem elemanının modeli (Haykin, 1994)

3.5 Kullanılan Ağ Yapıları

Bu tezde Çok Katmanlı Algılayıcılar, Radyal Temelli Fonksiyonlar ve Konik Kesit Fonksiyonlu Ağlar olmak üzere 3 farklı ağ yapısı ile eğitme yapılmıştır.

3.5.1 Çok Katmanlı Algılayıcılar (Multilayer Perceptron - MLP)

En genel yapısıyla MLP modeli, bir giriş katmanı, bir veya daha fazla ara katman, bir de çıkış katmanından oluşur. Bilgi akışı daima ileri doğrudur. Bu yüzden ileri beslemeli ağ modeli de denir. Giriş katmanındaki nöron sayısı problemdeki giriş sayısına bağlıdır. Ara katman sayısı ve ara katmanlardaki nöron sayısı deneme-yanılma yolu ile bulunur. Çıkış katmanındaki nöron sayısı ise yine uygulanan probleme dayanılarak belirlenir. Pratikte 3 katmandan fazla katman kullanılmaz. Katman sayısına giriş katmanı dahil değildir. Örneğin: 1 tane gizli katman varsa 2 katmanlı ağ, 2 tane gizli katman varsa 3 katmanlı ağ denir. Ara katmandaki giriş-çıkış ilişkileri hakkında bilgi sahibi değiliz. Ara katmanlarda non-lineer çıkış katmanında ise lineer fonksiyonlar vardır. 2 katmanlı MLP yapısı Şekil 3.3'de görülmektedir.



Şekil 3.3 İki katmanlı MLP yapısı

3.5.2 Radyal Temelli Fonksiyonlar (Radial Basis Function – RBF)

Çok katmanlı YSA'nın tasarımında eğiticili geriye yayılım öğrenme algoritması bir en iyileme uygulamasıdır. Radyal temelli fonksiyon ağı tasarımı ise çok boyutlu uzayda eğri

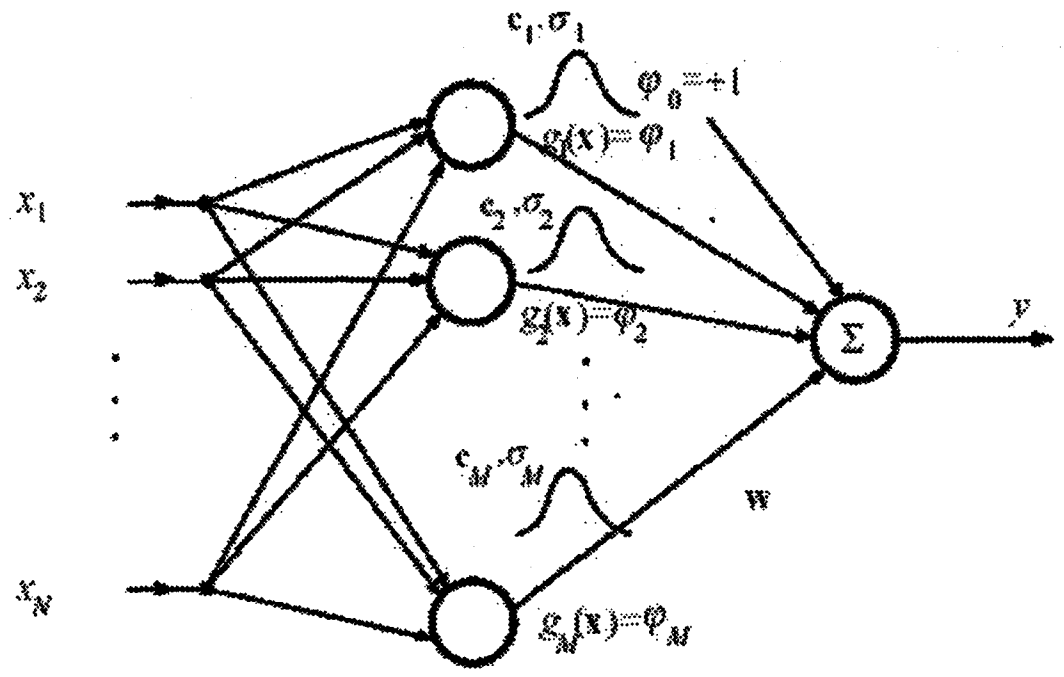
uydurma yaklaşımıdır ve bu nedenle RBF'nin eğitimi, çok boyutlu uzayda eğitim verilerine en uygun bir yüzeyi bulma problemine dönüşür. RBF'nin genellemesi ise eğitim sırasında bulunan çok boyutlu yüzeyin kullanılmasına eşdeğerdir. Radyal temelli fonksiyonlar, sayısal analizde çok değişkenli problemlerin çözümünde kullanılmış ve YSA'nın gelişmesi ile birlikte bu fonksiyonlardan YSA tasarımında yararlanılmıştır. RBF, ileri beslemeli YSA yapılarına benzer şekilde giriş, saklı ve çıkış katmanından oluşur ancak, giriş katmanından saklı katmana dönüşüm, radyal tabanlı aktivasyon fonksiyonları ile doğrusal olmayan sabit bir dönüşümdür. Saklı katmandan çıkış katmanına ise doğrusal bir dönüşüm gerçekleştirilir. Şekil 3.4'de en genel gösterimiyle bir RBF yapısı verilmiştir. RBF'de uyarlanabilecek serbest parametreler; merkez vektörleri, radyal fonksiyonların genişliği ve çıkış katman ağırlıklarıdır. Çıkış katmanı doğrusal olduğundan ağırlıklar, eğim düşme ya da doğrusal en iyileme yöntemleri ile kolayca bulunabilir. Merkezler, girişler arasından rastgele ve sabit olarak seçilebilmekle birlikte RBF'nin performansını iyileştirmek amacıyla merkez vektörlerinin ve genişliğin uyarlanması için çeşitli yöntemler geliştirilmiştir. Merkez vektörleri, eğim düşme yöntemine göre eğiticili öğrenme algoritması ile uyarlanarak, dik en küçük kareler yöntemi ile ya da kendiliğinden düzenlemeli yöntemle giriş örneklerinden öbekleme yapılarak belirlenebilir. (Coskun ve Yildirim, 2003) RBF'nin genel matematiksel ifadesi aşağıda verilmiştir.

$$G(x) = \sum_{i=1}^{N} W_{ij}\varphi_i(x) \tag{3.1}$$

Burada W_{ij} gizli katmandaki i. nörondan çıkış katmanındaki j. nöron arasındaki ağırlığı, $\varphi_i(x)$ ise aktivasyon fonksiyonunu gösterir.

RBF'de gizli katman aktivasyon fonksiyonu olarak genellikle Gauss Fonksiyonu kullanılır. Gauss Fonksiyonu x giriş vektörünü, c_i merkezi, $\|x - c_i\|$ standart öklid uzaklığını, σ_i de genişliği göstermek üzere aşağıdaki şekilde ifade edilir.

$$\varphi(r) = \exp\left(-\frac{\|x - c_i\|^2}{2\sigma_i^2}\right) \tag{3.2}$$



Şekil 3.4 RBF modeli (Haykin, 1994)

3.5.3 Konik Kesit Fonksiyonlu Ağlar (Conic Section Function Neural Network-CSFNN)

Konik Kesit Fonksiyonlu Ağların anafikri, MLP ve RBF ağlarının birarada kullanılmasıdır. Yeni yayılım kuralı, (MLP ve RBF yayılım kurallarını içeriyor) bir koninin analitik denklemleri kullanılarak elde edilebilir. Şekil 3.5 bu ağların yapısını göstermektedir. Burada x, sağ dairesel koninin üzerinde herhangi bir nokta, ω [-π/2,π/2] aralığında herhangi bir değer, *v* koninin tepe noktası ve *a* koninin eksenini tanımlayan birim vektör olsun. Böylece dairesel koninin denklemi

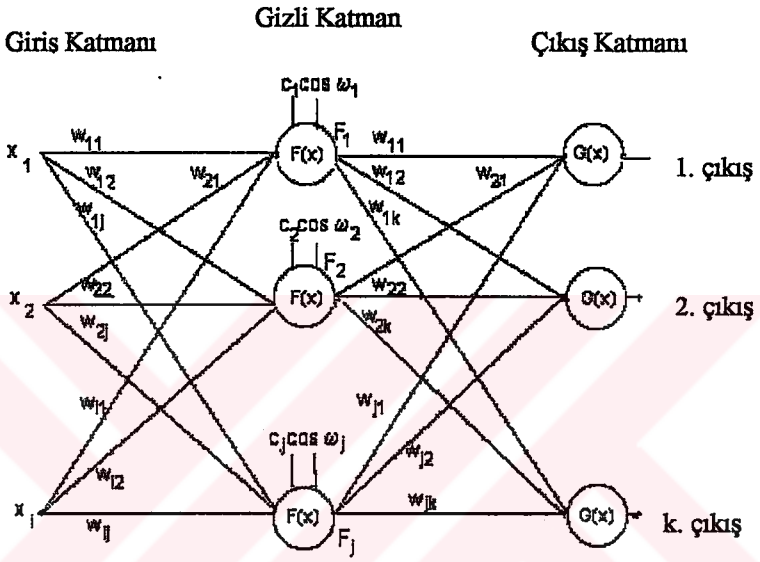
$$(\vec{x} - \vec{v})\vec{a} = \cos\omega \left\| (\vec{x} - \vec{v}) \right\| \tag{3.3}$$

olmaktadır. İki boyutlu uzay için vektörler ve koordinatları $x=(x_1,x_2)$, $v=(v_1,v_2)$ ve $a=(a_1,a_2)$ ise 3.3 denklemi aşağıdaki gibi yazılabilir.

