YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

TEZİN ADI

BİYOMEDİKAL MÜHENDİSLİKTE NÖRAL DEVRELER İNCELEMESİ

YÜKSEK LİSANS TEZİ

TEZİ YAPAN : Elk.Müh.Cezmi Ateş
TEZ YÖNETİCİSİ : Doç.Dr.Galip Cansever

İSTANBUL 1992

İÇİNDEKİLER

ÖZET

Bu çalışma üç ana bölümden oluşmaktadır.Birinci bölümde yapay nöral ağlara (YNA) bir giriş yapılmakta ve temel yapı blokları ,öğrenme kuralları ve eğitme algoritmaları anlatılmaktadır.Öğrenme kuralları olarak beş önemli kural aktarılmıştır.

İkinci bölümde Biyomedikal Mühendislikteki bazı problemlere YNA ile çözüm arayan altı uygulama anlatılmaktadır.Bu uygulamalar aktarılırken kullanılan YNA yöntemleri üzerinde önemle durulmuş ve sonuçları ağ başarısını göstermek amacıyla ayrıntılı olarak verilmiştir.

Üçüncü bölüm bir diyet programı çalışmasıdır. Yaklaşık 150 cm boyundaki ve yaşları 15-55 arasında değişen kadın ve erkekler için, ideal kiloyu ve günlük dinlenme halindeki kaloriyi hesaplayan bir YNA programı gerçekleştirilmiştir.Tartışma ve sonuç bölümünde bu programın neticeleri konusunda bir yorum yapılmıştır.

SUMMARY

This study consists of three sections.In the first section ,artificial neural networks (ANN) are introduced and fundamental structural blocks ,training rules and training algorithms are stated.Five important rules have been put forward as training rules.

In the second section ,six applications looking for solutions to the Biomedical Engineering problems using artificial neural networks have been mentioned.While these applications are mentioned, the artificial neural network procedures used are examined carefully and the results of the applications are expressed in detail to demonstrate the performance of the network.

The third section is a study of a diet program. An artificial neural network software that is calculating the ideal weight and daily calory required in the resting period for males and females with 150 cm average height and having age between 15 and 55 ,has been implemented.In the Discussion and Results section ,same comments about the results of the program has been stated.

BİRİNCİ BÖLÜM
NÖRAL AĞLARLA HESAPLAMA TEKNİĞİ

I.1. TARİHÇE

Yapay nöron ağları (YNA) enformasyon işleme sistemleri ile ilgili teknolojik bir disiplindir.Temel olarak insan vücudundaki sinir sistemi esas alınmıştır ve işlem elemanlarından oluşur.Bu işlem elemanları daha sonraki bölümlerde anlatılacağı üzere ağın işleyişini meydana getirirler.

1980'li yıllara kadar klasik anlamda programlama yapılıyordu.Yani datalar programlanıyordu.Transfer fonksiyonu ise bilgisayarda simüle ediliyordu.45 yıldır kullanılan bu yöntemden sonra özellikle 1985'ten itibaren nöral ağ uygulamaları özerinde geniş bir şekilde çalışıldı.

Genel olarak nöral ağ çalışmaları dört zaman dilimine bölünebilir.[1] Karanlık zaman da denilen ilk dilimde çalışmalar sadece geliştirme aşamasındaydı ve teorikti.Bu dilim 1969'da Minsky ve Papert'in perceptron'la ilgili kitapları yayınlanıncaya kadar sürdü.1969-1982 yılları arasındaki çalışmalarda teori artık oturuyor ve 1982'de Hopfield'ın "Neural Networks and Physical Systems" adlı makalesi ile ikinci dilim de sona eriyordu.Rönasans olarak da adlandırılan üçüncü dilim de 1986'da Rumelhart ve Mc Clelland'ın "Parallel Distributed Processsing" [2] kitabı ile kapanıyordu.Bu üçüncü devirde teorik çalışmalar büyük ölçüde tamamlanmıştı.1986'dan günümüze kadar olan son dilime "nörokonnektivizm" denilmektedir ve uygulama çalışmaları son derece yoğun bir şekilde devam etmektedir.

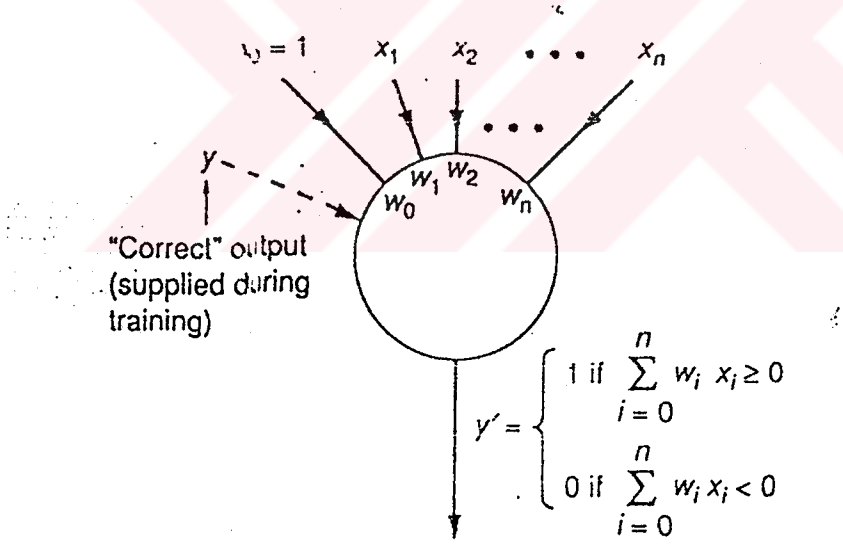
YNA temel olarak basittir.Klasik bir bilgisayar algoritmasının programlaması mükemmel bir yazılım gerektirir.En küçük bir hata dahi programların çalışmasını engeller.YNA ise bir algoritma veya kurallar geliştirmeye gerek duymayan bir sistemdir.Gerekli olan "öğrenme"yi uygulamak için giriş ve çıkış bilgilerinden oluşan bir bilgi setine ihtiyaç vardır.Bu verilerin elde edilebildiği her alanda (form tanıma,data analizi,kontrol) YNA uygulanabilir.Bazı örneklerde klasik bilgisayar algoritmalarına zaman,ekonomi ve özellikle dayanıklılık

açısından üstünlük sağlamıştır.

I.2.YNA'NIN KAVRAMLAŞTIRILMASI VE TEMEL YAPI BLOKLARI

I.2.1.YNA'nın TANIMI:

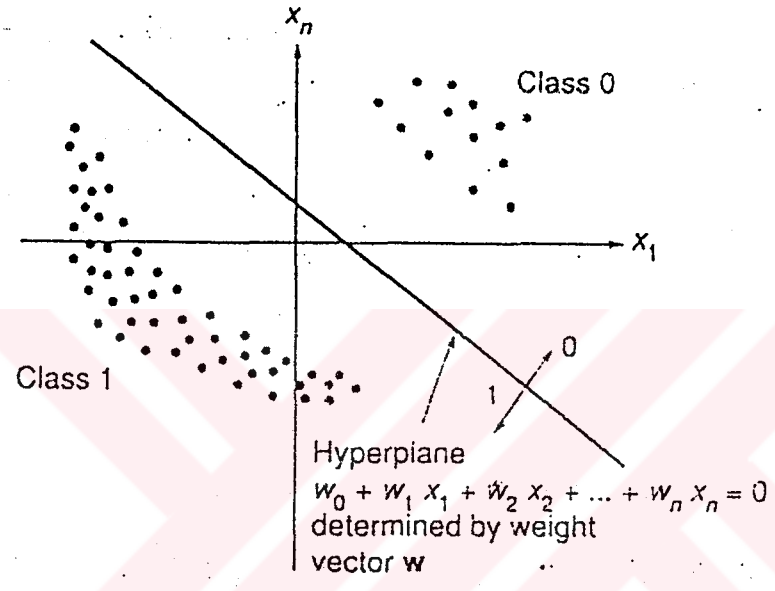
YNA paralel dağıtılmış bir enformasyon işleme sistemidir.Bu sistem tek yönlü işaret kanalları (bağlantılar,connections) ile birbirine bağlanan işlemsel elemanlardan oluşur.Çıkış işareti bir tane olup çoğaltılabilir.(fan out).İşlemsel elemanın (processing element) çıkışı istenen matematiksel tipte olabilir.İşlemler tamamen lokal olarak yapılır yani her bir eleman kendi işlemini yapar.Giriş işaretinin halihazırdaki değerleri ve yerel hafızadaki değere bağlı olarak çıkış işareti her bir eleman için lokal olarak belirlenir.Bu anlatılanlar aşağıdaki perceptron işlem elemanında kolayca görülebilir.



Şekil I.1 Perceptron işlem elemanı

Perceptron probleminde temel amaç n-boyutlu bir giriş vektörünü sınıf 0 ve 1 olarak ayırabilmektir.Öyle ki giriş vektöründeki noktalar şekil I.2. 'de olduğu gibi hyperplane doğrusunun 1 tarafında ise perceptron işlem elemanının çıkışı 1 ,0 tarafında ise perceptron işlem elemanının çıkışı 0 olmalıdır.

Şekil I.1.'deki $x_0,x_1,x_2,...x_n$ n boyutlu bir giriş vektörüdür.$w_0,w_1,...w_n$ ise bu giriş değerlerine karşı belirlenmiş ve "ağırlık" adı verilen sayısal değerlerdir.y girişi ise doğru değeri bilinen çıkış işaretidir.(Mesela ölçülmüş bir değerdir.) y' ise perceptron işlem elemanının çıkışıdır.Burada amaç öğrenme kuralları kullanarak y' ile y'yi aynı yapmaya çalışmaktır.



Şekil I.2 n-boyutlu uzayda karakter (form) sınıflama problemi

y' çıkışı perceptrona özgü olarak şöyle verilir.

$$y' = \begin{cases} 1 , & \sum_{i=0}^{n} w_i x_i \geq 0 \\ 0 , & \sum_{i=0}^{n} w_i x_i < 0 \end{cases}$$

Ağırlıklar supervised öğrenme kurallarına göre şu formülle (örneğe özgü) ayarlanır.Ağırlık ayarlama işlemi iterasyonla yapılır.

$$w^{yeni} = w^{eski} + (y - y')\, x$$

Eğitme işlemi sonunda y=y' olur veya çok çok küçük

hatalarla yaklaşım yapılır.Eski indisi ağırlığın bir önceki iterasyondaki değerini ifade eder.

YNA işleyişi için klasik bir software gerekebilir fakat bu tamamlandıktan sonra YNA artık bir hardware yapısına bürünür.

Veri ----------> YNA -----> Çıkış
Programlama

Genel olarak YNA'nın kapasiteleri ve özellikleri hakkında teoremler ispatlamak mümkündür.Fakat nasıl çalıştıklarını anlamak büyük önem arzeder.

Nöron ağları ile hesaplama şu üç konuda aktivite gösterir:

1.Yapı ve Teorisi : Formal (makinaya bağımlı) matematiksel NA tanımı , modellemesi ve ilgili konularda teori geliştirmek ,oyuncak problemler kullanarak nedensellik araştırmasıdır.(XOR v.s.)

2.Fiziksel Tasarım :Nöron ağları tasarlamaya çalışmak ve bilinen diğer sistemler ile beraberce çalışmasını sağlamaktır.

3.Uygulamalar :En kalabalık çalışma alanıdır.Binlerce matematikçi ,iş analistleri,mühendisler bu konuda çalışmaktadırlar. Uygulama alanı da uzmanları daha olmaktadır.

YNA ile hesaplama yapmak oldukça basit bir disiplindir. Problemlere direkt yaklaşma imkanı arttıkça genel bir set ile problem çözmek mümkün olacaktır.

I.2.2.TEMEL KAVRAMLAR ,TANIMLAR ve YAPI BLOKLARI

YNA temel olarak basit yapılardır.Bu basit yapıların genel olarak bir araya gelmesini sağlayan maddeleri incelemek bu çalışmanın önemli noktalarından biridir.Bu bölümde bir genel model ve birtakım tanımlayıcı terimler tanıtılmaktadır.

Bir paralel dağılmış enformasyon işleme yapısı olan

YNA'nın aşağıdaki alt tanım ve kısıtlamaları vardır.

1.Düğümler işlem elemanları olarak kullanılır.

2.Düğümler arasında bağlantılar vardır.Her bağlantı tek yönlü işaret iletim yoludur ve gecikmesiz olarak görev yapar.

3.Her işlem elemanı istenildiği sayıda giriş bağlantısı alabilir.

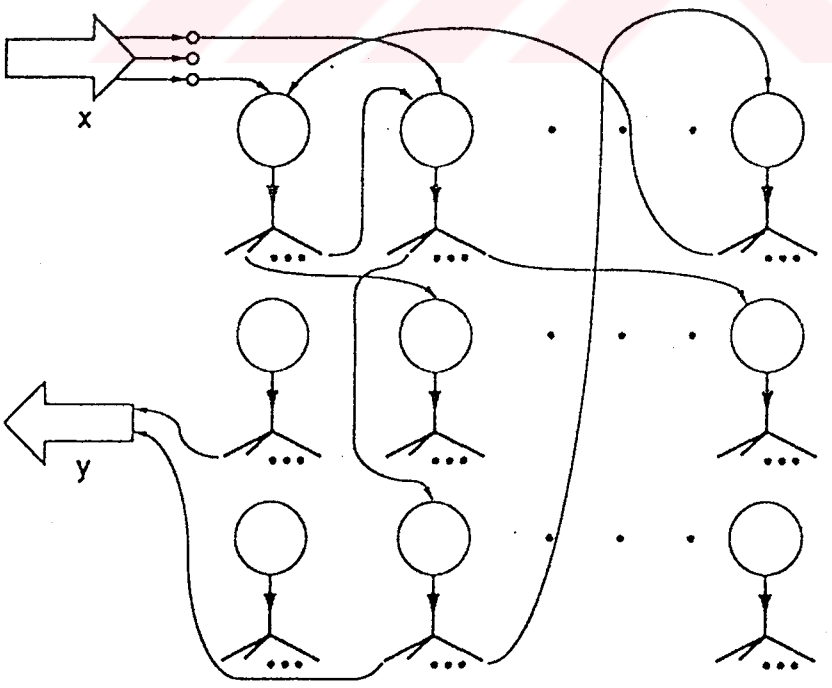
4.Her işlem elemanında tek bir çıkış bağlantısı olabilir.Fakat bu bağlantı kopya edilebilir.

5.İşlem elemanları yerel bellek taşıyabilir.

6.Her işlem elemanının bir transfer fonksiyonu bulunur. Transfer fonksiyonları sürekli veya kısmen sürekli olabilir.Kısmen sürekli çalışma konumunda işlem elemanı "aktif" ise bir çıkış üretir.Yerel bellekteki kabul edilebilir işaretler giriş işareti olabilir.

7.Giriş işaretleri yapay nöron ağlarına bilgi taşır,sonuç ise işaretlerinden alınabilir.

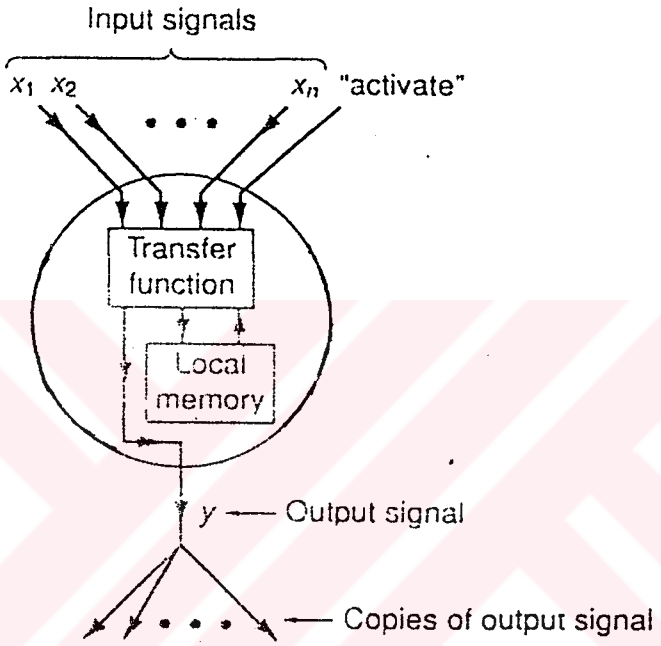
YNA yapısı denince bilgisayarda matematiksel olarak nasıl gerçeklendiği akla gelecektir.



Şekil I.3 Tipik bir nöral ağ yapısı.

Her bir işlem elemanının bir çıkışı vardır ama çıkış çoğaltılabilir.

Aşağıdaki şekilde genel bir işlem elemanı görülmektedir.Literatürde nörod,ünite,işlem elemanı,nöron kelimeleri ile belirtilir.Aktif girişi işlem elemanını aktif yapar ve bir çıkış üretilir.



Şekil I.4 Genel bir işlem elemanı.

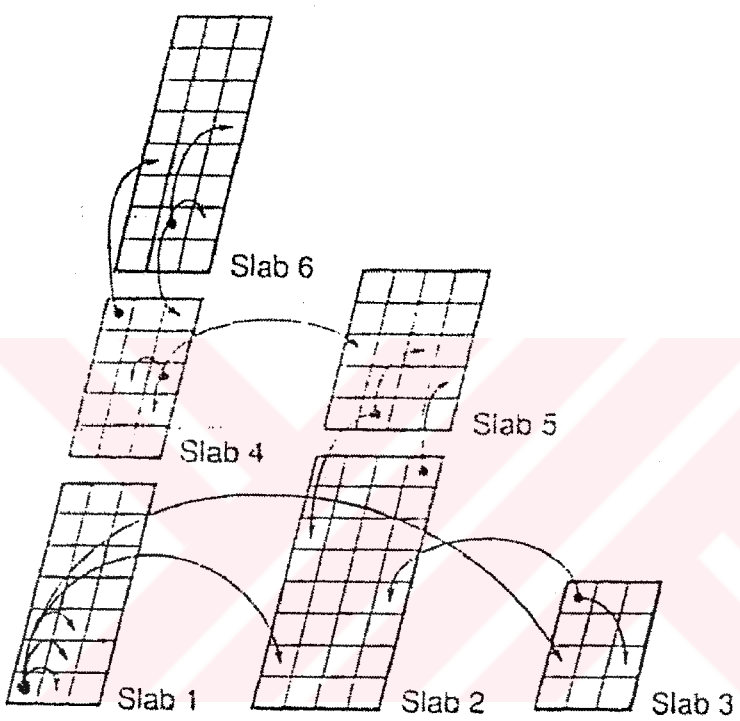
YNA ağı bir takım alt kümelere ayrılabilir.Bu alt kümelerdeki elemanların transfer fonksiyonları aynıdır ve bu küçük gruplara katman veya tabaka denir. (Layer veya slab) Bu multi-layer (çok tabakalı) ağa Şekil I.5 'teki yapı örnek gösterilebilir.

Pek çok nöron ağı giriş katmanı diye bir bölüm içerir. Giriş katmanı dış dünyadan alınan bilgileri ağa aktarır. Bir transfer fonksiyonu yoktur.

Bir çok uygulamada üç tabakalı ağ yapısı kullanılır. Giriş tabakasından başka gizli (hidden) ve çıkış (output) tabakası vardır.Ağın başarısında gizli tabakadaki işlem

elemanı sayısının büyük önemü vardır ve kaç tane olması gerekitiğini açıklayan kesin bir teori yoktur.Ancak çalışmalar sürmektedir.

NA'nın transfer fonksiyonu ve yerel bellek bir öğrenme kuralı ile giriş-çıkış işareti arasındaki bağıntıya göre ayarlanır.Aynı şekilde aktif yapma giriş işlemi için bir zamanlama fonksiyonu tanımlanması gerekebilir.



Şekil I.5 Her katman aynı transfer fonksiyonuna sahiptir ve aynı anda çıkış verir.Bağlantı konfigürasyonu herhangi bir şekilde tamamen isteğe göre değiştirilebilir.

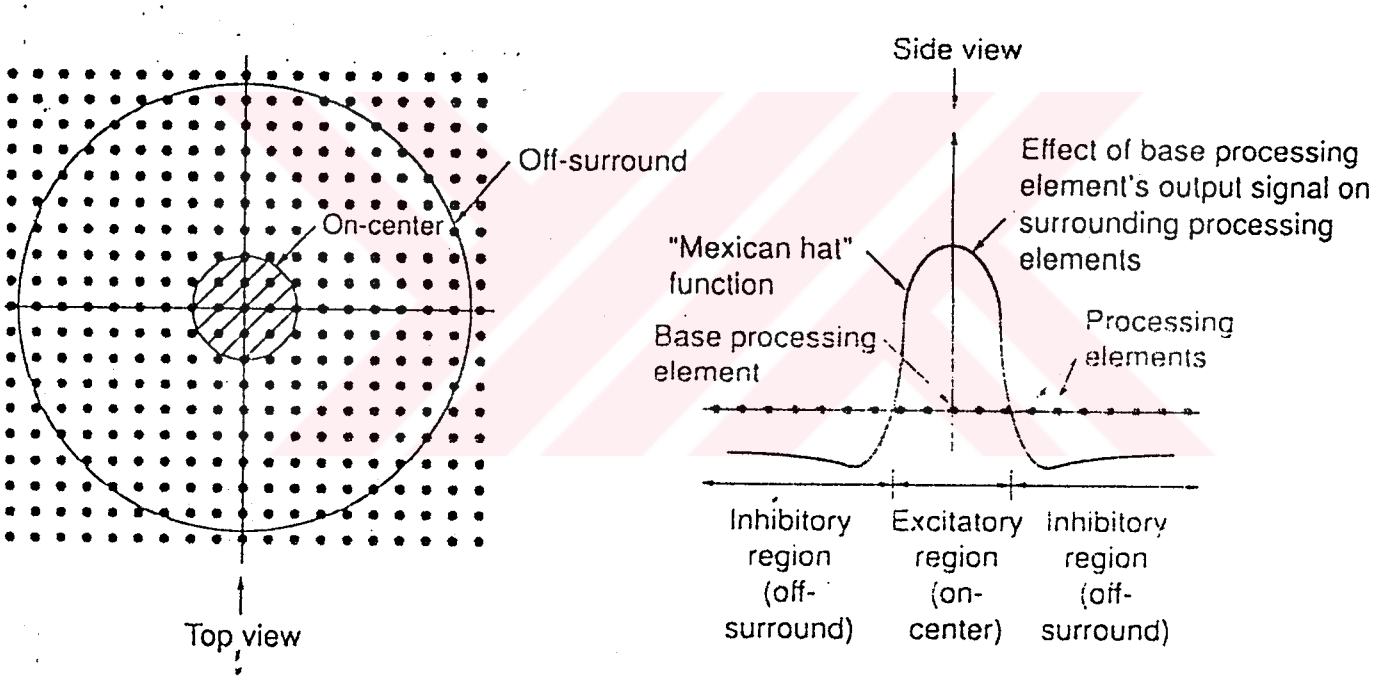
YNA'nın önemli parçası olan bağlantılarda taşınan işaretin cinsi de tanımlanmalıdır.Bağlantı işareti her cinsten olabilir.Tam,sanal,binary sayılar bağlantılarda yer alabilir.Örneğin;

. 7 bitlik bir tamsayı 2'ye tümlenen formatıyla (-64'den +63'e kadar)
. -1.8 ile 3.44 arasındaki reel sayılar
. Fuzzy sayıları

Bu açıklamalar YNA'nın çok değişik bir hesaplama yöntemi olduğunu gösterir.

I.2.3.GİRİŞ İŞARETİ SINIFLARI

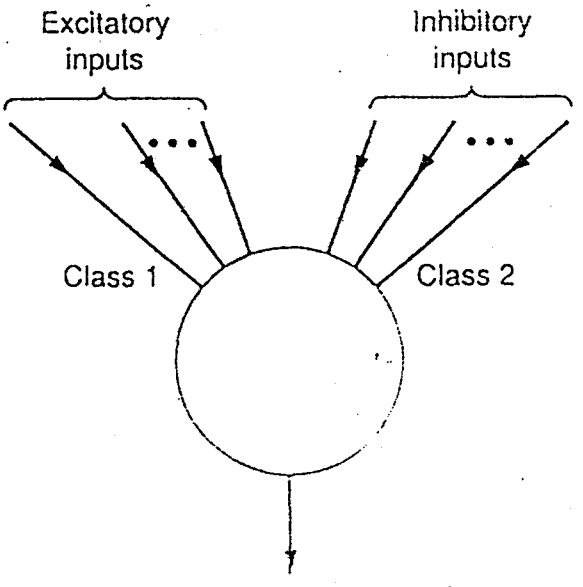
İşlem elemanının tranfer fonksiyonu gelen bütün giriş işaretleri için tanımlıdır.Diğer taraftan bazen değişik katmanlardan gelen işaret tiplerinin ve katman davranışlarının farklı olması doğaldır.Önemli olan nokta işaretlerinin hangi bölgelerden geldiğinin bilinmesidir. Değişik bölgelere göre işaretlerin sınıflarını tanımlayabiliriz.Sıklıkla karşılaşılan giriş işareti sınıflama tipi on-center/off-surround (merkezde evet / çevrede hayır) olarak gösterilebilir.Şekil I.6 bu yapıyı göstermektedir.



Şekil I.6 Merkezde evet/çevrede hayır sınıflama tipi.

İşlem elemanı tetikleyici girişlerini kendine yakın komşu girişlerden ,yasaklanan girişlerini daha uzaktan alır.

Böylece işlem elemanına gelen girişler sınıflarına göre değerlendirilmiş olur.Tetikleyici bçlgeden gelen girişler yasaklanan sınıfı oluşturur.Şekil I.7 böyle bir işlem elemanını gösterir.



Şekil I.7 Tetikleyici ve yasaklanan girişlere sahip bir işlem elemanı

I.2.4.BAĞLANTI GEOMETRİLERİ

Bağlantıların nereden başlayıp nereye gittiğini bilmemiz gereklidir.Böylece bir bağlantı diyagramı çizebiliriz.

1'den N'ye kadar olan bir işlem elemanı kümesinin bağlantıları NxN boyutlu bir matris ile gösterilebilir.

$$M = m_{ij}$$

Burada N_2 bağlantısı vardır.Ayrıca ;

$m_{ij} = 1$, i elemanından j elemanına bağlantı var
$m_{ij} = 0$, i elemanından j elemanına bağlantı yok

demektir.Geometrik bir bağlantı yaklaşımı söz konusu olabilir.Burada bağlantılar çeşitli geometrik bölgeler arasında demetler halinde düşünülebilir.

Tanım 1 :YNA katmanının geometrisi (G,Θ) çifti ile gösterilir.N noktalı G setinin Θ setine birebir dönüşümü gözönüne alınır.Bütün bu işlemler N boyutlu Öklid uzayında düşünülmelidir.

YNA ağlarının geometrisini her bir katmanın

geometrisinden oluşur.

Tanım 2 :YNA'nın arakesidi $i=n_1+k_r$ olarak tanımlanır .

n_1 : Başlangıç indeksi

n_2 : Son eleman indeksi olmak üzere

$1 \leq n_1 \leq i \leq n_2 \leq N$ olduğu örnek olarak söylenebilir. "r" bir tamsayıdır ve k=0,1,2,3 olarak değişir.

Tanım 3 :YNA arakesitlerinin farklı bileşimi söz konusu olabilir.

Tanım 4 :Bağlantı demetlerişu kurallara uymak zorundadır.

1.Bağ demeti oluşturan işlem elemanları aynı bölgeden çıkmalıdır.

2.Bağlantı demetinin işaretleri aynı matematik tipten olmalıdır.

3.Bağlantı demetinin işaretleri aynı sınıftan olmalıdır.

4.Bağlantı demetinin bir seçim fonksiyonu (Φ) olmalıdır.

$\Phi = T \dashrightarrow 2^S$ T :Hedef bölgesi

S :Kaynak bölgesi

Seçim fonksiyonları için çeşitli tipler vardır.Bazı bilinenler ;

. Tam seçim (full) :Hedef bölgesindeki her işlem elemanı, kaynak bölgesindeki her elemana gider.

. Düzgün - Dağılmış olasıl (uniformly random) :Her hedef bölgesi elemanı N kaynak bölgesi elemanına bağlıdır.

. Bire - bir (one-to-one) : Bir hedef bölgesi elemanı bir kaynak bölgesi elemanına bağlıdır.

I.3.ÖĞRENME KURALLARI

Bu bölümde YNA ile ilgili beş öğrenme kuralı üzerinde durulacaktır .Eşitliklerde gösterilen bu kurallar transfer fonksiyonu ve yerel bellek için kullanılırlar.Çeşitli YNA yapılarına ve ele alınan problemlerin özelliklerine göre bu kurallar seçilerek uygulanırlar.

I.3.1. TANIMLAR

Öğrenme kuralları kendi başlarına büyük bir anlam ifade etmeyeblirler.Ancak enformasyon bölgesi ve eğitme yöntemiyle bir değer kazanırlar.

I.3.1.1.Enformasyon Bölgeleri:

Her öğrenme kuralı için bir enformasyon bölgesi vardır. Eğer öğrenme kuralını enformasyon bölgesine bağlı olarak düşünürsek , bir veya birden fazla sayıda giriş işareti sınıfından işaretlerle,nöral ağ işlem elemanları eğitilir. YNA dağılmış parametreli ve paralel bir sistem olduğundan ,her bir işlem elemanı izole edilmiş bir oda gibi düşünülebilir.İşlem elemanının yapacağı operasyon ,girişine uygulanan işaretleri transfer fonksiyonmuna göre değerlendirip bir çıkış üretmektir.Bu bahsedilen transfer fonksiyonlarının en çok kullanılanı "Sigmoid" transfer fonksiyonudur.

$$f(t) = \frac{1}{1 + e^{-t}}$$ Sigmoid transfer fonksiyonu

Sonuçta her işlem elamanı kendisine verilen yerel veriye göre kendisini ayarlayarak ,bütün yapay nöron ağının, enformasyon bölgesini öğrenmesini sağlar. Enformasyon bölgesi olasılık-yoğunluk fonksiyonu ile de tanımlanabilir.

Enformasyon bölgesi bir çok uygulamada ,gerçek değerlerin 0 ila 1 arasında normalize edilmesi gerektir. Normalize etmek gerçek değeri 5 olan bir girişi mesela 0.005 şeklinde ağa uygulamaktır. Normalizasyon aynı anda bütün girişlere uygulanabilir.

I.3.1.2.Ağırlık Uzayı (Weight Space)

Bir çok YNA'nın öğrenme işlevi işlem elemanlarının ağırlığı değiştirilerek sağlanır.Böylece, tanımlayacağımız ağırlık değiştirerek öğrenme de ,iyi bir model kullanıp ağırlıkların bu modele göre değiştirilmesi esastır.

Basit bir matematiksel model olarak her bir işlem

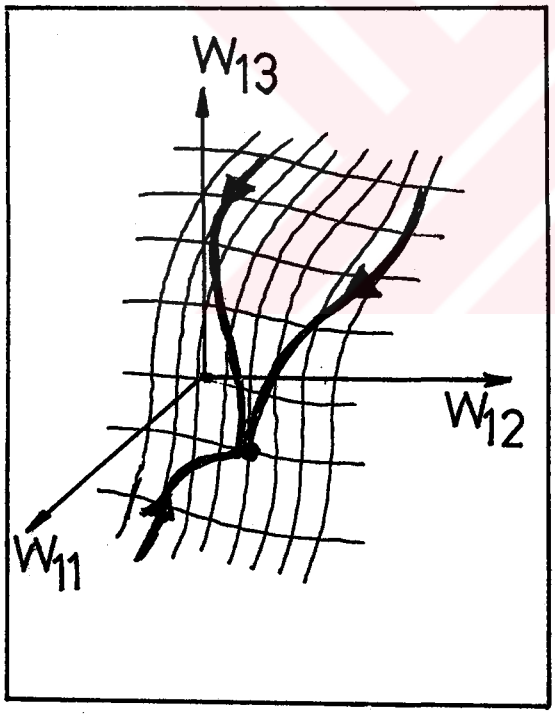
elemanının "n" adet gerçel ağırlığı olduğunu düşünelim ve N adet işlem elemanını gözönüne alalım.

$$w = (w_{11},w_{12},w_{13},\ldots,w_{1n},w_{21},w_{22},w_{23},\ldots,w_{2n},\ldots w_{N1},w_{N2},\ldots,w_{Nn})^T = (w_1^T,w_2^T,\ldots,w_N^T)$$

$w_1,w_2,\ldots,w_N$:işlem elemanlarının ağırlık vektörleridir.

$$w_1 = \begin{bmatrix} w_{11} \\ w_{12} \\ \cdot \\ \cdot \\ w_{1n} \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots \quad w_N = \begin{bmatrix} w_{N1} \\ w_{N2} \\ \cdot \\ \cdot \\ w_{Nn} \end{bmatrix}$$

YNA ağırlık vektörü N n-boyutlu öklid uzayında yayılır. YNA'nın enformasyon işleme performansı ,ağın ağırlık vektörünün belirli bir değeri ile bulunacaktır.



Şekil I.8 böyle bir anlatımı ifade eder. Ağırlık vektörü ile çalışan YNA'da önemli noktalardan birisi bir öğrenme kuralı geliştirip enformasyon bölgesi kullanarak ağırlık vektörü "w" 'yi istenilen YNA performansı verecek noktaya yöneltmektir. Genellikle öğrenme kuralı için bir performans ya da maliyet fonksiyonu tanımlanır.Minimizasyon veya Maximizasyon ile "w" vektörü bulunur.

Şekil I.8 Üç boyutlu uzayda ağırlık vektörü.

I.3.1.3.Eğitme Algoritmaları

Eğitme algoritmaları YNA'nın ayrılmaz bir parçasıdır. Eğitme algoritması eldeki problemin özelliğine göre öğrenme kuralını YNA'na nasıl adapte edeceğimizi belirtir.

Üç çeşit öğrenme algoritması sıklıkla kullanılır.

1).Süpervizör ile eğitme (Supervised training)

2).Skor (grade) ile eğitme (graded training)

3).Kendini düzenleme ile eğitme (self-organization training)

Süpervizör ile eğitmede elimizde doğru örnekler vardır. Sözgelimi $(X_1,X_2,X_3,...,X_N)$ şeklindeki bir giriş vektörünün $(Y_1,Y_2,Y_3,...,Y_N)$ şeklindeki çıkış vektörü ,tam ve doğru olarak bilinmektedir.Her bir (X_1,Y_1) ,(X_2,Y_2),...,(X_N,Y_N) çifti için ağ doğru sonuçları verecek şekilde seçilen bir öğrenme kuralıyla beraber eğitilir.

