YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRONİK RETİNA TASARIMINA UYGUN YAPAY SİNİR AĞI YAPISININ ARAŞTIRILMASI

Elektronik ve Hab. Müh. Nihan COŞKUN

FBE Elektronik ve Haberleşme Mühendisliği Anabilim Dalı Elektronik Programında Hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı: Doç. Dr. Tülay YILDIRIM (Y.T.Ü.)

Jüri Üyeleri : Prof. Şefik SARIKAYALAR (Y.T.Ü.)

Yrd.Doç.Dr. Şeref Naci Engin (Y.T.Ü.)

İSTANBUL, 2003

İÇİNDEKİLER

SİMGE LİSTESİ

w	YSA'da hücrenin ağırlıklar matrisi
x	YSA'da hücrenin giriş vektörü
v	YSA'da hücrenin net girişi
y	YSA'da hücrenin çıkışı
$\varphi(.)$	YSA'da hücrenin aktivasyon fonksiyonu
μ	Mobilite
Cox	Birim alan başına düşen parazitik kapasite
W	Transistör genişliği
L	Transistör uzunluğu
C	Radyal tabanlı fonksiyonda merkezler
σ	Radyal tabanlı fonksiyonda yayılma parametresi
e	YSA'da hata vektörü
ϖ	Konik Kesit Fonksiyonlu Sinir Ağlarında açılma açısı
∂	Türev Operatörü

KISALTMA LİSTESİ

MLP	Multilayer Perceptron (Çok Katmanlı Algılayıcı)
RBF	Radial Basis Function (Radyal Tabanlı Fonksiyon)
PNN	Probabilistic Neural Networks (Olasılıksal Sinir Ağları)
GRNN	General Regression Neural Network (Genelleştirilmiş Regresyonlu Sinir Ağları)
CSFNN	Conic Section Function Neural Network (Konik Kesit Fonksiyonlu Sinir Ağları)
VLSI	Very Large Scaled Integrated (Circuits)
YSA	Yapay Sinir Ağları
YGKİ	Yöresel Geri Küresel İleri
BHYSA	Bellek Hücreli Yapay Sinir Ağı
İBYSA	İleri Beslemeli Yapay Sinir Ağı
GBYSA	Geri Beslemeli Yapay Sinir Ağı
PDF	Olasılıksal Yoğunluk Fonksiyonu

ŞEKİL LİSTESİ

ÇİZELGE LİSTESİ

ÖNSÖZ

Bu tezin hazırlanmasında ve yaptığımız bilimsel çalışmalarda gerek yönlendirme gerekse kaynak temininde her türlü desteği ile bana yardımcı olan başta saygıdeğer hocam sayın Doç.Dr. Tülay YILDIRIM'a, yetişmemde emeği geçen tüm hocalarıma, her zaman her koşul altında daima yanımda olduklarını bildiğim tüm aileme, anneme,babama ve canım kardeşime, hiç bir zaman desteğini benden esirgemeyen ve esirgemeyeceğini de bildiğim değerli arkadaşım Ahmet KAHRAMAN'a, her tür ruh halime iki yıl boyunca katlanan çok sevgili oda arkadaşlarım Tuba KIYAN, Burcu KAPANOĞLU ve Revna ACAR'a teşekkürlerimi sunarım.

ÖZET

Görme olayı, bir cisimden göze gelen ışınların reseptörler üzerindeki etkisiyle gerçekleşir. Göze gelen ışınlar saydam tabakada (kornea) kırıldıktan sonra, göz bebeğinden girerek merceğe gelir. Göz merceği ışığı bir kere daha kırar ve kırılan bu ışınlar gözdeki camsı cismi geçtikten sonra retina üzerinde ters bir görüntü meydana getirir. Bu şekilde retinaya gelen ışınlar görme reseptörlerini uyararak görme sinirlerinde impulsları başlatır. İmpulslar sinirlerle beynin görme merkezine iletilerek değerlendirilir. Bundan sonra cisim düz, net ve renkli olarak görülür.

Görüntünün bir lens üzerine düşürülerek 2 boyutlu olarak algılanması ve yapay nöronlar yoluyla beyne iletilmesi bilim dünyasında Carver Mead ve grup arkadaşlarının 1970'li yıllara dayanan çalışmalarıyla başlamıştır. Silikon Retina adını verdikleri bu çalışmada Mead ve arkadaşları nöronların ve sinapsların elektronik modellerini çıkararak görüntüyü ayırt etmeye çalışmışlardır.

Görüntü algılaması ve görüntünün ayrıştırılarak sınıflandırılabilmesi, son yıllarda yapay sinir ağı teknolojisinde de kullanılmaktadır. Önceden verilen görüntü verisine dayanarak başka görüntülerin de kestirimi veya görüntü üzerindeki birkaç piksellik bölgenin kaybı durumunda görüntünün doğru şekilde algılanabilmesi için kullanılan sistem ezberlemeden öte eğiticili bir sistem olmalıdır. Bu yüzden yapay sinir ağları ile görüntü sınıflandırma üzerine yapılan çalışma sayısı gün geçtikçe artmaktadır.

Bu tezde görme sisteminin elektronik tasarımına en uygun yapay sinir ağı yapısı bulunması amaçlanmıştır. Görüntünün algılanması ve daha sonra piksellere ayrılmasıyla elde edilen bilgi kümesi, yapay sinir ağları ile eğitilerek örnek bilgi kümesinde olmayan görüntüler için de ağın tanıma yeteneğinin arttırılması için çalışılmıştır. Kullanılan sinir ağı modelleri arasında çok katmanlı algılayıcılar (MLP), radyal temelli fonksiyonlu ağlar (RBF), istatistiksel sinir ağları (PNN), genelleştirilmiş regresyonlu ağlar (GRNN) ve son olarak da 1994 yılında Dorffner tarafından ortaya atılan konik kesit fonksiyonlu sinir ağları (CSFNN) bulunmaktadır. Konik Kesit Fonksiyonlu Yapay Sinir Ağında esas fikir, bir Çok Katmanlı Algılayıcı birimi ile bir Radyal Temelli Fonksiyonlu ağ birimi arasında ilişki saglayarak bunlara ait karar bölgelerini içine alacak bir birimin fonksiyonunu genelleştirmektir.

Tezin birinci bölümünde konuya giriş yapılmış, ikinci bölümde ise görüntünün gözde nasıl algılandığı açıklanarak görüntü sınıflandırma metodları üzerinde durulmuştur. Üçüncü bölümde sinir ağı tarihçesi ve yapay sinir ağlarının özellikleri kısaca anlatılmıştır. Dördüncü bölümde temel sinir ağı algoritmaları açıklanmıştır. Beşinci bölümde ise konik kesit fonksiyonlu sinir ağları hakkında detaylı bir inceleme yapılarak, 2000 yılında yapılan doktora tezinde (Dr. Lale Özyılmaz) önerilen MATLAB programının üzerinde bazı değişikliklerle programın MATLAB 6.0'a adaptasyonu sağlanmıştır. Daha sonra ise bu program, UCI Makina Öğrenmesi veri tabanında bulunan görüntü veri seti üzerinde denenmiş ve sonuçlar daha önce incelenen MLP, RBF, PNN ve GRNN gibi ağlar ile karşılaştırılarak görüntünün sınıflandırılarak algılanabilmesi için en iyi sonuç veren ağ yapısı belirlenmiştir.

Anahtar kelimeler: Görüntü algılama, görüntü algılama için elektronik devreler, yapay sinir ağları, konik kesit fonksiyonlu sinir ağları.

ABSTRACT

Biologically seeing becomes true by the effects of the signals on the receivers of the eye. Signals coming to eye first change direction on the cornea (transparent layer), than get in to the pupil and come to the lens. In the lens layer, light signals change direction once more and form the image reversly on the retina. After the light signals stimulate the receivers on the eye, they start the impulse at the eye's neurons.

The studies about the 2D image recognition and transmition to the brain by the artificial neural network have started with Carver Mead and his colleagues in 1970's. They called their studies "Artifical Silicon Retina" and discovered electrical circuits of the neurons and synapses.

Image recognition and segmentation are used in neural network technology in recent years. The system used in image recognition or in the conditions of having lost some pixels of the image should be a supervised system instead of memorizing system.

In this thesis the most suitable artificial neural network configuration for electrically design of visiual system is examined. For this purpose Multilayer Perceptron (MLP), Radial Basis Functions (RBF), Probabilistic Neural Networks (PNN), Generalized Regresion Neural Networks (GRNN), Conic Section Function Neural Networks (CSFNN) are used in image database.

In the first chapter of the thesis there is an introduction in to the topic, in the second chapter image recognation and image segmentation are explained. In the third chapter neural network history and properties of neural network are given. In chapter four artificial neural network configurations mentioned above are explained in detail. After that in chapter five a program in matlab 6.0 is done for simulation of CSFNN and finally in the last chapter the comparisons of the results are given.

Keywords: Image recognition, electric circuits for image recognition, artificial neural network, conic section function neural networks.

1. GİRİŞ

Canlılar dış ve iç çevrelerden gelen uyarıları alarak değişen ortam şartlarına göre sürekli kendilerini dengede tutmak zorundadırlar. Organizmayı değişen ortam şartlarından haberdar eden yapılardan biri de gözdür. Göz görünen ışıktan 3500 ile 7800 A^{o}'luk dalga boyuna sahip olanlara duyarlıdır. Mor ötesi ve kırmızı ötesi dalga boylarını algılayamaz. Göz embriyonik gelişme evresinden ön beynin uç bölümünden farklılaşarak oluşur. Göz yuvarlağı dıştan içe doğru sert tabaka, damarlı tabaka ve ağ tabaka (retina) olarak sıralanır.

- Sert tabaka : Gözün içine yerleştiği, gözü koruyan sağlam bir dokudur.
- Damar tabaka : Sert tabakanın altında gözün beslenmesini sağlayan kısımdır.
- Ağ tabaka (Retina) : Bu tabakada ışığa duyarlı reseptörler ve sinir hücreleri ile döşenmiş kompleks bir yapıdır. İnsan gözünde ışıkla ilgili duyu reseptörleri buraya yerleştiğinden ışığa karşı duyarlı tek bölgedir.

Retinanın görmede en etkili görev yapan bir kısmı vardır ki buraya fovea çukuru (sarı benek) denir. Kornea ve mercek, ışığı toplayarak bu noktaya düşmesini sağlar. Bu kısma parlak ışık rengi ve cisimlerin ayrıntılarını vermekle sorumlu koni hücreleri kümelenmiştir. Işığa duyarlı diğer çomak hücreleri fovea çukurundan uzakta ve retinanın kenar bölgelerinde yerleşirler. Bu hücreler loş ışıkta görev yaparlar ve renksiz görürler. Loş ortamda cisme değilde cismin biraz yan tarafına bakılırsa görme olayı daha net olmaktadır. Çünkü loş ortamda görmeyi sağlayan çomak hücreleri retinanın kenarına yerleşmişlerdir. (Başaran,1995)

Caltech'de Carver Mead'in grubu tarafından keşfedilen silikon çiplerde (silikon retina) büyük oranda biyolojik nöral ağlardan esinlenilmiştir. Mead biyolojik nöronlara mümkün olduğunca çok benzeyen analog nöral çipler yapmaya çalışmıştır; bunlara son derece düşük güç tüketimi, tamamen analog donanım ve devamlı çalışabilme yeteneği dahil etmiştir. (Mead,1989)

Görüntü sınıflandırma, bir görüntünün anlamlı küçük sınıflara bölünebilmesi işlemini içermektedir. Bu sınıflandırma işlemi makina görüşünü inceleyen bir çok çalışmada incelenmiş ve çok çeşitli teknikler geliştirilmiştir. Temelde bütün incelenen algoritmalar dört kategoriye ayrılabilir. Bunlar piksel sınıflandırma, kenar tabanlı ayırma, bölgesel ayırma ve karma ayırma teknikleridir.

Yapay sinir ağları kullanılarak yapılan sınıflandırmada önce görüntüye ait sonuçları bilinen değerlerle ağ eğitilir (eğitim), daha sonra bilinmeyen görüntülerin ayırt edilmesi işlemi (test) gerçekleştirilir.

İnsan beynini bilim dallarına uygulamak uzun zamandır bilimsel araştırmaların gündemini oluşturmaktadır. Yapay sinir ağları ya da kısaca YSA da, insan beyninin çalışma sisteminin yapay olarak benzetimi çalışmalarının bir sonucu olarak ortaya çıkmıştır.

Yapay sinir ağları insan vücudundaki sinir sisteminin bazı fonksiyonlarını modelleyen ve bazı yeteneklerini yakalamak isteyen basit hesapsal birimlerin (nöronlar) yoğun bir paralel dizisidir; başka bir deyişle, teorik hale getirilmiş zeka ve beyin faaliyetlerinin matematiksel modelleridir. Ancak biyolojik sistemler o kadar karmaşıktır ki yapay sinir ağı için kullanılan biyolojik modellerin fazlaca basite indirgenmiş biçimleri şeklindedir.

Yapay sinir ağlarının bilgisayar ile simülasyonlarına ilk defa 1940'lı yıllarda başlanmıştır. 1943 yılında McCulloh ve Pitts yapay nöronu tanımlamışlardır. Bunun paralelinde bilgisayar teknolojisinin gelişimiyle, nöral fonksiyonların hesaplanmasının kolaylaştığı ve basit nöron birleşimlerinin aktivitesinin arttığı gözlenmiştir. Pratikteki uygulamaların matematiksel modellerinin gelişmesi ise daha sonraki yıllarda McLelland, Rumelhart, Hopfield ve Kohonen'in çalışmalarıyla olmuştur. Macaristanda T.Roska'nın çalışmalarıyla ilk defa hücresel yapay sinir ağları alanında sonuçlar elde edilmiştir. Tek katmanlı ağlarında bazı basit işlemleri gerçekleştirememesi yapay sinir ağlarına olan ilginin azalmasına neden olmuştur. 1969 yılında Minsky ve Papert, tek katmanlı ağlarla ayrıcalıklı veya (XOR) işleminin yapılamayacağını ortaya koymuştur. 1965'ten 1984'e kadar olan yıllar arasında birçok araştırmacı bu konuda çalışmalarda bulunmuştur. Ağırlıklı toplama elemanlarının öğrenmeleri ile ilgili matematiksel teoremler Sun-Ichi Amari (1972-1977) tarafından geliştirilmiştir. Ayrıca Japonya'da Kunihiko Fukushima neocognitrons adı verilen yeni bir sinir ağı mimari şekli geliştirmiştir. 1974-1982 yılları arasında Grossberg ve Carpenter, sinir ağları ile ilgili mimari şekilleri ve teoremler dizisi ortaya atmıştır. (Hecht, Wl; Harvey, 1994; Haykin, 1994; Zurada, 1995; Schalkoff, 1997)

Çok Katmanlı Algılayıcı (Multilayer Perceptrons-MLP) ve Radyal Temelli Fonksiyonlu (Radial Basis Function-RBF) yapay sinir ağları örüntü tanıma, sınıflandırma problemleri gibi birçok alanda kullanılabilir. Birçok araştırmacı bu ağların eğitme algoritmalarının geliştirilmesi için değişik çalismalar yapmıştır. MLP için en popüler öğrenme algoritması olan hatanin geriye yayılması 1986 yılında McClelland ve Rumelhart tarafindan geliştirilmiştir. RBF ağlarının eğitilmesi için değişik metodlar kullanılmaktadır. RBF ve MLP aglarında en uygun eğitme algoritmasının hangisi olduğu konusunda araştırmalar devam etmektedir. Olasılıksal sinir ağları (PNN) 1990 yılında Donald Specht tarafindan geliştirilmiştir. Olasılıksal sinir ağları bir kalıp katman dahilinde dağıtım işlevini geliştirmek için eğiticili bir

eğitim dizisi kullanır. Hatırlama kipinde bu işlevler öğrenilmiş bir kategori veya sınıfın parçası olan bir girdi özelliği vektörünün benzerliğini tahmin etmek için kullanılır (Specht, 1990). Konik kesit fonksiyonlu sinir ağında genel durumda birbirinden farklı propagasyon kuralı, aktivasyon fonksiyonu ve öğrenme kuralına sahip olan MLP ve RBF ağlarının çeşitli kombinasyonlarla birleştirilmesine çalışılmıştır. Maruyama, Girosi ve Poggio ilk olarak radyal temelli fonksiyon olarak sigmoid fonksiyon kullanarak genelleştirilmiş RBF ve MLP arasında bir bağlantı sağlamışlardır (Maruyama vd., 1992). Tarassenko ve Roberts, bir RBF sınıflayıcıyı hatanın geriye yayılması ile eğitmişlerdir (Tarassenko ve Roberts, 1994). (Hirahara ve Oka, 1993), MLP ve RBF için iki ayrı modül kullanmış ve bunları bir lineer birleştiriciden geçirerek hibrid bir model geliştirmişlerdir. 1992'de hem kapalı hem açık karar sınırlarını birleştiren projeksiyon sinir ağı önerilmiştir (Wilensky ve Manukian, 1992). Diğer araştırmacılarda (Tsoi, 1989; Geva ve Sitte, 1992; Smyth, 1992; Weymaere ve Martens, 1994) RBF/MLP hibrid ağları ve her iki ağ için değişik öğrenme algoritmaları geliştirmişlerdir. Dorffner MLP ve RBF'i bir ağda birleştiren Konik Kesit Fonksiyonlu Sinir Ağını tanıtmıştır (Dorffner, 1994; Yıldırım ve Marsland, 1997).

2. GÖRÜNTÜ ALGILAMA, SINIFLANDIRMA VE YAPAY SİNİR AĞLARI

Görme olayı, bir cisimden göze gelen ışınların reseptörler üzerindeki etkisiyle gerçekleşir. Göze gelen ışınlar saydam tabakada (kornea) kırıldıktan sonra, göz bebeğinden girerek merceğe gelir. Göz merceği ışığı bir kere daha kırar ve kırılan bu ışınlar gözdeki camsı cismi geçtikten sonra retina üzerinde ters bir görüntü meydana getirir. Bu şekilde retinaya gelen ışınlar görme reseptörlerini uyararak görme sinirlerinde impulsları başlatır. İmpulslar sinirlerle beynin görme merkezine iletilerek değerlendirilir. Bundan sonra cisim düz, net ve renkli olarak görülür.

Karanlıkta Görme

Göz, çok değişik sınırlar içerisinde bulunan ışık şiddetlerine "Rodopsin" sistemi ile karşılık verir. Rodopsin parçalanarak yeniden eski haline gelme özelliğine sahiptir. Işık karşısında parçalanan rodopsin görme olaylarının impulsunu başlatır. Rodopsin ışık karşısında opsin ve retinal adı verilen maddelere parçalanmaktadır. Opsin, bir protein molekülüdür. Retinal ise A vitamininden türevlenmiş bir maddedir. Retinal, A vitamini ile reaksiyona girer ve cis-retinal adı verilen bir madde üretilir. Cis-retinal ve opsin yeniden birleşerek rodopsini oluşturur. Bu sırada impuls oluşturulur. Oluşan impuls sinir hücresine aktarılır ve optik sinirle de beyin merkezine aktarılır ve optik sinirle beynin görme lobunda görüntü ortaya çıkar. Görüntü görme merkezinde ters olarak algılanır, ancak görme olayı açıklandığı şekilde olduğundan artık organizmaya normal gelmektedir. Gözde görme olayları esnasında bir cisim 6 salise kadar görüntü şeklinde kalabilmektedir. Bu kalma sayesinde film seyrederken kesik kesik olan sahneler süreklilik arzederek seyredilebilmektedir.

Renkli Görme

Koni hücrelerinde renkli görmenin kimyasal yapısını oluşturan madde "iyodopsin"dir. Bu maddenin gerçekleştirdiği reaksiyon hakkındaki bilgiler rodopsin reaksiyonu kadar açık ve net değildir. Koni hücreleri fazla ışık karşısında uyarılabilen hücrelerdir. Az ışıkta çalışmazlar. Çünkü konilerin görevi renkli görmeyi sağlamaktır. Konilerde renkli görme olayı birkaç görüş ile açıklanmaktadır. Bunlardan birine göre; ışığa tepki gösteren üç tip koni vardır. Bunlar yeşil, mavi ve kırmızı konilerdir. Bu koniler içerisinde kırmızı yeşil ve mavi renk reseptörleri çıkarılmıştır. Üç çeşit koni bazen tek başına bazen de iki veya üçü bir arada renklere karşı duyarlılık göstermekte ve bu sayada bütün renkler algılanmaktadır. Özellikle ara renkler bu şekilde algılanmaktadır. Örneğin yeşil koni; yeşil ve mavi ışığa karşı duyarlı olmaktadır. Bunlar arasında en şiddetli tepkiyi yeşile göstermektedir. Renk körlüğü olayı bu

üç koniden herhangi birinin olmaması sonucunda ortaya çıkmaktadır. Çünkü ara renklerin ayırt edilmesinde her üç koninin gözde olması gerekmektedir. (Başaran, 1995)

Bir sınıfta kürsüde konuşma yapan konuşmacının sağında ya da solunda bir kapının açılması gibi ani bir değişim sözkonusu olduğunda dinleyicilerin tümünün gözleri o yöne çevrilir. Bu tepki ani ve oldukça istemsiz gibidir. David Sparks, David Robinson ve arkadaşlarının yaptıkları deneyler sayesinde gözlerin nereye çevrilmeleri gerektiğini bilmelerine ilişkin oldukça iyi bilgiler edinilmiştir. Beyinde bulunan tepeciğin üst bölgesi bir duyusal harita olarak tanımlanabilirse, orta ve derin bölgelerinde de bir hareket haritası var gibidir. Bu bölgelerdeki nöronların ateşlemeleri, gözün hedefe sıçraması için konumunda gereken değişikliğin yönü ve miktarını belirtir. Bu işaret, gözün harekete geçmeden önceki konumundan büyük ölçüde bağımsızdır. Beyin sapına gönderilen mesaj ne yönde ve ne kadar bir sıçrama gerektiği yolundadır. Örneğin bir nöron, sıçramanın belli bir yönüyle ilgili olabilir ve ateşlemesinin sıklığı sıçrama uzaklığını belirtebilir. Böylece küçük bir nöron takımıyla tüm yönler ve uzaklıklar belirtilebilir. Alternatif bir yolla daha çok sayıda nöron kullanılarak, her bir nöron belli bir sıçrama vektörünü- yani sıçrama yönü ve uzaklığını- belirtebilir. Ancak gerçek durum oldukça farklıdır. Belli bir oynama için tepecikte belli bir nöron takımı hızlı ateşlemeye başlar. İşte bu etkinliğin hareket haritasındaki ağırlık merkezi sıçrama vektörünü belirler. Böylece tek bir tepecik nöronu bir çok değişik sıçrama türünde rol alabilir. Sıçramanın vektör özelliğini belirleyen, toplu olarak etkin durumdaki nöronlardır. Kısacası tek bir göz hareketini çok sayıda nöronlar denetler. Göz hareketinin hızı ise etkin nöron takımının ateşleme sıklığıyla orantılı olabilir. Böylece sıçramanın yönü, ilgili nöronların ne kadar hızlı ateşlediklerine değil, yalnızca bu nöron takımının etkinlik merkezinin hareket haritasındaki konumuna bağlı olmuş olur. Bu düzenleme aslında bir nöron kümesinin göz hareketlerinin hızı ve yönü gibi parametreleri nasıl belirlediklerine güzel ve tipik bir örnektir. Avantajı, nöronlardan bir kaçının etkin durumdan çıksa bile sistemin çalışır durumda olmasıdır.

İzdüşüm ya da harita sözcüğü görme sisteminde iki değişik anlamda kullanılır. İzdüşümün genel anlamı, alıcı bölgede birbirine komşu olan akson uçlarının gönderici bölgede birbirinden çok uzakta olmayan nöronlardan çıkmasıdır. Bu, gönderici bölgenin alıcı bölgede bir çeşit haritasını oluşturacaktır. Sözcük daha dar anlamıyla "retinotopik " izdüşüm olarak da kullanılır. Belli bir görme bölgesindeki komşu nöronların, retinada komşu noktaların (ve böylece 3B görüş alanının 2B izdüşümündeki komşu noktaların) etkinliğine tepki göstermeleri demektir. Görme sisteminde ilerlendikçe retinotopik izdüşüm giderek

karmaşıklaşır ancak bir bölgedeki nöronların bir sonrakine izdüşümü hala çok iyi korunarak gerçekleşiyor olabilir (Crick, 2000)

Böceklerde Görme, Petek Göz (Birleşik Göz)

Petek göze böceklerde rastlanır. Böceklerde göz binlerce gözden oluşur. Her göz kornea, mercek, pigment hücreleri ve impulsları taşıyan nöronları bulundurur. Bu şekildeki her göze "ommatidium" adı verilir. Ommatidiumların her biri birim alanını gözler. Sonra bu birimleri birleştiği resimler tam olarak görüntüyü sinir merkezine aktarır. Böcek gözleri bazı durumlar için insan gözüne göre üstün özelliklere sahiptir. İnsan saniyede 50 titreşimi farkeder. Halbuki böcekler 250'den fazla titreşimi farkeder. Yani böcekler televizyonu kartpostalların geçişi gibi izler. Çünkü filmlerin titreşim sayısı insana göre hareket eder. Ayrıca böceklerde ultraviyole ışığına karşı da duyarlılık vardır. Yani U.V. ışığını yansıtan bütün cisimleri algılayabilirler (Crick, 2000).

2.1 Görüntü Sınıflandırma

Görüntü sınıflandırma, bir görüntünün anlamlı küçük sınıflara bölünebilmesi işlemini içermektedir. Bu sınıflandırma işlemi makina görüşünü inceleyen bir çok çalışmada incelenmiş ve çok çeşitli teknikler geliştirilmiştir. Temelde bütün incelenen algoritmalar dört kategoriye ayrılabilir. Bunlar piksel sınıflandırma, kenar tabanlı ayırma, bölgesel ayırma ve karma ayırma teknikleridir. Piksel sınıflandırma, piksel ölçülerinin belirli aralıklarda olmasına dayalı sınıflandırmadan ibarettir. Kenar tabanlı ayırma, genellikle ardarda iki kademeden oluşur. Bunlar kenar sınırları belirleme ve kapalı bölge içinde kalan resimdeki hatların belirlenmesidir. Bölgesel ayırma, direk bölgeler üzerinde işlem görür. Karma tekniği ise birkaç tekniği birarada yürütmektedir.

Görüntü işleme; analog ve dijital olmak üzere iki kategoride incelenebilir. Dijital işleme, analog işlemeye göre hatayı tolere etme yeteneği ve bozulmaya karşı gösterdiği dirençle birçok uygulamada tercih edilir duruma gelmiştir ve bu özellikleriyle yapay sinir ağı genel yapısına tam uyum göstermektedir.

Dijital görüntüleme görüntü alanındaki nesneleri, elektronik sensörleri ve görüntü kalitesi parametrelerini kullanarak görme kalitesini arttırmak için gelişmiş hesaplama tekniklerini kullanarak algılamayı amaçlayan prosestir.

Düşük karşıtlıklı (contrast), net olmayan, gürültülü görüntülerde bir çok problem ortaya çıkar ve algoritmaların algılanmasına ve verimli görüntülü işlemelerin geliştirilmesine engel olur.

Dijital görüntüleme veya bilgisayar görüntüsü, görüntü işleme ve örüntü tanıma tekniklerini içerir. Görüntü işleme teknikleri görüntü çoklamak, yönetmek ve görüntü analizleriyle ilişkilidir.Dijital görüntü işleme metotları, iki uygulama alanında ortaya çıkar:

- İnsanın algılaması ve işlemesi için görüntü içeriğinin geliştirilmesi
- Makine algılaması için arka plan verilerinin işlenmesi

Bazı görüntü işleme metotları aşağıdaki yöntemleri içerir:

- Sayısallaştırma ve sıkıştırma
- Yeniden yapılandırma, çoklama ve saklama
- Eşleştirme, tanımlama ve tanıma

Yapay sinir ağları kurallarla yönetilmeyen ve alışılmış tekniklerin başarısız olduğu veya uygun olmadığı kanıtlandığı durumlarda problem çözümlerinde başarı göstermişlerdir. Yapay sinir ağı uygulamaları, örnek tanıma (sınıflama, kümeleme, özellik seçimi), görüntü analizleri, görüntü sıkıştırma, renk gösterimi, kısımlama gibi görüntü işlemenin diğer işlemlerinde de görülür.

2.1.1 Görüntü Sistemi Tasarlama Parametreleri ve Modellemeleri

Sistem modelleme, fiziksel parametreler, geometriksel parametreler, sistem karakteristikleri, gözlemci deneyimleri, monitör parametreleri ve çeşitli faktörleri içerir. Örnek olarak, hedef tanıma için elektro-optiksel görüntüleme sisteminden söz ederken algılanan görüntü kalitesi birçok parametreden etkilenebilir.

Modelin geçerli olmadığı durumlarda görüntünün performansını tahmin etmek için modelleme kullanmak doğru olmayan tahminleri yönetebilir. Genellikle birçok teknik kullanılır ve sonuçlar birleştirilir. Örnek olarak Russo ve Ramponi çok sensörlü veri birleşimi için güçlü fuzzy metotlarını amaçlamışlardır. Benzer şekilde, nöral ağa birleşmiş psikolojik olarak motive edilmiş darbeler görüntü birleştirme mantığına dayalı modelleme birçok nesne algılama sonuçlarını birleştirerek mammografi ve otomatik hedef tanıma gibi uygulamalarda kullanılabilirler (Yıldırım, 2003).

2.1.2 Çok sensörlü görüntü sınıflama

Çok sensörlü data sınıflaması YSA uygulaması olarak birçok çalışmada kullanılmıştır. Çok sensörlü görüntü sınıflaması yapılandırılmış görüntünün eğiticili sınıflanmasına dayanır. Bu teknik aynı görüntüden bilgileri almak için çeşitli sensörlerin kullanıldığı durumlarda uygulanır. Örnek olarak: uzaktan algılama (remote sensing), tıbbi teşhislerde, görsel

denetlemelerde ve endüstriyel ürünlerin görüntülenmesinde, robotik vd (Yıldırım, 2003).

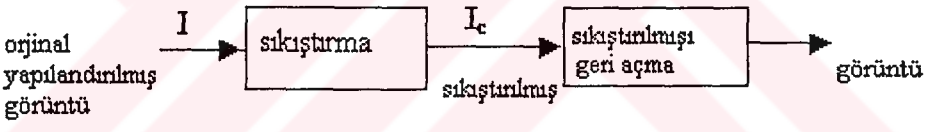
2.1.3 Desen (Pattern) Tanıma ve Sınıflama

Çok gürültülü şartlarda görüntü işlemenin en zor problemlerinden biri de pattern tanımadır. 1988'de Arsenault, YSA'nın sınıflama ve pattern tanıma performansını geliştirmek için yeni teknikler geliştirmiştir. Ağ katmanları arasında iç bağlantıyı değiştirerek ya da giriş dataların ön işlemenin ortalamalarıyla ağ içine sabit tanıtarak yüksek performans elde etmiştir. Birçok araştırmacı sinir ağlarına dayanan görüntü sınıflama sisteminin performansını daha da geliştirmeye çalışmıştır.

2.1.4 Görüntü Sıkıştırma

Görüntü sıkıştırma, görüntü işleme konusuyla her zaman ilgili olmuştur. Şekil 2.1'de genel görüntü sıkıştırma blok diyagramı verilmiştir. Mümkün olan uygulamalar aşağıdaki işlemleri içerir:

- Görüntü göndermek (archival) ve geri almak (retrival) (Medikal Görüntüleme)
- Görüntü iletimi.(telekonferans, televizyon yayınları)
- Görüntü tanıma



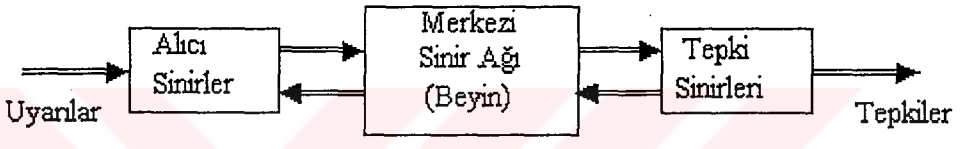
Şekil 2.1 Genel görüntü sıkıştırma blok diyagramı

Biraz bilgi kaybıyla görüntü sıkıştırmak için yapay sinir ağı tabanlı birçok yaklaşım geliştirilmiştir. Genelde, görüntü sıkıştırmada hem doğrusal hemde doğrusal olmayan sinir ağları kullanılabilir. Temelde bir veya iki katmanlı algılayıcı yapısına dayanırlar, burada ilk katman sıkıştırmayı, ikincisi de yapılandırmayı sağlar.

Panagiotidis, medikal görüntülerin sıkıştırılması için sinir ağı yaklaşımlarını kullanmıştır. Yüksek sıkıştırma oranlarına ulaşmak için görüntü alanlarının gerisinde karşıtlık bölgeleri diferansiyel olarak kodlamışlardır. (Yıldırım, 2003)

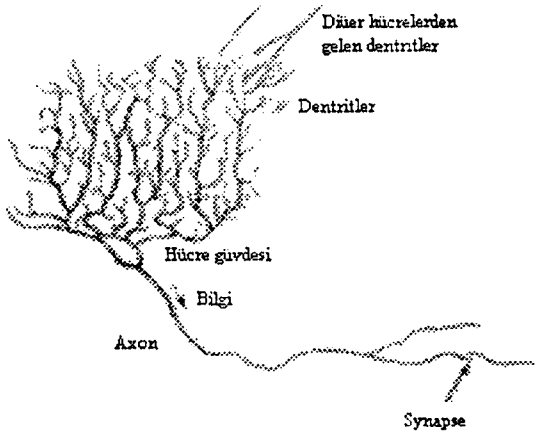
2.2 Yapay Sinir Ağları

İnsan beynini bilim dallarına uygulamak uzun zamandır bilimsel araştırmaların gündemini oluşturmaktadır. Yapay sinir ağları ya da kısaca YSA, insan beyninin çalışma sisteminin yapay olarak benzetimi çalışmalarının bir sonucu olarak ortaya çıkmıştır. İnsan beyninde 10^{10} tane sinir hücresi vardır. Hücre başına 10^4 tane bağlantı olduğu düşünülürse, insan beyninin algılamasının bile zor olacağı rakamda sinirsel bağlantı bulunmaktadır. 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 ise, beyinin ürettiği elektriksel darbeleri organizma çıktısı olarak uygun tepkilere dönüştürür. Şekil 1.1'de bir sinir sisteminin blok gösterimi verilmiştir.



Şekil 2.2 Biyolojik sinir sisteminin blok gösterimi

Merkezi sinir ağında bilgiler, alıcı ve tepki sinirleri arasında ileri ve geri besleme yönünde değerlendirilerek uygun tepkiler üretir. Bu yönüyle biyolojik sinir sistemi, kapalı çevrim denetim sisteminin karakteristiklerini taşır. Merkezi sinir sisteminin temel işlem elemanı sinir hücresidir (nöron) ve insan beyninde yaklaşık on milyar sinir hücresi olduğu tahmin edilmektedir. Sinir hücresi; hücre gövdesi, dendritler ve aksonlar olmak üzere üç 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. Şekil 2.3'de görüldüğü gibi akson-dendrit bağlantı elemanı sinaps olarak söylenir.



Şekil 2.3 Biyolojik Sinir Hücresi ve Bileşenleri.

Sinapsa gelen ve dendritler tarafından alınan bilgiler genellikle elektriksel darbelerdir ancak, sinapstaki 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 sinaps belirler.