$$(x_1 - v_1)a_1 + (x_2 - v_2)a_2 = \cos\omega\sqrt{(x_1 - v_1)^2 + (x_1 - v_2)^2} \tag{3.4}$$

Konik Kesit Fonksiyonlu Ağların yayılım kuralı 3.4 denklemi ile tanımlanır. Bütün bunları n-boyutlu giriş uzayı için genellersek 3.5 denklemini elde ederiz.

$$\sum_{i=1}^{n+1}(x_i - v_i)a_i = \cos\omega\sqrt{\sum_{i=1}^{n+1}(x_i - v_i)^2} \tag{3.5}$$



Şekil 3.5 Konik Kesit Fonksiyonlu Ağ modeli

Açılma açısı (2ω) 90 derece olduğunda x noktaları ve v arasındaki uzaklık dairenin yarıçapına eşit olduğundan dairenin merkez koordinatı c, v'nin yerine kullanılabilir. 3.5 denkleminin sağ tarafı sol tarafından çıkarıldığında Konik Kesit Fonksiyonlu Ağların yayılım kuralı aşağıdaki gibi elde edilir.

$$y_j = \sum_{i=1}^{n+1}(x_i - c_{ij})a_{ij} - \cos\omega_j\sqrt{\sum_{i=1}^{n+1}(x_i - c_{ij})^2} \tag{3.6}$$

burada a_{ij} MLP'deki giriş ile gizli katman arasındaki ağırlıkları, c_{ij} RBF'deki merkez koordinatlarını, i ve j giriş ve gizli katmanı gösteren indislerdir. Bu durumda y_j Konik Kesit Fonksiyonlu Ağların aktivasyon değeridir. Kolayca görüldüğü üzere bu denklem MLP ve

RBF gibi 2 temel kısımdan oluşur. ω açısı $\pi/2$ olduğu zaman denklem sadece MLP kısmına dönüşür. Denklemin ikinci kısmı da RBF'nin girişleri ile merkezleri arasındaki öklid uzaklığıdır.

4. İMZA TANIMA VE DOĞRULAMA PROBLEMİ

İmza kişiye ait özel bir el yazısı türüdür. Herkesin imzası farklı olduğu gibi, aynı kişinin farklı zamanlarda attığı imzalar da farklı olabilir. Bu tezde 8 farklı kişinin imzası için imza tanıma işlemi yapılmıştır. Bu kişilerden alınan imzalar Şekil 4.1'de verilmiştir. Ayrıca her kişi için 2'şer taklit (sahte) imza atarak imza doğrulama işlemi de gerçekleştirilmiştir. Sahte imzalar da Şekil 4.2'deki gibidir.

4.1 Önişlemler

Görüntü işleme aşamasında bir A4 sayfası 16'ya (8*2) bölündü. Kişilerden siyah veya mavi mürekkepli kalemle çerçevelerin dışına taşırmadan imza atmaları istendi. Kurşun kalem kullanılarak atılan imza iskelet çıkarma aşamasında kötü sonuç verdiği için (imzada kesintiye sebep olup, parçalı imza görüntüsündeydi) mürekkepli kalem kullanıldı. 8 kişiden toplanan 32'şer imza tarayıcı kullanılarak 300dpi'de siyah beyaz olarak tarandı. Taranan 32 imzanın 25 tanesi eğitmede 7 tanesi testte kullanıldı.

Gürültü indirgeme aşamasında beyaz arka plan beyaz üzerindeki siyah pikseller Bölüm 2.3.2'de anlatılan yöntemle yokedildi.

İmza resmindeki gürültülerin çıkarılmasından sonra imza alanının kesilip alınması işlemi için yatay ve düşey izdüşümlü parçalama metodu kullanılarak imza alanı arka plandan kesilip alındı.

Genişlik normalizasyonu için kabul edilen değere* ulaşması için, genişlik-yükseklik oranının değişmemesi şartıyla resmin boyutu ayarlandı.

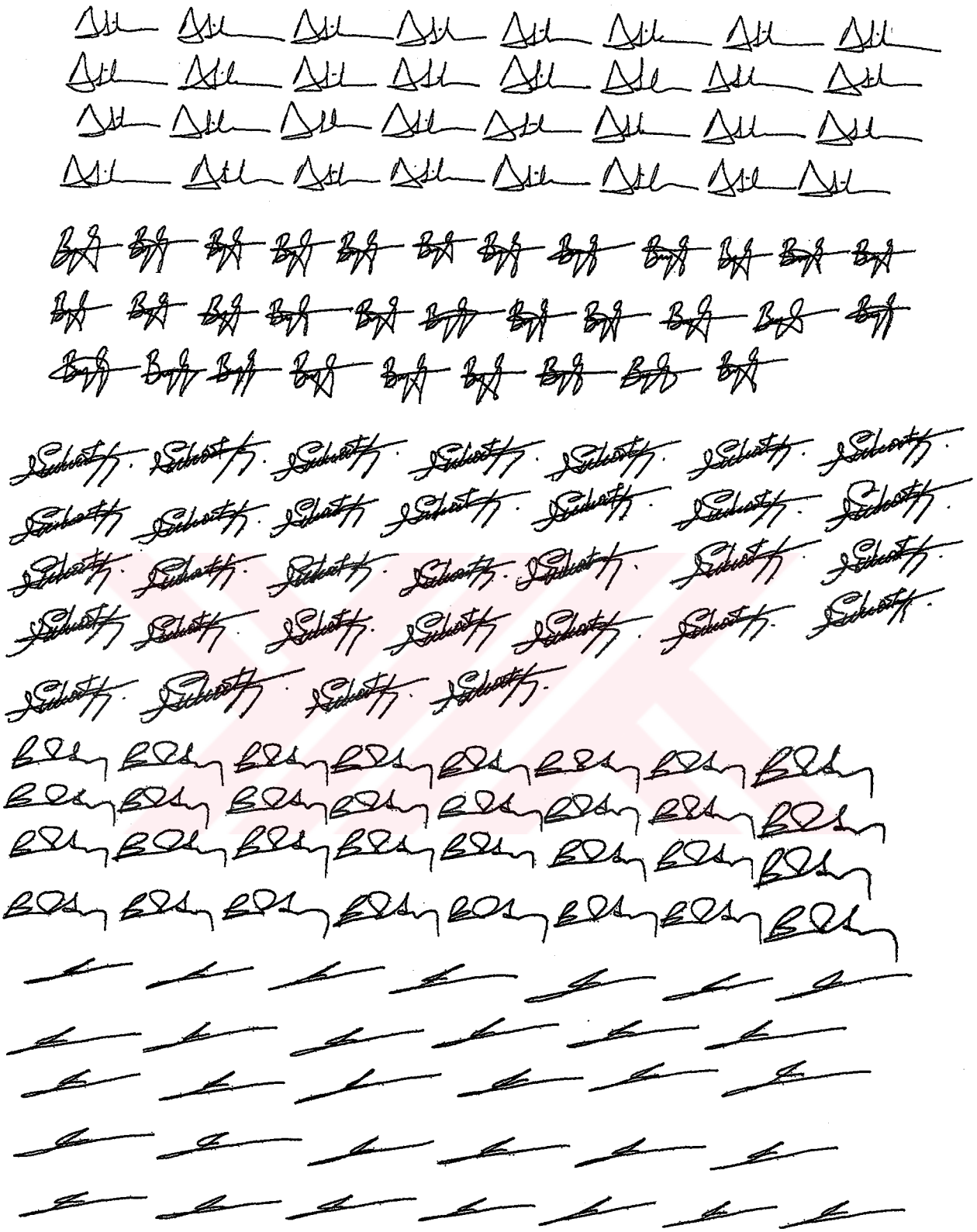
İskeleti oluşturma aşaması için Lam ve Suen (1991) tarafından tanımlanmış iskelet oluşturma tekniklerinin basitleştirilmiş versiyonu kullanıldı. Basitleştirilmiş algoritma aşağıdaki 3 adımı içermektedir.

