T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GENETİK ALGORİTMA İLE YAPAY SİNİR AĞLARINDA YAPI VE PARAMETRE OPTİMİZASYONU

YÜKSEK LİSANS TEZİ

Ayşegül ÖZDEMİR

Anabilim Dalı: Elektronik-Bilgisayar Eğitimi

Programı: Kontrol Sistemleri

Tez Danışmanı: Prof. Dr. Muammer GÖKBULUT

ŞUBAT-2010

T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GENETİK ALGORİTMA İLE YAPAY SİNİR AĞLARINDA YAPI VE PARAMETRE OPTİMİZASYONU

YÜKSEK LİSANS TEZİ

Ayşegül ÖZDEMİR
06231106

Anabilim Dalı: Elektronik-Bilgisayar Eğitimi

Programı: Kontrol Sistemleri

Tez Danışmanı: Prof. Dr. Muammer GÖKBULUT

Tezin Enstitüye Verildiği Tarih: 8 Şubat 2010

ŞUBAT-2010

T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GENETİK ALGORİTMA İLE YAPAY SİNİR AĞLARINDA YAPI VE PARAMETRE OPTİMİZASYONU

YÜKSEK LİSANS TEZİ

Ayşegül ÖZDEMİR

06231106

Tezin Enstitüye Verildiği Tarih: 8 Şubat 2010
Tezin Savunulduğu Tarih: 26 Şubat 2010

Tez Danışmanı: Prof. Dr. Muammer GÖKBULUT (F.Ü)
Diğer Jüri Üyeleri: Doç. Dr. Engin AVCI (F.Ü)
Yrd. Doç. Dr. Beşir DANDIL (F.Ü)

ŞUBAT-2010

ÖNSÖZ

Tez çalışmam süresince sağladığı destek ve sabrı dolayısıyla değerli danışmanım Sayın Prof. Dr. Muammer GÖKBULUT'a, tezi hazırlamamda desteğini benden esirgemeyen arkadaşım Nehir YASAN'a ve aileme teşekkürü bir borç bilirim.

Ayşegül ÖZDEMİR

ELAZIĞ-2010

İÇİNDEKİLER

Sayfa No

ÖZET

Genetik Algoritma ile Yapay Sinir Ağlarında Yapı ve Parametre Optimizasyonu

Katmanlı yapay sinir ağları (YSA) herhangi bir doğrusal olmayan sürekli fonksiyonu arzu edilen doğrulukta yaklaştırma yeteneğine sahiptir. Bu özelliğinden dolayı YSA sistem modelleme, sinyal işleme ve tahmini, örüntü tanıma ve kontrol gibi uygulamalarda yaygın olarak kullanılmaktadır. Genellikle uygulamalarda sabit bir orta katman hücresine sahip ve katmanlar arasında tam bağlantıların yapıldığı ileri beslemeli katmanlı YSA kullanılır ve geriye yayılım algoritması ile eğitilir. Daha az hücre ve kısmi bağlantıya sahip olan küçük ağlar, sınırlı bilgi işleme kapasitesi nedeniyle iyi bir performans sağlamazken fazla hücre ve tam bağlantılı büyük ağlar gereksiz bağlantılara, işlem yoğunluğuna ve eğitim sürecinin yavaşlamasına neden olur. Ayrıca, YSA'nın parametre sayısı arttıkça eğitim sürecinde amaç fonksiyonunun yerel minimumlarına yakalanma olasılığı artar. Bu nedenlerle, yapay sinir ağlarının yapı ve parametre optimizasyonu önemli bir sorundur.

Bu tez çalışmasında, yapay sinir ağlarının yapısının ve parametrelerinin genetik algoritma (GA) ile ayarlanması incelenmiştir. Bu amaçla, ileri beslemeli YSA' nın optimizasyonu için kullanılan GA da gerçek kodlama tekniği kullanılmış ve her bir kromozom, YSA'nın ağırlıklarını, polarmalarını ve bu ağırlık ve polarmaların bağlantı anahtarlarını ihtiva eden bir vektör olarak kodlanmıştır. Bu ağlar, genetik-yapay sinir ağı (GYSA) olarak söylenir ve bağlantı anahtarları ile GYSA'nın etkisiz ağırlıkları ve hücreleri elimine edilebilir. Bu tezde önerilen GYSA'nın sistem modelleme problemindeki performansları incelenmiştir. Doğrusal olmayan dinamik sistemlerin modellenmesinden elde edilen sonuçlar, optimize edilmiş GYSA'nın iyi bir modelleme performansına sahip olduğunu göstermiştir.

Anahtar Kelimeler: Yapay Sinir Ağları, Genetik Algoritma, Sistem Modelleme, Optimizasyon

ABSTRACT

Structure and Parameter Optimization of Neural Networks Using Genetic Algorithm

A multilayer neural network (NN) can approximate any nonlinear continuous function at the desired accuracy. Hence, neural network are widely used for applications such as system identification, signal processing and prediction, pattern recognition and control. In the applications, fully connected feed forward neural networks including the fixed hidden layer units are commonly used and it is trained using the back propagation algorithm. Small networks having the less units and partial connection may not provide the good performance owing to its limited information processing power. On the other hand, large networks having the more units and full connection between layers may have unnecessary connections, more calculations time and slow convergence of the training process. Furthermore, if the NN has more parameters, the probability of trapping the local minimum of the performance criteria increases. Therefore, structure and parameter optimization of the NN is an important problem.

In this thesis, genetic algorithm (GA) is applied to solve the problem of tuning the structure and parameters of the neural networks. For this purpose, the real coding technique of GA is used for optimization of the feed forward neural network and each chromosome is encoded as a vector including the neural network weights, biases and switches for weights and biases. These networks are called as genetic-neural networks (GNN) and ineffective weights and units of GNN can be eliminated with the switches. System identification performance of the GNN proposed in this thesis is investigated. Results obtained from the identification of nonlinear dynamic systems show that the optimized GNN has good identification performance.

Keywords: Artificial Neural Networks, Genetic Algorithms, System Identification, Optimization.

ŞEKİLLER LİSTESİ

Sayfa No

Sayfa No

TABLOLAR LİSTESİ

Sayfa No

KISALTMALAR LİSTESİ

GA :Genetik Algoritma
GY :Geriye Yayılım
GYSA :Genetik Yapay Sinir Ağları
YSA :Yapay Sinir Ağları

1. GİRİŞ

Günümüzde sistem modelleme, bilgi sınıflandırma, denetim gibi çok değişik problemlerin çözümünde yapay sinir ağı kullanılmaktadır [1]. İnsan beynindeki sinir sistemine benzer bir yapıya ve işleyişe sahip olan YSA, genelleme yapabilme özelliği sayesinde son derece karmaşık fonksiyonlara yakınsayabilmektedir. YSA öğrenme yeteneği ile, karmaşık sistemlerin modellenmesinde oldukça önemli bir araç olmaktadır. Sistem dinamiğinin bir kısmının bilinebildiği veya sistemin matematiksel modelinin çıkarılmasının oldukça zor veya tamamıyla imkansız olduğu durumlarda, YSA sistem verileri ile eğitilerek sistemin modellenmesini sağlamaktadır [2].

Yapay sinir ağlarının ileri beslemeli ve geri beslemeli olmak üzere iki ana türü vardır. Uygulamalarda, katmanlar arası bütün hücrelerle tam bağlantılı ileri besleme katmanlı ağlar kullanılarak dinamik sistemlerin modellenmesi yapılabilmektedir. Geri beslemeli yapay sinir ağlarındaki ileri besleme ve geri besleme bağlantıları ile sinyal zıt doğrultularda iletilebilmektedir. Bu çeşit sinir ağlarının dinamik hafızaları vardır ve bir andaki çıkış hem o andaki hem de önceki girişleri yansıtır. Bu nedenle geri beslemeli ağlar dinamik sistemlerin modellenmesinde kullanılmıştır [1,3].

Genellikle uygulamalarda tüm hücrelerle bağlantıların yapıldığı tam bağlantılı ileri beslemeli yapay sinir ağı ve geri yayılım algoritması kullanılarak eğitim yapılır. Bu eğitim esnasında sinir ağının yapı ve parametrelerini ayarlamak zor bir optimizasyon problemidir [4,5]. Problemin karmaşıklığı arttıkça ağda kullanılan hücre sayısı da artar. Hücre sayısının artması eğitme zamanının uzamasına ve ağın gürültüye karşı duyarlı olmasına sebep olur. Hücre sayısı çok azsa, ağın öğrenmesi optimuma yaklaşamaz ve hata fonksiyonunda dalgalı bir davranış ortaya çıkar [6].

Yapay sinir ağlarında ağırlıkların belirlenmesinde kullanılan birçok öğrenme algoritması vardır. En yaygın kullanılan öğrenme algoritmalarından biri geri yayılım algoritmasıdır. Geri yayılım algoritması eldeki veri ile ağın çıktısı arasındaki farka dayalı olarak ağırlıkların güncellenmesini gerçekleştirir [7]. Yalnız bu algoritmanın bazı sakıncaları vardır. Geriye yayılım algoritmasında, hatanın en az düzeye indirilebilmesi için, geri yayılım kuralı ağırlık vektörünü negatif eğim boyunca kaydırır. Ancak bu esnada yerel en az bölgelerine düşme durumu söz konusudur. GY geniş, multimodal ve türevi alınamayan fonksiyonların global minimumunu bulmada etkisizdir ve optimum sonuca

yavaş ulaşır. Bu nedenlerle, yapay sinir ağının yapısal optimizasyonu ve parametre adaptasyonu üzerine çeşitli çalışmalar yapılmıştır [8-9].

Genetik algoritma, doğal seçme ve genetik operatörleri temel alan yapay bir genetik sistemdir ve tanımlanmış olan amaç fonksiyonunun (hata fonksiyonunun) değerini sürekli düşürmek suretiyle optimal sonucu bulmaya çalışan bir algoritmadır [10]. Genetik algoritmalar, sezgisel bir arama yöntemi olup son yıllarda birçok mühendislik problemine uygulanma olanağı bulmuş yapay zeka tekniklerinden biridir. Genetik operatörler yardımıyla "doğal seçilim" sürecinin bilgisayar ortamında benzeştirilmesini esas alan genetik algoritmaları diğer arama yöntemlerinden ayıran en önemli özellikler; amaç fonksiyonunu kullanması dolayısıyla türev hesaplarına dayanmaması, tek noktada değil birçok noktada (bir grup çözüm içinde) arama yapması, parametrelerin kodlanmış diziler halinde temsil edilmesi ve stokastik bir arama yöntemi olmasıdır. Bu özellikler de, genetik algoritmaların, özellikle klasik arama yöntemlerinin yetersiz kaldığı karmaşık çözüm uzayları ile karakterize edilen optimizasyon problemlerine uygulanmasını elverişli kılmaktadır. Genetik algoritmalar, bu tip problemlerde global optimum çözüme genellikle daha yüksek bir olasılıkla ulaşabilmektedir [11].

Yapay sinir ağının yapı ve parametre optimizasyonunu sağlamak için, global arama algoritmalarına dayanan GA sık kullanılan bir yöntemdir. Yapay sinir ağları ve genetik algoritmanın ortak kullanımı ile türetilen Genetik Yapay Sinir Ağları (GYSA), sağladığı öğrenme yeteneği ile daha etkin yapay sistemler tasarlama olanağı sunmuştur. Ayrı ayrı, çeşitli amaçlarla kullanılan bu yaklaşımların etkileşimli kullanımı konu olduğunda bahsedilebilecek uygulamalar, GA kullanarak YSA bağlantı ağırlık değerlerinin, YSA yapısının ve YSA öğrenme kurallarının belirlenmesidir [12].

Bu tez çalışmasında, yapay sinir ağlarının yapısının ve parametrelerinin genetik algoritma (GA) ile ayarlanması incelenmiştir. Bu amaçla, ileri beslemeli YSA' nın optimizasyonu için kullanılan GA da gerçek kodlama tekniği kullanılmış ve her bir kromozom, YSA'nın ağırlıklarını, polarmalarını ve bu ağırlık ve polarmaların bağlantı anahtarlarını ihtiva eden bir vektör olarak kodlanmıştır. GYSA denilen bu ağlarla doğrusal olmayan dinamik sistemler modellenmiştir. Modellemeye başlarken tam bağlantılı ileri beslemeli ağ yapısı kullanılmıştır. Eğitim esnasında bağlantı anahtarları kullanılarak gereksiz olan hücre ve bağlantılar elimine edilmiş, optimum ağ yapısı ve parametreler elde edilmiştir. Optimizasyon sonunda modellenen sistemin genelleme performansı test edilmiştir. GYSA ile sadece geriye yayılım algoritması ile eğitilen YSA modelleme

performanslarının kıyaslanabilmesi için iki farklı dinamik sistemin modellemesi gerçekleştirilmiştir. Modelleme sonucunda elde edilen ağ yapısına ve model çıkışlarına bakıldığında GA'nın YSA'nı optimize etmede ve sistem modellemede daha başarılı olduğu görülmüştür.

2. YAPAY SİNİR AĞLARI

2.1. Yapay Sinir Ağlarının Tarihçesi

Yapay sinir ağları, insan beyninden esinlenerek geliştirilmiş, ağırlıklı bağlantılar aracılığıyla birbirine bağlanan ve her biri kendi belleğine sahip işlem elemanlarından oluşan paralel ve dağıtılmış bilgi işleme elemanlarıdır [1]. Beynin üstün özellikleri, bilim adamlarını üzerinde çalışmaya zorlamış ve beynin nörofiziksel yapısından esinlenerek matematiksel modeli çıkarılmaya çalışılmıştır. Beynin bütün davranışlarını modelleyebilmek için fiziksel bileşenlerinin doğru olarak modellenmesi gerektiği düşüncesi ile çeşitli yapay hücre ve ağ modelleri geliştirilmiştir. Böylece, yapay sinir ağları denen günümüz bilgisayarlarının algoritmik hesaplama yöntemlerinden farklı bir bilim alanı ortaya çıkmıştır.

Yapay sinir ağlarının dayandığı ilk hesaplama modelinin temelleri 1940'ların başında araştırmalarına başlayan W.S. McCulloch ve W.A. Pitts'in, 1943 yılında yayınladıkları bir makaleyle atılmış olmuştur. Daha sonra 1954 yılında B.G. Farley ve W.A. Clark tarafından bir ağ içerisinde uyarılara tepki veren, uyarılara adapte olabilen model oluşturulmuştur. 1960 yılı ise ilk neural bilgisayarın ortaya çıkış yılıdır. 1963 yılında basit modellerin ilk eksiklikleri fark edilmiş, ancak başarılı sonuçların alınması 1970 ve 1980'lerde termodinamikteki teorik yapıların doğrusal olmayan ağların geliştirilmesinde kullanılmasına kadar gecikmiştir. 1985 yapay sinir ağlarının oldukça tanındığı, yoğun araştırmaların başladığı yıl olmuştur [12].