Sinir hücreleri iki şekilde çalışırlar:

- Uyarılma: Sinir hücreleri nötr durumda –85mV'a polarize olmuşlardır.Bu değer 40mV'luk eşik gerilimini aşarsa Na iyonları hücre içine girer ve hücre uyarılır.
- Bastırma: Bu değer -90mV gibi bir değere ulaştığında K iyonları hücre dışına çıkar ve hücrenin etkisi bastırılır.

Yapay sinir ağları insan vücudundaki sinir sisteminin bazı fonksiyonlarını modelleyen ve bazı yeteneklerini yakalamak isteyen basit hesapsal birimlerin (nöronlar) yoğun bir paralel dizisidir; başka bir deyişle, teorik hale getirilmiş zeka ve beyin faaliyetlerinin matematiksel modelleridir. Ancak biyolojik sistemler o kadar karmaşıktır ki yapay sinir ağı için kullanılan biyolojik modellerin fazlaca basite indirgenmiş biçimleri şeklindedir. Yapay sinir ağları, kesin kurallarla gösterimi zor olan, daha çok algılamaya yönelik bilgileri işlemekte kullanılırlar. Olayları genelleştirme yetenekleri ve eksik, belirsiz, bozulmuş bilgileri işleyebilme ve esnek olarak çalıştırabilmeleri önemli özelliklerindendir. Bu ağlarda kullanılan basit işleme elemanları insan beyninin işleme elemanı olan nöronların modelidir. İnsan sinir sisteminin problemleri çözebilmek için öğrenme özelliği olduğu gibi yapay sinir ağlarının da bu özelliği mevcut bulunmaktadır. (Özyılmaz, 2000)

2.2.1 Yapay Sinir Ağlarının Özellikleri

Yukarıda verilen açıklamalardan, YSA'nın hesaplama ve bilgi işleme gücünü, paralel dağılmış yapısından, öğrenebilme ve genelleme yeteneğinden aldığı söylenebilir. Genelleme, eğitim yada öğrenme sürecinde karşılaşılmayan girişler için de YSA'nın uygun tepkileri üretmesi olarak tanımlanır. Bu üstün özellikleri, YSA'nın karmaşık problemleri çözebilme yeteneğini gösterir. Günümüzde birçok bilim alanında YSA, aşağıdaki özellikleri nedeniyle etkin olmuş ve uygulama yeri bulmuştur.

2.2.1.1 Doğrusal Olmama

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

2.2.1.2 Öğrenme

YSA'nın arzu edilen davranışı gösterebilmesi için amaca uygun olarak ayarlanması gerekir. Bu ayarlama, hücreler arasında doğru bağlantıların yapılmasını ve bağlantıların uygun ağırlıklara sahip olması gerektiğini ifade eder. Yapay sinir ağları, programlama yerine örneklerle eğitilir. Çocukların annelerini tanımaları için karşılaştırmalı fizyoloji hakkında hiçbir şey bilmelerine gerek olmadığı gibi, programlayıcılar da sinir ağlarına tanınacak cisimlerin nicel tanımları veya söz konusu cisimleri benzer cisimlerden ayırmak için lojik kriter kümeleri sağlamak zorunda değillerdir. Bunun yerine bir sinir ağına bazen tanımları ile beraber, cisim örnekleri de girilir. Ağ, ağırlık matrisindeki değerleri değiştirerek bunları öğrenir ve ağa bir giriş uygulandığı zaman o girişe uygun çıkış cevabı üretir.

Yapay sinir ağları eğiticili ve eğiticisiz olmak üzere iki şekilde eğitilmektedirler. Eğiticili öğrenmede ağa hem giriş hem de istenen çıkış bilgisi (hedef vektörü) girilir. Her denemeden sonra ağ kendi çıkışını doğru cevaplarla karşılaştırır ve çıkış hatası kabul edilebilecek seviyeye ininceye kadar ağırlıklarını değiştirerek tekrarlama yapar. Eğiticisiz öğrenmede hiçbir hedef vektörü yoktur. Giriş vektörü sisteme uygulanır ve sistem, girişin benzer veya ayrılan özelliklerinden yararlanarak uyumlu bir çıkış (muhtemelen eğitimden önce tahmin edilemeyen) üretecek şekilde kendisini organize eder. Böyle sistemler daha çok sınıflama ve kümeleme problemleri için kullanılmışlar ve özellikle Kohonen ile Grossberg tarafından geliştirilmişlerdir. (Lippman, 1987; Hinton, 1989, Ozyılmaz 2000)

2.2.1.3 Genelleştirme

YSA, ilgilendiği problemi öğrendikten sonra, eğitim sırasında karşılaşmadığı test örnekleri için de arzu edilen tepkiyi üretebilir. Örneğin, karakter tanıma amacıyla eğitilmiş bir YSA, bozuk karakter girişlerinde de doğru karakterleri verebilir yada bir sistemin eğitilmiş YSA modeli, eğitim sürecinde verilmeyen giriş sinyalleri için de sistemle aynı davranışı gösterebilir.

Bir yapay sinir ağının geliştirilmesindeki en kritik parametrelerden biri genelleştirme, yani ağın gelecekteki performansıdır: Ağın, eğitim kümesinde mevcut olmayan durumlar için ne kadar iyi tahminlerde bulunabildiğinin belirlenmesidir. Öğrenme süresince, eğiticili bir sinir ağının çıkışları eğitme kümesindeki girişleri verilen hedef değerlere yaklaştırır. Bu yetenek tek başına yararlı olabilir; fakat, bir sinir ağını kullanmanın amaçlarından biri genelleştirme yapmaktır. Yani, ağın çıkışlarının, eğitme kümesinde verilmeyen girişler için de hedef değerlere yaklaştırmaktır. Genelleştirme her zaman mümkün olmayabilir. Tipik olarak iyi bir genelleştirme için üç koşul gereklidir:

- Ağa uygulanan girişlerin hedefe ait yeterli bilgiyi içermesi.
- Girişleri doğru çıkışlara bağlayan yani öğrenmeyi sağlayan fonksiyonun yumuşak geçişli olması. Başka bir deyişle girişlerdeki küçük bir değişiklik çıkışlarda da küçük bir değişiklik üretmelidir.
- Eğitme durumlarının yeterince geniş ve kullanılan alt kümelerin (istatistiksel terminolojide "örnekler") genelleştirilmesi istenen (istatistiksel terminolojide "populasyon") bütün durumları temsil etmesi. (Schalkoff, 1997; Sarle, 1999, Özyılmaz 2000)

2.2.1.4 Uyarlanabilirlik

YSA, ilgilendiği problemdeki değişikliklere göre ağırlıklarını ayarlar. Yani, belirli bir problemi çözmek amacıyla eğitilen YSA, problemdeki değişimlere göre tekrar eğitilebilir, değişimler devamlı ise gerçek zamanda da eğitime devam edilebilir. Bu özelliği ile YSA, uyarlamalı örnek tanıma, sinyal işleme ve denetim gibi alanlarda etkin olarak kullanılır.

2.2.1.5 Hata Toleransı

YSA, çok sayıda hücrenin çeşitli şekillerde bağlanmasından oluştuğundan paralel dağılmış bir yapıya sahiptir ve ağın sahip olduğu bilgi, ağdaki bütün bağlantılar üzerine dağılmış durumdadır. Bu nedenle, eğitilmiş bir YSA nın bazı bağlantılarının hatta bazı hücrelerinin etkisiz hale gelmesi, ağın doğru bilgi üretmesini önemli ölçüde etkilemez. Bu nedenle, geleneksel yöntemlere göre hatayı tolere etme yetenekleri son derece yüksektir.

2.2.1.6 Donanım ve Hız

YSA, paralel yapısı nedeniyle büyük ölçekli entegre devre (VLSI) teknolojisi ile gerçeklenebilir. Bu özellik, YSA nın hızlı bilgi işleme yeteneğini artırır ve gerçek zamanlı uygulamalarda arzu edilir.

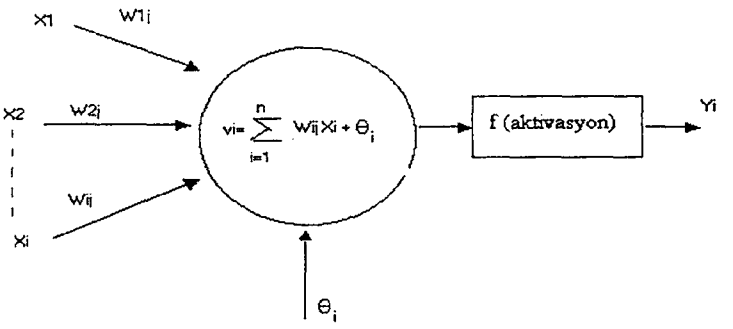
2.2.1.7 Analiz ve Tasarım Kolaylığı

YSA' nın temel işlem elemanı olan hücrenin yapısı ve modeli, daha önce açıklandığı gibi bütün YSA yapılarında yaklaşık aynıdır. Dolayısıyla, YSA' nın farklı uygulama alanlarındaki yapıları da standart yapıdaki bu hücrelerden oluşacaktır. Bu nedenle, farklı uygulama alanlarında kullanılan YSA' ları benzer öğrenme algoritmalarını ve teorilerini paylaşabilirler. Bu özellik, problemlerin YSA ile çözümünde önemli bir kolaylık getirecektir.

2.3 Yapay Hücre Modelleri

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, Şekil 2.4'de görüldüğü gibi girdiler, ağırlıklar, birleştirme fonksiyonu, aktivasyon (etkinleştirme) fonksiyonu ve çıktılar olmak üzere 5 bileşenden meydana gelir. Girdiler, diğer hücrelerden ya da dış ortamlardan hücreye giren bilgilerdir. Bilgiler, bağlantılar üzerindeki ağırlıklar üzerinden hücreye girer ve ağırlıklar, ilgili girişin hücre üzerindeki etkisini belirler. Birleştirme fonksiyonu, bir hücreye gelen net girdiyi hesaplayan bir fonksiyondur ve genellikle net girdi, girişlerin ilgili ağırlıkla çarpımlarının toplamıdır. Birleştirme fonksiyonu, ağ yapısına göre maksimum alan, minimum alan ya da çarpım fonksiyonu olabilir. Aktivasyon fonksiyonu ise birleştirme fonksiyonundan elde edilen net girdiyi bir işlemden geçirerek hücre çıktısını belirleyen ve genellikle doğrusal olmayan bir fonksiyondur. Hücre modellerinde, net girdiyi artıran +1 değerli polarma girişi yada azaltan -1 değerli eşik girişi bulunabilir ve bu giriş de sabit değerli bir giriş olarak girdi vektörü (x_0), katsayısı ise (genellikle b ile gösterilir) ağırlık vektörü (W_0) içerisine alınabilir. Genel olarak hücre modelleri Şekil 2.4'deki gibi olmakla birlikte gerçekleştirdiği işleve göre hücreler statik yada dinamik bir davranış gösterebilirler.

2.3.1 Statik Hücre Modeli



Şekil 2.4 Statik Hücre Modeli

Şekil 2.4'de; ağırlıkların sabit olduğu ve hücrede geri besleme yada geciktirilmiş sinyaller kullanılmadığı dikkate alınırsa bu hücre statik bir işlevi gerçekleştireceğinden bu model, statik hücre modeli olarak söylenebilir.

Statik hücrenin matematiksel modeli Eşitlik (2.1) deki gibi yazılabilir.

$$v=\sum_{i=0}^{x} w_i x_i \quad \text{yada} \quad v=\sum_{i=0}^{x} w_i x_i + b \tag{2.1}$$

Burada; W- hücrenin ağırlıklar matrisini, x- hücrenin giriş vektörünü, v- hücrenin net girişini, y- hücre çıkışını ve φ(.)- hücrenin aktivasyon fonksiyonunu göstermektedir. Eşitlik (2.1) den, x giriş vektörünün bileşenlerinin dış (geri beslemesiz) girişler olması durumunda hücrenin doğrusal olmayan statik bir işlevi gerçekleştireceği görülmektedir.

2.3.1.1 Aktivasyon Fonksiyonları

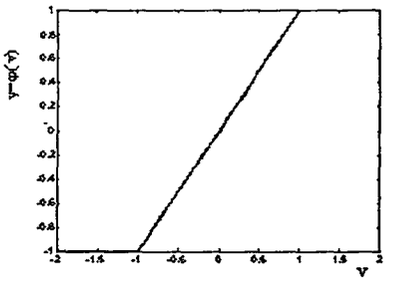
Hücre modellerinde, hücrenin gerçekleştireceği işleve göre çeşitli tipte aktivasyon fonksiyonları kullanılabilir. Aktivasyon fonksiyonları sabit parametreli yada uyarlanabilir parametreli seçilebilir.

Doğrusal ve Doyumlu-doğrusal Aktivasyon Fonksiyonu

Doğrusal bir problemi çözmek amacıyla kullanılan doğrusal hücre ve YSA' da yada genellikle katmanlı YSA' nın çıkış katmanında kullanılan doğrusal fonksiyon, hücrenin net girdisini doğrudan hücre çıkışı olarak verir. Doyumlu doğrusal aktivasyon fonksiyonu ise aktif çalışma bölgesinde doğrusaldır ve hücrenin net girdisinin belirli bir değerinden sonra

hücre çıkışını doyuma götürür. Doyumlu doğrusal aktivasyon fonksiyonunun Eşitlik (2.2)'de matematiksel tanımı, Şekil 2.5'de ise grafiği görülmektedir.

$$y = \begin{cases} 1 & v > 1 \\ v & -1 < v < 1 \\ -1 & v < -1 \end{cases} \tag{2.2}$$



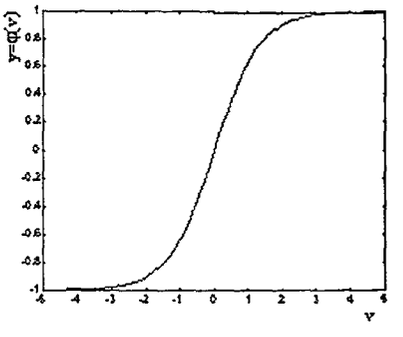
Şekil 2.5 Doyumlu doğrusal aktivasyon fonksiyonu

Sigmoid Aktivasyon Fonksiyonu

Şekil 2.6'da grafiği verilen çift yönlü sigmoid (tanh) fonksiyonu, türevi alınabilir, sürekli ve doğrusal olmayan bir fonksiyon olması nedeniyle doğrusal olmayan problemlerin çözümünde kullanılan YSA'larında tercih edilir. Çift yönlü sigmoid fonksiyonun tanımı Eşitlik (2.3)'de ve tek yönlü sigmoid fonksiyonunun matematiksel ifadesi ise Eşitlik (2.4)'de verilmiştir.

$$\varphi(v) = a\frac{1 - e^{\lambda v}}{1 + e^{\lambda v}} \tag{2.3}$$

$$\varphi(v) = a\frac{1}{1 + e^{\lambda v}} \tag{2.4}$$

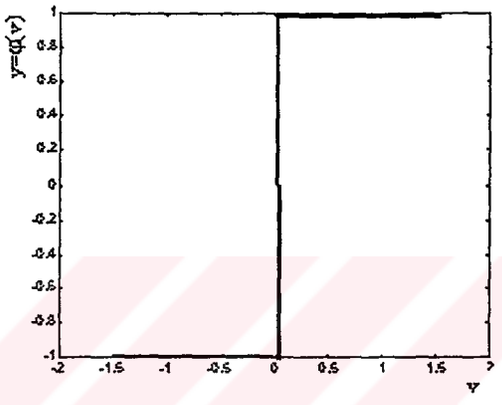


Şekil 2.6 Sigmoid (tanh) aktivasyon fonksiyonu.

Eşik Aktivasyon Fonksiyonu

McCulloch-Pitts modeli olarak bilinen eşik aktivasyon fonksiyonlu hücreler, mantıksal çıkış verir ve sınıflandırıcı ağlarda tercih edilir. Şekil 2.7'de verilen ve Perseptron (Algılayıcı) olarak da söylenen eşik fonksiyonlu hücrelerin matematiksel modeli aşağıdaki gibi tanımlanabilir.

$$y = \begin{cases} 1 & v \geq 0 \\ -1 & v < 0 \end{cases} \tag{2.5}$$



Şekil 2.7 Eşik aktivasyon fonksiyonu

2.3.2 Dinamik Hücre Modelleri

Şekil 2.3 de verilen yapay hücre modeli, x girişlerinden y çıkışlarına doğrusal olmayan statik bir dönüşümü gerçekleştirir. Örüntü tanıma ve sınıflandırma uygulamalarında statik hücre yada YSA modelleri uygun olmakla birlikte sistem modelleme ve denetimi gibi dinamik problemlerin çözümünde dinamik hücre yada YSA yapılarının kullanılması gereklidir.

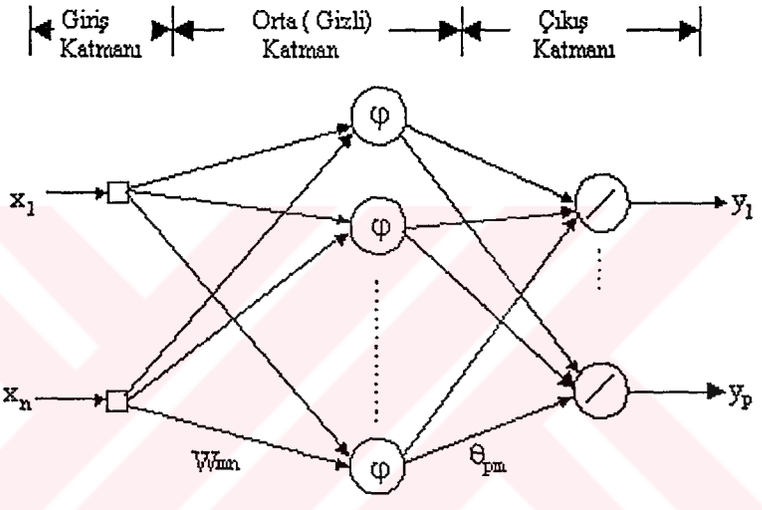
2.4 Yapay Sinir Ağı Yapıları

Yapay sinir ağları, hücrelerin birbirleri ile çeşitli şekillerde bağlanmalarından oluşur. Hücre çıkışları, ağırlıklar üzerinden diğer hücrelere ya da kendisine giriş olarak bağlanabilir ve bağlantılarda gecikme birimi de kullanılabilir. Hücrelerin bağlantı şekillerine, öğrenme kurallarına ve aktivasyon fonksiyonlarına göre çeşitli YSA yapıları geliştirilmiştir.

2.4.1 İleri Beslemeli Yapay Sinir Ağları (İBYSA)

İleri beslemeli YSA'da, hücreler katmanlar şeklinde düzenlenir ve bir katmandaki hücrelerin çıkışları bir sonraki katmana ağırlıklar üzerinden giriş olarak verilir. Giriş katmanı, dış

ortamlardan aldığı bilgileri hiçbir değişikliğe uğratmadan orta (gizli) katmandaki hücrelere iletir. Bilgi, orta ve çıkış katmanında işlenerek ağ çıkışı belirlenir. Bu yapısı ile ileri beslemeli ağlar doğrusal olmayan statik bir işlevi gerçekleştirir. İleri beslemeli 3 katmanlı YSA'nın, orta katmanında yeterli sayıda hücre olmak kaydıyla, herhangi bir sürekli fonksiyonu istenilen doğrulukta yaklaştırabileceği gösterilmiştir. En çok bilinen geriye yayılım öğrenme algoritması, bu tip YSA'ların eğitiminde etkin olarak kullanılmakta ve bazen bu ağlara geriye yayılım ağları da denmektedir. Şekil 2.8'de giriş, orta ve çıkış katmanı olmak üzere 3 katmanlı ileri beslemeli YSA yapısı verilmiştir.



Şekil 2.8 İleri beslemeli 3 katmanlı YSA.

Herhangi bir problemi çözmek amacıyla kullanılan YSA da, katman sayısı ve orta katmandaki hücre sayısı gibi kesin belirlenememiş bilgilere rağmen nesne tanıma ve sinyal işleme gibi alanların yanı sıra ileri beslemeli YSA, sistemlerin tanımlanması ve denetiminde de yaygın olarak kullanılmaktadır.

2.4.1.1 Tek katmanlı ileri beslemeli ağlar

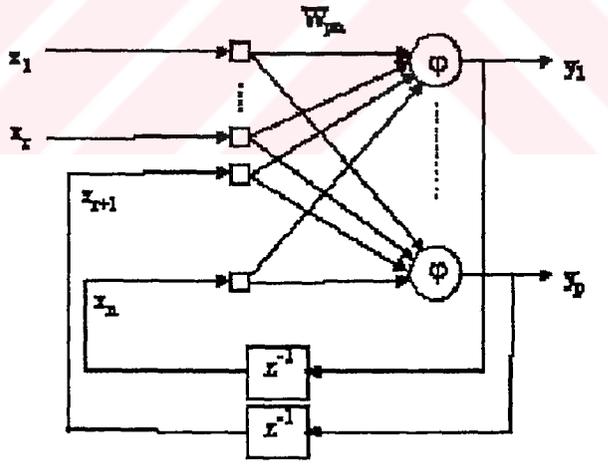
Genelde tek bir hücre istenilen giriş çıkış işlevini yerine getiremez. Bunun için bağlantı ağırlıkları dışında aynı özelliklere sahip hücreler bir araya getirilerek katmanlar oluşturulur. Buradaki tek katman terimi, nöronların çıkış katmanına karşı düşmektedir. Herhangi bir hesaplamanın yapılmadığı giriş katmanı göz önöne alınmaz. Bu tür ağlar algılayıcı (perceptron) olarak bilinir (Hertz vd, 1991; Haykin, 1994; Zurada, 1995).

2.4.1.2 Çok katmanlı ileri beslemeli ağlar

Çok katmanlı ileri beslemeli ağlar, tek katmanlı ağların arka arkaya bağlanmasmdan oluşurlar. Bir katmanın girişi, bir önceki katmanın çıkışıdır. Giriş katmanı sadece girişi çoğullamaktadır yani girişi ile çıkışları aynıdır. Giriş ile çıkış arasında kalan katmanlara gizli katmanlar adı verilmektedir. Genelde bir veya iki gizli katman kullanmak yeterli olmaktadır. Gizli katmanlardaki hücre sayıları farklı olabilir. Çok katmanlı ağlar karmaşık karar ayrılmasında kullanılırlar (Haykin, 1994; Zurada 1995).

2.4.2 Geri Beslemeli Yapay Sinir Ağları (GBYSA)

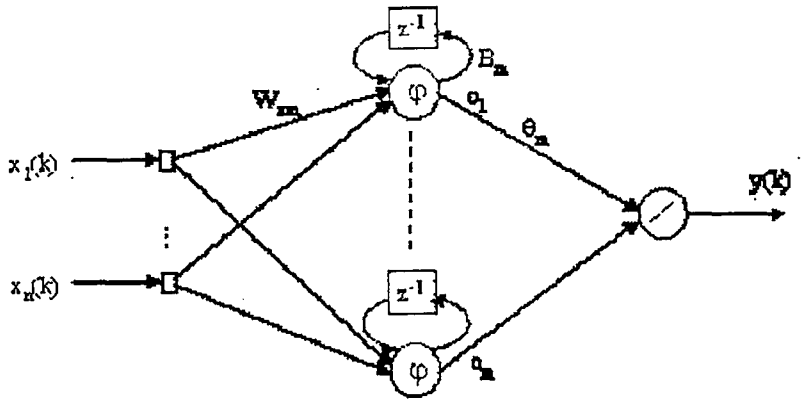
Geri beslemeli YSA' da, en az bir hücrenin çıkışı kendisine ya da diğer hücrelere giriş olarak verilir ve genellikle geri besleme bir geciktirme elemanı üzerinden yapılır. Geri besleme, bir katmandaki hücreler arasında olduğu gibi katmanlar arasındaki hücreler arasında da olabilir. Bu yapısı ile geri beslemeli YSA, doğrusal olmayan dinamik bir davranış gösterir. Dolayısıyla, geri beslemenin yapılış şekline göre farklı yapıda ve davranışta geri beslemeli YSA yapıları elde edilebilir. Bu nedenle, bu bölümde bazı geri beslemeli YSA yapılarında örnekler verilecektir. Şekil 2.9'da iki katmanlı ve çıkışlarından giriş katmanına geri beslemeli bir YSA yapısı görülmektedir.



Şekil 2.9 Geri Beslemeli İki Katmanlı YSA.

Şekil 2.9'da verilen geri beslemeli YSA da giriş vektörü, r adet dış giriş ve p adet gecikmiş ağ çıkışlarından oluşmaktadır.

Geri beslemeli sinir ağları; hücreler arası ya da katmanlar arası geri besleme yapılış şekline göre farklı isimlerle söylenir. Genellikle derecesi bilinmeyen dinamik sistemlerin tanımlanmasında kullanılan diğer bir YSA yapısı, gizli katman hücrelerinde öz geri beslemenin kullanıldığı ve yöresel geri-küresel ileri beslemeli (YGKİ) olarak söylenen Şekil 2.10'da verilen YSA' dır.

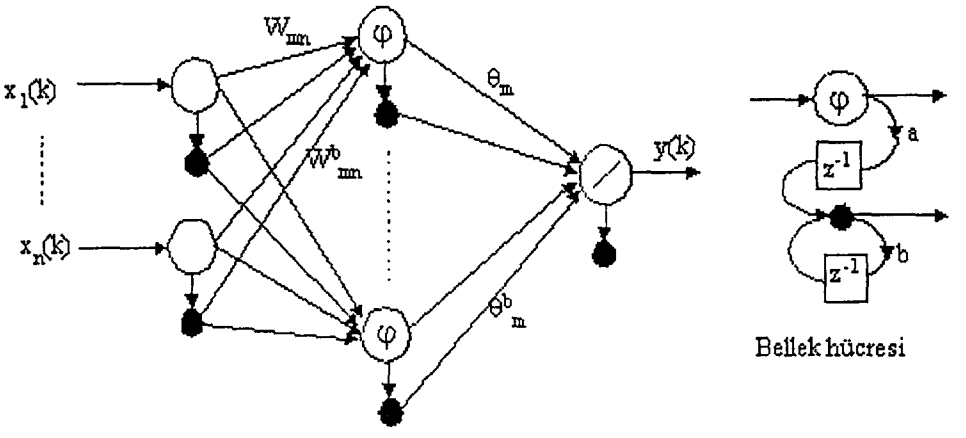


Şekil 2.10 YGKI yapay sinir ağı

YGKİ ağlar, ileri beslemeli YSA' nın eğitim algoritmalarında gerçekleştirilen küçük değişikliklerle eğitilebilmeleri nedeniyle ileri ve geri beslemeli YSA' nın ortak özelliklerini taşımaktadır. Özellikle bozucu ve ölçülemeyen girişleri olan dinamik sistemleri modellemek amacıyla kullanılmış ve başarılı sonuçlar alınmıştır.

2.4.3 Bellek Hücreli YSA Yapıları (BHYSA)

Doğrusal olmayan sistemlerin tanılanması ve denetiminde, katmanlı YSA yapıları etkin olarak kullanılmaktadır. YSA ile sistem tanılamada, doğru model yapısının seçilebilmesi ve model girişlerinin belirlenebilmesi için sistemin giriş ve çıkışının gecikme derecelerinin bilinmesi gerekir. Sistemin derecesinin doğru belirlenememesi, modelde temsil edilemeyen dinamikler nedeniyle kararlı ve değişen dinamik şartlarda doğru bir model elde edilmesini etkiler. Bu nedenle, geri beslemeli YSA yapıları kullanılarak sistemin derecesine ihtiyaç duymayan tanı modelleri geliştirilmiştir. Şekil 2.11'de Bellek Hücreli Yapay Sinir Ağları (BHYSA) olarak söylenen ve ağdaki her bir hücre için bir bellek hücresinin kullanıldığı katmanlı-geri beslemeli bir ağ yapısı verilmiştir.



Şekil 2.11 Bellek hücreli yapay sinir ağı ve bellekli bir hücrenin yapısı.

BHYSA'da her bir ağ hücresine ait olan bellek hücresi, bir ağırlık (b_j) üzerinden öz geri besleme girişine ve başka bir ağırlık üzerinden (a_j) ait olduğu hücrenin gecikmiş girişine göre bir çıkış üretir. Çıkış katmanında ise genellikle sadece öz geri besleme kullanılır.

BHYSA'nın doğrusal olmayan bir sistemi modelleme ve denetim yeteneği, sadece sistemin o anki giriş ve bir önceki çıkış verileri model girişi alınarak incelenmiş ve tatmin edici sonuçlar alındığı gösterilmiştir. İleri beslemeli katmanlı YSA'nın sadece gizli katmanında bellek hücreleri kullanılarak bellek hücresinin, ait olduğu hücre çıkışının geçmişteki örneklerini giriş olarak aldığı ve zaman gecikmeli YSA olarak söylenen geri beslemeli ağ yapıları da incelenmiştir.

2.5 Yapay Sinir Ağlarında Öğrenme

McCulluch ve Pitts'in sinir ağının temel problemi olan öğrenmenin nasıl olacağına, Donald Hebb 1949 yılında yayınladığı "The Organization of Behavior" isimli kitabında çözüm getirmiştir. Bu kitapta sinapsların şartlara uyum sağlama yeteneği incelenmiş ve Hebb kuralı adı verilen öğrenme kuralı anlatılmıştır. Bu kuralda temel fikir iki sinir bağlantısı arasındaki kuvvet (ağırlığın değeri), sinirlerin aynı zamanda etkinleştirmesine bağlıdır. Hebb, sinir faaliyetlerini örnek alarak bunların hafızadaki basit bir yerde yerleşebileceğini varsaymış ve bu kurama göre sinirlerin birbirlerini ortaklaşa uyardıklarını ve bu uyarım neticesinde aralarındaki sinaptik bağlantı kuvvetinin kendi etkinlikleri çarpımı oranında artacağını ortaya koymuştur.

Yapay sinir ağlarında bilgi, ağdaki bağlantıların ağırlıklarında depolanır. Bir ağda öğrenme kısaca, istenen bir işlevi yerine getirecek şekilde ağırlıkların ayarlanması sürecidir. Yapay

sinir ağlarınında öğrenme sinirler arasındaki ağırlıkların değiştirlmesi ile gerçekleşmektedir. Buna göre sinirler arası bağlantılar üzerindeki ağırlıkların, belirli bir yöntem (öğrenme kuralları) uyarınca dinamik olarak değiştirilebilen ağlar eğitilebilir. Eğitilebilen yani öğrenebilen ağlar, yeni şekilleri tanıyabilir veya verilen bir girişin hangi sınıfa ait olduğuna karar verebilir (sınıflandırma). Yapay sinir ağlarında öğrenme düğümler arasındaki ağırlıkların, düğümlerdeki etkinlik veya aktivasyon işlevlerinin değişkenlerinin ayarlanmasıyla yapılmaktadır.

1950'li yıllardan bu yana bir çok araştırmacı Hebb'in kurallarını temel alarak öğrenmenin nasıl daha iyi olacağı konusunda araştırmalarını sürdürmektedir. Bu araştırmacılar genellikle çalışmalarında Hebb'in ortaya koyduğu sinirler arasındaki metabolik değişme ve sinaptik kuvvetin yani öğrenme değişkenleri ve ağırlıkların nasıl ayarlanacağı konusunu ele almışlardır ve bunlara uygun öğrenme yöntemleri geliştirmişlerdir. Temelde bu öğrenme yöntemleri, eğiticili ve eğiticisiz olmak üzere iki gruba ayrılmıştır.

2.5.1 Eğiticili Öğrenme

Yapay sinir ağlarında gerçek çıkış istenen çıkışla kıyaslanır. Rasgele değişen ağırlıklar ağ tarafından öyle ayarlanır ki, bir sonraki döngüde gerçek çıkış ile istenen çıkış arasında daha yakın karşılaştırma üretebilsin. Öğrenme yöntemi, bütün işleme elemanlarının anlık hatalarını en aza indirmeye çalışır. Bu hata azaltma işlemi, kabul edilebilir doğruluğa ulaşana kadar ağırlıklar devamlı olarak derlenir.

Eğiticili öğrenmede, yapay sinir ağı kullanılmadan önce eğitilmelidir.Eğitme işlemi, sinir ağlarına giriş ve çıkış bilgileri sunmaktan oluşur. Bu bilgiler genellikle eğitme kümesi olarak tanımlanır. Yani, her bir giriş kümesi için uygun çıkış kümesi ağa sağlanmalıdır.

Birçok uygulamada,a ağa gerçek veriler uygulanmak zorundadır. Bu eğitme safhası uzun zaman alabilir. Sinir ağı, belirli bir sıralamadaki girişler için istenen istatistiksel doğruluğu elde ettiği zaman eğitme işlemi tamamlanmış kabul edilir ve eğitme işlemi bitirilir. Öğrenim aşaması tamamlandıktan sonra ağ kullanılmaya başlandığında, bulunan ağırlıkların değeri sabit olarak alınır ve bir daha değiştirilemezler. Bazı ağ yapılarında ağ çalışırken çok düşük oranda eğitmeye izin verilir. Bu işlem ağların değişken koşullara uyum sağlamasına yardımcı olur.

Eğer sistemin önemli olan özellikleri ve ilişkileri öğrenmesi gerekiyorsa, o zaman eğitme kümesi, bütün ihtiyaç duyulan bilgileri içermesi gerekir. Eğer ağ sadece bir örnekle eğitilirse,

bir olay için çok hassas olan bütün ağırlıklar kümesi, bir sonraki olayda yeterli çözüm vermez. Yeni şeyler öğrenme safhasında eski olaylar unutulabilir. Sonuç olarak, sistem gerekli bilgilerle birlikte öğrenmek zorundadır.

Giriş ve çıkış bilgilerinin nasıl sunulacağı veya nasıl kodlanacağı, bir ağı başarılı bir şekilde yönlendirmek için önemli bir unsurdur. Yapay sinir ağları sadece sayısal giriş bilgileri ile çalışırlar. Bu yüzden ham bilgiler genellikle ölçeklendirilmelidir.

Eğiticili öğrenmede giriş ve çıkış çiftlerinden oluşan eğitim bilgileri vardır. Ağ giriş bilgisine göre ürettiği çıkış değerini, istenen değerle karşılaştırarak ağırlıkların değiştirilmesinde kullanılacak bilgiyi elde eder. Girilen değerle istenen değer arasındaki fark hata değeri olarak önceden belirlenen değerden küçük oluncaya kadar eğitime devam edilir. Hata değeri istenen değerin altına düştüğünde tüm ağırlıklar sabitlenerek eğitim işlemi sonlandırılır. Eğitim işlemi sırasında her bir eğitim bilgisi çifti için oluşan hata değerine göre ağırlıkların değiştirilmesine "örüntü kipi" öğrenme, tüm eğitim kümesi için hataların toplanarak toplam hata değerine göre ağırlıkların değiştirilmesine ise "küme kipi" öğrenme denilmektedir.

Eğer verilen girişe karşılık amaçlanan çıkış üretilemiyorsa, ağın çıkış değerindeki hatayı en küçükleyecek şekilde bağlantı ağırlıklarının değiştirilmesi sağlanır (Elmas, 2003).

2.5.2 Eğiticisiz Öğrenme

Eğiticisiz öğrenmede sistemin doğru çıkış hakkında bilgisi yoktur ve girişlere göre kendi kendisini örnekler. Eğiticisiz olarak eğitilebilen ağlar istenen ya da hedef çıkış olmadan giriş bilgilerinin özelliklerine göre ağırlık değerlerini ayarlar.