Adım 1: Atılmaya aday bütün noktaları işaretle. (8 komşuluğunda en az 1 beyaz ve en az 2 siyah piksel olan siyah pikseller)

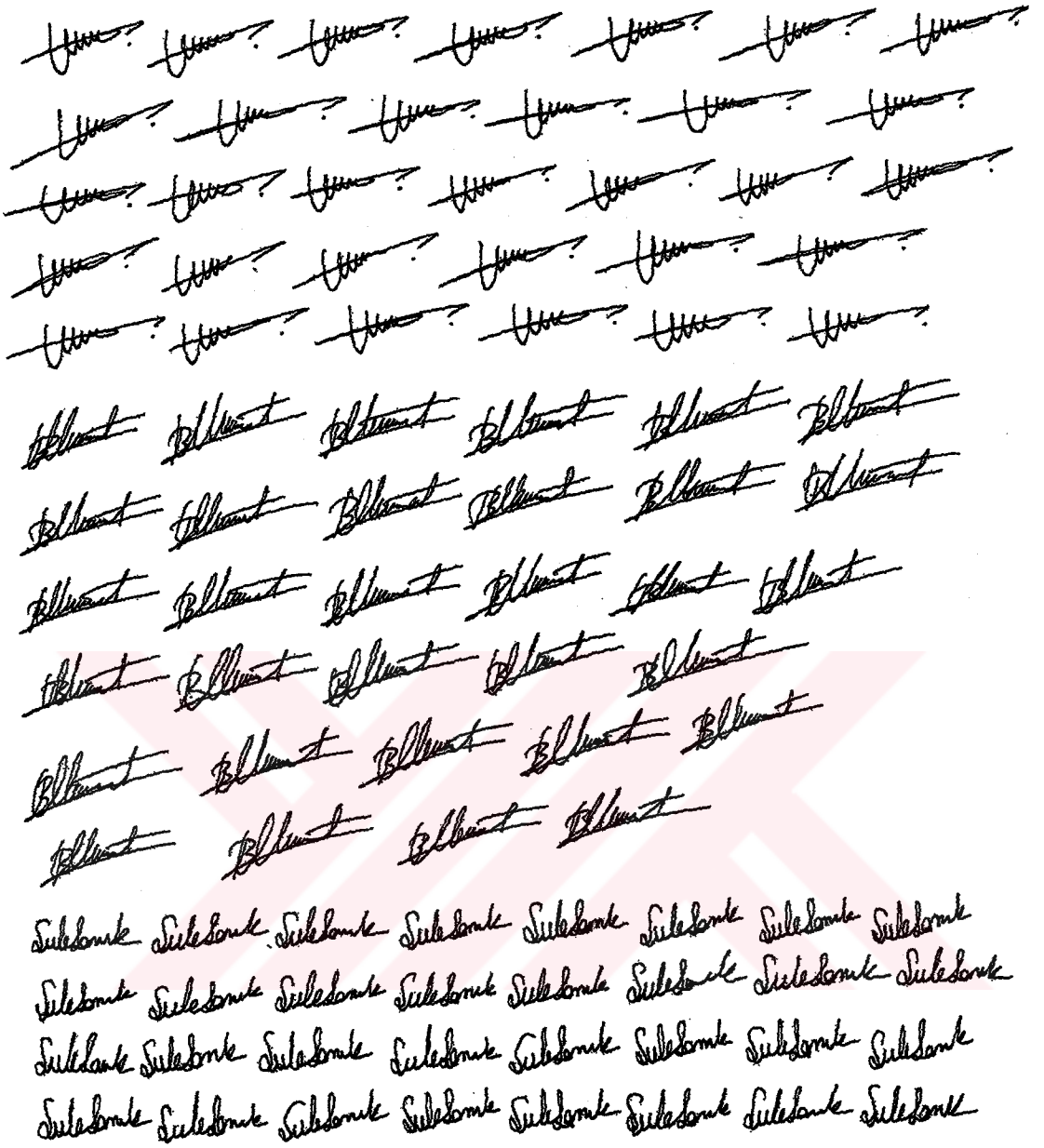
Adım 2: İmza eğrisini takip ederek, tüm pikselleri sına ve imzada kesintiye sebep olmayacak durumdaki pikselleri at.

---

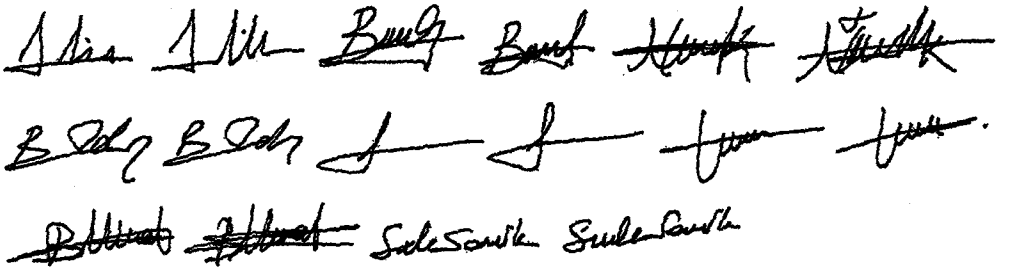
* Tezde bu değer 400 kabul edildi



Şekil 4.1a Gerçek imzalar



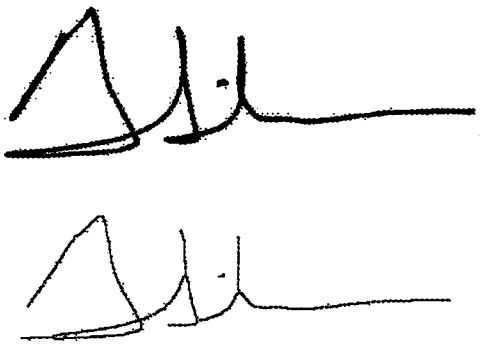
Şekil 4.1b Gerçek imzalar



Şekil 4.2 Sahte imzalar

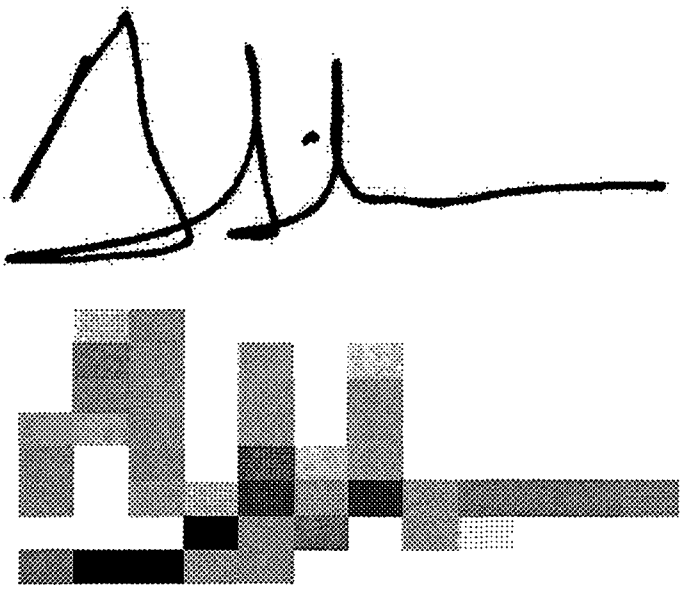
Adım 3: Eğer en azından 1 piksel silinmiş ise Adım 1'e git ve işlemi bir kez daha tekrarla.

Bu algoritmanın bütün imzalara uygulanması sonucunda imzaların iskeletleri çıkarılmış olur. Şekil 4.3'de iskeleti çıkarılmış bir imza görülmektedir.



Şekil 4.3 Bir imza örneği ve bu imzanın iskeleti

Bu tezde genel özellik özellik olarak imzanın yatay merkezi, imzanın düşey merkezi ve taban çizgisi kayması özellikleri kullanılmıştır. Bu üç özellikten 3 tane değer elde edilmiştir. Ve bu üç değer yapay sinir ağının girişine uygulanır. Grid bilgisi ise bize 8x12'lik bir matris verir. Bu matris 96x1'lik bir vektör haline dönüştürülerek YSA'nın girişine uygulanır. Böylece toplam 99 (96+3) tane giriş oluşturulmuştur. Şekil 4.4'de bir imza ve bu imzanın gridi verilmiştir.