2.2. Yapay Sinir Ağlarının Üstünlükleri ve Sakıncaları

Yapay Sinir Ağlarının Üstünlükleri: Yapay sinir ağ modelleri biyolojik sinir ağlarının çalışmasından esinlenerek ortaya çıkarılmıştır. Canlılarda bulunan sinir sisteminin modellenmesi sayesinde yapay sinir ağları biyolojik sinir sisteminin üstünlüklerine sahip olmuştur.

Doğrusal Olmama: Yapay sinir ağları özellikle doğrusal olmayan sistemlerde tahmin yapma açısından istatistik hesaplamalarına göre daha kolay ve doğru sonuç vermesinden dolayı sık kullanılan bir yöntem haline gelmiştir. Özellikle işletmecilik ve finans

alanlarında olmak üzere tahmin gerektiren birçok alanda kullanılmaktadır. Yapay sinir ağlarının temel elemanlarından olan yapay sinir hücrelerinin doğrusal sonuçlar vermeyişinden dolayı bu özellik ağa da yansımıştır. Doğrusal olmama özelliğinden dolayı yapay sinir ağları karmaşık problemlerin çözümünde de sıkça kullanılmaktadır

Paralellik: Klasik problem çözme algoritmalarının aksine yapay sinir ağları paralel çalışmaya uygun bir yapıya sahiptir. Bu özelliği sayesinde çok daha hızlı problem çözebilme yeteneğine sahip olmuştur.

Hata Toleransı: Yapay sinir ağları özellikle doğrusal olmayan sistemlerde tahmin yapma açısından istatistik hesaplamalarına göre daha kolay ve doğru sonuç vermesinden dolayı sık kullanılan bir yöntem haline gelmiştir. Özellikle işletmecilik ve finans alanlarında olmak üzere tahmin gerektiren birçok alanda kullanılmaktadır. Yapay sinir ağlarının temel elemanlarından olan yapay sinir hücrelerinin doğrusal sonuçlar vermeyişinden dolayı bu özellik ağa da yansımıştır. Doğrusal olmama özelliğinden dolayı yapay sinir ağları karmaşık problemlerin çözümünde de sıkça kullanılmaktadır

Bilgisayar üzerinde çalışan bir elemanın zarar görüp devre dışı kalması o elmanın içinde bulunduğu sistemin çalışmamasına neden olur. Ancak paralel çalışabilme özelliği ve yapay sinir hücrelerinin bağımsız çalışabilme yapısından dolayı yapay sinir ağında herhangi bir eleman zarar gördüğünde ağın geri kalanı sorunsuz bir şekilde çalışmaya devam eder. İlk olarak yanlış sonuçlar verebilse de daha sonra yeni yapısını öğrenerek eski performansında çalışmaya devam edebilir.

Öğrenebilirlik: Klasik algoritmaların çoğu verilen formüllerin hesaplanması ile aynı girdiler için daima aynı çıktıları üretirler. Lineer olan bu algoritmaların aksine yapay sinir ağları sayesinde programlar öğrenme yeteneği de kazanmışlardır. Klasik algoritmalarda tam olarak tanımlı bir çözüm yolu olmayan problemler çözülemezken yapay sinir ağları sayesinde problemler çözüm yöntemi hakkında herhangi bir bilgi verilmeksizin çözülebilir. Yapay sinir ağlarının bu tip problemleri çözebilmesi için gereken tek şey örnek girdiler için sonuçların verilmesidir.

Genelleme: Yapay sinir ağları üzerinde çalıştığı probleme göre eğitildikten sonra eğitim sırasında karşılaşmadığı durumlar için de yanıt verebilir. Örneğin bir satranç taşının görüntüsünün tanıtılmasından sonra bu taşın görüntüsünü içeren ancak gürültülü bir görüntü verildiğinde bile yapay sinir ağı bu taşı tanıyabilir.

Uyarlanabilirlik: Yapay sinir ağı üzerinde çalıştığı probleme göre kendini düzenleyerek ağırlıklarını belirler. Bir problemi çözmek için eğitilen yapay sinir ağı

herhangi bir başka problemde de kolaylıkla kullanılabilir. Bunun için gereken tek şey yeni problemin girdi ve çıktılarıyla ağın tekrar eğitilmesidir.

Hız: Yapay sinir ağları paralel yapısı nedeniyle hızlı bir şekilde çalışıp problem çözme yeteneğine sahiptir. Aynı özelliğinden dolayı donanım üzerinde de kolaylıkla gerçeklenebilir.

Analiz ve Tasarım Kolaylığı: Yapay sinir ağlarının temel yapı taşı olan yapay sinir yapısı bütün yapay sinir ağlarında aynıdır. Bundan dolayı yapay sinir hücresinin tasarımından sonra bu temel eleman ile yapay sinir ağları kolaylıkla oluşturulabilir. Yapay sinir ağlarının temel yapısının da aynı olmasından dolayı bu ağlar her türlü problemin çözümünde kullanılabilinir.

Yapay Sinir Ağlarının Dezavantajları

Eğitim Süreci: Yapay sinir ağları oluşturulduklarında hiçbir bilgi içermediğinden dolayı direk olarak kullanılamazlar. Herhangi bir problem çözümünde kullanılacak olan yapay sinir ağının problemde kullanılmadan önce eğitilmesi şarttır. Bu eğitim süresi problemin çözümünden çok daha uzun zaman alabilir.

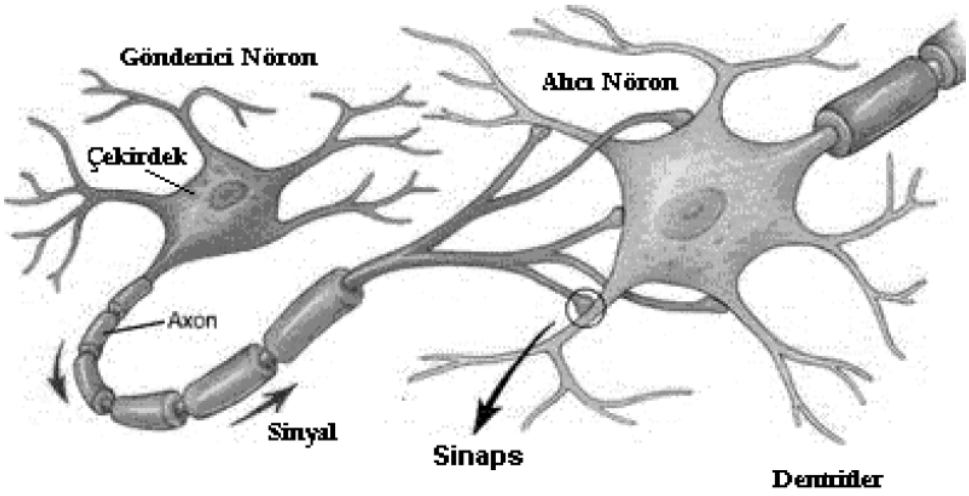
Başlangıç Koşullarına Bağlı Olması: Yapay sinir ağları başlangıç koşullarından bağımsız olarak çok kolay dahi olsa herhangi bir problemi çözemezler. Karar verme anında sadece daha önce öğrendiği koşullara göre sonuç üretebilir. Eğitim sırasında verilen örnekler ağın sonraki problemleri çözmesinde de etkilidir [34].

2.3. Yapay Sinir Ağlarının Biyolojik Kökeni

İnsan beyni, düşünme, hatırlama ve problem çözme yeteneklerine sahip karmaşık bir sistemdir. Beyin fonksiyonlarının bilgisayarla taklit edilmesine yönelik girişimlerin başarısı henüz kısmi olmaktan öteye gidememiştir.

Biyolojik sistemlerde öğrenme, hücreler arasındaki sinaptik (synaptic) bağlantıların ayarlanması ile olur. Yani, insanlar doğumlarından itibaren bir yaşayarak öğrenme süreci içerisine girerler. Bu süreç içinde beyin sürekli bir gelişme göstermektedir. Yaşayıp tecrübe ettikçe sinaptik bağlantılar ayarlanır ve hatta yeni bağlantılar oluşur. Bu sayede öğrenme gerçekleşir. Bu durum YSA için de geçerlidir. Öğrenme, eğitme yoluyla örnekler kullanarak olur; başka bir deyişle, gerçekleşme girdi/çıktı verilerinin işlenmesiyle, yani eğitme algoritmasının bu verileri kullanarak bağlantı ağırlıklarını bir yakınsama sağlanana kadar, tekrar tekrar ayarlamasıyla olur [34].

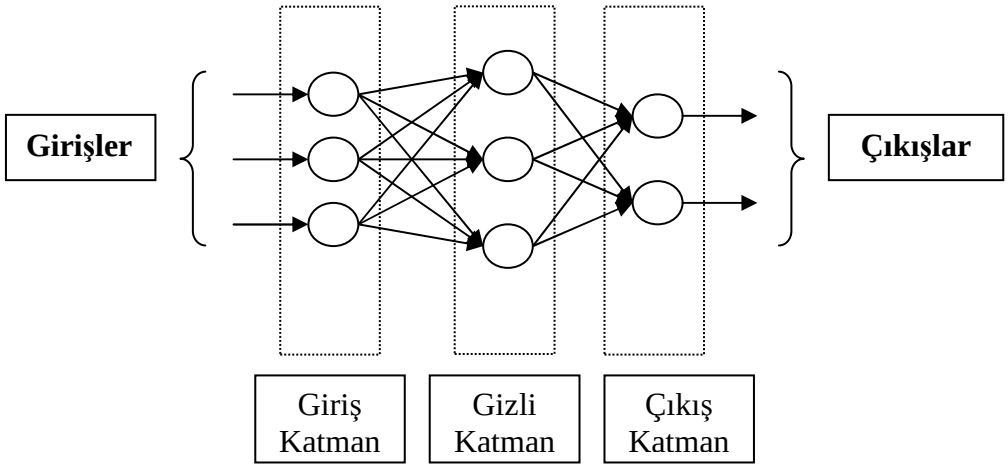
YSA'lar, ağırlıklandırılmış şekilde birbirlerine bağlanmış birçok işlem biriminden (hücreler) oluşan matematiksel sistemlerdir. Bir işlem birimi, aslında sık sık transfer fonksiyonu olarak anılan bir denklemdir. Bu işlem birimi, diğer hücrelerden sinyalleri alır; bunları birleştirir, dönüştürür ve sayısal bir sonuç ortaya çıkartır. Genelde, işlem birimleri kabaca gerçek hücrelere karşılık gelirler ve bir ağ içinde birbirlerine bağlanırlar; bu yapı da sinir ağlarını oluşturmaktadır [15].



Şekil 2.1. Gerçek sinir hücresinin şematik gösterimi

2.4. Yapay Sinir Ağının Yapısı

Sinir hücreleri bir grup halinde işlev gördüklerinde ağ (network) olarak adlandırılırlar ve böyle bir grupta binlerce hücre bulunur. Yapay hücreler birbirleriyle bağlantılar aracılığıyla bir araya gelmeleri yapay sinir ağını oluşturmaktadır.Yapay sinir ağıyla aslında biyolojik sinir ağının bir modeli oluşturulmak istenmektedir. Hücrelerin aynı doğrultu üzerinde bir araya gelmeleriyle katmanlar oluşmaktadır.Katmanların değişik şekilde bir birleriyle bağlanmaları değişik ağ mimarilerini doğurur. YSA'lar üç katmadan oluşur. Bu katmanlar sırasıyla; Girdi katmanı, ara katman, çıktı katmanıdır [13]. Şekil 2.2'de yapay sinir ağının katmanları görülmektedir.



Şekil 2.2. YSA katmanları

Girdi Katmanı: Dış çevreden giriş alan sinyalleri içerir. Girdi katmanına bir giriş geldiğinde, sinirleri diğer katmanlara giriş olacak şekilde çıkış üretir. Bu işlem koşulları yerine getiren kesin durum oluşuncaya kadar veya çıkış katı çağrılıncaya kadar devam eder ve çıkışı dış çevreye aktarır.

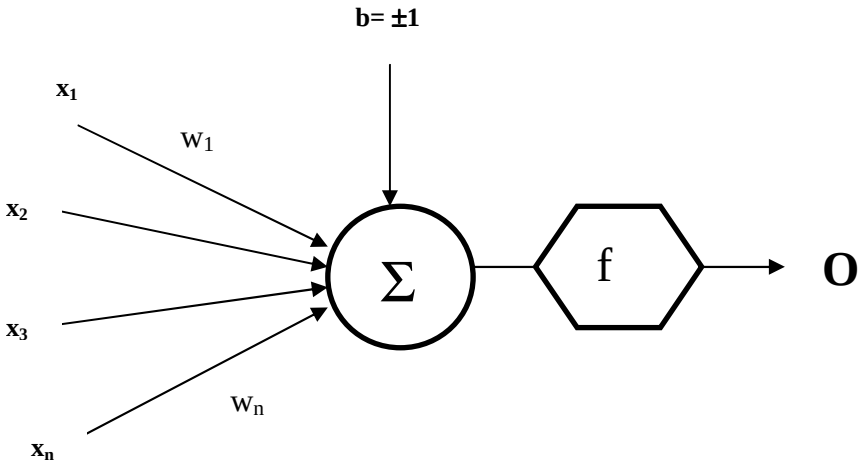
Ara Katman: Girdi katmanından alınan girişlerin işlenerek çıktı katmanına aktarıldığı kısımdır. Bir ağ içinde birden fazla ara katman olabilir. Gizli katman sayısı ağın en iyi çalışabileceği sayıda seçilmelidir.

Çıkış Katmanı: Ara katmandan gelen bilgileri işleyerek ağın girdi katmanından sunulan girdi seti için üretmesi gereken çıktıyı üreten katmandır [1].

2.5. Yapay Sinir Hücresi

Temel bir yapay sinir ağı hücresi biyolojik sinir hücresine göre çok daha basit bir yapıya sahiptir. En temel hücre modeli Şekil 2.3'te görülmektedir. Yapay sinir ağı hücresinde temel olarak dış ortamdan ya da diğer hücrelerden alınan veriler yani girişler, ağırlıklar, toplama fonksiyonu, aktivasyon fonksiyonu ve çıkışlar bulunmaktadır. Dış ortamdan alınan veri ağırlıklar aracılığıyla hücreye bağlanır ve bu ağırlıklar ilgili girişin etkisini belirler. Toplam fonksiyonu ise net girişi hesaplar, net giriş, girişlerle bu girişlerle ilgili ağırlıkların çarpımının bir sonucudur. Aktivasyon fonksiyonu işlem süresince net çıkışını hesaplar ve bu işlem aynı zamanda hücre çıkışını verir. Genelde aktivasyon fonksiyonu doğrusal olmayan bir fonksiyondur. Şekil 2.3'te görülen b bir sabittir, bias

veya aktivasyon fonksiyonunun eşik değeri olarak adlandırılır. Hücrenin matematiksel modeli şöyledir [34].