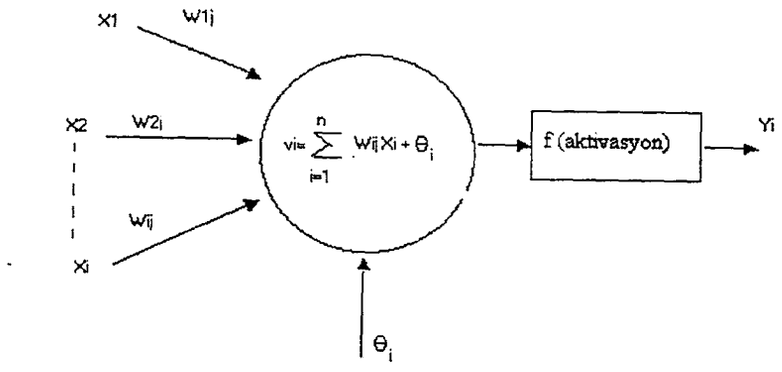
Eğiticisiz öğrenmede ağ istenen dış verilerle değil girilen bilgilerle çalışır. Bu tür öğrenmede gizli nöronlar dışarıdan yardım almaksızın kendilerini örgütlemek için bir yol bulmalıdırlar. Bu yaklaşımda, verilen giriş vektörleri için önceden bilinebilen performansını ölçebilecek ağ için hiçbir çıkış örneği sağlanmaz, yani ağ yaparak öğrenmektedir (Elmas,2003).

2.5.3 Öğrenme Kuralının Kavranması

Yapay sinir ağları beyne benzetilmiş sistemlerin basitleştirilmiş matematik modelleri gibi düşünülebilir. Bu işlem paralel bağlı bilgisayar ağı gibidir. Fakat, programlarla özel iş yapan geleneksel bilgisayarların aksine eğitilebilir. Yeni olayları, yeni bağlantıları ve yeni işlemsel bağlantıları öğrenebilir.

Yapay sinir ağları temel eleman olan işlem elemanı veya düğüm diye adlandırılan sinirlerden

oluşurlar. Şekil 2.12'de bir yapay sinir (düğüm, işlem elemanı) görülmektedir.



Şekil 2.12 Bir yapay sinir (düğüm)

Sinyaller tek yönlü olarak x_j sinir girişlerinden ağa girer. Sinir çıkış sinyali aşağıdaki eşitlikte verilir.

$$yi = f(w,x) = f(w^T x) = f\left(\sum_{j=1}^{n} w_{ij} x_i\right) \qquad (2.6)$$

Burada $w_i = (w_1,...,w_n)^T \in R^n$ ağırlık vektörüdür. İşlev f (w^T x) etkinlik (aktarım) işlevidir. Ağın değerlerii giriş vektörü ile tanımlanır.

$$ağ = w^T x = w_1 x_1 + ... + w_n x_n \qquad (2.7)$$

Burada ilk durum çıkış değeri sıfir ise aşağıdaki eşitlik ile hesaplanır.

$$y_i = f(ağ) = \begin{cases} 1 & \text{eğer } w^T x \\ 0 & farklı \end{cases} \qquad (2.8)$$

Şekil 2.12'de görülen θ, eşik düzeyi olarak adlandırılır ve bu tip düğüme düzenli eşik düzeyi denir.

2.5.4 Öğrenme Oranları

Yapay sinir ağlarındaki öğrenme oranı denetlenebilir birkaç etkene bağlıdır. Daha düşük bir öğrenme oranı yeterli derecede eğitilmemiş bir sistem üretmek için çok daha fazla zaman harcanacağı anlamına gelmektedir. Daha yüksek öğrenme oranındaki bir ağ, daha yavaş öğrenen bir ağa göre gerçek ayrımlar yapamayabilir.

Bu etkenler, bir ağın eğitilmesinde ne kadar süre alacağını belirlemede önemli bir rol oynar. Bu etkenlerin herhangi birini değiştirmek, eğitme süresini çok fazla arttırabilir ve hatta çok büyük hatalara sebep olabilir.

Bir çok öğrenme işleminde, öğrenme oranı veya öğrenme sabiti kullanılır. Öğrenme oranı genellikle pozitif ve sıfır ile bir arasında bir değer olmaktadır. Eğer birden büyük öğrenme oranı seçilirse ağırlıkları uyarlayacak öğrenme algoritmasını ayarlamak kolay olur ancak ağda salınımlar ortaya çıkar. Küçük öğrenme oranı, anlık hataları hızlı bir şekilde düzeltilmemesine karşın en iyi sonuca ulaşma şansını yükseltmektedir.

2.5.5 Öğrenebilen Algoritmaların Kavranması

Bir algoritmanın öğrenebilmesi için izlenecek yol şöyle özetlenebilir. Eğitim kümesinden K eğitim parçaları alındığında

$$(x1, y1), \ldots, (x^K, y^K) \tag{2.9}$$

şeklinde gösterilir. Burada $x^K = (x_1^K, \ldots, x_n^K)$, $y^K = (y_1^K, \ldots, y_n^K)$, k = 1,....., K'dır.

Bu durumda;

Adım 1 n>0 seçilir.

Adım 2 w_i ağırlık vektörü küçük rasgele değerler seçilir, çalışma hatası κ: = 1 iken çıkış *E*'ye ayarlanır

Adım 3 Eğitim burada başlar, x^K değeri x: = x^K, y: = y^K olur ve çıkış hesaplanır.

$$y_i(x) = sign(w_i, x),\ i = 1, \ldots, m \tag{2.10}$$

Adım 4 Ağırlık güncellenir.

$$w_i = w_i + \eta(y_i - sign(w_i, x))x,\ i = 1, \ldots, m \tag{2.11}$$

Adım 5 Her bir dönüşte hata E ile sunulan hataya eklenerek hesaplanır.

$$E := E + \frac{1}{2}\|v_i - z_i\|^2 \tag{2.12}$$

Adım 6 Eğer κ: =κ+1'den $\kappa < K$ ise üçüncü adıma dönülür. Aksi takdirde Adım 7'ye geçilir.

Adım 7 Eğitim aşaması tamamlanır. $E = 0$ için eğitim çalışması durdurulur. Eğer $E>0$ ise E çıkışta κ: =1 gibi bir dizidir ve yeni bir adım için Adım 3'e dönülür.

3. YAPAY SİNİR SİSTEMLERİ İÇİN ANALOG VLSI DEVRELER

Yapay sinir ağlarında birçok tipte analog nöron modeli bulunmaktadır. Bu modellerin ortak özelliği nöronun sigmoidal bir nonlineerlik göstererek kendine gelen sinapslardan işareti alabilmesi (Kirchoff'un akımlar yasası) ve kendinden sonra gelen sinapslara bu işareti iletebilmesi gerekir.

3.1 Nöronlar İçin Analog Devreler

3.1.1 Analog Eviriciler

En basit analog nöron modeli iki eşlenik (p ve n kanal) transistör içeren CMOS eviricidir. Nöronun kazancı transistörlerin iletkenlikleri dolayısıyla da W/L oranlarına da bağlıdır. Ancak evirici çip haline getirildikten sonra dışarıdan elektronik olarak kazancının ayarlanabilmesi mümkün değildir.

3.1.2 Analog Çarpıcı Tabanlı Nöronlar

Basit geçiş iletkenliği kuvvetlendiricisi ya da Gilbert çarpıcısı kullanarak daha esnek nöronlar elde edilebilir. Carver Mead, transistörleri eşik altı bölgesinde çalışan ve bu sayade özellikle görsel tasarımlar için önemli olan güç tüketimini düşük tutabilen bu ve buna benzer birçok devre yapısı önermiştir. Diğer uygulamalarda ise transistörler eşiküstü bölgesinde çalıştırılmış, güç tüketimini arttığı gözlenmiş ancak bu sayede hızın da arttığı belirtilmiştir. Bir yapay sinir ağında sinapsların sayısına karşılık nöronların sayısı oldukça az ise nöronların alan minimizasyonu temel amaç değildir. Bu durumda nöronlar için daha ayrıntılı devreler olan Gilbert geniş-dağılımlı analog çarpıcı devreleri kullanmak daha avantajlı olmaktadır. Bu çarpıcıların çıkış akımının hiperbolik tanjant fonksiyonunun nonlineerliğine oldukça iyi yaklaştığı görülmüştür.Ayrıca bu nöronların oldukça iyi kazanca sahip olarak akım kaynağı gibi davrandığı ve bu sayede kendinden sonra gelen devreyi sürme yeteneğinin arttığı gözlemlenmiştir. Dikkat edilmelidir ki bu devrenin geniş gerilim dağılımı sağlayabilmesi transistörlerinin eşiküstü bölgesinde çalışması sayesinde olmuştur (Haykin, 1994).

3.2 Sinapslar İçin Analog Devreler

$x_i = \Sigma W_{ij}.V_j$ gibi nöral aktivasyon değerlerinin toplamında ve bazı öğrenme kurallarının gerektirdiği hesaplamalarda analog yöntemler kullanılmaktadır. Bunun için de öncelikle analog çarpıcı yapıları incelenmektedir.

3.2.1 Basit Çarpıcılar

Bir MOS transistörde I_D akımı aşağıdaki şekildeki gibi verilebilir.

$$I = 2a.\beta.V_1.V_2 \tag{3.1}$$

Burada a sabit bir değer, V_1 drain gerilimi, $V_2 = V_g - V_t$, V_t eşik gerilimi, β iletkenlik parametresi $\beta = \mu.Cox.(W/L)$ ile verilmektedir. Bu yaklaşım unipolar ağırlıklar ve $0<V_j<1$ aralığında değişen unipolar nöral aktivasyon değerleri için direk olarak devreye uygulanabilir. Ancak bipolar aktivasyon değerleri ve ağırlıkları için devreye her sinaps başına birkaç transistör daha eklemek gerekebilir. EEPROM teknolojisi bipolar çarpıcıları iki transistör kullanarak da sağlayabilmektedir.

3.2.2 İkili Hafızalarda Ağırlıkların Depolanması

Aktivasyon değerinin hesaplanabilmesi için ayrı bir yöntemdir ancak prensibi dijital çalışmaya dayanmaktadır. Bu sistemde ağırlıklar ikili hafizalarda depolanır ve hafizanın içerikleri D/A çarpıcı sayesinde analog akım çıkışı sağlanır. Ancak bu yöntem çip alanının oldukça geniş tutulmasına sebep olabilir.

3.2.3 Ağırlıkların Kapasite Üzerinde Yük Olarak Depolanması

Bu yaklaşımla sinaptik alanın az yer tutması sağlanabilir. Bir diğer özelliği ise ağırlıkların saklanması ve güncellenmesi de kapasiteler tarafından sağlandığı için tamamen analog bir yapıya sahip olmasıdır. Dezavantajı ise; kapasiteler üzerindeki gerilim küçük sızıntı akımları yüzünden zamanla bozulacağından, ağırlıkların periyodik olarak yenilenmek zorunda oluşudur. Öğrenme devresi yoluyla ağırlıkların sürekli yenilenmesi devreyi dış etmenlerin değişmesi koşulunda bile çalışmasını sağlamaktadır.

3.2.4 EEPROM Tabanlı Ağırlıklar

EEPROM'lar yükteki değişmeyi önlemek için Floating Gate (FG) uygularlar. FG yükü genellikle tünel akımlarını kullanarak modifiye edilirler ancak ultraviyole ışınları da endirek olarak analog ağırlıkların değişmesine neden olabilmektedirler.

VLSI teknolojisi ile yapay sinir ağları aşağıdaki iki önemli nedenden ötürü birbirleriyle oldukça iyi örtüşmektedirler (Baser, 1992):

- VLSI teknolojisi ile yüksek fonksiyonel yoğunluğa erişilebilmiş hatta bir çip üzerinde nöronları çalıştırılması başarılabilinmiştir.

- Sinir ağlarının sıradan topolojisi ve az sayıdaki iyi tanımlanmış aritmetik çalışmaların (öğrenme algoritmalarındaki) önemli ölçüde VLSI devrelerin tasarımını ve serimini basitleştirmektedir.

Günümüzde çok katmanlı perceptronlarla kurulan genel amaçlı çipler mevcuttur. Boltzman makineleri, ortalama-alan-teorili makineler ve kendi kendine organize olan sinir ağları vs...

3.3 Temel Tasarım Kavramları

Arttırılabilir fonksiyonel yoğunluğu, kullanım kolaylığı ve düşük maliyeti CMOS teknolojisinin Yapay sinir ağlarının VLSI uygulamalarında kullanımını arttırmıştır. Gerek genel amaçlı çipler, gerekse özel amaçlı çipler için tasarım kriterleri aşağıdaki gibi özetlenebilir.

Toplamların Hesaplanması

Bütün nöronlar için fonksiyonel bir gereksinimdir. Her bir giriş vektörünün bir ağırlıkla çarpılması ve daha sonra bu çarpımların toplanmasını sağlar.

$$V_j = \sum W_{ji} X_i \tag{3.2}$$

Burada W_{ji}, sinapsın ağırlığı; X_i, i. sinapsa uygulanan impuls ve V_j, aktivasyon potansiyelini belirtmektedir.

Bilgi Sunumu

Genellikle sinir ağları düşük doğruluk içeren bilgilere yada uygulamaya /algoritmaya bağlı bilgilere sahiptir.

Çıkış Hesaplaması

Çıkış nöronu için en genel aktivasyon fonksiyonu nonlineer bir fonksiyondur. Örneğin sigmoid fonksiyonu veya logistic fonksiyonu bazende eşik fonksiyonudur.

Öğrenim Karmaşıklığı

Her öğrenme algoritması kendine has bazı hesaplamalar gerektirebilmektedir. Bir çok öğrenme algoritması sinaptik ağırlıklarda güncelleme yaparken yerel hesaplamalara yer verir ancak bazı algoritmalar ise ek gereksinimlere ihtiyaç duyar.

Ağırlıkların Depolanması

Yapay sinir ağlarında eski ağırlık değerlerinin bir yerlerde saklanması gerekli ki yeni ağırlıklar hesaplanırken güncelleme değeri eski ağırlıklara eklenebilsin.

İletişim

Eğer nöronlar arası bağlantının band genişliğine yararı düşük ise silikon yüzeye göre pahalı olan metalde verimsizliğe yol açar. Bağlantılılık bir silikon çip üretiminde belki de en ciddi sınırlamadır.

3.4 VLSI Uygulama Kategorileri

Yapay sinir ağları analog uygulamalarında bilgi işareti sürekli bir genliğe sahiptir, diğer yandan sayısal uygulamalarda giriş işareti belirli ayrık işaret kümelerinden oluşmaktadır. Her iki yaklaşımı birden içeren hybrid kombinasyon sinir ağını oluştururken diğer temalarında özellikleri sağlanır (Haykin, 1994).

3.4.1 Analog Teknikler

1987'de IEEE yayınlarından birinde Carver Mead, VLSI teknolojisi ile yapay sinir ağlarının birleştirilebileceğini savunmuştur. VLSI teknolojisinin temel yapı taşlarından biri de MOS transistördür. MOS transistörü anlayabilmek için V_{GS} gerilimine bağlı I_D fonksiyonunu incelemek gereklidir. Bu durumda MOS transistör genel olarak iki bölümde incelenir.

- Eşiküstü Bölgesi: I_D, V_{GS}'nin kuadratik fonksiyonudur.
- Eşikaltı Bölgesi : I_D, V_{GS}'nin eksponansiyel fonksiyonudur.

Aynı şartlar altında eksponansiyel nonlineerlik yani eşikaltı bölge tercih edilir, çünkü birim akım başına düşen geçiş iletkenliği daha fazladır. Bundan da öte eşikaltı bölgesinde MOS transistör iki kullanışlı fonksiyon daha sağlar.

- Küçük V_{DS} gerilimleri için transistör mükemmel bir simetri ile kontrollü iletken gibi davranır. Bu mod lineer bölge olarak adlandırılır.
- Daha büyük V_{DS} gerilimleri için transistör gerilim kontrollü akım kaynağı gibi davranır.

Nöromorfik ağların analog VLSI uygulamalarında MOS transistörler eşikaltı bölgesinde çalıştırılırlar. CMOS transistörlerin eşikaltı çalışmaları aşağıdaki yararlı karakteristikleri ortaya çıkarır.

- Eksponansiyel akım kazancı 10pA-10μA

- Gerilim-akım çevirimi (eksponansiyel) veya akım-gerilim çevirimi (logaritmik) tek bir transistör tarafından sağlanabilir.
- Düşük güç tüketimi(10^{-12} – 10^{-6} W)

Böyle bir nörobilgisayar tarafından yapılan analog hesaplamalar zamana, gerilime, akıma bağlı fonksiyonlar şeklindedir. Bu fonksiyonlar eleman düzeyinde, devre düzeyinde ve yapısal düzeyde aşağıdaki şekilde tanımlanmışlardır (Haykin, 1994).

Eleman Düzeyinde

MOS transistörün n kanal yada p kanal olmasına göre transistör eşikaltı bölgesinde aşağıdaki gibi davranır.

n kanal için;

$$I_D = I_0 \exp\left(\frac{kV_G}{U_T}\right)\left[\exp\left(-\frac{V_S}{U_T}\right) - \exp\left(-\frac{V_D}{U_T}\right)\right] \tag{3.3}$$

p kanal için;

$$I_D = I_0 \exp\left(-\frac{kV_G}{U_T}\right)\left[\exp\left(\frac{V_S}{U_T}\right) - \exp\left(\frac{V_D}{U_T}\right)\right] \tag{3.4}$$

Burada I_0,sıfır bias akımı; k, gövde etkisi parametresi; $U_T=k_bT/q$ ısısal gerilim; k_b, Boltzmann sabiti; T, Kelvin cinsinden sıcaklık ve q, elektrik yüküdür.

Devre Düzeyinde

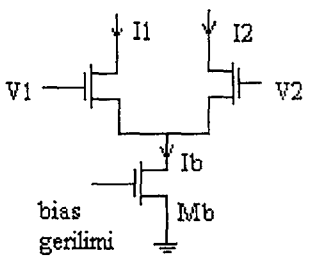
Bu düzeyde önemli olan yük ve enerjinin korunumudur. Kirchoff'un akımlar ve gerilimler yasası tamamen korunur. ($\Sigma Ii=0$; $\Sigma Vi=0$)

Yapısal Düzey

Bu düzeyde ise daha çok matematiksel diferansiyel denklemler kullanılarak gerekli fonksiyonlar tanımlanır.

Andreu'nun 1992'de tanımladığı analog yaklaşım minimalistik tasarımı amaçlamaktadır. Bu yaklaşıma göre yalnız bir transistör bile kazanç ve bazı başka fonksiyonları

Ancak bunun tersi olan Mead'in yaklaşımında geçiş iletkenliği kuvvetlendiricisi temel blok olarak kullanılmıştır.(Şekil 3.1)



Şekil 3.1 Temel geçiş iletkenliği kuvvetlendiricisi

Bu temel eleman V_1 - V_2 giriş fark geriliminin çıkış akımına çeviren bir fonksiyon görmektedir. Alttaki M_b transistörü akım kaynağı gibi davranır ve I_b akımını sağlar.I_b, üstteki M_1 ve M_2 transistörlerinde büyüklükleri doğrultusunda ikiye bölünür. Bu iki transistörün V_{DS} gerilimleri saturasyonda çalışmalarını sağlayacak büyüklükte seçilmiş olarak varsayılırsa

$$(1)\ \varphi(V_J) = \frac{1}{1+\exp(-Vj)} \tag{3.5}$$

ve

$$(2)\ I_D = I_0 \exp\left(\frac{kV_G}{U_T}\right)\left[\exp\left(-\frac{V_S}{U_T}\right) - \exp\left(-\frac{V_D}{U_T}\right)\right], \text{ den} \tag{3.6}$$

$$I_{OUT} = I_b.\ \tanh\left(kV_{in} / 2U_T\right) \tag{3.7}$$

bulunur.

$$Vin = V_1 - V_2 \tag{3.8}$$

$$I_{OUT} = I_1 - I_2 \tag{3.9}$$

Görüleceği üzere fark gerilimi V_{in} giriş işareti olarak, fark akımı I_{out} çıkış işareti olarak düşünülürse bu devre, orjinle asimetrik olan sigmoidal nonlineerliği modelleyebilen ve çıkışı hesaplayan bir temeli oluşturabilmektedir. Ürünlerin toplamı hesaplaması analog yapılar için oldukça uygundur.

CMOS teknolojisinde karşılıklı iletim sağlamayan sinapslar için en doğal seçim doymada çalıştırılan bir MOS transistördür. Özellikle transistörün izole edilmiş geçit ucuna uygulanan giriş gerilimi, savakda düşük iletkenlik üretir. Bu düzende geniş çıkış yolu sağlar. Şekil 3.2'de hem n kanal hemde p kanal transistörle oluşturulmuş uyarıcı ve bastırıcı sinapslar verilmiştir.

Şekil 3.2 Uyarıcı ve bastırıcı sinaps yapıları

Nöromorfik hesaplamalar için kullanılan analog VLSI sistemlerinde bazı ek işlemlere gerek duyulabilir. Bunlar ;

- Çarpma: Analog sistemlerde iki işaretin çarpılabilmesi için dört bölgeli çarpıcı kullanılır. Bu çarpıcıda her bölge giriş işaretlerinin belirli bir kombinasyondaki çarpımlarına denk gelir.
- Toplama: Çok sayıdaki giriş işaretinin bir araya getirildiği analog işlem.
- Derecelendirme: Derecelendirme faktörü ile işaretlerin belli oranda çarpımı.
- Normalizasyon: Normalizasyonun amacı giriş işaretlerinin dinamik uzaklıklarının azaltılmasını sağlamaktır.

3.4.2 Dijital Teknikler

Analog yaklaşıma göre dijital yaklaşımın iki avantajı bulunmaktadır.

- Tasarım ve yapım kolaylığı: Dijital VLSI teknolojinin kullanımı yüksek güvenirlik, ağırlıkların depolanmasında kolaylık ve düşük maliyet gibi avantajlara sahiptir.
- Esneklik : Dijital yaklaşımın en önemli özelliklerinden biridir çünkü bu çok karışık algoritmaların kullanımını ve bu sayede mümkün uygulama sayısının artmasını sağlamaktadır. Esneklik eksikliği analog sistemler için önemli bir limitasyondur. Bununla birlikte dijital VLSI teknolisinin dezavantajı da vardır. Çarpma işleminin dijital ortamda uygulanması hem yüzey alanını hem de güç tüketimini arttırmaktadır. Bu nedenle de ne zaman analog VLSI dan vazgeçip dijital tekniklerin kullanılması gerektiği sorunu henüz tam cevaplanabilmiş değildir. Ancak en genel amaçlı kullanımlarda dijital VLSI teknolojisi kullanılabilir (Haykin, 1994).

3.4.3 Karma Teknik

Nöral VLSI için çip alanı küçük ve hızı yüksek olduğundan analog tasarım daha çekici gelmektedir. Dijital tekniklerin ise uzak mesafeler arası iletişim için ayrı bir önemi vardır. Çünkü dijital işaretler kolay bozulmaz, gürültü içermez, kolayca gönderilir ve alınabilirler. Her iki özelliği sağlamak amacıyla yapay sinir ağları VLSI uygulamalarında Karma (hibrid) tekniği oluşturulmuştur. Bu teknikte analog ve dijital devreler bir çip üzerinde yapılandırılmıştır. (Murray, 1991)

3.5 Silikon Retina

Caltech'de Carver Mead'in grubu tarafından keşfedilen ilk çipler büyük oranda biyolojik sinir ağlarından esinlenerek tasarlanmıştır. Mead biyolojik nöronlara mümkün olduğunca çok benzeyen analog nöral çipler yapmaya çalışmıştır; bunlara son derece düşük güç tüketimi,tamamen analog donanım ve devamlı çalışma dahildir. Yapı şu şekilde tarif edilebilir. Işık, üçlü sinapslardan bipolar hücrelere giden primer bir yol içeren foto-reseptörler tarafından elektrik sinyallerine dönüştürülür. Bipolar hücreler retinanın üretim hücreleri olan retinal ganglion hücreleri ile bağlantılıdır.Aynı zamanda üçlü sinapslar yoluyla foto-reseptörlere bağlı olan yatay hücreler direkt olarak fotoreseptörlerin altına yerleşiktir ve bipolar hücrelere götüren aksonlarla bağlantılı sinapsları vardır. (Mead,1989) Sistem üçlü sinapsların 3 elementinin terimleri ile tarif edilebilir:

- Fotoreseptör ışık yoğunluğunun logaritmasını üretir
- Yatay (horizontal) hücreler fotoreseptörün zamanda ortalamasını alan bir ağ oluştururlar
- Bipolar hücrenin çıktısı, fotoreseptör çıktısı ve yapay hücre çıktısı arasındaki farkla orantılıdır.

Görsel uygulamaların çoğunda ilk amaç lineerliği sağlamaktır. Elektronik görüntü algılama aygıtları (imager) ile nöromorfik algılayıcıların fotosensörlerin gereksinimleri arasında önemli farklılıklar vardır. Bir görüntü algılama aygıtının ilk işlemi; gelen ışığı, daha sonra geri dönüştürülebilip orjinal resme dönebilecek şekilde hatasız olarak elektriksel işarete çevirmektir. Diğer yandan nöromorfik devreler daha önceden resimlenmiş çevreler hakkında bilgi edinerek araştırma yapar. Dolayısıyla bir piksel üzerine düşen ışık yoğunluğu ikincil önem taşır.

Fotosensör devreler için ise esas önemli olan, çevredeki ışığın şiddeti değişse bile, nesnenin yeri ve şekli değişmediği sürece çıkışında sabit bir işaret gösterebilmesidir. Buda devreleri kontrasta duyarlı yapabilmekle başarılabilir. Burada bahsedilen kontrast, nesne yoğunluğu ile

arka fon yoğunluğunun oranıdır.

Bir fototransistör ışık yoğunluğunu lineer olarak akıma çevirir, daha sonra bu akım MOSFET'lere bağlı iki diyot tarafından gerilime dönüştürülür. Akım değerlerinin düşük seviyelerde olmasından dolayı transistörler eşik-altı bölgede çalışırlar. Bu bölgede çıkış gerilimi akımla logaritmik olarak değişir. Logaritmik kodlama aydınlatmanın sabitlenebilmesinin sağlanmasında ilk aşamadır. Çünkü bu sayede görüntüdeki ışık yoğunluğunun ufak değişimleri bile çıkış gerilimine katkı bias değerleri oluşturabilir.

Çıkış gerilimi, bir iletkenlik kuvvetlendirici ve kazanç geri beslemesi ile eşleştirilmiş buda birbirine çok yakın piksellere bağlı olan bir direnç şebekesine bağlanmıştır. Aydınlatmanın sabit tutulabilmesi çıkış gerilimi ile bir başka iletkenlik kuvvetlendiricisindeki geriliminin farkının alınması ile sağlanmıştır. Bu fark bütün aydınlatmadaki değişikliklerin katkı bias değerleri oluşturmasını önlemektedir. Bair, bu yaklaşıma, farklı katsayıları sabitlenmiş ve alçak geçiren filtreden geçirilmiş iki görüntünün farkını alarak erişmiştir.

MOSFET'lere bağlı diyotlardaki akımın logaritmik kodlanmasında oluşan bir problemde transistör uyumsuzluklarıdır. Bu problemi çözebilmek için Delbruck adaptif fotosensor aşaması önermiştir. Burada yüksek kazanca sahip işaretler hızlı taranırken, düşük kazanca sahip işaretler yavaş taranmaktadır. Bu devrede ayrıca fotosensörün yanıt süresini geliştirebilmek amacıyla aktif geribesleme de kullanılmıştır.

Yukarıda bahsedilen devreler gerilim modlu devreler olarak tanımlanabilir. Yakın zamanlarda ise araştırmacılar akım modlu devrelerin daha az yer kaplayan transistör devreleri olduğunu göstermişlerdir.

Eşikaltında çalışan MOSFET'lerin nonlineer eksponansiyel akım-gerilim ilişkilerine karşın, görüntü yumuşatma gibi lineer işlemler akım domeninde gerçekleştirilebilmektedir. Bununla birlikte lineerlik prensipleri ile karmaşık hesaplamalarda yapılabilmektedir.

Akım modlu tasarım tekniğine dayanan ilk silikon retinalardan biri Boahen ve Andreou tarafından yapılmıştır. Ancak bu devrede yüksek kazanç geribeslemesinden kaynaklanan geçici kararsızlıklar görülmüştür. Boahen bu problemlerin üstesinden gelen dış retinayı (koniler ve yatay hücreler) modelleyen bir devre öne sürmüştür. Venier de her pikselde oluşan foto akım baz alınarak yapılmış dirençsel ağın iletkenliğini ayarlayabilen kontrasta duyarlı silikon retina tabakası tanımlamıştır.

Kontrast, nesnedeki ışık yoğunluğu ile (foreground) I_F, arka fondaki ışık yoğunluğu I_B

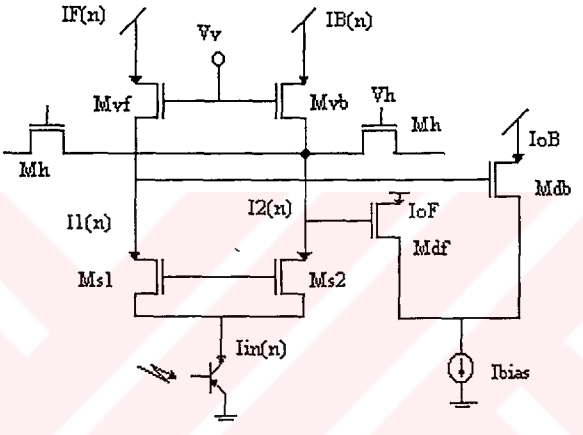
arasındaki farkı ölçer. Michelson kontrastı normalize olarak tanımlanır.

$$C_M = \frac{I_F - I_B}{I_F + I_B} \tag{3.10}$$

Işık yoğunluğunun sinüs dalgasının açılımı aşağıdaki gibidir:

$$I(x) = I_{mean}[1+C_M \sin(wx + \Phi)] \tag{3.11}$$

Burada Imean; ortalama parlaklığı ifade etmektedir.



Şekil 3.3 Kontrasta duyarlı silikon retina'nın bir pikseli

Şekil 3.3'de silikon retinanın bir pikselini oluşturan devre verilmiştir. M_{S1} ve M_{S2} giriş fotoakımı $I_{in}(n)$ akımını iki eşit akıma, $I_{1n}(n)$ ve $I_{2n}(n)$ akımlarına böler. M_{vf} transistörü ile $I_{1n}(n)$, $I_F(n)$ e taşınır. $I_F(n)$, n. pikseldeki ön fon ışık yoğunluğudur.

$I_2(n)$ akımı, M_{VB} ve M_H transistörleri tarafından oluşturulan ağ üzerinden geçer. Burada M_H, komşu piksellerdeki benzer noktalara bağlıdır. $I_B(n)$, n. pikseldeki arka fon ışık yoğunluğu $I_2(n)$'in alçak geçirgen filtreden geçirilmiş hali gibidir. Komşuluk sayısının büyüklüğü V_h ve V_v gerilimleri arasındaki fark ile kontrol edilir. Fark ne kadar büyükse, resmin düzgünlüğü de o kadar iyi olur.

Burada tüm transistörler eşikaltı bölgede çalışmaktadırlar.

Sonuçta:

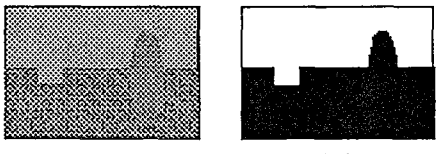
$$I_{OF} = \frac{I_{BIAS}}{2}\left(1 + C_M(n)\right) \tag{3.12}$$

$$I_{OB} = \frac{I_{BIAS}}{2}\left(1 - C_M(n)\right) \text{ bulunur.} \tag{3.13}$$

Her pikseldeki arka fonun, bu pikselin komşuluklarındaki görüntü yoğunluklarının ağırlıklı ortalaması olarak tanımlanmasından dolayı, I_{OF} çıkışı merkezdeki alıcı alana yakındır ve bu çıkış arttıkça merkezdeki yoğunluk, yüzeydeki yoğunluğun üstüne çıkmaktadır. Benzer şekilde I_{OB} çıkışıda alıcı alana yakındır. Öndeki fon yüzeyine pürüzsüzlük eklemek M_{VF} transistörünün source düğümüne difüzyon görevi gören transistörler eklemekle sağlanabilir. Ancak bu durum da tek pikselin yüzey alanını genişletir (Bertram, E., Choi, T., 2001)

3.5.1 Görüntü Sınıflandırmaya Dayalı Retina

Bu retinanın çalışma prensibi referans görüntü ile devre üzerine yansıtılan retina arasındaki karşılaştırmaya dayanır. Referans gösterlen binary görüntü devredeki latchlerde her siyah değeri için lojik '0', her beyaz değer için lojik '1' olacak şekilde depolanır. Şekil 3.4.a'da algılanacak resim ve Şekil 3.4.b'de de referans resim gösterilmektedir.



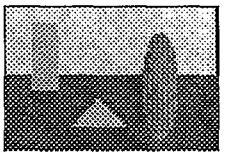
Şekil 3.4 a)Algılanacak resim b) Referans resim

Retina üzerine bir görüntü yansıtıldığı zaman, referans görüntünün siyah ve beyaz kısımlarına ait pikseller tarafından üretilen akımlar ayrı ayrı toplanırlar ve toplam siyah ve toplam beyaz alan için akımları oluştururlar. I_{siyah} ve I_{beyaz} olarak nitelendirilen akımlar şu şekilde hesaplanır:

$$I_{beyaz} = K\sum_{j=0}^{N-1}\sum_{j=0}^{N-1} R(i,j).X(i,j) \tag{3.14}$$

$$I_{siyah} = K\sum_{j=0}^{N-1}\sum_{j=0}^{N-1}\overline{R(i,j)}.X(i,j) \tag{3.15}$$

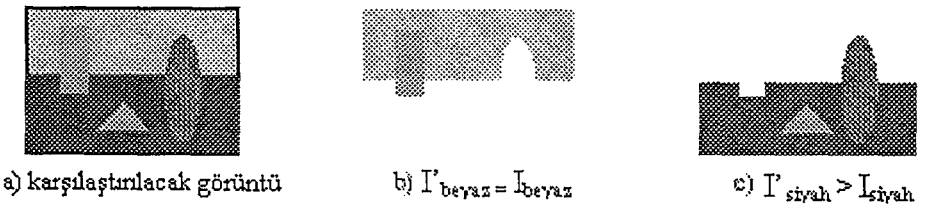
Burada X(i,j) parlaklık değişiminin (i,j) koordinat düzlemine göre değişimini göstermektedir. K ise parlaklık değişimi ile piksel tarafından üretilen akım arasındaki lineer ilişkiyi tanımlar. Şekil 3.5 ile referans görüntü arasındaki benzerlik derecesi incelenmiştir. Görüldüğü gibi Şekil 3.5'de beyaz bir üçgen bulunmaktadır. Yukarıda anlatıldığı gibi beyaz kısımdaki akımlar hesaplanırsa referans resimin beyaz akımı ile Şekil 3.5'in beyaz akımının aynı olacağı ancak siyah akımlarının farklılık göstereceği bulunur.



Şekil 3.5 Karşılaştırılacak görüntü (Voon, L.Y., vd.)

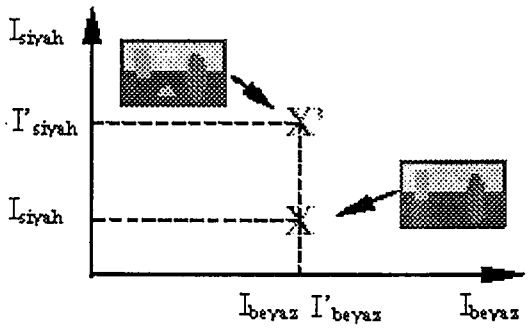
Bu durum Şekil 3.6 ve Şekil 3.7'da daha iyi görülmektedir.