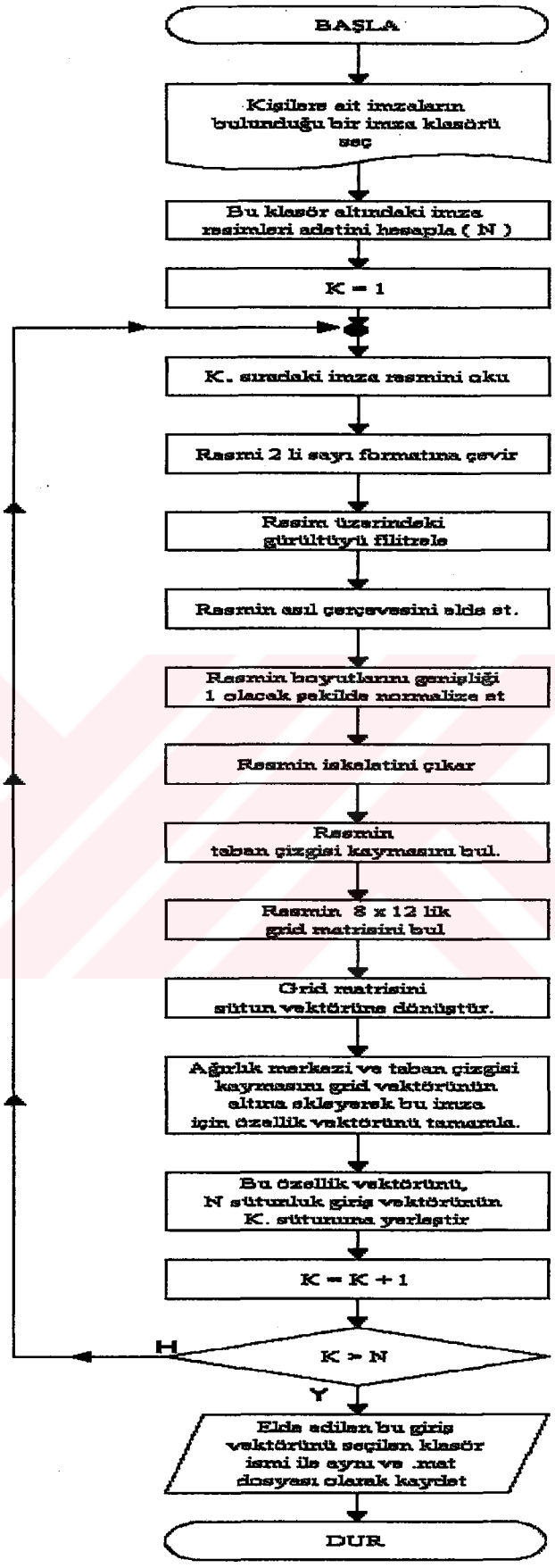


Şekil 4.4 Bir imza örneği ve bu imzanın gridi

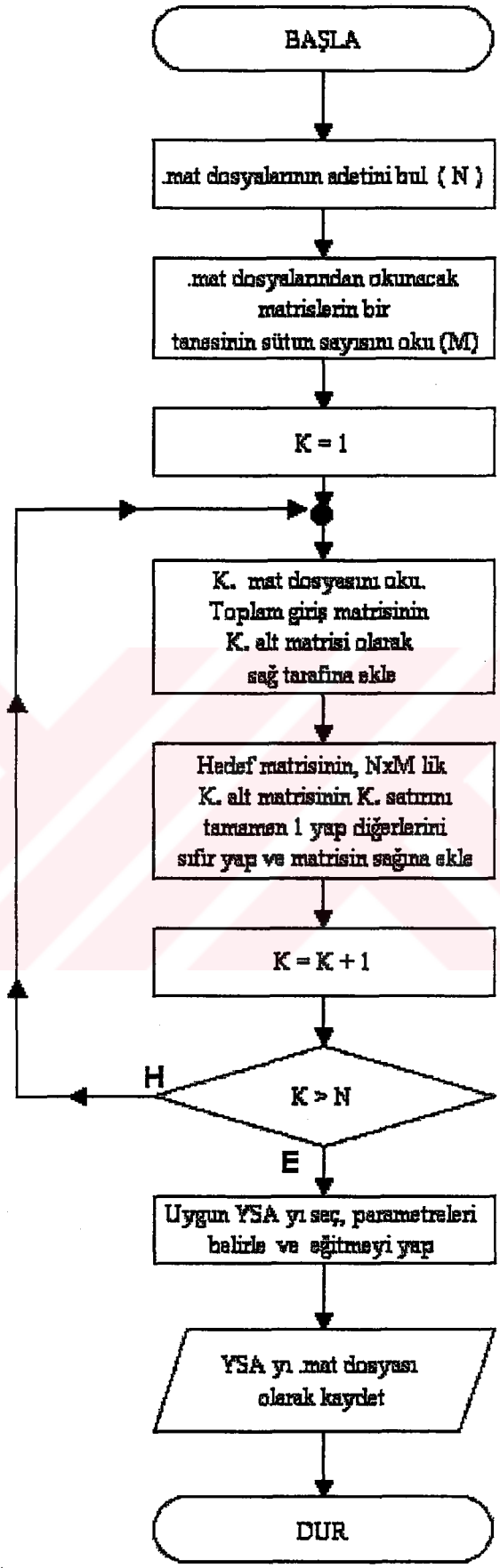
Özellik çıkarma işleminin sonunda 99 x 200 boyutunda bir eğitme girişi elde edilmiştir. Giriş vektörünün boyutu çok yüksek olduğu için temel bileşen analizi (PCA) yöntemi kullanılarak giriş vektörünün boyutu azaltıldı ve 20 x 200'e düşürüldü. Sonuçlar ayrı ayrı verildi. Oluşturulan YSA imza tanıma için 56 (7*8), imza doğrulama için 16(2*8) imza ile test edildi. MLP ve RBF yazılımında kullanılan giriş vektörünün oluşturulması, eğitme ve tanımaya ait işaret akış diyagramları Şekil 4.5'de verilmiştir. CSFNN için oluşturulan işaret akış diyagramları ise Şekil 4.6'dadır.

4.2 Temel Bileşen Analizi (PCA)

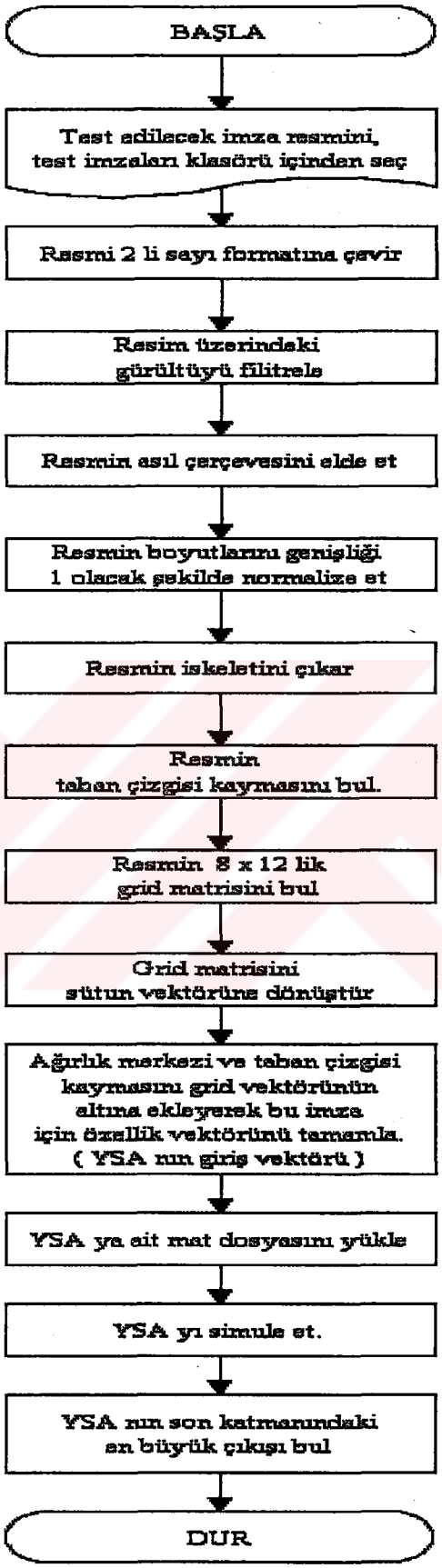
Temel Bileşen Analizi (PCA), örnek tanıma ve görüntü sıkıştırma gibi yüksek boyutlu veri kümelerinde boyutu azaltmak için sıklıkla kullanılır. Ortalama, değişinti matrisi ve bu matrisin özvektör ve özdeğerlerinin hesabına dayanan istatistiksel bir yöntemdir. Bu tezde incelenen imza tanıma ve doğrulama probleminde de giriş vektörünün boyutunu azaltma ihtiyacı duyulmuş ve temel bileşen analizi yöntemi kullanılarak eğitme yapılmıştır. Sonuçlar Tablo 4.3'de gösterilmiştir.