Şekil 2.3. Temel yapay sinir ağı hücresi

Çıkış, $o = f(w.x + b)$ şeklinde hücre çıkışı hesaplanır. Buradaki w ağırlıklar matrisi, x ise girişler matrisidir. n giriş sayısı olmak üzere;

$w = w_1, w_2, w_3, ..., w_n$

$x = x_1, x_2, x_3, ..., x_n$

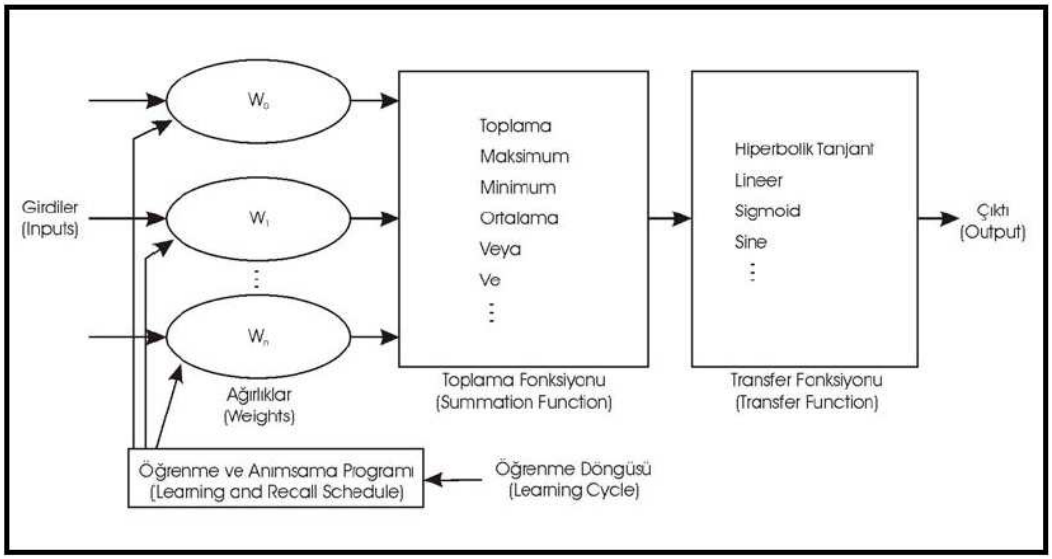
Şeklinde yazılabilir. Formalize edersek;

$$net = \sum_{i=1}^{n} w_i x_i + b \quad \text{ve} \quad o = f(net) \tag{2.1}$$

$$o = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \text{ şeklinde de yazılabilir.} \tag{2.2}$$

2.6. Hücre Modelleri

Hücreler, yapay sinir ağlarının bilgi işleyen yapısal elemanlarıdır. Bir hücrenin yapısında üç temel yapıtaşı bulunur. Şekil 2.4'te hücre modelleri görülmektedir.



Şekil 2.4. Yapay bir hücrenin detaylı yapısı

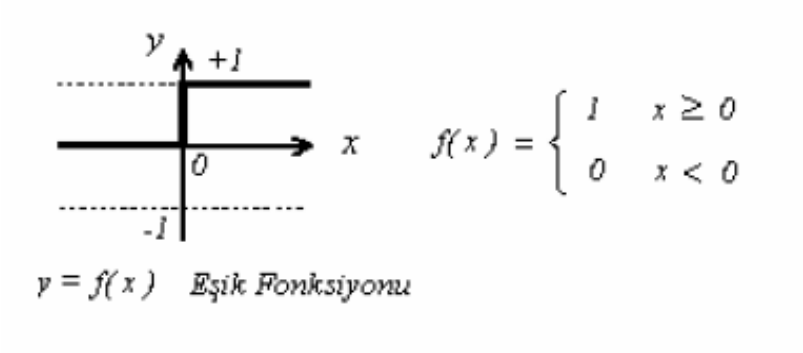
1. Sinaps adı verilen bağlantılar: Her sinapsın kendine ait w_{ij} ile gösterilen bir ağırlık çarpanı vardır. Bu ifadede *i* ile söz konusu hücre, *j* ile sinapsın giriş uygulanan ucu tanımlanmaktadır. Ağırlık çarpanı pozitif değerli olabileceği gibi negatif değerli de olabilir.
2. Toplayıcı: Uygun ağırlıkların uygulanmış olduğu giriş sinyallerini toplamak için kullanılır.
3. Aktivasyon fonksiyonu: Transfer fonksiyonu olarak da geçen aktivasyon fonksiyonu, birleştirme fonksiyonundan elde edilen net girdiyi bir işlemden geçirerek hücre çıktısını belirleyen ve genellikle doğrusal olmayan bir fonksiyondur. Genelde bir hücrenin normalize edilmiş genliği [0,1] veya [-1,1] kapalı aralığında ifade edilir [1,12].

2.7. Aktivasyon Fonksiyonu Çeşitleri

$\arg = I_j + b_j$ tanımlaması kullanılarak bir hücre için aktivasyon fonksiyonu $\Phi(\arg)$ ifadesiyle gösterilir. Aktivasyon fonksiyonun üç temel tipi takip eden alt başlıklarda verilmiştir.

2.7.1. Eşik Fonksiyonu

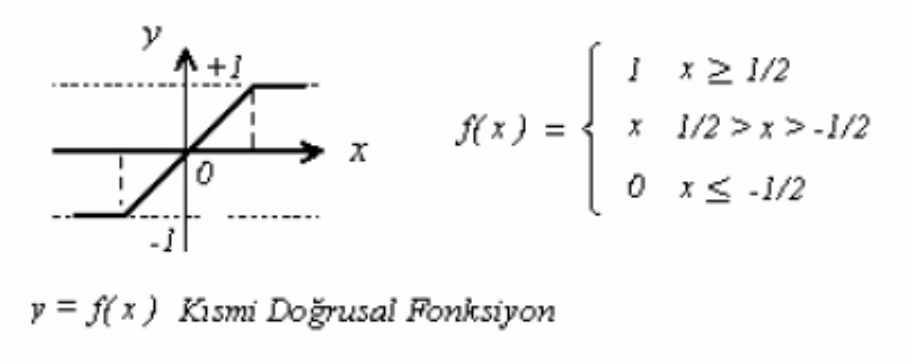
Eşik fonksiyonu kullanılarak yapılmış bir hücre literatürde McCulloch-Pitts modeli olarak adlandırılır. Fonksiyonun grafiği Şekil 2.5'te gösterilmiştir.



Şekil 2.5. Eşik fonksiyonu

2.7.2. Kısmi Doğrusal Fonksiyon

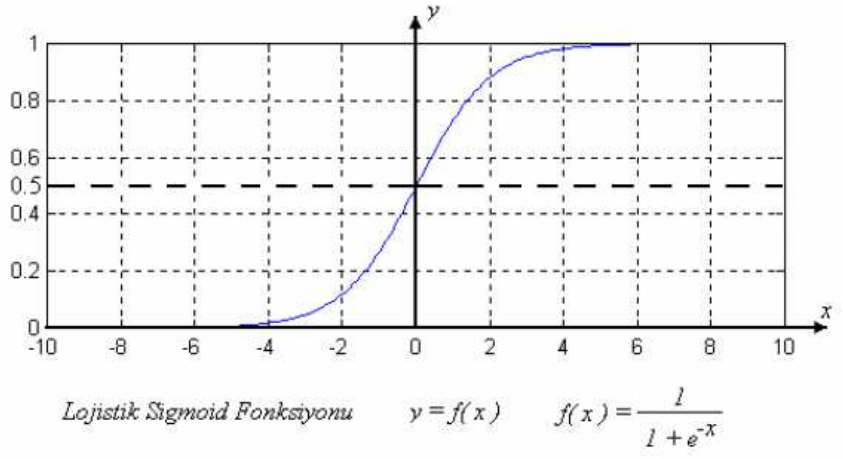
Doğrusal olmayan bir genlik artımı sağlayan bu aktivasyon fonksiyonu Şekil 2.6'da gösterilmiştir. Eğer doğrusal bölgedeki genlik arttıran katsayı yeterince büyük alınırsa parçalı doğrusal fonksiyon eşik fonksiyonuna dönüşür.



Şekil 2.6. Doğrusal fonksiyon

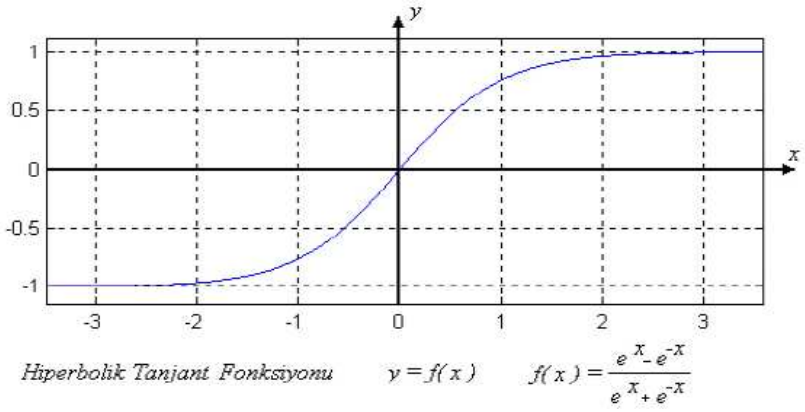
2.7.3. Sigmoid Fonksiyonu

Yapay sinir ağları oluşturulurken en çok kullanılan aktivasyon fonksiyonudur. Doğrusal ve doğrusal olmayan davranışlar arasında denge sağlayan sürekli artan bir fonksiyon olarak tanımlanır. Sigmoid fonksiyona bir örnek lojistik fonksiyondur ve Şekil 2.7'de gösterilmiştir. Görüleceği üzere sigmoid fonksiyonunun türevi alınabilirken eşik fonksiyonunun türevi alınamaz.



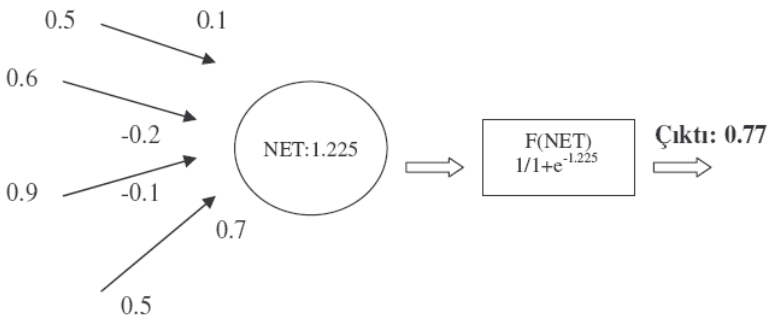
Şekil 2.7. Lojistik sigmoid fonksiyonu

Hiperbolik tanjant fonksiyonu da sigmoid fonksiyon örneğidir ve Şekil 2.8'de görülebilir [12].



Şekil 2.8. Hiperbolik tanjant fonksiyonu

Şekil 2.9'daki bir yapay sinir hücresinin çalışma örneği gösterilmiştir.



Şekil 2.9. Yapay sinir hücresinin çalışma örneği

Ağırlıklı toplam alınarak hücreye gelen net bilgi, şu şekilde hesaplanır;

NET: 0.5 * (0.1) + 0.6 * (-0.2) + 0.9 (-0.1) + 0.5 (0.7) = 1.225 (2.3)

Hücrenin sigmoid fonksiyonuna göre çıktısı;

Çıktı = 1/(1+e-1.225) = 0.77 (2.4)

elde edilir.

2.8. Yapay Sinir Ağı

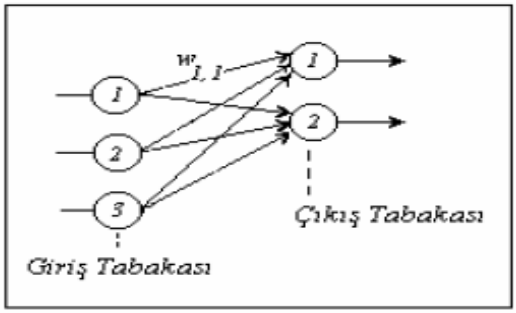
Yapay sinir ağı, öngörülen sayıda yapay sinir hücresinin, veri işlemek amacıyla belirli bir mimaride yapılandırılmasıyla şekillenir. Bu yapı, genellikle, numaralandırılan birkaç katmandan oluşur. İlk katman, çoğunlukla numaralandırılmayan, giriş katmanıdır. Bu katmanın numaralandırılmaya katılmayışının sebebi, giriş katmanındaki elemanların ağırlık çarpanları ve aktivasyon fonksiyonlarının olmaması sebebiyle veri girişinden başka bir işlem yapmamalarıdır. Çıkış katmanı da son katmandır. Tercihe bağlı olarak farklı sayıda olabilen diğer ara katmanların ortak adı gizli katmandır [13].

2.8.1. Ağ Yapıları

Ağ yapıları tek katmanlı ileri beslemeli, çok katmanlı ileri beslemeli ve döngülü yapay sinir ağları olmak üzere üç temel başlıkta toplanabilir.

1. İleri Beslemeli Sinir Ağları: İleri beslemeli YSA' da, hücreler katmanlar şeklinde düzenlenir ve bir katmandaki hücrelerin çıkışları bir sonraki katmana ağırlıklar üzerinden giriş olarak verilir. Giriş katmanı, dış ortamlardan aldığı bilgileri hiçbir değişikliğe uğratmadan orta katmandaki hücrelere iletir. Ağa sunulan giriş, orta ve çıkış katmanında işlenerek ağ çıkışı belirlenir. Bu yapısı ile ileri beslemeli ağlar doğrusal olmayan statik bir işlevi gerçekleştirir.

Tek katmanlı ileri beslemeli yapay sinir ağı en basit ağ yapısıdır. Bir giriş katmanı ve bir çıkış katmanı vardır. Örnek yapısı Şekil 2.10'da gösterilmiştir. Bu tip bir ağda bilgi girişten çıkışa doğru ilerler yani ağ ileri beslemedir. Tek katmanlı olarak isimlendirilmesinin sebebi, giriş katmanının veri üzerinde hiçbir işlem yapmadan veriyi çıkış katmanına iletmesidir.