Skor ile eğitmede giriş işaretlerine karşılık gelen çıkış işaretleri tam olarak bilinmemektedir.Çıkış işareti yerine skor verilir ve ağın değerlendirmesi yapılır. Özellikle kontrol uygulamaları için idealdir. Çeşitli maliyet (cost) fonksiyonları kullanılabilir.

Kendi kendini düzenleyen ağ, giriş işaretine göre kendi kendini organize eder. Olasılık yoğunluk fonksiyonlarına ,sınıflandırma ve form tanıma problemlerine uygulanabilir.

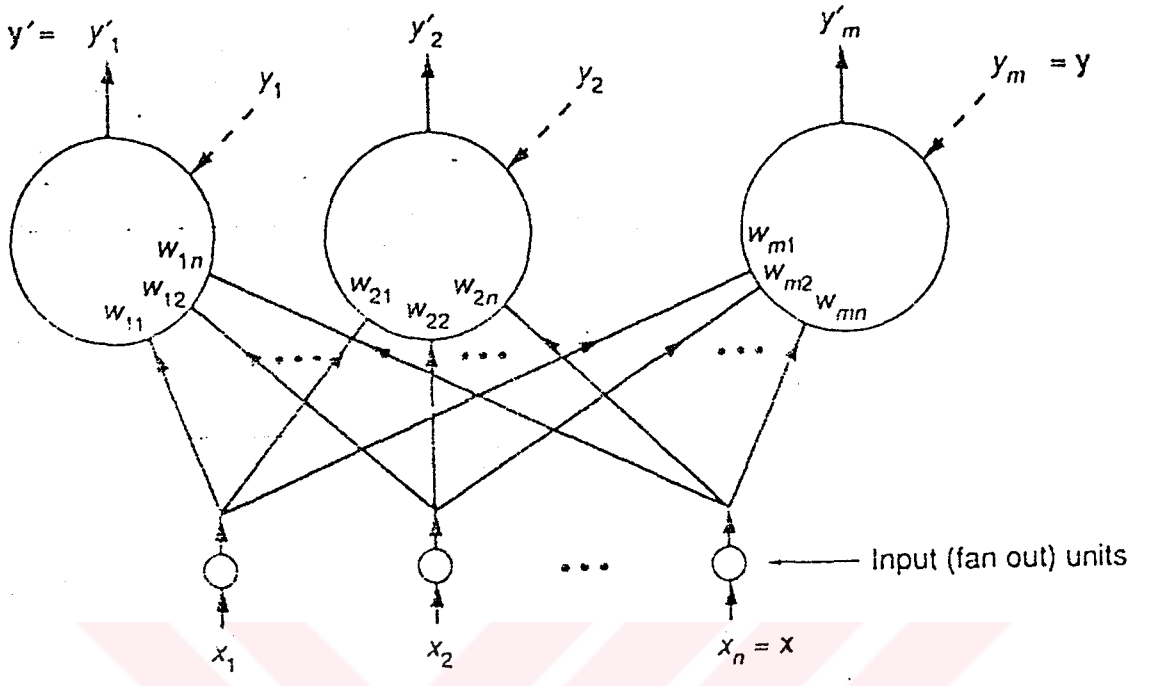
I.3.2.RASTLANTISAL ÖĞRENME

1949'da Kanada'lı fizyolojist Donald Hebb "The Organization and Behaviour" kitabında hücresel seviyede beynin öğrenme mekanizmasını açıklamaya çalışmıştır.Bu bölümde Hebb'in öğrenme kuralının nöral heaplamadaki formu anlatılacaktır.

Hebb'in öne sürdüğü üzre bir nörona dendrid yoluyla gelen bir aksonal giriş onun bir darbe üretmesine neden olur.Sonraki aksonal girişlerin darbe üretme olasılığı artar.

I.3.2.1.Hebb Öğrenme Kuralı ve Lineer Assosiyatör

Çıkış tabakasındaki her bir işlem elemanının çıkışı ,giriş sinyali X'in lineer bir kombinasyonudur.$y_j=y_1,y_2,...,y_n$ girişleri öğrenme boyunca çıkış işlem elemanlarında varolur. Bu yapı Hebb öğrenme kuralını uygulayacaktır. Buradaki y_j'ler bilinen gerçek değerlerdir.



Şekil I.9 Lineer Assosiyatör YNA yapısı.X vektörü ağa en aşağıdan girer ve y' vektörü en üstten alınır.Giriş vektörü ,giriş (fan out) ünitelerine çoğaltılarak çıkış (output) tabakasındaki herbir işlem elemanına uygulanır.

Şekildeki y' çıkışları girişlerin bir lineer kombinasyonu ise ;

$y' = Wx$ yazılabilir. (I.1)

Burada W mxn boyutunda ağırlık matrisidir.

Hebb öğrenme kuralında öğrenme boyunca ağırlıklar aşağıdaki gibi ayarlanır.

$$w_{ij}^{yeni} = w_{ij}^{eski} + y_{ki}x_{kj} \quad (I.2)$$

(y_{ki} :Çıkış tabakası i işlem tabakasındaki y_k değeri)
(x_{kj} :Giriş tabakası j işlem elemanından gelen giriş)

Görüldüğü gibi (x_k,y_k) çiftlerinden w_{ij}'ler elde

edilebilir.I.2 eşitliği vektör olarak gösterilebilir.

$$w^{yeni} = w^{eski} + y_k x_k^T \qquad (I.3)$$

W ağırlık matrisinin başlangıç değerinden başlayıp küçük artmalar ile alması gereken son değer şöyledir:

$$W = y_1 x_1^T + y_2 x_2^T + ... + y_L x_L^T \qquad (I.4)$$

Bu son gösterime Hebb'in dış çarpım kuralı denir.Eğer $(X_1,X_2,X_3,...,X_L)$ vektörleri ortonormal iseler ,bu defa şu yazılabilir:

$$y_k = Wx_k \quad k = 1,2,3,...,L \qquad (I.5)$$

Bunun anlamı, ağın istenen giriş/çıkış transformasyonu gerçekleştirebilmesi demektir.Bu x_k vektörlerinin ortonormal olması yüzündendir. Buradan şu yazılır:

$$x_i.x_j = x_i^T x_j = x_j^T x_i = \delta_{ij} \qquad (I.6)$$

δ_{ij} 'ye kronocker-delta terimi denir.

$\delta_{ij} = 1$ eğer $i=j$

$\delta_{ij} = 0$ eğer $i<>j$

(I.4) ve (I.5) eşitliklerinden;

$$WX_k = (y_1 x_1^T + y_2 x_2^T + y_k x_k^T + ... + y_L x_L^T).X_k$$

$$WX_k = y_k.(X_k^T X_k)$$

$$WX_k = y_k \qquad (I.7)$$

Eğer x_k vektörleri ortonormal değilse sonuçta bir hata bulunur;

$$WX_k = y_k + \mu \qquad (I.8)$$

Bu çoğu zaman karşılaşılan bir sonuçtur. Çünkü n-boyutlu uzayda L tane ortonormal matris bulmak güçtür.

X_k vektörleri ortonormal olmadığı zaman en iyi ağırlık vektörünü bulmak için karesel ortalama hata eşitliğinden Psödo-Ters formülü çıkarılmıştır.

$$F(w) = \frac{1}{L} \sum_{k=1}^{L} | y_k - WX_k |^2 \qquad (I.9)$$

Hebb kuralı $W=YX^T$ idi. I.9 eşitliğini minimum yapan değer $Y=WX$ 'tir.X^+ 'yı X'in pssdo-ters matrisi olarak tanımlayarak ortonormal olmayan giriş vektörleri için yeni performans eşitliği şu şekilde olur:

$$W = X^+Y \qquad (I.10)$$

Her matrisin psödo-ters matrisi bulunabilir. [3] Psödo-ters matris özellikleri şöyle gösterilir :

$$A\,A^+\,A = A$$

$$A^+\,A\,A^+ = A^+$$

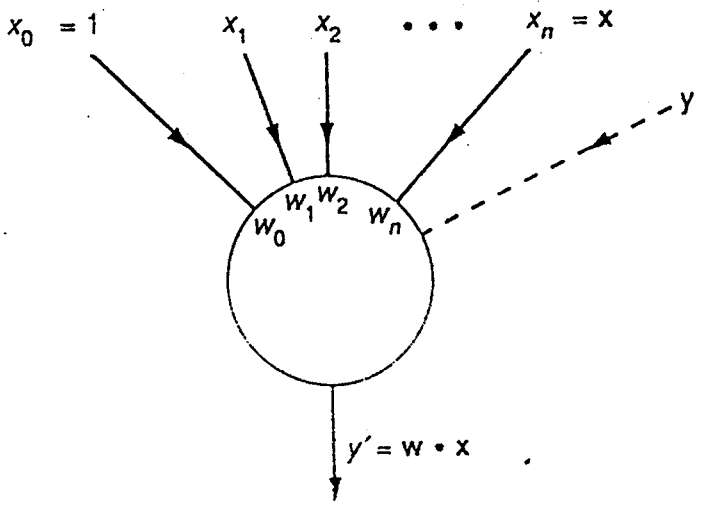
$$A\,A^+ = (A\,A^+)^T$$

$$A^+\,A = (A^+\,A)^T$$

I.3.3.PERFORMANS ÖĞRENME

Bu bölümde anlatılacak öğrenme kuralı ,ağ performansını ölçmek için bazı performans fonksiyonları tanımlamaktadır.Karesel ortalama hata (MES) kriteri ile , en iyi ağırlık vektörü bulunması performans öğrenme kuralının başlıca amacıdır.İşlem elemanı olarak ADALINE (Adaptive Linear Element) kullanılacak ve Widrow tarafından öne sürülen kural tanıtılacaktır.

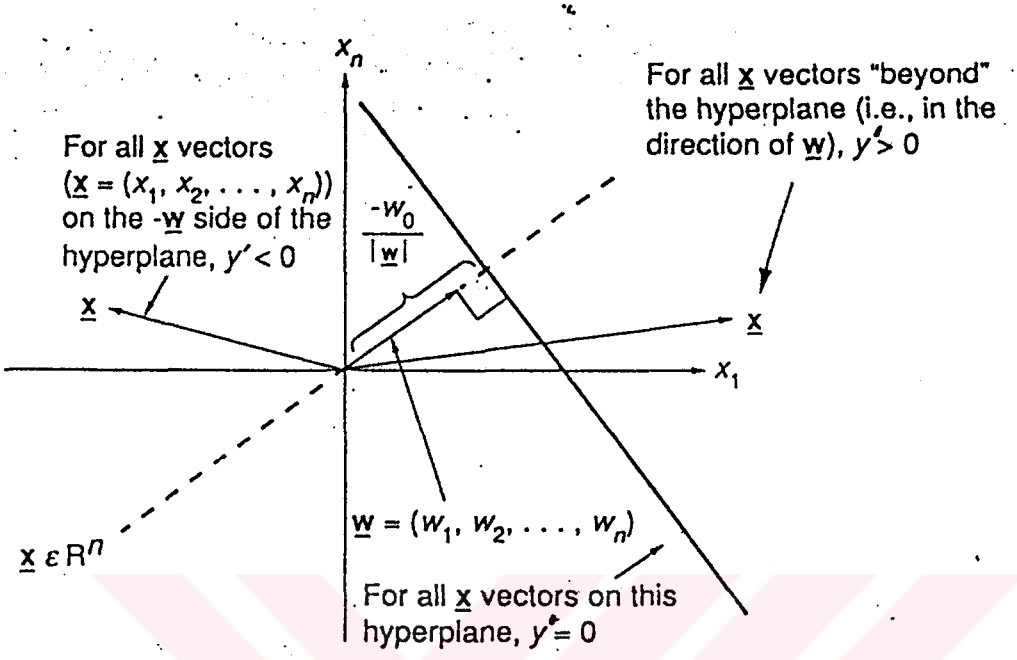
I.3.3.1 ADALINE



Şekil I.10 ADALINE işlem elemanı. Girişlerine $x=(x_0,x_1x_2...,x_n)$ vektörü uygulanır ve çıkışta $y' = w.x = w_0x_0 + w_1x_1 + ... + w_nx_n$ sonucu alınır.

İlke olarak ADALINE işlem elemanına reel X vektörü girer ve sonuçta yine reel y' çıkışı sağlanır.X_0 girişi daima 1'dir ve bias olarak adlandırılır.y girişi ise X girişlerine karşı değerleri kesinlikle bilinen y değerleridir.

Şekil I.11 ADALINE işlem elemanının geometrisini gösterir. Temel fikir $\underline{X} = (x_1,x_2,...,x_n)^T$ vektörlerinin n-boyutlu uzayında , W ağırlık vektörünün bir hiperdüzlem (hyperplane) tanımladığıdır.Madem ki, X giriş vektörü $\underline{X}$ tarafından oluşturulur, çıkış ta $\underline{X}$ 'e bağlı olan bu düzlem tarafından belirlenebilir.Her bir hiperdüzlem y'=sabit bir yüzeydir.Dikkat edilirse bias terimi tanımlanmamış olsa y'=0 düzlemi orjinle çakışır.



Şekil I.11 Giriş/Çıkış ilişkisinde Adaline geometrisi

I.3.3.2 WIDROW Öğrenme Kuralı :

MES kriteri şu formülle ifade edilir :

$$F(w) = \lim_{N \to \infty} \left[\frac{1}{N} \right] \sum_{k=1}^{N} (y_k - y'_k)^2 \qquad (I.11)$$

İstatistikteki E (beklenen değer operatörü ile I.11 denklemi şöyle ifade edilebilir.

$$F(w) = E \left[(y_k - y'_k)^2 \right] \qquad (I.12)$$

Bu sistemin amacı bir modelin bulunması olabilir.y_k işleyişin çıkış işareti ,y'_k ise adaline işlem elemanının çıkış işaretidir.

$$F(w) = E\left[(y_k - y'_k)^2\right]$$

$$= E\left[(y_k - w^T x_k)^2\right]$$

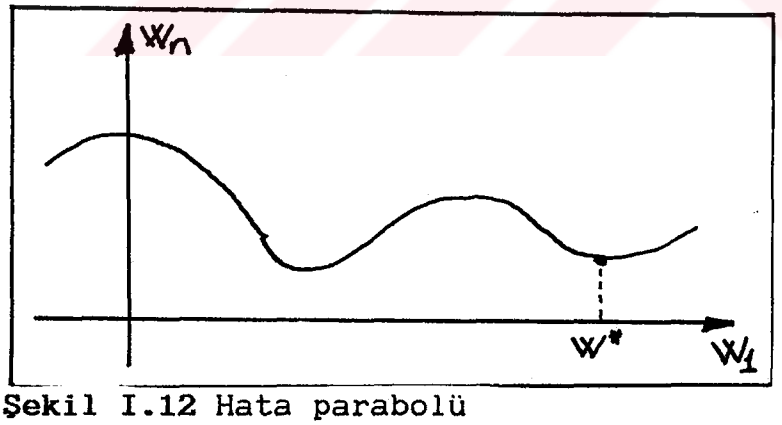
$$= E\left[y_k^2 - 2y_k w^T x_k + w^T x_k x_k^T w\right]$$

$$= E\left[y_k^2\right] - 2.E\left[y_k w^T x_k\right] + E\left[w^T x_k x_k^T w\right]$$

$$= p - 2w^T q + w^T R w \qquad (I.13)$$

Bu formül quadratik bir formüldür ve bir paraboldür ve paraboloid bir yüzey gösterir.Bundan dolayı R matrisi;

$$R = E\left[x_k x_k^T\right] \geq 0 \quad (\text{Pozitif yarı tanımlama})$$



Şekil I.12 Hata parabolü

Şimdi w^* çözümünü arayalım.Performansın w vektörüne göre gradyanını alalım. $Grad_w(F(w))$ operasyonu daima w'yı artıracak şekilde hareket eder.Buradan F(w)'nın gradyanını 0'a eşitlersek bir w değeri bulabiliriz.

$$Grad_w(F(w)) = Grad_w(p - 2w^T q + w^T R w)$$

$$= -2q+2Rw^* = 0$$

$$w^* = R^+q \qquad (I.14)$$

(Burada $R^+=R^{-1}$ demektir)

Genellikle R ve q'yu bulmak düşünülmez.Widrow ve öğrencisinin çözümü daha çok kullanılır.Bu bir nümerik çözümdür.w vektörü w_0'dan başlayıp aşağıya minimum noktasına yani w^*'ya doğru kaydırılmalıdır.

Widrow öğrenme kuralında , -Grad F (Grad F'in ters yönü) en hızlı olarak azalmaya uğraşacaktır.Eğer -Grad F'yi tahmin edebilirsek bu vektörü performans yüzeyinde kaydırarak w^*'ı bulabiliriz.

$$\text{Grad}_w\ F(w) = \text{Grad}\left[\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} (y_k-y'_k)^2\right]$$

$$= \lim_{N\to\infty} \left[\frac{1}{N}\right] \text{Grad}\left[\sum_{k=1}^{N} (y_k-y'_k)^2\right]$$

$$= \lim_{N\to\infty} \left[\frac{1}{N}\right] \sum_{k=1}^{N} \text{Grad}\ (y_k-y'_k)^2$$

$$= \lim_{N\to\infty} \left[\frac{1}{N}\right] \sum_{k=1}^{N} 2\ (y_k-y'_k)\ \text{Grad}\ (-y'_k)$$

$y_k = w^T x_k$ idi. Böylece ;

Grad $(-y'_k)$ = Grad $(-w^T x_k)$ 'dir .Ve,

$(y_k-y'_k)$ ---> δ_k hata olarak düşünülürse

$$\text{Grad}_W F(w) = \lim_{N->\infty} \left[\frac{1}{N}\right] \sum_{k=1}^{N} 2(y_k - y'_k)(-x_k)$$

$$= \lim_{N->\infty} \left[\frac{1}{N}\right] \sum_{k=1}^{N} 2\delta_k x_k \qquad \text{Buradan,}$$

$$\text{Grad}_W F(w) = -2\ E\left[\delta_k x_k\right] \quad \text{elde edilir.}$$

$\delta_k x_k$ terimini bulup -2 ile çarpıp Grad F'yi tahmin edebiliriz.Ve nihayet

$$W_{k+1} = W_k + \alpha\ \delta_k x_k \qquad (I.15)$$

şeklindeki bu kural Widrow-Hoff öğrenme kuralıdır.LMS öğrenme kuralı veya "Delta" kuralı olarak da literatürde yer alır.

Buradaki α 0'dan büyük bir sabittir. Yapılan pratik çalışmalar sonucu $0.01 < \alpha < 10$ olduğu gözlenmiştir.

Uygulamada;

.x_k girişleri toplanarak $\delta_k x_k$ terimi için ortalama bulunur.(Batching operation) Bu yaklaşım F'nin uygun olduğu bir değeri tahmin edildikten sonra tek bir ortalama ile ağırlık değerinin bulunmasını sağlar.

.Momentum yaklaşımı

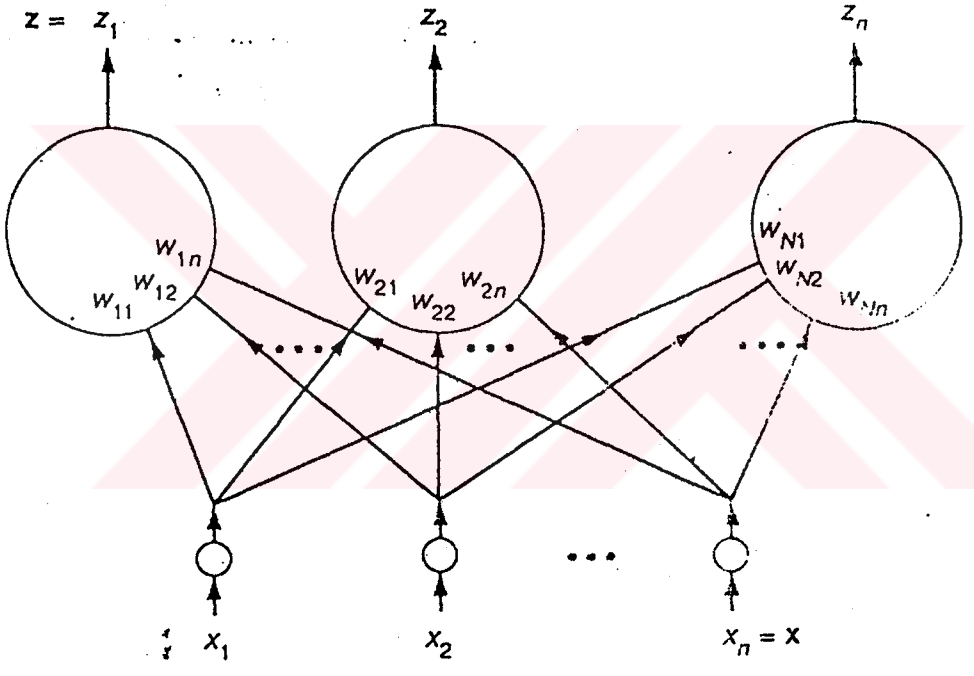
$$W_{k+1} = W_k + \alpha\ (1-\beta)\ \delta_k x_k + \beta\ (W_k - W_{k-1})$$

şeklindedir ve β α'den oldukça büyük seçilir. ($\beta \approx 0.9$)

I.3.4 YARIŞMACI (COMPETITIVE) ÖĞRENME

Yarışmacı kategoriye bağlı öğrenme kurallarının

hepsinde ortak özellik bir yarışma stratejisinin ,işlem elemanlarının tamamına veya bir kısmına uygulanmasıdır. Kohonen öğrenme kuralı,Hebb ve Widrow öğrenmesinden ,kendi kendine düzenleme (self-organization) öğrenme algoritması kullandığı için ayrılır.Buradaki amacımız R^n uzayında bulunan ve p olasılık -yoğunluk fonksiyonu ile dağılan giriş vektörlerine karşı ağırlık vektörlerini düzenlemektir. [4]



Şekil I.13 Kohonen katmanı .Bu tabakadaki N Kohonen işlem elemanının herbirine $x_1,x_2,..,x_n$'e kadar "n" giriş gelir.

Şekilde görülen Kohonen katmanında w_i ağırlık vektörüdür.

$$wi = (w_{i1}, w_{i2}, \ldots, w_{in}) \quad i=1,2,\ldots,N$$

Burada her bir işlem elemanı için "giriş yoğunluğu" kavramı tanımlanır.Sözgelimi x_j girişi işe i işlem elemanı arasında w_{ij} ağırlığı bulunur.

Kohonen işlem elemanı giriş yoğunluğu : $I_i = D(w_i, x)$ (I.16)

Buradaki D(u,v) uzaklık ölçü fonksiyonudur.Değişik uzaklık ölçü fonksiyonları vardır.

Öklid uzaklığı = $d(u,v) = |u-v| = \sqrt{[((u-v)^2)]}$

Hamming uzaklığı = d = İki binary sayının farklı bit sayısı

Yarışma olayının şartı şudur ; Her bir Kohonen işlem elemanı kendi giriş yoğunluğunu hesapladıktan sonra diğerleri ile karşılaştırır.Bir yarışma ile en küçük giriş yoğunluğu olan eleman bulunur ve çıkış işareti "1" alınır. Diğer işlem elemanlarının çıkış işaretleri "0" kalır.Dikkat edilirse bu yarışma sonucu hangi w_i'in x'e yakın olduğu saptanabilir.

I.3.4.1 Kohonen Öğrenme Kuralı

$\underline{x} = (x_1, x_2, x_3, \ldots, x_n)$ girişleri p(x) olasılık yoğunluk fonksiyonu ile n-boyutlu Öklid uzayında dağılmış olsun. Yukarıda anlatıldığı gibi Kohonen katmanındaki i. işlem elemanının yarışmayı kazandığını kabul edelim.Genel bir Kohonen işlem elemanı için öğrenme kuralı ;

$$w_i^{yeni} = w_i^{eski} + \alpha\,(x - w_i^{eski}).z_i \qquad (I.17)$$

şeklindedir. α 0-1 arası sabittir.$(0 < \alpha \leq 1)$

Yarışmayı kazanan i işlem elemanının ağırlık değişimi şöyle olacaktır;

$$w_i^{yeni} = (1-\alpha)\, w_i^{eski} + \alpha\, x \qquad (\text{kazanan } z_i = 1)$$

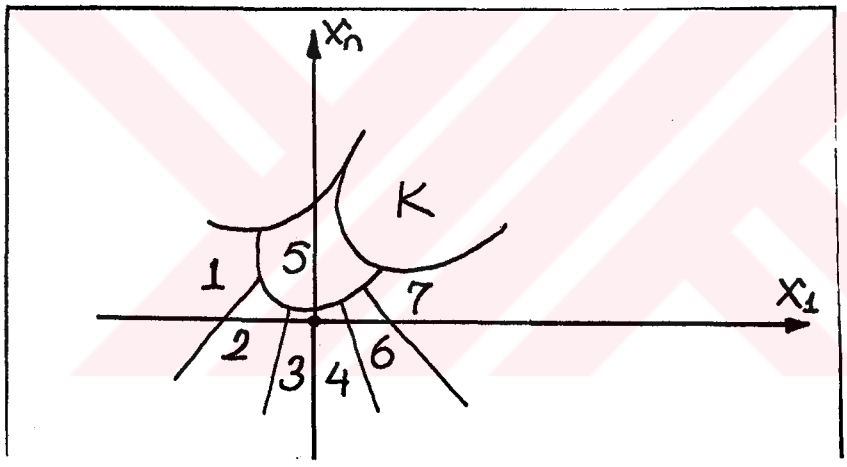
(I.18)

Kaybedenlerin ise ağırlıkları $w_i^{yeni} = w_i^{eski}$ olarak kalacaktır.

Böyle bir yarışmanın amacı ağırlıkları p(x) olasılık yoğunluk fonksiyonu ile dağılan girişlere benzetmeye çalışmaktır. Kohonen katmanının ağırlıkları $\underline{x}$ girişi etrafında bir bulut oluşturacak şekilde dağılmaktadır.

α öğrenmenin başlarında 0.8 sonlarında 0.1 değerini alabilmektedir.

. K-Mean Algoritması



Şekil I.14 w_i ,giriş vektörünü K tane bölgeye ayırır.

Şekilden de görüldüğü gibi w_i vektörüyle giriş vektörü K tane bölgeye ayrılır.Yani giriş vektörünü K tane sınıfa ayırırız.Böylece her bir sınıftaki formu daha kolay tanıma imkanı doğar.Fakat w_i'ın eşit olasılıkla dağılmayan durumları problem oluşturur.w_i ancak aşağıdaki hallerde eşit olasılıklıdır.

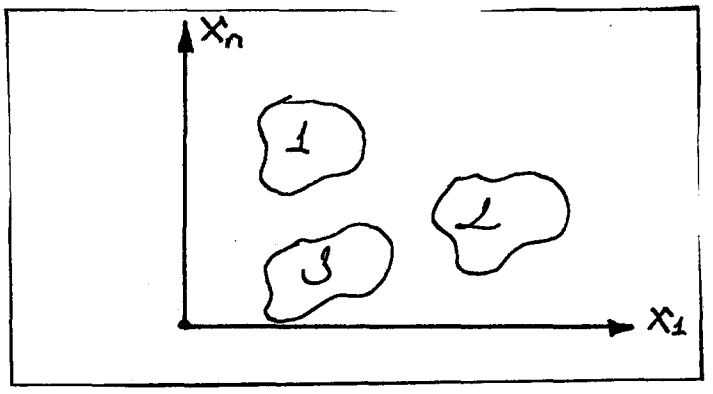
p olasılık yoğunluk fonksiyonu ve x'ler giriş olmak üzere ;

1).Basit bölgede :Herhangi iki nokta arasında bağlantı mümkün olur.

2).p(x) fonksiyonu basit bir bölgede sabit olarak

kalıyorsa...

Eğer sabit bir p(x) yoksa ve bölge basit bir bölge olmayıp birkaç basit bölgeden oluşuyor ve aralarında bağlantı bulunmuyorsa ,giriş vektörünü ağırlık vektörü ile K tane bölgeye ayırmak mümkün olmaz.



Şekil I.15 Aralarında basit bağlantı bulunmayan bölgeler.

I.3.5 FİLTRELEME İLE ÖĞRENME

Bu kategoriye giren öğrenme kurallarının ortak özelliği ,giriş işaretinin bir zaman serisi olarak düşünülerek ,buna ilişkin ağırlığın ,çıkış işaretini de (zaman serisi şeklinde) gözönünde tutarak bulunmasıdır.Bu öğrenmede belirli bir giriş işaretine ilişkin ağırlık ,çarpımsal olarak ortalama aktiviteyi öğrenmeye çalışır.

. Flywheel denklemi ;

Bu matematiksel denklem, boşlukta z açısal hızıyla dönen bir tekerleğin zamana göre davranışını gösterdiğinden bu adla anılır.Bu denklem şöyle ifadesini bulur:

$$Z(t+1) = Z(t) + a\,[\, I(t) - Z(t)\,] \qquad (I.19)$$

[I(t)-Z(t)] kısmına sürücü kısmı denir. a 0 ile 1 arasında bir sabittir.I(t) ise giriş işaretidir.
I.19 denklemi düzenlenirse ;

$$Z(t+1)-bZ(t) = aI(t) \qquad (I.20)$$

Burada b=1-a olduğu malumdur.-aZ terimi dönen tekerlek

için frenleme momentini ,aI terimi ise giriş momentini ifade eder.

Bu denklemin çözümü aranırsa ;

$$Z(t) = I^{-} + \sum_{s=1}^{t-1} b^{t-1-s} \text{ delta } t \qquad (I.21)$$

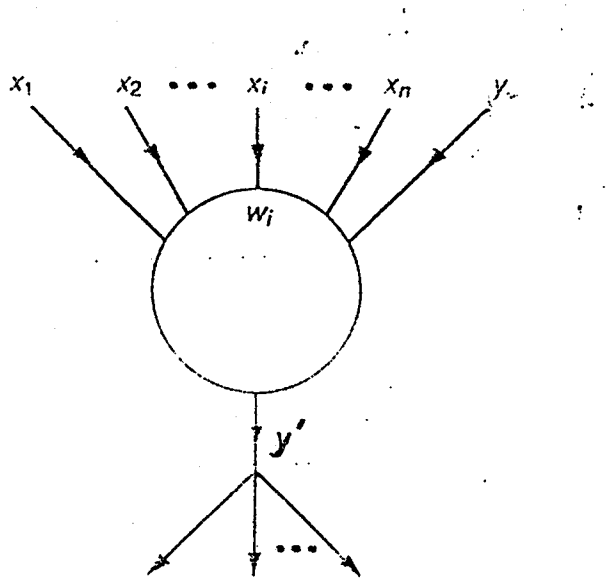
bulunur.Toplamdaki ikinci terim 1/a çok büyük olduğu için 0 olur.Bu halde flywheel denklemi ;

$Z(t)=I^{-}$ şekline dönüşür.Bu sonuç az sonra açıklanacak Grossberg çıkarımına çok benzer.

I.3.5.1 Grossberg Öğrenme Kuralı :

Şekil (I.16) 'de filtreleme ile öğrenmede kullanılan bir işlem elemanı görülmektedir.

Grossberg öğrenmesi beynin öğrenmesine yakın bir işleyiş arzeder.Şekildeki X_i uyarıları şartlı uyarma adını alır ve beynin çeşitli alanlarından gelen işaretlere benzetilebilir. y ise şartsız uyarma yani dışardan gelen sensör uyarılarıdır.



Şekil I.16 Filtreleme ile öğrenme elemanı.

Grossberg öğrenme kuralına göre X_i 0'dan farklı olduğu zaman (aktif) ağırlık w_i ; o anki y girişi ve X_i'nin çarpımsal ortalamasını öğrenecektir. [3]

$$w_i^{yeni} = w_i^{eski} + a\,[\,x_i.y - w^{eski}\,].u(x_i) \qquad (I.22)$$

a 0 ile 1 arasında bir katsayıdır.u_s birim adım fonksiyonu olarak şöyle tanımlanır :

$$u(s) = \begin{cases} 1 & s > 0 \\ 0 & \text{diğer haller} \end{cases} \qquad (I.23)$$

$x_i \leqslant 0$ olduğu zaman ise ağırlıklar değişmez.

x_i'ın 0'dan büyük olduğu zamanlar şöyle olacaktır;

$$w_i^{yeni} = w_i^{eski} + a\,[\,x_i.y - w_i^{eski}\,] \qquad (I.24)$$

(I.24) denkleminin Flywheel denklemine ne kadar çok benzediği ortadadır.Uzunca bir eğitim sonunda ;

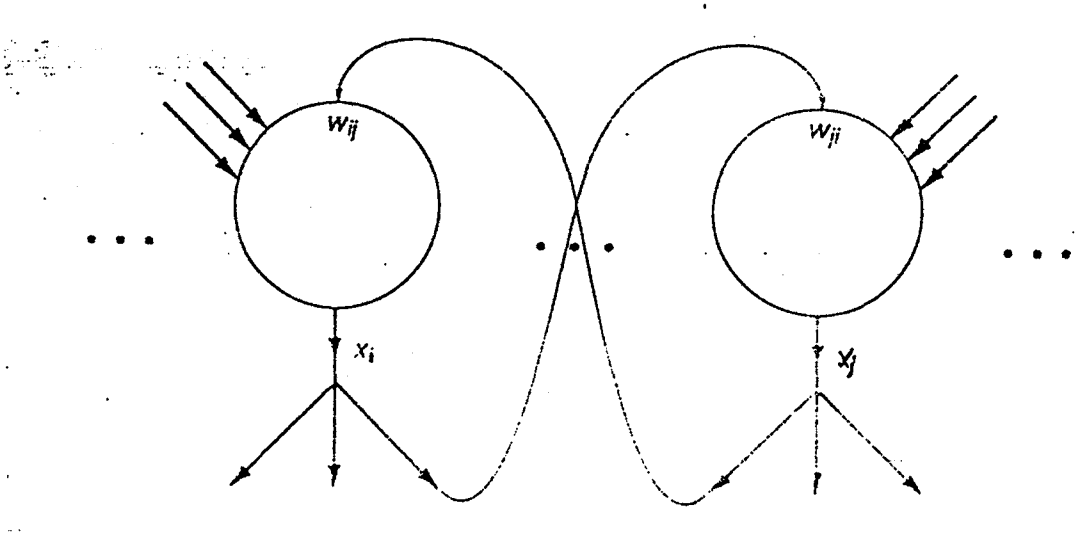
$w_i = X_{iy}$ olacaktır. X_iy terimine zaman ortalaması denir.