Şekil 3.6 Referans görüntü için hesaplanan akım değerleri (Voon, L.Y., vd.)



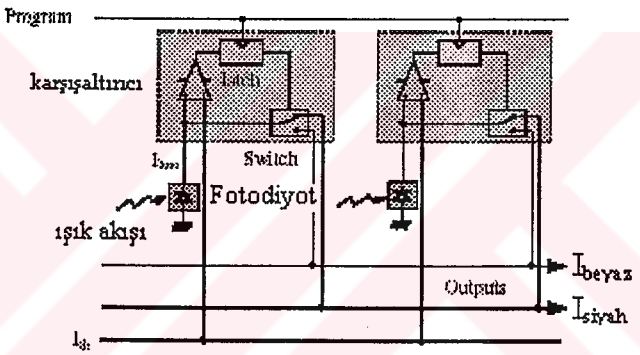
Şekil 3.7 Karşılaştırılacak görüntü için hesaplanan akım değerleri (Voon, L.Y., vd.)

I_{siyah} ve I_{beyaz} değerleri hesaplanarak bu iki resmin birbirinden farklı oldukları anlaşılır. Bu durumda Şekil 3.8'de gösterilmiştir.



Şekil 3.8 Referans resim ve karşılaştırılan resim arasındaki fark (Voon, L.Y., vd.)

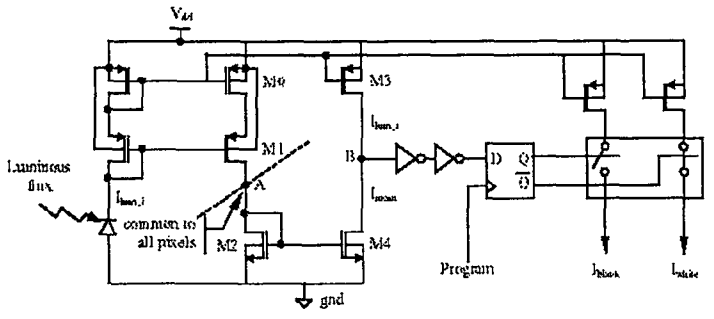
Yukarıda anlatılan ayırımları yapabilecek programlanabilir piksel yapısı Şekil 3.9'da gösterilmiştir.



Şekil 3.9 Görüntü sınıflandırmaya dayalı retinanın piksel yapısı (Voon, L.Y., vd.)

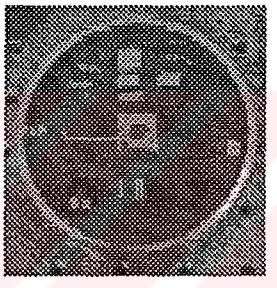
Her piksel, bir fotodiyot, saklayıcı (hafiza birimi) ve anahtarlama devre yapısına sahiptir. Programlama esnasında, belirlenen görüntü devre üzerine yansıtılır ve parlaklık değişimine karşılık her fotodiyotun ürettiği akım I_{th} eşik akımıyla akım karşılaştırıcı devresi kullanarak karşılaştırılır. Karşılaştırmanın sonucu ikili bir değerdir ve bu değer "program" girişine darbe uygulanarak kapılar üzerinde saklanır. Bu değer anahtarların bütün fotodiyotları birbirlerine bağlaması veya açık kalması yönünde tetikler. Anahtarların durumu yeni bir hafizalama durumu sözkonusu oluncaya kadar aynı kalır.

Devre MOS transistör bazına indirgenirse Şekil 3.10'daki gibi gösterilebilir (Voon, L.Y.,vd.)



Şekil 3.10 Transistör bazında piksel yapısı (Voon, L.Y.,vd.)

Çalışma koşulları içindeki retina Şekil 3.11'de gösterilmiştir.



Şekil 3.11 Retina çipi (Voon, L.Y.,vd.)

4. TEMEL YAPAY SİNİR AĞI YAPILARI

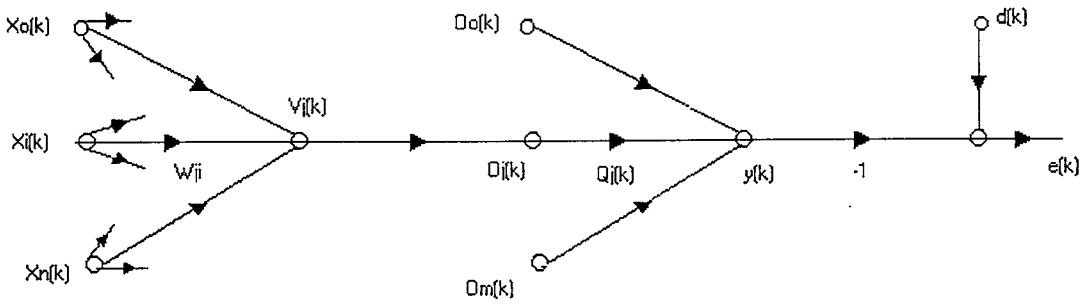
Görüntü algılaması ve görüntünün ayrıştırılarak sınıflandırılabilmesi, son yıllarda yapay sinir ağı teknolojisinde de kullanılmaktadır. Önceden verilen görüntü verisine dayanarak başka görüntülerin de kestirimi veya görüntü üzerindeki birkaç piksellik bölgenin kaybı durumunda görüntünün doğru şekilde algılanabilmesi için kullanılan sistem ezberlemeden öte eğiticili bir sistem olmalıdır. Bu çalışmada kullanılan temel yapay sinir ağı yapıları, çok katmanlı algılayıcılar, radyal temelli fonksiyonlu sinir ağları, olasılıksal sinir ağları ve genelleştirilmiş regresyonlu sinirsel ağlar ve özellikleri bu bölümde incelenmektedir.

4.1 Çok Katmanlı Algılayıcı (MLP)

Bir çok katmanlı algılayıcı yapısı birçok birimin oluşturduğu bir kümedir. Bu algılayıcı birimler bir araya gelerek ağdaki katmanları oluşturur. Bu katmanlar da bir araya gelerek ağı oluşturur. MLP'de üç temel katman vardır. Bunlar giriş katmanı, gizli katman ve çıkış katmanıdır. Giriş ve çıkış katmanı dışındaki tüm katmanlar gizli katman olarak adlandırılır. Genelde ağın eğitiminde eğiticili yöntem kullanılmaktadır. En yaygın yöntem hatanın geriye yayılımı algoritmasıdır.

Katmanlı YSA'lara uygulanabilmesi nedeniyle önem kazanan geriye yayılım öğrenme algoritması eğim düşme algoritmasının katmanlı YSA'lara uyarlanmış halidir. Algoritmada geçen sinyaller ayrık zamanda gösterilmiş ve ayrık zaman değişkeni k olmak üzere x(k), ağ giriş vektörü; o(k), orta katman çıkış vektörü; d(k), arzu edilen çıkış vektörü; y(k), gerçek ağ çıkış vektörü ve W ve θ ağırlıklar matrisini göstermektedir. Eğim düşme esasına dayanan geriye yayılım algoritmasında seçilen amaç ölçütünün (performans kriteri) bir katmandaki ağırlığa göre doğru eğiminin hesaplanması gerekir. Bu nedenle, ağ çıkışındaki hata sinyali, katmanlardan geriye doğru yayılır. Aşağıda açıklanacak olan algoritma, çıkış katmanı doğrusal olan üç katmanlı tek çıkışlı ileri beslemeli YSA'ya göre çıkarılmıştır. Çok çıkışlı YSA, tek çıkışlı YSA'nın benzeri olduğundan konunun açıklanabilmesi bakımından bir eksiklik oluşturmayacaktır.

Çıkış katmanı doğrusal olan ileri beslemeli üç katmanlı YSA'nın sinyal akış şeması Şekil 4.1'de verilmiştir.



Şekil 4.1 İleri Beslemeli Üç Katmanlı YSA Sinyal Akış Şeması

Şekil 4.1'deki sinyal akış şemasından YSA'nın ileri yöndeki matematiksel modeli aşağıdaki gibi yazılır.

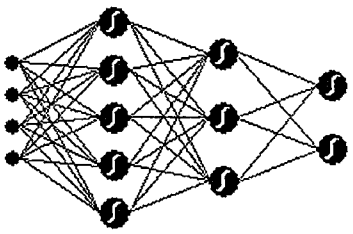
$$v_j = \sum_{i=0}^{n} w_{ji} x_i \ , \quad o_j = \varphi(v_j) \quad \text{j=1,2,.....,m} \tag{4.1}$$

$$y_l = \sum_{j=0}^{m} \theta_j . o_j \tag{4.2}$$

Tek çıkışlı bir YSA için ağ çıkış hatası, hataların kareleri (örneksel amaç ölçütü) ve toplam amaç ölçütü (hataların karelerinin ortalaması) aşağıdaki gibi tanımlanır.

$$e(k) = d(k) - y(k) \ , \ E(k) = (1/2)e^2(k) \tag{4.3}$$

Şekil 4.2'de çok katmanlı algılayıcı yapısına ait örnek bir ağ yapısı görülmektedir.



Şekil 4.2 Çok katmanlı algılayıcı yapısına ait örnek ağ yapısı

Toplam amaç ölçütü, N adet eğitim örneği için hataların karelerinin ortalaması olduğundan örneksel ya da toplu amaç ölçütünün ağırlıklara göre eğimi bulunarak geriye yayılım algoritması gerçeklenebilir. Hataların karelerinin eğimine göre, her bir eğitim örneğinin uygulanışında ağırlıklar yenilenirse örneksel öğrenme kuralı elde edilir. Toplam amaç

ölçütünün eğimine göre, N adet eğitim örneğinin uygulanışından sonra ağırlıklar yenilenirse toplu öğrenme kuralı elde edilir. Buna göre, hataların karelerinin, doğrusal çıkış katmanındaki bir ağırlığa göre eğimi, zincir kuralına göre kısmi türevlerle belirlenebilir. Geriye yayılım algoritmasında öğrenmenin yavaş olmasının iki temel nedeni vardır.

- Ağırlık uzayı boyunca hata yüzeyi oldukça düzgün olduğunda, hata yüzeyinin bir ağırlığa göre türevi çok küçüktür. Dolayısıyla ağırlığa uygulanacak düzeltme çok küçük olacağından ağın öz yeteneğinin iyileşmesi uzun zaman alacaktır. Diğer taraftan, hata yüzeyi çok girintili çıkıntılı olabilir ve hata yüzeyinin ağırlığa göre türevi büyük olacağından yüzeyin en azından uzaklaşılabilir.
- Hatanın ağırlıklara göre negatif eğim vektörü, hata yüzeyinin en azından uzaklaşan bir yönü verebilir ve ağırlıklar hatalı yönde düzeltilir. (Özyılmaz, 2000)

Yukarıda da bahsedildiği gibi öğrenme oranı küçük seçilirse öğrenme yavaşlayacak, büyük seçilirse ağırlık değişimleri salınımlı ve kararsız olacaktır. Bu sakıncaların etkisi, geriye yayılım algoritmasında ağırlıklara uygulanacak düzeltme miktarını belirleyen denklemlere momentum terimi (β) eklenerek azaltılabilir. Momentum katsayısı $0<\beta<1$ arasında seçilir. Amaç ölçütünün bir ağırlığa göre eğimi, ardışıl iki iterasyonda aynı işaretli ise ağırlıklara uygulanacak düzeltme artar, aksi halde azalır. Momentum katsayısı ile YSA'nın öğrenme oranında belirli bir hızlanma elde edilebilir.

4.1.1 Geliştirilmiş Geriye Yayılım Öğrenme Algoritmaları

Geriye yayılım algoritmasının iki önemli sakıncası, yavaş yakınsama ve yöresel en aza yakalanma olarak özetlenebilir. Yöresel en az, geriye yayılım algoritmasının doğal bir özelliğidir ancak, çeşitli yöntemlerle öğrenme oranı ya da momentum katsayısı uyarlanarak YSA'nın öğrenme hızı artırılabilir. Öğrenme oranı uyarlamasında genellikle 3 kriter uygulanır.

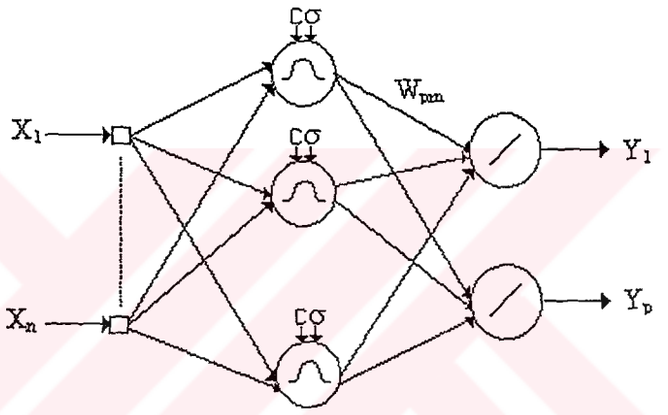
- Bütün ağ için seçilen tek bir öğrenme oranı, ağdaki bazı ağırlıklar için en iyi hata yüzeyini verirken diğer bazı ağırlıklar için uygun olmayabilir. Bu nedenle YSA'daki her bir ağırlık için öğrenme oranı seçilmelidir.
- Her bir öğrenme oranı,her eğitim iterasyonunda uyarlanmalıdır.
- Ardışıl eğitim iterasyonlarında amaç ölçütünün bir ağırlığa göre eğimi aynı işaretli ise o ağırlığın öğrenme oranı arttırılmalı, aksi halde azaltılmalıdır.

Bu temel kriterlerden yararlanarak çeşitli öğrenme oranı ve momentum katsayısı uyarlama yöntemleri geliştirilmiştir.

4.2 Radyal Temelli Fonksiyon Ağları (RBF)

Katmanlı yapay sinir ağlarının 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'in eğitimi, çok boyutlu uzayda eğitim verilerine en uygun bir yüzeyi bulma problemine dönüşür. RBF'in 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.

Şekil 4.3'de genel bir RBF yapısı verilmiştir. Burada c değerleri merkezleri, σ değerleri ise yayılma parametresini belirtmektedir.



Şekil 4.3 RBF ağ yapısı

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 uyarlamalı ve doğrusal bir dönüşüm gerçekleştirilir.

RBF'de uyarlanabilecek serbest parametreler; merkez vektörleri, radyal fonksiyonların genişliği ve çıkış katman ağırlıklarıdır. Çıkış katmanı doğrusal olduğundan ağırlıklar, eğim düşme yada doğrusal en iyileme yöntemleri ile kolayca bulunabilir. Merkezler, girişler arasından rastgele ve sabit olarak seçilebilmekle birlikte RBF'in performansını iyileştirmek amacıyla merkez vektörlerinin ve genişliği uyarlanması için çeşitli yöntemler geliştirilmiştir. Merkez vektörleri, eğim düşme yöntemine göre eğiticili öğrenme algoritması ile uyarlanarak, dik en küçük kareler yöntemi ile, yada kendiliğinden düzenlemeli yöntemle giriş örneklerinden öbekleme yapılarak belirlenebilir.

4.2.1 RBF Ağlarının Eğitilmesi

Bir RBF ağının eğitilmesi, RBF birim merkezlerinin, gizli katmandan çıkış katmanına olan ağırlıkların ve σ uzaklık ölçekleme parametresinin belirlenmesi ile yapılır.

4.2.1.1 RBF Birim Merkezlerinin Belirlenmesi

RBF ağının performansı kritik olarak seçilen merkezlere bağlı bulunmaktadır. Alıcı alanların merkez koordinatlarının belirlenmesi için çeşitli metodlar kullanılmaktadır. Örneğin eğitme kümesindeki her bir giriş vektöründe bir merkez yerleştirilebilir. Fakat bu durumda gerekenden çok daha fazla küme ve gizli katman nöronu oluşabileceğinden bu yöntem pek uygun değildir. Pratikte merkezler verilerin bir alt kümesi olarak seçilir. Bu seçim yapılırken gizli düğümlerin sayısı bütün giriş uzayını kaplayacak yeterlikte olmalıdır. Küme merkezlerini bulmak için en iyi yaklaşımlardan biri K-ortalamalı kümeleme algoritmasıdır. Bu algoritmaya göre giriş bilgilerinin yoğun olduğu yerlerde merkezler yoğun bir şekilde dağıtılır. (Haykin,1994;Middleton, 1997)

4.2.1.2 Uzaklık Ölçekleme Parametresinin Belirlenmesi

Uzaklık ölçekleme parametresi (genişlik parametresi) σ, RBF ağlarında alıcı bölgelerin çapını belirleyen bir büyüklüktür. Bu parametre merkezler birbirine yakınsa küçük, merkezler birbirinden uzaksa büyük seçilmelidir. Genelde ise σ değeri olarak kümeleme merkezleri ve eğitme kümesindeki örnekler arasındaki ortalama mesafe alınır.

Eğitim sırasındaki ağırlıkların belirlenmesi adımında ise Lineer En Küçük Kareler (Linear Least Squares) yöntemi kullanılarak hata istenen bir değere azalacak şekilde saklı katmandan çıkışa olan ağırlıklar belirlenir.

4.2.2 RBF Öğrenme Algoritmaları

RBF ağlarında öğrenmeye ilişkin bir çok yaklaşım mevcuttur. Bunlardan bir çoğu öğrenme işini iki kısma ayırır. Buna göre ilk öğrenme işlemi gizli katmanda gerçekleşir. Daha sonra öğrenme çıkış katmanında devam eder. Gizli katmandaki öğrenme eğiticisiz öğrenme algoritmalarından biri kullanılarak yapılır. Çıkış katmanındaki öğrenme ise eğiticili öğrenmedir.

RBF ağı için geliştirilen çeşitli öğrenme algoritmaları aşağıda kısaca özetlenmiştir (Chen vd., 1993, Özyılmaz, L.,2000):

4.2.2.1 Sabit merkezlerde En Küçük Kareler Yöntemi

RBF merkezleri giriş bilgisinden rasgele seçilir. Eğitim seti problemi iyi temsil edecek şekilde seçilirse bu yöntem iyi sonuç vermektedir. Merkezler belirlendikten sonra en küçük kareler yöntemi ile ağırlıklar eğiticili modda belirlenir.

4.2.2.2 Ortogonal En Küçük Kareler Yöntemi

Bu algoritmada uygun RBF merkezlerinin ve ağırlıkların belirlenmesi eş zamanlı olarak yapılır. Bu prosedür radyal temelli fonksiyon merkezlerini uygun bir ağ ortaya çıkana kadar rasyonel bir biçimde tek tek seçmektir.

4.2.2.3 İteratif Kümeleme ve En Küçük Kareler Yöntemi

Bu algoritmada RBF merkezleri bir iteratif kümeleme algoritması kullanılarak ayarlanır ve ağırlıklar iteratif en küçük kareler yöntemi ile güncelleştirilir. Burada merkezlerin belirlenmesi eğiticisiz olarak yapılır.

4.2.2.4 Dinamik Komplekslik Öğrenme Algoritması

Bu iteratif öğrenme yönteminde, her yeni bir temel fonksiyonla önceki arasında oluşturulan bir açı değerine ve kestirim hatasına bağlı olarak ağa yeni bir temel fonksiyon eklenir.

4.3 Olasılıksal Sinir Ağları (PNN)

Olasılıksal sinir ağları 1990'da Donald Specht tarafından geliştirilmiştir. Olasılıksal sinir ağları bir kalıp katman dahilinde dağıtım işlevini geliştirmek için danışmanlı bir eğitim dizisi kullanılır. Hatırlama kipinde bu işlevler öğrenilmiş bir kategori veya sınıfın parçası olan bir girdi özelliği vektörünün benzerliğini tahmin etmek için kullanılır. Öğrenilen kalıplar, verilen bir girdi vektörü için en uygun sınıfı belirleyecek her bir kategorinin, aynı zamanda nispi sıklık olarak da adlandırılan, bir ön olasılığı vasıtasıyla birleşebilir veya ölçülebilir. Eğer kategorilerin nispi sıklığı bilinmiyorsa bu durumda tüm kategoriler eşit derecede uygun farz edilir ve kategorinin belirlenmesi yalnızca, girdi özellik vektörünün bir sınıfın dağıtım işlevine yakınlığı temel alınarak gerçekleştirilir.Hatanın geriye yayılımı algoritmasıyla eğitilen ileri beslemeli bir ağdan en temel farkı, PNN'in eğitim setini sadece bir eğitme adımında kullanmasıdır. (Specht, 1990)

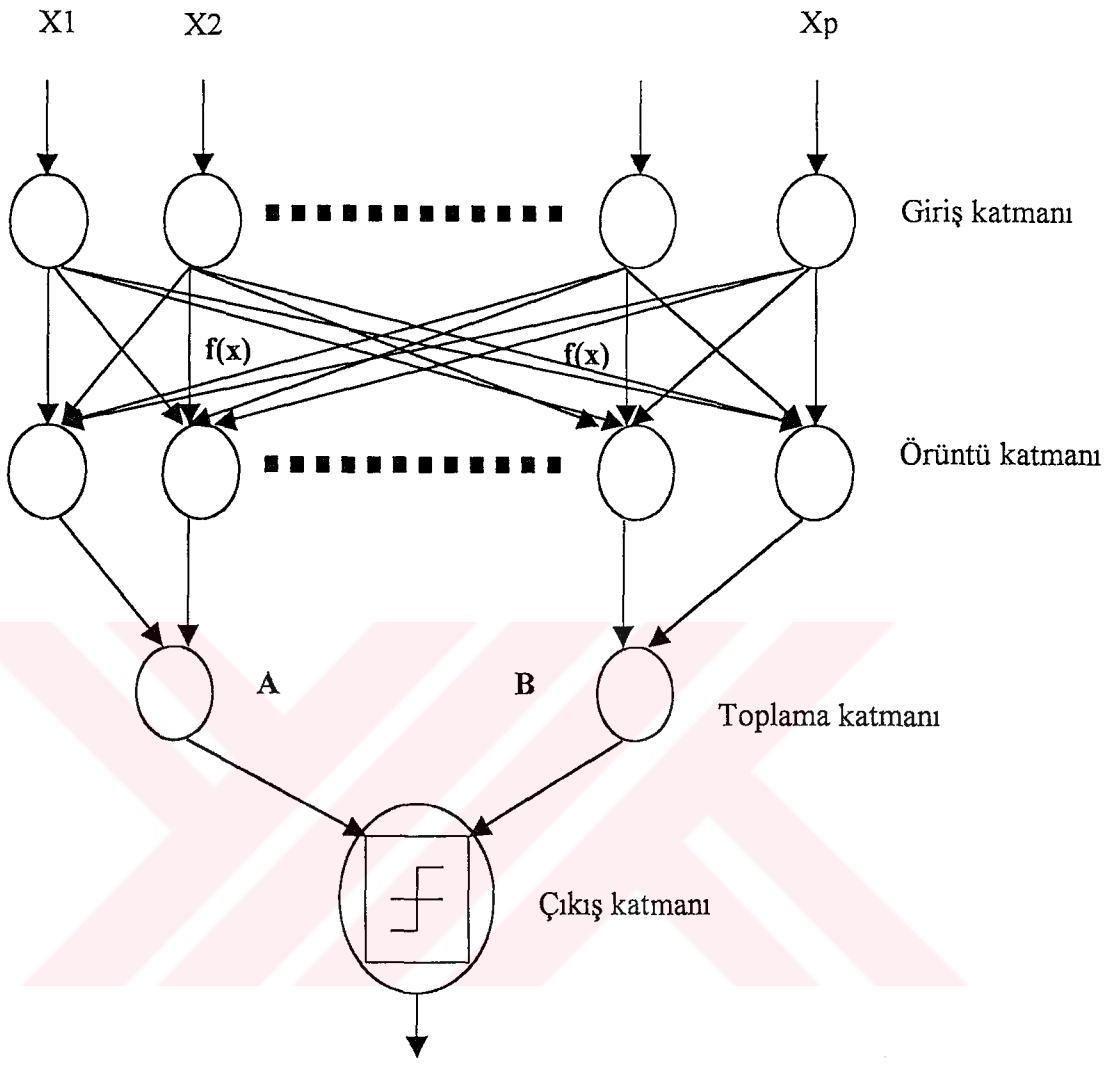
Olasılıksal sinir ağları sınıflandırılacak nesnelere tanımlanması için ihtiyaç duyulan ayrılabilir değişkenlerin varlığı kadar çok elemana sahip bir girdi katmanı içerir. Eğitim dizisini organize eden bir kalıp katmana sahiptir, örneğin her bir girdi vektörü bireysel bir işleme elemanı tarafından temsil edilir ve son olarak, ağ tanınacak sınıflar kadar çok işleme elemanına sahip olan ve toplam katmanı olarak adlandırılan bir çıktı katmanı içerir. Bu katmandaki her bir eleman aynı sınıfla ilişkili kalıp katman dahilinde işlem elemanları yoluyla birleşir ve bu kategoriyi çıktıya hazırlar. Eğer girdiler ağa girmeden daha önce norm haline dönüştürülmezse, girdi vektörünü normlaştırmak için bazen dördüncü bir katman eklenir. Kalıp katmanda doğru şekilde nesne ayrışımını sağlamak için girdi vektör norm haline dönüştürülmelidir.

Kalıp katmanında, çalıştırma kümesindeki her bir girdi vektörü için bir işleme elemanı bu0lunmaktadır. Normalde her bir çıktı sınıfı için eşit miktarlarda işleme elemanları bulunmaktadır. Aksi taktirde, bir ya da daha fazla sınıf hatalı şekilde kaydırılabilir ve ağ hatalı sonuç verir. Kalıp katmanındaki her bir işleme elemanı bir defa çalıştırılır. Bir girdi vektörü çalışan vektöre denk geldiğinde yüksek bir çıktı değeri vermesi için bir unsur çalıştırılır. Çalıştırma işlevi, sınıflandırma sonuçlarını daha iyi çıkaran global bir düzeltme faktörü içerir. Bir özel vektör kategorisi, girdinin istenen çıktısı tarafından açıkça belirtildiği için çalıştırma vektörleri mutlaka çalıştırma kümesindeki her hangi bir özel düzende olmak zorunda değillerdir. Öğretici işlev, doğru olan çıktı sınıfındaki ilk çalıştırılmayan işleme elemanını basit bir şekilde seçer ve onun bağıl değerlerini çalıştırma vektörüne uydurmak için değiştirir.

Kalıp katmanı, bir çıktı veren bir girdi vektörüne sadece en yüksek uyumu gösteren konumda yaklaşan bir şekilde çalışır. Bu tarzda her hangi bir verilen girdi vektörü için sadece bir sınıflandırma kategorisi oluşturulur. Eğer girdi, kalıp katmanına programlanan kalıplara iyi şekilde bağlanmazsa çıktı oluşmaz.

Parzen tahmini, objelerin sınıflandırılmasına ince ayar yapması için kalıp katmanına eklenebilir. Bu, bir işleme elemanının içine kurulan her bir çalıştırma kalıbına oluşma sıklığı ilave edilerek yapılır. Esas itibarı ile bir sınıfta bulunan her bir örnek için oluşan olasılık dağılımı, kendi çalıştırma düğümlerine yayılır. Bu şekilde bir objenin daha kesin bir beklentisi objeyi bir sınıf üyesi olarak tanıtan niteliklerine ilave edilir.

PNN bir Bayes-Parzen sınıflandırıcıdır. Olasılıksal sinir ağılarında bir nörondaki aktivasyon fonksiyonu, eğitim örneklerine dayanan olasılıksal yoğunluk fonksiyonundan (PDF) hesaplanarak türetilmiştir.



Şekil 4.4 Olasılıksal sinir ağının yapısı

PNN'in topolojisini oluşturan katmanlar şöyle şekillendirilirler:

Giriş katmanı m boyutlu giriş vektörü uygulanmak üzere m nörona sahiptir. İlk katman her giriş örneği için bir model oluşturur. İkinci saklı katman her sınıf için bir toplama işlemi gerçekleştirir. Her toplama işlemi bir önceki katmandaki çıkışlardan bir toplam belirlemek amacıyla kullanılır. Çıkış katmanı, toplamın çıkışlarından en yüksek olasılığı seçerek karar kuralını uygulamak için oluşturulan katmandır. Bu karar katmanı kazanan alır ağları ile kurulabilir.

Olasılıksal sinir ağının çalıştırılması, geri yayılım ile olduğundan daha da basittir. Bununla birlikte, eğer kategoriler arasındaki ayrılık değişken ise ve aynı zamanda oldukça farklı ise kalıp katmanı oldukça büyük olabilir.

4.3.1 Parzen Penceresi Sınıflandırılması

Olasılıksal sinir ağlarının gelişimi Parzen penceresi sınıflandırılmasına dayanır.Parzen penceresi methodu, olasılık yoğunluğu fonksiyonunun(pdf) hesabını pencere sayısının süper pozisyonunu, fonksiyon ile değiştirerek sentezleyen parametrik olmayan bir prosedürdür.

Parzen penceresi sınıflandırıcısı verilen eğitme örneğini kullanarak olasılık yoğunluğu fonksiyonunu hesapladıktan sonra bir sınıflandırma kararı verir. Çok kategorili sınıflandırıcısı kararı aşağıdaki gibi belirtilir.

$p_k f_k > p_j f_j$, hepsi için j=/=k (4.4)

Burada p_k, k sınıfindaki örneklerin öncelikli olasığının oluşumudur, f_k ise k sınıfının tahmini olasılık yoğunluk fonksiyonudur.

Olasılık yoğunluk fonksiyonunun hesaplanması aşağıdaki algoritmayla yerine getirilir:

- d boyutundaki Gaussianların değerlerini toplayıp, her eğitme örneğinin değerlendirip ve tahmini olasılık yoğunluğunu oluşturmak için toplam ölçeklendirilir.
- $f_k(x) = (1/(2p)^{d/2} s^d) (1/N) S_i = 1^{NK} \exp [-(\mathbf{x}-\mathbf{x}_{ki})^T (\mathbf{x}-\mathbf{x}_{ki})/ (2s^2)]$ (4.5)
- Burada $\mathbf{x}_{ki}$, k sınıfından d boyutsal bir i-th örneğidir.
- $f_k(\mathbf{x})$ fonksiyonu her eğitme örneği için Gaussian olasılık dağılımının küçük multiolasılıksal değişkenlerin toplamıdır.Sağlanan örneklerin ötesinde olasılık dağılımını kullanmak genelleştirmeyi sağlar.$f_k(x)$ fonksiyonundaki k indeksi sınıfların dağılımları arasındaki yayılma farkını belirtir.
- Eğitme örnekleri ve bu örneklerin Gaussian dağılımlarının sayısı arttıkça tahmin edilen olasılık yoğunluk fonksiyonu gerçek olasılık yoğunluk fonksiyonuna yaklaşır.
- Sınıflandırma sonucu aşağıdaki eşitsizlikten bulunur:
- $S_i = 1^{Nk} \exp[-(\mathbf{x}-\mathbf{x}_{ki})^T (\mathbf{x}-\mathbf{x}_{ki}) / (2s^2)] > S_i = 1^{Nj} \exp[-(\mathbf{x}-\mathbf{x}_{ji})^T (\mathbf{x}-\mathbf{x}_{ji}) / (2s^2)]$, hepsi için j=/=k (4.6)
- Bu eşitsizlik her sınıftaki örneklere bağımlı frekansların öncelikli olasılıkları kullanarak türetilmiştir:
- $p_k = N_k/N$ (4.7)
- Burada N, eğitilmiş örneklerin sayısı; N_k, k sınıfindaki örneklerin sayısı'dır.

Parzen pencereleri sınıflandırıcıları sınıflandırmayı yapmak için örneklerin giriş eğitim setlerini kullanır, bu bilgisayarın hafizasında eğitim setinin depolanmasına ihtiyaç var anlamına gelir. Hesaplanmanın hızı eğitim setinin boyutu ile orantılıdır.

Düzeltme faktörü eğitim için seçilmesi gereken tek faktördür ve bu Gaussian fonksiyonlarının bir sapmasıdır:

- Çok küçük sapmalar iyi genelleştirilemeyen çok sivri yaklaşımlara neden olur.
- Çok büyük sapmalar detayları düzeltir.
- Uygun sapma deneysel seçilir.

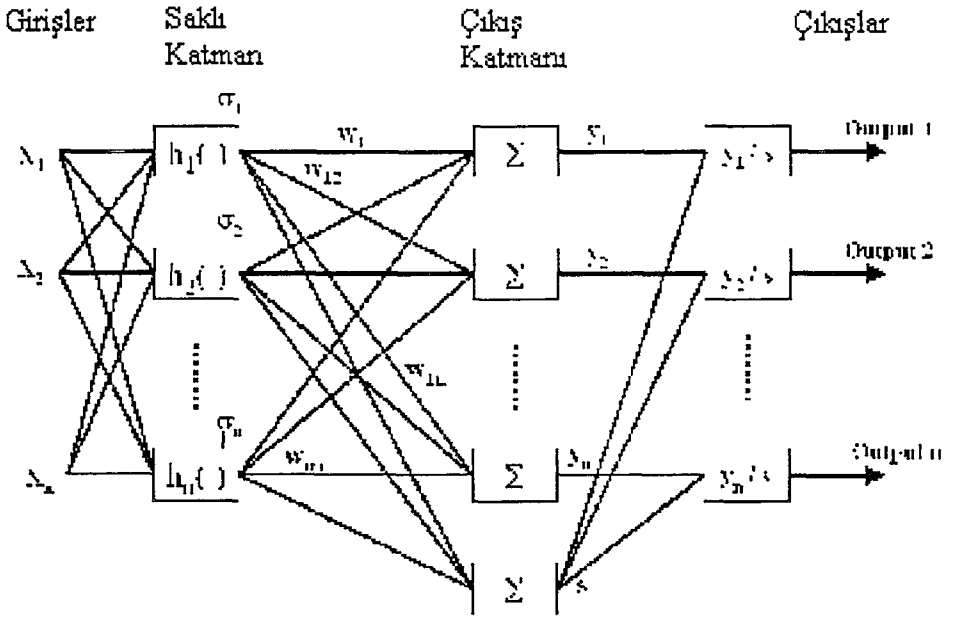
4.3.2 Olasılıksal Sinir Ağları ile Çok Katmanlı Algılayıcıların Karşılaştırması

Her iki ağ yapısıda ileri beslemeli olmalarına rağmen olasılıksal sinir ağları ile çok katmanlı algılayıcılar arasında bazı farklılıklar bulunmaktadır. Bu farklar şöyle belirtilebilir:

- Olasılıksal sinir ağları aşağıdaki avantajları sağlar:
- Eğitim hızının arttırılması: Olasılıksal sinir ağı geriye yayılımdan 5 katdan daha hızlıdır.
- Hızlı olan "artımsal eğitime" olanak verir bu da zor olamayan birleşmiş eğitim örneklerini sağlar.
- Gürültü örneklerine karşı duyarlıdır.
- Olasılıksal sinir ağları çok katmanlı sinir ağlarını eğitmek için bazı kullanışlı geriye yayılım karakteristiklerine sahiptir:
- Öğrenme kapasitesi: Verilen eğitim örnekleri ile bunların sınıflandırması arasındaki ilişkiyi sağlar.
- Genelleşitme kabiliyeti: Eğitim örneklerindeki genelleştirmeyi belirler ve önceden belirlenmiş sınıflarda görünmeyen örneklerin sınıflandırmasına olanak sağlar.

4.4 Genelleştirilmiş Regresyon Ağları (GRNN)

Genelleştirilmiş Regresyonlu Ağlar, radyal tabanlı ağların genellikle fonksiyon yaklaştırma problemleri için kullanılmakta olan özel bir halidir. Bu ağlar belirli sayıda saklı katman nöronu ile önemli ölçüde iyi başarı ile sürekli fonksiyonlara yaklaşımı sağlarlar. MLP' deki gibi tekrarlı eğitme işlemine ihtiyaç duymamaktadır. Giriş ve çıkış arasında, eğitim kümesinden elde ettiği bulgularla herhangi sıradan bir fonksiyona yaklaşabilir. Anlaşılacağı üzere, eğitim kümesinin boyutları büyüdükçe yaklaşımdaki hata oranı sıfira yakınsar. Şekil 4.4'de GRNN ağ yapısı verilmektedir.