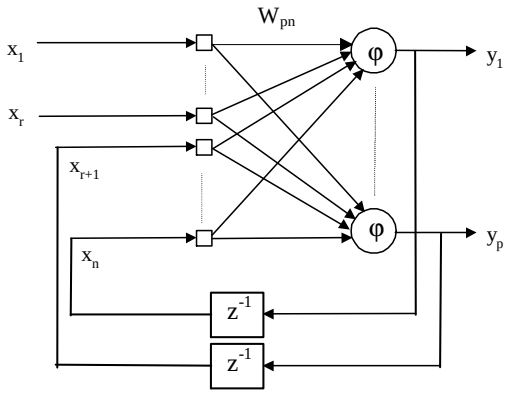


Şekil 2.10. Tek katmanlı ileri beslemeli sinir ağı

Yapay sinir ağımızda tek katman bulunması durumu genelde problemlerin büyük çoğunda karşılaşılan bir durumdur ve kısaca giriş ve çıkış arasında sonlu bir kümeden yine sonlu bir kümeye bağlantı bulunması durumunda kullanılır.

İleri beslemeli çok katmanlı YSA'nın, orta katmanında yeterli sayıda hücre olmak kaydıyla, herhangi bir sürekli fonksiyona istenilen doğrulukta yakınsanabileceği gösterilmiştir. En çok bilinen geriye yayma öğrenme algoritması, bu tip YSA'ların eğitiminde etkin olarak kullanılmakta ve bazen bu ağlara geriye yayma ağları da denilmektedir [12].

2. Geri Beslemeli Yapay Sinir Ağı: Geri beslemeli YSA'da en az bir hücrenin çıkışı kendisine ya da diğer hücrelere giriş olarak verilir ve genellikle geri besleme bir geciktirme elemanı üzerinden yapılır. Geri besleme, bir katmandaki hücreler arasında olduğu gibi katmanlar arasındaki hücreler arasında da olabilir. Bu yapısı ile geri beslemeli yapay sinir ağı, doğrusal olmayan dinamik bir davranış gösterir. Dolayısıyla, geri beslemenin yapılış şekline göre farklı yapıda ve davranışta geri beslemeli YSA yapıları elde edilebilir.. Şekil 2.11'de geri beslemeli ağ yapısı görülmektedir [16].



Şekil 2.11. Geri beslemeli yapay sinir ağı

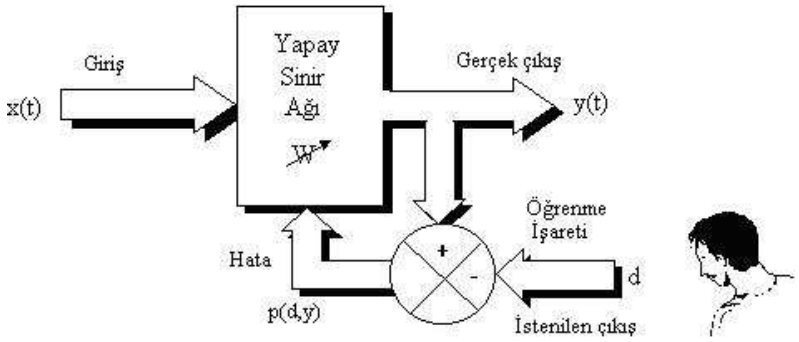
2.9. YSA'da Öğrenme

YSA'nın gerçekleştirdiği iki temel fonksiyon, öğrenme ve hatırlamadır. Öğrenme, YSA'nın ilgilendiği ortam tarafından ağın belirli bir süre uyarılmasıyla YSA'nın serbest parametrelerinin, arzu edilen öz yeteneği sağlayacak şekilde ayarlanması işlemidir ve öğrenmenin tipi, parametre değişikliklerinin yapılış şekline bağlıdır [14].

2.9.1. Eğiticili Öğrenme

Bu öğrenme metodunda, yapay sinir ağının eğitimi için eğitici veriler kullanılmaktadır. Eğitim seti giriş bilgileri ve istenen bilgiler olmak üzere iki ayrı vektör gibi düşünülebilir. Vektörlerin her bir karşılıklı elemanları bir eğitim çiftini oluşturmaktadır. Eğitim seti ağın eğitimine başlamadan önce belirlenmektedir. Ağın eğitimi için, öncelikle bağlantı ağırlıklarına rastgele değerler atanmaktadır. Daha sonra eğitim çiftlerine bağlı olarak bir algoritma dahilinde ağırlıklar yenilenmektedir.

İstenilen bilgiler ve ağın çıkışı arasındaki fark (hata) azalıncaya kadar eğitim sürdürülmektedir. Ağ çıkışındaki, hatanın azalması ağırlıkların kararlılık kazanması demektir. Ağırlıklar istenilen kararlılığa ulaştığında eğitim bitirilmektedir. Blok diyagram olarak Şekil 2.12'de gösterilmiştir.

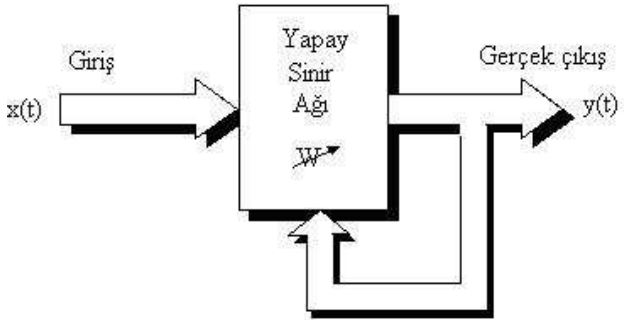


Şekil 2.12. Eğiticili öğrenme

2.9.2. Eğiticisiz Öğrenme

Eğiticisiz öğrenme moduna "Kendi kendine öğrenilebilen mod" da denilmektedir. Bu öğrenme modunda eğitim seti kullanılmamaktadır. Ağ, birbirine benzer giriş bilgilerini

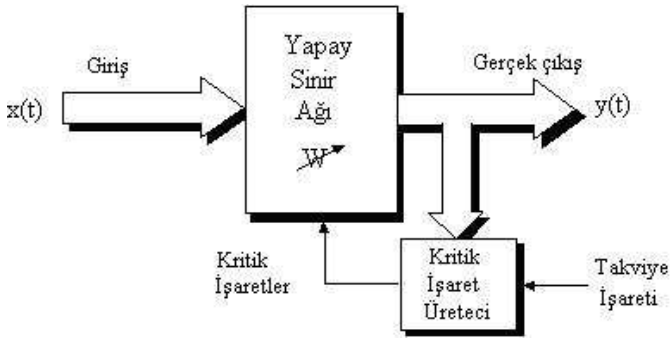
gruplamakta, veya giriş bilgisinin hangi gruba ait olduğunu göstermektedir. Ağ eğitimi için sadece giriş bilgileri yeterli olmakta, referans alınacak (eğitici) bilgiye ihtiyaç duyulmamaktadır. Ağın performansını kendiliğinden izlenmesi söz konusudur. Ağ, giriş sinyallerinin yönüne veya düzenine bakmakta ve ağın fonksiyonuna göre ayarlama yapmaktadır. Ağ kendini nasıl organize edeceği hakkında bir miktar bilgiye sahip olmalıdır. Şekil 2.13'te eğiticisiz öğrenme yapısı gösterilmiştir.



Şekil 2.13. Eğiticisiz öğrenme

2.9.3. Takviyeli Öğrenme

Bu öğrenme kuralı eğiticili öğrenmeye yakın bir metottur. Denetimsiz öğrenme algoritması istenilen çıkışın bilinmesine gerek duymaz. Hedef çıktıyı vermek için "öğretmen" yerine, burada YSA' ya bir çıkış verilmemekte fakat elde edilen çıkışın verilen girişe karşılık iyiliğini değerlendiren bir kriter kullanılmaktadır. Optimizasyon problemlerini çözmek için Hilton ve Sejnowski'nin geliştirdiği Boltzman kuralı veya genetik algoritma tasdikli öğrenmeye örnektir [27]. Şekil 2.14'te takviyeli öğrenme yapısı gösterilmiştir.



Şekil 2.14. Takviyeli öğrenme

2.10. Eğitim ve Test Verisi Seçimi

Yapay sinir ağının eğitimi ve sınaması için toplanan veri sistemin düzgün çalışma uzayını kapsamalıdır. Örnek kayıtlarının çalışma uzayının sınırlarını belirlediği ve yapay sinir ağlarının yalnızca eğitildiği çalışma aralığı için güvenilir sonuç verebildiği unutulmamalıdır. Genel özelliklerin net olarak belirlenmesi için örnek kaydı koleksiyonunun geniş olması tercih edilir. Bu kayıtların bir kısmı eğitim aşamasında kullanılırken bir kısmı sınama aşamasında ağın genelleştirme yeteneğinin teyidi amacıyla kullanılır. Sınamanın başarısızlığı durumunda, sınama amacıyla kullanılan kayıtların bir kısmı eğitim verisine katılarak eğitim ve sınama işlemleri kabul edilebilir bir performans kriterine kadar tekrarlanır.

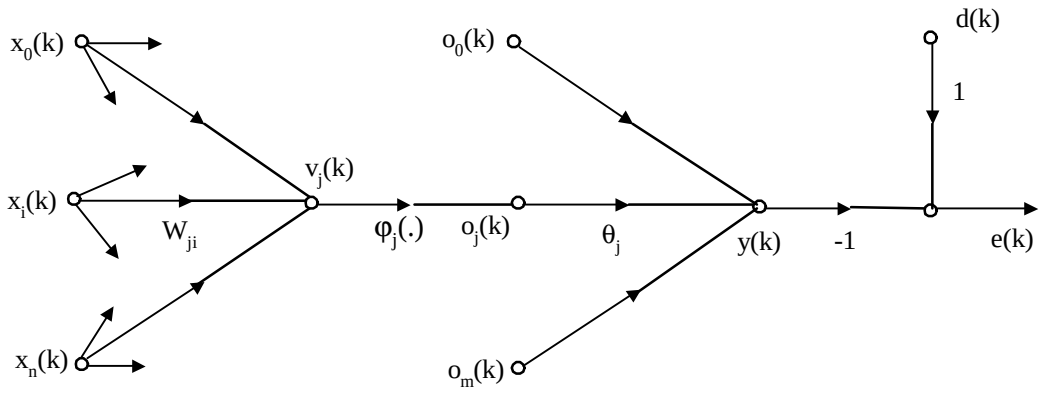
Yapay sinir ağı eğitiminde karşılaşılan temel bir sorun ezberlemedir. Yapay sinir ağının eğitim sürecindeki hata seviyesi, test sürecindeki hata seviyesine göre bariz farklılıklar gösterdiği takdirde ezberleme sorunu ile karşılaşılmış olur. Bu da tanımlanması istenen fonksiyonel ilişkiden ziyade eğitim verisindeki gürültü gibi tuhaflıkların da öğrenildiği anlamına gelir.

Ezberlemeyi azaltmak için yapılabilecekler:

- Eğitimde kullanılan kayıt sayısını arttırarak gürültünün ortalamasının kendiliğinden düşmesini sağlamak,
- Serbest parametre olan hücrelerin sayısını kullanılması gerekenin asgarisi ile sınırlamak,
- Eğitimi, ezberleme başlamadan kesmek. Yöntem, çapraz değerlemeli eğitim olarak adlandırılır. Esası, eğitim sırasında ezberleme kontrolü yapmaya dayanır. Eğitim aşamasında biri eğitim, diğeri kontrol için olmak üzere iki veri grubu kullanılır. Her epokun sonunda her iki grup için hatanın RMS (hataların kareleri toplamı) değeri hesaplanır ve kontrol kümesinin hatasında değişim olmadığı halde eğitim kümesinin hatasının azalmaya devam ettiği aşama tespit edilmeye çalışılır. Bu aşamada eğitim kesilir [12].

2.10.1. Geri Yayılım Algoritması

Çıkış katmanı doğrusal olan ileri beslemeli 3 katmanlı YSA'nın sinyal akış şeması Şekil 2.15'teki gibi çizilir.



Şekil 2.15 İleri beslemeli 3 katmanlı YSA sinyal akış şeması

Şekil 2.15'teki sinyal akış şemasından YSA'nın ileri yöndeki matematiksel modeli aşağıdaki gibi yazılır.

$$v_j = \sum_{i=0}^{n} W_{ji}.x_i \quad , \; o_j = \varphi(v_j) \qquad j = 1.2.....m$$
$$y_\ell = \sum_{j=0}^{m} \theta_j.o_j \tag{2.5}$$

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

$$e(k) = d(k)\text{-}y(k) \; , \; E(k) = \frac{1}{2} e^2(k) \quad , \quad J = \frac{1}{N} \sum_{k=1}^{N} E(k) \tag{2.6}$$

Toplam amaç ölçütü, N adet eğitim örneği için hataların karelerinin ortalaması olduğundan örneksel ya da toplu amaç ölçütünün ağırlıklara göre eğimi bulunarak geriye yayılım algoritması gerçeklenebilir. Hatalarının karelerinin eğimine göre, her bir eğitim örneğinin uygulanışında ağırlıklar yenilenirse örneksel öğrenme kuralı elde edilir. Toplam amaç ölçütünün eğimine göre, N adet eğitim örneğinin uygulanışından sonra ağırlıklar yenilenirse toplu öğrenme kuralı elde edilir. Buna göre, hataların karelerinin, doğrusal çıkış katmanındaki bir ağırlığa göre eğimi, zincir kuralına göre kısmi türevlerle belirlenebilir.

$$\frac{\partial E(k)}{\partial \theta_j(k)} = \frac{\partial E(k)}{\partial e(k)} . \frac{\partial e(k)}{\partial y(k)} . \frac{\partial y(k)}{\partial \theta_j(k)} \tag{2.7}$$

3 katmanlı YSA' nın matematiksel denklemleri (Denklem 2.5) ve sinyal akış şemasından,

$$\delta^2(k)=\frac{\partial E(k)}{\partial e(k)}.\frac{\partial e(k)}{\partial y(k)}=-e(k) \tag{2.8}$$

ve

$$\frac{\partial y(k)}{\partial \theta_j(k)}=o_j(k) \tag{2.9}$$

olarak bulunur. Buna göre çıkış katmanındaki ağırlıklara uygulanacak düzeltme ve yeni ağırlıklar aşağıdaki gibi elde edilir.

$$\Delta\theta_j(k)=\alpha.\delta^2(k).o_j(k) \qquad ,\ \theta_j(k+1)=\theta_j(k)-\Delta\theta_j(k) \tag{2.10}$$

Burada, δpq= p. katmandaki q. hücrenin yöresel hatası olarak söylenir. Hataların karelerinin orta katmandaki bir ağırlığa göre eğimi kısmi türevlerle,

$$\frac{\partial E(k)}{\partial W_{ji}(k)}=\frac{\partial E(k)\partial e(k)\partial y(k)\partial o_j(k)\partial v_j(k)}{\partial e(k)\partial y(k)\partial o_j(k)\partial v_j(k)\partial W_{ji}(k)} \tag{2.11}$$

gibi yazılır. YSA' nın matematiksel denklemlerinden yararlanarak türevler alınırsa,

$$\delta_j^1(k)=\frac{\partial E(k)}{\partial e(k)}.\frac{\partial e(k)}{\partial y(k)}.\frac{\partial y(k)}{\partial o_j(k)}.\frac{\partial o_j(k)}{\partial v_j(k)}=-e(k)\theta_j(k)\varphi'(v(k)) \tag{2.12}$$

ve

$$\frac{\partial v_j(k)}{\partial W_{ji}(k)}=x_i(k) \tag{2.13}$$

bulunur. Buna göre orta katmandaki ağırlıklara uygulanacak düzeltme ve yeni ağırlıklar aşağıdaki gibi elde edilir.

$$\Delta W_{ji}(k+1)=\alpha.\delta_j(k).xi(k) \quad ,\ W_{ji}(k+1)=W_{ji}(k)-\Delta W_{ji}(k) \tag{2.14}$$

Geriye yayılım algoritmalarının sonuç denklemlerinden, ağdaki bir ağırlığa uygulanacak düzeltmenin hücre girişi ve o hücrenin yöresel hatası ile orantılı olduğu görülmektedir.