(I.24) denklemi flywheel denklemine iyice benzetmek amacı ile şöyle de yazılabilir:

$$w_i(t+1) = w_i(t) + a\,[\,x_i.y - w_i(t)\,] \qquad (I.25)$$

I.3.6 TEMPORAL ÖĞRENME

Bu kategoride ağırlık değerleri ,seçilen işlem elemanı giriş işaretinin kendisi ve türevlerine bağlı olarak saptanır.Söz konusu türevler nedeniyle dinamik koşulları da test etmiş oluruz.Bu öğrenme kuralı birbirinden bağımsız olarak Kosko ve Klopf tarafından 1980'lerde önerildi. İşaretin türevlerinin de kullanılması gerektiğini öne süren bir kuraldır.



Şekil I.17 Temporal öğrenme için işlem elemanı

Bu öğrenmede işlem elemanları için şu kurallar geçerlidir.

. İşlem elemanları negatif olmayan çıkış işaretlerine sahiptirler.

. X_i X_j'den önce aktif olur (X_i'nin aktif olmasından sonra X_j aktif olur)

Kosko/Klopf öğrenme kuralı nöron ağının eğitilmesi sırasında birbiri ardısıra aktif olan işlem elemanları arasında temporal(geçici) bağlantılar oluşturmak fikrine dayanır.

Şekildeki işlem elemanları için w_{ji} ağırlığı arttırılır.Diğeri olan w_{ij} ağırlığı azaltılır.Bu dizi pek çok defalar tekrar edilir ve w_{ji} büyültülür , w_{ij} küçültülür.

Eğitme sırasında X_j'den X_i'ye fazlaca bir geçiş olmayacaktır.Bu tip bağlantılara temporal bağlantılar adı verilir ve yapay nöron ağlarına zamanla değişen bir dinamik sağlarlar.

Temporal aktif olma sırası yaklaşımı ve bağlantılarla YNA'ya bir zamanlama öğretmiş oluruz.

$w_{ij} = w_{ji}$ olursa YNA kendisine ileriye ve geriye doğru

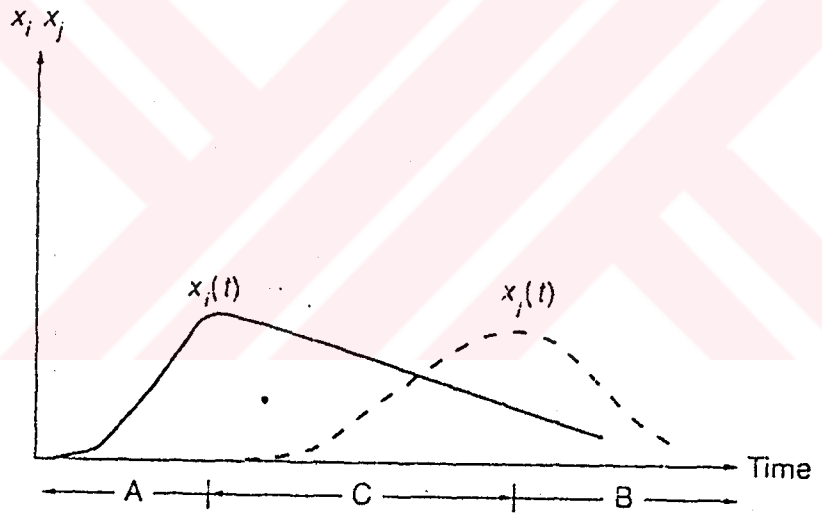
tanıtılan şekle eşit cevap verir.Zamanlamanın önemi kalmaz. Bu istenen bir olay değildir.

Kosko/Klopf öğrenme kuralı şu formülle verilir ;

$$w_{ji} = w_{ji}^{eski} - a\, w_{ji}^{eski} + [-b\, w_{ji}^{eski} + c\, X_j X_i].u(X_j)u(-X_i) \quad (I.26)$$

Burada u birim adım fonksiyonu a,b,c pozitif sabit değerler ve a « b 'dir.$\dot{X}_j$ ve $\dot{X}_i$ X_j ve X_i 'ın değişiminin zaman oranıdır.

"a"'nın amacı artması istenmeyen ağırlıkların 0'a gitmesini sağlamaktır fakat öğrenmede bir etkisi yoktur. Ayrıca ($-bw_{ji}^{eski} + cx_jx_i$) teriminin Grossberg denklemini andırdığı ortadadır.



Şekil I.18 İşlem elemanı aktivasyonlarının temporal sıralaması

A bölgesi :

$$\dot{X}i \geq 0 \qquad u(\dot{X}i)\ u(-\dot{X}j) = 0$$
$$\dot{X}j = 0 \qquad u(\dot{X}j)\ u(-\dot{X}i) = 0$$

B bölgesi :

$$\dot{X}i \leq 0 \qquad u(\dot{X}i)\ u(-\dot{X}j) = 0$$
$$\dot{X}j \leq 0 \qquad u(\dot{X}j)\ u(-\dot{X}i) = 0$$

C bölgesi :

$\dot{X}i < 0 \qquad u(\dot{X}i)\ u(-\dot{X}j) = 0$

$\dot{X}j > 0 \qquad u(\dot{X}j)\ u(-\dot{X}i) = 1$

Böylece işlem elemanı aktivasyonlarına göre Kosko/Klopf formülüyle ağırlıklar ayarlanabilir. [3]

İKİNCİ BÖLÜM

BİYOMEDİKAL PROBLEMLERE NÖRAL AĞ YAKLAŞIMI

II.1 EKG'nin DİJİTAL İZLEMESİ İÇİN BİLGİ SIKIŞTIRMA TEKNİĞİ [5]

EKG'nin 24 saat kaset-kayıt izlemesi olan Holter izleme yöntemi,kardiyak hasarların ve ritm bozukluklarının tespitinde çok faydalıdır.Konvansiyonel Holter sistemi EKG kaydı için manyetik kaset kullanır.Bununla birlikte Holter izleyiciler ,IC hafıza kartlarıyla ,sisteme kayıtta doğruluk,kararlılık ve sağlamlık sağlar.

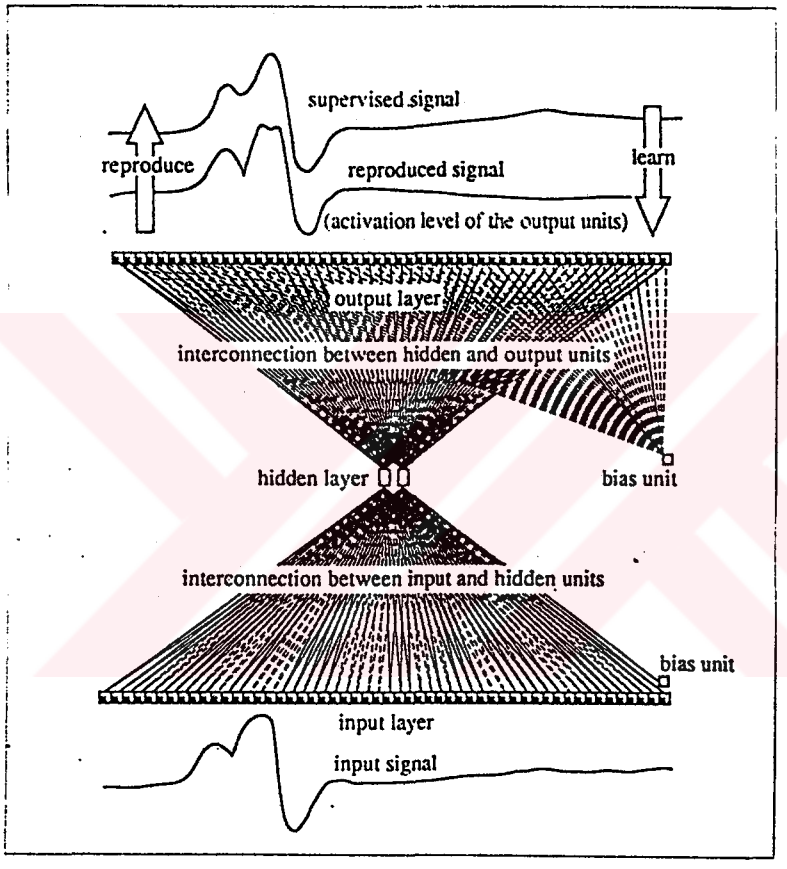
24 saat boyunca iki kanal için dijitize edilmiş EKG bilgilerinin miktarı 20 MB civarındadır ve kullanılan IC hafıza kartı kapassitesi 256 KB'tan 512 KB'a kadar değişir. Bu yüzden Holter sistemi için bilgi sıkıştırılması (Data Compression) gerekir.Ayrıca 2 kanaldan EKG bilgilerinşn 24 saat boyunca 512 KB'lik IC karta depolamak için 1/30 sıkıştırma oranına ihtiyaç vardır.Burada bilgi sıkıştırılmasının algoritması üzerinde durulacak ve yapay YNA kullanımı anlatılacaktır.

Gizli tabakasında birkaç ünite içeren üç tabakalı YNA (Giriş ve çıkış tabakaları arasındaki YNA katmanına gizli veya orta tabaka denir) EKG dalgalarının özelliklerini ,gizli tabaka ,işlem elemanlarındaki (ünite) aktivasyon seviyelerinin fonksiyonu olarak çıkartmakta kullanır.Giriş ve çıkış tabakalarındaki ünite sayıları eşittir.Hatanın geriye iletimi algoritması [6][7] öğrenme için kullanılır. Ağ girişteki sinyalle aynı olan süpervise (denetleme) sinyali ile çalışır.Anlatılan YNA yapısı Şekil II.1'den izlenebilir.

Şekil II.2 iki adet üç tabakalı YNA kullanan Holter izleme sisteminin fonksiyonel blok diyagramını gösterir. Göğüs üzerinden iki elektrodla ölçülen EKG'ler ,AD çevirici tarafından 100 Hz örnekleme frekansı ile örneklenir ve dijitize edilir.

Sayısal sinyal bilgisi buffer ve R-dalga dedektörüne gönderilir.EKG normalde ,kalp atışları ile eşit aralılklı periyodik bir sinyaldir.R dalgası alçak geçiren filtre ile tespit edilir.EKG her kalp atışının parçalarına R-dalga

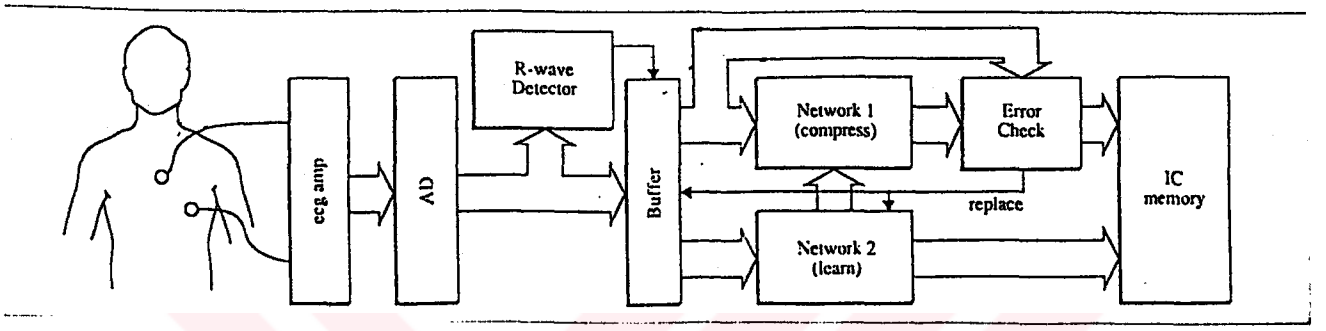
sinyali ile bölünür.En son 30 ardarda gelen sinyal ilk giriş-ilk çıkış (first in-first out) bufferda saklanır. Giriş sinyal parçası ,kalp atış süresi ile bağlantılı 70 örneklemeden oluşur.P ve ST parçaları süresince her iki örnekleme ,yüksek frekans bileşenleri içermediğinden orijinal sinyalden elde edilir.



Şekil II.1 3 tabakalı YNA.Ağ giriş ve çıkışta 70 işlem elemanından (ünite) ,gizli tabakada birkaç işlem elemanından oluşur.Gizli üniteler giriş ve çıkış ünitelerine tam bağlantılı yapıdadır.

Bir çift üç tabakalı YNA,dijital sinyal işleyici içeren YNA işleyicisine uyarlanmıştır. 1.Ağ (Network-1) bilgi sıkıştırması için ve 2.Ağ (Network-2) akım sinyali ile öğrenme için kullanılır.Sıkıştırılan bilgi IC bellekte depo

edilir.Her ağ giriş tabakasında 70 üniteden ,gizli tabakada birkaç üniteden , çıkış tabakasında 70 üniteden oluşur. Gizli üniteler ,giriş ve çıkış ünitelerinin herbirine tam bağlantılı olarak bağlanır.Gizli tabaka ve çıkış tabakası üniteleri "bias" (çıkışı daima 1 olan) üniteye de bağlanır.



Şekil II.2 Bir çift YNA kullanan dijital-Holter izleme sisteminin fonksiyonel blok diyagramı.

Kaydın başında 1.Ağ bufferdaki giriş sinyali ile aynı olan süpervise(denetleyici) sinyal ile forma sokulur.

Ağ bir defa çalışmaya başladığında ,dalga şekillerinin ortak özellikleri ağın bağlantılar arası ağırlıkları ile kodlanır.

W_{jk} ; $j = 1 \rightarrow N_h$; $k = 1 \rightarrow N_O$

Burada N_O çıkış üniteleri sayısıdır.Gizli ünitelerin aktivasyon seviyeleri ;

h_j ; $j = 1 \rightarrow N_h$ şeklindedir.

Burada N_h ,her ardarda gelen kalp atışları için dalga

şekillerinin özelliklerinin vurgulayan gizli ünitelerin sayısıdır.

Madem ki orijinal dalga şekilleri, bağlantılar arası ağırlıklar ve gizli ünitelerdeki aktivasyon seviyesinden yeniden üretilmektedir, sadece ağın bağlantılar arası ağırlıkları ve gizli ünitelerin aktivasyon seviyeleri , her ard arda gelen kalp atışı için bellekte depolanmak zorundadır ve bu işlem orijinal bilgi serisi depolanmasına tercih edilir.

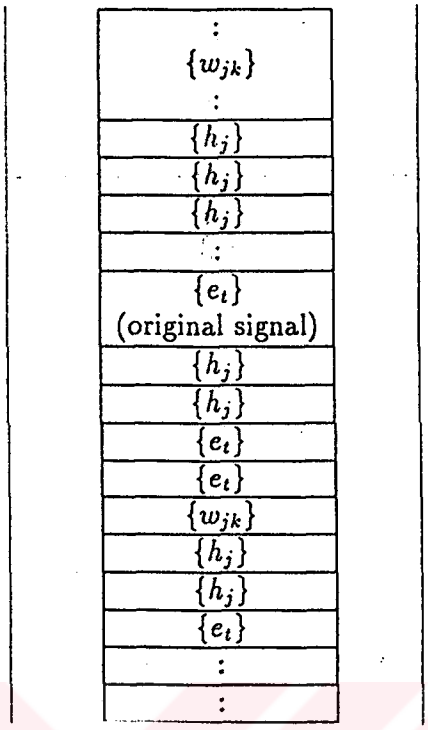
EKG dalga şekli hareket sebebiyle değişebilir.Örneğin; ayakta veya yatma ile... Dalga şekli bir kez değiştiğinde YNA iyi çalışmaz.Bu olay gözönüne alınarak iki tane ağ hazırlanmıştır.EKG dalgası ,öğrenmede kullanılan için olduğu gibi kaldığında Ağ-1 iyi çalışır ve gizli tabaka aktivasyon seviyeleri dalganın özelliklerini temsil eder. Giriş sinyal serisi çıkış ünitelerinin aktivasyon seviyelerinde kesin olarak yeniden üretilir.

Ağın yoğunluğu hata ile izlenir."E" hatası ,giriş sinyali serisi ile çıkış ünitelerinin aktivasyon seviyeleri arasındadır.Ağ-1 akım dalgasına uyduğu sürece ,hata küçük değerini muhafaza eder.Bununla birlikte dalga şekli bir defa değiştiğinde ,hata çıkış ünitesindeki aktivasyon seviyesinin giriş sinyali ile aynı zamanda değişmemesinden ötürü süratle artar.

Başlangıçta ,Ağ-2 Ağ-1 'in aldığı bağlantılar arası ağırlıkların aynılarına sahiptir.Bununla birlikte, Ağ-2 sürekli olarak en son bilgi ile öğrenir ve bağlantılar arası ağırlılarını buna göre uyarlar.Ağ-2 her zaman akım sinyalinşn dalga şekli ile harakete geçer ve sisteme adapte olur.

Şekil II.3 IC belleğe bilgi depolanmasını gösterir. Başlangıçta Ağ-1'in bağlantılar arası ağırlıkları depolanır (w_{jk}) .Sonra gizli ünitelerin aktivasyon seviyeleri (h_j) ,her ard arda gelen kalp atışı için saklanır.Eğer hata (E) belirlenen değeri geçerse orijinal giriş sinyali serisi

(e_t),(h_j)'nin yerine saklanır.

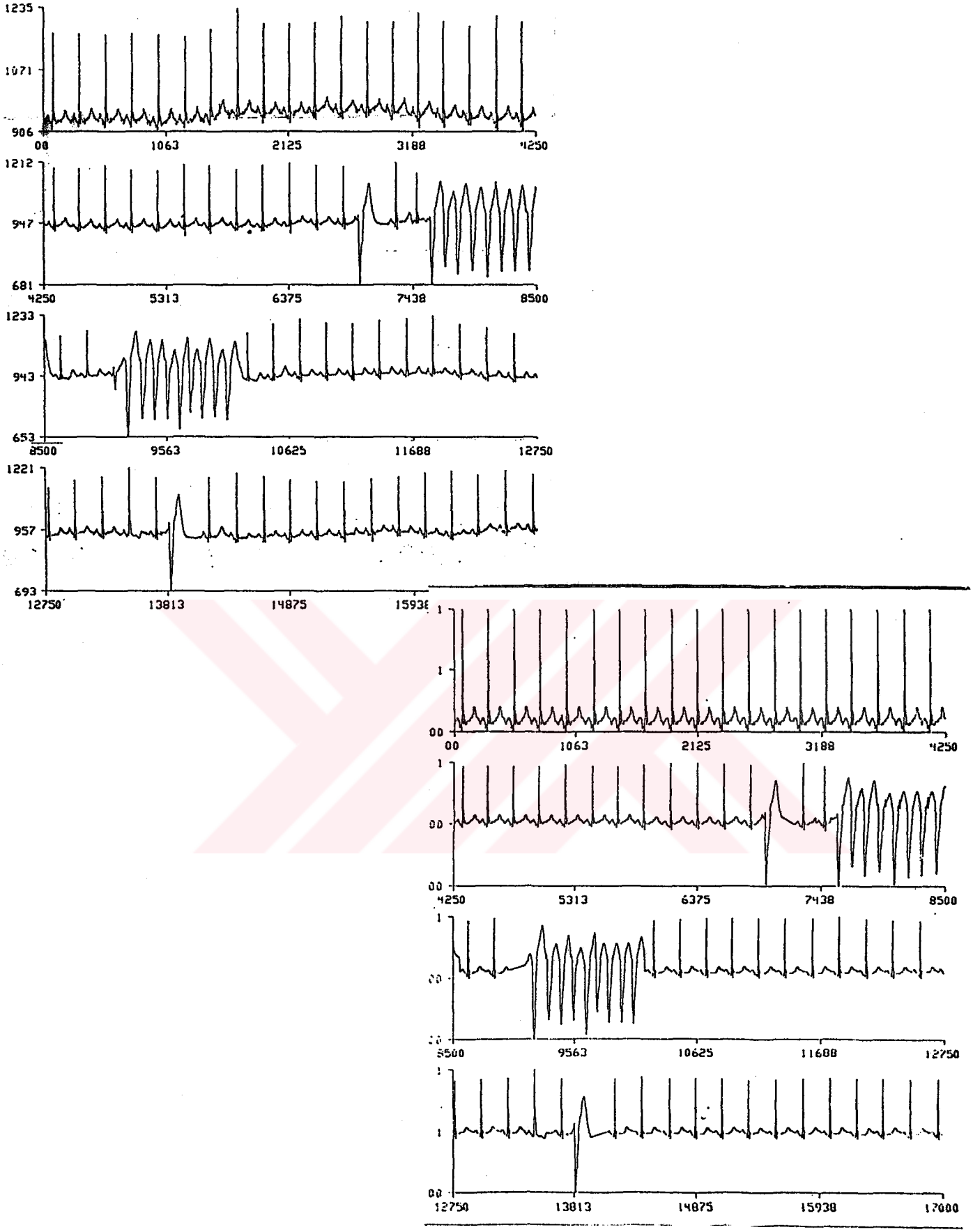


Şekil II.3 Değişik bilgilerin IC kartta depolanması

Eğer hata sık sık belirlenen miktarı aşarsa ,(hata seçimi uygulayıcının inisatifinde) Ağ-1 akım dalga şekline uyamaz.Bu halde Ağ-2 Ağ-1'e kopyalanır ve yeni bağlantılararası ağırlıklar bellekte depolanır.Bu işleyiş ile Ağ-1 bazen akım sinyaline uymak için Ağ-2 ile yer değiştirebilir.

Ağın birçok dalga şekline uyma kabiliyeti gizli tabakadaki ünite sayısına sıkı sıkıya bağlıdır.N gizli üniteli ağ 2^N dalga şekli karakterine kadar iyi çalışabilir. Holter izlemede zamanla bu kadar çok değişen dalga şekli yoktur.Madem ki,bir çift ağ sistemi EKG dalgasının yerine ağ sistemini getirip izleme yapabilmektedir, o zaman iki gizli ünite Holter izleme için yeterli olmaktadır.

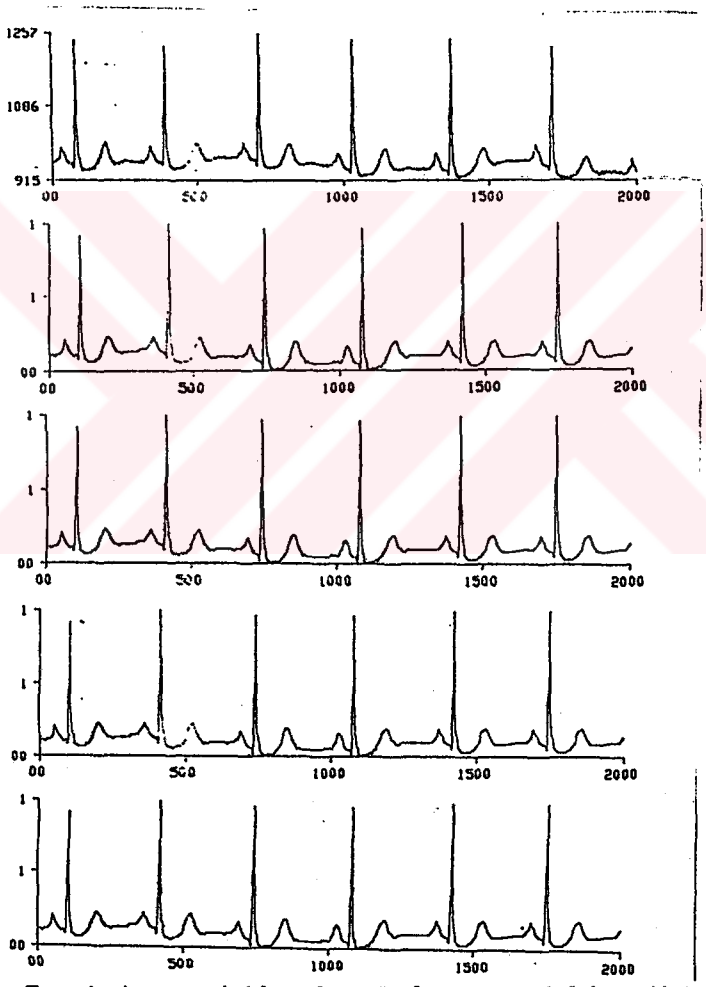
Şekil II.4(a) bir data seti göstermektedir.Bu şekil zamansız ventriküler kasılmanın sebep olduğu anormal dalga şekli değişimini ortaya koyar. Şekil II.4(b) gizli ünitelerin aktivasyon seviyelerinden ve gizli çıkış üniteleri arasındaki bağlantılararası ağırlıkların yeniden



Şekil II.4 Aritmalı bir hastanın EKG'nin orijinal dalga şekli (a).Çıkış ünitelerinin aktivasyon seviyelerinden elde edilen yeniden üretilmiş dalga şekli (Reproduced waveform)(b).

üretilen dalga şeklini gösterir.Yeniden üretilen dalga ile orijinal dalga,temel olarak özdeştir.

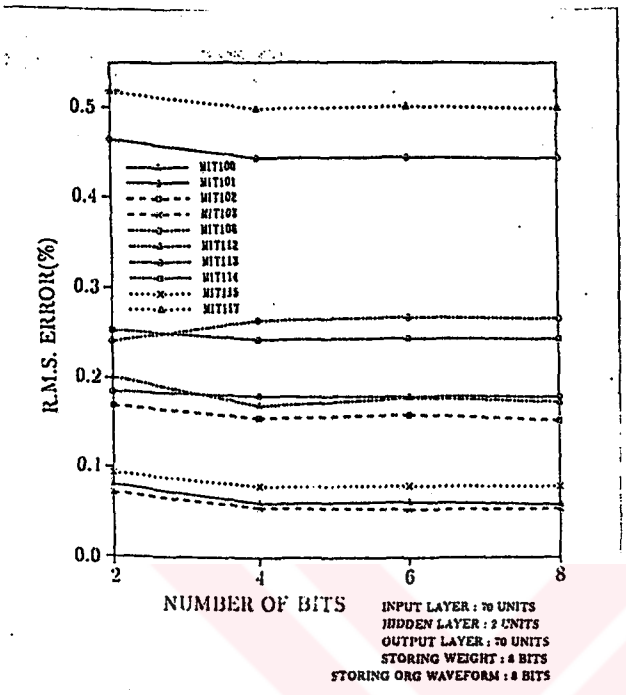
IC kartta depolanan bilgi ondalıklı sayılara ihtiyaç göstermez.Sabit kısımlı tamsayılar orijinal dalganın yeniden üretilmesi için ihtiyacı karşılar.Bilgi depolamak için kaç bit kullanılacağı sorusu araştırılmış ve gizli ünitelerdeki aktivasyon seviyelerinin depolanması için farklı bit sayıları ile gerçekleştirilmiş yeniden üretilen dalga ve orijinal dalga şekilleri Şekil II.5'te gösterilmiştir.Görüldüğü gibi sadece iki bit ile iyi bir dalga şekli elde edilebilir.



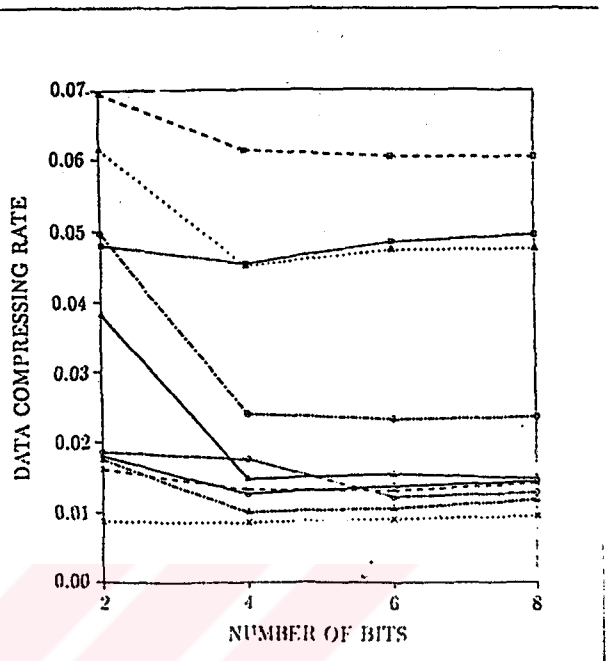
Şekil II.5 (a) orijinal dalga şekli (b)gizli ünite aktivasyon seviyelerini depolamak için 2 bşt kullanılan yeniden üretilmiş dalga (c),(d),(e) sırasıyla 4,6,8 bitlik depolama ,için yeniden üretilmiş dalga şekilleri...

Orijinal ve yeniden üretilen dalgalar arasındaki karesel hata (RMS Root-Means-Square-Error) = $\sqrt{(e_1-e_2)}$ veri

tabanındaki birkaç bilgi seti için incelenirse ,Şekil II.6 ve II.8'den görüleceği gibi iki bit için yapılan hata diğerlerinden çok az büyüktür.



Şekil II.6: *Birkaç bilgi seti için, gizli ünitelerin aktivasyon seviyelerini depolamakta kullanılan çeşitli bit genişlikleri ve RMS hataları arasındaki değişimler.



Şekil II.7: *Birkaç bilgi seti için gizli ünite aktivasyon seviyelerini depolamak için kullanılan bit genişliği ile değişen bilgi sıkıştırma oranı (Data compression rate)

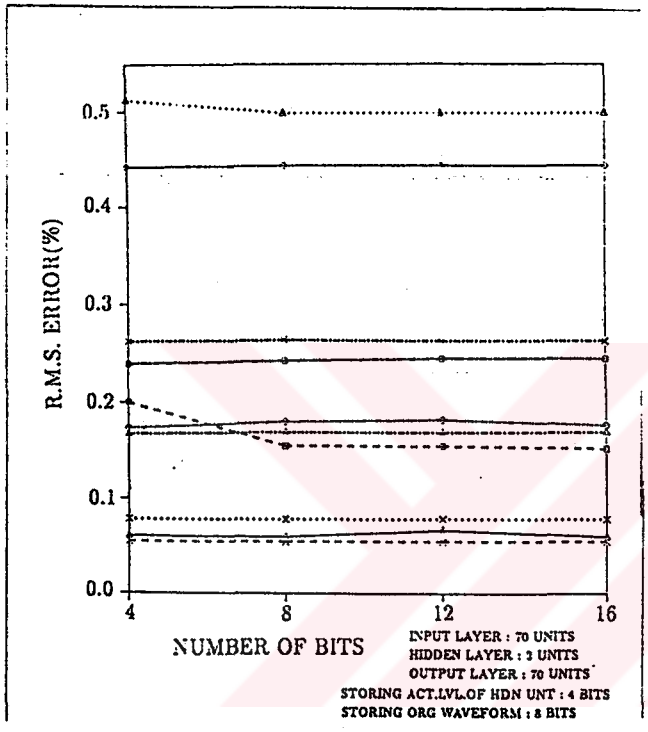
* MIT terimi Massachusetts Institute of Technology - Beth Israel Hospital işbirliği ile alınan veri tabanlarını gösterir.

Şekil II.7 birkaç bilgi seti için gizli ünitelerin aktivasyon seviyelerinin depolamakta kullanılan bit miktarı ile değişen bilgi sıkıştırma oranını gösterir.Bu oran şu şekilde tanımlamır.

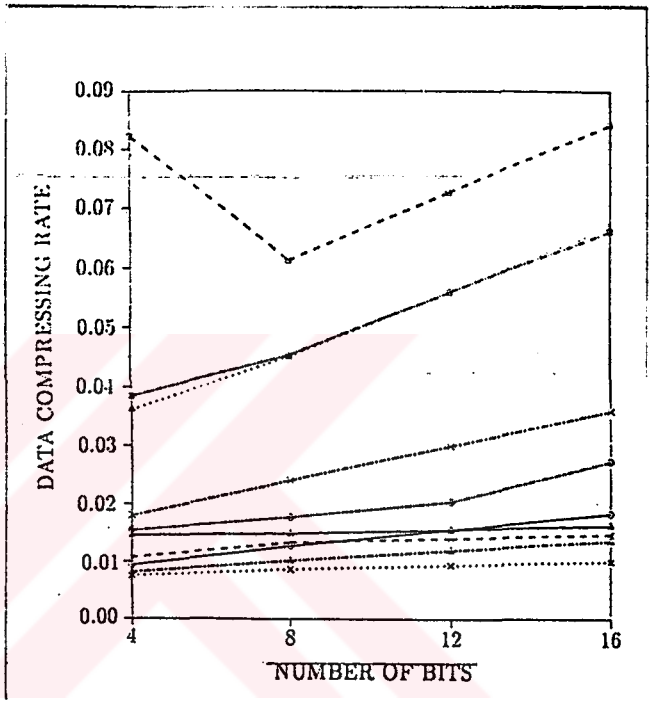
$$DCR = \frac{(\text{Bilgi depolama için bit sayısı})^{**}}{\text{Orijinal bilgi bit sayısı}}$$

** Burada bağlantılar arası ağırlıklar (w_{jk}), gizli ünitelerin aktivasyonları (h_j) ,hata için orijinal giriş serisi ve eşiği aşan hata bu bilgiye dahildir.

2 bitin sıkıştırma oranı diğerlerinden çok daha kötüdür.Bu aktivasyon seviyelerinin saklanması için iki bit miktarının ,dalga şekli ve özelliklerini yeniden çıkarmada yeterli olmamasındandır.Bu yüzden Ağ-1'in bağlantılararası ağırlıkları sık sık Ağ-2 ile yer değiştirir.Bu deneylerden [5] 4 bitin gizli ünitelerinin aktivasyon seviyelerini depolamada optimal bit miktarı olduğu sonucu ortaya çıkarılabilir.



Şekil II.8: Birkaç bilgi seti için gizli ve çıkış üniteleri-ndeki bağlantılar arası a-ğırlıları depolamad kullanılan değişik bit uzunlukları ile değişen RMS hatası.



Şekil II.9: Birkaç bilgi seti için,gizli ve çıkış tabaka-sındaki bağlantılararası a-ğırlıları depolamakta kullanılan bit genişlikleri ve DCR oranının değişimi.