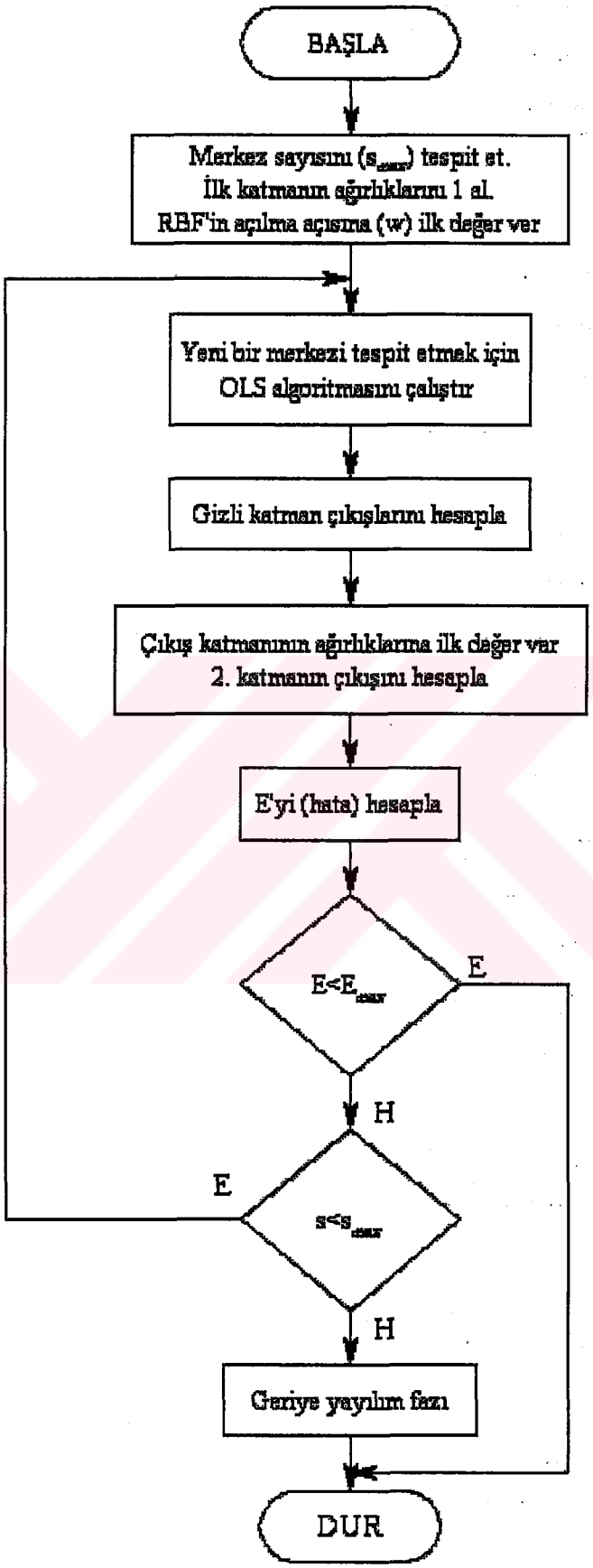
Şekil 4.5a Giriş vektörü oluşturmaya ait işaret akış diyagramı



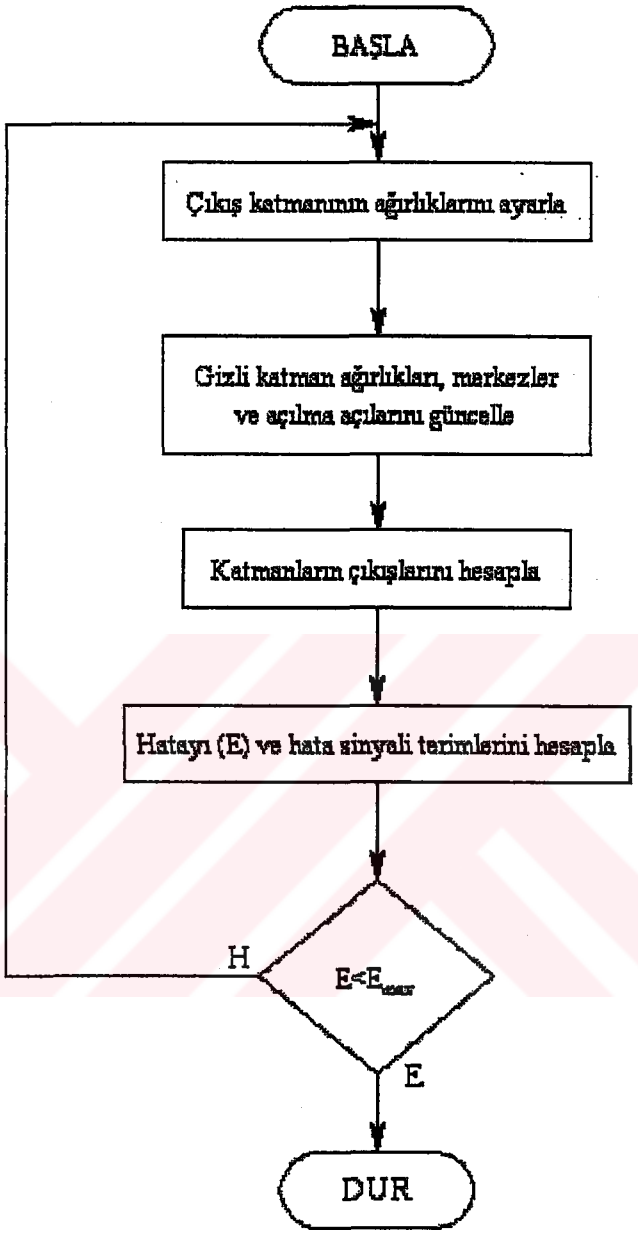
Şekil 4.5b Eğitmeye ait işaret akış diyagramı



Şekil 4.5c Tanımaya ait işaret akış diyagramı



Şekil 4.6a CSFNN ile eğitmeye ait işaret akış diyagramı



Şekil 4.6b Geriye yayılım algoritmasına ait işaret akış diyagramı

4.3 Simülasyon Sonuçları

Simülasyonlar için Matlab 6.5 Neural Network Toolbox'ı kullanılmıştır. Yazılım Ekler kısmında sunulmuştur. Sonuçlar MLP, RBF ve CSFNN yöntemlerini kullanarak tanıma ve doğrulama için ayrı ayrı tablolar halinde gösterilmiş ve başarı yüzdeleri verilmiştir.

4.3.1 İmza Tanıma

MLP ile yapılan eğitmede 1.gizli katmanda 60, 2.gizli katmanda 24 ve çıkış katmanında 8 nörondan oluşan bir ağ yapısı tasarlanmıştır. Momentum ve adaptif öğrenme oranının ayarlanabildiği geriye yayılım algoritması kullanılmıştır. Eğitmede öğrenme oranı 0.05 ve momentum 0.09 seçilerek en iyi sonuçlar alınmıştır. Ağırlıklar rastgele seçildiği için ağ 10 defa eğitilerek sonuçların ortalaması alınmıştır. Egitme yaklaşık olarak 200 adımda tamamlanmıştır.

RBF ile yapılan eğitmeden sonra gizli katmanda 175 nörona ulaşılmıştır. Genişlik değeri olarak 10 seçilmiştir.

CSFNN ile yapılan eğitmede ise ağ, 12 nörondan oluşan 1 gizli katman ve 8 nörondan oluşan çıkış katmanı olacak şekilde tasarlanmıştır. Genişlik değeri olarak 1 seçilmiştir.

4.3.1.1 Normal SR

MLP ve RBF ile eğitme işlemi gerçekleştirilmiştir. Sonuçlar Tablo 4.1'de gösterilmiştir. Ancak giriş vektörünün boyutu çok yüksek olduğu için CSFNN ile eğitme ve test başarıyla gerçekleştirilememiştir.

Bu tablolardan da görüldüğü gibi, MLP ve RBF ile yapılan eğitmede eğitme imzalarının %100 doğru sınıflanmasına rağmen CSFNN ile ancak %25'i sınıflanabilmiştir. Test imzalarında ağın başarısı MLP için %98,0375, RBF için %100, CSFNN için %0'dır.

Çizelge 4.1 SR için sınıflanan eğitme imzaları sayısı

Yöntem	Eğitme İmzaları								Başarı (%)
	A	B	C	D	E	F	G	H	
MLP(ort.)	25	25	25	25	25	25	25	25	100
RBF	25	25	25	25	25	25	25	25	100
CSFNN	25	0	0	25	0	0	0	0	25

Çizelge 4.2 SR için sınıflanan test imzaları sayısı

Yöntem	Test İmzaları								Başarı (%)
	A	B	C	D	E	F	G	H	
MLP(ort.)	7	7	6.7	6.3	7	7	7	6.9	98.0375
RBF	7	7	7	7	7	7	7	7	100
CSFNN	0	0	0	0	0	0	0	0	0

4.3.1.2 PCA Kullanılarak SR

Yukarıdaki tablolardan da görüldüğü gibi giriş vektörünün boyutu çok yüksek olduğu için ağın performansı CSFNN için oldukça kötüdür. Bunu önlemek için temel bileşen analizi (PCA) kullanarak giriş vektörünün boyutu azaltılmış ve 99x200'den 20x200'e düşürülmüştür. Yeni boyuta göre yapılan eğitmede bulunan sonuçlar Tablo 4.3 ve 4.4'de sunulmuştur.