Yukarıda, hataların karelerinin ağırlıklara göre eğimi belirlenerek örneksel öğrenme ile geriye yayılım algoritması çıkarılmıştır. Toplu amaç ölçütünün ağırlıklara göre yaklaşık eğimi Denklem 2.15'teki gibi belirlenerek toplu öğrenme ile de geriye yayılım algoritması elde edilir.

$$\frac{\partial J}{\partial W_{ji}} = \frac{1}{N}\sum_{k=1}^{N} \frac{\partial E(k)}{\partial W_{ji}} \tag{2.15}$$

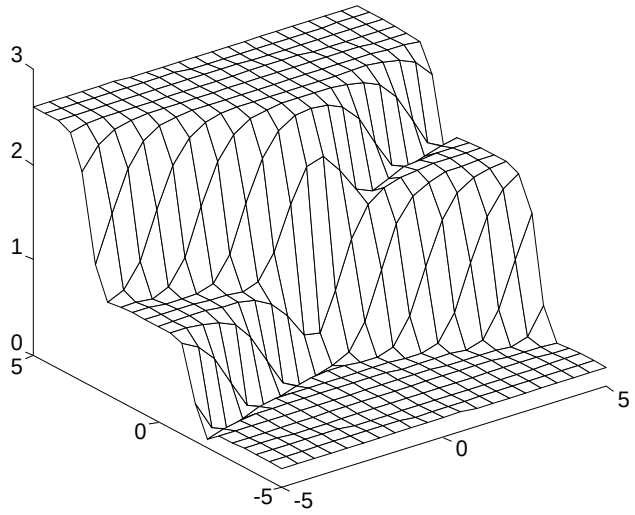
Gerçekte örneksel öğrenme, toplu öğrenme algoritmasının bir yaklaşığıdır, ancak gerçek zaman uygulamalarında örneksel öğrenme gereklidir. Her bir örnekteki örneksel eğim yerine, geçmişteki n1-1 adet örneksel eğim de saklanarak YSA' nın ağırlıkları, yine gerçek zamanda olmak üzere o anki ve geçmişteki n1 adet örneksel eğimlerin ortalamasına göre yenilenebilir [14].

2.10.2. Geriye Yayılım Öğrenme Algoritmasının Sakıncaları

Geriye yayılım öğrenme algoritması, çok boyutlu ağırlık uzayında hata yüzeyinin eğimine de bağlı olarak en azına, birinci dereceden yaklaşımdır. Bu nedenle, hata yüzeyinin en azına, doğrusal yönde salınımlarla ulaşılır ve sonuçta öğrenme yavaş olur. Şekil 2.16'da örnek bir hata yüzeyi verilmiştir.

Geriye yayılım algoritmasında öğrenmenin yavaş olmasının iki temel nedeni vardır.

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



Şekil 2.16. Örnek bir hata yüzeyi

Doğrusal olmayan YSA' da hata yüzeyinin diğer bir özelliği, küresel en az ile birlikte yöresel en azları da ihtiva etmesidir. Geriye yayılım algoritmasında, hata yüzeyinin eğimine göre en aza yaklaşıldığından bir yöresel en azda, ağırlıklara uygulanacak düzeltme amaç ölçütünü artırabilir. Böyle bir yöresel en azda öğrenmeyi durdurmak gerekebilir ve özellikle yöresel en az, küresel en azdan uzak bir noktada ise YSA, arzu edilmeyen bir davranış gösterebilir [14].

2.10.3. Geri Yayılımlı Eğitimi Etkileyen Etmenler

Bias: Her bir hücre için bir bias elemanı ilave edilebilir. Aktivasyon fonksiyonunun apsisi kestiği noktayı öteleyerek hücrenin eşik seviyesinde değişiklik etkisi yaratır. Genelde, eğitim hızını olumlu etkiler. Giriş elemanlarının biası (+1) olmak durumunda olmasına karşın diğer biaslar herhangi bir değer alabilir ve eğitilebilir.

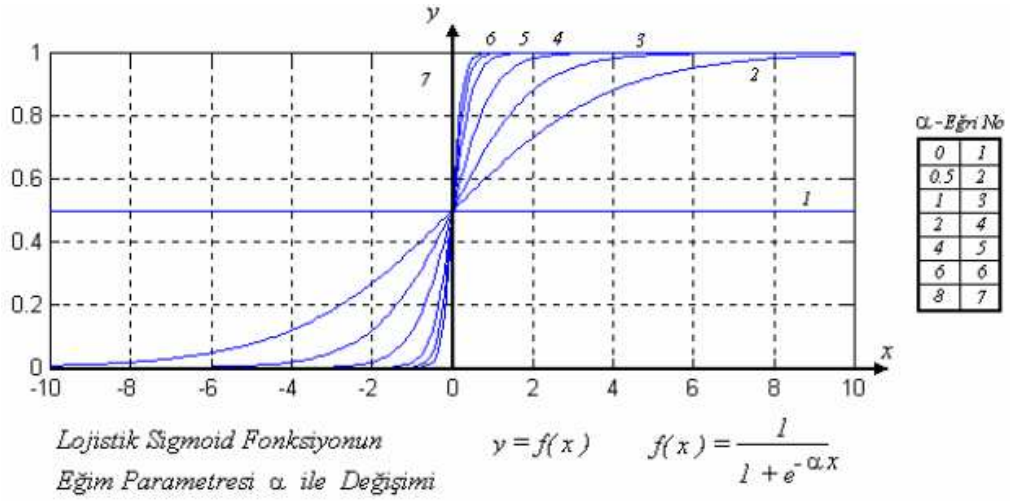
Momentum: Hareketli bir cismin momentumunun etkisine benzer, eğitim sürecinin yönünün korunması sağlar. Bunun için, ağırlık ayarlaması sırasında, önceki ağırlık değişimiyle orantılı bir terim ilave edilir. μ , momentum terimi olmak üzere denklem (2.13) elde edilir:

$$w_{pq,k}(N+1) = w_{pq,k}(N) - \eta_{p,q}\delta_{pq,k}\Phi_{p,j}(I) + \mu\Delta w_{pq,k}(N) \qquad (2.16)$$

Momentum teriminin hesaplamaya katılması adım sayısında ve toplam ağ hatasında bir düşüş meydana getirmektedir. Momentum katsayısı yüksek alındığında ağdaki toplam hatanın sıfıra doğru daha fazla bir eğimle yaklaştığı görülmektedir [1].

Öğrenme Katsayısı(η): Pozitif değerli olmak zorundaki öğrenme katsayısı 2'den büyük seçildiğinde YSA'nın kararsızlığına, 1'den büyük seçildiğinde de çözüme ulaşmaktansa salınım yapmasına sebep olur. Öğrenme katsayısı için (0,1) uygun aralıktır. Bu aralıkta seçilen katsayının büyüklüğüyle, öğrenme adım aralığı orantılıdır. Eğitim verisinin değişim oranına uygun katsayı seçilmelidir. Uyarlanabilir öğrenme katsayısı kullanımı da başvurulabilecek yöntemlerdendir.

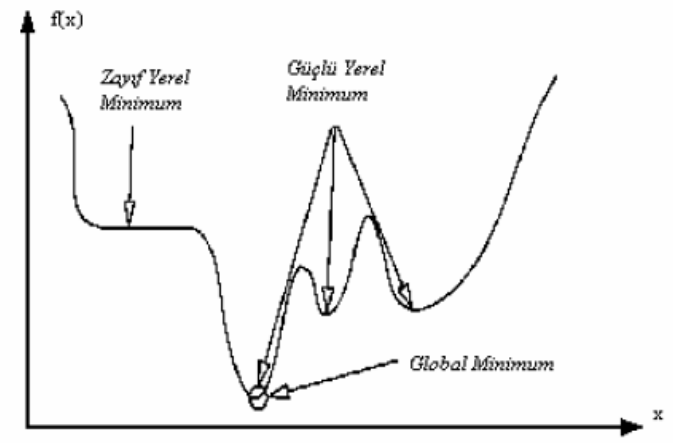
Sigmoid Fonksiyonunun Eğim Parametresi(α): Eğim parametresine bağlı değişim Şekil 2.17'de gösterilmiştir. Artan ağırlık değerleri, hücrenin, sigmoid fonksiyonun eğiminin (türevinin) çok küçük değerli bölgelerinde işlem yapmasına sebep olur. Geri yayımlanan hata terimi türevle orantılı olduğundan, düşük eğimde yeterli eğitim gerçekleşmez. Eğimin ayarlanması, eğitim süresini ve başarısını doğrudan etkiler.



Şekil 2.17. Lojistik sigmoid fonksiyonunun eğim parametresiyle değişimi

Yerel Minimum: Geri yayılım algoritmasının sıkıntı yaratan yanı, yerel minimuma takılmasıdır. Algoritma, gradyan azaltma yöntemini kullandığı için hata yüzeyinin eğimi negatif olduğu sürece ağırlıkların minimuma ulaşmayı sağlayacak şekilde ayarlandığı kabul edilir. Hata yüzeyi, kolaylıkla düşülebilecek fakat kurtulmanın mümkün olamayabileceği tepe ve çukurları barındırabilir. Atanan ilk ağırlık değerleri, civarında, azalan gradyan yönünde arama yapılacak noktayı belirler. Rastgele seçilen ilk noktadan,

global minimuma kadar olan mesafede yerel minimum problemini yaratabilecek hata değerleriyle karşılaşmak muhtemeldir. Şekil 2.18 yerel minimumları göstermektedir [1,34].



Şekil 2.18. Karşılaşılan minimum türleri

3. GENETİK ALGORİTMA

3.1. Genel Bilgiler

Genetik algoritma (GA) ilk defa 1975 yılında John F. Holland tarafından ortaya atılmış bir arama yöntemidir. Genetik algoritma, yönlendirilmiş rastgele araştırma algoritmalarının bir türüdür [18]. Algoritma, canlıların doğada yaşama ve neslini devam ettirme teorisi üzerine kurulmuştur. Mevcut şartlarda popülasyon içinde en güçlü kromozomların soyunu devam ettirme olanağının daha fazla olduğu tezine dayanmaktadır. Yöntem; kalıtım, mutasyon, doğal seçim ve çaprazlama gibi biyolojik evrim kurallarından esinlenilerek geliştirilmiştir.

Algoritma, araştırma uzayında mevcut olan çözümlerin oluşturduğu bir başlangıç popülasyonu kullanır. Bu başlangıç popülasyonu, her bir kuşakta tabii seçme ve tekrar üreme işlemleri vasıtası ile art arda geliştirilir. En son kuşağın en uygun yani en kaliteli kromozomu, problem için optimal bir çözümdür. Bu çözüm, her zaman optimum olmayabilir, ancak kesinlikle optimuma en yakın olan çözümdür.

GA'lar arama ve optimizasyon için sezgisel yöntemlerdir. Geniş arama algoritmalarının aksine, genetik algoritmalar en iyiyi seçmek için tüm farklı durumları üretmez. Bundan dolayı, mükemmel çözüme ulaşamayabilir. Fakat zaman kısıtlamalarını hesaba katan en yakın çözümlerden biridir. GA'lar şartlara uyum sağlayabilir. Bunun anlamı, önceden hiç bilgisi olmamasına karşın, olayları ve bilgiyi öğrenme ve toplama yeteneğine sahip olmasıdır.

GA çok boyutlu bir araştırma uzayında, global optimum bir çözümü rahatlıkla bulabilmektedir. Genetik algoritmanın paralel işlem yapabilen bilgisayarlarda kullanılmaya elverişli yapısı, zaman alıcı problemlerin kısa zamanda çözümü için çekici bir alternatif olmasını sağlamıştır. Genetik algoritma, klasik optimizasyon yöntemleri ile çözümü mümkün olmayan veya çözüm süresi problemin büyüklüğü ile üstel olarak artan problemlerin çözümünde etkin olarak kullanılmaktadır. Genetik algoritmanın temel avantajı, optimize etmeye çalıştığı problemin tabiatı ile ilgili herhangi bir bilgiye ihtiyaç duymamasıdır. GA'nın farklı problemlere uyarlanmasında en önemli adım, probleme özgü genetik kodlama ve uygunluk fonksiyonunun belirlenmesidir.

GA'ların çoklu mümkün çözümleri araştırma gibi önemli özelliklere sahip olmaları

ve amaç fonksiyonunun gradyanının bilinmesine ihtiyaç duymamaları en önemli üstünlükleridir. GA'nın bu nedenle diğer geleneksel yöntemlerden, karakteristik olarak dört özelliği ile farklıdır:

• GA parametrelerle değil, onların kodları ile çalışır.

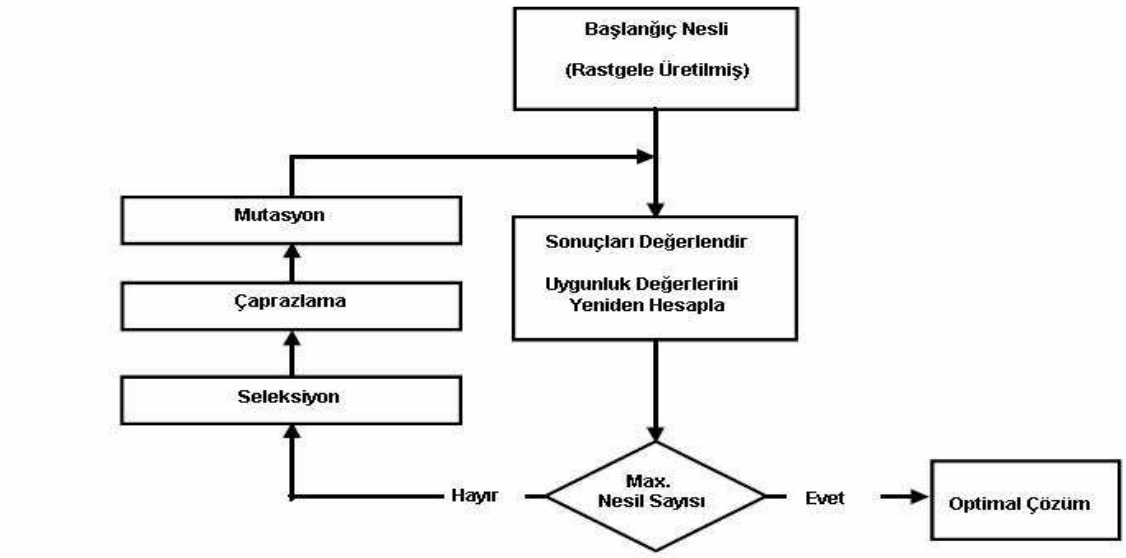
• GA tek bir noktadan değil, bir noktalar topluluğundan arama yapar.