Şekil 4.4 GRNN ağ yapısı

GRNN, standart teknikler gibi sürekli değişkenler üzerinde yargıya varılabilmesi içinde kullanılır. Temelinde standart bir istatistiksel yöntem olan Kernel yaklaşımını kullanmaktadır. Bu tanıma göre, bağımlı bir y değişkeninin bağımsız bir x değişkenine göre regresyonu, verilen x girişleri ve eğitim kümesine göre y için en çok olasılığa sahip değere yaklaşır. Yaklaşım yöntemi ortalama karesel hatayı en düşük değere yaklaştıracak şekilde belirlenir. GRNN, belirli bir eğitim kümesinde x ve y giriş ve çıkışları için bileşik olasılık yoğunluk fonksiyonunun da tahmini için kullanılmaktadır. Ağırlık matrisi w_{ij}, eğitilmez, eğitim setinden belirlenen hedef değerler ağırlık matrisi olarak atanır. (Avci, M.,Yıldırım T.)

GRNN yaklaşım problemlerinde X uzayındaki bir noktanın rasgele değişken Z değerinin en yüksek olasılığını bulmak için karar aşaması olarak kullanılır. Bunun gibi hesaplamalar yukarıda bahsedilen uzaydaki bazı sonlu noktaların Z değerlerini temel alarak yapılırlar. Maksimum komşuluk ilkesi ile, hatayı en aza indirmek için hedef fonksiyonları altında ortalama karesel hata bulunur. GRNN, parametric olmayan bir hesaplayıcıdır. Yani, problemin kararı için ilk verilen model kullanılmaz. Z'nin X uzayındaki koordinatlarıyla birleşmesi sadece Olasılıksal Yoğunluk Fonksiyonu'nun (PDF) ortalamasıyla olur. Eğer pdf değeri biliniyorsa Z değeri aşağıdaki formül ile bulunur.

$$Z = \frac{\int_{-\infty}^{+\infty} Z\, \phi(X,Z)\, dZ}{\int_{-\infty}^{+\infty} \phi(X,Z)\, dZ} \tag{4.8}$$

Burada $\phi(X,Z)$ fonksiyonu Parzen'in parametric olmayan metoduyla hesaplanan koşulsal olasılık fonksiyonunu belirtir.

$$\phi(X,Z) = \frac{1}{(2\pi)^{3/2} h^3 n} \sum_{i=1}^{n} \exp\left(-D_i^2 / 2h^2\right) \exp\left(-(Z - Z^i)^2 / 2h^2\right) \tag{4.9}$$

$$D_i^2 = (x - x^i)^2 + (y - y^i)^2 \tag{4.10}$$

(4.10) denklemi, (4.9)'da yerine koyularak aşağıdaki denklem elde edilir.

$$Z_m(X) = \frac{\sum_{i=1}^{n} Z_i \exp\left(-D_i^2 / 2h^2\right)}{\sum_{i=1}^{n} \exp\left(-D_i^2 / 2h^2\right)} \tag{4.11}$$

veya

$$Z_m(X) = \frac{\sum_{i=1}^{n} Z_i w_i}{\sum_{i=1}^{n} w_i} \tag{4.12}$$

$$w_i = \exp\left(-D_i^2 / 2h^2\right) \tag{4.13}$$

burada W_i ağırlıkları, D_i hesaplanan nokta ile merkez arasındaki X koordinatı üzerindeki uzaklıktır. (Specht,1991)

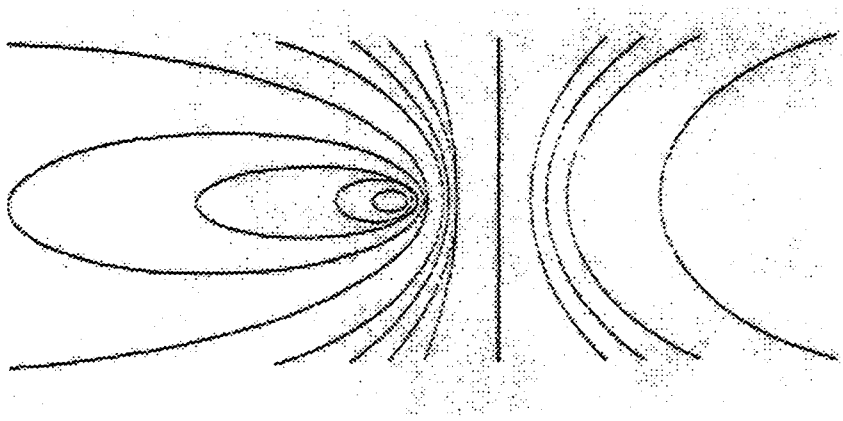
5. KONİK KESİT FONKSİYONLU SİNİR AĞLARI

Yapay sinir ağları kullanılarak yapılacak sınıflama işi, geometrik terimler kullanılarak tarif edilebilir. Eğer girişler ayrılabilir iki sınıfa ait ise, yani girişler bir hiperdüzlemin iki tarafında yer alıyorsa, bu durumda algılayıcı yaklaşımı yöntemi başarılı olacak ve hiperdüzlemin bu iki sınıf arasında yerleşmesini sağlayacaktır. Eğer ayrılacak girişler, giriş uzayının belli bölgelerinde kümelenmişlerse bu durumda bir veya daha fazla gizli katmana sahip bir MLP ağı problemi çözecektir. Sınıflama işleminde karşılaşılabilecek diğer karar bölgeleri ise lokal olanlardır. Buna örnek olarak hiperküresel karar bölgeleri verilebilir. Giriş uzayının sonsuz bir kısmını içeren hiperdüzlemsel (açık) karar bölgelerinden farklı olarak, hiperküresel (kapalı) karar bölgelerinde her bir birimin alıcı alanı lokaldir ve giriş uzayının küçük bir bölümü ile sınırlıdır. Bir RBF ağı ile hiperküresel karar sınırlarını elde etmek mümkündür.

Konik Kesit Fonksiyonlu Yapay Sinir Ağı ilk önce Dorffner tarafından tanımlanmıştır (Dorffner,1994; Dorffner ve Porenta,1994). Bu ağ, hiperdüzlem (düz çizgi) ve hiperkürelerin (daire) konik kesit fonksiyonlarının özel durumları olmasını esas alan bir sinir ağı yapısıdır. Bunlar, sırasıyla, Çok Katmanlı Algılayıcı (MLP) ve Radyal Temelli Fonksiyonlu (RBF) ağların karar sınırlarıdır. Bu iki durumun arasında yine karar bölgesi olarak geçerli olan, bazıları sınırsız ve sonsuz, bazıları sınırlı ve lokal olan elipsler, hiperboller ve paraboller gibi karar sınırları da vardır. Konik Kesit Fonksiyonlu Yapay Sinir Ağında esas fikir, bir Çok Katmanlı Algılayıcı birimi ile bir Radyal Temelli Fonksiyonlu ağ birimi arasında ilişki sağlayarak bütün bu karar bilgilerini içine alacak bir birimin fonksiyonunu genelleştirmektir. Konik Kesit Fonksiyonlu Yapay Sinir Ağı, giriş uzayını, verilen girişleri farklı sınıflara kolayca ayırabilecek lokalleşmiş bölgelere ayırır ve kapalı (hiperküre) karar sınırlarını açık (hiperdüzlem) olanlara veya açıkları kapalılara çevirebilir ve bu bölgeleri uygun olduğu yerlerde kullanabilir. Bölgenin tipi verilen uygulamadaki veri dağılımına bağlıdır.

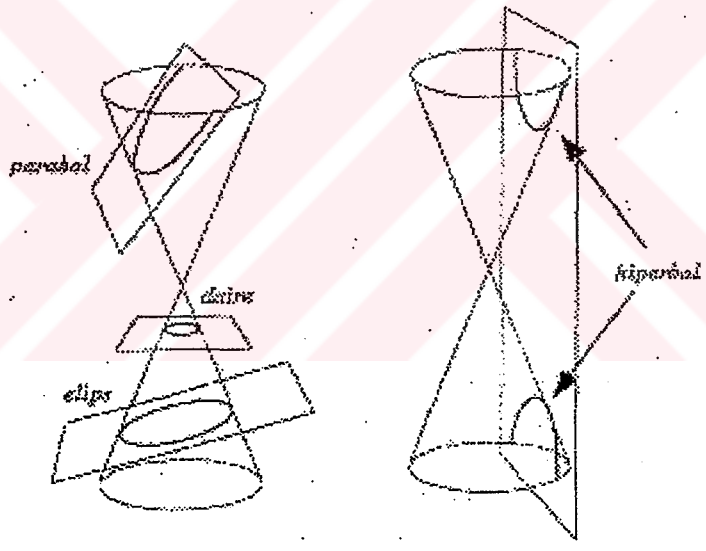
5.1 Konik Kesitler

Matematiksel olarak konik kesitler, bir düzlemde sabit bir noktaya olan mesafeleri ile bir doğruya olan mesafelerinin oranlarının birbirine eşit olduğu noktaların yer eğrisidir. Eğrinin şekli bu oran ile belirlenir. Bu oran koninin dış merkezliği olarak adlandırılır ve e ile gösterilir. $0< e <1$ ise konik kesit bir elips, $e=1$ ise parabol ve $e >1$ ise hiperboldür. e'nin aldığı değerlere göre karşılık gelen eğriler Şekil 5.1'de görülmektedir.



Şekil 5.1 e'nin aldığı değerlere göre karşılık gelen eğriler (Özyılmaz, 2000)

Konik kesitler, bir düzlem ve bir koninin kesiştirilmesi ile oluşan arakesitlerdir. Bir koninin bir düzlemle kesişmesi sonucunda oluşan ara kesit eğrileri Şekil 5.2'de görülmektedir. Bu şekiller koninin eksenleri ile düzlemin yaptığı değişik açılara bağlı olarak elde edilirler.

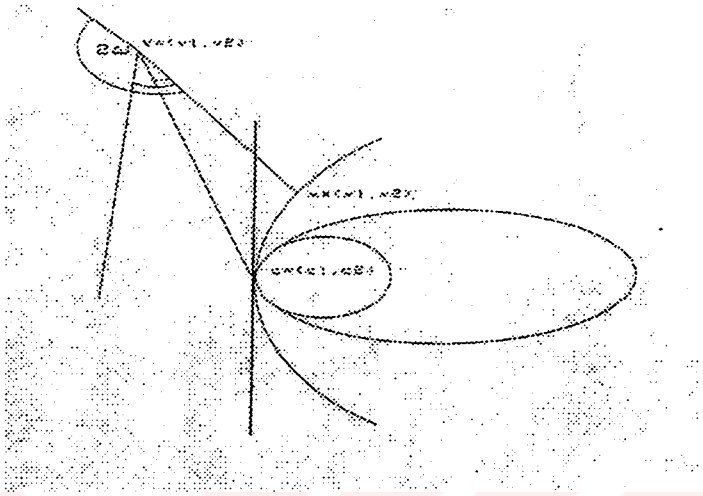


Şekil 5.2 Bir koninin bir düzlemle kesişmesi sonucunda oluşan ara kesit eğrileri (Özyılmaz,2000)

Bir koni üzerinde herhangi bir nokta alınsın ve x noktası olarak isimlendirilsin. Koninin tepe açısı 2ω olsun. Tepe açısı, $[-\pi/2,\pi/2]$ aralığında bir değer olabilir. v koninin tepe noktası ve a da koninin eksenini tanımlayan birim vektör olsun. Bu durumda analitik olarak koni şu şekilde tanımlanabilir.

$$(x - v)a = \cos \omega \|x - v\| \qquad (5.1)$$

Şekil 5.3'de tepe noktası V ve tepe açısı ω olan üç-boyutlu bir koninin bir daire, bir parabol ve bir düzlem oluşturacak şekilde bir düzlemle yaptığı arakesitler iki-boyutlu uzayda görülmektedir. Bu şekil, doğrusal (hiperdüzlem) ve dairesel (hiperküre) olmak üzere sırasıyla MLP ve RBF için karar sınırlarını göstermektedir. Diğer karar sınırları elips ve paraboldür.



Şekil 5.3: Tepe noktası V ve tepe açısı ω olan üç-boyutlu bir koninin bir daire, bir parabol ve bir düzlem oluşturacak şekilde bir düzlemle yaptığı arakesitler (Özyılmaz, 2000)

Eşitlik (5.1)'deki nokta ve vektörlerin koordinatları iki-boyutlu uzay için aşağıdaki gibi tanımlanır.

$$(x_1 - v_1)a_1 + (x_2 - v_2)a_2 = \cos\omega\sqrt{(x_1 - v_1)^2 - (x_2 - v_2)^2} \tag{5.2}$$

Yukarıdaki eşitlik n-boyutlu giriş uzayı için şu şekilde geliştirilebilir.

$$\sum_{i=1}^{n+1}(x_i - v_i)a_i = \cos\omega\sqrt{\sum_{i=1}^{n+1}(x_i - v_i)^2} \tag{5.3}$$

Eşitliğin bu şekli n-boyutlu giriş uzayı için hiperkoni ve hiperdüzlem arasındaki kesişimi ifade etmektedir. Koninin tepe noktası koordinatı olan v yerine daire merkezi c kullanılabilir. Çünkü x noktası ve v arasındaki mesafe, 2ω ile verilen koninin tepe açısı 90° olduğunda dairenin yarı çapına eşittir. (Dorffner,1994)

5.2 Konik Kesit Fonksiyonlu Yapay Sinir Ağı

n katmanlı bir ağ için Çok Katmanlı Algılayıcı'nın çıkış fonksiyonu aşağıdaki eşitlikle verilir.

$$y_j = f(\Sigma w_{ij} x_j - \theta_j) \tag{5.4}$$

Burada w_{ij} önceki katmana olan ağırlıklar, x_j önceki katmanın çıkışı, θ_j eşik ve f sigmoid fonksiyonu gibi bir eşik fonksiyonudur.

Radyal Temelli Fonksiyonlu (RBF) ağlar, sadece bir gizli katmana sahiptir. Gizli katmandaki temel fonksiyonlar alıcı alanlarla giriş vektörleri arasındaki Öklid mesafesini hesaplayarak giriş vektörünün lokalleşmiş bir cevabını üretirler. Bir Radyal Temelli Fonksiyonlu (RBF) ağın genel formu;

$$y_j = \Sigma w_{ij} \Phi_j \left(\| x - c_j \| \right) \tag{5.5}$$

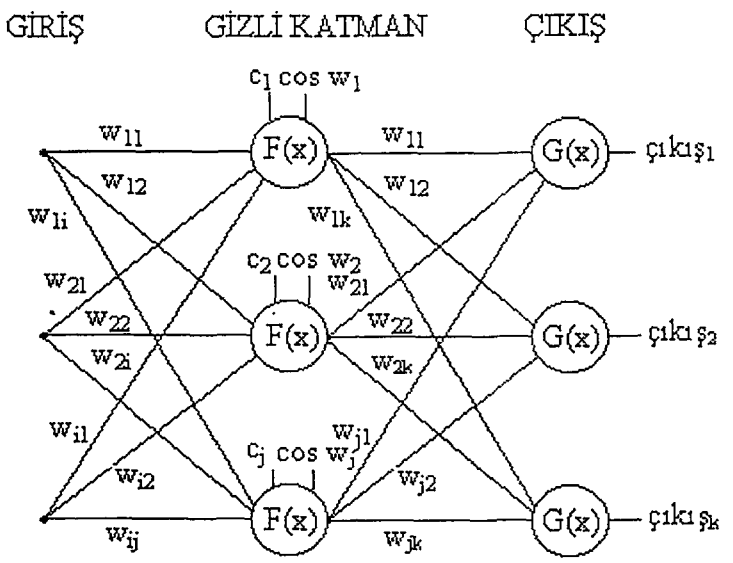
şeklinde verilir. Burada y_j ağın çıkışı, w_{ij} ikinci katmanın ağırlıkları, ϕ_j nonlineerlik, c_j merkezler ve x ağın girişidir.

Eşitlik (6.3)'den

$$y_j = \sum (x - c_{ij}) w_{ij} - \cos \omega_j \sqrt{\sum_{i=1}^{n+1} (x_i - c_{ij})^2} \tag{5.6}$$

$$x_{n+1} \equiv 0 \tag{5.7}$$

edilir. Bu eşitlik CSFNN için genel aktivasyon ifadesini vermektedir. Burada, i ve j sırasıyla giriş katmanındaki ve gizli katmandaki birimlere karşı gelen indislerdir. Kolaylıkla fark edilebileceği gibi bu eşitlikte MLP ve RBF'tekine benzeyen iki temel kısım vardır. Burada, w_{ij} faktörleri Çok Katmanlı Algılaycı'daki ağırlıkları ve c_{ij} faktörleri de Radyal Temelli Fonksiyonlu ağ yapısından Çok Katmanlı Algılayıcı ağ yapısına geçişi sağlayan koninin açılma açısıdır. Eşitlik (5.3) dikkatle incelenirse şu sonuçlara varılabilir: ω açısı π/2'ye eşit olduğu zaman eşitliğin ikinci kısmı sıfır olur ve sadece birinci kısım etkin kalır. Yani, ağ bir MLP ağı gibi davranır. İkinci kısım ise bir RBF ağındaki merkezler ile girişler arasındaki Öklid mesafesini vermektedir. Şekil 5.4'de CSFNN'nin genel yapısı görülmektedir.



Şekil 5.4 CSFNN'nin genel yapısı

Konik Kesit Fonksiyonlu Yapay Sinir Ağı'nın MLP ve RBF'e göre avantajları şu şekilde belirtilebilir:

- Bu ağ yapısında öğrenme daha hızlıdır. Burada RBF'teki hız ve MLP'deki hata minimizasyonu birleştirilmiştir. Buda karışık problemler ve standart RBF'le karşılaştırıldığında yüksek boyutlu girişler için etkili çözümler verir.
- RBF, MLP'den daha hızlı yakınsamasına karşın, radyal temelli fonksiyonlar giriş uzayının boyutuyla eksponansiyel olarak arttığı için giriş verisinin boyutu yüksek olduğunda RBF ağları zaman tüketicidir. Bu nedenle yüksek boyutlu girişlere sahip problemlerde MLP tercih edilir. Dolayısıyla iki ağ yapısı arasında geçişi sağlayan CSFNN verilen uygulamaya göre optimum ağ çözümüne izin verir. (Dorffner, 1994; Yıldırım, 1997; Özyılmaz, 2000)
- Kapalı karar sınırlarını açık sınırlara veya benzerlerine çevirme
- Giriş uzayını belirli bir yerle sınırlanmış bölgelere bölme
- Hızlı öğrenme
- Gizli katmanları daha etkili kullanma
- Hiperküresel ağların hızı ile geriye yayılım algoritmasının hata minimizasyonunu birleştirme
- Daha karmaşık problemler ve daha çok boyutlu girişler için daha iyi verimlilik

5.3 Konik Kesit Fonksiyonlu Yapay Sinir Ağının Eğitilmesi

Konik Kesit Fonksiyonlu Yapay Sinir Ağının eğitilmesi, birim merkezlerinin, bağlantı ağırlıklarının ve açılma açısının belirlenerek belli bir hata kriterine veya sınıflama oranına ulaşılıncaya kadar bu parametrelerin güncelleştirilmesi ile yapılır. CSFNN'nin eğitme fazları ve öğrenme yöntemleri aşağıdaki gibi özetlenebilir.

Eğitme Fazları:

- Başlangıç fazı
  - Merkezlerin belirlenmesi
  - Ağırlık ve açılma açısının başlatılması
- Öğrenme fazı

5.4 Öğrenme Yöntemleri

5.4.1 Koni katlama yöntemi ile öğrenme

Bu yöntemde CSFNN, MLP, önceden eğitilmiş MLP veya RBF ağları gibi başlatılır. Bir gizli birimin eşik ve ağırlık değerleri rasgele veya bir kümeleme işlemi ile belirlenir. Gizli katmandan çıkışa olan ağırlıklar delta kuralı ile bulunabilir veya tüm ağ önceden belirlemiş ağırlıklar kullanılarak standart geriye yayılma algoritması ile eğitilebilir. (Dorffner, 1994; Özyılmaz, 2000)

5.4.2 Geriye yayılmalı öğrenme

Bu yöntemde ağ, RBF ağı gibi başlatılır. Ortogonal En Küçük Kareler yöntemi ile merkezler belirlendikten sonra ağ, standart geriye yayılma algoritması ile eğitilir. Burada sadece ağırlıklar değil aynı zamanda merkezler ve açılma açısı da güncelleştirilir. (Yıldırım ve Marsland, 1997; Özyılmaz, 2000) Güncelleştirmede iki yol izlenir:

- Aynı iterasyonda ω açılma açısı, merkezler ve ağırlıkların güncelleştirilmesi
- Parametrelerin farklı iterasyonlarda güncelleştirilmesi

5.5 Adaptif Öğrenmeli Konik Kesit Fonksiyonlu Yapay Sinir Ağı

Adaptif Öğrenmeli Konik Kesit Fonksiyonlu Yapay Sinir Ağı, başlangıç (parametrelerin başlatılması) ve öğrenme (parametrelerin güncelleştirilmesi) fazı olmak üzere iki fazda eğitilir:

Başlangıç fazında merkezlerin yerleri belirlenir; ayrıca ω ile verilen koninin açılma açısı ve

katmanlar arası bağlantı ağırlıkları başlatılır. Bu fazda merkezler belirlenirken iki farklı yol izlenmiştir:

- Önceden belirlenen bir merkez sayısına ulaşıncaya kadar Ortogonal En Küçük Kareler yöntemi kullanılarak merkez değerleri belirlenir.
- Merkez değerleri sıfira atanır.

Koninin açılma açısı, ağın başlangıçta Radyal Temelli Fonksiyonlu Ağ gibi davranmasını sağlayacak şekilde seçilir. Ağırlıklar belirlenirken, girişten gizli katmana olan ağırlıklar sıfira atanır. Gizli katmandan çıkışa olan bağlantı ağırlıkları ise rasgele başlatılır.

Öğrenme fazı merkezlerin belirlenme yöntemine göre iki farklı şekilde yapılmaktadır. Eğer merkez değerleri sıfira atanmış ise Çok Katmanlı Algılayıcılarda kullanılan hatanın geriye yayılımı yöntemi ile tüm parametreler (ağırlıklar, merkezler, açılma açısı) farklı iterasyonlarda sırayla güncelleştirilir. Merkez değerlerinin OLS ile belirlendiği durumda ise ağ, önceden belirlenen bir merkez sayısına ulaşıncaya kadar RBF ağları gibi eğitilir. Daha sonra geriye yayılım yöntemi ile dokuz iterasyon boyunca ağırlıklar değiştirilir, sonraki bir iterasyonda da açılma açısı güncelleştirilir. Her iki durumda da güncelleştirme işlemi sırasında, geriye yayılma yönteminde kullanılan öğrenme oranı algoritmanın performansına göre değiştirilmektedir. Ayrıca algoritmaya momentum terimi eklenerek öğrenmenin iyileştirilmesine çalışılmıştır. Eğitme işlemi ya önceden belirlenen kabul edilebilir bir hata değerine ulaşıldığında ya da sınıflandırma işleminin tamamlanması halinde sonuçlandırılır.

Eğitme algoritması adım adım verilecek olursa;

Adım 1: Merkez sayısı ve toplam karesel hata veya sınıflanacak örnek sayısı belirlenir. Birinci katman ağırlıkları sıfira atanır ve ω açılma açısı $\pi/4$'den başlar.

Adım 2: Merkezler sıfira atanır veya OLS algoritması kullanılarak girişlerden üretilir. Gizli katman ve ikinci katman çıkışları hesaplanır.

Adım 3: Toplam karesel hata ve sınıflanan örnek sayısı hesaplanır.

Adım 4: Katman ağırlıklarını, duruma bağlı olarak merkezleri ve gizli katmandaki ω açılma açısını güncellemek için adaptif geriye yayılma algoritması çalıştırılır. Her bir katmandaki δ işaret vektörleri hesaplanır.

Adım 5: Çıkış katmanlarının ağırlıkları ayarlanır.

Adım 6: Gizli katman ağırlıkları, merkezler ve açılma açısı güncelleştirilir.

Adım 7: Katmanların çıkışları yeniden hesaplanır.

Adım 8: Yeni toplam karesel hata ve sınıflanan örnek sayısı hesaplanır. Eğer bu değer, hata hedefinden büyükse veya istenen sayıda örnek sınıflanamadıysa 4. adıma gidilir. Aksi taktirde eğitmeye son verilir.

5.6 Merkezlerin Belirlenmesi İçin Kullanılan Ortogonal En Az Kareler Yöntemi

Ortogonal En Az Kareler Yöntemi (OLS) yöntemi minimum sayıda merkez seçimi için kullanılmaktadır. (Chen vd.,1991). Merkezler, RBF ağlarının eğitiminde olduğu gibi veri tabanından seçilir.

$$d(n) = \sum_{i=1}^{M} p_i(n)\theta_i + e(n) \qquad n = 1,2,..., N \tag{5.8}$$

Burada $d(n)$, arzu edilen çıkıştır ve aynı zamanda bağımlı değişken olarak adlandırılır. θ_i model parametreleri ve $p_i(n)$, regresörlerdir ve aynı zamanda $x(n)$'*in* sabit fonksiyonlarıdır.

$$p_i(n) = p_i(x(n)) \tag{5.9}$$

Hata işareti $e(n)$, regresörlerle ilişkisiz varsayılır. Eşitlik (5.8) matris formunda yazılırsa;

$$d = P\theta + E \tag{5.10}$$

burada

$$d = [d(1), d(2),..., d(N)]^T \tag{5.11}$$

$$P = [p_1, p_2, ..., p_M]^T \tag{5.12}$$

$$\theta = [\theta_1, \theta_2, ..., \theta_M]^T \qquad 1 \le i \le M \tag{5.13}$$

$$E = [e(1), e(2), ..., e(N)]^T \tag{5.14}$$

Bir temel vektör kümesi içinden p_i regresör vektörleri alınır ve en küçük kareler çözümü θ, bu vektörler tarafından d'nin ortogonal temelli vektörlerle taranan uzay üzerindeki izdüşümünün $P\theta$ olması koşulunu sağlar. Diğer bir deyişle, izdüşümün karesi, istenen çıkış enerjisinin bir parçasıdır. Farklı regresörler genel olarak ilişkilidir. Tek bir regresörün çıkış

enerjisini nasıl etkileyeceği açık değildir. OLS metodu, p_i dizilerinin bunlara karşı gelen bir dizi ortogonal temel vektör kümesine dönüşümünü içerir ve böylelikle her bir temel vektörden istenen çıkış enerjisine bireysel olarak p_i'nin katkısının hesaplanabilmesi mümkün olur. P regresör matrisi,

$$P=WA \tag{5.15}$$

Şeklinde ayrıştırılır. Burada A, MxM boyutunda, köşegenlerinde 1 olan ve 1'lerin altında da sıfır bulunan bir kare matristir. W, NxM boyutunda, köşegen sütunları w_i olan bir matristir.

$$W^T W = H \tag{5.16}$$

Burada H, h_i elemanlarından oluşur.

$$h_i = w_i^T w_i = \sum_{n=1}^{N} w_i(n) w_i(t),\ 1 \le i \le M \tag{5.17}$$

Ortogonal temel w_i vektörlerinin bir kümesi ile taranan uzay, p_i dizisinde taranan uzayla aynıdır ve eşitlik (5.10) eşitlik (5.18) gibi yeniden yazılabilir:

$$D= Wg + E \tag{5.18}$$

Ortogonal en az kareler çözümü aşağıdaki gibi verilebilir.

$$g = H^{-1} W^T d \tag{5.19}$$

$$g_i = w_i^T d / (w_i^T w_i),\ 1 \le i \le M \tag{5.20}$$

Klasik Gram-Schmidt yöntemi bir anda A'nın bir stününu hesaplar ve P'yi aşağıdaki gibi ortogonalleştirir:

$$w_i = p_i \tag{5.21}$$

$$\alpha_{ik} = w_i^T p_k / (w_i^T w_i),\ 1 \le i \le k \tag{5.22}$$

$$w_k = p_k - \sum_{i=1}^{k-1} \alpha_{ik} w_i \tag{5.23}$$

burada k=2,...,M'dir. Bu yöntemle k. Sütun, k. Adımda önceden ortogonalleştirilmiş k-1 tane sütunun her birine ortogonal yapılır ve işlem k=2,...,M için tekrarlanır. OLS metodu normal LS metoduna göre daha üstün sayısal özelliklere sahiptir. RBF ağlarında *x(n)* veri noktalarının sayısı çoğunlukla çok büyüktür ve veri kümesinde merkezler alt küme olarak seçilme durumundadır. Genel olarak M ile gösterilen bütün aday regresörlerin sayısı çok büyük olabilir ve uygun modelleme için sadece önemli regresörlere ihtiyaç duyulabilir. Bu önemli regresörler, OLS algoritması kullanılarak seçilebilir. *d(n)*'in enerjisi veya karelerinin toplamı

$$d^T d = \sum_{i=1}^{M} g_i^2 w_i^T w_i + E^T E \tag{5.24}$$

ile ifade edilir. Eğer *d* istenen çıkış vektörü ise *d(n)* 'nin değişimi şöyle verilebilir.

$$N^{-1} d^T d = N^{-1} \sum_{i=1}^{M} g_i^2 w_i^T w_i + N^{-1} E^T E \tag{5.25}$$

Hatadaki azalma aşağıdaki gibi tanımlanabilir:

$$e_i = g_i^2 w_i^T w_i / (d^T d),\ 1 \le i \le M \tag{5.26}$$

Regresör seçimini özetlersek:

Birinci adımda $1 < i < M$ için,

$$w_1^{(i)} = p_i \tag{5.27}$$

$$g_1^{(i)} = (w_1^{(i)})^T d / ((w_1^{(i)})^T w_1^{(i)}),\ 1 \le i < k \tag{5.28}$$

$$[err] = (g_1^{(i)})^2 (w_1^{(i)})^T w_1^{(i)} / (d^T d) \tag{5.29}$$

hesaplanır. Hesaplanan hata değerleri içindeki en büyük hatanın indisi tespit edilir ve buna göre

$$w_1 = w_1^{(i)} = p_i \tag{5.30}$$

$$[err]_1^{(i1)} = max\,([err]_1^{(i)},\ 1 \le i \le M) \tag{5.31}$$

seçilir.

K.adımda $k \geq 2$ ve $1 \leq i \leq M$

$$\alpha_{jk}^{(i)} = w^T{}_j p_i / (w_j{}^T w_j), \quad 1 \leq i < k \tag{5.32}$$

$$w_k{}^{(i)} = p_i - \sum_{j=1}^{k-1} \alpha_{jk}{}^{(i)} w_j \tag{5.33}$$

$$g_k{}^{(i)} = (w_k{}^{(i)})^T d / ((w_k{}^{(i)})^T w_k{}^{(i)}) \tag{5.34}$$

$$[err]_k{}^{(i)} = (g_{ki}{}^{(i)})^2 (w_k{}^{(i)})^T w_k{}^{(i)} / (d^T d) \tag{5.35}$$

hesaplanır ve aynı şekilde her i. Adımda hesaplanan hataların içinden en büyük olanının indisi tespit edilip w'ya atamalar yapılır.

$$[err]_k^{(ik)} = max([err]_k^{(i)}), \; 1 \leq i \leq M \tag{5.36}$$

$$w_k = w_k^{(ik)} = p_{ik} - \sum_{j=1}^{k-1} \alpha_{jk} w_j \tag{5.37}$$

$$\alpha_{jk} = \alpha_{jk}{}^{(i_k)}, \; 1 \leq i \leq k \tag{5.38}$$

Bu işlem M. Adımda

$$1 - \sum_{j=1}^{m} [err]_j < \rho \tag{5.39}$$

tamamlanır. Tolerans, $0 < \rho < 1$ arasında tanımlanmıştır. Birinci adımda ilk merkez yerleştirilir ve hata terimi hesaplanır. k.adımda, yeterli bir ağ yapısı elde edinceye kadar merkezler birer birer eklenir.