Şekil II.8 orijinal ve yeniden üretilen sinyaller arasındaki hatanın gizli ve çıkış üniteleri arasındaki bağlantılar arası ağırlık depolamasındaki kullanılan bit miktarı ile değişmesini gösterir.Bu bit miktarları için hata miktarlarında büyük farklılıklar olmasına rağmen ,4 bit ile yapılan hata diğerlerinden biraz daha büyüktür.

Şekil II.9 ise bağlantılararası ağırlıkları 8 bit ile depolamanın optimum durumunu gösterir.

Bu çalışma [5] aktivasyon seviyelerini 4 bit ile ,bağlantılararası ağırlıkları 8 bit ile depolama sonucu ;

1/50 , 1/100 sıkıştırma oranlarının % 0.1 ile % 0.5 hata miktarları ile gerçeklenebileceğini gösterir.Bu hata miktarları sistemin klinik amaçlı uygulanması için gayet optimal gözükmektedir.

II.2 EKOKARDİOGRAFİK GÖRÜNTÜLERDEN KARDİYAK HASTALIKLARIN TESPİTİ

Bu bölümün amacı iki boyutlu ekokardiyagrafik görüntülerin analizindeki; hatanın geriye iletmeli nöral algoritmanın faydalarını değerlendirmek ve yapılan bir uygulamayı aktarabilmektir.[8] İki boyutlu ekokardiyografik (2DE) görüntülerden sağlanan gri-skala seviyeleri doğrudan doğruya kardiyak dokusundan gelen yansıma sinyalinin yoğunluğuna bağlıdır. Gri-skala ultrasonografisi görüntü yapısı sağlar ve myokardal dokunun nitelik ve nicelik yönünden analizine izin verir. Araştırmacılar iki boyutlu kardiyografik imajları ortalama gri-seviye hesaplaması [9], gri-seviye histogram istatistiği [10] ve gri-seviye renk kodları [11,12] ile analiz etmişlerdir. Bu çalışmada 2DE bilgileri myokardal dokunun iki ayrı bölgesinden elde edilmiştir. Ventriküler septum (kalpte sağ ve sol karıncıkları ayıran bölge) ve sol sentriküler arka duvarı... Birinci bölge için 6 normal denek (NOR) , anterior ventriküler septum enfaktüslü beş denek (AVS) ve hipertropik (bir organın normalden fazla büyümesi) cardiyopatili (kalp adalesi, akut veya kronik hastalığı) (HCM) yedi denek ile çalışılmıştır.[8] İkinci bölge için beş normal denek ve 6 PW'li (posterior duvarı myokardal enfaktüslü) denek ile çalışılmıştır. Back-propogation nöral devre birinci bilgi seti için normal, AVS ve HCM ; ikinci bilgi seti için normal ve PW' yi ayırmak için kullanılmıştır. Aynı bir devre öğrenme tekniği bilgi setlerinin her ikisini de analiz etmek için oluşturulmuştur.

NA iki temel parçadan oluşur.*Nörodlar ve bağlantılar...

Bağlantılar bir nöronun çıkışından diğerinin girişine tek yönlüdür. Nörodlar NA'da yanlız işlem elemanlarıdır ve fonksiyonları temel olarak beyin nöronlarının aynıdır; girişe bağlı olarak çıkışı hesaplamak...

Nörod işleyişi,bağlantı ağırlığını bağlı olan giriş değeri ile çarpıp toplamak ve bu değeri transfer fonksiyonunda hesaplayıp çıkışa göndermektir.Transfer fonksiyonu bu çalışmada sigmoidal fonksiyondur.

$$\text{Çıkış} = f\left(\sum W_i X_i\right) \qquad \text{(II.1)}$$

Burada f transfer fonksiyonu, $X=(X_1,X_2,X_3,\ldots X_n)$giriş vektörü ve $W=(W_1,W_2,W_3,\ldots W_n)$ bağlantı ağırlık vektörüdür.

Hatanın geriye iletilmeli nöral devre tabakalardaki nörodları organize eder ve bir tabakadan diğer tabakaya giden bağlantıları oluşturur.Aynı tabakadaki farklı nörodlar arasındaki bağlantıya izin verilmez.Genellikle üç tabaka kullanılır.Giriş tabaka, orta tabaka, çıkış tabakası...Ağ öğrenmesi çıkış tabakasında üretilen hataya dayalı olarak bağlantı ağırlıkları ayarlanarak tamamlanır.İlk önce bağlantı ağırlıkları rastgele alınır ve öğrenme bilgisi giriş tabakasında oluşturulur.Öğrenme bilgisi öğrenmesi beklenen ağın bütün karekteristiklerini temsil eden bilgi setleri sayılarıdır.Giriş nörodlar tarafından işlenir ve rastgele çıkış Çıkış tabakasında oluşturulur.Bu Çıkış arzu edilen çıkıştan hata üretmek için çıkarılır.Sonra bağlantı ağırlıkları delta kuralı olarak bilinen formül kullanılarak ayarlanır.

$$W(t+1) = W(t) + BEX/X^2 \qquad \text{(II.2)}$$

E hata değeridir.X giriş vektörü, W(t) akım ağırlık vektörü , W(t+1) ayarlanan ağırlık vektörüdür.Öğrenme sabiti B minimum hataya yakınsama

ölçüsünün göstergesidir.Öğrenme sabiti normalde 0'la 1 arasındadır.Her seferinde delta kuralı uygulanır ve bu yüzden ağırlıklar ayarlanır,nörod çıkışları bilgi setini tanımak için arzu edilen çıkışlara yaklaşmak zorundadır.

Bununla birlikte , delta kuralı hata değerini lokal minimuma indirmesine rağmen global olarak bunu gerçekleştiremeyebilir.Telafi için momentum terimi delta kuralına eklenir.

$$W(t+1) = W(t) + BEX/X^2 + A\,(W(t)-W(t-1)) \quad (II.3)$$

Burada A 0'la 1 arsında bir sabitir ve W(t-1) önceki ağırlık vektörüdür.

Delta kuralı uygulamasında çıkış tabakası açık ve basittir,fakat ağırlıkları ayarlayan orta tabakadaki nörodlar başka bir formüle ihtiyaç duyarlar.

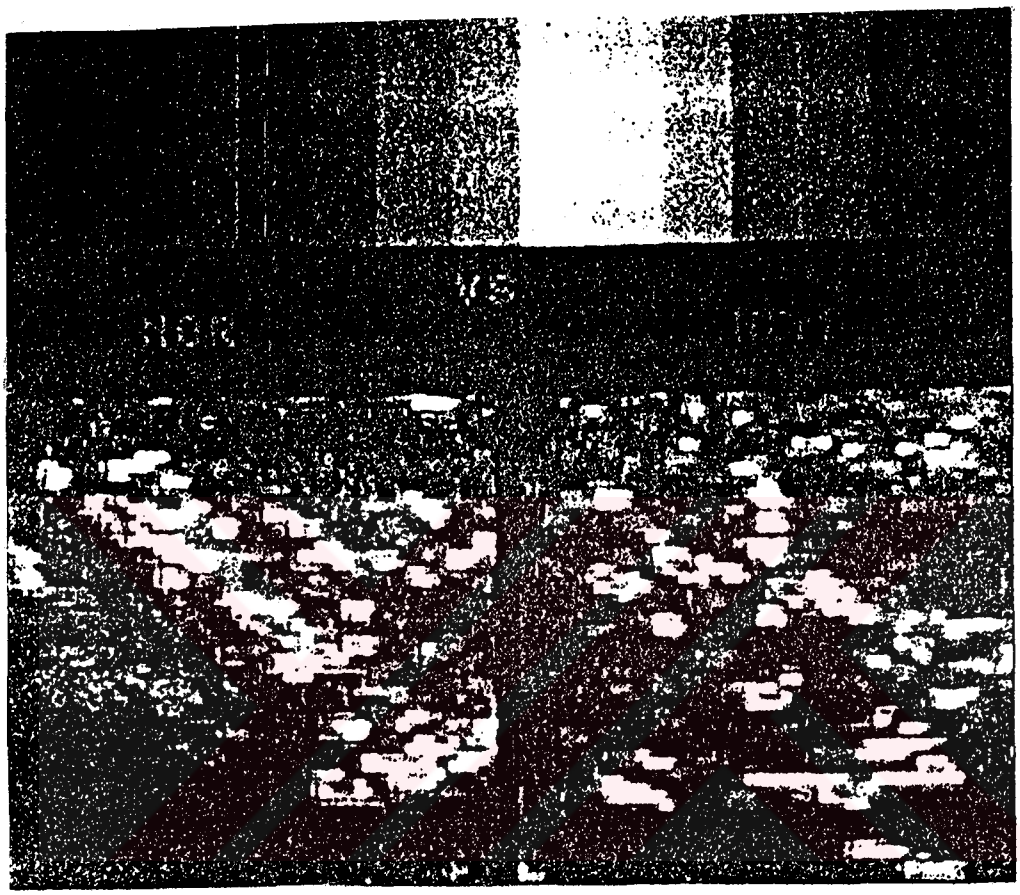
$$e_i = f'(I)(\quad W_{ij}.E_j) \quad (II.4)$$

Burada e_i orta tabaka i nörodundaki hata , f' aktivasyon fonksiyonunun türevi , I orta tabaka i nörodundaki girişler ve W_{ij} çıkış nörodundan orta tabaka nöroduna olan ağırlıklardır.

Bu çalışmada [8] 11 normal , 7 Hypertopik kardiyopati ve 11 myokardal enfaktüslü hasta kullanılmıştır.Elde edilen 2DE imajlar 256*256 pixellik matrisle dijital forma dönüştürülüp 256 olabilir gri-skala seviyesine indirgenmiştir.Sadece EKG'nin T-dalgası sonuna bağlı olan son-sistolik imajlar kullanılmıştır.Gri seviyeler nöral ağın gerektirdiği şekilde 0'la 1 arasında normalize edilmiştir.Şekil II.10 ve II.11 2DE gri seviyelerin normal ve HCM (ventriküler septum),normal ve myokardial enfaktüslü PW'deki farklılıklarını gösterir.

Her bilgi seti için bir obje test etmek diğerleri öğretmek için kullanılır.Nesnelerin sayısı az olduğundan

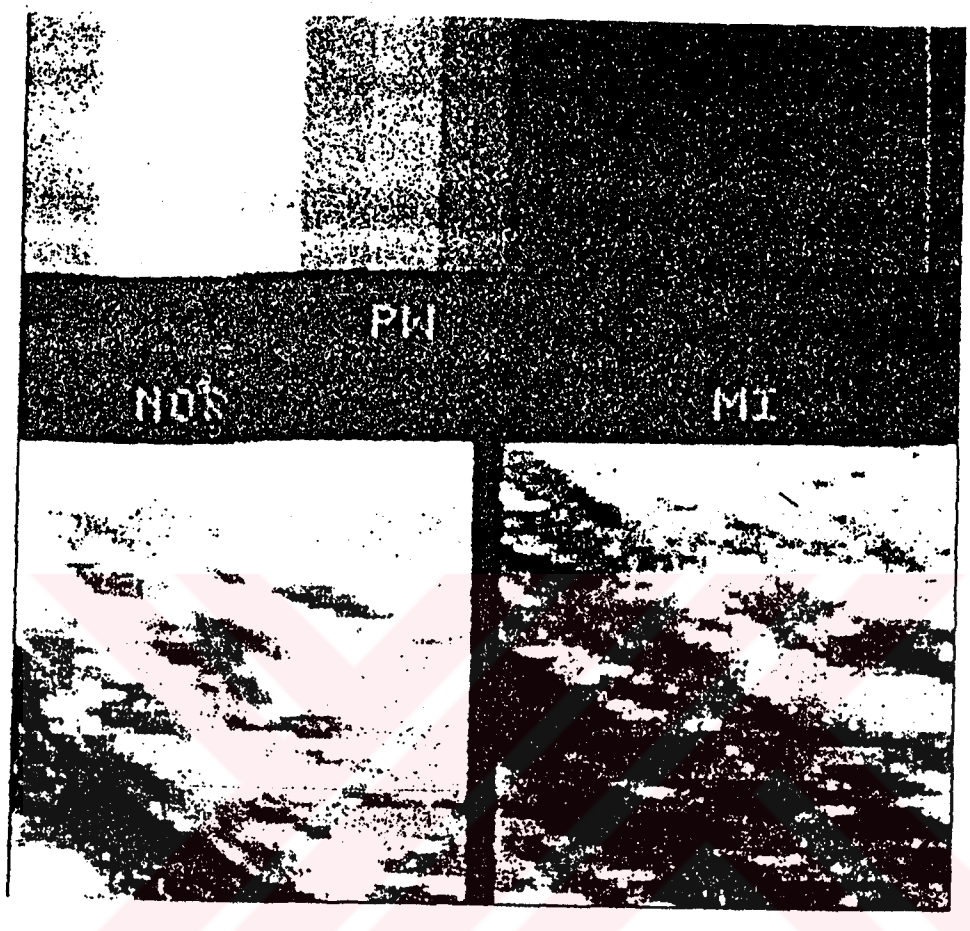
analiz , öğretme ve test grupların her biri için ayrı ayrı yapılır.Ör; 5 normal 6PW'den oluşan bilgi seti kullanılırsa sonucu elde etmek için 11 defa analiz yapmak gerekir.Her seferinde sırayla olmak üzere data setlerinin bir tanesi test diğerleri analiz için kullanılır.



Şekil II.10 Normal ve HCM'nin ventriküler septumdaki görünümleri

Hatanın geriye yayılmalı NA'nın dizaynı büyük ölçüde data setleri ile belirlenir.Girişteki nörodların sayısı giriş parametrelerinin sayısı ile sabitlenir.Bu çalışmada ilgilenilen bölgenin matrisi 100 pixel içerdiğinden giriş tabakasında 100 nörod vardır.Çıkış tabakası ise arzu edilen bilgi sınıflaması ile belirlenir.Bundan dolayı ventriküler septum bölgesi için çıkış tabakasında normal,HCM ve AVS sınıflarına göre 3 nörod bulunur.Sol ventriküler posterior duvar için çıkış tabakasında iki nörod vardır.Orta tabaka öğrenme bilgisinin gereklerine ve özelliklerinin sayısı ile değişiklik gösterecektir.Orta tabakadaki nörod sayısının belirlenmesinde belirli bir algoritma olmadığı için deneme

yanılma metodu ile orta tabaka nörod sayısı 36 olarak bulunmuştur.Birden fazla orta tabakalı NA kullanılabilir.Fakat bir tabaka yeterlidir.



Şekil II.11 Normal ve myokardal enfarktüslü sol ventriküler posterior duvarı

Bu uygulamada hatanın geriye yayılmalı nöral ağın iki fatklı tipi kullanılmiştır.Tam bağlantılı ve örneklemeli... Tam bağlantılı ağda her bir tabakadaki nörodların çıkışı gelecek tabakadai bütün nörodların girişine bağlıdır.Örneklemeli ağ bağlantı bölgeselliği kullanarak tabakalar arasındaki bağlantı sayısını azaltır.Belli bir çevrede tanımlanmış nörodlar birbirine bağlanır.İki boyutlu görüntü bilgisi kullanıldığında bu teknik etkili olarak dikdörtgensel bölgelere ayırır.

Bu bölgeleri bir sonraki nörod tabakasına gönderir. Örneklemenin büyüklüğü iki parametre ile kontrol

edilir.Örneklenen dikdörtgensel alanın genişliği ve bu bölgelerin kesişme alanının miktarıdır.Kesişme alanı tabakaya aktarılacak örneklemenin sayısını ve iki komşu alanın arasındaki boşluğu kontrol eder.

Örnek olarak bu çalışmada giriş tabaksında 100 nörod ,orta tabakada 36 nörod kullanılmıştır.Tam bağlantılı ağda bu bağlantıların sayısı 3600 dür.Beşe beş örnekleme ve dörde dört kesişme faktörü ile bağlantı sayısı 900 e indirilir.

Örneklemeli ağ tam bağlantılı ağdan; data setlerindeki artıştan meydana gelen bağlantılar arası patlamaya izin vermediğinden daha üstündür.Bu yüzden proses zamanı azalır.Öğrenme başladığında bütün bağlantı ağırlıkları 0 ile 1 arasında rastgele tayin edilir.Normalize edilmiş 2DE gri seviye değerleri giriş nörodlarında oluşturulur ve ağırlıklar delta kuralı ve momentum terimiyle ayarlanır.Bu uygulamada [8] öğrenme oranı, delta ağırlığı ve öğretme iterasyonu 1,0.5 ve 800 olarak söylenen sıraya göre verilmiştir.Öğrenmeden sonra ağırlıklar sabitlenir ve test datası girişe uygulanır.

Hatanın geriye iletildiği NA öğrenme örneklerinin büyük miktarları ile mükemmel çalışır.Bu çalışmada bilgi setleri küçük ve ele alınan anormallikler tanınması çok zor olanlardır.Bundan dolayı hatalar hariç çalışma makul sonuçlar vermiştir.Özgüllük ve hassasiyet testlerinin sonuçları YNA tiplerinin ve data setlerinin her biri için tablo II.3'de gösterilmiştir.Sonuç olarak tam bağlantılı ağ örneklemeli ağdan daha iyi bir sınıflama performansı sağlamıştır.Mantıklı bir düşünüşle örneklemeli ağ tam bağlantılı ağın bağlantı sayısının azaldığıbir versiyonudur.Bununla birlikte deney sonuçları bir yanlış sınıflama ile (örneklemeli ağ normal,HCM ve AVS için normal objeleri AVS olarak sınıflamıştır) örneklemeli ağ NA işleyişini küçük hata artışları ile azaltan makul bir yöntemdir.

AVS grubunun 3 üyesi ,tam bağlantılı için normal

olarak tanımlanmış ,geriye kalan 2 obje ise HCM olarak sınıflandırılmıştır. Örneklemeli ağ 2 AVS'yi normal olarak ve 3 AVS'yi HCM olarak tanımlamıştır.Bununla birlikte sadece normal ve AVS kombinasyonundan alınan ve öğretilen AVS'lerden bazıları normalden ayrılmışlardır. (Tablo II.1(d)) AVS'nin ayrılmasının güçlüğü öğretme örnekleri çoğaltılarak engellenebilir.Talyum 201 imajlarına dayalı kroner arter bozuklukları böyle çalışmalarla sınıflandırılabilir ve 2DE imajların sınıflandırılması NA kullanılarak klinik olarak uygulanılabilir.[13]

Tablo II.1

a) 2DE nesnelerin açıklanması :

Sınıflama	Nesne sayısı	İlgilenilen bölge
Normal	6	Ventriküler septum
HCM	7	Ventriküler septum
AVS	5	Ventriküler septum
Normal	5	Sol ventriküler Posterior duvarı
PW	6	Sol ventriküler Posterior duvarı

b).Tabakadaki nörodların kombinasyonu ve tabakalar arası bağlantılar :

Sınıf	Giriş tabaka	Orta tabaka	Çıkış tabaka
Nörod sayısı	100	36	2
X*Y boyutu	10*10	6*6	2*1
X*Y formu	5*5	4*6	2*1
X*Y kesişmesi	4*4	2*1	0

c)Tam bağlantılı* ve örneklemeli ağ** için duyarlılık ve özgüllük testi sonuçları :

Sınıf	Duyarlılık	Özgüllük
*HCM	5/7	9/11
*AVS	0/5	11/13
*PW	5/6	5/5
**HCM	5/7	8/11
**AVS	0/5	10/13
**PW	5/6	5/5

d) Tam bağlantılı* ve örneklemeli ağ** için duyarlılık ve özgüllük test sonuçları

Sınıf	Duyarlılık	Özgüllük
*AVS	2/5	6/6
**AVS	3/5	6/6

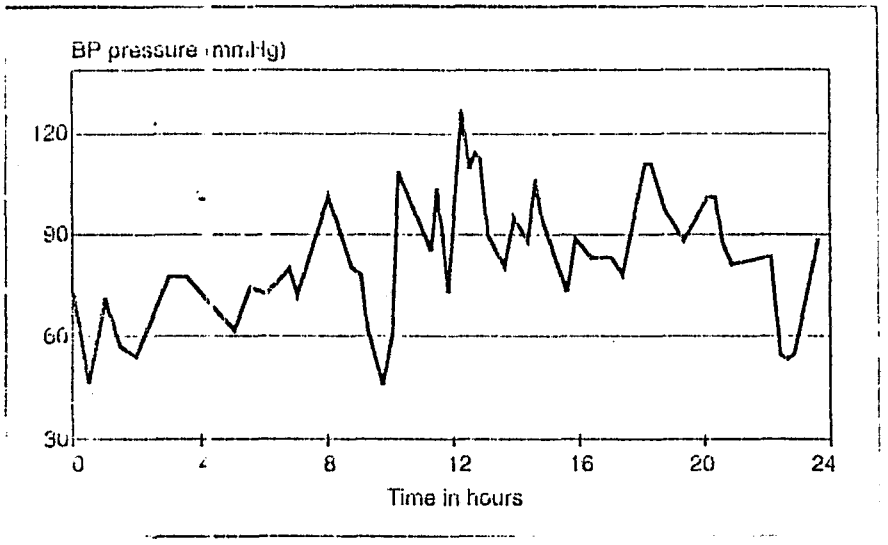
II.3 HİPERTANSİYON TEŞHİS VE TEDAVİSİ

Bu bölümde Floransa Üniversitesi Elektrik-Elektronik fakültesinde gerçekleştirilmiş HYPERNET (Hipertansiyon Nöral Uzman Terapist) sistemi anlatılacaktır.[14]

Hipertansiyon dünya nüfusunun yaklaşık % 15 - % 20 'sinde görülen patolojik bir olaydır.Bu limit olarak kabul edilen 90 mm-Hg basıncının üzerine çıkan sürekli bir diastolik basınç artışını ifade eder.

Bu tanımlama negatif hatta pozitif hata sağlayan teşhissel yaklaşımla bağlantılıdır.Bunlar önemsenmeyecek büyüklükte olan ,ihtiyari ve gün boyunca reaktif kan basıncından sağlanan hatalardır.

Kan basıncı dinamiği ,şekil II.12 'de görüldüğü gibi kan basıncı zaman serilerinin gözden geçirilmesi ile anlaşılabilir.Bu şekilde hipertensif bir hastanın diastolik kan basıncı bilgilerinin değişimi çizilmiştir.



Şekil II.12 Hipertansif bir hastanın diastolik kan basıncı zaman serileri.

Bu dinamizm sağlıklı ve hipertansif hastalar için kan basıncı sınırlarını tanımladığında, zamanla değişen referans limitlerini de içermelidir ki, bu limitler yaşa, cinsiyete bağlı olarak değişir.

İdeal hipotansif tedavisi hipertansiyonun kaynaklarına hareket edecek yaklaşımda olmalıdır.Ama bilinir ki, birçok hipertansif hasta için bilinen bir sebep bulunamamıştır. Bu yüzden tedavi, genellikle kan basıncı değerlerinin kontrolünü mükemmel olarak sağlama ve istenmeyen etkilerin var olmasını önlemeye yönelik olarak seçilir.

Aşırı tansiyon artışlarının giderilmesinde, antihipertansif ,ilaçların zamanlamaları ve dozajları önemlidir. İlaçlar maximum etkilerini en yüksek kan basıncı artışlarında sağlarlar.Bu optimizasyon gün boyunca kullanılan ilaçların dozajlarında da bir azalmaya imkan tanır.

Günümüzde hipertansif tedavide ilk yol olarak kullanılan dört sınıf ilaç vardır.DİÜRETİKLER,BETA-ADRENOCEPTOR-BLOCKING, ANJIOTENSİN-DÖNÜŞTÜRME-ENZİM-İNHİBİTÖRLERİ ve Ca-KANALLARI BLOKAJ İLAÇLARI... Bazı patolojik durumlara göre bir veya birkaçı seçilebilir.Farklı durumlar için farklı ilaçların bileşimleri de hazılanabilir.

Hypernet'te şekil II.13 'de gösterilen bir işlem elemanı (ünite) kullanılmıştır.İşlem elemanına gelen bir i girişi o girişin ağırlığı ile çarpılır.Bu işlem her bir giriş elemanı için yapılırsa şu elde edilir ;

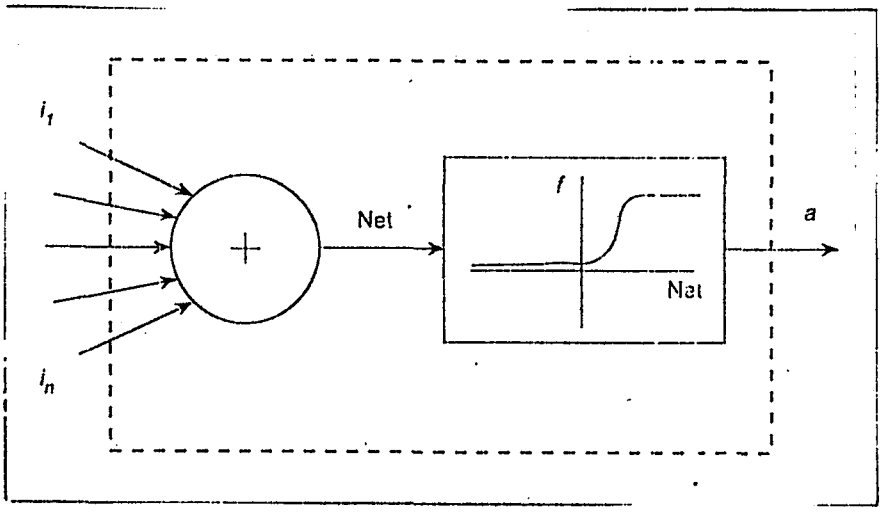
$$Net = \Sigma^{n}_{j=1}\ w_i . i_j$$

Net ifadesi işlem elemanının transfer fonksiyonunda değerlendirilir ve çıkış belirlenir.Burada transfer fonksiyonu sigmoidal fonksiyondur.

$$f(Net) = \frac{1}{1+e^{(-net-bias)}} = a$$

Bias terimi birçok YNA'da kullanılan ve eşik etkisini ifade eden bir parametredir.Sabit olarak da seçilebilir.

Hypernet modüler yapıda bir sistemdir.Her bir modül bir veya birkaç müstakil YNA'dan oluşur.Fakat herbir YNA genellikle üç tabakalıdır (giriş,gizli ve çıkış tabakaları) .Hypernet ,geliştirenler tarafından ileri beslemeli ağ olarak tanımlanmıştır.Bunun anlamı hangi tabaka ve hangi modülde olursa olsun ,her ünite kendi çıkışını çıkış tabakasına en yakın üniteye gönderir.Fakat aynı tabakadaki üniteler arasında bağlantıya izin verilmez.



Şekil II.13 İşlem elemanının (ünite) fonksiyonel diagramı

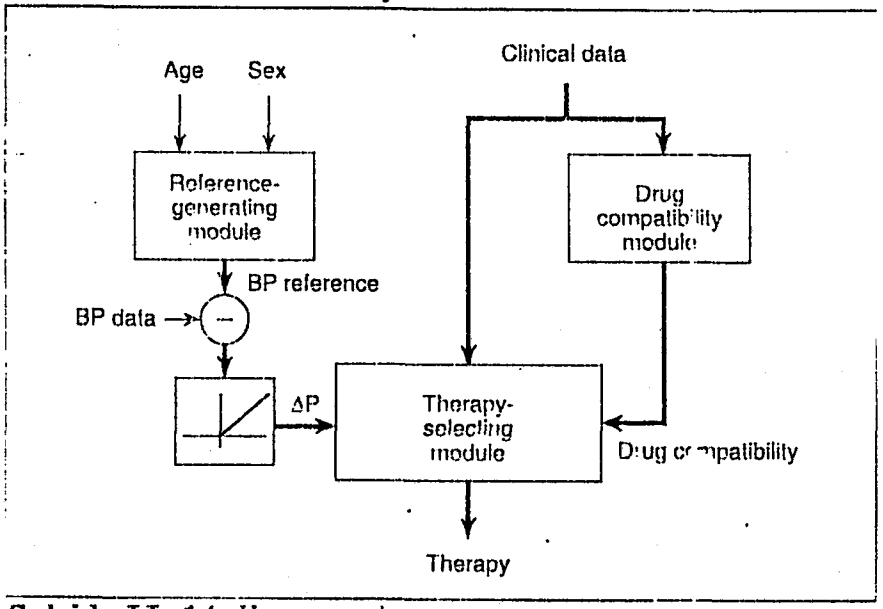
Sistemde eğitme algoritması olarak hatanın geriye iletimi (back-propagation) algoritması kullanımıştır.[6]

Hypernet'te gönderilen girişler ,kişinin 24 saatlik sistolik ve diastolik basınç zaman eğrileridir ve hastalığa ait (anamnestic) bilgileri içerir.Sistem çıkışında 4x24 işlem elemanı vardır ve ilaçların hastaya verilecek saatlik dozajını ayarlar.Normal kişiler için doğal olarak herhangi bir çıkış sözkonusu değildir.

Sistem üç ana modüle ayrılmıştır.Referans üretme modülü (RGM ; Referance-generating-module) , ilaç - uyuşumu modülü (DCM ; Drug-compatibilities-module) ve tedavi seçme modülü (TSM ; Theraphy-selection-module) şeklindeki bu modüller Şekil II.14 'te gösterilmiştir.Her modül bir veya iki tane hatanın geriye iletilmesi algoritmasıyla çalışan YNA içerir.

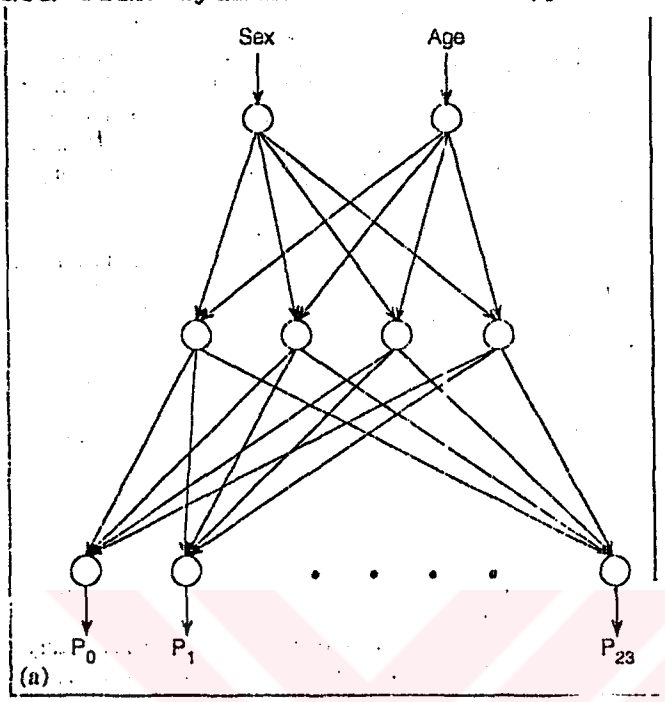
Bu tür modüler bir yapı sistemdeki kullanıcı başarısını arttırır ve işleyiş hakkında açıklama yapan dahili test etme noktaları sağlar.

RGM her biri 2 giriş ,4 gizli ve 24 çıkış ünitesine sahip ,üç tabakalı iki adet YNA içerir.Bu modül şekil II.15 'te gösterimiş olup ,hastanın 24 saatlik kan basıncı zaman serileri ile aynı yaş ve cinsiyetteki normal bir kişinin tipik zaman serilerini karşılaştırarak sistolik ve diastolik basınç artışlarını belirler.

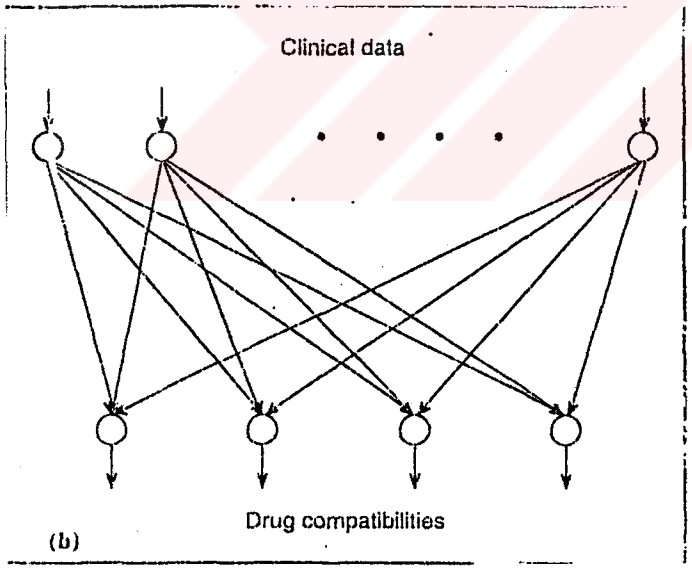


Şekil II.14 Hypernet yapısı

DCM 17 giriş ve 4 çıkış üniteli basit bir YNA'dır. Fonksiyonu hastanın klinik raporlarını analiz edip ,her bir ilaca olan uyumunu bulmaktır.(Şekil II.16)



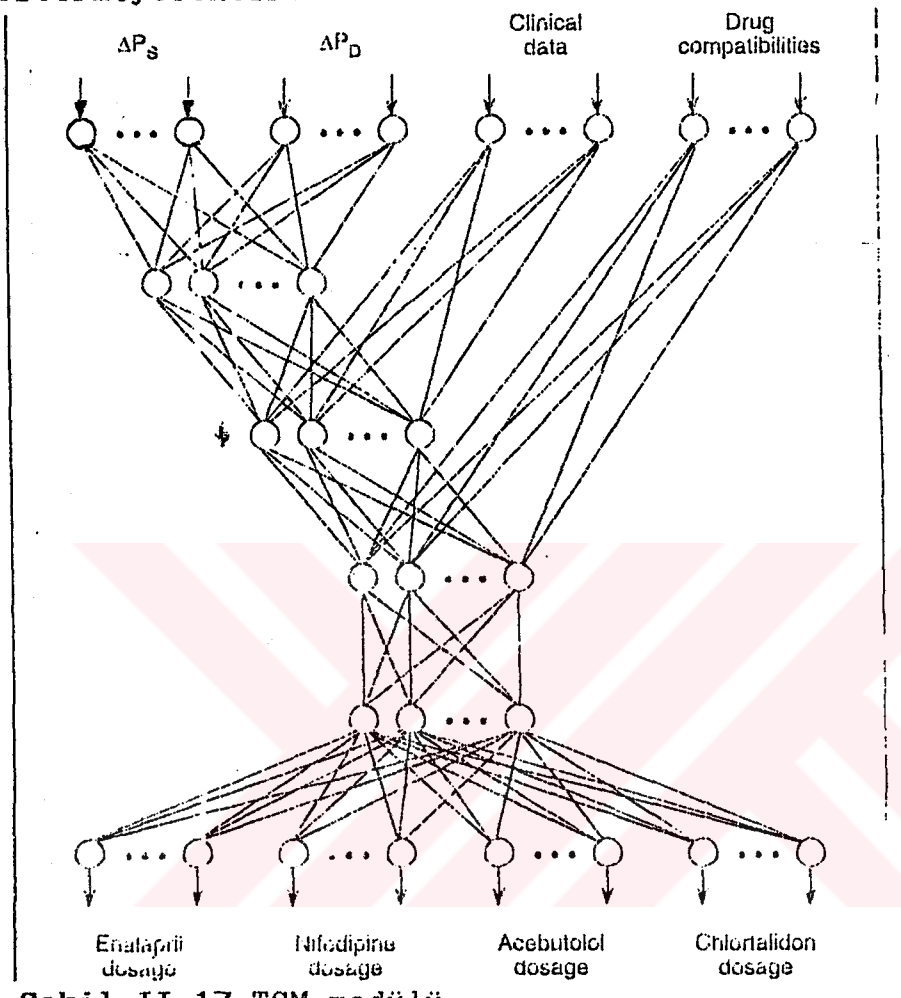
Şekil II.15 RGM Modülü



Şekil II.16 DCM Modülünün yapısı

Şekil II.17'da gösterilen TSM modülü 1.girişleri ilaç uyumları ,2.girişleri referans sistolik ve diastolik kan basınçları ile hasta izlemesinden sağlanan sistolik ve diastolik basınçları arasındaki farklar olan delta P_s,delta P_d 3.girişleri teşhise etki eden beş anemnestik bilgi içeren ,altı tabakalı bir YNA'dır.Bu modülün 96 adet çıkış

ünitesinin aktivasyonları a_{td} ; (t=0,...,23 ve d=1,2,3,4) (24 dozaj değeri x 4 ilaç) bu sistemin işleyişinin son çıktılarıdır.Kişi normal ise sistem hiçbir çıkış göstermeyecektir.