• GA türevleri veya diğer yardımcı bilgileri değil, amaç (uygunluk) fonksiyonu bilgisini kullanır.

• GA aramayı deterministik kurallarla değil, olasılıklı geçiş kuralları ile yapar.

Her problemin çözümü için GA kullanmak iyi bir yol değildir. Birkaç parametreli analitik fonksiyonun çözümünde klasik metotlar daha hızlıdır. Böyle durumlarda, nümerik metotlar tercih edilmelidir. Paralel bilgisayarlar kullanılırsa GA daha hızlı sonuç verebilir [19,20].

Genetik algoritmanın temel akış diyagramı Şekil 3.1'de görüldüğü gibidir.



Şekil 3.1. Genetik algoritma temel akış diyagramı

İşlem, çözüm uzayından belirli bir sayıda başlangıç çözümün seçilmesi ile başlar. Her adımda çaprazlama, mutasyon ve uygunluk değerlendirmesi yapılır. Sonlandırma kriteri sağlanıncaya kadar bu işlem devam eder. Sonlandırma kriteri; belli bir sayıda iterasyon, uygunluk değerinin beklenen bir değere ulaşması veya belli sayıda iyileşme olmadan sürdürülen iterasyon sayısı olabilir.

3.2. GA Uygulama Alanları

GA, son yıllarda geliştirilmiş olup, geniş bir alanda uygulamaya başlanmıştır. Bilgisayar bilimi, robot bilim, işletme, mühendislik, eğitim, matematik, tıp ve ziraat gibi geniş bir yelpazede uygulama örnekleri ile karşılaşılmaktadır.

Optimizasyon: GA'lar devre şemaları ve iş-atölye planlamaları gibi sayısal ve birleştirici optimizasyon problemlerini içeren çok çeşitli alanlarda kullanılmaktadır.

Otomatik Programlama: GA'lar özel amaçlı bilgisayar programlarının geliştirilmesinde, hücresel hareketliler ve ağ sıralanması gibi hesaplamaya dayalı yapıların tasarımında kullanılmaktadır.

Yapay Zeka ile Öğrenme: GA'lar hava durumu ve protein yapısını önceden tahmin gibi sınıflandırma ve öngörü işlerini içeren yapay zeka ile öğrenme uygulamalarında kullanılmaktadır.

GA'lar aynı zamanda sinir ağlarının ağırlıkları, sınıflayıcı sistemlerin kuralları ya da sembolik üretim sistemleri ve robotlar için algılayıcılar gibi yapay zeka ile öğrenmenin çeşitli özelliklerini geliştirmekte de kullanılmaktadır.

Ekonomik: GA'lar yenilik yaratma işlemlerinin modellenmesinde, teklif verme stratejilerinin geliştirilmesinde ve ekonomik pazarların geliştirilmesinde kullanılmaktadır.

Sosyal Sistemler: GA'lar böcek kolonilerindeki sosyal davranışın evrimi ve daha genel olarak, çok temsilcili sistemlerde işbirliği ve iletişimin gelişmesi gibi çeşitli sosyal sistemlerin evrimsel yönlerini araştırmakta kullanılmaktadır.

Toplum Genetiği: GA'lar, üreme yoluyla yeni nesillere aktarılan ve ebeveynden geçen kalıtsal özelliklerin evrimsel yönünün araştırılmasında kullanılmaktadır.

Evrim ve Öğrenme: GA'lar kişisel öğrenmenin ve nesilden nesile öğrenmenin birbirlerini nasıl etkilediğini araştırmakta kullanılmaktadır.

Yukarıda anlatılan ve daha birçok alanda GA uygulamalarının başarılı sonuçlar vermesi, hem araştırmacıların konuya ilgisini daha da artırmış hem den GA'nın konferans ve yayınlarda bilgisayar biliminin bir alt dalı haline gelmesini sağlamıştır.

Birçok yapısal optimizasyon problemleri GA ile başarılı bir şekilde çözülebilmektedir (Chen ve Chen,1997,s.1323). Özellikle optimizasyon problemlerinde, daha etkin sonuçlar elde etmek için GA ile başka teknikler bir arada, diğer bir deyişle karma GA'lar kullanılmıştır.

Bir grup araştırmacı ise, GA'yı bulanık mantık tekniği ile bir arada kullanarak,

belirsizlikleri aşmaya çalışmışlardır. Buna örnek olarak, tedarik zincirinde talep dalgalanmalarından kaynaklanan belirsizliklerin aşılması çalışması verilebilir [21].

3.3. Genetik Algoritmanın Çalışma Prensibi

GA, belirli problemlerin muhtemel çözümlerini kromozom benzeri veri yapıları şeklinde kodlar. Böylece her çözüm belirli kriterlere göre kodlanmış diziler haline getirilir ve bu diziler kromozom olarak çağrılır. Kromozomların kodlanması farklı şekillerde olabilir.

3.3.1. GA Kromozomların Kodlanması

GA uygulamasına başlarken önce optimize edilmesi gereken parametreler istenilen şekle dönüştürülmek zorundadır. Buna kodlama denir. Kodlama GA için önemli bir konudur. Çünkü sistemden gözlemlenen bilgiye bakış açısı büyük ölçüde sınırlandırılabilir. Gen stringi probleme özel bilgiyi depolar. Gen olarak adlandırılan her bir öğe, genellikle değişkenler stringi olarak ifade edilir. Değişkenler ikili veya reel sayı şeklinde gösterilebilir ve aralığı probleme özel olarak tanımlanır.

3.3.1.1. İkili Kodlama

Bu yöntem ilk GA uygulamalarında kullanıldığı için hala en çok kullanılan yöntemlerdendir. Burada her kromozom, 0 ve 1'lerden oluşan bit dizisidir ve ikili diziyle ifade edilir. Bu dizideki her bit, çözümün bir özelliğini taşır. Dizinin tümü ise bir sayıya karşılık gelir.

Kromozom	A	101110010110
Kromozom	B	010110100000

Şekil 3.2. İkili kodlama için kromozom örneği.

İkili kodlama, hatta küçük boyutlu problemler için çok büyük kromozom vektörü gerektirmektedir. Çözülecek probleme bağlı olarak çeşitli şifreleme yöntemleri vardır.

3.3.1.2. Permütasyon Kodlama

Bu kodlama, Gezgin Satıcı Problemi (GSP) ve iş sıralama problemleri gibi pemutasyon problemlerinde kullanılır. Burada her kromozom bir numaralar dizisidir.

Kromozom	A	35127604
Kromozom	B	01562347

Şekil 3.3. Permütasyon kodlama için kromozom örneği

Bu problemlerde olan ikili düzende kodlama da kullanılabilir. Örneğin, her onluk sayı için ikili düzendeki sayılar yazılır. (Yapısına göre bilgisayarlar verilen ikili biçimde işlemektedir.)

0 için	000
1 için	001
… …	…
7 için	111 gibi.

Bu durumda 8 genden (örneğin 8 şehirden) oluşan permütasyon kodlamalı kromozomlarımız 24 genden oluşacaktır ve her üç gen bir şehri ifade edecektir.

3.3.1.3. Değer Kodlama

Bu kodlama, gerçel gibi kompleks sayıların yer aldığı problemlerde kullanılır. Bu tür problemler için ikili kodlama çok zordur. Burada her kromozom bazı, değerler dizisidir. Bu değerler ise problemle ilişkilidir; örneğin gerçel sayı, karakter veya kompleks nesneler olabilir.

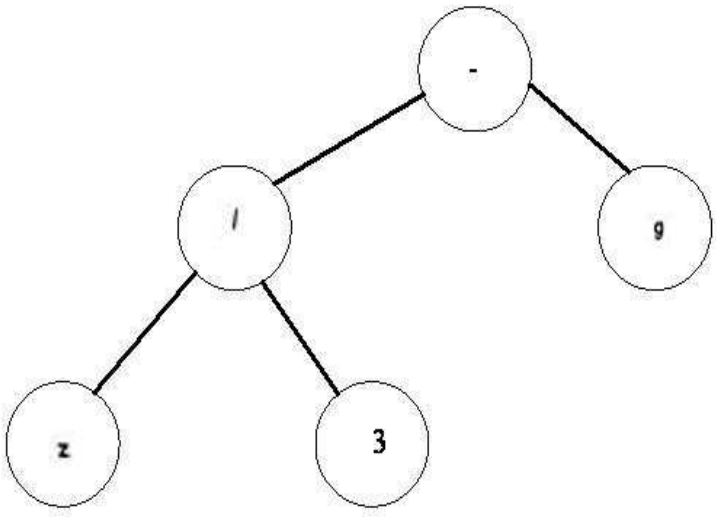
Kromozom A	3.274 6.121 2.456 0.299 2.115
Kromozom B	ABCJFJKHDERJFDLFFEGHK
Kromozom C	(geri), (sağ), (ileri), (geri), (sol)

Şekil 3.4. Değer kodlama için kromozom örneği

Bu kodlama bazı özel problemler (örneğin bir yapay sinir ağının ağırlık katsayılarının bulunması) için çok uygundur.

3.3.1.4. Ağaç Kodlama

Ağaç kodlama genellikle genetik programlamada programlar ve ifadeler oluşturmak için kullanılır. Ağaç kodlamada her kromozom, adından da anlaşıldığı gibi nesneler ve nesneler arası işlemleri içeren bir ağaç yapısından oluşur.



Şekil 3.5. Ağaç kodlama için kromozom örneği

Ağaç kodlama, program geliştirmek için uygundur. Örneğin, LISP ve Prolog gibi programlama dillerinde ağaçtan yapısal olarak sık bir biçimde kullanılır. Ağaç kodlamada çaprazlama ve mutasyon çok kolay bir şekilde uygulanabilir.

Televizyon yarışmalarından iyi bildiğimiz verilen hedef değere en yakın sonucu üretecek aritmetik ifadenin bulunması, bu kodlamanın kullanıldığı problemlere verilebilecek örneklerdendir. Bu ifade, sayılardan ve “ +, -, , / ” aritmetik işlemlerden oluşmaktadır. İlk dört giriş sayısı 1 ile 10 arasından ve son iki sayı ise ‘25, 50, 75, 100 arasından seçilir. Burada önemli bir sınırlama vardır. Bu sınırlama yarışmacının her sayıyı yalnızca bir defa kullanabilmesidir. Örneğin (2, 3, 5, 8, 25, 100) sayıları ile 467 sayısını veren ifadeyi bulmaya çalışalım. (5*100) - (25 +8) = 467 mümkün çözümlerden biridir. Fakat her zaman kesin sonucun bulunması mümkün değildir. Bu durumda ise genetik algoritma en yakın sonucu veren ifadeyi bulmaya çalışır. Problemin durum uzayının belli sınırlamalar olduğunda bile 33.664.168 noktadan oluştuğu düşünülürse, onun basit arama

yöntemiyle çözülemeyeceği kesindir. Buna başka bir örnek olarak ise yalnız tek bir kez 1'den 9'a kadar sayıları ve yine dört aritmetik işlemi " +, -, , / " kullanılarak 100 değerinin elde edilmesini gösterebiliriz [22].

3.3.2. Başlangıç Popülasyonu ve Kromozom Uzunluğu

Popülasyon sayımızın çokluğu, çözüm uzayımızı daha iyi örneklediğinden en iyi çözümü elde etme olasılığımız artacaktır. Diğer taraftan da programın çözüm sürecinde popülasyonun büyüklüğüne ve çevrim sayısına doğrudan bağlıdır. Bu nedenle popülasyon genişliğinin gereğinden büyük seçilmesi süreci geciktireceği gibi iyi çözüme ulaşmamıza katkıda da bulunmayabilir. Popülasyonun büyüklüğünün, probleme bakılarak belirlenmesi en uygun yoldur.

Şayet popülasyonun geniş olması gerekiyorsa, çözüm uzayımızı bir çok alana bölüp aramalarımızı lokalize etmek uygun bir yol olabilir. Zira bazı problemlerde, bizi çözüme ulaştıracak aralığa gelene kadar çok fazla zaman kaybedebiliriz.

Kromozom uzunluğu da oldukça önemli bir parametredir. Bazen kromozomdaki bazı bitlerin optimum değeri erken belirlenebilir. Bu durumda bu bitler bir sonraki çevrimlere dahil edilmeden hesap dışı bırakılabilir.

3.3.3. Seleksiyon

GA'nın amaca hitap edebilmesi, seçim kriterinin yani doğal seleksiyonun yeterince iyi olmasına bağlıdır. En uygun kromozomların seçilebilmesi için "uygunluğun" ne olduğunu iyi tanımlanmalıdır. Diğer bir ifadeyle bizi en iyiye (optimal) götürecek olan uygunluk fonksiyonun zayıf çözümleri eleyecek, kuvvetli çözümleri ise yaşatacak şekilde modellenmesi gerekmektedir.

Uygunluk fonksiyonunda, her problemin kendi doğal karakteristiğinde yer alan parametreleri kullanılır. Bu parametrelerin, güdülen amaca uygun olarak formüle edilmesiyle de uygunluk fonksiyonu elde edilmiş olur.

GA'nın her iterasyonunda, elde edilen kromozomlar, uygunluk fonksiyonuna tabi tutulalarak uygunluk değerleri hesaplanır. Daha sonra yüksek uyuma sahip olanlar seçilir.

Bu noktada birçok seçim yöntemi kullanmak mümkündür. En çok kullanılan seçim yöntemleri aşağıda sırasıyla verilmiştir.

3.3.3.1. Rastgele Seçim

Kromozomların eşleştirilmesinde rastgele sayı üreteci kullanılır. Kromozomlar 1'den başlayarak sıralanır. Birinci eşleştirmeyi bulmak için iki adet rastgele sayı üretilir.

N_{iyi} =Popülasyondaki yüksek uygunluk değerine sahip kromozom sayısı.