Bu tablolardan da görüldüğü gibi eğitme imzalarının doğru sınıflanma yüzdeleri MLP ile yapılan eğitmede %87.4, RBF de %100, CSFNN'de %96'dır. Test imzalarında ağın başarısı MLP için %92.143, RBF için %98,21 ve CSFNN için %96,43'dür.

Çizelge 4.3 SR için PCA kullanılarak sınıflanan eğitme imzaları sayısı

Yöntem	Eğitme İmzaları								Başarı (%)
	A	B	C	D	E	F	G	H	
MLP(ort.)	25	25	16,7	25	20,8	12,5	25	24,8	87.4
RBF	25	25	25	25	25	25	25	25	100
CSFNN	21	25	25	25	23	23	25	25	96.43

Çizelge 4.4 SR için PCA kullanılarak sınıflanan test imzaları sayısı

Yöntem	Test İmzaları								Başarı (%)
	A	B	C	D	E	F	G	H	
MLP(ort.)	7	6,3	5,6	6.3	6,6	7	6,4	6,4	92.143
RBF	7	7	7	7	7	7	7	6	98.21
CSFNN	7	7	7	7	6	7	6	7	96.43

4.3.2 İmza Doğrulama

İmza tanıma için veri kümesine 8 kişiye ait 2'şer tane (toplam 16) taklit imza eklenmiş ve bu imzaların tanınıp tanınmadığı araştırılmıştır. Tablolarda taklit edilen 2'şer imzanın tanınıp tanınmadığı görülmektedir. Bu tablolarda ağ tarafından, taklit edilen imzanın, gerçek sahibine ait olduğu söylenmişse 0 (ağ tarafından doğru sınıflanamamış), bu imza tanınamamışsa 1 (taklit imza ağ tarafından doğru sınıflanmış) alınmıştır.

4.3.2.1 Normal SV

Çizelge 4.5 SV için tanınamayan taklit imza sayısı

Yöntem	Test İmzaları								Başarı (%)
	A	B	C	D	E	F	G	H	
MLP(ort.)	1	1	2	0	1	0	0	0	31.25
RBF	0	0	2	0	0	0	1	0	18.75
CSFNN	0	2	2	0	1	0	1	0	37.5

Tablolardan da görüldüğü gibi SV için ağın başarı yüzdeleri; MLP için %31,25 , RBF için %18,75 ve CSFNN için %37,5'dir.

4.3.2.2 PCA Yöntemiyle SV

Çizelge 4.6 SV için PCA kullanılarak tanınamayan taklit imza sayısı

Yöntem	Test İmzaları								Başarı (%)
	A	B	C	D	E	F	G	H	
MLP(ort.)	0	1	2	0	0	2	0	0	31.25
RBF	0	1	0	0	0	0	0	1	12.5
CSFNN	2	2	1	2	0	0	2	0	56.25

Tablolardan da görüldüğü gibi SV için ağın başarı yüzdeleri; MLP için %31,25 , RBF için %12,5 ve CSFNN için %56,25 dür.

5. SONUÇLAR

Bu tezde 8 kişiden toplanan imzalar kullanılarak imza tanıma ve doğrulama işlemi gerçekleştirilmiştir.

İmza tanıma ve doğrulama için MLP, RBF ve CSFNN ile normal ve PCA yöntemi kullanılarak eğitme yapılmış ve eğitme ve test örneklerinin sınıflanma sayıları ve ağların başarı yüzdeleri verilmiştir. Bölüm 4'deki tablolardan görüldüğü gibi imza doğrulama işleminin başarı yüzdeleri çok düşüktür. Teze başlarken amaç imza tanıma işlemi yapmaktı. Buna daha sonra deneme amaçlı imza doğrulama eklenmiştir. Fakat yazılan programda kullanılan özellikler imza tanıma için yeterli olmasına rağmen, imza doğrulama için yeterli olmamıştır. Dolayısıyla uğraşılarak atılan iyi bir taklit imzayı ağ tanıyamamıştır. Bunun için daha ayırt edici özellikler (kapalı döngü sayısı, imzanın açısı, köşe sayısı vb.) çıkarılması gerekmektedir.

Tezin asıl amacı olan imza tanıma kısmında MLP ve RBF ile yapılan eğitme ve RBF ile yapılan test başarı yüzdelerinin %100 bulunması bakımından önemlidir. Ayrıca dikkat çekici sonuçlardan biri de CSFNN ile yapılan eğitmede kullanılan nöron sayısının azlığıdır. CSFNN ile 20 nöron kullanılırken MLP'de 92, RBF'de ise 183 nöron kullanılmıştır ki bu da pratikte uygulanması açısından CSFNN'lerin üstün olduğunun ispatıdır.

KAYNAKLAR

Bajaj, R., Chaudhury, S., (1996), "Signature Verification Using Multiple Neural Classifiers", Pattern Recognition Society, 30:1-7.

Baltzakis, H., Papamarkos, N., (2001), "A new signature verification technique based on a two-stage neural network classifier", Engineering Applications of Artificial Inteligence, 14:95-103.

Coskun, N., Yıldırım, T. (2003), "Yapay Sinir Ağları İle Hepatit Hastalığı Teşhisi", Boğaziçi Üniversitesi Biyomedikal Mühendisliği Ulusal Toplantısı, 29-30 Mayıs 2003, İstanbul.

Çoban, Y., Kapanoğlu, B., Yıldırım, T., (2003), "Standart Öklid Mesafesi Hesaplayan Bir CMOS Analog Devrenin RBF Ağlarına Uyarlanması", Elektrik-Elektronik-Bilgisayar Mühendisliği 10. Ulusal Kongresi, 18-21 Eylül, 2003, İstanbul.

Hanmandlu, M., Madasu, V.K., Madasu, S., (2003), "Neuro-Fuzzy Approaches to Signature Verification", 2nd National Conference on Document Analysis and Recognition (NCDAR-2003), 11-12 July 2003, Mandya, India.

Haykin, S., (1994), Neural Networks: A Comprehensive Foundation, Macmillan College Publishing, New York.

Huang, K., Yan, H., (1996), "Off-line Signature Verification Based on Geometric Feature Extraction and Neural Network Classification", Pattern Recognition Society, 30:9-17.

Keit, T.H., Palaniappan, R., Raveendran, P., Takeda, F., (2001), "Signature Verification System using Pen Pressure for Internet and E-Commerce Application", ISSRE2001 International Symposium on Software Reliability Engineering, 27-30 November, Hong Kong.

Lam, L., Suen, C.Y., (1991), "A dynamic shape preserving thinning algorithm", Signal Processing, 22:199-208

Murshed, N.A. Bortolozzi, F., Sabourin, R., (1995), "Off-Line Signature Verification using Fuzzy ARTMAP Neural Network", IEEE International Conference on Neural Networks, 27 November-1st December 1995, Perth, Australia.

Öz, C., Ercal, F., Demir, Z., (2003), "Signature Recognition and Verification with ANN", ELECO'2003 International Conference on Electrical and Electronics Engineering, 3-7 December 2003, Bursa.

Yıldırım, T., (2002), "Nöron Ağları ve Uygulamaları Ders Notları"

Yıldırım, T., (1997), "Development of Conic Section Function Neural Networks in Software and Analogue Hardware", Ph.D. Thesis, Liverpool University, UK.

Zhou, R.W., Quek, C., (1996), "An Automatic Fuzzy Neural Network Driven Signature Verification System", ICNN'96 International Conference on Neural Networks, 2-6 June 1996, Washington, DC, USA.

INTERNET KAYNAKLARI

[1] http://www.batitrakya.dostweb.com

EKLER

Ek 1 SRVS eğitme için MLP ile yazılmış program
Ek 2 SRVS eğitme için RBF ile yazılmış program
Ek 3 MLP ve RBF ile eğitilmiş ağın test edilmesi için yazılmış program
Ek 4 SRVS için CSFNN ile yazılmış program

Ek 1 SRVS eğitme için MLP ile yazılmış program

```
%---------------------------------- egitme ---------------------
clear;
clc;
%-- once tum giris vektorlerini yukleyelim
giris_vek=[];
imza_klasor=[];
mat_files=dir(fullfile(pwd,'\egitme_girisleri\*.mat'));
for k=1:length(mat_files)
   file_name=[pwd '\egitme_girisleri\' mat_files(k).name];
   load(file_name);
   [ans,fnm(k,:)]=fileparts(file_name);
   eval(['[ans,L_egitme]=size(' fnm(k,:) ');']);
   eval(sprintf('%s%s%s%d%s','giris_vek=[giris_vek ',fnm(k,:),'(:,1:',L_egitme,')];'));
   imza_klasor=[imza_klasor;fnm(k,11:length(fnm(k,:)))];
   eval(['clear ' fnm(k,:) ';']);
end
save imza_klasor imza_klasor;
% giris vektoru yukarida elde edildi
%-- target vector
[r,c]=size(giris_vek);
HEDEF=zeros(length(mat_files),L_egitme);   %HEDEF vektoru
for k=1:length(mat_files)
   HEDEF(k,(k-1)*L_egitme +1:k*L_egitme)=1;
end
%input range vector
input_ranges=[zeros(r,1) ones(r,1)];
%neural network u olustur.
agim=newff(input_ranges,[60,24,8],{'logsig','logsig','logsig'},'traingdx');
%Show training progress every 100 epochs
agim.trainParam.show=100;
```

```
%Set learning rate to 0.05
agim.trainParam.lr=0.05;
%Set maximum epochs to 10000
agim.trainParam.epochs=10000;
%Set error goal to 0.001
agim.trainparam.goal=1e-3;
%train the network
agim=train(agim,giris_vek,HEDEF);
save agim agim;
```