Şekil II.17 TSM modülü

TSM modülünün ilk gizli tabakasında sistolik ve diastolik basınç artışları için 12 ünite bulunur.Bu ünitelerin fonksiyonları sistolik aşırılıkları diastoliklere göre entegre etmek ve ağırlıkları buna göre ayarlamaktadır.Bu işlemin sebebi bu tabakaya hiçbir hastalığa ait bilgi gelmemesidir.

TSM modülünün ikinci tabakasında ,bir önceki gizli tabakayı temsilen 12 ve klinik bilgiler için de 5 ünite vardır.Bu tabaka hipertansiyon hakkında hasta izlemesinin ipuçlarını taşır.

Üçüncü gizli tabaka bir önceki gizli tabakanın

aktivasyonlarının ve DCM ünitesinin çıkışları için 12 ünite içerir.Bu tabaka bir sonraki gizli tabaka ile beraber ve çıkış tabakasıyla birlikte tedaviyi gerçekleştirir.TSM'de bu kadar çok tabaka olmasının sebebi, çözülmesi gereken probleme birçok parametrenin etki etmesidir.

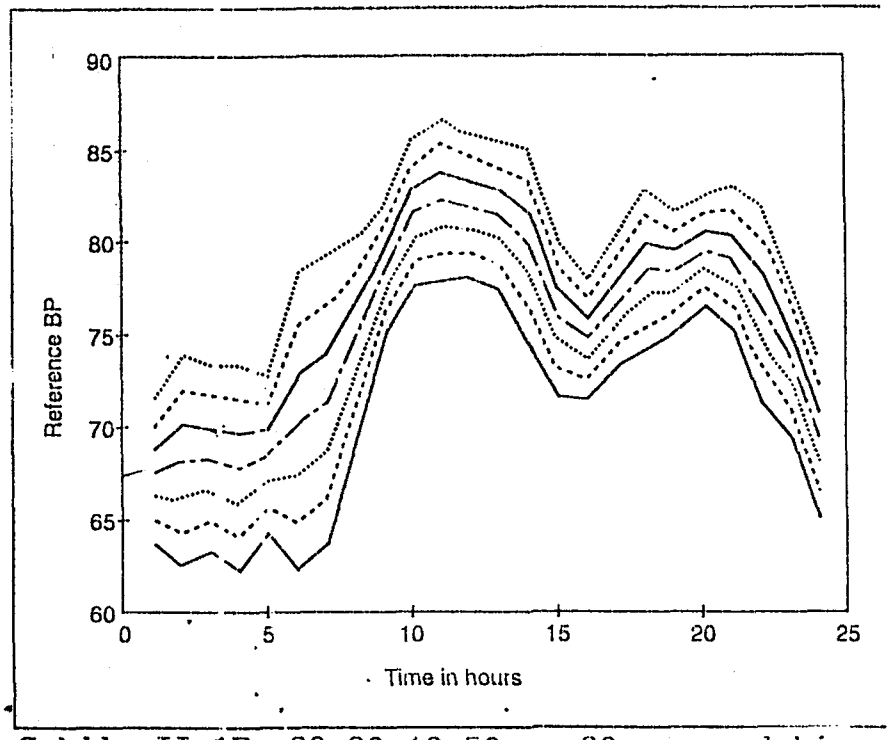
Uygulamada C programlama dili kullanılmış ve YNA düzenleyicisi (compiler) giriş değerlerini YNA için uygun değerlere dönüştürüp öğrenme işlemini başlatmıştır.Bu işleyiş 20 MHz 80386 PC'de saniyede 20.000 ağırlık hesaplama şeklinde gelişmiştir.Hesaplanan çıkışlar YNA düzenleyicisi tarafından uygun büyüklüklere çevrilir.

Ağları öğretmede ve test etmede 300 sağlıklı ve 85 hipertansif insanın kan basıncı zaman serileri kullanılmıştır.

RGM modülünü öğretmek için yaşları 20 ila 80 arasında değişen insanlardan sağlanan 200 adet kan basınç zaman serisi kullanılmıştır.RGM'in her bir YNA yapısındaki bağlantıların sayısı, öğretme datası sayısından çok azdır. Bundaki amaç haritalanma gerektiren basit hafızalama yerine ağa prototip öğretmeye çalışmaktır.RGM'nin öğrenme fazı boyunca YNA'lara ilk önce cinsiyeti ve yaşı gözönüne almaksızın tek bir prototipe yönelirler.Bu ilk prototip 200 zaman sersinin ortalamasının çok gerisinde kalmış ve "gürültü" olarak isimlendirilmiştir.Fakat bütün bilgiler girildikten sonra ağ kişilerin yaş ve cinsiyetlerinden meydana gelen zaman serileri farklılıklarını keşfetmiştir. Yani ağ referans kan basıncı zaman serilerini ,yaş ve cinsiyet parametrelerine bağlı şekilde yorumlayarak üretmiştir.Şekil II.18 diastolik kan basıncı zaman serilerinin öğrenme sonrası değerlerini ,yaşı parametre olarak gösterir.

DCM'nin öğrenme seti 30 Normal ve 30 hipertansif deneğin ilaç bileşimlerini ve anamestik dataları içerir.

Testler 10 Normal ve 10 hipertensif hastanın ilaç bileşimlerinin doğru tahminlerini gösteren bir set ile yapılmıştır.



Şekil II.17 20-30-40-50....80 yaşındaki erkekler için öğrenme sonrası diastolik kan basıncı zaman serileri değerleri.

TSM 'nin öğrenme seti 60 normal 60 hipertensif deneğin ilaç bileşimlerini tedavi bağımlı klinik bilgilerini ve basınç aşırılıklarını içeren bir settir.TSM 'nin fazlalaştırılmış tabakaları ve boyutlar yüzünden öğrenme fazı RGM ve DCM 'den daha uzun zaman alır.(Yaklaşık 50 saat CPU zamanı 20 MHz - 80386 PC için)

Öğrenme prosesine yardımcı olmak için hipertensif hastaların tedavisinde gözönüne alınan genel kriterler şunlardır:

(1) - 24 saat etkili ilaçlar tek doz olarak uygulanmalıdır. Kısa etkili ilaçlar günde üç kereye kadar verilmek zorundadırlar.Diüretikler aktiviteleri yüzünden bir tek sabah dozuyla sınırlanmalıdır.
(2) - Eğer mümkünse ,düşük uyuşumlu ilaçlar uygulanmalıdır.
(3) - Eğer kan basıncı artışları bütün gün sürerse ,tek doz ilaç uygundur ,diğer taraftan kısa etkili ilaç ta önerilir. Her halde her ilacın en güçlü etkiyi maximum basınç artışında gösterdikleri bulunmuştur.
(4) - Aynı ilacın iki uygulaması arasındaki zaman aralığı 6

saatten kısa olmamak zorundadır.
(5) - Gece boyunca ilaç uygulanmamalıdır.
(6) - Basınç artışları gün boyunca alabildiğine ilerler ve yüksek değerlerle karakterize edilirlerse iki veya daha fazla ilacın birleşimi kullanılabilir.
(7) - Önemli artışlar yoksa ilaç uygulanmamalıdır.
(8) - Ağızdan verilen ilaçlar için hesaba katılan ilaçlar hem yarım ,tam ve değersiz doz olarak önerilir.

Öğrenme ağı bu kurallara göre davranır .(Şekil II.19)

Öğrenme boyunca ,küçük basınç artışlarının dizginlenen etkisine çıkışlarda kural 7 uygulanarak birkaç iterasyonda ulaşılabilir.Kural 2'yi öğrenmek için biraz daha çok iterasyon gerekir.

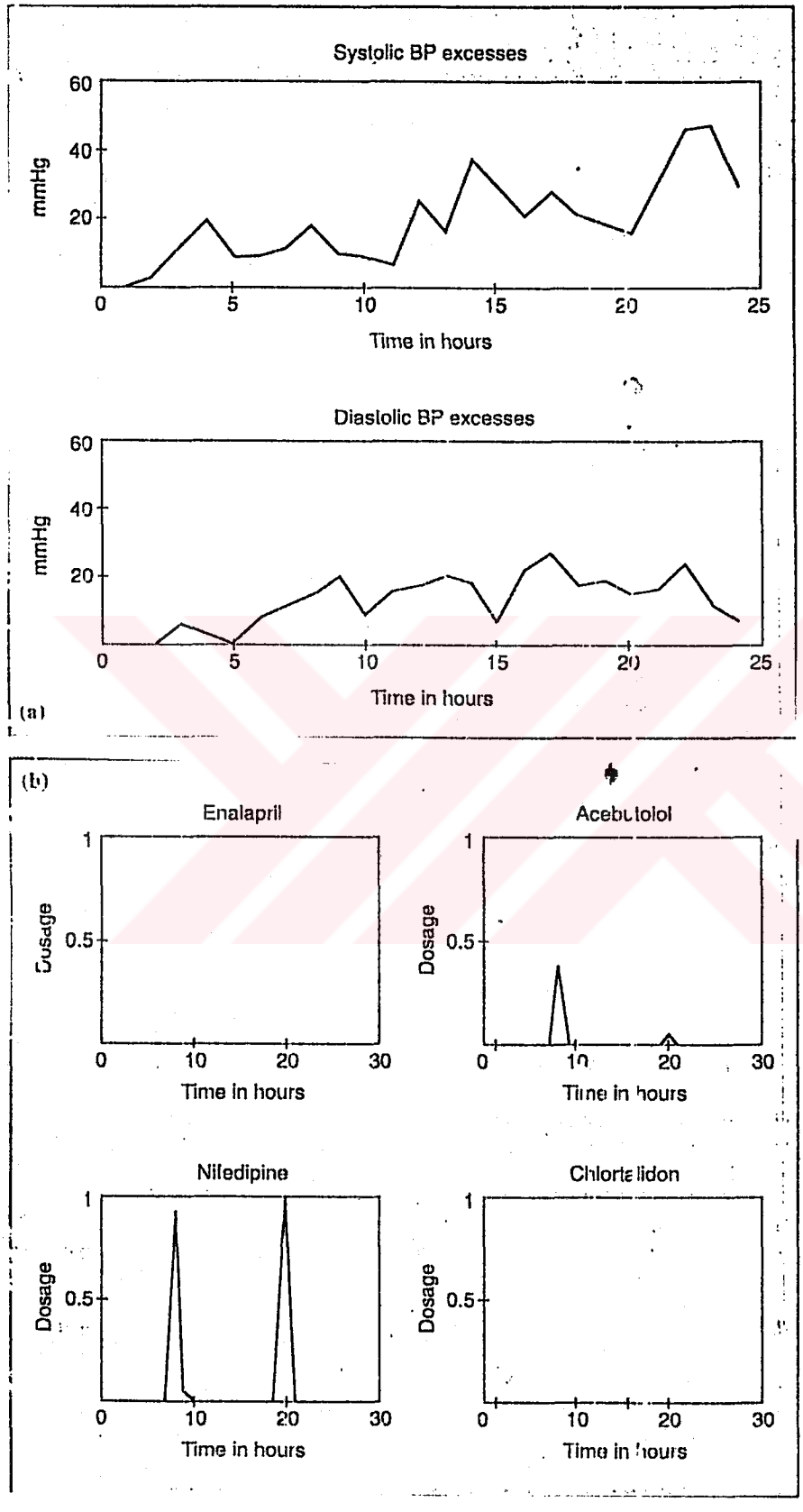
Aynı esneklik 1,3,4,5 ve 8. öğrenme kuralları için de geçerlidir.Bütün çıkışlar 0,0.5 ve 1 değerlerine çabucak ulaşırlar ; geniş pikler sınırlandırılmış ve sonuç çıkışlar impuls şeklindedirler (Şekil II.19 (b)).

İki veya daha fazla ilacın söz konusu olduğu birleşimlerde hastaların ilaç uyumuna göre en uygun çözelti seçilmelidir.Bu yüzden 6.kuralın keşfedilmesi çok daha fazla iterasyon gerektirir.

Test sonuçları sonrasında ;

	Doğru	Yanlış
Tedavi	11	1
Tedavi Yok	22	1
Toplam	33	2

tablosu çıkarılmış ve % 94 iyileştirme % 92 duyarlılık ve % 96 öznellik sonuçlarına ulaşılmıştır. Doktorların incelemesi ile ,Hypernet sistemi %82 uygulanabilir bulunmuştur.



Şekil II.19 Hipertensif hastanın basınç aşırılıkları (a) , Hypernet sistemi çıkışları (b).

II.4 MYOELEKTRİK SİNYAL ANALİZİ

Çok açılı protezlrin kontrolü bir çift yüzey elektrodundan sağlanan myoelektrik sinyaller (MES) kullanılarak yapılabilir.Bu özellikle dirsek üzerinden kolunu kaybetmiş insanlarda uygulanabilir. Benzer durumlarda kontrol staratejisi,myoelektrik sinyallerden kolayca çıkarılabilen karakteristik parametrelerin herbiri için aynı olan tekrarlanabilen kas kasılma karakterlerinin kümesini üretmektir.Bu parametreler kullanılarak farklı kas kasılması hareketlerini sınıflamak mümkündür. Kas kasılması sınıflarının herbiri ,protezde parça fonksiyonu tetiklemek için kullanılır.Sınıflar arasında iyi bir ayrıma sahip olmak için ,protez yerine takıldığında gerçekleşen ve normal kol fonksiyonu ile hiçbir benzerlik göstermeyen olağandışı kas kasılmalarının bir kümesini üretmek gerekir. Kullanıcının bu yeni kas kasılmalarını tekrar tekrar yapabilmeyi öğrenmeyi öğrendiği işleyiş yavaştır ve sıklıkla güvenilir bir kontrol üretemez.Kullanıcı öğrenmesini azaltmak amacıyla ,protez kontrol fonksiyonunu harekete geçirmek için kullanılan kasılmaların ,protez tarafından yapılmaya başlanan doğal fonksiyona benzemesi istenir.Pek çok geleneksel karakter tanıma kontrol sistemlerindeki bir başka kısıtlama da hesaplama miktarının büyük miktarda olması, bunun da kontrol mekanizmasında önemli gecikmelere yol açmasıdır.

Değişik kas kasılma karakterlerini ayırdedebilme yetisi ,MES'te tutılan bilgiye bağlıdır.Örnek olarak ,üç kasın yakınına yerleştirilmiş bir çift deri-yüzey elektrodlarından elde edilen sinyal, üç kasın herbirinin aktivitelerine ait bazı bilgiler içerecektir.Bu bilgiyi tanımlama problemi daha komplekstir ve MES'in doğasından ileri gelmektedir. [15] Yüzey elektrodundan belirlenen MES ,elektrod yakınında üreyen bütün motor ünite sinyallerinin kabaca bir toplamıdır.Yüzey elektrodun tek bir kasın bütününden aldığı sinyal değeri de tamamen rastlantısal bir yağı gösterir ve asenkron olarak tetiklenmiş motor ünite puls dizilerinin toplamından oluşur.Tek bir kastan alınan

ve diferansiyel olarak büyütülen MES ,sıfır ortalama değeri ve kas kasılma seviyesiyle orantılı değişiklik ile karakterize edilir.

Deri altındaki kas aktivitesi hakkında daha ileri bilgiler sağlayabilen bu sinyalden başka özellikler çıkarmak mümkündür.Bu teknik bütün MES'e katkıda bulunan bir kastan daha fazlasının olduğu durumlarda da uygulanabilir.Yüzey elektrodunun yakınında birkaç kas varsa sonuç-MES ; her kastaki aktiviteye ait bilgiler içerecektir. Madem ki elektrot ile kas arasındaki doku yüzünden bir filtre etkisi vardır, bu bilgi kasların komşuluğu ile ilişkili olarak yerleştirilen elektroda büyük ölçüde bağlı olan sinyalde nasıl gösterilecektir ? Kasların kümesinin içindeki kasılan karakter alınan sinyalden çıkarılan özelliklere dayalı olarak tanımlanabilir.Graupe tarafından gerçekleştirilen deneylere [17] dayalı olarak, bu tip kas aktiviteleri ,MES özelliklerinden tanımlanabilir.MES sinyallerinin sınıflamasıyla her bir sınıf normal kol fonksiyonlarını temsil edecek şekilde protezlerde kullanılabilir.

MES'ten sağlanan özelliklerin çıkarılması optimal tahmin tekniklerine uygun olacaktır.Optimizasyon problemi veya daha özel olan fonksiyonel minimizasyon ,bölmeli Hofield ağı (Discrete Hopfield Network) ile çözümlenebilir. [16] Bu ağın iteratif çözüm metodları kullanmasından dolayı ,MES özelliklerini çıkarmada ,ön yaklaşım düzenlemleriyle beraber ,hızlıdır.Bununla birlikte non-iteratif yöntemler kullanarak, kesin sonuç metodları ile bütün özellik ve parametreler MES'ten ayıklanabilir. fakat bu gereksiz hesaplama yükü artışları meydana getirecektir. İteratif tip çözüm kullanarak doğruluk ve hesaplama yükü arasındaki denge regüle edilebilir.Üstelik bölmeli Hofield ağı paralel olarak çalışır ve bu yüzden donanım uygulaması için çok uygundur.

Bölmeli Hopfield YNA'dan başka nöral ağ sınıflayıcıları da MES sinyallerini sınıflamada kullanılabilir.Dahili

düzenlenebilir doğaları yüzünden nöral ağ sınıflayıcıları uygulamada ,normalde gereken kullanıcı öğranmesinin miktarını azaltmak için kullanılabilir.Yine önemli bir özellikleri ,girişlerin ,olasılık yoğunluk fonksiyonu hakkında çoğu kez bir varsayım yapmayı gerektirmemeleridir. Çok katmalı perceptron (Multilayer perceptron) YNA'da MES özelliklerini sınıflamada kullanılmıştır.[16]

Graupe [17] ME sinyallerini başarıyla sınıflamış ve deneysel sonuçlarda başarı sağlayan bir ME özellikleri kümesi bulmuştur.Graupe zaman-serileri modeline dayalı olarak ,tek kanal MES'lerden özellik ayrımı için bir algoritma geliştirmiştir.Bu özellik ayrımı daha sonraları tek bir üst kol elektrod çifti ile sağlanan ME sinyallerinin değişik tiplerini sınıflamak için de kullanılmıştır.Buradaki amaç tek bir yerden alınan ME sinyalleri ile serbest protezlerin çok açılı kontrolünü sağlamak idi.Fonksiyonel ayrımın yüksek dereceleri gerçekleştirildi ama bu birkaç saatlik kullanıcı öğrenmesinden sonra oluyordu.Ayrıca özellik ayırma algoritmasına uygulanmak için kullanılan hesapla ilgili donanım klinik uygulama için yavaştı.Ayırma tekniği hareketli zaman-serileri modeline dayanıyordu;

$$\dot{x}_k = \Sigma_{i=1}^{n} a_i \cdot x_{k-i} \qquad \text{(II.5)}$$

Burada $\dot{x}_k$ örneklenmiş MES'in tahmin değeri, x_k ise k. örnekleme değeridir.Bu n. basamak lineer model alınan MES'teki n örneklenmeye bağlıdır.a_i parametresi üst koldaki kas kasılmalarının farklı tiplerini belirler.

Bu zaman-serilerinin parametrelerini "a_i" ayırmak için Graupe tarafından kullnılan minimizasyon algoritması sıralı en küçük karelerdir(SLS).Bu yaklaşımın arkasındaki prensip ,her "k" örnekleme zamanında güncel sinyal değeri x_k ve tahmini $\dot{x}_k$ arasındaki farkı minimize etmektir.Basit en küçük kareler maliyet (cost) fonksiyonu bu farkın karesi alınarak elde edilebilir.Zaman serileri parametreleri hesaplama problemi böylece ,MES bilgi kümesini bütünüyle

içine alan bu fonksiyonun minimize edilmesi haline gelir.BU minimizasyon problemi şu eşitliklerle gösterilebilir ;

$$\dot{a}_r = \dot{a}_{r-1} + P_r.X_r(x_r - X_r^T.\dot{a}_{r-1}) \qquad (II.6)$$

$$P_r = P_{r-1} - \frac{P_{r-1}.X_r.X_r^T.P_{r-1}}{1 + X_r^T P_{r-1} X_r} \qquad (II.7)$$

Burada X_r x_{r-1}'den x_{r-n}'ye kadar örnekleme içeren sütun vektör ve r bilgi örnekleme akışını temsil eden tamsayıdır. $\dot{a}_r$ sütun vektör olarak ifade edilen zaman-serileri parametrelerinin temsilcisidir. $\dot{a}_{r-1}$ r-1 zamanında $\dot{a}_r$ vektörünün bir önceki değeridir.P_r r. iterasyona göre nxn boyutunda bir matristir ve (II.7) 'teki değere sahiptir.

Başlangıçta sütun vektör $\dot{a}_0$ sıfırlanır ve P_0 matrisi ,birim matrisin bazı sabitlerle çarpımı olarak kurulur.

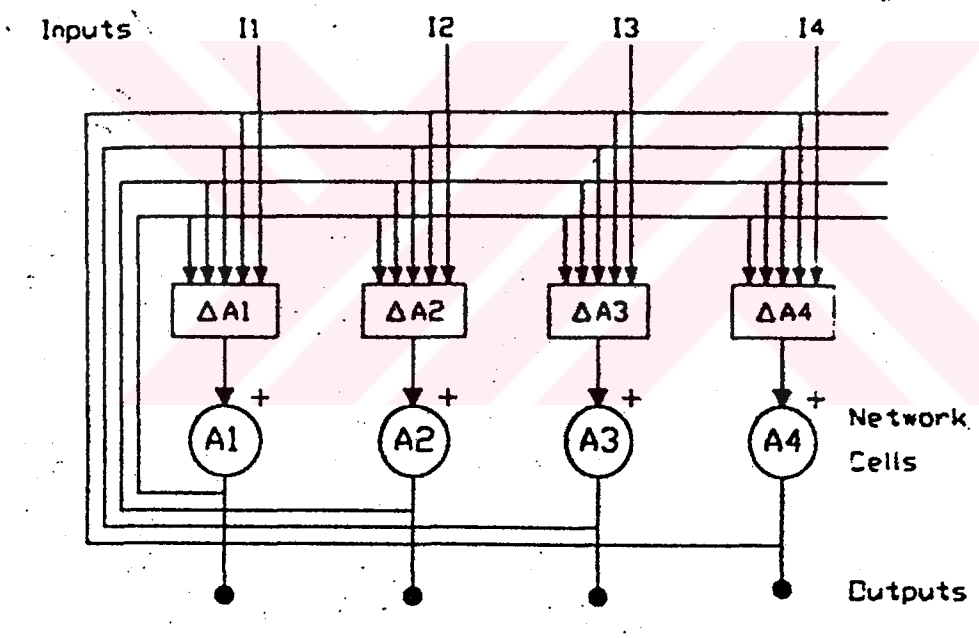
Graupe deneysel çalışmalarında elektrodlardan aldığı MES sinyallerini 500 Hz'de örneklenmiş ve 80 ila 120 arasında bilgi noktasını SLS algoritmasına uygulamıştır.MES'lerin 0.2 - 0.3 s kayıt aralığında kararlı hale yakın davranış gösterdiği bulunmuştur.250 Hz'lik band genişliği sınırlamasının 500 Hz'lik örnekleme frekansı ile ayrım için yeterli bilgi sakladığı gösterilmiştir.Model derecesi MES'ten ayrılan özelliklerin sayısını belirtir.Bu ayrıca tahmini $\dot{x}$ 'in örneklenmiş anlık x'e ne kadar uyacağını da kontrol eder.Deneyler $n > 4$ için önemli bir özellik ayrımı ilerlemesi olmadığını ortaya koyar.Ayrıca kullanılan SLS algoritmasının matris çarpımları yüzünden hesaplama yükü model derecesi ile dramatik olarak artmaktadır.Bu yüzden Graupe çalışmalarının birçoğunda 4.derece modeller kullanmıştır.

MES özelliklerinden sonra , bunların YNA kullanarak nasıl ayrıldıklarını inceleyelim.

.BÖLMELİ HOPFIELD YNA İLE MES ANALİZİ

Bölmeli Hopfield YNA ,optimizasyon problemlerinin çözümü

için etkili bir araçtır.[18] Tipik uygulamalrı kısıtlamalı veya kısıtlamasız birkaç değişkenli fonksiyonların minimizasyonunu da içerir.Ağ yapısı ,t anında herbirinin akım çıkışları olan ,n tane işlem elemanı kümesini içerir. Herbir işlem elemanı (ünite) çıkışı ,optimizasyon probleminde verilen değişkeni temsil eder.Ünite çıkışlarının sistemin anlık durumunu temsil eden birtek n boyutlu vektör olduğunu kolayca varsayabiliriz.Böyle bir YNA Şekil II.20'de görülebilir.Bu vektör giriş değerlerinin kümesine dayanan update(yenileme) kuralı kullanarak, herbir zaman aralığında iteratif olarak modifike edilebilir.



Şekil II.20 Bölmeli Hopfield ağı yapısı

LIAPUNOV fonksiyonu "j" oluşturulabilir ve Hopfield yenileme kuralıyla beraber j'nin karalı minimuma gidişi gösterilebilir ;

$$j = -\frac{1}{2} \Sigma_i \Sigma_j T_{ij} a_i a_j - \Sigma I_i a_i \qquad (II.8)$$

Bu gösterimde j. hücrenin çıkışı a_j'dir.Bu durum vektöründeki j. elemana bağlıdır.Değişken I_i ferdi ağ girişlerini temsil eder.Herbir T_{ij} ,i. hücrenin çıkışı ve j. hücrenin girişi arasındaki bağlantı ağırlığının karşılığıdır.Bu bağlantı ağırlıkları ,sistemin şimdiki haline dayalı olarak herbir ünite çıkışının nasıl yenileceğini kontrol eder.Bu yolla ,T_{ij} ve I_i 'ler ağ yapısını açıkça belirtir.Bağlantı ağırlıklarının simetrik olması durumunda , $T_{ij} = T_{ji}$,Hopfield yenileme algoritması sistem enerjisi J'yi minimize eder."r." iterasyona bağlı olarak ünite çıkışlarını değiştirmek için yenileme eşitliği şöyle verilir ;

$$\text{Delta } a_{ir} = \Phi_i \left(\Sigma_{j=1}^{n} T_{ij}\, a_{jr} + I_i\right) \quad \text{(II.9)}$$

Burada değişken Φ_i algoritmanın yakınsama oranını belirten kazanç terimini temsil eder. J fonksiyonunda görülen yenileme kuralının etkisi ,a_i çıkışlarına bağlı olarak J'nin gradyanı alınarak bulunabilir.Sonuç işaretin tersini alarak ,bunun ,yenileme ilişkisinin kesik kesik (discrete) adımlarla J'de minimuma hareket edecek şekilde sistem durumunu değiştirdiğinin bir kanıtı olduğu söylenebilir.Her bir ünite çıkışı bu ilişki kullanılarak şöyle yenilenebilir;

$$a_{ir} = a_{i(r-1)} + \text{delta } a_{ir}$$

delta a_{ir} sıfıra eşit olduğunda YNA yakınsaması denen olay gerçekleşir ve $a_{ir} = a_{i(r-1)}$ olur. Fakat bu mükemmel sonuç her zaman gerçeklenemez.Pratikte önemli olan delta a_{ir}'nin belirlenen ϵ hata değerinden az olmasıdır.

Hopfield algoritması ortalama MES modelinin zaman-serisi parametrelerinin hesaplamasında kullanılabilir.J fonksiyonu şöyle verilir;

$$J = \frac{1}{2} \Sigma_{i=1}^{M} (x_i - \dot{x}_i)^2 \quad \text{(II.10)}$$

Bu eşitlik Hopfield enerji eşitliğine benzeyecek şekilde büyütülebilir.II.10'daki karesel ifade açılarak ve n. basamak zaman-serisi modeli bu eşitlikte yerine konularak ,verilen terimlerle yeniden düzenlenirse;

$$J = -\frac{1}{2} \Sigma_{j,l=1}^{n} T_{jl}\, a_j\, a_l - \Sigma_{j=1}^{n} I_j\, a_j + \frac{1}{2} \Sigma_{i=1}^{M} x_i^2 \qquad (II.11)$$

bulunur.Burada;

$$T_{jl} = - \Sigma_{i=\beta}^{M} x_{i-j} \cdot x_{i-l}$$

ve

$$I_j = \Sigma_{i=\beta}^{M} x_{i-j} \cdot x_i$$ olacaktır.

Bu eşitliklerde (i-l) > 0 ve (i-j) > 0 olacaktır.Düşük toplama ,indeksi ß sadece özel M bilgi noktaları gözönünde tutularak elde edilir. Genellikle M büyüktür ve ß algoritma performansını düşünmeksizin zaman-serileri modelinin derecesi olan n'e eşitlenir.

.DENEYSEL ÇALIŞMALAR

MES sınıflamak için "Bölmeli Hopfield YNA" kullanarak bir çalışma yapılmıştır.[16] Bunun için ME bilgilerinin birçok kümesi Hopfield algoritmasını test etmek için kaydedilir. Bu bilgiler bipolar yüzey elektrodu çifti kullanan diferansiyel yükseltme ile sağlanır.[19] İki Beckman yer değiştiren (floating) elektrod deneğin üst kol kısmına takılır.Birisi Triceps brachii (kolun üst kısmının orta tarafındaki üç başlı kas) kasının üzerinde diğeri deltoidin merkezden uzaktaki parçası seviyesindeki biceps brachii (iki başlı kas) üzerindedir.Alınan sinyal 500 Hz'de örneklenir ve dijital olarak saklanır.Bütün hesaplamalar için değerler -10.0 V ve +10.0 V sahasında ölçülendirilir. Hopfield ve SLS algoritmasının her ikisi de 80386 CPU bilgisayarda Fortran dilinde yazılmıştır.II.9'teki kazanç

terimi ağ yakınsamasını tatmin etmek zorundadır.

$$\Phi i = A.\left(\frac{2}{|T_{ii}|}\right) \qquad (II.12)$$

Yukarıdaki denklemdeki A değeri deneyler sonucu 0.8 olarak bulunmuştur.T_{ii} i. ünite girişine olan ağırlığı temsil eder.Her testin başında Hopfield algoritması için durum vektörü -4.0 ve +4.0 arasında düzenli olarak yayılan değerlerle rastgele kurulur.

Hopfield ve SLS algoritmaları zaman-serileri parametreleri olan a_i'leri hesaplama ve işlemleri yapma zamanları temel alınarak birbiriyle karşılaştırılabilir. Bütün sonuçlar göstermiştir ki; Hopfield algoritması SLS algoritmasında olduğu gibi aynı parametreleri üretir. Hopfield algoritması hata kriteri olan ϵ'in artırılmasında SLS'den daha kötü sonuçlar çıkarılmıştır. Hopfield algoritmasında II.12'den gelen kazanç terimi kullanılarak yaklaşım olayların bütününde ,100 iterasyondan daha az bir iterasyonda meydana gelir. $\epsilon=10^{-4}$ yaklaşım kriteri Hopfield algoritmasında kullanılarak yapılan hesaplamalar ,SLS algoritmasıyla karşılaştırıldığında iki önemli kesinlik söz konusudur. Hopfield algoritmasının hata minimumu üzerine çıkışma eğilimi iyi bilinmesine rağmen testlerde böyle bir bulgu yoktur.Bu enerji fonksiyonu J'nin bu olayda tamamen konkav olduğu sonucunu çıkarır ve bu yüzden düşme eğimi yaklaşımı kullanarak minimizasyon sağlanmalıdır.

İki optimizasyon yaklaşımı arasındaki bir farklılık onların işleyiş zamanlarında gözlenir.Benzer hesaplama teknikleri ve yazılım dili kullanmasına rağmen ,Hopfield algoritması tablo II.2'den de görüleceği üzere SLS'den 2 veya 3 kat daha hızlıdır.O zaman SLS yerine RLS (Fast Recursive Least Squares) yani hızlı takip eden en az kareler algoritmasının kullanılması düşünülebilir.