Kromozom=roundup{ N_{iyi} x rastgele sayı}

Burada roundup fonksiyonu, sayıları en yüksek sayıya yuvarlar. Örneğin; rastgele olarak üretilen 6 sayı 0.1535, 0.6781, 0.0872, 0.1936, 0.7021 ve 0.3933 ise bu sayılar, 6 ile çarpılıp bir üst tamsayıya yuvarlanarak; 1, 5, 1, 2, 5, 3 değerleri elde edilir. Buna göre kromozom1- kromozom5, kromozom1 - kromozom2 ve kromozom5 - kromozom3 eşleştirilecektir [19].

3.3.3.2. Ağırlıklı Seçim

İlk olarak amaç fonksiyonunda, kromozomların uygunluk değerleri hesaplanır. Hesaplanan uygunluk değerleri en küçükten en büyüğe doğru sıraya konur. Tablo 3.1'de görüldüğü gibi $N_{pop} \leq N_{ipop}$ ise tutulur, geriye kalanlar atılır. Burada N_{pop} değeri N_{ipop}'a kadar olabilir. Genelde popülasyonun %50'sinin seçilmesi $(N_{pop} = N_{ipop/2})$ uygun seçenektir. Seçilen N_{pop}'un yarısı N_{iyi}, yarısı da $N_{kötü}$ olarak ayrılır. N_{iyi} olanlar eşleştirme havuzuna konurken $N_{kötü}$ olanlar eşleştirme havuzundan atılır. Toplam popülasyon sayısı N_{ipop} =24'tür. GA'nın her bir iterasyonunda popülasyonun 12'si tutulur ve bu kromozomların altı tanesi eşleştirme havuzuna atılır [19,23].

Tablo 3.1. Ağırlıklı seçim

Kromozomlar	Uygunluk Değerleri
N_{ipop} { N_{iyi} { 0000000000 0000	-13778 } N_{pop}
N_{iyi} { 1111101101 0010	-13770
N_{iyi} { 0001011000 0010	-13765
N_{iyi} { 1100001100 1010	-13755
N_{iyi} { 0110110110 1001	-13690
$N_{kötü}$ { 1111101101 0010	-13670
$N_{kötü}$ { 0000011000 0010	-13690
$N_{kötü}$ { 0011101101 0010	-13677
$N_{kötü}$ { 0011010001 0010	-13676
$N_{kötü}$ { 1111101101 0010	-13668
0001011011 1010	-13665
1101101100 1010	-13647
1111111111 0010	-13630
0001111000 1010	-13570
1000001100 1000	-13670
0110110110 1001	-13690
1111101101 0010	-13677
1000000001 1000	-13565
1101101101 0010	-13440
1000001111 1000	-13404

3.3.3.3. Rulet Çemberi

Bu seçim yöntemine orantılı seçim mekanizması da diyebiliriz. Bu seçim metoduna göre her kromozomun seçimle şansı, uyumluluk değeriyle orantılıdır. Uyumluluk değeri yüksek olan kromozomun seçilerek yeni popülasyona eklenme olasılığı yüksektir. Ancak uyumluluk değeri düşük olanlarında seçilme olası vardır [24].

Seçilecek kromozomu belirlemek için, öncelikle Denklem (3.1) ile uygunluk değerlerinin toplamı bulunur.

$$UT = \sum_{i=1}^{PS} U_i \tag{3.1}$$

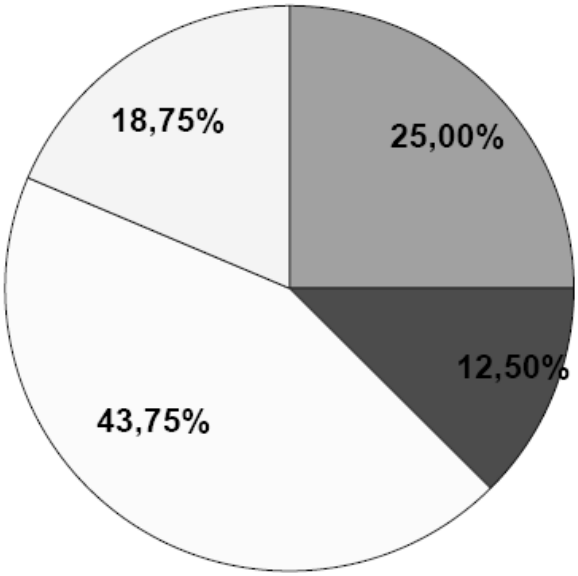
Burada PS popülasyondaki kromozom sayısı (popülasyon büyüklüğü), U_i ise her bir kromozomun uygunluk değeridir. Elde edilen toplam uygunluk değeri kullanılarak her bir kromozomun seçilme olasılığı (O_i) yüzde olarak Denklem (3.2) ile hesaplanır.

$$O_i = \frac{U_i}{UT} * 100 \tag{3.2}$$

Örnek olarak, 4 adet kromozomdan oluşan bir popülasyon için kromozomların uygunluk değerleri ve seçilme olasılıkları Tablo (3.2)'de görüldüğü gibi hesaplanmıştır. Hesaplanan olasılık değerlerinin sonucu rulet tekerleğinde Şekil 3.6'da görüldüğü gibi gösterilebilir [22].

Tablo 3.2 Kromozomların uygunluk değerleri ve seçilme olasılıkları

Kromozom No	**Kromozom Dizisi**	**Uygunluk değeri**	**Seçilme Olasılığı (%)**
1	0 0 1 1 0	100	25
2	0 1 0 1 0	50	12,5
3	1 0 1 1 0	175	43,75
4	1 0 0 1 0	75	18,75
Toplam		**400**	**100**

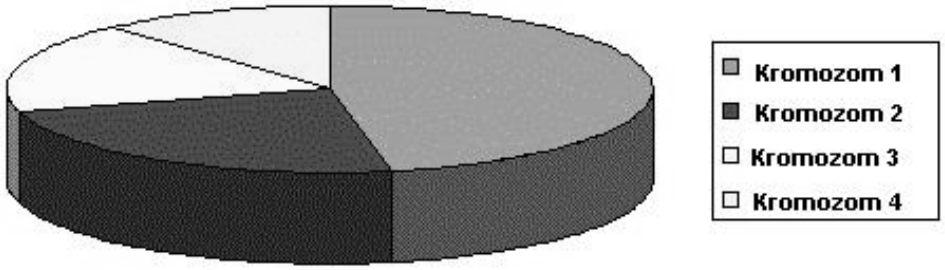


Şekil 3.6. Rulet tekeri seçimi

3.3.3.4. Sıralı Seçim

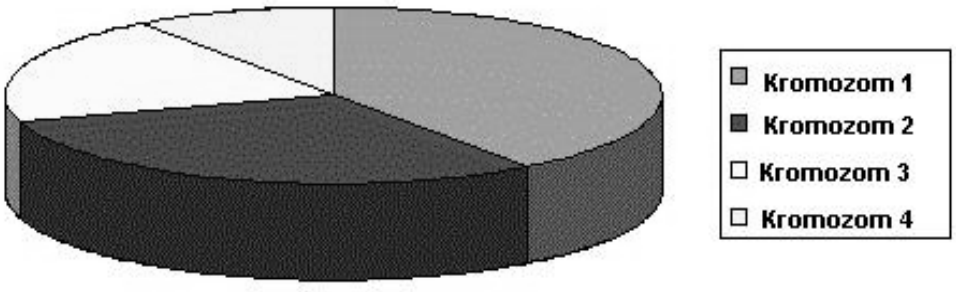
Her kromozom uyumluluk fonksiyonuyla hesaplanmış uyum değerine göre sıralanır. Daha sonra kromozomlar en kötü uyum değerine sahip olandan en iyi değere sahip olana doğru sıralanır. En uyumsuz kromozomun değeri 0, en uyumlu kromozomun değeri is N (toplam kromozom sayısı) kadardır. Böylece kromozomların seçilme olasılığı, doğrusallaştırılmış artan fonksiyon haline getirilir. Bu çeşitlilik yaratmak açısından Rulet Çemberinden daha iyi sonuç veren bir seçim yöntemidir.

Uyumluluk değerinin büyüklüğüne göre kromozomların seçilme ihtimalinin Şekil 3.7'deki gibi olduğunu düşünelim.



Şekil 3.7. Kromozomların uyumluluk değerine göre ağırlıkları

Sıralama Seçim metoduna göre kromozomların seçilme ihtimali Şekil 3.8'deki gibi olacaktır.



Şekil 3.8. Kromozomların uyumluluk değerine göre sıralanması

3.3.3.5. Seçkinlik

Bir popülasyondan çaprazlama yöntemi ile yeni popülasyonun oluşturulması sırasında, bulunan en iyi çözümün kaybedilmesi olasılığı vardır. Bu riski ortadan kaldırmak amacıyla, her popülasyondaki en iyi bir veya birkaç kromozom doğrudan yeni popülasyona kopyalanır. Doğal ortamda karşılığı olmayan bu işleme seçkinlik adı verilir. Elit sayısının yüksek olması, iyi kromozomların popülasyonda baskınlık sağlayarak genetik çeşitliliğin azalmasına ve dolayısı ile algoritmanın erken yakınsamasına yani yerel optimuma takılmasına sebep olabilmektedir.

3.3.4. Çaprazlama

Mevcut neslin seçilmiş kromozomlarından yeni kromozomlar oluşturmak üzere çaprazlama işlevi yürütülür. Bu operatörün temel fonksiyonu, yeni nesillerin atalarının birebir kopyası olmalarını engellemektir.

Çaprazlama operatöründen kromozomların iyi özellikleri birleştirerek daha iyi kromozomlar oluşturması beklenir. Böylece arzu edilen en iyi kromozoma yani çözüme ulaşılır. Çaprazlama sonucunda iyi olmayan kromozomlar de türeyebilir ancak onlar uygunluk değerleri düşük olacağından seleksiyon gereği elenecektir [24].

3.3.4.1. Çaprazlama Oranı

Çaprazlama olasılığı çaprazlamanın hangi sıklıkta yapılacağını belirtir. Eğer hiç çaprazlama yapılmaz ise (çaprazlama olasılığı %0) yeni kromozomlar eski kromozomların aynısı olur ama bu yeni kuşağın eskisiyle aynı olacağı anlamına gelmez. Eğer bu oran %100 olursa yeni kromozomlar tamamıyla çaprazlama ile elde edilir. Çaprazlama eski kromozomlardan iyi taraflar alınarak elde edilen yeni kromozomların daha iyi olması umuduyla yapılır [21].

3.3.4.2. Çaprazlama Operatörleri

Programlama sürecinde dört farklı çaprazlama operatörü kullanılabilir. Bu operatörlerden hangisinin seçileceği ise problemin çözümü öncesi yapılacak değerlendirmelere bakılarak karar verilir.

a. Tek noktalı gen takası: Bir çaprazlama noktası seçilir. Bu noktaya kadar olan bitler (genler) birinci ebeveynden, geriye kalanlar ise diğerinden alınarak yeni bir kromozom oluşturulur.

Kromozom-l: 11011 00100110110

Kromozom-2: 11011 111000011110

Döl-1: 11011 11000011 110

Döl-2: 11011 00100110110

b. Çift noktalı gen takası: Burada iki kırılma noktası seçilir. İlk noktaya kadar olan bitler birinci ebeveynden iki nokta arasındaki bitler ikinci ebeveynden, kalanlar ise tekrar birinci ebeveynden yeni kromozoma kopyalanır.

11001011 + 11011111 = 11011111

c. Tek biçimli takas: Bu çaprazlama biçiminde bitler rastgele şekilde her iki ebeveynden yeni kromozoma kopyalanmaktadır.

11001011 + 11011101 = 11011111

d. Aritmetik takas: Yeni bir kromozom oluşturmak için değişik aritmetik işlemler uygulanır.

11001011 + 11011111= 11001001 (AND) VE işlemi [22]

e. Karıştırma metodu: Gerçek kodla kodlanmış kromozomlu GA'da daha iyi sonuç veren bu metoda göre iki parametrenin değeri karşılaştırılır ve Denklem 3.3'teki formül kullanılarak çaprazlama işlemi gerçekleştirilir.

$$Nesil_1 = [P_{a1}, P_{a2}, P_{a3}, \ldots\ldots\ldots\ldots P_{aNpar}]$$

$$Nesil_2 = [P_{b1}, P_{b2}, P_{b3}, \ldots\ldots\ldots\ldots P_{bNpar}]$$

$$P_{yeni} = \beta P_{an} + (1-\beta) P_{bn} \qquad (3.3)$$

β =0 ve 1 arasında üretilen rastgele sayı

P_{an} = Anne kromozomun n. parametresi

P_{bn} = Baba kromozomun n. parametresi

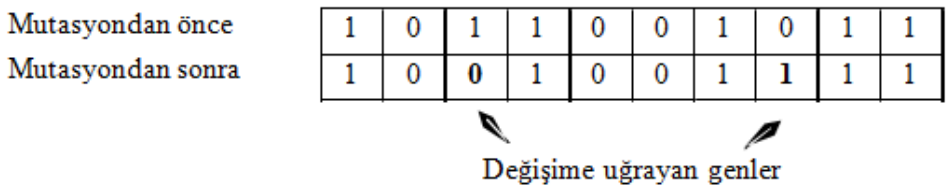
İkinci nesil, β yerine $1-\beta$ konularak ve birinci neslin tümleyeni alınarak bulunur. $\beta=1$ ise, P_{an} baskın gelir ve P_{bn} ölür. $\beta=0$ ise P_{bn} baskın gelir ve P_{an} ölür. $\beta=0.5$ olduğu zaman, sonuç iki parametrenin ortalaması olarak ortaya çıkar [23,19].

3.3.5. Mutasyon

Mutasyon işlemi, yeni kromozomlar oluşturulurken ebeveynin tüm özelliklerinin aynısını almaları yerine, özelliklerde küçük bazı değişikliklerin olması ilkesine dayanır. Çaprazlanarak elde edilen kromozomların genleri üzerinde küçük değişiklikler yapılarak arama uzayı mevcut çözüm noktalarının çevresine yayılır.

Mutasyon işlemi, probleme ve genetik kodlamaya göre değişiklik arz eder. Gerçek kodlarla kodlanmış kromozomlarda mutasyon işlemi, genleri temsil eden ondalık sayılara küçük bir miktarın eklenmesi veya çıkarılması şeklinde uygulanır. Ancak, gezgin satıcı problemi gibi, genetik kodlaması gidilecek düğüm noktalarından oluşacak şekilde tasarımlanmış problemlerde mutasyon işlemi, seçilen iki farklı düğüm noktasının yer değiştirmesi şeklinde yapılabilir.