5.7 Ağırlıkların Güncelleştirilmesi

İleri beslemeli ağlarda aktivasyon fonksiyonu eşitlik (5.40)'deki gibi tanımlanabilir.

$$out_{pj} = f_j (y_{pj}) \tag{5.40}$$

d istenen çıkış ve *out* ağın yanıtı olmak üzere; *p.* patern için toplam karesel hata

$$E_p = \frac{1}{2}\sum_j \left(d_j - out_j\right)^2 \tag{5.41}$$

şeklinde tanımlanır. Karesel hatayı minimum yapmak için gradyen azalma algoritması kullanılır. Karesel hata fonksiyonunun w'ya göre gradyeninin negatif doğrultusunda hareket edilerek hata azaltılır. Buna göre herhandi bir ağırlık için değişim şu şekilde tanımlanabilir.

$$\Delta_p w_{ij} = -\alpha \frac{\partial E_p}{\partial w_{ij}} \tag{5.42}$$

Zincir kuralı ile (5.42) eşitliğinin ikinci kısmı aşağıdaki formda yazılabilir:

$$\frac{\partial E_p}{\partial w_{ij}} = \frac{\partial E_p}{\partial y_{pj}} \cdot \frac{\partial y_{pj}}{\partial w_{ij}} \tag{5.43}$$

Eşitliğin sağ tarafında çarpım halinde olan iki terim ayrı ayrı hesaplanırsa;

$$\frac{\partial y_{pj}}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}}\left[\sum_{k=1}\left(out_{pk} - c_{kj}\right)w_{kj} - \cos\omega_j \sqrt{\sum_{k=1}\left(out_{pk} - c_{kj}\right)^2}\right] \tag{5.44}$$

buradan,

$$\frac{\partial y_{pj}}{\partial w_{ij}} = out_{pi} - c_{ij} \tag{5.45}$$

elde edilir.

$$\frac{\partial E_p}{\partial y_{pj}} = -\delta_{pj} \tag{5.46}$$

olarak tanımlanırsa, p. patern için toplam karesel hatanın w_{ij} ağırlığıyla değişimi

$$-\frac{\partial E_p}{\partial w_{ij}} = \delta_{pj}\left(out_{pi} - c_{ij}\right) \tag{5.47}$$

şeklinde yazılabilir. Dolayısıyla p. patern için w_{ij}'deki değişim

$$\Delta_p w_{ij} = \eta \delta_{pj} \left(out_{pi} - c_{ij} \right) \tag{5.48}$$

olarak bulunur. Hata işaret terimi;

$$\delta_{pj} = -\frac{\partial E_p}{\partial y_{pj}} = -\frac{\partial E_p}{\partial out_{pj}} . \frac{\partial out_{pj}}{\partial y_{pj}} \tag{5.49}$$

zincir kuralı ile 5.31 eşitliğindeki gibi tanımlanırsa, çarpımın ikinci terimi;

$$\frac{\partial out_{pj}}{\partial y_{pj}} = f'_j \left(y_{pj} \right) \tag{5.50}$$

dir. Çarpımın birinci terimi de

$$\frac{\partial E_p}{\partial out_{pj}} = -\left(d_{pj} - out_{pj} \right) \tag{5.51}$$

şeklinde elde edilir. Sonuç olarak hata işaret terimi,

$$\delta_{pj} = \left(d_{pj} - out_{pj} \right) f'_j \left(y_{pj} \right) \tag{5.52}$$

olarak bulunur. Gizli katman ağırlıklarının ayarlanmasında bütün katman çıkışlarının hata terimleri etkilidir. Buna göre;

$$\sum_k \frac{\partial E_p}{\partial y_{pk}} . \frac{\partial y_{pk}}{\partial out_{pj}} = \sum_k \frac{\partial E_p}{\partial y_{pk}} . \frac{\partial}{\partial out_{pj}} \left[\sum_i \left(out_{pi} - c_{ik} \right) w_{ik} - \cos \omega_k \left\| out - c_j \right\| \right] \tag{5.53}$$

$$\frac{\partial}{\partial out_{pj}} \left[\left[\sum_i \left(out_{pi} - c_{ik} \right) w_{ik} - \cos \omega_k \left\| out - c_j \right\| \right] \right] = w_{jk} - \cos_k \frac{out_j - c_j}{\left\| out_p - c_j \right\|} \tag{5.54}$$

$$\sum_k \frac{\partial E_p}{\partial y_{pk}} . \frac{\partial y_{pk}}{\partial out_{pj}} = -\sum_k \delta_{pk} \left[w_{jk} - \cos_k \frac{out_j - c_j}{\left\| out_p - c_j \right\|} \right] = A \tag{5.55}$$

Gizli katman birimleri için hata işaret terimi tekrar yazılacak olursa

$$\delta_{pj} = f'_j \left(y_{pj} \right) A \tag{5.56}$$

Sonuç olarak ağırlıkların güncelleştirilmesi çıkış katmanı için:

$$\Delta_p w_{ji} = \eta out_{pj}(1 - out_{pj}) A(out_{pi} - c_{ij}) \tag{5.57}$$

olarak tanımlanabilir. Burada aktivasyon fonksiyonu olarak logaritmik sigmoid fonksiyonu kullanılmıştır.

$$out_{pj} = \frac{1}{1 + e^{-y_{pj}}} \tag{5.58}$$

Sigmoid fonksiyonunun türevi ise 5.59'da verilmiştir.

$$f'_j(y_{pj}) = \frac{\partial out_{pj}}{\partial y_{pj}} = out_{pj}(1 - out_{pj}) \tag{5.59}$$

5.8 Merkezlerin güncelleştirilmesi

Merkezlerin güncelleştirilme işlemi de, ağırlıkların güncelleştirilmesine benzer şekilde genelleştirilmiş delta kuralından yararlanılarak yapılır.

$$\Delta_p c_{ij} \infty - \frac{\partial E_p}{\partial c_{ij}} \tag{5.60}$$

Hatanın merkezlere göre değişimini bulmak için zincir kuralı uygulanarak; (5.60) eşitliği iki parça halinde yazılabilir: Toplam karesel hatanın aktivasyona göre türevi ve aktivasyonun merkezlere göre türevi.

$$\frac{\partial E_p}{\partial c_{ij}} = \frac{\partial E_p}{\partial y_{pj}} \frac{\partial y_{pj}}{\partial c_{ij}} = -\delta_{pj} \cdot \frac{\partial y_{pj}}{\partial c_{ij}} \tag{5.61}$$

eşitliğindeki ikinci terim

$$\frac{\partial y_{pj}}{\partial c_{ij}} = -w_{ij} + \cos\omega_j \frac{a_i - c_i}{\|a_i - c_i\|} = B \tag{5.62}$$

olarak bulunur. Gizli katmandaki merkezlerin ayarlanmasında yine çıkış katmanındaki hata terimlerinin etkisi vardır. Buna göre, gizli katman birimleri için merkezlerin güncelleştirilmesi aşağıdaki gibi tanımlanabilir:

$$\Delta_p c_{ij} \infty out_{pj}(1 - out_{pj})A.B \tag{5.63}$$

Çıkış katmanındaki bütün düğümler için, $\cos\omega = 0$ yani, CSFNN ifadesindeki (Eşitlik (5.6)) RBF kısmının etkisi olmadığı için merkezler güncelleştirilmez.

5.9 Açılma açısının güncelleştirilmesi

Açılma açısının güncelleştirilmesi, yine genelleştirilmiş delta kuralından yararlanılarak yapılır. Bunun için kural

$$\Delta_p w_j \infty - \frac{\partial E_p}{\partial w_j} \tag{5.64}$$

ile tanımlanır. Hatanın açılma açısına göre türevi,

$$\frac{\partial E_p}{\partial w_j} = \frac{\partial E_p}{\partial y_{pj}} \frac{\partial y_{pj}}{\partial w_j} = -\delta_{pj} \cdot \frac{\partial y_{pj}}{\partial w_j} \tag{5.65}$$

ile verilir. Burada çarpımın ikinci türevi,

$$\frac{\partial y_{pj}}{\partial w_j} = \sin w_j \left\| out - c_j \right\| = C \tag{5.66}$$

olarak bulunur. Gizli katman birimleri için açı güncelleştirilmesi, çıkış katmanından gelen hata terimlerinin de etkisi hesaba katıldığında, çıkış katmanında ağın MLP özelliği gösterdiği ve CSFNN ifadesindeki (Eşitlik (5.6)) RBF kısmının çıkış katmanında etkili olmadığı düşünülmüştür. Bu nedenle, açılma açısının güncelleştirilmesi sadece gizli katman için yapılmaktadır.

5.10 Adaptif Öğrenme Oranı ve Momentum

Çok katmanlı ağlar için hata yüzeyinin şekli parametre uzayının farklı bölgelerinde çok farklı olabilir. Eğitme işlemi süresince öğrenme oranını ayarlayarak yakınsama hızlandırılabilir. Burada önemli olan öğrenme oranının ne zaman ve ne kadar değiştirileceğini belirlemektir. Adaptif öğrenme oranı ile sürekli olarak bir çevrimdeki hata *E(k)*, bir önceki çevrimdeki hata *E(k-1)* ile karşılaştırılmaktadır. Böylece eğimin arttığı yerlerde öğrenme oranı azaltılarak veya düz yüzeylerde öğrenme oranı arttırılarak yakınsamanın hızlanması sağlanabilir.

Yakınsamayı hızlandırmada kullanılan ikinci bir yöntemde momentum kullanarak ağın bir minimum hata civarında daha ileri bir öğrenme olmaksızın osilasyon yapmasını engellemek yani ağı daha kararlı hale getirmektir. Daha önce açıklandığı gibi momentum terimi, önceki ağırlık değişiminin etkisini belirleyen bir sabittir. Önceki ağırlık değişimi $\Delta w(k-1)$ büyükse, yeni değişim $\Delta w(k)$ de büyük olacaktır. Bu da, hata-ağırlık uzayındaki küçük dalgalanmaları düzleştirmeyi sağlar. Değişimin yönü doğruysa ve öğrenme minimuma doğru gidiyorsa; momentum, bu kararlılık etkisinden dolayı daha büyük öğrenme oranı değerlerine izin verir. Momentum terimi kullanılmadığı zaman düşük öğrenme oranındaki minimuma ulaşma çok zaman alır, yüksek öğrenme oranı için ise osilasyonlardan dolayı minimuma asla ulaşılamaz. Momentum eklendiği zaman minimuma daha hızlı ulaşılabilinir.(Zurada,1995;Hagan vd.,1996)

Konik Kesit Fonksiyonlu Yapay Sinir Ağı'nın eğitilmesinde yakınsamayı hızlandırmak için standart geriye yayılma algoritmasındaki öğrenme oranı adaptif yapılmış ve momentum terimi eklenmiştir. Öğrenme oranını ayarlamak için değişik yaklaşımlar vardır. Burada algoritmanın performansına göre öğrenme oranının değiştirildiği Değişken Öğrenme Oranlı geriye yayılmalı algoritmasının (Variable Learning Rate Backpropagation-VLBP) kuralları kullanılmıştır. Buna göre:

Eğer ağırlık güncelleştirilmesinden sonra karesel hata (tüm eğitim seti üzerinden) ξ oranında (genellikle %1.05'tir) artıyorsa, bu güncelleştirme işlemi iptal edilir. Öğrenme oranı, 0 ile 1 arasında olan bir faktörle çarpılır ve momentum katsayısı sıfıra atanır.

Eğer ağırlık güncelleştirilmesinden sonra karesel hata azalıyorsa, güncelleştirme işlemi geçerli sayılır ve öğrenme oranı 1'den büyük bir faktör ile çarpılır. Eğer momentum katsayısı sıfıra atanmışsa, bu iptal edilerek orijinal değerine ayarlanır.

Eğer karesel hata ξ oranından daha az artıyorsa, ağırlık güncelleştirilmesi geçerlidir. Fakat öğrenme oranı ve momentum katsayısı değiştirilemez.

6. SİMULASYON SONUÇLARI

Bu bölümde, 5. bölümde açıklanan sinir ağı yapılarının, California Üniversitesi tarafından hazırlanan UCI Makina Öğrenmesi veri kümesinde bulunan görüntü veri setine uygulanması ve MATLAB 6.0'da elde edilen simulasyon sonuçları verilmiştir.

6.1 Görüntü Veri Kümesi Özellikleri

Kullanılan veri seti EK 1'de sunulmaktadır. Veriler, rastgele seçilmiş 7 dışçevre resminden elde edilmiştir. Resimler sınıflandırma yapılmasını kolaylaştırabilmek için ilk önce 3x3 lük piksellere ayrılmıştır. Eğitim için 210 örnek, test içinse 2100 örnek kullanılmıştır. Bu veri seti 19 sürekli özelliğe sahiptir. Kayıp özellik bulunmamaktadır. 7 sınıf bulunmaktadır. Bunlar tuğla görünüm, gökyüzü, yapraklar, çimento, pencere, yol ve çimen'dir. Her sınıf için 30'ar tane eğitim örneği, 300'er tane de test örneği bulunmaktadır.

1. region centroid coloumn : Bölgenin merkezi pikselinin sütunu
2. region-centroid-row : Bölgenin merkezi pikselinin satırı
3. region-pixel-count : bir bölgedeki pixel sayısı = 9.
4. short-line-density-5 : line extraction algorithm sonuçları,düşük kontrastlı 5 uzunluklu çizgilerin sayısını sayar, eğer küçükeşit 5 ise bölgeye doğru gider.
5. short-line-density-2 : short-line-density-5 ile aynıdır ancak yüksek kontrastlı çizgileri sayar, büyükeşit 5 için.
6. vedge-mean : bölgedeki yatay bitişik pikseller arası kontrastı ölçer,. Bu özellik düşeyde kenarların belirlenmesi için kullanılır.
7. vegde-sd : 6. özellik gibi
8. hedge-mean : bölgedeki düşey bitişik pikseller arası kontrastı ölçer,. Bu özellik yatayda kenarların belirlenmesi için kullanılır.
9. hedge-sd : 8.özellik gibi.
10. Intensity - mean : (R + G + B) / 3 bölgesi üzerindeki ortalama
11. rawred-mean : kırmızı bölge (R) üzerindeki ortalama
12. rawblue-mean : mavi bölge (B) üzerindeki ortalama

13. rawgreen-mean : yeşil bölge (G) üzerindeki ortalama

14. exred-mean : kırmızı bölgeyi aşan kısmın değeri : (2R - (G + B))

15. exblue-mean : mavi bölgeyi aşan kısmın değeri : (2B - (G + R))

16. exgreen-mean : yeşil bölgeyi aşan kısmın değeri: (2G - (R + B))

17. value-mean : RGB'nin 3-d nonlinear dönüşümü

18. saturatoin-mean : 17. özellik gibi

19. hue-mean : 17. özellik gibi

6.2 Çok Katmanlı Algılayıcı Ağ Yapısı ile Görüntü Veri Kümesi İçin Simülasyon Sonuçları

Bölüm 5'te açıklanan çok katmanlı algılayıcı ağ yapısına bu veri kümesi Matlab 6.0'ın sağladığı bir çok eğitme algoritması ile uygulanmıştır. Bu algoritmalar kısaca aşağıdaki gibi açıklanabilir:

Traingd fonksiyonu steepest descent algoritmasını kullanmaktadır.

Traingdm ise en dik iniş algoritmasını momentum terimi ile birlikte uygulamaktadır. Momentum terimi ağın hata fonksiyonunda yerel minimuma takılmasını önleyerek daha doğru sonuçlara ulaşılmasını sağlar.

Traingda, en dik iniş algoritmasını değişken öğrenme oranı ile uygular. Böylece en dik iniş algoritmasının performansı da arttırılmış olur.

Traingdx, adaptif öğrenme oranı ile momentum terimini birleştiren fonksiyondur.

Trainrp, çok katmanlı algılayıcı ağlarında bazen ortaya çıkan sigmoidal fonksiyonun kullamından kaynaklanan hataların azaltılmasını sağlar. Ağırlıklar güncellenirken performans fonksiyonunun türevine signum fonksiyonunu uygular.

Traincgf, Fletcher-Reeves Güncellemesi yöntemini kullanır. Burada önemli olan şimdiki karesel normun bir önceki karesel norma olan oranıdır.

Traincgp, Polak-Ribiere Güncellemesi yöntemini kullanır. Burada da önemli olan türevdeki bir önceki değişimin iç çarpımıdır.

Traincgb, Powell-Beale Restarts yöntemini uygular. Burada da önemli olan şimdiki türev ile

bir önceki türev arasındaki ortagonalliktir.

Trainbfg, Quasi-Newton algoritmasını uygulamaktadır. Bu algoritma için ikinci türev hesaplanmaktadır ve önemli olan Hessian Matrisidir.

Trainoss, kek adım sekant, yukarıda bahsedilen eğitim algoritmalarının bazı limitasyonlarını engeller.

Trainlm, Levenberg - Marquardt algoritmasını uygulayarak Hessian Matrisinin hesaplanmasındaki zorlukları ortadan kaldırmaktadır.

Bu algoritmaların denenmesi ile elde edilen sonuçlar ve karşılaştırmalar Çizelge 6.1'de verilmiştir.

Çizelge 6.1 Çok katmanlı algılayıcı sinir ağı yapısında eğitme algoritmalarının karşılaştırılması

Öğrenme Algoritması	Açıklama	Hata Oranı	Eğitim Sınıflandırma Oranı
trainlm	Levenberg-Marquardt Algoritması	14.7e-5	99.53
traingd	En Dik İnişli Geriyeyayılım Algoritması	0.04	14.28
traingda	Adaptif Öğrenme Oranlı En Dik İnişli Geriyeyayılım Algoritması	0.04	14.28
traingdx	Adaptif Öğrenme Oranlı ve Momentum Terimli En Dik İnişli Geriyeyayılım Algoritması	0.041	14.25
traingb	Powell-Beale Başlangıç Değerli Birleşik Gradyent Geriyeyayılım	0.014	38.30
traingf	Reeves-Fletcher Başlangıç Değerli Bileşik Gradyent Geriyeyayılım	0.036	14.33
traingp	Polak-Ribiere Başlangıç Değerli Birleşik Gradyent Geriyeyayılım	0.021	26.70
trainrp	Resilient Geriyeyayılım	0.0035	43.40
trainbfg	Quasi-Newton Geriyeyayılım	0.035	14.33
trainscg	Skalalanmış Birleşik Gradyent Geriyeyayılım	0.0147	39.6

Çizelge 6.2 Trainlm algoritmasının sonuçları

	Sınıflar							Toplam	
	1	2	3	4	5	6	7	Eğitim	Test
Doğru	256	256	178	215	182	272	280	209	1639
Yanlış	44	44	122	85	118	28	20	1	461

Çizelge 6.3 Traingd algoritmasının sonuçları

	Sınıflar							Toplam	
	1	2	3	4	5	6	7	Eğitim	Test
Doğru	0	0	0	300	0	0	0	30	300
Yanlış	300	300	300	0	300	300	300	180	1800

Çizelge 6.4 Traingda algoritmasının sonuçları

	Sınıflar							Toplam	
	1	2	3	4	5	6	7	Eğitim	Test
Doğru	0	0	0	300	0	0	0	30	300
Yanlış	300	300	300	0	300	300	300	180	1800

Çizelge 6.5 Traingdx algoritmasının sonuçları

	Sınıflar							Toplam	
	1	2	3	4	5	6	7	Eğitim	Test
Doğru	0	0	0	300	0	0	0	30	300
Yanlış	300	300	300	0	300	300	300	180	1800

Çizelge 6.6 Trainrp algoritmasının sonuçları

	Sınıflar							Toplam	
	1	2	3	4	5	6	7	Eğitim	Test
Doğru	292	164	54	94	115	211	198	126	1128
Yanlış	8	136	246	206	185	89	102	84	972

Çizelge 6.7 Trainbfg algoritmasının sonuçları

	Sınıflar							Toplam	
	1	2	3	4	5	6	7	Eğitim	Test
Doğru	0	0	0	300	0	0	0	30	300
Yanlış	300	300	300	0	300	300	300	180	1800

Çizelge 6.8 Trainbr algoritmasının sonuçları

	Sınıflar							Toplam	
	1	2	3	4	5	6	7	Eğitim	Test
Doğru	275	202	101	179	154	264	227	160	1402
Yanlış	25	98	199	121	146	36	73	50	698

Çizelge 6.9 Traincgb algoritmasının sonuçları

	Sınıflar							Toplam	
	1	2	3	4	5	6	7	Eğitim	Test
Doğru	0	0	11	47	31	0	0	6	89
Yanlış	300	300	289	253	269	300	300	204	2011

Çizelge 6.10 Traincgf algoritmasının sonuçları

	Sınıflar							Toplam	
	1	2	3	4	5	6	7	Eğitim	Test
Doğru	0	0	0	300	0	0	0	30	300
Yanlış	300	300	300	0	300	300	300	180	1800

Çizelge 6.11 Traincgp algoritmasının sonuçları

	Sınıflar							Toplam	
	1	2	3	4	5	6	7	Eğitim	Test
Doğru	0	0	0	21	3	0	0	2	24
Yanlış	300	300	300	279	297	300	300	208	2076

Çizelge 6.12 Trainoss algoritmasının sonuçları

	Sınıflar							Toplam	
	1	2	3	4	5	6	7	Eğitim	Test
Doğru	0	0	0	300	0	0	0	30	300
Yanlış	300	300	300	0	300	300	300	180	1800

6.3 Radyal Temelli Fonksiyonlu Ağ Yapısı ile Görüntü Veri Kümesi İçin Simülasyon Sonuçları

RBF ile yapılan simülasyonlarda merkez vektörlerinin atanması ve yayılma parametresinin doğru belirlenmesi simülasyon sonucunu önemli bir şekilde etkilemektedir. Bu simülasyonda MATLAB 6.0 Neural Network Toolbox'taki hazır RBF ağı eğitme programı kullanıldığından atanan merkez değerleri kullanıcı tarafından sınırlanmamıştır. Sadece deneme yanılma yöntemiyle optimum yayılma parametresi belirlenmiş ve sonuçlar bu durumda verilmiştir. UCI veri kümesindeki görüntü veri seti simülasyonu için kullanılan yayılma parametresi değeri 2'dir. Çizelge 6.13'de RBF ile yapılan simülasyon sonuçları verilmektedir.

Çizelge 6.13 RBF ile sınıflandırma sonuçları

	Sınıflar							Toplam	
	1	2	3	4	5	6	7	Eğitim	Test
Doğru	31	31	34	30	298	55	30	210	509
Yanlış	269	269	266	270	2	245	270	0	1591
% Başarı	10	10	11	10	99	18	10	100	24.3

6.4 Olasılıksal Sinir Ağı Yapısı ile Görüntü Veri Kümesi İçin Simülasyon Sonuçları

RBF ile eğitme yöntemi gibi PNN ile eğitim sırasında da yayılma parametresi sonucu etkileyen parametredir. Bu simülasyonlarda da karşılaştırma yapabilmek amacıyla yayılma parametresi (spread) 2 olarak alınmıştır.

Çizelge 6.14 PNN ile sınıflandırma sonuçları

	Sınıflar							Toplam	
	1	2	3	4	5	6	7	Eğitim	Test
Doğru	273	300	205	245	241	282	295	210	1841
Yanlış	27	0	95	56	59	18	5	0	259
% Başarı	91	100	68	81	80	94	98	100	87.6

6.5 Genelleştirilmiş Regresyonlu Sinir Ağı Yapısı ile Görüntü Veri Kümesi İçin Simülasyon Sonuçları

RBF ve PNN ile eğitme yöntemi gibi GRNN ile eğitim sırasında da yayılma parametresi karşılaştırma yapabilmek amacıyla yayılma parametresi (spread) 2 olarak alınmıştır. Çizelge 6.15'de GRNN ile yapılan simülasyonlar sonucunda elde edilen sonuçlar verilmektedir.

Çizelge 6.15 GRNN ile sınıflandırma sonuçları

	Sınıflar							Toplam	
	1	2	3	4	5	6	7	Eğitim	Test
Doğru	261	300	198	247	237	282	296	210	1821
Yanlış	39	0	102	53	63	18	4	0	279
% Başarı	87	100	66	82	79	94	98	100	86.7

6.6 Konik Kesit Fonksiyonlu Sinir Ağı Yapısı ile Görüntü Veri Kümesi İçin Simülasyon Sonuçları

Konik Kesit Fonksiyonlu Yapay Sinir Ağında esas fikir, bir Çok Katmanlı Algılayıcı birimi ile bir Radyal Temelli Fonksiyonlu ağ birimi arasında ilişki sağlayarak bütün bu karar bilgilerini içine alacak bir birimin fonksiyonunu genelleştirmektir. Bu ağ yapısında merkezler, ağırlık matrisi, açı ve bias terimleri güncellenerek optimum sonuç elde edilmektedir. İterasyon sayısı arttıkça hata daha da azalmakta ancak belli bir iterasyondan sonra hata oranı sabit kalarak ağ öğrenmekten öte ezberleme yönüne kaymaktadır. Bu yüzden yine en uygun iterasyon sayısı deneme yanılma yolu ile bulunmuştur.

Çizelge 6.16 CSFNN ile sınıflandırma sonuçları

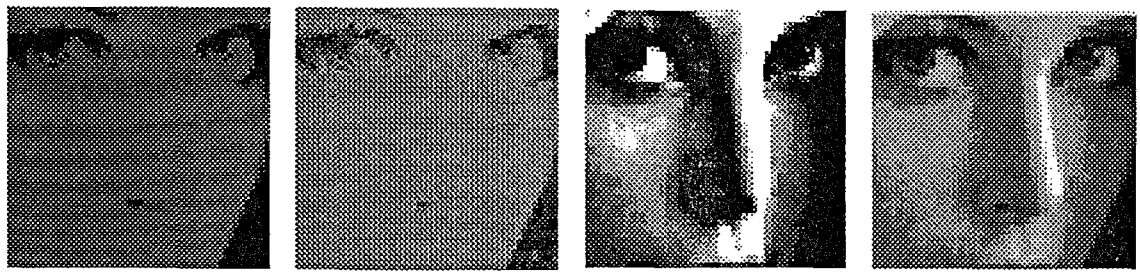
	Sınıflar							
	1	2	3	4	5	6	7	Eğitim
Doğru	30	30	29	30	30	30	30	209
Yanlış	0	0	1	0	0	0	0	1
% Başarı	100	100	96.6	100	100	100	100	99.7

6.7 Lenna ve Peppers Görüntüleri İçin Konik Kesit Fonksiyonlu Sinir Ağı Simülasyon Sonuçları

Bu uygulamada Lenna ve Peppers Görüntüleri gri skalalı olarak incelenmiştir. Bunun için öncelikle görüntüler MATLAB 6.0'da Image Processing Toolbox kullanılarak piksellerine ayrılmış ve her piksel için 0.0 ile 1.0 arasında bir değer elde edilmiştir. Burada 0.01 siyah rengi 0.99 ise beyaz rengi göstermektedir. Buna göre de 0 ile 1 arasındaki değerler, siyah ile beyaz arasında gri skalalı olarak oluşturulmuştur. Şekil 6.1'de Lenna Görüntüsü'nün orjinal hali görülmektedir.

Şekil 6.1 Lenna görüntüsü

Piksellerine ayrılan görüntünün sadece yüz kısmı hedef alınarak konik kesit fonksiyonlu ağ eğitilmiş ve iterasyon sayısına bağlı olarak elde edilen sonuçlar şekil 6.2'de verilmiştir.



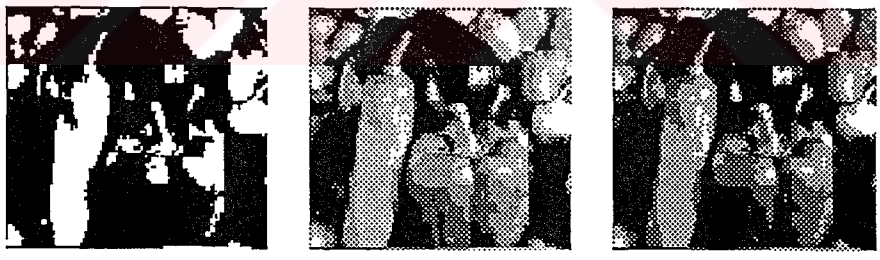
Şekil 6.2 Lenna görüntüsüne ait konik kesit fonksiyonlu sinir ağı çıktıları iterasyon sayısı sırasıyla 50,500,1000 ve 1500.

Şekil 6.2'den de görüleceği üzere ağ 50 iterasyonda belli zıtlıklara (kontrast) sahip pikselleri rahatça sınıflandırabilmiştir. Ancak piksellerin değerleri birbirine yaklaştığı yerlerde

(kontrastın azaldığı yerler) ağın çıkışı doğru yanıt vermemektedir. Ne var ki, iterasyon sayısının artmasıyla gerek kontrastın fazla olduğu gerekse az olduğu yerlerde dahi ağ tamamen doğru olmasa bile doğruya yakın bir çıkışa ulaşabilmiştir.

Şekil 6.3'de biber resimlerinin olduğu gri skalalı "Peppers" görüntüsü görülmektedir. Lenna görüntüsü için yapılan her işlem bu görüntü için de yapılmış ve elde edilen sonuçlar şekil 6.4'de verilmiştir.

Şekil 6.3 Peppers görüntüsü



Şekil 6.4 Peppers görüntüsüne ait sonuçlar, sırasıyla iterasyon sayısı 50,100,150

Şekil 6.4'deki sonuçlara bakıldığında konik kesit fonksiyonlu ağ yapısının Peppers görüntüsü için Lenna görüntüsü kadar iyi sonuç vermediği gözlemlenmiştir. Bunun nedeni, Peppers görüntüsünün orjinal halinin 256x256 çözünürlüğe (true color) sahip olması ve gri skalalı olarak kaydedilirken bazı bilgilerin yanlış veya eksik olarak kaydedilmesi olarak düşünülmüştür.

7. SONUÇLAR

Bu tezde görme sisteminin elektronik tasarımına en uygun yapay sinir ağı yapısı bulunması amaçlanmıştır. Görüntünün algılanması ve daha sonra piksellere ayrılmasıyla elde edilen bilgi kümesi, yapay sinir ağları ile eğitilerek örnek bilgi kümesinde olmayan görüntüler için de ağın tanıma yeteneğinin arttırılması için çalışılmıştır. Kullanılan sinir ağı modelleri arasında çok katmanlı algılayıcılar (MLP), radyal temelli fonksiyonlu ağlar (RBF), istatistiksel sinir ağları (PNN), genelleştirilmiş regresyonlu ağlar (GRNN) ve son olarak da 1994 yılında Dorffner tarafından ortaya atılan konik kesit fonksiyonlu sinir ağları (CSFNN) bulunmaktadır.

Yapılan simülasyonlarda olasılıksal sinir ağları ile konik kesit fonksiyonlu sinir ağı yapılarının görüntü veri kümesi üzerinde oldukça iyi sonuçlar verdiği gözlemlenmiştir. Bu yüzden de görme sisteminin elektronik tasarımına en uygun yapay sinir ağı olarak bu iki ağ yapısı seçilmiştir. Konik kesit fonksiyonlu sinir ağı simülasyonlarında iterasyonlar oldukça zaman aldığından daha çok sayıda adımda simülasyon yapılamamıştır. İleriki çalışmalarda iterasyon sayısı arttırılarak görüntülerin daha da iyi algılanması sağlanabilir.

7.1 Gelecek Çalışma

Bu çalışmadan sonra, ileriki aşamalarda, simülasyonları yapılan ağ yapılarının elektronik olarak tasarlanması ve uygun ortamlarda denenmesi hedef olarak alınabiir. Görüntünün sınıflandırılmasına dayalı yapay retina yapısının geliştirilmesi bu çalışmaların en verimli sonucu olacaktır.

7.2 Konu ile İlgili Yapılan Yayınlar

Görüntü sınıflandırmasının yapay sinir ağları ile denenmesi üzerine yapılan yayınlar aşağıdaki listede verilmiştir.

Coskun, N., Yıldırım, T., (2003), "İmge Bölütlemede Yapay Sinir Ağlarının Kullanımı", SIU 2003 Sinyal İşlemeleri ve Uygulamaları Konferansı 18-20 Haziran 2003, Koç Üniversitesi, İstanbul, Türkiye

Coskun, N., Yıldırım, T., (2003), "The Effects Of Training Algorithms In MLP Network On Image Classification", International Joint Conference On Neural Networks 20-24 Temmuz 2003, Portland, USA

Coskun, N., Yıldırım, T., (2003), "Image Segmentation Usıng Statistical Neural Networks", 13th International Conference On Artificial Neural Networks and 10th International Conferance On Neural Information Processing 26-29 Haziran 2003, Istanbul, Türkiye

KAYNAKLAR

Avci, M., Yıldırım T., (2002), "Classification of Escherichia Coli Bacteria by Artificial Neural Networks", IEEE International Symposium on Intelligent Systems, Vol III, Varna, Bulgaristan, sf. 16-20.

Başaran, A., (1995), Tıbbi Biyoloji, Bilim Teknik Yayınevi, İstanbul

Bertram, E., ve Choi, T., (2001), "A Michelson Contrast Sensitive Silicon Retina", Iconip 2001 Proceedings

Crick, F., (2000), Şaşırtan Varsayım, Tübitak Yayınları, Ankara

Elmas, Ç., (2003), Yapay Sinir Ağları (Kuram, Mimari, Eğitim, Uygulama), Seçkin Yayınevi, Ankara

Harvey, R.L., (1994), Neural Network Principles, Prentice-Hall, Inc., USA

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

Hecht-Nielsen,R., (1991), Neurocompiting, Addison-Wesley Publishing, USA

Hinton, G.E., (1989), "Connectionist Learning Procedures", Artificial Intelligence 40:185-234.

Hirahara, M., ve Oka, N., (1993), "A Hybrid Model Composed of A Multilayer Perceptron and A Radial Basis Function Network", International Joint Conference on Neural Networks, IJCNN'93, Nagoya, Japonya

Lippman, R.P., (1987), "An Introduction to Computing with Neural Nets",IEEE ASSP Magazine, 4-22.

Maruyama M., Girosi, F., ve Poggio, T., (1992), A Connection Between GRBF and MLP, A.I. Memo No.1291, Massachusets Institute of Technology, Boston.

Mead, C., (1989), Analog VLSI and Neural Systems, Addison-Wesley Publishing, Canada

Özyılmaz, L., (2000), Konik Kesit Fonksiyonlu Yapay Sinir Ağında Öğrenme Algoritmasının Geliştirilmesi ve Ağın Çeşitli Problemler İçin Performansı ile Duyarlığının İncelenmesi, Doktora Tezi, Yıldız Teknik Üniversitesi, Türkiye

Specht, D.F., (1991), "A General Regression Neural Network", IEEE Transactions on Neural Networks, 2, 568-576.

Specht, D.F., "Probabilistic Neural Networks and the Polynomial Adaline as Complementary Techniques for Classification", IEEE Transactions on Neural Networks, 1, 1990, 111-121.

Voon, L.Y., Cathebras, G., Lamelle B., Gorria P., Bellach B., Aubreton O., (2002), "100x100 Pixels CMOS Retina For Real Time Binary Pattern Matching", Optical Engineering Letters, Vol.41, No.5, pp.924-925.

Yıldırım, T., (1997), Development of Conic Section Function Neural Networks in Software and Analogue Hardware, Doktora Tezi, University of Liverpool, UK

Yıldırım, T., ve Marsland, J.S., (1997), "A Unified Framework for Connectionist Models", Perspectives in Neural Computing: Connectionist Representations, ed. J.A. Bullinaria, D.W.

Glasspool, and G. Houghton, 26-39, Springer-Verlag, London, UK

Yıldırım,T., (2003), Yapay Sinir Sistemleri Tasarımı ders notları, İstanbul

Zurada, J.M., (1995), Introduction to Artificial Neural Systems, PWS Publishing, Boston

Internet Kaynakları

[1] http:// www.yapay-zeka.org

[2] http://ftp.ics.uci.edu/pub/ml-repos/machine-learning-databases/

EKLER

EK 1 KONİK KESİT FONKSİYONLU SİNİR AĞI İÇİN SİMULASYON PROGRAMI

%Bu program konik kesit fonksiyonlu sinir ağları için simulasyon amacıyla yazılmıştır. %Programda önce ağ RBF ağı gibi başlatılmış ve OLS (ortogonal en küçük kareler- %orthogonal least squares) yöntemi ile ağın ilk merkez değerleri atanmıştır. Daha sonra ise %atanan merkez değeri sayısının en uygun değerine ulaşılıncaya kadar merkez vektörleri %oluşturulmuştur. Bundan sonra ise ağ geriye yayılımlı algoritma ile MLP gibi eğitilmeye %başlanmış ve güncellenecek parametreler merkez değerleri, ağırlıklar, açı ve bias terimleri %olarak belirlenmiştir.