Ek 2 SRVS eğitme için RBF ile yazılmış program

```
clear;
clc;
%-- once tum giris vektorlerini yukleyelim
giris_vek=[];
imza_klasor=[];
mat_files=dir(fullfile(pwd,'\egitme_girisleri\*.mat'));
for k=1:length(mat_files)
   file_name=[pwd '\egitme_girisleri\' mat_files(k).name];
   load(file_name);
%---------
   [ans,fnm(k,:)]=fileparts(file_name);
   eval(['[ans,L_egitme]=size(' fnm(k,:) ');']);
   eval(sprintf('%s%s%s%d%s','giris_vek=[giris_vek ',fnm(k,:),'(:,1:',L_egitme,')];'));
   imza_klasor=[imza_klasor;fnm(k,11:length(fnm(k,:)))];
   eval(['clear ' fnm(k,:) ';']);
end
save imza_klasor imza_klasor;
% giris vektoru yukarida elde edildi
%-- target vector
[r,c]=size(giris_vek);
HEDEF=zeros(length(mat_files),L_egitme);   %HEDEF vektoru
for k=1:length(mat_files)
   HEDEF(k,(k-1)*L_egitme +1:k*L_egitme)=1;
end
%neural network u olustur.
agim=newrb(giris_vek,HEDEF,1e-3,10);
%--- agi kaydet ----------
save agim agim;
```

Ek 3 MLP ve RBF ile eğitilmiş ağın test edilmesi için yazılmış program

```
clear;
clc;
fonk_path=[pwd '\fonksiyonlar'];
addpath(fonk_path);
%--- test imzasi tespit ediliyor -------
cd([pwd '\test_imzalari']);
[fname,fpath]=uigetfile('*.jpg','TEST ETMEK ISTEDIGINIZ IMZAYI SECINIZ');
cd ..
resim_adi=[fpath fname];
%---- imza resmi okunuyor ----------------------
resim_orj=imread(resim_adi);
resim_orj=resim_orj(:,:,1);        %ilk katmani al
disp([fname ' imzasinin sahibi arastiriliyor...']);
%-----------
giris_vek=giris_vektoru_olustur(resim_orj);
%-------  test imzasi icin agi simule et ----------
load agim;
karar_vek=sim(agim,giris_vek);
load imza_klasor;
[ans,yeri]=max(karar_vek);
if max(karar_vek)>0.5
  karar_msj=sprintf('\n\nbu imza %s kisisine aittir !',imza_klasor(yeri,:));
else karar_msj=sprintf('\n\nbu imza tanınamıyor !');
end
disp(karar_msj);
%------
figure, imshow(resim_orj,'notruesize');
%--------
rmpath(fonk_path);
```