Number of Points (M)	Convergence Criterion (ϵ)	Hopfield Avg. Time (s)	SLS Average Time (s)	Fast RLS Avg. Time (s)
100	10^{-4}	0.052	0.11	0.054
125	10^{-4}	0.052	0.13	0.073
150	10^{-4}	0.055	0.16	0.110
125	10^{-2}	0.037	0.13	0.073
1000	10^{-4}	0.108	1.05	0.600

Tablo II.2 SLS ,RLS ve Hopfield algoritmaları için yakınsama oranlarını gösteren tablo.

Buradaki ortalamalar altı farklı ME sinyali kullanılarak yapılan 1800 deneme esas alınarak hazırlanmıştır.Hopfield algoritmasını durdurmakta kullanılan yaklaşım kriteri ϵ algoritma hızına önemli bir etkiye sahiptir.Bu kriter algoritmanın işleyiş zamanını temsil eder.Hopfield algoritmasında analiz yaparken daha fazla bilgi noktası kullanmak için işleyiş zamanında önemli bir değişiklik yapmaz.Fakat bu SLS ve RLS için aynı değildir ve bu algoritmaların hızları büyük ölçüde M bilgi nokta sayısına bağlıdır.100 < M < 150 için RLS ve Hopfield algoritmalarında hesaplama hızı bakımından farkın küçük olmasına rağmen ,M=1000 için Hopfield algoritması RLS'den 5 kez daha hızlıdır.Bütün bu sonuçlar bölmeli Hopfield YNA'nın MES işlemi için uygun olduğu sonucunu doğurur.

.ÇOK TABAKALI PERCEPTRON İLE MES ANALİZİ

Tek kanal MES kullanarak 4 adet kol fonksiyonu arasında sınıflama yapmak için çok tabakalı perceptron da kullanılabilir.MES özelikleri ME sinyallerinden çıkarıldıktan sonra ,bu sinyalleri sınıflama problemi özellik uzayında bir karakteri tanıma şekline dönüşecektir. ME sinyalleri sınıflamak için çok tabakalı perceptron

,zaman-serisi parametrelerini "a_i" ve sinyal güç seviyesini kullanır.Zaman serisi parametrelerinin 4. derece modeldeki a_2,a_3,a_4 bileşenlerinin sinyal sınıflama bilgisine önemli bir katkısı yoktur.Dolayısıyla a_1 parametresi ve P bilgisi sınıflama için yeterlidir.

a_1 ve P bilgileri çok tabakalı perceptron YNA için giriş olarak kullanılır.Bu tür ağ-sınıflayıcı ,analog giriş üreten problemler için uygundur.Ağı eğitirken kullanılan algoritma süpervise öğretme algoritmasıdır.Yani çğrenme bilgisi, ağ bütün girişlere karşı arzu edilen çıkışları üretene kadar ağda varolur.Bu konuda daha önce yapılan çalışmalar da dikkate alınacak olursa üç tabakalı YNA yapısı böyle bir problem için uygun olacaktır.

Şekil II.21'de görülen YNA bu problemi çözmek için sunulabilir.Bir giriş ,bir gizli ve bir de çıkış tabakasına sahiptir.Her giriş ünitesi gizli tabakadaki her üniteye ileri besleme yöntemi ile bağlanır.Üniteler arası bağlantılar, bir ünitenin çıkışının diğerine olan etkisini gösteren W ağırlıklarına sahiptir.Giriş üniteleri analog giriş değerlerini tutmaktan başka bir fonksiyona sahip değildir.Çıkış ve gizli tabakadaki üniteler ise şöyle bir transfer fonksiyonuna sahiptir;

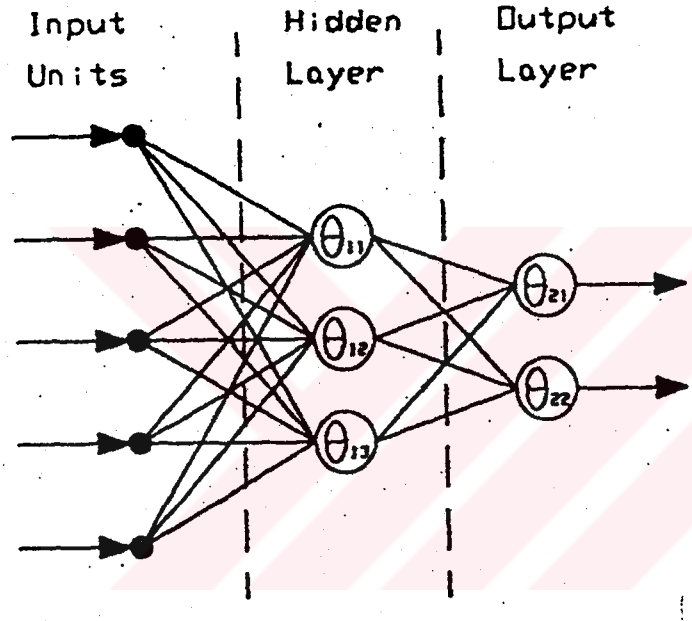
$$F(\alpha) = \frac{1}{1 + e^{-(\alpha-\sigma)}} \qquad \text{(II.13)}$$

Burada ;

$$\alpha = \Sigma_{i=1}^{N} W_i X_i \text{ 'dir.} \qquad \text{(II.14)}$$

Her üniteye net olmayan giriş ,bir önceki tabakadan gelen tek tek ağırlıklandırılan çıkışların toplamı olarak temsil edilebilir.Her ünitenin çıkışı bu girişlerin toplamının bir fonksiyonudur.II.13 eşitliği her ünitede kullanılan sigmoid fonksiyonunu gösterir.Değişken α II.14'da açıklandığı üzere girişlerin toplamıdır.X_i i.

ünitenin bir önceki tabakadaki çıkışıdır ve bağlantı ağırlığı W_i ile birleştirilir.Değişken σ üniteye verilen özel eşik değerini temsil eder.



Şekil II.21 Üç tabakalı perceptron

Çok tabakalı perceptron'ların öğrenmesi ,bağlantı ağırlıklarını ve ünite eşik değerlerini değiştirmek sureti ile olur.Back-propagation (hatanın geriye yayılımı) algoritması bu ağ için uygun olarak kullanılabilir.Adından da anlaşılacağı gibi,teknik bağlantı ağırlıklarını ayarlamak için ,çıkış hatasını ağın gerisine doğru yaymayı hedef alır.Buradaki çıkış hatası terimi ,öğrenme aşamasında ,ağın o anki çıkış değeri ile istenen çıkış arasındaki fark olarak bilinir.Çıkış ünitelerinin sigmoidal fonksiyonu

yüzünden ,ağ çıkışları binary formunda olmaya eğilimlidir. Tipik sınıflama düzeninde , her bir çıkış, bilgi kümesinin belli bir parçasını temsil edecek bir şekilde oluşturulur. Bu yüzden sınıflamanın anlamsız olmaması için ,her bir giriş karakteri için sadece bir ağ çıkış ünitesi aktif olmalıdır.Hatanın geriye iletimi algoritması şu şekilde gösterilebilir:

$$\text{Delta } W_{ji} = B.C_j.O_i$$

Ağda birbirini takip eden her iki tabaka arasında , biri primer diğeri sekonder şeklinde bir tanımlama yapılabilir. Bu tanımlamaya göre ,primer tabakanın ağırlıklandırılmış çıkışları sekonder tabakaya gider.Yukarıdaki Delta W_{ji} eşitliği , sekonder tabaka i. ünite ve primer tabaka j. ünite arasındaki bağlantı aralığının değişimini ifade eder.Örnek olarak ağın son iki tabakası gözönüne alınıyorsa ,ilgili bağlantı ağırlığı ,çıkış tabakasının i. ünitesi ile gizli tabakanın j. ünitesi arasındadır.Varsayımlar gözönünde tutularak "O_i" terimi sekonder tabaka i. ünite çıkışıdır. Burada B terimi öğrenme oranını kontrol eder.C_j terimi ise ağdaki her çıkış ünitesi için şöyle verilir ve hatayı temsil eder;

$$C_j = (t_j - O_j) \cdot \frac{\tau\, O_j}{\tau\, net_j}$$

(Not : τ kısmi türevi göstermektedir)

Aynı şekilde her bir gizli ünite için hata terimi ;

$$C_j = \frac{\tau\, O_j}{\tau\, net_j} \; \Sigma_k \; C_k.W_{kj}$$ şeklindedir.

Birinci çıkarımda t_j ; j. çıkış ünitesinin arzu edilen

cevabıdır.Sekonder tabakadaki j. ünitenin çıkışının kısmi türevi ,bütün ünite girişleri netj'e göre alınır.Bütün ünite girişleri ,bütün ağırlılklandırılmış primer tabaka çıkışlarının toplamıdır.Bu diferansiyel eşitlik ağağıda gösterilen şekle sigmoid transfer fonksiyonundan dolayı dönüştürülebilir;

$$\frac{\tau O_j}{\tau net_j} = O_j (1-O_j) \text{ 'dir.}$$

Bu çıkarım kullanılarak kullanışlı iki hata terimi elde edilir.

Çıkış ünitesi için : $C_j = (t_j-O_j)\ O_j\ (1-O_j)$ (II.15)
Gizli ünitesi için : $C_j = O_j(1-O_j)\ \Sigma\ C_k W_{kj}$ (II.16)

İkinci çıkarımdaki toplam ifadesi ağın gelecek tabakasıyla ilgili olan tüm ağırlık ve hata terimlerinin toplanmasıyla elde edilmiştir.Bu yolla ağırlıklar çıkış ünitesinin başlatmasıyla yenilenir.B terimi kazanç terimi olarak da ifade edilebilir.

.DENEYSEL ÇALIŞMALAR

MES bilgileri, kesilme işlemi yapıldıktan sonra, kolunun dirseğin aşağısında kalan parçası çok kısa olan erkek deneklerden sağlanmıştır. [16] Normal kol fonksiyonlarına uygun olan 4 güçlü kas kasılma dizisini üretmeleri deneklerden istenir.Bunlar özellikle eklemi uzatma (elbow extension) ,eklemi içeriye doğru bükme (elbow flexion) , bileği içeri doğru döndürme (wrist pronation) ve bileği dışa döndürme (wrist supination) hareketleridir.Madem ki, denek doğuştan bir kesikliğe sahip değildir ,o zaman bu fonksiyonları gerçekleştirme kabiliyetine sahipti ve yakındır.Denek daha önceden bu fonksiyonları gerçekleştirmekte kullandığı ön kol kasılmalarını ,ön kolunu yitirdiği halde hala taklit edebilir.Bir çift Beckmann elektrodu üç ana kas arasındaki sinyalleri

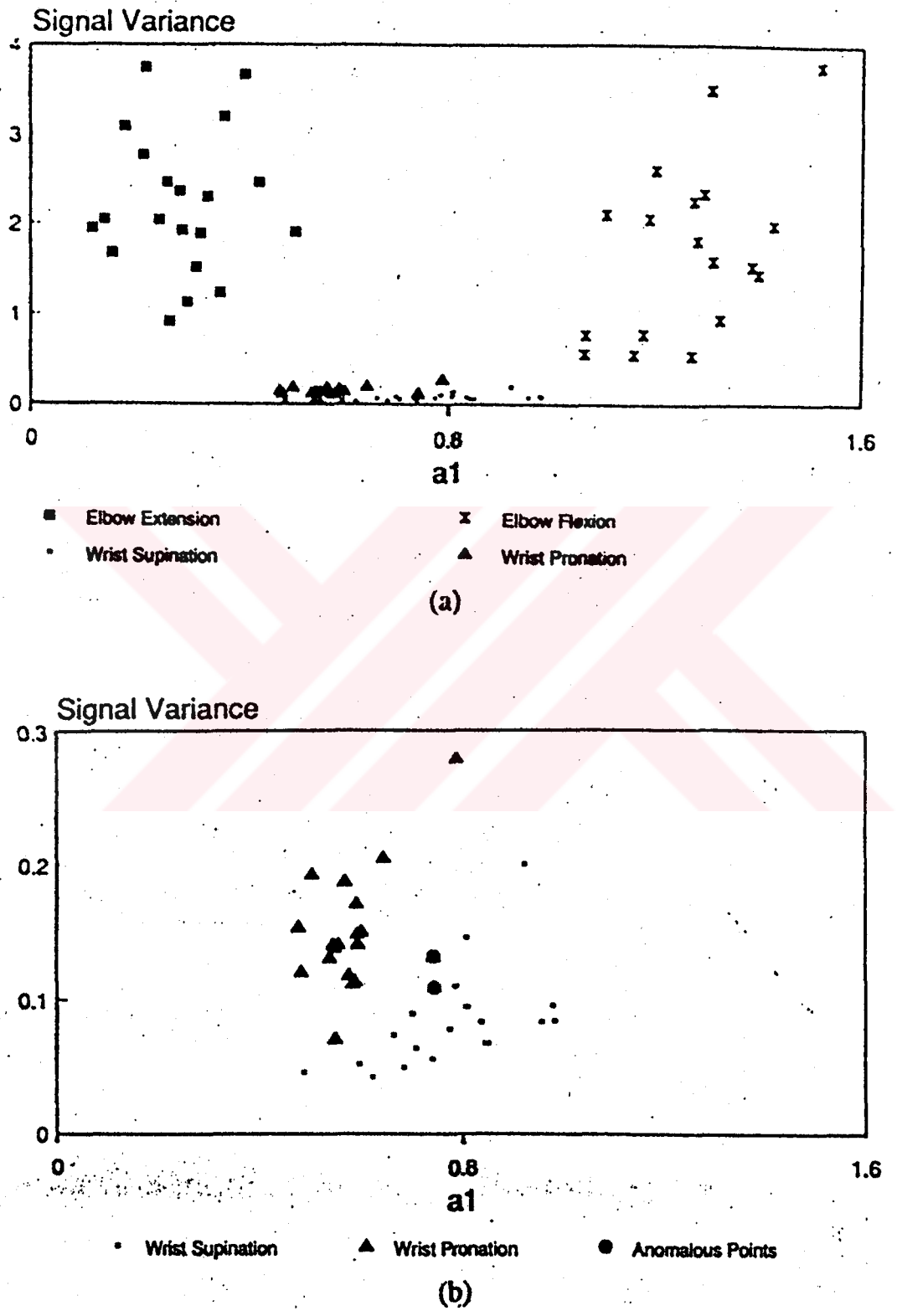
maksimize edilmiş bir şekilde alabilmek için bir önceki deney sonuçları bölümündeki gibi yerleştirir.MES diferansiyel olarak yükseltilir ve dijital formda saklanır. Örneklemeler iki saniyelik bir periyotta 500 Hz'lik örnekleme frekansı ile yapılır.MES bilgileri ağa gönderilmeden önce -10.0 V ve +10.0 V arasında ölçeklendirilir.İki saniyelik örneklemenin herbiri için bilgilerden özeliklerin kümesi çıkartılır.Zaman-serisi parametresi a_1 ve ortalama sinyal gücü P her 1000 noktalı bilgi setinin içindeki 100 birbirini takip eden bilgi içeren kümeden sağlanır.Böyle bir çalışmada 2 giriş (a_1 ve P), 4 gizli ve 4 çıkış (sınıflaması istenen kol fonksiyonları) üniteden oluşan bir YNA kullanılmıştır.[16] Şekil II.22 a_1 parametresi ve P sinyal ortalama gücü girişleri kullanan bu ağın 4 kol fonksiyonunu nasıl ayırdığını gösterir.

Şekil II.22(b) 'deki Anamalous noktalar ağın ilk önce tanıyamadığı noktalardır ve ilk aşamada bilgi setinden çıkarılmışlardır.İlk görüntülemeden sonra yeniden bilgi setine katılan bu noktalar 2768 iterasyon ortalaması gibi yüksek bir ortalama ile tanınmıştır.

Deneyler sonucu B=3.0 'lık bir kazanç ve 0.10'luk bir yakınsama ile YNA'nın yakınsama hatası üretmediği bulunmuştur.Bu deneyleri gösteren tablo II.3 aşağıdadır.

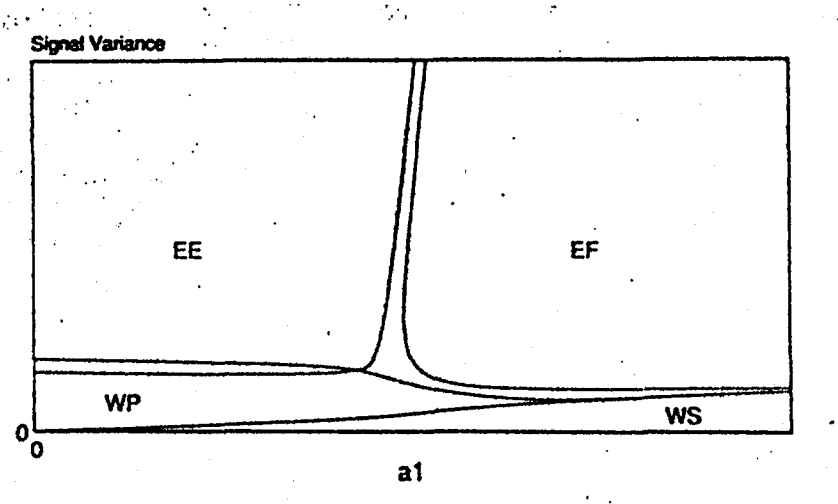
Gain	Convergence Criterion	Average Number of Iterations	Convergence Failures (%)
0.5	±0.25	***	100
0.9	±0.25	727	18
2.0	±0.25	544	10
3.0	±0.25	532	10
4.0	±0.25	526	20
3.0	±0.10	577	0

Tablo II.3 Değişik kazanç ve yakınsama kriteri için iterasyon sayılarını ve yakınsama hatalarını gösterir.



Şekil II.22 (a) a_1'e karşı sinyal değişimi
(b) Wrist fonksiyonları için sinyal değişimi

Şekil II.23 ise kullanılan sigmoid transfer fonksiyonu yüzünden asimptotik eğriler formunda olan ayrılmış bölge sınırlarını gösterir.



Şekil II.23 Perceptron ayrılması bölge sınırları

Lipmann'ın da önerdiği gibi [18] bu noktaların oluşturduğu grafikler sadece sınır bölgelerinde eğriseldir. Diğer yerlerde lineer davranış gösterirler.

Sonuç olarak ,bölmeli Hopfield ağı ve Çok tabakalı perceptron,protezlerin kontrolünde kullanılabilecek ideal yöntemlerdir.

II.5 BİLGİSAYAR YARDIMLI MAKRO MOTOR ÜNİTESİ POTANSİYEL SINIFLAMASI

Kasların elektriksel aktiviteleri bu yüzyılın başından beri geniş olarak araştırılmıştır.Takip eden yıllardaki hızlı teknolojik gelişmeler ,kas fiberlerinin tek tek ve motor üniteyi oluşturan fonksiyonel gruplarının özellikleri araştırıldı.Bu bölümde motor ünite potansiyel sınıflaması yapılarak bu sınıflama ile üç adet kas bozukluğunu teşhis etmeye çalışan bir araştırma aktarılacaktır.[20]

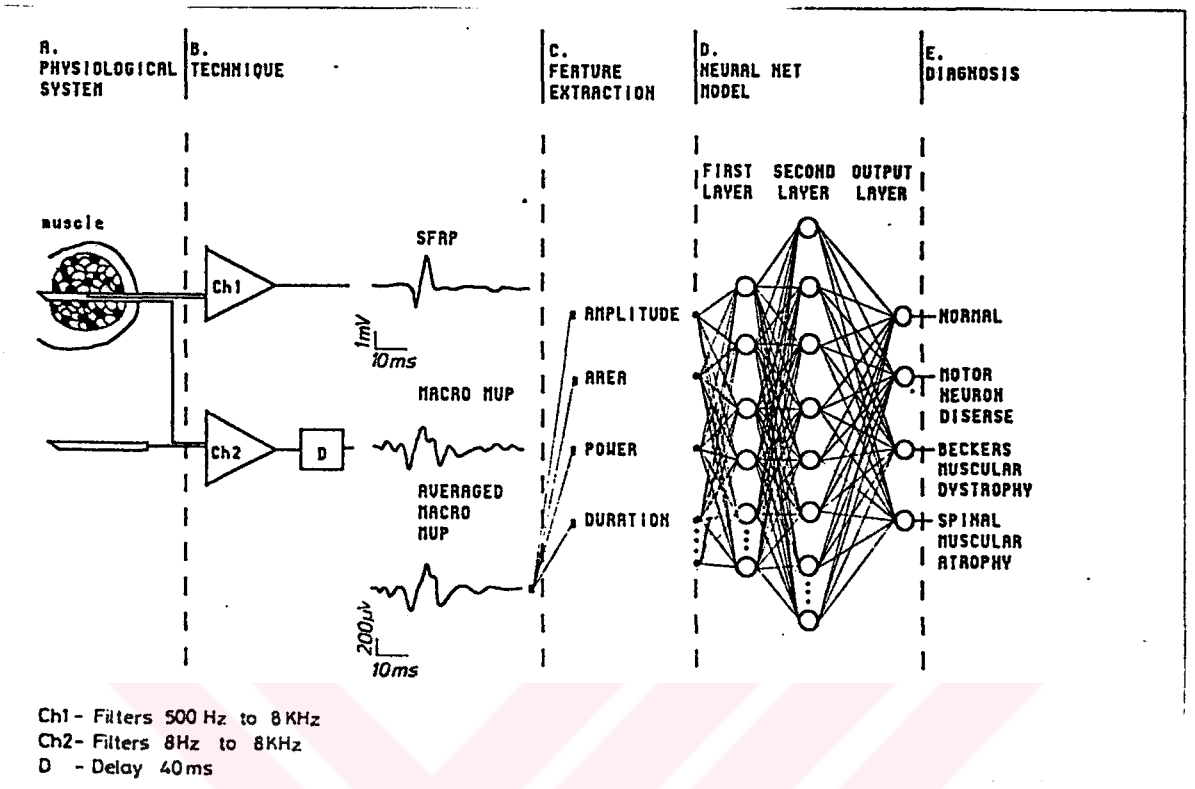
Motor ünite ,bir sinir axonu tarafından innerve edilmiş (merkezi sinir sisteminden gelen normal uyarıların bu

sinirlerle ait olduğu organa gitmesi) 2 ila 1000 fiberden ibarettir ve fiberler böylece 2 ila 30 Hz arasındaki tipik frekanslarda deşarj olurlar.Motor ünite potansiyeli (MUP) ölçmek için değişik yöntemler geliştirilmiştir. Stalberg [21] tarafından geliştirilen yöntem buna bir örnektir.Makro EMG metodu,15 mm uzunluklu 0.8 mm çaplı tüp şeklinde iğne elektrod ve 25 µm çaplı bir side-port elektrod kullanır.Side-port elektrod ,kasların yumuşak ve ihtiyari kasılmalarını düzenlediği tek motor üniteye ait bir veya daha fazla kas fiberlerinin aktivitelerini kaydeder.Bir tek fiber aksiyon potansiyeli (SFAP) ,içinde tüp (cannula; vücuda sokulan tüp) sinyalinin beslendiği sinyal ortalayıcıyı tetiklemek için kullanılır. (Şekil II.24) Genelde ünitenin 200 civarındaki deşarjları ,makro motor ünite potansiyellerini (MMUP) elde etmek için 80 ms tarama süresiyle ortalanmıştır.Son 20 potansiyelde motor ünite potansiyelinin parametrelerini makul tahmin etmek için tek kastan ölçülür.

MMUP bilgisi sinyalin tepeden tepeye büyüklüğü yine sinyalin 50 ms'lik genişliğin merkezinde olacak şekilde analiz edilir.Potansiyel süresini de belirleyen ek bir parametre oluşturulur.Bu süre % 90 güç süresidir .Yani sinyalin % 90'lık gücünü ihtiva eden bölgenin zaman olarak genişliğidir.

Bu çalışmada [20] teşhis edilmesi istenen kas bozuklukları üç tanedir.Bunlar motor nöron hasarı (MND) ,Becker kas distropisi (BMD) ve spinal kas atropisi (SMA) bozukluklarıdır.MND ve SMA'nın her ikisi de nörojenik işleyiş ile ilgilidir veya MND'de olduğu gibi yaşlı nüfusta hızlı ilerlemeye sahiptir ya da SMA'de olduğu gibi genç nüfusta daha kronik ve etkilidir.Her iki hasarda da ,motor nöronlarının kaybedilmesini yaşayan nöronlar tarafından kas fiberlerinin yeniden sinirlendirilmesini takip eder ki; bu olay motor ünitenin büyüklüğünün MMUP büyüklüğündeki artışın birbirine bağlı olması gibidir.Tersine BMD, motor ünitelerinin sayısının sabit kaldığı fakat fiberlerin tek tek kaybı yüzünden her bir ünitenin büyüklüğünün azaldığı

bir işleyiştir.Bu durumda MMUP büyüklüğü daha düşüktür.

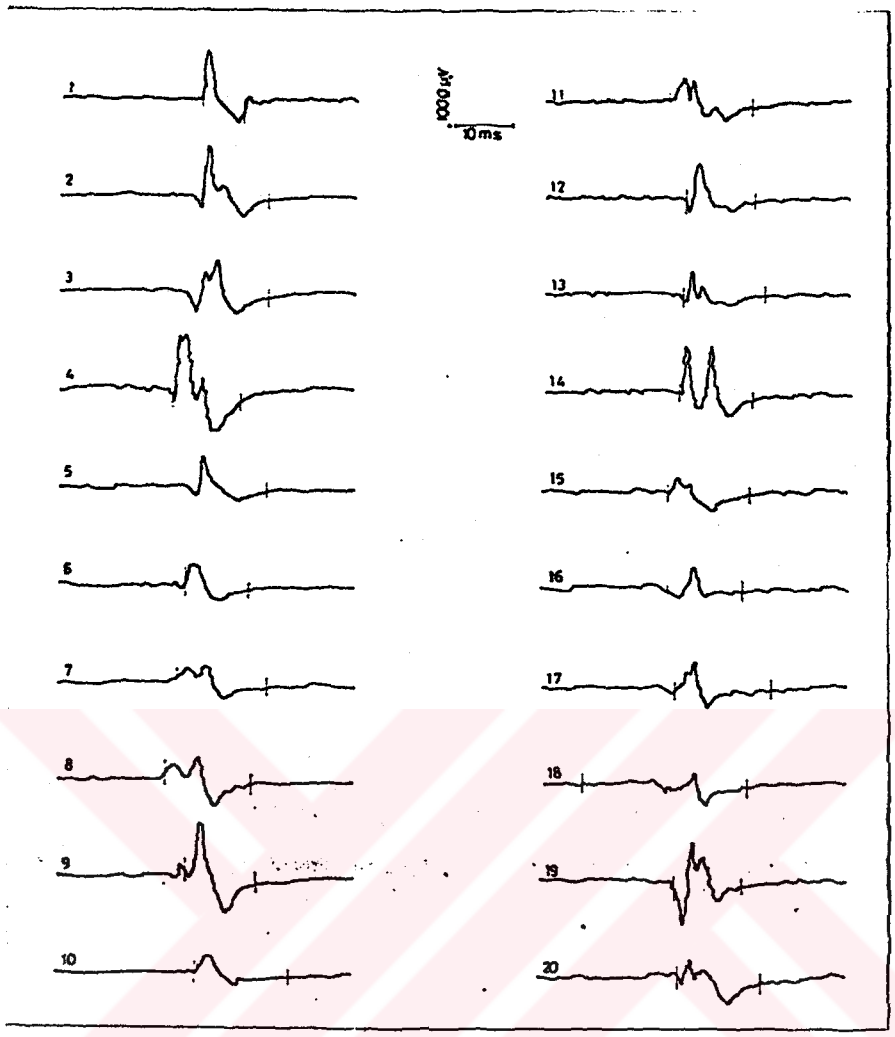


Şekil II.24 Bilgisayar yardımlı makro EMG için entegre edilmiş sistem yaklaşımı.

Bu çalışmada kullanılan öğrenme algoritması hatanın geriye yayılımıdır.Momentum terimi ise 0.9 olarak seçilmiştir.

MMUP bilgileri biseps birachii kasından isteğe bağlı biraz sertçe kasılmalarla meydana gelir.(Şekil II.25 Tablo II.4) Herbir MMUP'dan çıkarılan özellikler şunlardır :

a) **Genlik** :Minimum pozitif tepe ile maksimum negatif tepe arasındaki büyüklük
b) **Alan** : 50 ms analiz zamanı üzerine düzeltilmiş sinyalin integral toplamı
c) **Ortalama güç** : 50 ms üzerine yapılan herbir örneklemenin karelerinin toplamının, örnekleme sayısına bölünmesi
d) **Süre** : Gücün % 90'lık bölümünü içeren MMMUP sinyali parçasının zaman uzunluğu.



Şekil II.25 MND hastalarının MMUP dağişimleri .Kısa dik çizgiler her bir MMUP boyunca analizin başlangıç ve bitim noktalarını belirtir.

41 denekten kaydedilen 820 MMUP bilgisinin dağılımı tablo II.5 'den görülebilir.Şekil II.26'de ise 41 hastanın parametrelerinin 3_D nokta dağılım çizimi görülmektedir.

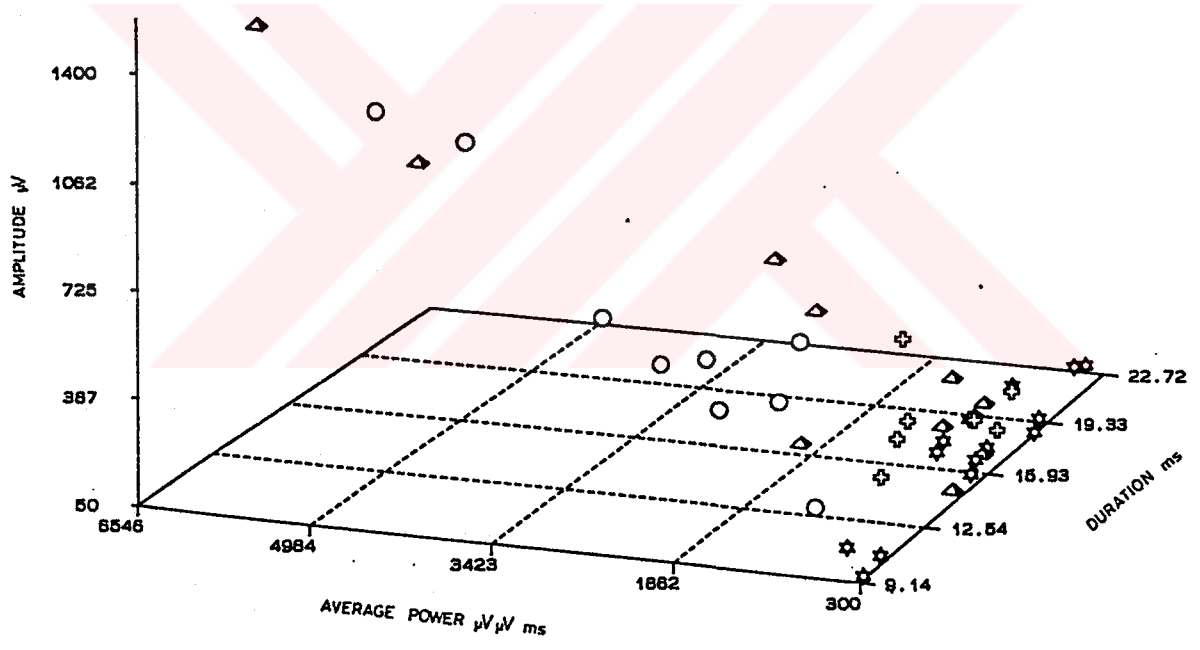
Çalışma gerçekleştirilirken iki farklı yol üzerinde durulmuştur.Bunlar da YNA'nın 8 girişli vektöre mi yoksa 80 girişli vektöre göre mi tasarlanacağıdır.Bu iki ayrı yoldan elde edilen parametreler tablo II.6 ve tablo II.7'ten görülebilir.

No	Amp %V	Area %V	Average Power %Vms	Duration ms
1	1430	3225	3258	11.61
2	1449	4564	3772	11.52
3	1054	4986	3496	13.44
4	1898	7481	10182	10.56
5	848	3233	1480	11 .84
6	632	3190	1424	11.20
7	659	3692	1611	15.68
8	927	4614	2727	14.72
9	1778	6432	7175	11.84
10	566	3048	1009	16.00
11	744	3959	1982	13.12
12	1012	3399	2097	11.84
13	647	2320	810	13.44
14	1351	5109	4633	12.16
15	619	3440	1493	13.12
16	598	2474	795	12.48
17	836	4233	1796	15.68
18	555	3712	1352	27.52
19	1618	5577	4805	11.20
20	831	4322	2398	13.12

Tablo II.4 MND hastalarının MMUP parametreleri

Group	No. of Subjects	Age Range	No. of Macro MUPs
Normal	7	19-46	140
Motor Neuron Disease	9	27-59	180
Becker's Muscular Distrophy	14	17-63	280
Spinal Muscular Distrophy	11	21-72	220

Tablo II.5 MMUP bilgilerinin dağılımı.



Şekil II.26 41 deneğin herbirinin ortalama güç ,esas süre ve esas esas alanının 3_D gösterimi.

Öğrenme ve test etme işlemleri iki ayrı protokole göre yapılmıştır.Birincisinde 20 denek öğrenme kümesini oluşturmak üzere ratgele seçilmiştir, 21 denek ise test kümesini oluşturmuştur.İkinci protokolde 31 denek öğrenme kümesini 10 tanesi ise test kümesini ratlantısal olarak

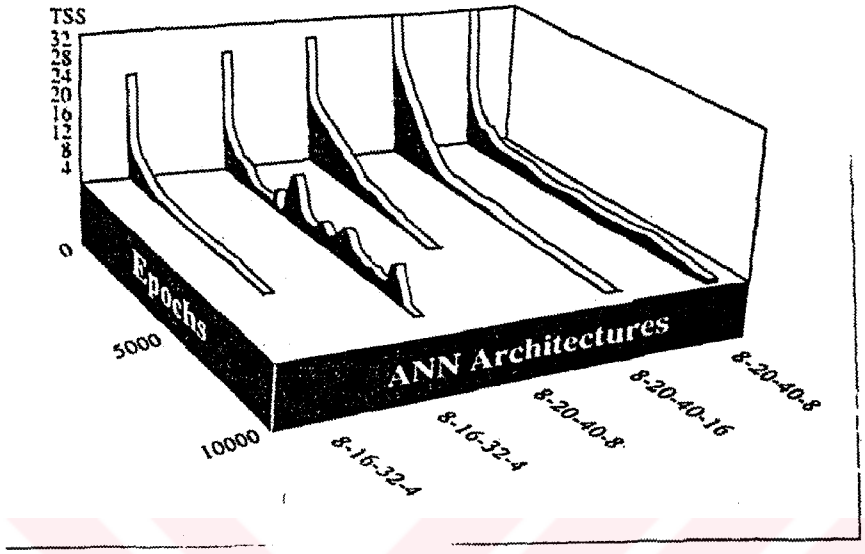
meydana getirir.İlk çalışmalarda gözlenen değerlerin sınırlarına, genlik ,alan, ortalama güç ve süre parametrelerini adapte etmek için bu parametrelerin logaritmik dönüşümü yapılmıştır.Enteresan olarak, bu parametrelerin değiştirilmiş değerleri yerine gerçek değerleri alınırsa sonuç daha verimlidir.