%Programda;
%p: giriş vektörü
%t: hedef vektörü
%c: merkez vektörü
%ww: açı değeri
%w1: giriş katmanı ile saklı katman arasındaki ağırlık vektörü
%w2: gizli katman ile çıkış katmanı arasındaki ağırlık vektörü
%spread: yayılma parametresi
%a1: gizli katman çıkışı
%a2: son çıkış

% RADYAL TABANLI OLARAK BAŞLATILAN AĞ İÇİN OLS İLE MERKEZ %DEĞERLERİNİN BULUNARAK ÇIKIŞIN VE ÇIKIŞTAKİ HATANIN HESABI

```
d=t';
[r,q] = size(p);
[s2,q] = size(t);
b = sqrt(-log(.5))/spread;
x=dist(p',p);
P = logsig(x*b);
PP = sum(P.*P)';
dd = sum(d.*d)';
e=((P' * d)' .^ 2)./(dd* PP');
```

% En çok hataya sahip vektör hesaplanıyor

```
pick = findLargeColumn2(e);
used = [];
left = 1:q;
C = P(:,pick);
PP(pick,:) = [];
e(:,pick) = [];
used = [used left(pick)];
left(pick) = [];
c = p(:,used)';
[crr,css]=size(c);
ww=pi/4*ones(1,crr);
w1=ones(crr,r);
```

```
%-------------------------------
c = p(:,used)';
[crr,css]=size(c);
ww=pi/4*ones(1,crr);
w1=ones(crr,r);
a1=logsig(consec(c,p,w1,ww)*b)
[w2,b2]=solvelina(a1,t);
a2=purelin(w2*a1,b2)
sse=sumsqr(t-a2)

%OLS ILE MERKEZ VEKTORLERININ HESAPLANMASI

for k=1:s1-1
   if (sse<eg)
      break
   end
cj=C(:,k);
a=cj'*P/(cj'*cj);
for i=1:q
   P=(P(:,i)-cj)*a;
end
PP=sum(P.*P)';
e=((P' * d)' .^ 2)./(dd* PP');
pick = findLargeColumn2(e);
C = [C,P(:,pick)];
PP(pick,:) = [];
e(:,pick) = [];
used = [used left(pick)];
left(pick) = [];

%HATA HESABI
c=p(:,used)';
[cr,cc]=size(c);
ww=pi/4*ones(1,cr);
w1=zeros(cr,r);
a1=logsig(consec(c,p,w1,ww)*b);
[w2,b2]=solvelina(a1,t);
a2=purelin(w2*a1,b2);
sse=sumsqr(t-a2)
end

% GERİYE YAYILMALI ALGORİTMA İLE AĞ EĞİTİLİYOR

% Egitim baslangic
[crr,css]=size(c);
ww=pi/4*ones(1,crr);
dw10=0;
dw20=0;
db10=0;
db20=0;
```

```
dom0=0;
[a1,a2]=simcsfnn(c,p,w1,ww,b,b1,w2,b2)
e=t-a2;
SSE=sumsqr(e)

%---------------------------------
for k=1:me
   %HATA KONTROLU
   if SSE<eg
      break
   end
   %GERIYE YAYILMA FAZI
   d2=feval('deltalog',a2,e);
   d1=feval('deltalog',a1,d2,w2);
   %OGRENME FAZI
   for i=1:9
      for m=1:crr
         cn=c(m,:)'*ones(1,q);
         pnew=(cn-p);
      end
      dw1=learnbp(pnew,d1,lr);
      dw2=learnbp(a1,d2,lr);
      dw1=dw1+mom*dw10;
      dw2=dw2+mom*dw20;
      db1=db1+mom*db10;
      db2=db2+mom*db20;
      tw1=w1+dw1;
      tb1=b1+db1;
      tw2=w2+dw2;
      tb2=b2+db2;
      dw10=dw1;
      dw20=dw2;
      db10=db1;
      db20=db2;
      [ta1,ta2]=simcsfnn(c,p,tw1,ww,b,tb1,tw2,tb2);
      te=t-ta2;
      TSSE=sumsqr(te)
      SSE
      %  ADAPTIF OGRENME
      if TSSE>SSE*err_ratio
         lr=lr*lr_dec;
      elseif TSSE<=SSE
         lr=lr*lr_inc;
        w1=tw1;
       b1=tb1;
       w2=tw2;
       b2=tb2;
       a1=ta1;
       a2=ta2;
       e=te;
       SSE=TSSE;
```

```
    else
    w1=tw1;
    b1=tb1;
    w2=tw2;
    b2=tb2;
    a1=ta1;
    a2=ta2;
    e=te;
    SSE=TSSE;
  end
  i=i+1
   end
end
   %ACININ GUNCELLESTIRILMESI
   d2=feval('deltalog',a2,e);
   d1=feval('deltalog',a1,d2,w2);
   om=ww;
   difom=sin(om)*dist(c,p);
   dom=(learnbp(difom,d1,lr))';
   dom=dom+mom1*dom0;
   om=om+dom;
   dom0=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
   ww=om
   [a1,a2]=simcsfnn(c,p,tw1,ww,b,b1,tw2,b2);
   e=t-a2;
   SSE=sumsqr(e)
```

% ALT PROGRAMLAR

% Bir Matristeki En Büyük Vektörü Hesaplama

```
function i = findLargeColumn2(m)
replace = find(isnan(m));
m(replace) = zeros(size(replace));
m = sum(m .^ 2,1);
i = find(m == max(m));
i = i(1);
```

```
%Konik Kesit Fonksiyonlu Ağın Çıkışı
function y=consec(c,p,w1,ww)
[s,r]=size(c);
[r2,q]=size(p);
```

```
y=zeros(s,q);
a=cos(ww);
b=dist(c,p);
for i=1:q
   x(i,:)=a.*b(:,i)';
end
y=dotp(c,p,w1)-x';

%---------------------------------
function [w,b] = solvelina(p,t)

if nargout <= 1
  w= t/p;
else
  [pr,pc] = size(p);
  x = t/[p; ones(1,pc)];
  w = x(:,1:pr);
  b = x(:,pr+1);
end

% ------------------------------

function [a1,a2]=simcsfnn(c,p,w1,ww,b,b1,w2,b2)
b=dist(c,p);
%b = sqrt(-log(.5))/spread;
[s,r]=size(c);
[r2,q]=size(p);
f=consec(c,p,w1,ww).*b;
[x,y]=size(f);
b1=1;
a1=logsig(f,b1*ones(x,y));
a2=logsig(w2*a1)
end

%-----------------------------

function d=deltalog(a,d,w)
if nargin==1
   d=a.*(1-a);
elseif nargin ==2
   d=a.*(1-a).*d;
else
   d=a.*(1-a).*(w'*d);
end

%---------------------------

function [dw,db]=learnbp(p,d,lr)
if nargin<2,error('not enough arguments');end
if nargin==3,d=lr*d;end
dw=d*p';
```

```
if nargout==2
   [R,Q]=size(p);
   db=d*ones(Q,1);
end

%--------------------------

% Eğitim Sınıflama Oranının Hesaplanması
say=0;
for i = 1:30
  if a2(:,i)<=0.4
     say=say+1;
    i=i+1;
  else
     i=i+1;
     say=say;
   end
end

for i = 31:60
  if a2(1,i)<=0.4 & a2(2,i)<=0.4 & a2(3,i)>=0.6
     say=say+1;
    i=i+1;
  else
     i=i+1;
     say=say;
   end
end

for i = 61:90
  if a2(1,i)<=0.4 & a2(2,i)>=0.6 & a2(3,i)<=0.4
     say=say+1;
    i=i+1;
  else
     i=i+1;
     say=say;
   end
end

for i = 91:120
  if a2(1,i)<=0.4 & a2(2,i)>=0.6 & a2(3,i)>=0.6
     say=say+1;
    i=i+1;
  else
     i=i+1;
     say=say;
   end
end

for i = 121:150
  if a2(1,i)>=0.6 & a2(2,i)<=0.4 & a2(3,i)<=0.4
```

```
      say=say+1;
     i=i+1;
  else
      i=i+1;
      say=say;
   end
end

for i = 151:180
  if a2(1,i)>=0.6 & a2(2,i)<=0.4 & a2(3,i)>=0.6
      say=say+1;
     i=i+1;
  else
      i=i+1;
      say=say;
   end
end

for i = 181:210
  if a2(1,i)>=0.6 & a2(2,i)>=0.6 & a2(3,i)<=0.4
      say=say+1;
     i=i+1;
  else
      i=i+1;
      say=say;
   end
end

dogru_siniflama_orani = (say/210)*100
```

EK2 UCI MAKİNA ÖĞRENMESİ VERİ KÜMESİNDEKİ GÖRÜNTÜ VERİ SETİ

REGION-CENTROID-COL,REGION-CENTROID-ROW,REGION-PIXEL-COUNT,SHORT-LINE-DENSITY-5,SHORT-LINE-DENSITY-2,VEDGE-MEAN,VEDGE-SD,HEDGE-MEAN,HEDGE-SD,INTENSITY-MEAN,RAWRED-MEAN,RAWBLUE-MEAN,RAWGREEN-MEAN,EXRED-MEAN,EXBLUE-MEAN,EXGREEN-MEAN,VALUE-MEAN,SATURATION-MEAN,HUE-MEAN

BRICKFACE,140.0,125.0,9,0.0,0.0,0.2777779,0.06296301,0.66666675,0.31111118,6.185185,7.3333335,7.6666665,3.5555556,3.4444444,4.4444447,-7.888889,7.7777777,0.5456349,-1.1218182

BRICKFACE,188.0,133.0,9,0.0,0.0,0.33333334,0.26666674,0.5,0.077777736,6.6666665,8.333334,7.7777777,3.8888888,5.0,3.3333333,-8.333333,8.444445,0.53858024,-0.92481726

BRICKFACE,105.0,139.0,9,0.0,0.0,0.27777782,0.107407436,0.83333325,0.52222216,6.111111,7.5555553,7.2222223,3.5555556,4.3333335,3.3333333,-7.6666665,7.5555553,0.5326279,-0.96594584

BRICKFACE,34.0,137.0,9,0.0,0.0,0.5000002,0.16666673,1.111111,0.47407418,5.851852,7.7777777,6.4444447,3.3333333,5.7777777,1.7777778,-7.5555553,7.7777777,0.57363313,-0.74427164

BRICKFACE,39.0,111.0,9,0.0,0.0,0.72222227,0.37407416,0.8888889,0.4296295,6.037037,7.0,7.6666665,3.4444444,2.8888888,4.888889,-7.7777777,7.888889,0.56291884,-1.1757725

BRICKFACE,16.0,128.0,9,0.0,0.0,0.5,0.077777766,0.66666675,0.31111118,5.5555553,6.888889,6.6666665,3.1111112,4.0,3.3333333,-7.3333335,7.111111,0.56150794,-0.98581076

BRICKFACE,26.0,67.0,9,0.11111111,0.0,1.0,0.8888903,2.4444447,3.1851847,20.0,19.555555,25.88889,14.555555,-1.3333334,17.666666,-16.333334,25.88889,0.4369392,-1.6232022

BRICKFACE,14.0,110.0,9,0.0,0.0,1.7222224,5.3518505,2.6666667,1.0222229,17.925926,18.88889,21.444445,13.444445,2.8888888,10.555555,-13.444445,21.444445,0.36884832,-1.3450956

BRICKFACE,11.0,108.0,9,0.0,0.0,1.3333335,0.80000025,1.3888888,0.95185167,17.666666,19.0,21.11111,12.888889,4.0,10.333333,-14.333333,21.11111,0.38875645,-1.3021333

BRICKFACE,85.0,101.0,9,0.0,0.0,1.3333334,1.2888881,1.2777777,1.2185181,21.296297,21.222221,26.777779,15.888889,-0.22222222,16.444445,-16.222221,26.777779,0.4047922,-1.5585992

BRICKFACE,18.0,145.0,9,0.0,0.0,0.38888896,0.018518496,0.6111111,0.3740741,3.925926,5.5555553,4.0,2.2222223,4.888889,0.22222222,-5.111111,5.5555553,0.6005291,-0.57094

BRICKFACE,23.0,55.0,9,0.0,0.0,2.2222216,3.6740727,1.7777773,0.7851852,23.444445,21.666666,31.11111,17.555555,-5.3333335,23.0,-17.666666,31.11111,0.43507022,-1.7711632

BRICKFACE,196.0,129.0,9,0.0,0.0,0.83333325,0.43333334,0.6666667,0.17777774,6.3333335,7.888889,7.3333335,3.7777777,4.6666665,3.0,-7.6666665,8.222222,0.54012346,-0.9327826

BRICKFACE,80.0,116.0,9,0.0,0.0,1.5,1.6333328,1.5555557,0.87407404,21.703703,21.222221,27.555555,16.333334,-1.4444444,17.555555,-16.11111,27.555555,0.40736422,-1.6322426

BRICKFACE,2.0,44.0,9,0.0,0.0,2.1666672,2.3888876,2.3888896,1.5296285,18.74074,17.333334,25.222221,13.666667,-4.2222223,19.444445,-15.222222,25.222221,0.45768142,-1.7537255

BRICKFACE,120.0,136.0,9,0.0,0.0,0.61111116,0.4185187,1.0000001,0.4444444,6.259259,7.7777777,7.2222223,3.7777777,4.5555553,2.8888888,-7.4444447,8.0,0.52954143,-0.92460656

BRICKFACE,146.0,124.0,9,0.0,0.0,0.4999999,0.16666664,0.38888875,0.107407376,6.037037,7.4444447,7.3333335,3.3333333,4.2222223,3.8888888,-8.111111,7.6666665,0.56349206,-1.0247301

BRICKFACE,23.0,85.0,9,0.0,0.0,1.4444445,1.0518525,1.777778,0.96296287,17.962963,18.88889,21.88889,13.111111,2.7777777,11.777778,-14.555555,21.88889,0.39975148,-1.386867

BRICKFACE,138.0,116.0,9,0.0,0.0,0.6111111,0.15185188,0.4444445,0.20740739,6.4814816,7.5555553,8.222222,3.6666667,3.2222223,5.2222223,-8.444445,8.333334,0.5591711,-1.1910701

BRICKFACE,229.0,124.0,9,0.0,0.0,0.888889,0.074073985,0.8888889,0.3407407,5.888889,7.111111,7.111111,3.4444444,3.6666667,3.6666667,-7.3333335,7.5555553,0.5458554,-1.0376626

BRICKFACE,22.0,116.0,9,0.0,0.0,0.38888875,0.10740741,0.33333325,0.13333334,5.6296296,6.7777777,7.0,3.1111112,3.4444444,4.111111,-7.5555553,7.3333335,0.57539684,-1.0996237

BRICKFACE,121.0,60.0,9,0.0,0.0,2.277778,2.329629,2.888889,2.8740742,26.74074,24.666666,35.22222,20.333334,-6.2222223,25.444445,-19.222221,35.22222,0.4223002,-1.776113

BRICKFACE,33.0,149.0,9,0.0,0.0,0.5555556,0.25185165,0.72222227,0.15185171,5.4444447,4.111111,8.666667,3.5555556,-4.0,9.666667,-5.6666665,8.666667,0.5783389,-1.9857028

BRICKFACE,80.0,95.0,9,0.0,0.0,1.2222223,1.0074079,0.944444,0.5518514,21.407408,21.333334,26.666666,16.222223,-0.22222222,15.777778,-15.555555,26.666666,0.39043593,-1.5673273

BRICKFACE,96.0,84.0,9,0.0,0.0,1.5000004,1.2777773,1.6111107,2.2851882,23.851852,23.555555,30.0,18.0,-0.8888889,18.444445,-17.555555,30.0,0.39879107,-1.598867

BRICKFACE,145.0,102.0,9,0.0,0.0,0.88888866,0.6074083,2.611111,1.4851857,23.074074,22.11111,29.777779,17.333334,-2.8888888,20.11111,-17.222221,29.777779,0.41776258,-1.6854404

BRICKFACE,18.0,138.0,9,0.0,0.0,0.88888884,0.5629629,0.83333325,0.29999986,5.740741,7.3333335,6.5555553,3.3333333,4.7777777,2.4444444,-7.2222223,7.3333335,0.5438712,-0.8621077

BRICKFACE,138.0,133.0,9,0.0,0.0,0.6666667,0.44444433,1.1666666,0.21111093,6.4444447,7.7777777,7.888889,3.6666667,4.0,4.3333335,-8.333333,8.222222,0.5582011,-1.0776595

BRICKFACE,121.0,113.0,9,0.0,0.0,1.722222,1.5296303,2.944444,1.5296295,20.25926,20.0,25.444445,15.333333,-0.7777778,15.555555,-14.777778,25.444445,0.39658895,-1.5856091

BRICKFACE,95.0,57.0,9,0.0,0.0,1.8333327,3.4111106,2.1111107,1.7185175,26.296297,24.666666,34.444447,19.777779,-4.888889,24.444445,-19.555555,34.444447,0.42569095,-1.7390174

SKY,140.0,25.0,9,0.0,0.0,0.99999875,1.4666697,1.1111107,0.118517555,128.0,117.77778,142.33334,123.888885,-30.666666,43.0,-12.333333,142.33334,0.17250364,-2.3545518

SKY,142.0,33.0,9,0.0,0.0,0.49999872,0.62360865,0.50000125,0.3496027,110.59259,96.77778,128.66667,106.333336,-41.444443,54.22222,-12.777778,128.66667,0.24777646,-2.4080186

SKY,66.0,41.0,9,0.0,0.0,0.61111194,0.32773077,0.38889185,0.32773316,109.703705,95.111115,128.88889,105.111115,-43.77778,57.555557,-13.777778,128.88889,0.26204562,-2.404745

SKY,165.0,99.0,9,0.0,0.0,0.88889056,0.47407392,0.7777786,0.47407353,93.40741,79.0,118.0,83.22222,-43.22222,73.77778,-30.555555,118.0,0.33041757,-2.2077565

SKY,228.0,20.0,9,0.0,0.0,1.0555547,0.49065518,0.8333333,0.7527733,125.0,114.0,140.55556,120.44444,-33.0,46.666668,-13.666667,140.55556,0.18889166,-2.348006

SKY,124.0,29.0,9,0.0,0.0,1.0000013,0.7111076,1.0555534,0.90741664,128.44444,119.22222,142.88889,123.22222,-27.666666,43.333332,-15.666667,142.88889,0.16561614,-2.271128

SKY,156.0,32.0,9,0.0,0.0,0.77777356,0.16296418,2.6111095,1.0407379,136.2963,129.77779,146.33334,132.77779,-19.555555,30.11111,-10.555555,146.33334,0.113055356,-2.280755

SKY,21.0,90.0,9,0.0,0.0,0.66666794,0.044444147,0.7777786,0.56296283,113.48148,105.888885,128.55556,106.0,-22.777779,45.22222,-22.444445,128.55556,0.17969723,-2.0978148

SKY,8.0,39.0,9,0.11111111,0.0,1.3888906,1.129629,1.8333334,0.699999,113.37037,102.55556,132.0,105.55556,-32.444443,55.88889,-23.444445,132.0,0.22295359,-2.1981578

SKY,122.0,11.0,9,0.0,0.0,1.0,0.31111577,2.8888905,5.051852,143.44444,136.88889,150.88889,142.55556,-19.666666,22.333334,-2.6666667,150.88889,0.09277301,-2.5216477

SKY,44.0,79.0,9,0.0,0.0,0.44444403,0.34426486,0.7777786,0.4036864,107.74074,93.888885,126.55556,102.77778,-41.555557,56.444443,-14.888889,126.55556,0.2580791,-2.3779652

SKY,7.0,18.0,9,0.0,0.0,1.2777786,0.7296265,0.9444453,0.37407914,138.62962,133.33334,147.55556,135.0,-15.888889,26.777779,-10.888889,147.55556,0.096352234,-2.2146115

SKY,188.0,42.0,9,0.0,0.0,0.7777786,0.5443299,1.6666679,1.2649081,108.92593,95.666664,126.22222,104.888885,-39.77778,51.88889,-12.111111,126.22222,0.24193405,-2.4103878

SKY,152.0,18.0,9,0.0,0.0,0.7777774,0.4554219,0.55555725,0.2721644,112.111115,97.22222,130.44444,108.666664,-44.666668,55.0,-10.333333,130.44444,0.2546684,-2.45498

SKY,120.0,74.0,9,0.0,0.0,0.3333346,0.08888922,0.50000125,0.07777796,101.85185,89.111115,123.22222,93.22222,-38.22222,64.111115,-25.88889,123.22222,0.2767844,-2.2205532

SKY,143.0,24.0,9,0.0,0.0,1.2777773,0.9074056,0.88888806,1.1407489,127.62963,117.666664,141.66667,123.55556,-29.88889,42.11111,-12.222222,141.66667,0.16939692,-2.349252

SKY,181.0,27.0,9,0.0,0.0,0.7222214,0.46296388,0.5,0.25555483,138.07408,132.55556,146.55556,135.11111,-16.555555,25.444445,-8.888889,146.55556,0.09626316,-2.2645319

SKY,107.0,21.0,9,0.0,0.0,0.66666156,0.51639783,1.1666666,0.4082483,126.77778,115.77778,141.88889,122.666664,-33.0,45.333332,-12.333333,141.88889,0.18402189,-2.3703718

SKY,226.0,83.0,9,0.0,0.0,0.8888893,0.5185186,1.0555521,0.50740635,90.62963,74.55556,116.888885,80.44444,-48.22222,78.77778,-30.555555,116.888885,0.36206177,-2.2390528

SKY,93.0,29.0,9,0.0,0.0,1.2222239,1.2296363,1.3888906,1.5740819,128.48148,119.0,142.77777,123.666664,-28.444445,42.88889,-14.444445,142.77777,0.16648434,-2.2977605

SKY,60.0,52.0,9,0.0,0.0,0.7222226,0.5963011,0.7777774,0.7407436,111.62963,101.0,129.22223,104.666664,-31.88889,52.77778,-20.88889,129.22223,0.21837935,-2.2296848

SKY,179.0,101.0,9,0.0,0.0,0.44444785,0.3851871,0.61110944,0.32963282,134.92592,126.44444,147.22223,131.11111,-25.444445,36.88889,-11.444445,147.22223,0.14110672,-2.3287346

SKY,112.0,30.0,9,0.0,0.0,0.55555725,0.2721644,1.2222227,0.7200825,113.25926,100.77778,130.11111,108.888885,-37.444443,50.555557,-13.111111,130.11111,0.22538376,-2.3853076

SKY,103.0,64.0,9,0.0,0.0,0.6666667,0.6992054,1.3333308,0.91893595,108.77778,96.333336,126.22222,103.77778,-37.333332,52.333332,-15.0,126.22222,0.2367968,-2.3554425

SKY,174.0,50.0,9,0.0,0.0,1.0000013,0.7601153,0.9444453,0.9525792,107.44444,94.666664,125.77778,101.888885,-38.333332,55.0,-16.666666,125.77778,0.2473368,-2.3372955

SKY,80.0,40.0,9,0.0,0.0,0.6111107,0.5741323,0.7222226,0.7722025,110.703705,96.22222,129.0,106.888885,-43.444443,54.88889,-11.444445,129.0,0.2540513,-2.4346924

SKY,67.0,71.0,9,0.0,0.0,1.6666666,0.8888911,1.4999987,0.29999846,125.96296,115.55556,140.88889,121.44444,-31.222221,44.77778,-13.555555,140.88889,0.17967218,-2.339758

SKY,92.0,56.0,9,0.0,0.0,0.44444275,0.029629406,0.8333333,0.5666677,126.0,115.888885,140.66667,121.44444,-30.333334,44.0,-13.666667,140.66667,0.17606053,-2.3267827

SKY,67.0,32.0,9,0.0,0.0,0.944444,1.0628421,1.7777786,1.3109215,126.22222,115.111115,142.22223,121.333336,-33.333332,48.0,-14.666667,142.22223,0.19062504,-2.333746

SKY,125.0,46.0,9,0.11111111,0.0,0.61110944,0.61161584,2.166668,0.7817352,124.55556,112.77778,141.0,119.888885,-35.333332,49.333332,-14.0,141.0,0.20012376,-2.357688

FOLIAGE,101.0,121.0,9,0.11111111,0.0,0.6666667,0.843274,1.5,0.88819396,3.4074075,1.1111112,6.0,3.1111112,-6.888889,7.7777777,-0.8888889,6.0,0.8435185,-2.5191648

FOLIAGE,21.0,122.0,9,0.0,0.0,0.44444445,0.4036867,0.44444445,0.4036867,0.5555556,0.0,1.2222222,0.4444
4445,-1.6666666,2.0,-0.33333334,1.2222222,0.5555556,-2.4445627

FOLIAGE,45.0,89.0,9,0.0,0.0,0.7777777,0.3407409,0.77777773,0.47407398,2.2962964,0.11111111,6.4444447,
0.33333334,-6.5555553,12.444445,-5.888889,6.4444447,0.9861111,-2.1239047

FOLIAGE,18.0,87.0,9,0.0,0.0,1.5555555,1.9626132,1.8888887,1.7469552,2.925926,1.2222222,5.6666665,1.88
88888,-5.111111,8.222222,-3.1111112,5.6666665,0.86277056,-2.2294934

FOLIAGE,52.0,102.0,9,0.0,0.0,0.72222227,0.50740725,0.8333333,0.5666668,2.8888888,0.6666667,6.3333335,
1.6666666,-6.6666665,10.333333,-3.6666667,6.3333335,0.89973545,-2.2617195

FOLIAGE,54.0,91.0,9,0.0,0.0,1.4444443,1.540741,0.8333333,0.25555572,3.2592592,0.5555556,8.0,1.2222222,
-8.111111,14.222222,-6.111111,8.0,0.94481075,-2.186723

FOLIAGE,80.0,87.0,9,0.0,0.11111111,24.388891,572.9964,44.722225,1386.3292,67.44444,58.77778,79.0,64.5
5556,-26.0,34.666668,-8.666667,79.0,0.30628127,-2.4221272

FOLIAGE,140.0,124.0,9,0.0,0.0,1.0,0.44444454,1.1111112,1.0518516,2.5185184,0.22222222,6.111111,1.2222
222,-6.888889,10.777778,-3.8888888,6.111111,0.97376543,-2.267462

FOLIAGE,69.0,139.0,9,0.0,0.0,2.8333328,1.7732588,2.1111112,1.6688871,18.074074,16.0,22.555555,15.6666
67,-6.2222223,13.444445,-7.2222223,23.88889,0.38661182,-1.7028334

FOLIAGE,9.0,80.0,9,0.0,0.0,2.9444444,13.751853,16.666666,71.5111,23.62963,17.333334,31.666666,21.8888
9,-18.88889,24.11111,-5.2222223,31.666666,0.5142537,-2.4315135

FOLIAGE,6.0,81.0,9,0.0,0.11111111,4.111111,8.740745,5.722223,28.50741,12.481482,7.6666665,18.88889,10
.888889,-14.444445,19.222221,-4.7777777,18.88889,0.6281558,-2.388561

FOLIAGE,74.0,129.0,9,0.0,0.0,0.22222222,0.029629637,0.11111111,0.029629635,0.5185185,0.0,1.5555556,0.
0,-1.5555556,3.1111112,-1.5555556,1.5555556,1.0,-2.0943952

FOLIAGE,41.0,75.0,9,0.0,0.11111111,15.388889,19.136257,26.611113,31.71359,55.0,47.444443,65.44444,52.
11111,-22.666666,31.333334,-8.666667,65.44444,0.29780185,-2.3567543

FOLIAGE,36.0,145.0,9,0.0,0.0,0.2777778,0.19629629,1.2777778,2.0629628,0.8518519,0.33333334,1.4444444,
0.7777778,-1.5555556,1.7777778,-0.22222222,1.4444444,0.25185186,-2.5309503

FOLIAGE,94.0,144.0,9,0.0,0.0,0.44444442,0.118518494,0.49999997,0.1666667,1.1481482,0.0,3.4444444,0.0,-
3.4444444,6.888889,-3.4444444,3.4444444,1.0,-2.0943952

FOLIAGE,18.0,90.0,9,0.0,0.0,1.0555555,0.37407416,0.6666667,0.22222227,2.8148148,0.5555556,6.888889,1.
0,-6.7777777,12.222222,-5.4444447,6.888889,0.9261464,-2.1643226

FOLIAGE,68.0,103.0,9,0.0,0.0,0.66666675,0.57777774,1.111111,0.96296334,2.148148,0.11111111,5.6666665,
0.6666667,-6.111111,10.555555,-4.4444447,5.6666665,0.9876543,-2.1854658

FOLIAGE,45.0,108.0,9,0.0,0.11111111,25.5,12.795401,27.277779,15.930981,49.814816,41.77778,61.11111,46.555557,-24.11111,33.88889,-9.777778,61.11111,0.35897508,-2.3604913

FOLIAGE,59.0,99.0,9,0.0,0.0,0.77777785,0.8296297,0.99999994,0.3555556,2.0370371,0.11111111,5.5555553,0.44444445,-5.7777777,10.555555,-4.7777777,5.5555553,0.9861111,-2.149743

FOLIAGE,67.0,136.0,9,0.0,0.0,6.722223,3.7083488,2.6666667,3.1972213,15.518518,9.0,25.333334,12.222222,-19.555555,29.444445,-9.888889,25.333334,0.66081303,-2.3075109

FOLIAGE,226.0,110.0,9,0.0,0.0,0.33333334,0.08888887,0.49999997,0.2111111,1.6666666,0.11111111,4.4444447,0.44444445,-4.6666665,8.333333,-3.6666667,4.4444447,0.9777778,-2.1559837

FOLIAGE,231.0,124.0,9,0.0,0.0,3.4444447,14.962965,1.8333334,6.4333353,3.0,1.4444444,5.888889,1.6666666,-4.6666665,8.666667,-4.0,5.888889,0.895369,-2.118937

FOLIAGE,103.0,125.0,9,0.0,0.0,0.9444445,0.82775915,0.83333343,0.6912146,1.7777778,0.44444445,3.8888888,1.0,-4.0,6.3333335,-2.3333333,3.8888888,0.9238095,-2.2375617

FOLIAGE,1.0,81.0,9,0.0,0.0,12.166667,267.45554,9.222222,205.36296,21.333334,14.0,30.555555,19.444445,-22.0,27.666666,-5.6666665,30.555555,0.5952822,-2.438409

FOLIAGE,230.0,124.0,9,0.0,0.0,0.2777778,0.10740743,0.2777778,0.15185183,0.6666667,0.0,2.0,0.0,-2.0,4.0,-2.0,2.0,1.0,-2.0943952

FOLIAGE,58.0,109.0,9,0.0,0.0,0.8888889,0.25185165,2.8333333,1.677777,4.296296,1.4444444,8.444445,3.0,-8.555555,12.444445,-3.8888888,8.444445,0.8644824,-2.2837873

FOLIAGE,59.0,120.0,9,0.0,0.0,2.1666667,1.9860634,1.4444441,1.8698385,19.074074,10.555555,33.11111,13.555555,-25.555555,42.11111,-16.555555,33.11111,0.6812664,-2.2312608

FOLIAGE,140.0,125.0,9,0.0,0.0,0.66666657,0.22222227,2.6666667,3.7777781,3.925926,1.5555556,7.7777777,2.4444444,-7.111111,11.555555,-4.4444447,7.7777777,0.85319865,-2.2342408

FOLIAGE,127.0,143.0,9,0.0,0.0,1.5,0.12222214,0.88888884,0.60740745,4.185185,0.8888889,9.444445,2.2222223,-9.888889,15.777778,-5.888889,9.444445,0.915376,-2.2574124

FOLIAGE,23.0,129.0,9,0.0,0.0,0.5,0.077777795,0.38888887,0.15185186,0.5185185,0.0,1.5555556,0.0,-1.5555556,3.1111112,-1.5555556,1.5555556,0.7777778,-2.0943952

CEMENT,191.0,119.0,9,0.0,0.0,1.1111107,1.2938615,0.9444459,0.772202,39.851852,36.22222,48.22222,35.111111,-10.888889,25.11111,-14.222222,48.22222,0.27171725,-2.0059998

CEMENT,219.0,80.0,9,0.0,0.0,1.2777767,0.3296295,0.6666667,0.9333318,39.703705,36.333336,48.22222,34.555557,-10.111111,25.555555,-15.444445,48.22222,0.28296396,-1.9626464

CEMENT,136.0,45.0,9,0.0,0.0,1.222222,1.5444059,2.0555553,1.5263131,53.074074,48.11111,63.77778,47.333332,-14.888889,32.11111,-17.222221,63.77778,0.25936732,-2.0459049

CEMENT,66.0,160.0,9,0.0,0.0,3.0000007,4.044447,2.777778,0.4296295,22.851852,18.222221,31.555555,18.777779,-13.888889,26.11111,-12.222222,31.555555,0.4260139,-2.1387258

CEMENT,190.0,105.0,9,0.0,0.0,1.8888893,2.2962983,2.166666,1.6777797,45.74074,41.22222,56.333332,39.666668,-13.555555,31.777779,-18.222221,56.333332,0.29599184,-1.9960818

CEMENT,37.0,78.0,9,0.0,0.0,1.0000006,0.17777735,2.5555565,2.8740728,43.814816,41.11111,51.88889,38.444443,-8.111111,24.222221,-16.11111,51.88889,0.25891644,-1.8860723

CEMENT,198.0,127.0,9,0.0,0.0,2.4444444,4.385187,8.555555,59.54075,40.74074,38.0,48.22222,36.0,-8.222222,22.444445,-14.222222,48.22222,0.24899939,-1.9083478

CEMENT,191.0,101.0,9,0.0,0.0,1.111112,0.7793635,1.1111113,1.186342,45.037037,39.0,57.11111,39.0,-18.11111,36.22222,-18.11111,57.11111,0.32273299,-2.0943027

CEMENT,243.0,120.0,9,0.0,0.0,4.4444447,4.359749,1.5555547,1.8338387,47.851852,44.77778,56.333332,42.444443,-9.222222,25.444445,-16.222221,56.333332,0.2453213,-1.9107349

CEMENT,230.0,117.0,9,0.0,0.0,3.2777786,0.92895794,1.4444433,0.86066324,39.037037,33.555557,49.77778,33.77778,-16.444445,32.22222,-15.777778,49.77778,0.33502588,-2.1062348

CEMENT,151.0,89.0,9,0.0,0.0,8.388889,4.577317,0.72222203,0.38968238,31.703703,27.333334,41.0,26.777779,-13.111111,27.88889,-14.777778,41.0,0.35127252,-2.0465415

CEMENT,176.0,100.0,9,0.0,0.0,1.9444441,0.7740748,1.4444447,0.6518514,55.37037,50.333332,66.88889,48.88889,-15.111111,34.555557,-19.444445,66.88889,0.26862052,-2.00619

CEMENT,118.0,126.0,9,0.0,0.0,0.6666667,0.39999974,1.8333336,2.1222224,20.555555,16.11111,28.666666,16.88889,-13.333333,24.333334,-11.0,28.666666,0.437078,-2.1588047

CEMENT,42.0,57.0,9,0.0,0.0,1.3888906,0.97562754,0.99999875,0.36514837,65.703705,59.444443,79.44444,58.22222,-18.777779,41.22222,-22.444445,79.44444,0.26677614,-2.0307028

CEMENT,217.0,148.0,9,0.0,0.0,2.0555556,2.551854,1.0555553,0.5074078,29.074074,21.444445,41.555557,24.222221,-22.88889,37.444443,-14.555555,41.555557,0.4837708,-2.2379715

CEMENT,14.0,146.0,9,0.11111111,0.0,0.888889,0.4740738,1.1111112,0.7407407,10.592592,8.0,15.666667,8.111112,-7.7777777,15.222222,-7.4444447,15.666667,0.48786426,-2.1089108

CEMENT,22.0,87.0,9,0.0,0.0,1.8888899,1.2590423,1.1666666,0.62361073,64.03704,55.77778,80.333336,56.0,-24.777779,48.88889,-24.11111,80.333336,0.30969977,-2.105073

CEMENT,130.0,32.0,9,0.0,0.0,1.1111113,1.047041,0.83333397,0.83665866,59.48148,54.22222,70.88889,53.333332,-15.777778,34.22222,-18.444445,70.88889,0.24900064,-2.042422

CEMENT,162.0,159.0,9,0.0,0.0,2.1666667,0.9603238,2.2222223,1.4555128,26.11111,25.222221,29.555555,23.555555,-2.6666667,10.333333,-7.6666665,29.555555,0.20284556,-1.8025542

CEMENT,150.0,158.0,9,0.0,0.0,2.166667,1.6333338,1.388889,0.41851807,8.444445,7.0,12.222222,6.111111,-4.3333335,11.333333,-7.0,12.222222,0.50308645,-1.9434487

CEMENT,163.0,68.0,9,0.0,0.0,1.833334,2.21111,1.5555559,0.96296287,56.77778,52.0,68.22222,50.11111,-14.333333,34.333332,-20.0,68.22222,0.26505277,-1.9843078

CEMENT,187.0,80.0,9,0.0,0.0,1.3333327,0.7111114,1.3333334,0.7111086,40.51852,37.77778,47.666668,36.11111,-8.222222,21.444445,-13.222222,47.666668,0.24472968,-1.942698

CEMENT,140.0,73.0,9,0.0,0.0,1.7222214,0.8277598,0.7777774,1.0036957,46.296295,45.666668,51.11111,42.11111,-1.8888888,14.444445,-12.555555,51.11111,0.1761556,-1.6815889

CEMENT,236.0,117.0,9,0.0,0.0,0.7777786,0.40368706,1.277778,0.74286777,45.88889,39.555557,58.444443,39.666668,-19.0,37.666668,-18.666666,58.444443,0.33047688,-2.0994802

CEMENT,112.0,90.0,9,0.0,0.0,7.38889,6.9455557,1.2777773,0.87981415,65.18519,57.666668,79.77778,58.11111,-22.555555,43.77778,-21.222221,79.77778,0.27834544,-2.108813