Ek 4 SRVS için CSFNN ile yazılmış program

```
clear all
figure(gcf)
% FUNCTION TO APPROXIMATE
cla reset
%-- once tum giris vektorlerini yukleyelim
giris_vek=[];
imza_klasor=[];
mat_files=dir(fullfile(pwd,'\egitme_girisleri\*.mat'));
for k=1:length(mat_files)
   file_name=[pwd '\egitme_girisleri\' mat_files(k).name];
   load(file_name);
%---------
   [ans,fnm(k,:)]=fileparts(file_name);
   eval(['[ans,L_egitme]=size(' fnm(k,:) ');']);
   eval(sprintf('%s%s%s%d%s','giris_vek=[giris_vek ',fnm(k,:),'(:,1:',L_egitme,')];'));
   imza_klasor=[imza_klasor;fnm(k,11:length(fnm(k,:)))];
   eval(['clear ' fnm(k,:) ';']);
end
save imza_klasor imza_klasor;
%-- target vector
[r,c]=size(giris_vek);
HEDEF=zeros(length(mat_files),L_egitme);   %HEDEF vektoru
for k=1:length(mat_files)
   HEDEF(k,(k-1)*L_egitme +1:k*L_egitme)=1;
end
HEDEF=0.1*(HEDEF==0)+0.9*(HEDEF==1);
%-----pca------
[pn,meanp,stdp,HEDEF,meant,stdt] = prestd(giris_vek,HEDEF);
[giris_vek,transMat] = prepca(pn,0.01);
giris_vek=[giris_vek(:,1:25),giris_vek(:,33:57),giris_vek(:,65:89),giris_vek(:,97:121),giris_ve
```

```
k(:,129:153),giris_vek(:,161:185),giris_vek(:,193:217),giris_vek(:,225:249)];
HEDEF=[HEDEF(:,1:25),HEDEF(:,33:57),HEDEF(:,65:89),HEDEF(:,97:121),HEDEF(:,129:
153),HEDEF(:,161:185),HEDEF(:,193:217),HEDEF(:,225:249)];
% DESIGN NETWORK
 echo on
 df = 1;   % frequency of progress displays (in neurons).(1)
 me = 100;  % maximum number of neurons.(100)
 eg = 0.0001; % sum-squared error goal. (0.001)
 sc = 1;    % spread constant radial basis functions.(10)
 s1 = 30;   % the number of centres(30)
 echo off
 % PLOTTING FLAG
 [r,q] = size(giris_vek);
 [s2,q] = size(HEDEF);
 % RADIAL BASIS LAYER OUTPUTS
 b = sqrt(-log(.5))/sc;
 P = ylogsig(dist(giris_vek',giris_vek)*b);
 PP = sum(P.*P)';
 d = HEDEF';
 dd = sum(d.*d)';
 % CALCULATE "ERRORS" ASSOCIATED WITH VECTORS
 e = ((P' * d)' .^ 2) ./ (dd * PP');
 %PICK VECTOR WITH MOST "ERROR"
 pick = findLargeColumn(e);
 used = [];
 left = 1:q;
 C = P(:,pick);
 P(:,pick) = []; PP(pick,:) = [];
 e(:,pick) = [];
 used = [used left(pick)];
 left(pick) = [];
```

```
% CALCULATE ACTUAL ERROR
c1 = giris_vek(:,used)';
[crr,css] = size(c1);
om = pi/4*ones(1,crr); %pi/4
w1 = ones(crr,r);
a1 = ylogsig(consec(c1,giris_vek,w1,om)*b);
[w2,b2] = ysolvelin(a1,HEDEF);
a2 = ylogsig(w2(:,1:1)*a1,b2);
sse = sumsqr(HEDEF-a2);
% TRAINING RECORD
tr = zeros(1,me);
tr(1) = sse;
% PLOTTING
newplot;
message1 = sprintf('RBF, neurons = 0, SSE = %g\n',sse);
fprintf(message1,0,sse);
yploterr(tr(1),'error')
for k = 1:s1-1
%%%CHECK ERROR
  if (sse < eg), break, end
% CALCULATE "ERRORS" ASSOCIATED WITH VECTORS
cj = C(:,k);
%---- VECTOR CALCULATION FOR CENTRES
a = cj' * P / (cj'*cj);
P = P - cj * a;
PP = sum(P.*P)';
e = ((P' * d)' .^ 2) ./ (dd * PP');
% PICK VECTOR WITH MOST "ERROR"
pick = findLargeColumn(e);
C = [C, P(:,pick)];
P(:,pick) = []; PP(pick,:) = [];
```

```
e(:,pick) = [];
used = [used left(pick)];
left(pick) = [];
% CALCULATE ACTUAL ERROR
c1 = giris_vek(:,used)';
[cr,cc] = size(c1);
om = pi/4*ones(1,cr); %pi/4
w1 = ones(cr,r);
a1 =ylogsig(consec(c1,giris_vek,w1,om)*b);
[w2,b2] = ysolvelin(a1,HEDEF);
a2 = ylogsig(w2*a1,b2);
sse = sumsqr(HEDEF-a2);
% TRAINING RECORD
tr(k+1) = sse;
% PLOTTING
   if rem(k,df) == 0
   %fprintf(message1,k,sse);
   fprintf('RBF, neurons = %g, SSE = %g\n',k,sse);
   yploterr(tr(1:(k+1)),'error');
   end
[S1,R] = size(c1);
b1 = ones(S1,1)*b;
end
echo on
%    TRAINING THE NETWORK
%    This uses backpropagation to train feed-forward networks.
df = 10;   % Frequency of progress displays (in epochs).
me = 140; % Maximum number of epochs to train.
eg = 0.05; % Sum-squared error goal.
lr = 0.02; % Learning rate.
tp = [df me eg lr];
```

```
%    Training begins...please wait (this takes a while!)...
echo off
% TRAINING PARAMETERS
df1 = feval('ylogsig','delta');
df2 = feval('ylogsig','delta');
% INITIALIZATION OF WEIGHTS
[crr,css] = size(c1);
om = pi/4*ones(1,crr); %pi/4
w1 = zeros(crr,r);
[a1,a2] = yhybrid(c1,giris_vek,w1,om,b,b1,w2,b2);
e = HEDEF-a2;
SSE = sumsqr(e);
% PLOTTING
clf
message = sprintf('TRAINBP: %%g/%g epochs, SSE = %%g.\n',me);
fprintf(message,0,SSE);
yploterr(tr(k),'error');
for i=k:me
% CHECK PHASE
 if SSE < eg, i=i-1; break, end
% BACKPROPAGATION PHASE
d2 = feval('ydlogsig',a2,e);
d1 = feval('ydlogsig',a1,d2,w2);
% LEARNING PHASE
 for ii=1:16
  for m=1:crr
   cn = c1(m,:)'*ones(1,q);
   pnew = (cn-giris_vek);
  end
 [dw1,db1] = ylearnbp(pnew,d1,lr);
 [dw2,db2] = ylearnbp(a1,d2,lr);
```

```
  w1 = w1 + dw1;
  b1 = b1 + db1;
  w2 = w2 + dw2;
  b2 = b2 + db2;
% PRESENTATION PHASE
  [a1,a2] = yhybrid(c1,giris_vek,w1,om,b,b1,w2,b2);
  e = HEDEF-a2;
  SSE = sumsqr(e);
% TRAINING RECORD
  tr1(ii+1) = SSE;
 end
 % UPDATE ANGLE
 d2 = feval('ydlogsig',a2,e);
 d1 = feval('ydlogsig',a1,d2,w2);
 difom = sin(om)*dist(c1,giris_vek);
 dom = (ylearnbp(difom,d1,lr))';
 om = om + dom;
 [omr,omc] = size(om);
 for s=1:omc
  if om(1,s) > pi/2
     om(1,s) = pi/2;
  end
  if om(1,s) < -pi/2
     om(1,s) = -pi/2;
  end
 end
 [a1,a2] = yhybrid(c1,giris_vek,w1,om,b,b1,w2,b2);
 e = HEDEF-a2;
 d2 = feval('ydlogsig',a2,e);
 d1 = feval('ydlogsig',a1,d2,w2);
 [a1,a2] = yhybrid(c1,giris_vek,w1,om,b,b1,w2,b2);
```

```
 e = HEDEF-a2;
 tr(i+1) = SSE;
 if rem(i,df) == 0
 fprintf(message,i,SSE);
    drawnow;
    yploterr(tr(1:(i+1)),'error');
 end
end
% TRAINING RECORD
tr = tr(1:(i+1));
% PLOTTING
if rem(i,df) ~= 0
 fprintf(message,i,SSE);
   drawnow;
   yploterr(tr,'error');
end
% ERRORS
yploterr(tr,'error');
str= str2mat(...
 ' We can plot the error vs. epoch (number of neurons) to see how',...
 ' quickly a solution was designed.',...
 '',...
 ' >> yploterr(err,eg);',...
 '',...
 [' The final error fell below the error goal of ' num2str(eg) ,...
 ' when the hidden layer reached ' num2str(i) ' neurons.'     ])
drawnow;
% SLIDE 6: TEST NETWORK
% %-- şimdi TEST vektorlerini yukleyelim
giris_vek=[];
imza_klasor=[];
```

```
mat_files=dir(fullfile(pwd,'\egitme_girisleri\*.mat'));
for k=1:length(mat_files)
   file_name=[pwd '\egitme_girisleri\' mat_files(k).name];
   load(file_name);

   %---------
   [ans,fnm(k,:)]=fileparts(file_name);
   eval(['[ans,L_egitme]=size(' fnm(k,:) ');']);
   eval(sprintf('%s%s%s%d%s','giris_vek=[giris_vek ',fnm(k,:),'(:,1:',L_egitme,')];'));
   imza_klasor=[imza_klasor;fnm(k,11:length(fnm(k,:)))];
   eval(['clear ' fnm(k,:) ';']);
end
save imza_klasor imza_klasor;
[pn,meanp,stdp] = prestd(giris_vek);
[giris_vek,transMat] = prepca(pn,0.01);%0.01
giris_vek=[giris_vek(:,26:32),giris_vek(:,58:64),giris_vek(:,90:96),giris_vek(:,122:128),giris_
vek(:,154:160),giris_vek(:,186:192),giris_vek(:,218:224),giris_vek(:,250:256)];
say=0;
for v=1:56
ax(:,v) = ysimcon(c1,giris_vek(:,v),w1,om,b1,w2,b2);
end
for m=1:7
      if (ax(1,m)>0.75)&(ax(2,m)<0.35)&(ax(3,m)<0.35)&(ax(4,m)<0.35)&(ax(5,m)<0.35)&
                                        (ax(6,m)<0.35)&(ax(7,m)<0.35)&(ax(8,m)<0.35)
   say=say+1;
   end
end
for m=8:14
      if (ax(1,m)<0.35)&(ax(2,m)>0.75)&(ax(3,m)<0.35)&(ax(4,m)<0.35)&(ax(5,m)<0.35)&
                                        (ax(6,m)<0.35)&(ax(7,m)<0.35)&(ax(8,m)<0.35)
                                                          say=say+1;
```

```
   end
end
for m=15:21
      if (ax(1,m)<0.35)&(ax(2,m)<0.35)&(ax(3,m)>0.75)&(ax(4,m)<0.35)&(ax(5,m)<0.35)&
                                        (ax(6,m)<0.35)&(ax(7,m)<0.35)&(ax(8,m)<0.35)
   say=say+1;
   end
end
for m=22:28
  if (ax(1,m)<0.35)&(ax(2,m)<0.35)&(ax(3,m)<0.35)&(ax(4,m)>0.75)&(ax(5,m)<0.35)&
                                        (ax(6,m)<0.35)&(ax(7,m)<0.35)&(ax(8,m)<0.35)
  say=say+1;
  end
end
for m=29:35
  if (ax(1,m)<0.35)&(ax(2,m)<0.35)&(ax(3,m)<0.35)&(ax(4,m)<0.35)&(ax(5,m)>0.75)&
                                        (ax(6,m)<0.35)&(ax(7,m)<0.35)&(ax(8,m)<0.35)
  say=say+1;
  end
 end
for m=36:42
  if (ax(1,m)<0.35)&(ax(2,m)<0.35)&(ax(3,m)<0.35)&(ax(4,m)<0.35)&(ax(5,m)<0.35)&
                                        (ax(6,m)>0.75)&(ax(7,m)<0.35)&(ax(8,m)<0.35)
   say=say+1;
   end
  end
for m=43:49
  if (ax(1,m)<0.35)&(ax(2,m)<0.35)&(ax(3,m)<0.35)&(ax(4,m)<0.35)&(ax(5,m)<0.35)&
                                        (ax(6,m)<0.35)&(ax(7,m)>0.75)&(ax(8,m)<0.35)
 say=say+1;
 end
```

```
end
for m=50:56
  if (ax(1,m)<0.35)&(ax(2,m)<0.35)&(ax(3,m)<0.35)&(ax(4,m)<0.35)&(ax(5,m)<0.35)&
                                        (ax(6,m)<0.35)&(ax(7,m)<0.35)&(ax(8,m)>0.75)
  say=say+1;
  end
 end
say
if say==56; break,end
```

ÖZGEÇMİŞ

Doğum tarihi	01.01.1975	
Doğum yeri	Akseki-Antalya	
Lise	1988-1991	Antalya Lisesi-Antalya
Ön Lisans	1991-1993	İstanbul Üniversitesi Sosyal Bilimler Meslek Yüksek Okulu Turizm ve Otelcilik Bölümü
Lisans	1994-1998	İstanbul Üniversitesi Mühendislik Fakültesi Elektrik-Elektronik Mühendisliği Bölümü
Yüksek Lisans	2001-....	Yıldız Teknik Üniversitesi Elektronik ve Haberleşme Müh. Anabilim Dalı, Elektronik Mühendisliği Bölümü

Çalıştığı kurum(lar)

1998-2000	Dünya Göz Hastanesi Bilgi İşlem Sorumlusu
2000-Devam ediyor	Kadir Has Üniversitesi Mühendislik Fakültesi Elektronik Mühendisliği Bölümü Araştırma Görevlisi