Çıkıştaki teşhisi belirten 4 çıkış ünitesi yeniden bölünerek daha fazla çıkış ünitesi oluşturulmuştur.Tablo II.6'da gisteridiği gibi model 4 ve 17 için 16 adet çıkış ünitesi vardır.Bunlardan 3 tanesi normal ,4'lü MND,3'ü BMD ve 6'sı SMA içindir.8 çıkış üniteli model 5 ayrı tablodan görülebilir.Bu yeniden bölünmeler nöromasküler hasarlardaki elektrofizyolojik karakteristiklerin önemli değişiklikleri yüzünden meydana gelir.

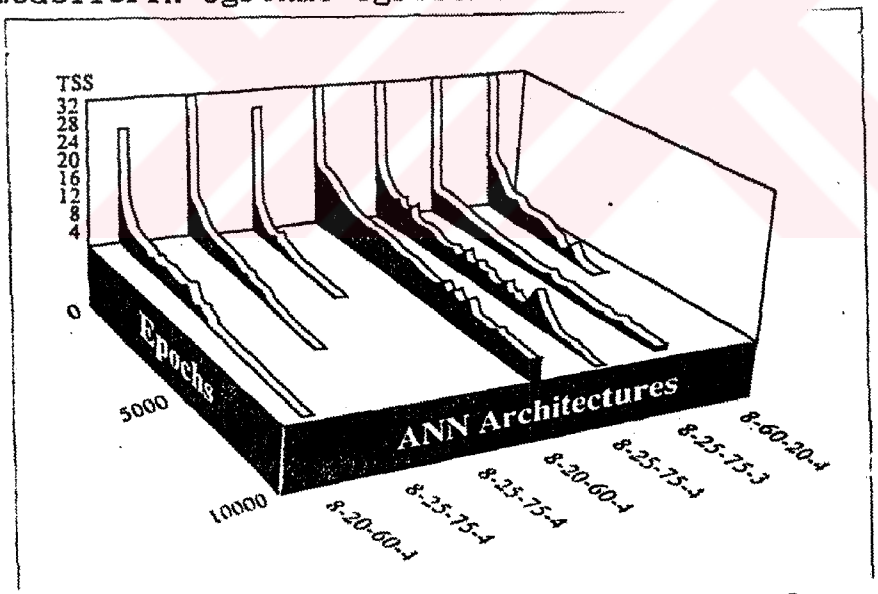
8 girişli modelde parametrelerin yerleştirilmesi açısından yaklaşıldığında ,yerleştirme sırasının bir önemi yoktur.Tersine 80 girişli modelde ,her bir MMUP için parametrelerin tek tek giriş vektörüne yerleştirilmesi zorunludur.Bundan dolayı bu modelde giriş vektörü oluşturmak güçleşir.Bu durum iki yaklaşımla düzenlenebilir. Tablo II.7'deki 1,2,3,5 ve 8. modellerde giriş vektörü herbir hasta için bütün 20 MMUP parametre değerleri seçilerek oluşturulmuştur.Bunlar, [genlik,alan,ortalama güç ,süre] x 20 şeklindedir.Tablo II.7'teki 4,6,7. modellerde ise giriş vektörü [genlik1 ..... genlik20] ,[alan1 ..... alan20] , [ortalama güç1 .....ortalama güç20], [süre1 ..... süre20] şeklindedir.Araştırmacıların burda yapmak istedikleri giriş vektörünü iki ayrı olasılık-yoğunluk fonksiyonu şeklinde belirtmek ve varsa farklılıkları görmektir.

Hastayı görebilmek için TSS terimi tanımlanmıştır.TSS istenen değerden,çıkış değerini çıkartıp karesini almaktır. TSS sonuca yakınsmanın bir ölçüsüdür.Hatanın geriye yayılmalı algoritmanın kullanıldığı bu çalışmada kazancın en yüksek değeri 0.001 en yüksek değeri 0.01'dir.Şekil II.27,28,29 8 girişli modellerin TSS kıstasının iterasyona

bağlı değişimini gösterir.Şekil II.30'de ise 80 girişli modellerin değişimi görülebilir.

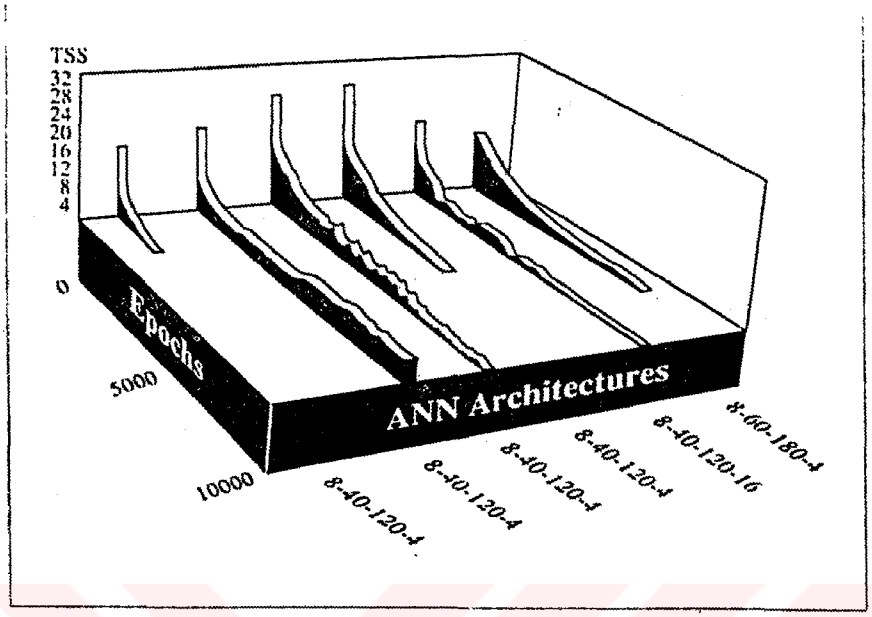


Şekil II.27 Tablo II.6'daki 1'den 5'e kadar olan modellerin öğrenme eğrileri

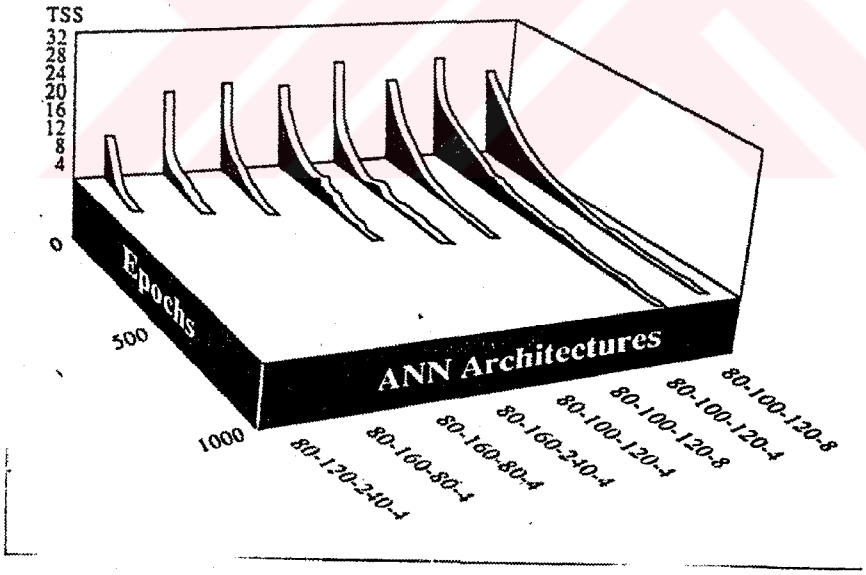


Şekil II.28 Tablo II.6'daki 6'dan 12'ye kadar olan modellerin öğrenme eğrileri.

Bütün şekillerden görüldüğü gibi bu çalışmadaki bütün YNA modelleri 2 gizli tabakaya sahiptir.Bu tamamen probleme özgü bir dizayndır.



Şekil II.29 Tablo II.6'teki 13'ten 18'e kadar olan modellerin öğrenme eğrileri.



Şekil II.30 Tablo II.7'deki 1 'den 8'e kadar 80 girişli modellerin öğrenme eğrileri.

Model	Inputs	L1	L2	Outputs	Gain	Epochs	TSS	Subjects (TS/ES)	Diagnostic Yield % (ES/TS)
1	8a	16	32	4	0.005	3500	1.65	20/21	52/95
2	8l	16	32	4	0.005	10000	1.55	20/21	45/95
3	8a	20	40	4	0.001	6159	0.89	31/10	70/100
4	8l	20	40	16	0.005	20000	1.49	31/10	70/97
5	8a	20	40	8	0.005	20000	2.94	31/10	60/94
6	8l	20	60	4	0.005	10000	0,02	20/21	50/100
7	8l	25	75	4	0.005	7000	0.04	20/21	60/100
8	8a	25	75	4	0.005	2400	0.48	20/21	43/100
9	8l	20	60	4	0.005	25000	1.42	31/10	50/90
10	8l	25	75	4	0.005	11000	0.64	31/10	70/100
11	8l	25	75	3	0.001	15000	1.30	23/7	57/96
12	8l	60	20	4	0.005	5000	0.24	20.21	62/100
13	8a	40	120	4	0.005	1150	0.49	20/21	48/100
14	8a	40	120	4	0.005	17000	4.10	20/21	57/85
15	8l	40	120	4	0.001	17000	0.89	31/10	50/100
16	8a	40	120	4	0.001	5605	0.89	31/10	70/100
17	8a	40	120	16	0.005	10000	1.10	31/10	80/100
18	8a	60	180	4	0.001	7740	0.82	31/10	80/100

Momentum: 0.9
L1: number of units in the first hidden layer
L2: number of units in the second hidden layer
TS: Training Set
ES: Evaluation Set
TSS: Total Sum of Squares
l: logarithmic
a: actual

Tablo II.6 8 girişli yapılar ve sonuçları

Model	Inputs	L1	L2	Outputs	Gain	Epochs	TSS	Subjects (TS/ES)	Diagnostic Yield % (ES/TS)
1	80pl	120	240	4	0.005	180	0.10	20/21	48/100
2	80pl	160	80	4	0.005	110	0.60	20/21	53/100
3	80pa	160	80	4	0.005	110	0.60	20/21	48/100
4	80sa	100	120	4	0.005	750	0.89	31/10	60/100
5	80pa	160	240	4	0.005	400	0.71	31/10	60/100
6	80sa	100	120	8	0.005	534	0.89	31/10	60/100
7	80sa	100	120	4	0.001	1055	0.89	31/10	60/100
8	80pa	100	120	8	0.001	965	0.89	31/10	60/100

Momentum: 0.9
L1: number of units in the first hidden layer
L2: number of units in the second hidden layer
TS: Training Set
ES: Evaluation Set
p: ([Amplitude, Area, Average Power, Duration]) * 20
s: sorted in ascending order: ([Amplitude1...Amplitude20] [Area1,...Area20] [Average Power1...Average Power20] [Duration1...Duration20])
TSS: Total Sum of Squares
l: logarithmic
a: actual

Tablo II.7 80 girişli yapılar ve sonuçları.

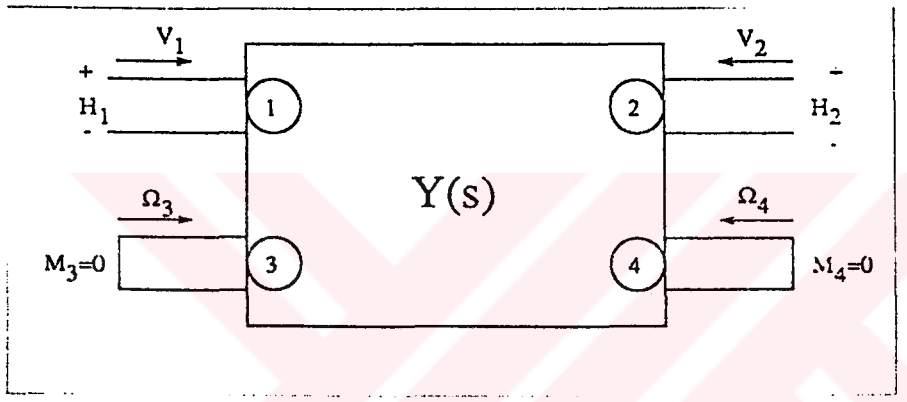
II.6 KEMİK KIRIKLARI İYİLEŞME DEĞERLENDİRİLMESİ

Kemik kırıkları ortopedik cerrahi ile iyileştirilen en genel hasarlardır.Sadece A.B.D.'de yılda ortalama 4.5 milyon bu tür cerrahi müdahale gerçekleştirilmektedir. Tedavi genellikle ve tipik olarak ,kırık bölgesindeki doku ayrımı başarıyla birleşinceye kadar, kırık bçlgenin hareketsiz bırakılmasından ibarettir.Kırılan kemiğin mekanik özellikleri ,iyileşme boyunca ,özellikle sertliği ,dayanımı ve yoğunluğu ile ilgili olmak üzere dramatik olarak değişir.Kırılan organın hareketsizliği ,yeniden kırılma riski olmadan normal ağırlığa dayanacak kadar yeterli sertliğe ulaşana kadar gereklidir.Kemik kırığı iyileşme değerlendirilmesi için teknikler ,radyografik ve klinik değerlendirmeleri içerir.Belirsiz olarak ,özellikle yaşlı hastalarda ,daimi hareket kaybına yol açacak şekilde bu hareketsizlik periyodunun gereksiz olarak çok uzun olabileceği söylenebilir.İşte bu yüzden kırığın iyileşme derecesinin, yeniden kırılma riskini ve gereksiz yere uzamış hareketsizlik süresini minimuma düşürmek için doğru olarak ölçülmesi gerekmektedir.

Kemik kırıklarının iyileşme değerlendirilmesinde birçok teknikler kullanılabilir.Ultrason,kırık bölge radyografisi ve diğer teknikler henüz kırılmış kemik yapısını tem ve doğru bir şekilde ölçebilmiş değildir.Bu uygunsuzluk sözü edilen fiziksel sistemin karmaşık yapısından ileri gelir.İşte bu yüzden YNA kullanarak kemik kırığı iyileştirme değerlendirilmesi ve kırık tespitinin her ikisi için de bazı ortopedistler tarafından kullanılan işitsel duyum tekniği ile aynıdır.Fizyoterapist uzun kemiğin sonuna refleks çekici veya parmağı ile vurur ve diğer taraftan stetoskopla dinler. Sesi kıyaslayarak kırığın olup olmadığını veya iyileşme derecesini saptar.Genellikle zarar görmemiş veya sağlam kemikleri ile iyileşmiş kemikler yüksek frekans bileşenleri içerirken ,az dayanımlılar alçak frekans bileşenleri içerir.Mantık olarak ayrı olmakla birlikte ,YNA ile değerlendirme yapmak için kemiğin frekans cevabını elde edecek bir matematiksel model

geliştirme ihtiyacı vardır.

Kırık kemiğin titreşim cevabı için sayısal model,YNA sinyal işleme işleyişi için dizayn edilmiştir.Model özelleştirilmiş güç girişlerine karşı titreşimsel cevap bilgilerini simüle etmek için kullanılır ve kemiğin titreşimsel cevabındaki değişik faktörlerin etkisini azaltır.Bu faktörler kırık boşluğu karakteristiklerini içerir.Örneğin; yoğunluk, elastiklik modülü,Poisson oranı ve uzunluğu gibi.Böyle bir model Şekil II.31'den görülebilir.



Şekil II.31 Sağlam kemik için YNA modeli.

Kırık kemik çaprazlamasına veya enine titreştirilir.Dört uçlu elektrik devre modeli Timoshenko [23] tarafından geliştirilmiş ve şu dört diferansiyel denklemle özetlenmiştir.

$$\text{(a)}\quad M / EI = -\frac{\tau\,\delta}{\tau\,x}$$

$$\text{(b)}\quad T = A.G.Z\left(\frac{\tau\,y}{\tau\,x} - \delta\right) \qquad \text{(II.17)}$$

$$\text{(c)}\quad \tau\,T / \tau\,x = \sigma\,.\,A\,.\left(\tau^2\,y / \tau\,y^2\right)$$

(d) $T - \tau M/\tau x = \sigma . I . (\tau^2 y / \tau t^2)$

(τ burada kısmi türev işareti olarak kullanılmıştır.)

T ve M parametreleri kesme kuvveti ve momentidir.A çapraz olan kesidini,G kesme modülünü ,Z Timoshenko kesme sabitini ve I çapraz kesitli eylemsizlik momentini gösterir.Ayrıca E young modülü (yoğunluk) ve y kirişin merkez düzleminin yerine geçer ve girişe olan x uzaklığının fonksiyonudur.δ kıvrılmadan mütevellit kiriş çapraz-kesidinin eğimidir.1. eşitlik 4-uçlu elektriksel devre modelini elde etmek için kullanılır.Devre admitans terimleri kullanarak açıklanırsa;

$$V(s) = [\ V_1(s)\ V_2(s)\ \Omega_3(s)\ \Omega_4(s)\ ]^T$$

$$= Y(s)\ F(s)$$

$$= Y(s)\ [\ H_1(s)\ H_2(s)\ M_3(s)\ M_4(s)\ ]^T \qquad (II.18)$$

Burada $V_i(s)$ ve $H_i(s)$ sırasıyla i. uçtaki (i=1,2) çapraz hız ve kesme kuvvetinin Laplace transformasyonudur. $M_i(s)$ ve $\Omega_i(s)$ sırasıyala i. uçtaki (i=3,4) açısal hız ve kıvrılma momentinin Laplace transformasyonudur. Y(s) 4x4 boyutunda 4 uçlu modelin admitans matrisidir ve materyal içeriği ile geometrinin bir fonksiyonudur.

Bu model uniform kemik bölgesinin titreşimsel cevabının simülasyonunda kullanılır.Sınır şartları 0 kıvrılma momentine göre kurulur ve kısace kıvrılma momentini sıfırlamak 3. ve 4. uçları kısa devre yapmak demektir.(Şekil II.31) Sadeleştirilmiş devre, elemanları $Y_{11}(f)$, $Y_{12}(f)$, $Y_{21}(f)$, $Y_{22}(f)$ olan kompleks 2x2 admitans matrisi $Y_m(f)$ ile karakterize edilir.Laplace değişkeni s, jw olarak yeniden yer alır ve f= $w/2\pi$ 'dir. Y_{ij} gösteriminde i ve j indisleri i. satırın j. sütununu gösterir.$Y_{ij}(f)$, çapraz hız $v_i(t)$ 'nin Fourier transformu $v_i(f)$ ile çapraz kuvvetin Fourier transformu $H_j(f)$'nin oranıdır.(i<>j).i ve j indisleri ölçme sahasındaki i ve j uçlarını belirtir.(i,j = 1,2) $Y_{11}(f)$ örneğin 1. uçtaki yük ve ikinci uçta kuvvet sıfırsa ;

$Y_{11}(f) = V_1(f) / H_1)$ 'dir.

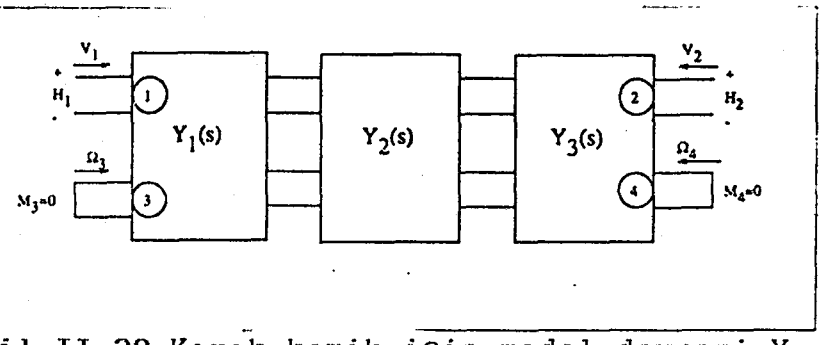
Bu ilişkiler şu eşitlikler kümesi ile ifade edilebilir.

$$V_1(f) = Y_{11}(f)\, H_1(f) + Y_{12}(f)\, H_2(f)$$

$$V_2(f) = Y_{21}(f)\, H_1(f) + Y_{22}(f)\, H_2(f) \qquad \text{(II.19)}$$

Simülasyon çalışmaları için ,sadeleştirilmiş $Y_m(f)$ matrisi ,cebirsel işlemlerle 4x4 admitans matrisinden sağlanır.Daha önce açıklandığı gibi sınır şartları bu işlem için kullanılır.Sınır şartlarına uygun olmak üzere Şekil II.32 'deki gibi kırık kemik modeli ,4 uçlu devrenin üçkez kaskad bağlanmasıyla oluşturulabilir.Bu üç bölümün herbiri kemikteki sağlam ,kırık ve yine sağlam bölgeleri sırasıyla belirtir.$Y_m(f)$ matrisi de benzer yoldan bulunabilir.Konunun pasifliği ve lineerliği yüzünden sadeleştirilmek devre daima simetriktir.(Y_{12}=Y_{21})

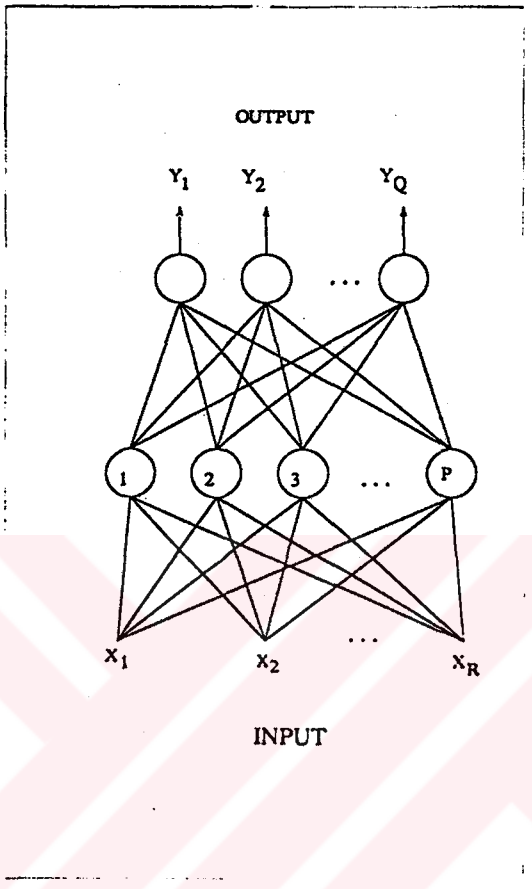
Bu çalışmada karakter tanıma problemlerinde yaygın olarak kullanılan ileri beslemeli ağ kullanılmştır.Bunun sebebi, giriş ve çıkış uzaylarının belli özelliklere yoğunlaşmasıyla uygun bir karakter tanıma performansı sağlanmasıdır.YNA yapısı Şekil II.33'ten görülebilir. Öğrenme kuralı olarak momentum terimsiz genelleştirilmiş delta kuralı kullanılır.



Şekil II.32 Kırık kemik için model devresi.Y_2 kırık bölge boşluğunu temsil eder.

Başlangıç noktası admitans değerlerinin kümesi olan işlenmemiş bilgilerdir.[$\{Y_{11}(f_j)$,$Y_{12}(f_j)$,$j=1,...M\}_i$ ve $\{Y_{11}(f_j)$,$Y_{12}(f_j)$,$j=1,...M\}_f$].Burada i ve f sırasıyla sağlam ve kırık organ bilgisine bağlıdır.Burada birtakım

kabuller yapmak zorunludur.Y11(f) ve Y12(f) sırasıyla bütün pratik hallerde doğrudur.Daha ileri olarak ,bu çalışmada admitansların uygun işleyiş yöntemiyle anlık kuvvet ve hız-zaman bilgilerinden elde edilebileceği kabul edilmiştir.

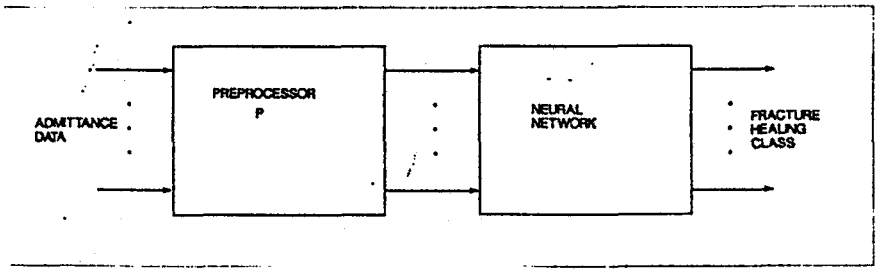


Şekil II.33 İleri beslemeli YNA .R'ler giriş Q'lar çıkış indisleridir ve gizli tabaka P adet gizli üniteden oluşur.

Çıkış y_j ,(j=1,...,R) ve $0 \leq y \leq 1$ ve $\Sigma y_j=1$ için "R" kanallı YNA'nın özel iyileşme hallerini temsil eder.Bu çalışmada iyileşmenin 4 halini temsilen R=4 alınmıştır.Yani YNA çıkışında ,kırık boşluğu sertliğinin sağlam kemik sertliğine oranı olan E_f/E_i 'nin dört değerini temsil eden dört işlem elemanı bulunur.Bunlar $E_f/E_i < 10^{-3}$, $10_{-3} \leq E_f/E_i < 10_{-2}$, $10_{-2} \leq E_f/E_i < 10_{-1}$, $10_{-1} \leq E_f/E_i < 1$ oranlarıdır.

YNA için girişler ya admitansların kendileri ya da ön-işleme tabi tutulmuş versiyonlarıdır.Birkaç ön-işlem organizasyonu ,YNA sınıflama hatasına olan etkileri sebebiyle oluşturulmuştur.Bu ön-işleme operasyonu Şekil

II.34'te P ile temsil edilmiştir.



Şekil II.34 Kırık iyileşme sınıfları için karakter tanıma işleyiş şeması.P ön işleyicisi anlık ağ girişleri sağlar.

Tablo II.8 satır admitans bilgisine uygulanan özel operasyonların llistesidir.Bütün durumlarda frekans ekseni 80 Hz'lik eşit aralıklı ve 50 Hz'ten 900 Hz'e kadar M=11 eşit aralığı bölünür.Herbir frekans bölgesindeki bu admitansların ortalama değeri alınır ve işleme boyunca kullanılır.İlk üç ön-işleme operasyonu sadece admitans bilgilerini kullanır.P-4,P-5 ve P-6 ön-işleme operasyonları için LTF olarak adlandırılan ve yük-transfer-faktörü olarak belirtilen bir terim tanımlanır.LTF ideal kuplaj şartları altında ,giriş ve çıkış arasında ölçülen max ortalama güçtür ve şöyle verilir;

$$LTF(f) = \max \{ P_{av}^{out} / P_{av}^{in} \}$$
$$= A(f) - (A(f)^2 - 1)^{0.5} \qquad (II.20)$$

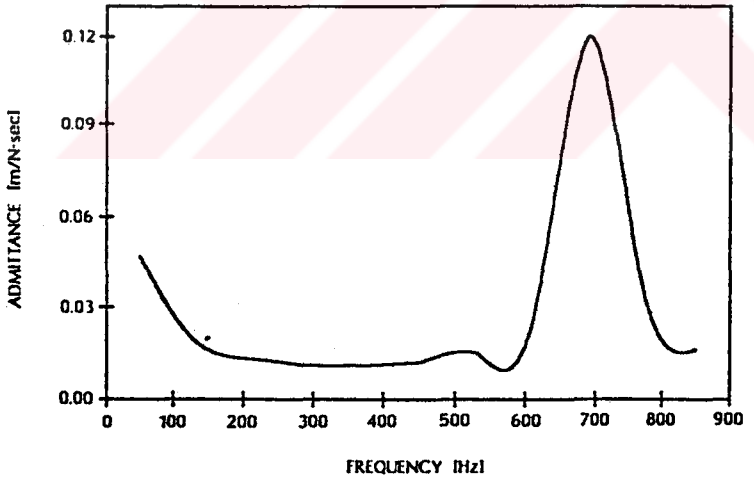
Burada P_{av} ortalama mekanik güçtür ve $A(f)=\{2\ Re[Y_{11}(f)]\ Re[Y_{22}(f)]-Re[Y_{12}(f)^2]\}/Y_{12}(f)Y_{12}(f)^*\}$ şeklindedir.* indisi kompleks-konjüge durumu gösterir.Not olarak LTF frekansın gerçel sayılar üzerinde tanımlanan bir fonksiyonudur ve $0 < LTF < 1$ 'dir.Sonuç olarak A(f) 'e 50-1000 Hz aralığında değer biçilir ve ağa tek giriş olarak gönderilir.

Şekil II.35'te $Y_{12}(f)$'nin 10-1000 Hz frekans aralığında değişimi görülmektedir.Şekil II.36'da ise kırık bir kemik için $Y_{12}(f)$'nin değişimi görülmektedir.

Preprocessing Type	Output of Preprocessor	Number of Outputs*
P-1	$[\{\|Y_{11}\|;\|Y_{12}\|\}_f$ and$\{\|Y_{11}\|;\|Y_{12}\|\}_i]$	4M
P-2	$[\{\|Y_{12}\|\}_f;\{\|Y_{12}\|\}_i]$	2M
P-3	$[\|Y_{12}\|_f/\|Y_{12}\|_i]$	M
P-4	$[LTF_f; LTF_i]$	2M
P-5	$[LTF_f/LTF_i]$	M
P-6	$[\overline{LTF_f}/\overline{LTF_i}]$	1

*M = number of individual frequency points

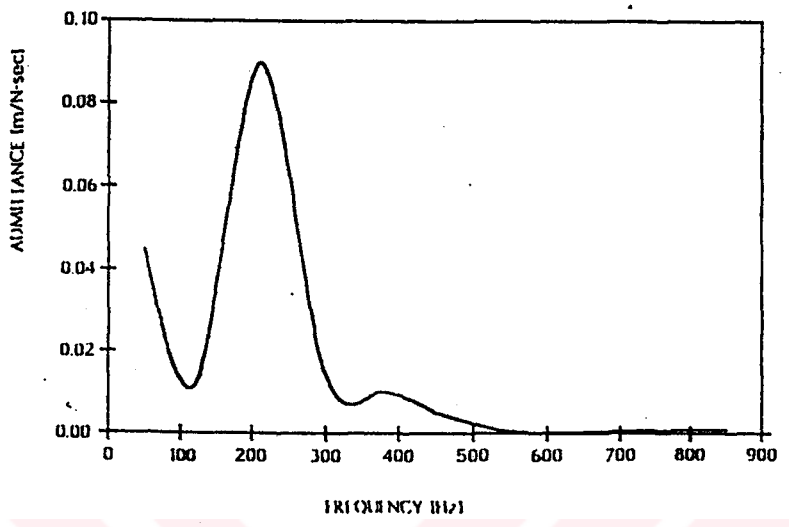
Tablo II.8 Ön-işleme operasyonları



Şekil II.35 Sağlam kemik için $Y_{12}(f)$ 'in değişimi

Herbir mekanik sertlik sınıfı için 25 olmak üzere 100 farklı karakter içeren bir bilgi seti oluşturulumuştur. Karakterler YNA'da ,öğrenme kümesinden rastgele seçimlerle alınarak oluşturulmuştur.Diğer bir 100 karakterlik bigi seti de ağı test etmek ve performansını ölçmek için

kullanılmıştır.YNA bu test setindeki bilgileri daha önce görmemiştir.Tablo II.9 'da ön-işleme operasyonları için sınıflama sonuçları yer alır.Bu tablodan ağın yakınsaması için gerekli iterasyon sayısını da görmek mümkündür.

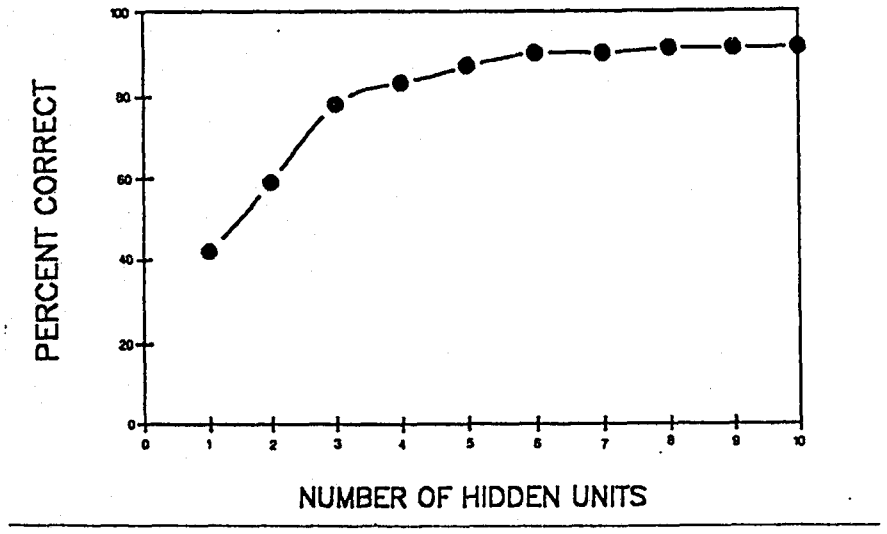


Şekil II.36 Kırık kemik için $Y_{12}(f)$ 'nin değişimi

Gizli tabakadaki ünite sayısındaki değişimin ağın doğruluğuna etkisi olduğu bilinmektedir.Bu konu araştırılmış ve Şekil II.37'den görüldüğü gibi P-3 ön-işleme operasyonu için 6 gizli ünite optimum sayı olarak gözükür.