CEMENT,197.0,121.0,9,0.0,0.0,21.666666,17.3628,0.94444436,0.9047208,41.037037,37.444443,49.444443,36.22222,-10.777778,25.222221,-14.444445,49.444443,0.28057775,-1.9955361

CEMENT,141.0,17.0,9,0.11111111,0.22222222,3.7222226,4.4493027,5.0,2.319004,44.592594,40.333336,54.0,39.444443,-12.777778,28.222221,-15.444445,54.0,0.26818594,-2.030048

CEMENT,79.0,28.0,9,0.0,0.0,4.277777,3.7618961,0.8333333,0.6582806,62.407406,53.444443,79.111115,54.666668,-26.88889,50.11111,-23.222221,79.111115,0.32455578,-2.1445074

CEMENT,169.0,102.0,9,0.0,0.0,1.0,0.35555485,0.8888893,0.2962955,58.22222,53.444443,69.66667,51.555557,-14.333333,34.333332,-20.0,69.66667,0.25975996,-1.9855843

CEMENT,208.0,65.0,9,0.0,0.0,1.3888874,1.2367799,26.444445,25.537477,56.703705,52.666668,64.44444,53.0,-12.111111,23.222221,-11.111111,64.44444,0.19713038,-1.9708116

WINDOW,189.0,144.0,9,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0

WINDOW,189.0,141.0,9,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0

WINDOW,210.0,153.0,9,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0

WINDOW,57.0,89.0,9,0.0,0.0,1.7222222,1.3066783,6.1666665,4.7034264,17.0,13.0,24.555555,13.444445,-12.0,22.666666,-10.666667,24.555555,0.47748852,-2.1297464

WINDOW,243.0,94.0,9,0.0,0.0,0.6666666,0.31111112,0.22222221,0.029629631,1.1851852,0.22222222,3.2222223,0.11111111,-2.8888888,6.111111,-3.2222223,3.2222223,0.9777778,-2.0712416

WINDOW,229.0,104.0,9,0.0,0.0,0.49999985,0.54772234,2.8333333,2.0412421,19.777779,16.333334,26.11111,16.88889,-10.333333,19.0,-8.666667,26.11111,0.37553233,-2.1506376

WINDOW,226.0,131.0,9,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0

WINDOW,196.0,95.0,9,0.0,0.0,1.722222,1.1238165,1.3333335,1.264911,7.6296296,6.7777777,10.777778,5.33333335,-2.5555556,9.444445,-6.888889,10.777778,0.51256585,-1.7492584

WINDOW,43.0,152.0,9,0.0,0.0,1.9444445,1.7074082,1.2222222,0.8296299,1.5185186,1.0,2.8888888,0.6666667,-1.5555556,4.111111,-2.5555556,2.8888888,0.64867723,-1.9332389

WINDOW,54.0,133.0,9,0.0,0.0,1.5555557,1.0074074,0.50000006,0.3444443,5.3703704,3.6666667,9.0,3.4444444,-5.111111,10.888889,-5.7777777,9.0,0.6304714,-2.0480003

WINDOW,157.0,85.0,9,0.0,0.0,1.2222223,1.2412657,0.22222233,0.17213264,18.925926,14.555555,26.88889,15.333333,-13.111111,23.88889,-10.777778,26.88889,0.45902446,-2.1609125

WINDOW,96.0,94.0,9,0.0,0.0,0.7222226,0.28518537,0.44444466,0.42963016,20.222221,16.0,28.777779,15.888889,-12.666667,25.666666,-13.0,28.777779,0.45154575,-2.084893

WINDOW,152.0,155.0,9,0.0,0.0,0.5,0.61111104,10.777778,131.80739,7.296296,5.3333335,11.0,5.5555553,-5.888889,11.111111,-5.2222223,11.0,0.5,-2.115567

WINDOW,208.0,34.0,9,0.0,0.0,1.7222224,1.7309811,0.4444445,0.50184834,14.444445,10.777778,21.0,11.555555,-11.0,19.666666,-8.666667,21.0,0.479958,-2.1623826

WINDOW,222.0,62.0,9,0.0,0.0,0.27777782,0.2509242,0.6666667,0.55777335,6.4074073,4.111111,11.444445,3.6666667,-6.888889,15.111111,-8.222222,11.444445,0.68080807,-2.0341523

WINDOW,123.0,152.0,9,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0

WINDOW,20.0,134.0,9,0.0,0.0,0.6666667,0.088888995,0.61111116,0.24074072,2.9629629,1.1111112,6.4444447,1.3333334,-5.5555553,10.444445,-4.888889,6.4444447,0.8292328,-2.133095

WINDOW,223.0,62.0,9,0.0,0.0,0.3333334,0.29814243,0.44444442,0.50184846,6.4444447,4.111111,11.444445,3.7777777,-7.0,15.0,-8.0,11.444445,0.6703704,-2.0487173

WINDOW,184.0,145.0,9,0.0,0.0,0.72222227,0.6116159,0.22222222,0.2721655,0.5555556,0.33333334,1.2222222,0.11111111,-0.6666667,2.0,-1.3333334,1.2222222,0.5277778,-1.9209436

WINDOW,225.0,58.0,9,0.0,0.0,0.3333335,0.42163706,0.4444445,0.34426522,8.333333,5.5555553,14.111111,5.3333335,-8.333333,17.333334,-9.0,14.111111,0.62222224,-2.0685637

WINDOW,58.0,113.0,9,0.0,0.0,3.4444447,1.2232318,0.7222223,0.38968182,20.851852,16.88889,28.444445,17.222223,-11.888889,22.777779,-10.888889,28.444445,0.4148072,-2.1256926

WINDOW,160.0,41.0,9,0.0,0.0,0.8333333,0.8096638,0.22222225,0.17213258,0.7037037,0.5555556,1.2222222,0.33333334,-0.44444445,1.5555556,-1.1111112,1.2222222,0.537037,-1.8451205

WINDOW,160.0,110.0,9,0.0,0.0,0.833333,0.6912145,0.4444437,0.2721654,31.62963,27.222221,39.333336,28.333334,-13.222222,23.11111,-9.888889,39.333336,0.3074441,-2.1872416

WINDOW,186.0,12.0,9,0.0,0.0,0.44444433,0.27216548,2.3333333,1.9663839,6.259259,3.8888888,11.333333,

3.5555556,-7.111111,15.222222,-8.111111,11.333333,0.6873016,-2.057978

WINDOW,142.0,111.0,9,0.0,0.0,1.1111113,0.45541987,0.55555534,0.34426492,28.74074,24.555555,35.88889,25.777779,-12.555555,21.444445,-8.888889,35.88889,0.31548372,-2.2026672

WINDOW,252.0,137.0,9,0.0,0.0,0.88888884,0.544331,0.6111111,0.71232533,0.7407407,0.33333334,1.7777778,0.11111111,-1.2222222,3.1111112,-1.8888888,1.7777778,0.75,-2.0076063

WINDOW,206.0,61.0,9,0.0,0.0,0.5555555,0.40368673,0.4999999,0.45946813,7.0740743,4.666667,12.222222,4.3333335,-7.2222223,15.444445,-8.222222,12.222222,0.64589113,-2.0487173

WINDOW,239.0,93.0,9,0.0,0.0,3.444444,13.496301,1.0555555,0.86296237,14.407408,9.888889,22.88889,10.444445,-13.555555,25.444445,-11.888889,22.88889,0.57700235,-2.1374934

WINDOW,207.0,115.0,9,0.0,0.0,1.0555555,0.3296295,0.16666669,0.033333343,1.2222222,0.44444445,2.8888888,0.33333334,-2.3333333,5.0,-2.6666667,2.8888888,0.93333334,-2.0655363

WINDOW,90.0,134.0,9,0.0,0.0,0.38888887,0.018518528,0.94444436,0.1518523,2.1111112,1.0,4.666667,0.6666667,-3.3333333,7.6666665,-4.3333335,4.666667,0.88148147,-2.0131435

PATH,86.0,197.0,9,0.11111111,0.11111111,1.611112,1.4516907,1.2777786,1.1038647,63.22222,56.22222,77.77778,55.666668,-21.0,43.666668,-22.666666,77.77778,0.28533262,-2.06802

PATH,4.0,189.0,9,0.0,0.0,2.0555565,3.8851852,11.722221,114.59634,26.444445,23.444445,33.0,22.88889,-9.0,19.666666,-10.666667,33.0,0.27147257,-2.1010017

PATH,134.0,187.0,9,0.0,0.0,1.8888887,1.5153537,2.1111114,1.7847083,59.37037,52.666668,74.111115,51.333332,-20.11111,44.22222,-24.11111,74.111115,0.30745777,-2.03348

PATH,252.0,201.0,9,0.0,0.0,4.6111107,5.495115,5.5555553,5.795274,40.296295,35.77778,49.0,36.11111,-13.555555,26.11111,-12.555555,49.0,0.2753682,-2.1758256

PATH,219.0,176.0,9,0.0,0.11111111,2.1111119,1.857917,5.111111,2.8414931,60.296295,53.333332,75.66667,51.88889,-20.88889,46.11111,-25.222221,75.66667,0.31446242,-2.0306482

PATH,205.0,190.0,9,0.0,0.0,1.2777773,0.9981452,1.6111107,1.1238158,49.48148,44.77778,60.666668,43.0,-14.111111,33.555557,-19.444445,60.666668,0.29078844,-1.9875989

PATH,202.0,193.0,9,0.0,0.0,1.111112,0.7503075,3.3888881,2.489236,41.703705,37.88889,50.666668,36.555557,-11.444445,26.88889,-15.444445,50.666668,0.27838665,-1.9929975

PATH,223.0,161.0,9,0.0,0.0,1.3888893,3.040736,8.999999,31.022234,42.037037,36.0,52.22222,37.88889,-18.11111,30.555555,-12.444445,52.22222,0.30841646,-2.282295

PATH,87.0,196.0,9,0.11111111,0.0,1.3333321,0.8692278,2.222222,1.6688869,61.592594,55.0,75.66667,54.11111,-19.777779,42.22222,-22.444445,75.66667,0.28752622,-2.0487363

PATH,105.0,193.0,9,0.0,0.0,3.222222,1.9051597,2.8888893,2.1773255,60.11111,53.11111,75.111115,52.1111

1,-21.0,45.0,-24.0,75.111115,0.30846536,-2.047766

PATH,77.0,185.0,9,0.0,0.0,2.944445,7.6629586,6.2222214,32.118523,32.555557,29.222221,39.666668,28.777779,-10.0,21.333334,-11.333333,39.666668,0.27036038,-2.0852716

PATH,88.0,180.0,9,0.0,0.0,1.4999994,0.93689823,3.833332,2.888675,50.407406,45.22222,62.333332,43.666668,-15.555555,35.77778,-20.222221,62.333332,0.3009182,-2.0087192

PATH,214.0,161.0,9,0.0,0.0,3.7222223,0.72963357,11.5,18.922234,40.444443,35.333336,49.666668,36.333336,-15.333333,27.666666,-12.333333,49.666668,0.29805613,-2.198939

PATH,198.0,183.0,9,0.0,0.0,1.0555547,1.1816497,3.3888881,1.5974506,54.037037,48.88889,66.55556,46.666668,-15.444445,37.555557,-22.11111,66.55556,0.2986217,-1.9749589

PATH,189.0,187.0,9,0.0,0.0,1.2222227,1.0036974,3.0555553,2.3890193,48.51852,44.11111,59.88889,41.555557,-13.222222,34.11111,-20.88889,59.88889,0.30575457,-1.9474715

PATH,124.0,191.0,9,0.0,0.0,3.5000007,2.7386136,3.277778,2.8001328,59.74074,52.88889,74.77778,51.555557,-20.555555,45.11111,-24.555555,74.77778,0.31361094,-2.03548

PATH,182.0,186.0,9,0.11111111,0.0,3.666666,2.0439608,1.7777767,1.2412661,48.444443,43.77778,59.88889,41.666668,-14.0,34.333332,-20.333334,59.88889,0.30452195,-1.9730555

PATH,151.0,163.0,9,0.0,0.0,2.0555556,2.2851863,18.166666,41.988934,40.0,34.555557,48.0,37.444443,-16.333334,24.0,-7.6666665,48.11111,0.2638465,-2.5174901

PATH,112.0,197.0,9,0.0,0.0,4.2222233,2.9938211,4.9444447,3.1227946,50.333332,44.666668,62.22222,44.11111,-17.0,35.666668,-18.666666,62.22222,0.29632849,-2.0712209

PATH,137.0,182.0,9,0.0,0.0,1.8333334,3.8555553,3.6111107,9.218522,34.925926,31.444445,42.88889,30.444445,-10.444445,23.88889,-13.444445,42.88889,0.28575617,-2.0011246

PATH,4.0,201.0,9,0.0,0.22222222,2.722222,1.6788458,5.2222214,3.1245742,53.703705,47.11111,67.22222,46.77778,-19.777779,40.555557,-20.777779,67.22222,0.30885938,-2.0752265

PATH,235.0,196.0,9,0.0,0.0,1.6666666,1.333334,2.222222,1.1482682,47.25926,41.77778,58.555557,41.444443,-16.444445,33.88889,-17.444445,58.555557,0.29423782,-2.0746155

PATH,167.0,189.0,9,0.0,0.0,3.7777774,2.2377224,3.7777774,1.544405,58.407406,51.333332,72.44444,51.444443,-21.222221,42.11111,-20.88889,72.44444,0.29552308,-2.1013258

PATH,250.0,176.0,9,0.0,0.11111111,1.611112,1.0628399,3.444444,1.6953092,53.666668,47.11111,67.66667,46.22222,-19.666666,42.0,-22.333334,67.66667,0.3181372,-2.0522583

PATH,244.0,194.0,9,0.0,0.0,1.722222,1.14342,2.277778,2.03761,49.74074,44.444443,61.666668,43.11111,-15.888889,35.77778,-19.88889,61.666668,0.30085307,-2.0218375

PATH,178.0,185.0,9,0.0,0.0,2.6111119,1.8063676,3.1111114,2.2771008,49.037037,44.0,60.666668,42.444443,

-15.111111,34.88889,-19.777779,60.666668,0.30003616,-2.006186

PATH,229.0,195.0,9,0.0,0.0,4.166666,2.2779152,4.111111,2.2476342,47.703705,42.444443,58.88889,41.7777
8,-15.777778,33.555557,-17.777779,58.88889,0.29027814,-2.0527046

PATH,5.0,210.0,9,0.0,0.11111111,2.1666672,1.6699982,4.444444,2.613356,51.296295,45.444443,64.333336,4
4.11111,-17.555555,39.11111,-21.555555,64.333336,0.3175664,-2.0208955

PATH,118.0,180.0,9,0.0,0.0,1.9444447,1.4819913,3.1111114,1.0886629,48.555557,44.11111,59.0,42.555557,-
13.333333,31.333334,-18.0,59.0,0.27882218,-1.9960423

PATH,86.0,193.0,9,0.0,0.0,3.2777793,1.6521593,2.222222,1.8094397,61.407406,53.666668,76.55556,54.0,-
23.222221,45.444443,-22.222221,76.55556,0.29957896,-2.1093748

GRASS,204.0,156.0,9,0.0,0.0,0.5000003,0.27888626,1.9999996,0.55777323,23.703703,17.333334,25.444445,2
8.333334,-19.11111,5.2222223,13.888889,28.333334,0.38903406,2.8649306

GRASS,71.0,180.0,9,0.11111111,0.0,1.222222,0.5629625,3.0,1.8666646,22.333334,18.333334,21.444445,27.2
22221,-12.0,-2.6666667,14.666667,27.222221,0.3273826,2.4538307

GRASS,60.0,181.0,9,0.11111111,0.0,1.6666666,1.2000005,2.6666667,2.0888875,19.62963,17.11111,17.88889,
23.88889,-7.5555553,-5.2222223,12.777778,23.88889,0.28569087,2.1891353

GRASS,103.0,216.0,9,0.0,0.0,0.88888884,0.6518517,2.1666663,1.1444442,14.555555,10.888889,13.666667,19
.11111,-11.0,-2.6666667,13.666667,19.11111,0.43169576,2.4475467

GRASS,89.0,221.0,9,0.0,0.0,1.3888888,1.4851848,1.4444445,1.5851862,14.074074,10.666667,12.444445,19.1
1111,-10.222222,-4.888889,15.111111,19.11111,0.44601154,2.3172119

GRASS,8.0,199.0,9,0.0,0.0,1.6666671,0.78881073,1.3888893,0.71232533,15.0,13.333333,11.777778,19.88889,
-5.0,-9.666667,14.666667,19.88889,0.40888458,1.9053026

GRASS,200.0,250.0,9,0.0,0.0,2.2777777,1.420746,1.4444445,1.0255988,9.222222,6.5555553,7.3333335,13.77
7778,-8.0,-5.6666665,13.666667,13.777778,0.5633903,2.1929355

GRASS,163.0,166.0,9,0.0,0.0,1.7777776,1.0962971,2.444445,1.0518519,16.962963,12.333333,16.333334,22.2
22221,-13.888889,-1.8888888,15.777778,22.222221,0.4462137,2.5158255

GRASS,186.0,173.0,9,0.0,0.0,1.4444445,0.82963055,2.1111114,1.9851853,17.148148,13.555555,15.555555,22
.333334,-10.777778,-4.7777777,15.555555,22.333334,0.39643326,2.332739

GRASS,2.0,245.0,9,0.0,0.0,1.8888888,2.162963,3.1666667,3.2777781,6.4074073,6.2222223,6.0,7.0,-
0.5555556,-1.2222222,1.7777778,7.2222223,0.19104938,1.7566451

GRASS,62.0,223.0,9,0.0,0.0,2.5,5.722223,1.6666666,2.4444442,6.6666665,4.888889,4.666667,10.444445,-
5.3333335,-6.0,11.333333,10.444445,0.5931498,2.0465877

GRASS,242.0,183.0,9,0.0,0.0,1.4999999,0.9368979,2.1666667,1.798147,15.37037,12.666667,12.444445,21.0,-

8.111111,-8.777778,16.88889,21.0,0.42024404,2.0758736

GRASS,200.0,215.0,9,0.0,0.0,1.5555559,0.95839405,4.722222,2.6450515,23.037037,17.11111,22.555555,29.444445,-17.777779,-1.4444444,19.222221,29.444445,0.42076695,2.5610487

GRASS,57.0,177.0,9,0.0,0.0,0.94444495,0.55185163,1.4444443,1.71852,19.592592,15.777778,18.444445,24.555555,-11.444445,-3.4444444,14.888889,24.555555,0.3589904,2.413262

GRASS,63.0,220.0,9,0.0,0.0,3.055556,15.2629595,3.6666667,6.0888896,8.185185,6.5555553,6.4444447,11.555555,-4.888889,-5.2222223,10.111111,11.555555,0.48671728,2.0931497

GRASS,117.0,224.0,9,0.0,0.0,3.0555556,2.5596585,3.0555556,3.0215273,20.25926,15.444445,17.777779,27.555555,-14.444445,-7.4444447,21.88889,27.555555,0.44151622,2.3041532

GRASS,233.0,211.0,9,0.11111111,0.0,2.6666667,6.0888896,1.6666666,1.7333333,15.444445,12.444445,15.222222,18.666668,-9.0,-0.6666667,9.666667,18.666668,0.33542147,2.5549645

GRASS,232.0,175.0,9,0.0,0.0,1.1666669,0.62360954,2.8333333,2.1679492,15.37037,12.444445,13.555555,20.11111,-8.777778,-5.4444447,14.222222,20.11111,0.38741234,2.2421958

GRASS,29.0,195.0,9,0.0,0.0,0.44444433,0.40368652,0.61111087,0.6804136,16.037037,14.444445,13.333333,20.333334,-4.7777777,-8.111111,12.888889,20.333334,0.34230694,1.9430399

GRASS,63.0,201.0,9,0.0,0.0,1.4444445,0.9185186,0.94444424,0.5518518,7.7777777,6.2222223,5.5555553,11.555555,-4.6666665,-6.6666665,11.333333,11.555555,0.54320127,1.9829868

GRASS,93.0,236.0,9,0.11111111,0.0,1.7777777,2.6962965,1.9999999,0.93333334,12.518518,9.555555,10.777778,17.222223,-8.888889,-5.2222223,14.111111,17.222223,0.44724658,2.2671974

GRASS,72.0,191.0,9,0.11111111,0.0,1.2777777,0.5074075,3.1111114,4.6074066,15.185185,12.111111,14.888889,18.555555,-9.222222,-0.8888889,10.111111,18.555555,0.3503698,2.545329

GRASS,52.0,170.0,9,0.0,0.0,0.55555564,0.4554201,0.94444436,0.38968262,25.444445,20.11111,25.333334,30.88889,-16.0,-0.33333334,16.333334,30.88889,0.34911168,2.6040132

GRASS,233.0,184.0,9,0.0,0.0,0.5000002,0.077777766,0.7777777,0.785185,11.851851,9.777778,9.888889,15.888889,-6.2222223,-5.888889,12.111111,15.888889,0.40555555,2.1286457

GRASS,237.0,191.0,9,0.0,0.0,1.0,0.31111106,1.5,1.0111109,7.3333335,5.3333335,5.4444447,11.222222,-6.0,-5.6666665,11.666667,11.222222,0.5358197,2.1224225

GRASS,36.0,243.0,9,0.11111111,0.0,1.888889,1.8518513,2.0,0.7111104,13.333333,9.888889,12.111111,18.0,-10.333333,-3.6666667,14.0,18.0,0.4522286,2.3683105

GRASS,186.0,218.0,9,0.0,0.0,1.1666666,0.74444425,1.1666665,0.65555507,13.703704,10.666667,12.666667,17.777779,-9.111111,-3.1111112,12.222222,17.777779,0.40134683,2.3826835

GRASS,197.0,236.0,9,0.0,0.0,2.4444444,6.829628,3.3333333,7.599998,16.074074,13.111111,16.666668,18.44

4445,-8.888889,1.7777778,7.111111,18.555555,0.29272884,2.7898002

GRASS,208.0,240.0,9,0.11111111,0.0,1.0555557,0.86296326,2.4444444,5.007407,14.148149,10.888889,13.0,18.555555,-9.777778,-3.4444444,13.222222,18.555555,0.42162097,2.3924873

GRASS,223.0,185.0,9,0.0,0.0,0.5,0.34960312,2.3888886,2.0807757,12.962963,11.555555,9.777778,17.555555,-4.2222223,-9.555555,13.777778,17.555555,0.44541803,1.8388497

EK3 LENNA GÖRÜNTÜSÜNE AİT MATRİS

149 174 179 186 182 131 90 100 116 113 114 104 93 80 60 33 35 35 52 69 105
130 150 152 153 157 152 150 151 155 169 175 177 178 168 171 173 175 178 186 184
184 184 172 143 113 87 74 59 40 27

165 181 188 178 113 83 101 114 123 116 112 130 126 108 88 74 53 45 35 32 40
67 112 138 145 145 140 146 151 156 166 172 173 173 169 169 180 176 172 177 166
151 117 85 68 52 49 56 70 66 63

180 189 172 94 81 92 93 106 105 106 72 118 130 116 90 96 62 63 55 49 47
50 70 104 137 137 139 141 143 149 163 168 175 176 171 169 174 164 147 126 89 87
78 85 82 56 56 72 100 75 67

191 158 74 65 74 70 72 79 79 75 26 67 115 80 61 65 68 88 68 75 65 69
67 90 115 125 127 136 136 145 163 172 177 179 175 170 172 166 134 109 96 87 52
40 25 20 23 30 64 48 47

144 54 53 54 51 59 36 33 35 30 16 29 46 24 20 25 52 86 64 92 85 81 81
89 101 116 127 129 127 142 158 175 175 181 177 174 174 150 107 59 39 27 19 22
21 17 18 24 31 33 38

43 42 38 31 22 26 23 22 19 18 20 24 26 24 16 10 14 27 85 106 92 90 88
93 103 110 118 125 130 140 160 173 177 182 185 182 159 90 40 29 25 30 22 39
39 24 20 21 28 26 28

31 30 19 20 21 22 24 18 14 16 16 37 67 57 27 17 14 12 27 74 102 96 90
91 99 106 114 119 128 140 162 175 187 187 191 185 80 23 24 19 28 33 50 86
106 63 20 13 31 32 26

33 22 17 20 19 20 18 17 15 24 27 70 150 150 88 36 13 12 13 41 88 98 89
90 98 100 109 117 125 137 161 177 192 194 197 175 31 23 22 19 24 42 119 91
143 101 41 17 19 26 26

26 17 11 26 49 29 34 28 9 30 48 56 173 188 168 126 43 25 59 40 79 87
81 86 95 101 110 111 125 138 156 185 199 204 195 111 39 30 48 24 27 43 62
83 155 123 57 28 13 24 23

21 19 14 35 73 48 47 62 26 40 67 39 173 190 189 176 109 37 75 64 63 81
75 84 90 99 103 106 122 143 166 189 207 207 152 62 81 38 68 61 53 64 36
114 149 121 65 32 18 25 17

49 26 14 38 72 88 39 54 59 55 31 97 198 199 197 192 147 64 53 82 57 72
74 80 84 94 99 112 130 146 170 195 206 197 112 60 107 87 52 65 66 46 89
159 148 113 59 28 14 14 12

81 59 16 25 68 86 82 46 31 42 94 176 193 193 189 177 135 94 65 86 75 74
86 78 84 94 103 111 129 147 182 204 209 187 82 77 91 112 95 72 77 108 161
160 146 103 57 27 19 16 16

99 83 53 50 63 81 101 96 90 117 167 183 190 196 191 158 108 115 91 78 95 80
78 84 90 91 95 105 126 152 189 212 210 173 79 60 75 88 115 115 122 137 145
133 112 71 42 15 24 22 16

110 104 94 71 66 68 83 99 115 127 139 155 164 164 158 157 133 112 95 94 95
86 81 80 90 92 88 110 126 153 185 212 210 169 106 78 128 112 118 100 93 109
112 73 52 37 27 28 35 33 24

110 112 107 91 98 91 71 81 62 91 103 102 109 99 125 145 151 155 141 109 87
86 85 90 88 92 92 107 121 151 182 212 213 176 151 142 115 107 105 85 82 84
81 56 41 38 54 51 56 45 24

112 110 112 112 113 98 84 90 69 96 105 99 89 92 123 120 130 131 117 102 85
84 92 95 88 89 88 105 118 142 176 210 216 182 144 127 125 110 100 94 99 92
94 80 62 52 62 64 51 41 21

120 113 107 112 118 107 106 103 95 99 106 107 105 119 126 129 124 117 121 118
89 86 96 98 92 90 86 95 120 138 177 207 216 190 156 136 137 127 115 107 103
94 90 85 72 69 72 73 61 47 21

122 123 123 121 116 109 113 111 108 110 113 122 124 130 136 137 132 134 136 111
95 95 100 99 91 87 88 90 114 141 176 201 214 195 169 154 142 131 127 120 112
95 93 94 86 79 82 79 66 49 26

131 133 132 132 128 121 123 128 123 131 134 138 141 144 139 143 134 141 132 107
93 99 106 99 87 90 89 89 113 137 167 201 216 196 174 164 152 137 129 124 119
105 99 94 88 82 82 76 73 56 29

135 141 134 139 148 148 140 138 140 139 149 151 147 145 139 146 138 141 124 105
101 111 107 99 92 91 90 89 111 141 168 199 213 202 176 165 158 143 131 122 115
106 100 98 93 87 86 83 71 59 27

133 145 135 140 147 152 147 143 142 143 148 148 144 142 147 147 141 130 121 108
103 109 101 96 89 91 89 85 101 127 168 192 214 204 177 164 153 151 135 119 111
106 101 100 95 88 87 84 77 60 34

132 141 145 143 149 152 148 156 152 151 147 143 145 152 153 150 139 128 119 114
107 110 107 98 90 96 85 85 97 124 161 192 213 208 184 161 146 139 137 129 117
105 98 94 90 91 91 84 80 67 35

132 140 147 147 149 158 151 144 156 154 157 155 151 152 154 149 143 133 120 115
117 112 107 99 94 94 88 91 98 118 154 194 212 206 187 160 146 140 135 129 119

107 101 96 89 90 90 87 81 63 31

131 139 142 148 150 156 156 153 153 155 152 154 155 156 149 142 135 127 125 117
113 110 108 102 93 93 91 87 95 119 152 191 210 211 192 161 145 136 134 131 119
113 107 96 94 89 91 88 82 64 26

126 135 145 150 152 156 155 157 161 156 153 149 154 149 148 142 136 131 121 116
115 115 111 105 98 96 86 94 96 118 143 189 214 215 195 160 144 139 136 126 120
114 107 96 96 90 92 89 77 56 27

123 132 140 143 148 151 156 155 161 161 150 147 150 147 148 136 131 125 120 113
110 115 108 102 96 88 89 95 97 112 146 183 208 214 200 162 144 144 132 129 122
114 109 100 93 90 90 86 74 49 27

122 123 128 137 139 146 153 155 153 161 152 148 151 145 141 136 131 123 109 109
104 112 104 101 92 85 92 92 100 115 136 173 205 215 202 163 141 136 134 123
117 122 109 102 95 94 90 85 75 43 30

121 122 127 135 139 143 150 150 153 153 151 144 146 144 140 135 128 120 114 106
103 103 104 97 90 86 86 92 96 114 129 155 200 216 205 164 134 134 131 126 114
115 104 102 96 93 92 84 68 38 23

120 126 124 130 139 142 152 148 149 149 149 142 145 140 137 134 124 120 107 98
99 98 103 95 87 88 83 89 99 105 128 156 195 213 209 170 139 140 128 123 113
112 105 98 96 94 88 83 63 32 23

119 126 127 131 130 135 141 148 139 145 147 144 145 141 136 135 127 120 103 97
97 98 96 94 91 82 82 87 92 98 130 156 193 211 214 183 129 132 126 122 116
112 108 98 93 93 90 72 52 30 14

119 124 122 129 134 128 134 142 141 144 142 143 141 135 134 130 123 117 105 100
94 95 96 91 88 84 82 85 88 105 125 151 180 209 219 197 136 134 125 123 115
113 104 100 95 92 84 70 47 27 20

111 121 118 127 132 127 129 131 134 140 137 140 135 137 138 130 121 111 103 98
94 92 93 91 86 82 82 84 88 101 121 139 169 209 221 205 145 130 127 121 119
113 103 102 94 93 84 62 45 16 17

113 117 118 118 121 126 126 124 131 136 138 133 133 134 128 124 114 109 100 90
89 86 94 99 85 84 85 78 90 98 120 137 161 207 223 207 151 118 118 120 121
110 104 97 91 89 84 58 30 17 20

113 115 120 119 119 120 122 129 130 131 137 131 135 130 127 119 115 112 95 82
78 89 96 99 92 87 86 84 93 99 114 138 157 203 224 211 148 116 117 117 116
108 100 94 93 87 77 37 18 17 21

109 111 120 120 117 121 123 125 126 130 129 131 132 131 124 117 108 109 83 69
81 95 103 100 103 103 95 87 89 97 108 137 160 199 224 210 143 116 118 117 109
104 100 92 89 87 66 25 20 18 20

109 109 112 119 119 120 127 125 127 128 129 133 130 129 125 118 114 111 86 68
81 99 105 101 106 107 106 89 81 87 108 134 163 190 212 205 127 117 116 114 112
109 104 95 91 81 38 17 15 18 19

109 107 114 118 117 117 116 121 126 122 129 128 129 128 126 117 113 109 83 70
86 99 97 94 85 97 96 91 80 79 97 117 152 183 194 189 124 118 117 116 110
104 94 91 90 65 22 17 17 20 22

108 107 119 113 118 120 123 120 122 125 127 128 130 127 130 121 114 107 97 79
94 97 87 30 18 37 61 66 81 83 86 98 138 172 186 152 137 117 115 112 106
104 99 96 86 39 22 20 20 21 27

112 110 111 119 117 115 117 118 126 127 128 127 128 128 129 120 117 115 102 80
86 92 80 55 46 66 66 55 54 71 81 91 118 162 172 144 131 123 116 109 105
102 94 95 70 24 22 29 21 21 26

109 111 111 116 117 110 119 119 122 122 122 127 131 126 131 124 117 118 111 93
81 85 89 91 91 93 91 79 75 70 81 89 123 169 158 144 128 123 116 109 103 98
98 88 46 18 22 26 27 19 30

108 110 109 115 111 113 118 115 125 117 123 127 128 128 128 127 120 119 115 108
102 95 84 87 89 93 96 81 81 101 133 157 171 171 153 139 123 120 115 112 103
100 94 76 27 24 20 26 24 24 29

103 106 110 109 111 113 116 114 122 122 123 122 128 128 130 125 125 125 121 115
109 107 96 96 96 107 123 116 110 138 178 184 181 171 151 139 125 117 112 107
97 98 94 57 27 27 22 22 23 23 31

104 108 108 108 111 107 111 118 120 122 125 128 130 128 128 128 124 127 120 117
117 112 107 105 109 124 156 158 124 144 179 182 179 174 154 134 117 113 115 105
103 102 84 27 24 27 25 26 20 26 29

104 104 102 107 106 110 115 121 121 121 121 122 125 128 127 126 122 121 122 120
116 116 112 112 123 141 177 180 143 152 180 183 180 177 150 133 120 111 114 103
101 96 55 21 26 27 21 22 25 27 41

106 107 103 102 104 103 108 113 118 118 120 120 118 122 121 126 122 117 119 119
116 116 117 121 127 154 190 186 158 151 184 187 184 179 147 130 114 114 111 104
104 89 26 23 21 24 24 25 22 24 40

104 101 104 103 108 105 107 108 112 115 116 118 115 120 121 127 121 116 114 120
122 119 122 123 136 163 197 192 167 153 186 191 181 173 149 131 109 112 107 106

100 63 22 22 23 25 23 24 21 29 40

102 108 100 101 107 99 106 103 105 112 114 115 112 113 115 124 114 119 113 117
120 125 126 129 141 166 198 194 167 147 175 192 184 174 147 121 106 106 106 106
96 30 18 20 21 38 27 22 26 28 38

104 108 102 105 100 99 102 102 106 105 107 111 112 112 109 116 110 112 113 114
116 122 124 130 141 168 193 199 174 168 195 191 184 176 150 115 94 100 108 105
68 25 20 18 26 38 25 22 23 33 36

101 102 104 103 101 99 100 105 108 107 101 106 110 103 103 111 114 112 106 110
121 127 133 130 131 132 147 153 179 180 155 139 136 154 145 102 91 107 111 101
31 23 19 23 28 25 25 28 29 27 41

98 101 105 106 101 102 101 106 99 95 92 91 95 100 102 110 105 111 107 111 106
104 101 96 93 91 105 104 118 122 111 101 104 115 106 76 97 113 111 74 23 22
19 23 29 23 23 22 29 29 44

94 99 103 107 107 106 107 114 91 57 53 70 83 88 93 95 92 91 87 81 78 79
79 75 76 72 81 87 101 97 62 51 67 78 61 70 97 113 111 74 23 22 19 23
29 23 23 22 29 30 50

ÖZGEÇMİŞ

Doğum tarihi	28.09.1979	
Doğum yeri	Eskişehir	
Lise	1993-1997	Haydarpaşa Anadolu Lisesi
Lisans	1997-2001	Yıldız Teknik Üniversitesi Elektrik- Elektronik Fak. Elektronik ve Haberleşme Mühendisliği Bölümü
Yüksek Lisans	2001-2003	Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Haberleşme Müh.,Elektronik Programı

Çalıştığı kurum(lar)

2001-Devam ediyor	YTÜ Elektrik Elektronik Fakültesi Elektronik ve Haberleşme Mühendisliği Elektronik Anabilimdalı Araştırma Görevlisi