Preprocessing Type	Iterations for Convergence	Classification Accuracy (Training vs. Running)	Number of Hidden Units
P-1	2900	88/79	25
P-2	1341	91/85	18
P-3	654	94/90	6
P-4	921	93/89	12
P-5	215	98/96	6
P-6	75	52/48	3

Tablo II.9 Ön-işleme operasyonlarına göre YNA performansı.



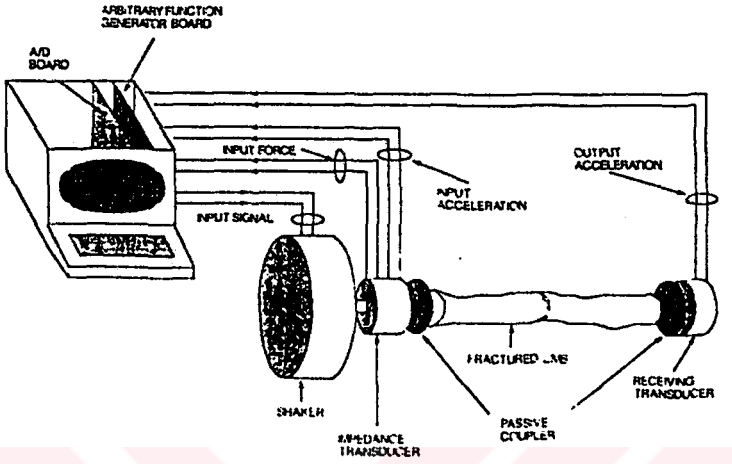
Şekil II.37 Gizli ünite sayısının sınıflama doğruluğuna etkisi (P-3 ön-işleme operasyonu için)

Transfer admitans bilgilerini elde etmek için kullanılan deneysel düzenek Şekil II.38'de gösterilmiştir.Mekanik enerji kemiğe ,bir empedans dönüştürücü ile akuple edilmiş mekanik sarnıcı tarafından uygulanır.Empedans dönüştürücü giriş ivmesini ve kuvvetini ölçer.Çıkışta bir alıcı-dönüştürücü vardır ve çıkış ivmesini ölçer.Çıkış ivmesi 1V/g hassasiyeti ile ölçülür.Bu ölçme kemiğin sonundaki çıkış-ivme ölçüsünden sağlandığı için kırık bölge giriş-sarsıcı ile çıkış dönüştürücü arasında kalır.Giriş ve çıkış dönüştürücülerinin herbirinin aynı pozisyonu ile yapılan ölçmeler sağlam kemik içinde gerçekleştirilir.Aynı anda sağlanan bu üç bilgi kanalı (giriş kuvveti, giriş ivmesi, çıkış ivmesi) bir mikrobilgisayara ve kanal başına 10 kHz örnekleme oranıyla işleyen A/D çeviriciyle gönderilir. Titreşim-sarsıcı sinüsoidal dalga kullanarak enerjilendirilir.Ör;

$$v(t) = A \sum_{j=1}^{K} \mathrm{Sin}\,(2\pi j f_0 t + \Theta_j)$$

burada f_0=40 Hz ,Θ_j fazları j=1...K olmak üzere $[-\pi,\pi]$ aralığında rastgele değişir ve K=25'tir.Zaman bilgisi

manyetik diskte depo edilir ve Fourier transformu ile Y_{11}, $Y_{12}=Y_{21}$ bilgileri elde edilir.Bu method Y_{22}'yi elde edemez. Bundan dolayı LTF ön-işleme operasyonu bu deneyle birlikte kullanılamaz.Fakat teknik modifike edilerek Y_{22}'de bulunabilir.



Şekil II.38 Transfer admitans bilgisi sağlamak için oluşturulmuş deney düzeneği.

ÜÇÜNCÜ BÖLÜM
UYGULAMA

III.1 YNA KULLANARAK DİYET PROGRAMI ÇALIŞMASI

III.1.1 AMAÇLAR

Tıp alanındaki araştırmalar insanın boy ve yaşına göre ideal kiloda olmasının sağlıklı bir hayat sürmesi bakımından önemli olduğunu ortaya çıkarmıştır.Ayrıca sebepleri ve tedavisi çok karmaşık olan şeker ve yüksek tansiyon gibi hastalıklarda da kişinin ideal kilosunda olup olmaması büyük önem arzeder.Böyle bir düşünüşle ,bu bölümde insanın cinsiyet,boy ve yaşına göre ideal kilosunu hesaplayan bir YNA programının prototipi hazırlanmaya çalışılmıştır.Diğer taraftan I. ve II. bölümlerde anlatılanların bir program üzerinde görülmesi amaçlanmaktadır.

Bu YNA programı boyu 1.50 cm dolaylarında ,yaşı 15-55 arasında olan kadınlar ve erkekler için ideal kiloyu ve dinlenme halindeki günlük kalori gereksinimlerini hesaplamaya çalışır.Bunun için ilk önce yaş,boy,cinsiyete bağlı olarak ideal kilo bilgisi [26] ve günlük dinlenme kalorisi katsayısı elde edilmiştir.[24] Günlük dinlenme kalorisinin hesaplama ihtiyacı özellikle yüksek tansiyon hastalarına uygulanan diyetler için bir referans nokta oluşturma düşüncesinden ileri gelmektedir.

III.1.2 YÖNTEM

Program gerçekleştirilirken ilk adım olarak cinsiyet ,boy,kilo ,yaş ve kalori değerlerinin 0 ila 1 arasında normalizasyonu yapılmıştır.Her boy bilgisini 1000'e kilo ve yaş bilgilerini 100'e kalori bilgisini 10000'e bölmek suretiyle bu işlem gerçekleştirilir.Cinsiyet bilgisi ise kadınlar için 0.1 erkekler için 0.9 sayılarıyla temsil edilmiş ve ağa uygulanmıştır.

Günlük kalori ihtiyacı şu şekilde bulunur;

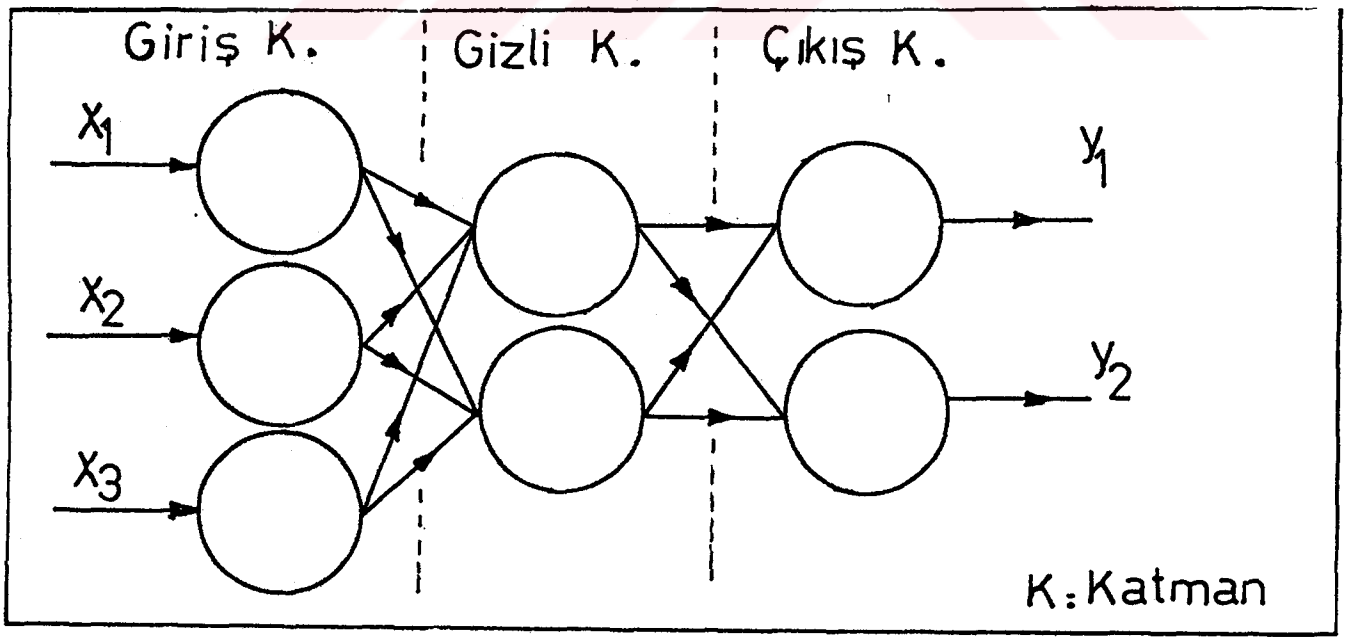
$$G.K.İ. = 24 \cdot a \cdot G \qquad (III.1)$$

Burada a [kcal/saat.kg] boyutunda bir katsayıdır.G ise

kişinin ideal kilosunu temsil eder.Paremetre a dinlenme hali için belirlenmiş bir değerdir ve 1.13 [kcal/saat.kg]'a eşittir.

Program sonuçlarında elde edilen sayılar için, her bilgi hangi sayıya bölündü ise yine aynı sayıyla çarpılarak denormalizasyon gerçekleştirilir ve kullanıma uygun büyüklükler rahatça bulunabilir.Boy bilgileri [cm] boyutundadır.

Programda gerçekleştirilen YNA yapısı Şekil III.1 'de gösterilmiştir.Üç tabakadan oluşan yapının giriş tabakasında 2 ünite bulunur.Giriş tabakasındaki üniteler, kişinin yaşını ,boyunu ve cinsiyet bilgisini temsil eder. Çıkış üniteleri ise bu girişlere göre programın hesapladığı ideal kilo ve günlük dinlenme kalorisini gösterir.Deneyler sonucu optimal çıkışın 2 gizli üniteyle sağlandığı sonucuna verilmiştir.Programda kullanılan eğitme algoritması supervised eğitme , öğrnme kuralı ise hatanın geriye iletimi (back propagation) yöntemidir.Program Yıldız Teknik Üniversitesi Lojik Devre Labaratuvarında 80486 PC'de uygulanmıştır[25].5000 iterasyon için yaklaşık öğrenme süresi 30 dk. civarındadır.



Şekil III.1 YNA yapısı.x_1,x_2,x_3 girişleri (boy,yaş, cinsiyet), y_1,y_2 çıkışları (ideal kilo,kalori) temsil eder.

Programda ağırlıklar genelleştirilmiş delta kuralı uygulamasında herbir ünitede kullanılan transfer fonksiyonu sigmoiddir.Gizli ünitedeki j. işlem elemanı için çıkış şöyle verilir.

$$O_{pj} = \frac{1}{1 + e^{-(netp)}} \qquad \text{(III.2)}$$

$$netpj = \Sigma_i W_{ji} O_{pi} + Q_j$$

İndis i giriş tabakası sayısına eşittir.Q_j ise j. gizli ünitenin kutup değeri olarak tanımlanan ve başlangıçta 0.03 değerine sahip bir sayıdır ve iterasyonlar boyunca değişir.W_{ji} bir önceki tabakanın i. ünitesine karşı, gizli tabakadaki j. ünitede belirlenmiş ağırlıktır.O_{pi} bir önceki tabakadaki eleman çıkışlarıdır.

Gizli ve çıkış tabakasındaki ağırlıklar şu formülle ayarlanır.

$$W_{ji}(t+1) = \epsilon . A_{pj} . O_{pi} + \alpha . W_{ji}(t) \qquad \text{(III.3)}$$

Burada ϵ A öğrenme sabiti ve α ise momentum terimidir.Bu programda α=0.9 ve ϵ=0.2 seçilmiştir.A_{pj} terimi ise gizli ve çıkış üniteleri için ayrı ayrıdır.

Gizli ünite için : $A_{pj} = O_{pj} (1-O_{pj}) \Sigma_k A_{pk} . W_{kj}$ (III.4)

Çıkış ünitesi için : $A_{pj} = (t_{pj}-O_{pj}) O_{pj}(1-O_{pj})$

III.3 denklemindeki O_{pi} ünite girişini temsil eder.t_{pj} terimi ise arzu edilen ünite çıkışıdır.Ağırlık $W_{ji}(t)$ bir önceki iterasyondaki değerdir.

Program Pascal programlama diliyle yazılmıştır.Dikkat edilirse matris yapısı yüzünden giriş,gizli ve çıkış ünitelerinin sayısı ve ϵ,α değerleri çok kolaylıkla

değiştirilebilir.Kısaca aynı program değişik problemleri çözmekte de rahatlıkla kullanılabilir. Arzu edilen çıkış değerlerinin "dess" matrisiyle girildiği ve süpervised öğrenme için programda kullanıldığı açıkça görülebilir.

III.1.3 SONUÇ VE TARTIŞMA

YNA çıkışları ve arzu edilen değerler karşılaştırıldığında ,bütün öğrenme boyunca yapılan hatanın (error-t) iterasyon sayısıyla ters orantılı olduğu görülür.Ağ global olarak az hatalı çalışmasına rağmen lokal olarak özellikle kalori çıkışında hatalar yapmıştır.Örneğin gerçek değeri 1450 kCal olması gereken 2. kalori çıkışı 1500 kCal olarak ağ tarafından hesaplanmıştır.Aynı çıkarım 8'den 17'e kadar olan kalori çıkışları için de gösterilebilir.

Çıkış numarası	(kCal) Gerçek değer	(kCal) Hesaplanan değer	Hata
10	1450	1510	+ %4
16	1640	1570	- %4
17	1670	1580	- %5

Sonuç olarak program % 90 civarında bir doğrulukla çalışmaktadır.Bu doğruluk oranı kötü bir oran değildir ve iterasyon sayısını daha da arttırmakla hatanın çok düşeceğine dair kanıtlar vardır.

Hatayı düşürmek için düşünülebilecek diğer alternatşfler α ve ϵ sabitleriyle oynamaktır.Gerçekten de literatürde ,kullanıcı tecrübesine dayalı olarak ,bu sabitleri değiştirmek suretiyle daha başarılı sonuçlar alınacağı söylenmektedir.Kesin olan α'nın ϵ'dan büyük olması gerektiğidir.

Önemli olan diğer bir nokta ,III.3 ve III.4 denklemlerinin MES(karesel ortalama hata) kuralından türetildiğini belirtmektir[3].

Belirtmelidir ki; program bu haliyle mükemmel değildir.

Özellikle α ve ϵ sabitleri konusunda ayrıntılı çalışmalar yapılarak ve daha yüksek doğruluk oranlarına ulaşmak mümkündür.

REFERANSLAR

1.Eberhart R.C. , Dobbins R.W, "Early Neural Network Development History", IEEE Engineering in Medicine and Biology, pg.15 , September 1990

2.Rumelhart DE,Mc Cleland JL,"Paralel Distributed Processing Vol I-II", MIT Press ,Cambridge Mass 1986

3.Robert Hect-Nielsen ,"Neuro Computing ", Uni. of California ,1989

4.Kohonen T., "Self Organization and Assosiative Memory "

5.Iwata A.,Nagasaka Y.,Suzumura N., "Data Compression of the ECG using Neural Network for Digital Holter Monitor", IEEE Engineering in Medicine and Biology, p. 53 , September 1990

6.Rumelhart DE,Hınton GE and Williams RJ ," Learning Representations by Back Propagating Errors Nature" , 323:9:533, 1986

7.Rumelhart DE,Hinton GE ve Williams RJ ,"Learning Internal Represantations by Error Propagation" ,Paralel Distributed Processing Vol 1 , p.318-363, 1986

8.Cios J.K ,Chen K., Langerderfer A.R.,"Use of Neural Networks ın Detecting Cardiac Diseases from Echocardiographic Images", IEEE Engineering in Medicine and Biology " ,p.58 , September 1990

9.Olshansky B,Collins SM, Skorton DJ, "Variation of Left Ventricular Mycardial Gray Level in Two-Dimensions as a Result of Cardiac Contraction

10.Skorton DJ,Melton HE,Pandian NC,"Detection of Acute Mycardial Infarction in Closed Chest Dogs by Analysis of Regional Two-dimensional Echocardiographic Gray Level Distribution ",Circulation 52 :36 ,1983

11.Anthony P,Chandraratna N,Ulene R,Nimalasuriya Aetal, "Differantiation Between Acute and Healed Mycardial

İnfarction by Signal Averaging and Color Encoding of Two-dimensional Echocardiography", Am J.of Cardiology 56:383 ,1985

12.Parisi AF,Nieminen MO,Boyle JE et al, " Enhanced Detection of the Evaluation of Tissue Changes After Acute Mycardial İnfarction Using Color-Encoded Two-dimensional Echocardiography", Circulation 66:764, 1982

13.Cios KJ,Goodenday LS,Langerderfer AR, Merhi M,"Neural Network in Detection of Coronary Artery Disease", IEEE Computer Society Press, in Press, 1990

14.Poli R,Cagnoni S, Livi R, Coppini G, Valli G,"A Neural Network Expert System for Diagnosing and Treating Hypertension",IEEE Transaction on Computer, p:64 March 1991

15.Basmajian JV,Deluca CJ,"Muscle ,Alive ,Their Functions Revealed by EMG", Baltimore ,MD,Williams and Wilkins,1985

16.Kelly FM,Parker AP,Scott NR,"The Application of Neural Networks to Myoelectric Signal Analysis :A Preliminary Study",IEEE Transaction on Biomedical Engineering,p.221 ,March 1990

17.Graupe D,"Functional Separation of EMG Signals via ARMA Identification Methods for Prothesis Control Purposes", IEEE Trans.Sys.Man Cybern. vol SMC-5,p.252,March 1975

18.Tank DW,Hopfield JJ,"Simple Neural Optimization Networks; an A/D converter ,signal decision circuitry and lineer programming circuit", IEEE Trans. Circuits Syst. Vol CAS-33 ,p.533-541,1986

19.Korurek M,"Tıp Elektroniğinde kullanılan kuvvetlendiriciler ve dönüştürücüler ",İTÜ Yayınları, s.47 ,1988

20.Schizas CN,Pattichis LT,Schofield IS,Fawcett PR, Middleton LT,"Artificial Neural Nets in Computer-Aided Macro Motor Unit Potential Classification", IEEE Engineering in Medicine And Biology,p.31,September 1990

21.Stalberg E,"Macro EMG; A New Recognition Technique ", J. Neural Neurosurg. Psych, 43:475-482,1980

22.Kaufman JJ,Chiabrera A,Hatem M,Hakim NZ,Figuerido M, Nasser P,Lattuğa S,Pilla AA,Siffert RS,"A Neural Network Approach for Bone Fracture Healing Assesment", IEEE Engineering in Medicine and Biology,p.23-30,September 1990

23.Graf RF,"Wave Motion in Elastic Solids",Ohio State University Press,Columbia OH,1975

24.Sencer E,"Beslenme ve Diyet",İstanbul,1991

25.Gürgen F,"Nöral Ağlarla Hesaplama Dersi Projesi", İstanbul 1992

26.Marmara Üniversitesi Diyet Bölümü dokümanı,İstanbul 1992

ÖZGEÇMİŞ

1969 yılında Adapazarı'nda doğdum.İlk, orta ve lise tahsilimi burada tamamladıktan sonra 1986 yılında Yıldız Teknik Üniversitesi Elektrik Mühendisliği Bölümünde öğrenimime devam ettim.1990 yılı Haziran döneminde mezun oldum.Aynı yılın Ekim ayında Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsünde yüksek lisans öğrenimine başladım.

İlgilendiğim alanlar, Biyomedikal Mühendisliği ,Lojik Devre ve Güç Elektroniği konularıdır.

Elk.Müh. Cezmi Ateş

EK_1

```
ROGRAM Yıldız_Net;
$N+,E+}
ONST
ayers_Max      = 3;       { Maximum number of layers }
odes_Max       = 3;       { Maximum number of node in any layer }
utputs_Max     = 18;       { Number of input combinations }
nput_Nodes     = 3;       { Number of input layer nodes }
utput_Nodes    = 2;       { Number of output layer nodes }
idden_Nodes    = 2;       { Number of hidden layer nodes }
psilon         = 0.2;
lpha           = 0.9;
terative_Max   = 5000;  { Maximum iteration number }
very           = 500;   { After every iteration print results }
AR
, ii, j, count, dcount, isylla, num : integer;
ps, alp, err, error, terr, terror, total_rate: double;
ayers, Inputs, Outputs : integer;
ode       : array [0..Layers_Max-1] of integer;
atrix     : array [0..Outputs_Max-1, 0..(Input_Nodes+Output_Nodes-1)] of doubl
;
atrix_tmp: array [0..trunc(Iterative_Max/Every), 0..Outputs_Max-1,
                  0..(Input_Nodes+Output_Nodes-1)] of double;


es      : double;
ate     : array [0..Outputs_Max-1] of double;
ut      : array [0..Layers_Max-1, 0..Nodes_Max-1] of double;
er      : array [0..Layers_Max-1, 0..Nodes_Max-1] of double;
heta    : array [0..Layers_Max-1, 0..Nodes_Max-1] of double;
theta   : array [0..Layers_Max-1, 0..Nodes_Max-1] of double;
eight   : array [0..Layers_Max-1, 0..Nodes_Max-1, 0..Nodes_Max-1] of double;
weight  : array [0..Layers_Max-1, 0..Nodes_Max-1, 0..Nodes_Max-1] of double;
ess     : array [0..Outputs_Max-1,0..Output_Nodes-1] of double;

t1: string[5];
t2, st3, st4: string[10];
utfile : text;


rocedure Wrand;
 Initializes node offsets & weights at the beginning of the program }
ar
i, ni, ni_end, no, no_end : integer;
rand48, power: double;
egin
 power:=1.0;
 for li:=1 to 31 do
     power:=power*2;
 power:=power-1;
 { Randomize offsets }
 for li:=1 to Layers_Max-1 do
 begin
   ni_end:=node[li];
   for ni:=0 to ni_end-1 do
   begin
```

```
    drand48:=Random(30)/100;
    theta[li,ni]:= drand48;
    dtheta[li,ni]:=0;
  end;
end;

{ Randomize weights }
for li:=1 to Layers_Max-1 do
begin
  ni_end:=node[li];
  no_end:=node[li-1];
  for ni:=0 to ni_end-1 do
      for no:=0 to no_end-1 do
      begin
        drand48:=Random(100)/100;
        weight[li,ni,no]:=drand48;
        dweight[li,ni,no]:=0;
      end;
end;
id;  { procedure WRAND }


ocedure Initial;
sets zero the output of each node before every iteration }
IR
 n, n_end: integer;
gin
for l:= 0 to Layers-1 do
begin
  n_end:= node[l];
  for n:= 0 to n_end-1 do
      out[l,n]:= 0;
end;
id;    { procedure INITIAL }


ocedure yread ( isylla: integer );
read input pattern                                          }
gin
case isylla of
     0: begin out[0,0]:=0.150; out[0,1]:=0.15; out[0,2]:=0.1 end;
     1: begin out[0,0]:=0.150; out[0,1]:=0.20; out[0,2]:=0.1 end;
     2: begin out[0,0]:=0.150; out[0,1]:=0.25; out[0,2]:=0.1 end;
     3: begin out[0,0]:=0.150; out[0,1]:=0.30; out[0,2]:=0.1 end;
     4: begin out[0,0]:=0.150; out[0,1]:=0.35; out[0,2]:=0.1 end;
     5: begin out[0,0]:=0.150; out[0,1]:=0.40; out[0,2]:=0.1 end;
     6: begin out[0,0]:=0.150; out[0,1]:=0.45; out[0,2]:=0.1 end;
     7: begin out[0,0]:=0.150; out[0,1]:=0.50; out[0,2]:=0.1 end;
     8: begin out[0,0]:=0.150; out[0,1]:=0.55; out[0,2]:=0.1 end;
     9: begin out[0,0]:=0.150; out[0,1]:=0.15; out[0,2]:=0.9 end;
    10: begin out [0,0]:=0.150; out[0,1]:=0.20; out[0,2]:=0.9 end;
    11: begin out [0,0]:=0.150; out[0,1]:=0.25; out[0,2]:=0.9 end;
    12: begin out [0,0]:=0.150; out[0,1]:=0.30; out[0,2]:=0.9 end;
    13: begin out [0,0]:=0.150; out[0,1]:=0.35; out[0,2]:=0.9 end;
    14: begin out [0,0]:=0.150; out[0,1]:=0.40; out[0,2]:=0.9 end;
```

```
      15: begin out [0,0]:=0.150; out[0,1]:=0.45; out[0,2]:=0.9 end;
      16: begin out [0,0]:=0.150; out[0,1]:=0.50; out[0,2]:=0.9 end;
      17: begin out [0,0]:=0.150; out[0,1]:=0.55; out[0,2]:=0.9 end;


  end;
end;   { procedure YREAD }


function sigmoid( x: double ): double;
{ sigmoid function }
var
y: double;
begin
  y:=1/(1+exp(-x));
  sigmoid:=y;
end;   { function SIGMOID }


procedure wforward;
{ wforward propagation }
var
li, ni, ni_end, no, no_end: integer;
x, th: double;
begin
  for li:=1 to Layers-1 do
  begin
    ni_end:= node[li];
    no_end:= node[li-1];
    for ni:= 0 to ni_end-1 do
    begin
      x:= 0;
      for no:= 0 to no_end-1 do
          x:= x + weight[li,ni,no] * out[li-1,no];
      out[li,ni]:= sigmoid( x-theta[li,ni] );
    end;
  end;
end;    { procedure FORWARD }


function back: double;
{ back propagation }
var
li, ni, ni_end, no, no_end: integer;
e, err, x, d, w, y: double;
begin
  err:=0;
  for ni:=0 to outputs-1 do
  begin
    y:= out[2,ni];
    des:= dess[isylla,ni];
    e:= des - y;
    der[2,ni]:= e*y*(1-y);
    err:=err+ e*e;
  end;
```

```
  for li:=2 downto 1 do
  begin
    no_end:= node[li-1];
    ni_end:= node[li];
    for no:= 0 to no_end-1 do
    begin
      x:= 0;
      y:= out[li-1,no];
      for ni:= 0 to ni_end-1 do
      begin
        d:= der[li,ni];
        w:= weight[li,ni,no];
        x:= x + d*w;
      end;
      der[li-1,no]:= y*(1-y)*x;
    end;
  end;
  back:= 0.5*err;
end; { function BACK }


procedure learning (alp, eps: double );
{ update offsets & weights }
var
li, lo, ni, ni_end, no, no_end: integer;
di, yo, ew, et: double;
begin
  for li:= 1 to layers-1 do
  begin
    ni_end:= node[li];
    for ni:= 0 to ni_end-1 do
    begin
      et:= der[li,ni];
      dtheta[li,ni]:= -eps*et + alp*dtheta[li,ni];
      theta[li,ni]:= theta[li,ni] + dtheta[li,ni];
    end;
      end;
  for li:=1 to layers-1 do
  begin
    lo:= li-1;
    ni_end:= node[li];
    no_end:= node[lo];
    for ni:= 0 to ni_end-1 do
    begin
      di:= der[li,ni];
      for no:= 0 to no_end-1 do
      begin
        yo:= out[lo,no];
        ew:= di*yo;
        dweight[li,ni,no]:= eps*ew + alp*dweight[li,ni,no];
        weight[li,ni,no]:= weight[li,ni,no] + dweight[li,ni,no];
      end;
    end;
  end;
```

```
end;   { procedure LEARNING }


procedure make_matrix( it, isylla: integer );
var
i,j: integer;
begin
  for i:= 0 to inputs-1 do
      matrix_tmp[it,isylla,i]:= out[0,i];
  for i:= 0 to outputs-1 do
      matrix_tmp[it,isylla,inputs+i]:= out[Layers_Max-1,i];
end;   { procedure MAKE_MATRIX }


begin { main }
{ print first header }

  FOR I:= 0 TO 0 DO WRITELN ('YILDIZ');
  node[2]:= Output_Nodes;
  Outputs:= Output_Nodes;
  node[1]:= Hidden_Nodes;
  node[0]:= Input_Nodes;
  Inputs:= Input_Nodes;
  Layers:= Layers_Max;
  eps:= Epsilon;
  alp:= Alpha;
  num:= Outputs_Max;

{ print second header }

{ ************************
    DESIRED DATA READING
  ************************ }
  dess[0,0]:=0.49;dess[0,1]:=0.134;
  dess[1,0]:=0.51;dess[1,1]:=0.139;
  dess[2,0]:=0.53;dess[2,1]:=0.145;
  dess[3,0]:=0.54;dess[3,1]:=0.147;
  dess[4,0]:=0.55;dess[4,1]:=0.150;
  dess[5,0]:=0.57;dess[5,1]:=0.155;
  dess[6,0]:=0.58;dess[6,1]:=0.158;
  dess[7,0]:=0.59;dess[7,1]:=0.161;
  dess[8,0]:=0.60;dess[8,1]:=0.164;
  dess[9,0]:=0.50;dess[9,1]:=0.136;
  dess[10,0]:=0.53;dess[10,1]:=0.145;
  dess[11,0]:=0.55;dess[11,1]:=0.150;
  dess[12,0]:=0.56;dess[12,1]:=0.153;
  dess[13,0]:=0.57;dess[13,1]:=0.155;
  dess[14,0]:=0.58;dess[14,1]:=0.158;
  dess[15,0]:=0.59;dess[15,1]:=0.161;
  dess[16,0]:=0.60;dess[16,1]:=0.164;
  dess[17,0]:=0.61;dess[17,1]:=0.167;

{ **********************

    INITIALIZE MATRIX
```

```
  ********************* }
  for i:= 0 to num-1 do
      for j:= 0 to ( inputs+Outputs-1 ) do
          matrix[i,j]:= 0;

  for ii:= 0 to ( Trunc(Iterative_Max/Every)-1 ) do
      for i:= 0 to num-1 do
          for j:= 0 to ( inputs+Outputs-1 ) do
              matrix_tmp[ii,i,j]:=0;

{ learning started at .... }

  Wrand;        { Randomize offsets & weights };
  j:= 0;
  assign ( outfile,'OUTFILE.TXT');
  rewrite( outfile );
  { ***************************************
              LEARNING PHASE
  *************************************** }
  while ( j < Iterative_Max ) do
  begin
  dcount:=dcount-1;writeln(dcount);
    for count:= 0 to Every-1 do
    begin
      error:= 0;
      for isylla:=0 to num-1 do
      begin
        initial;
        yread( isylla );
        wforward;
        err:= back;
        learning( alp, eps );
        error:= error+err;
      end;
      error:= error / outputs;
      eps:= 0.5 * sqrt( error );
      Inc( j );
    end;


{ ***********************************************
                  TESTING PHASE
  *********************************************** }
    terror:= 0;
    for isylla:= 0 to num-1 do
    begin
      yread( isylla );
      wforward;
      terr:= back;
      terror:= terror + terr;
      make_matrix( trunc( j/Every ) - 1, isylla );
    end;
    writeln (outfile, '                    inputs          output
  );
```

```
    writeln (outfile, '-----------   ----------------   --------  -----------
---');
    for i:= 0 to num-1 do
    begin
      str( i:5, st1 );
      write(outfile,st1+'         ');
      for ii:= 0 to inputs-1 do
      begin
        str( matrix_tmp[trunc(j/Every)-1,i,ii]:4:3, st1 );
        write(outfile,st1+'    ');
      end;
      write( outfile,'    ');
      for ii:= 0 to outputs-1 do
      begin
        str( matrix_tmp[trunc(j/Every)-1,i,inputs+ii]:4:3, st1);
        write( outfile, st1+'       ');
      end;
      str( 4:3, st1 );
      writeln( outfile, st1 );
    end;
    terror:= terror/outputs;
    str ( j:5, st1 ); str ( eps:9:6, st2 );
    str ( error:9:6, st3 ); str ( terror:9:6, st4 );
    writeln( outfile, 'iteration=',st1,'    eps=',st2,
             '    error_1=', st3, '    error_t=', st4 );
    writeln( outfile );
{ }
    total_rate:=0;
    for isylla:= 0 to outputs-1 do
    begin
      rate[isylla]:= trunc(matrix_tmp[trunc(j/Every)-1,isylla,isylla]);
      total_rate:= total_rate + rate[isylla];
    end;
    total_rate:= total_rate/outputs;
{ }
    if ( j = Iterative_Max ) then
    begin
      writeln( outfile, 'learn & test finished at time' );
    end;
  end;
  close ( outfile );
end.
```

	inputs			output	
0	0.150	0.150	0.100	0.500	0.148
1	0.150	0.200	0.100	0.513	0.149
2	0.150	0.250	0.100	0.526	0.150
3	0.150	0.300	0.100	0.539	0.151
4	0.150	0.350	0.100	0.552	0.152
5	0.150	0.400	0.100	0.564	0.153
6	0.150	0.450	0.100	0.577	0.153
7	0.150	0.500	0.100	0.589	0.154
8	0.150	0.550	0.100	0.600	0.155
9	0.150	0.150	0.900	0.516	0.150
10	0.150	0.200	0.900	0.528	0.151
11	0.150	0.250	0.900	0.541	0.152
12	0.150	0.300	0.900	0.554	0.153
13	0.150	0.350	0.900	0.566	0.154
14	0.150	0.400	0.900	0.579	0.155
15	0.150	0.450	0.900	0.591	0.155
16	0.150	0.500	0.900	0.602	0.156
17	0.150	0.550	0.900	0.613	0.157

eration= 4500 eps= 0.009749 error_1= 0.000380 error_t= 0.000379

	inputs			output	
0	0.150	0.150	0.100	0.499	0.148
1	0.150	0.200	0.100	0.513	0.149
2	0.150	0.250	0.100	0.526	0.150
3	0.150	0.300	0.100	0.539	0.151
4	0.150	0.350	0.100	0.552	0.152
5	0.150	0.400	0.100	0.564	0.153
6	0.150	0.450	0.100	0.577	0.154
7	0.150	0.500	0.100	0.589	0.155
8	0.150	0.550	0.100	0.601	0.156
9	0.150	0.150	0.900	0.515	0.150
10	0.150	0.200	0.900	0.528	0.151
11	0.150	0.250	0.900	0.541	0.152
12	0.150	0.300	0.900	0.554	0.153
13	0.150	0.350	0.900	0.567	0.154
14	0.150	0.400	0.900	0.579	0.155
15	0.150	0.450	0.900	0.591	0.156
16	0.150	0.500	0.900	0.603	0.157
17	0.150	0.550	0.900	0.614	0.158

eration= 5000 eps= 0.009366 error_1= 0.000351 error_t= 0.000350

rn & test finished at time