İkili kodlanmış kromozomlarda ise mutasyon işlemi, gen değerinin sıfırdan bire veya birden sıfıra değiştirilmesi şeklinde gerçekleştirilir. Şekil 3.9'da 10 bitlik bir dizi ile temsil edilen kromozoma mutasyon uygulanması gösterilmiştir.



Şekil 3.9 Mutasyona uğrayan 10 bitlik dizi

3.3.5.1. Mutasyon Oranı

GA'da mutasyon, seçim süreci sırasında popülasyonda kaybolmuş genleri yerine koyma veya başlangıç popülasyonunda bulunmayan gen dizilimlerini ortaya çıkarma gibi kritik iki görevi yerine getirir (Gen ve Cheng,1997:4). Ruiz ve Maroto'da (2006:788) benzer bir ifade ile, yerel optimumu engellemek ve yeni gen dizilimlerini ortaya çıkarmak için mutasyon operatöründen faydalanılması gerektiğini vurgulamaktadır. Mutasyon

olmadan popülasyonlar ebeveynlere bağımlılık ve genlerin silinmesi nedeniyle alelleri (Genlerin alabileceği değerleri) kaybedebilirler (Hamilton ve Ridley, 2005:14).

Gen (1996)'e göre mutasyon oranı yeni genlerin popülasyona deneme amaçlı sunulma oranını kontrol eder. Bu oran çok düşük ise, faydalı bir çok gen hiç denenmeyecek, bunun tam tersi bu oran çok yüksek olursa, daha çok tesadüfü tedirginlik oluşacak, yeni nesil ebeveynlere benzemeyi kaybetmeye başlayacaktır ve bunun sonucunda algoritma zamanla araştırma geçmişinden öğrenme yeteneğini kaybedecektir [25].

Mutasyon oranının belirlenmesinde ve uygulanmasında bazı konulara dikkat edilmelidir. Mutasyon oranının yüksek olması genetik çeşitliliği arttırmakla beraber, rastsallığı da arttırdığından algoritmanın yakınsama (çözüm noktasına yaklaşma) süresi artacaktır. Mutasyon oranının çok düşük olması ise, genetik çeşitliliği azaltacak ve erken yakınsamaya sebep olabilecektir. Erken yakınsama sonucunda genellikle yerel optimuma düşüldüğünden istenmeyen bir durumdur.

Mutasyon işleminin uygulanmasında gözden kaçırılmaması gereken başka bir konu ise, kromozomun bünyesindeki genlere uygulanacak değişikliğin sonucunun, genin kromozom içerisindeki konumuna bağlı olduğudur. Diğer bir ifade ile, kromozomun genlerinin birinci elemanına uygulanacak bir değişiklik, son elemanına uygulanacak bir değişiklikten çok daha belirgindir. 8 elemanlı ikili sayıdan oluşan bir kromozomun birinci elemanında yapılacak bir değişikliğin ondalık karşılığı 128 iken (2^7), 8. elemanında yapılacak değişikliğin ondalık karşılığı 1'dir. Bu sebeple, iterasyonun sonuna yaklaştıkça daha küçük değişikliğe sebep olacak mutasyona izin verilmesi uygun görülmektedir [20].

3.4. GA'nın Çalışma Yöntemi

Standart bir GA'nın çalışma yöntemi aşağıda açıklandığı gibi özetlenebilir:

1. Olası çözümlerin kodlandığı bir çözüm grubu oluşturulur. Çözüm grubuna biyolojideki benzerliği nedeniyle popülasyon, çözümlerin kodlarına da kromozom denir. Popülasyonda bulunacak kromozom sayısı için bir standart yoktur. Genel olarak önerilen 100–300 aralığında bir büyüklüktür. Büyüklük seçiminde yapılan işlemlerin karmaşıklığı ve aramanın derinliği önemlidir. Popülasyon bu işlemden sonra rastgele oluşturulur. Kromozom sayısı belirlendikten sonra probleme bağlı olarak kromozomların kodlanması gerekmektedir.

2. Popülasyondaki her kromozomun ne kadar iyi olduğu bulunur. Kromozomların ne

kadar iyi olduğunu bulan fonksiyona uygunluk fonksiyonu denir. Bu fonksiyon işletilerek kromozomların uygunluklarının bulunmasına ise evrimleşme adı verilir. Bu fonksiyon GA'nın beynini oluşturmaktadır. GA'da probleme özel çalışan tek kısım bu fonksiyondur. Uygunluk fonksiyonu, kromozomları problemin parametreleri haline getirerek onların bir bakıma şifresini çözmekte ve sonra bu parametrelere göre hesaplamayı yaparak kromozomların uygunluğunu bulmaktadır. Çoğu zaman GA'nın başarısı bu fonksiyonun verimli ve hassas olmasına bağlıdır.

3. Seçilen kromozomlar eşlenerek çaprazlama ve mutasyon operatörleri uygulanır. Bu işlemler sonucunda yeni bir popülasyon oluşturulur. Eşleme, kromozomların uygunluk değerlerine göre yapılır. Bu seçimi yapmak için rulet tekerleği seçimi, sıralama seçimi ve sabit durum seçimi gibi seçme yöntemleri kullanılır. Çaprazlama işlemi toplumda çeşitliliği sağlarken, mutasyonun etkisi yalnızca bir çözüm üzerinde olmaktadır.

4. Yeni kromozomlara yer açmak için eski kromozomlar ortadan kaldırılır. Eski kromozomlar çıkartılarak sabit büyüklükte yeni bir popülasyon sağlanır.

5. Tüm kromozomların uygunlukları tekrar hesaplanır ve yeni popülasyonun başarısı bulunur.

6. İşlemler tekrarlanarak verilmiş zaman içerisinde daha iyi olan yeni nesillerin oluşturulması gerçekleştirilir.

7. Sonuçta popülasyonların hesaplanması sırasında en iyi kromozomlar bulunduğunda çözüm elde edilmiş olur [21].

4. GENETİK YAPAY SİNİR AĞLARI (GYSA)

4.1. Giriş

Katmanlı bir yapay sinir ağının eğitimi esnasında çeşitli sorunlarla karşılaşılmaktadır: i) Ağın topolojisini en iyi belirleyecek bir yöntem mevcut değildir, katman sayısı ve katmanlardaki hücre sayıları tasarımcı tarafından belirlenmek zorundadır. ii) Ağın eğitiminde kullanılan geriye yayılım algoritmasının bazı sakıncaları vardır. iii) Çok katmanlı ağda hücre sayısı bir defa belirlendikten sonra eğitim sırasında değiştirilemez [26].

Gizli katmandaki hücre sayısı öğrenme performansını etkileyen bir parametredir. Eğer gizli katmandaki hücrelerin sayları çok azsa, ağın öğrenme işlemi optimum değere yaklaşmaz ve hata fonksiyonun dalgalı bir davranışı ortaya çıkar. Bundan dolayı girdi-çıktı tasarımları arasındaki ilişkiyi öğrenemez. Eğer hücrelerin sayısı çoksa, ağ sadece girdi-çıktı listesini depolayacak ve zayıf bir genelleme performansı sergileyecektir. Bu durum, optimal YSA boyutunun veri yapılarına uygun olması ve problemle tam anlamıyla bağlantılı bir model inşa etmesi gerektiği anlamına gelmektedir [1,18].

Geriye yayılım algoritması, YSA çıktısı ile hedef değer arasındaki farktan doğan, hatanın karesinin ortalaması veya bu ortalamanın karekökünün minimizasyonunu hedefler. Yalnız bu algoritma çok tepeli fonksiyonlarda yerel minimum problemiyle baş edemez ve türevlenemeyen fonksiyonlarla işlem yapamaz [28]. Geriye yayılım algoritmasında, amaç fonksiyonun yerel minimumlara yakalanma riski vardır ve optimum sonuca yavaş ulaşır. GY geniş, multimodal ve türevi alınamayan fonksiyonların global minimumunu bulmada etkisizdir. Bu nedenlerle, yapay sinir ağının yapısal optimizasyonu ve parametre adaptasyonu üzerine çeşitli çalışmalar yapılmıştır [8,9].

Yapay sinir ağının yapı ve parametre optimizasyonunu sağlamak için, global arama algoritmalarına dayanan, yerel minimumlara yakalanmayan, karmaşık, çok tepeli ve türevlenemeyen fonksiyonların da dahil olduğu geniş bir problem uzayına çözüm sağlayabilen GA kullanılmaktadır. Hata fonksiyonun türevine ihtiyaç duymaması nedeniyle, türevlenemeyen hata fonksiyonlarıyla karşılaşıldığında sıkıntı yaşamaz. Ayrıca, eğitilmek istenen YSA'nın türüyle ilgili kısıtlama getirmez.

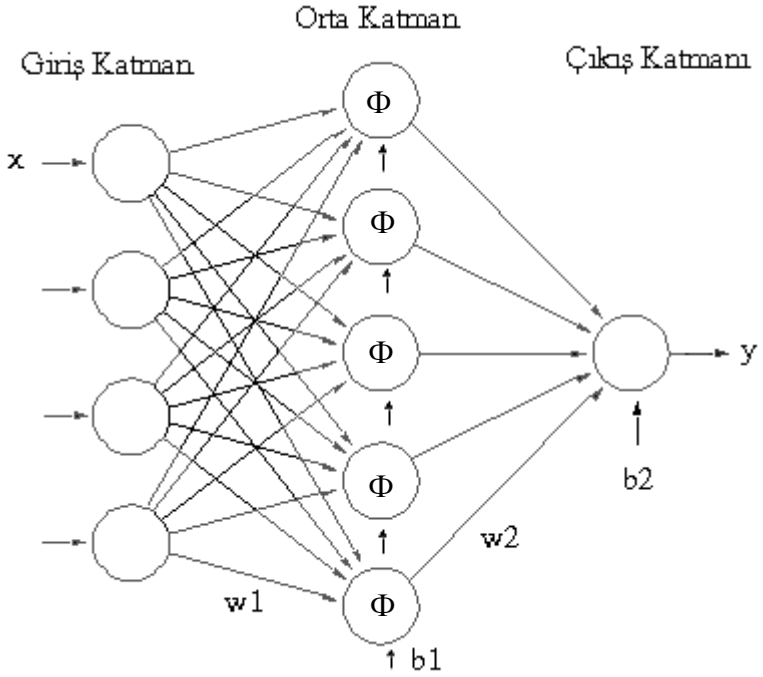
Genetik algoritmalar, performans kriteri ve bir popülasyon (çözüm kümesi) kullanarak global optimum değerini arayan paralel optimizasyon algoritmalarıdır. Genetik algoritmalar karmaşık ve düzensiz çözüm kümelerinde kullanıldıklarında başarılı sonuçlar verirler. Farklı çeşitteki optimizasyon problemlerine uygulandıklarında başarılı sonuçlar verdikleri gözlemlenmiştir. Buna ek olarak, genetik algoritmalar çok boyutlu ve lineer olmayan optimizasyon problemlerinde de başarılı sonuçlar vermiştir. Tamsayı programlama, dinamik programlama, branch-and bound metodları gibi bileşik optimizasyon tabanlı diğer algoritmalar çokta fazla olmayan değişkenler için bile çalışma zamanı ve kullandıkları kaynaklar bakımından pahalıdırlar ve sadece sınırlı sayıdaki alternatiflerle başa çıkabilirler [29].

Genetik algoritmalar ile ağa ait ve klasik yöntemlerde kullanıcı tarafından girilmesi gereken değerler en iyi biçimde bulunurken, genetik algoritmaların sunduğu araştırma metodu sayesinde yerel minimumda kalma riski olmaz. Genetik algoritmalar ile yapay sinir ağının eğitilmesi sırasında hem ağın topolojisi, hem de ağa ait ağırlıklar belirlenebilir [26].

Genetik algoritmalar ile yapay sinir ağlarının birlikte kullanımı genellikle üç şekilde gerçekleştirilmektedir. Birinci tip uygulamada genetik algoritmalar, yapay sinir ağlarının eğitiminde kullanılacak en uygun öznitelikleri araştırır. İkinci tip uygulamada genetik algoritmalar, incelenen ağın eğitim algoritmasında bulunan parametrelerin belirlenmesinde kullanılır. Üçüncü tip uygulamada ise ağın ağırlık katsayıları ve topolojisi ya birlikte ya da ayrı ayrı genetik algoritmalar ile bulunur [30]. Bu tez çalışmasında üçüncü tip olan YSA'nın hem ağırlık katsayısı hem de topolojisinin ayarlanması sağlanmaya çalışılmıştır.

4.2. GYSA Yapısı

Bu bölümde GA metodunu kullanarak YSA'nın parametre ve yapı optimizasyonunun gerçeklenişi verilmiştir. İleri beslemeli YSA' da, hücreler katmanlar şeklinde düzenlenir ve bir katmandaki hücrelerin çıkışları bir sonraki katmana ağırlıklar üzerinden giriş olarak verilir. Giriş katmanı, dış ortamlardan aldığı bilgileri hiçbir değişikliğe uğratmadan orta (gizli) katmandaki hücrelere iletir. Bilgi, orta ve çıkış katmanında işlenerek ağ çıkışı belirlenir. Bu yapısı ile ileri beslemeli ağlar doğrusal olmayan statik bir işlevi gerçekleştirir.



Şekil 4.1 İleri beslemeli yapay sinir ağı

İleri beslemeli 3 katmanlı GYSA'nın matematiksel modeli, x- giriş vektörünü, w- ağırlık değerlerini, b- polarma ağırlılarını, δ bağlantı anahtarlarını, y- ağ çıkış vektörünü göstermek üzere Denklem 4.1'deki gibi yazılabilir.

$$y_k(t) = \sum_{j=1}^{n_h} w2_{kj}\,\delta_{kj}^2 \tan sig\left[\sum_{i=1}^{n_i} w1_{ji}\,\delta_{ji}^1 x_i + b1_j^1\,\delta_j^1\right] + \delta_k^2 b2_k \qquad (4.1)$$

Burada;

$k = 1,2,.....,n_o$

$y_k(t)$: Bir t değişkenin fonksiyonları olan çıktıları.

$x_i(t)$: Bir t değişkenin fonksiyonları olan girdileri.

n_i :Girdilerin sayısı.

n_h : Gizli katmandaki hücrelerin sayısı.

$w1_{ji}$: j. gizlenmiş hücreyle i. girdi arasındaki bağlantı ağırlığı.

$w2_{kj}$: k. çıktının ve j. gizlenmiş hücrenin arasındaki bağlantı ağırlığı

$b1_j$ ve $b2_k$: Gizlenmiş hücrelerin ve çıktı hücresinin polarma ağırlıları.

tansig(.): Sigmoid fonksiyonu gösterir. Denklem 4.2'de tanjant sigmoid fonksiyonun matematiksel ifadesi gösterilmiştir.

$$\tan sig(a) = \frac{1-e^{-v}}{1+e^{-v}} \tag{4.2}$$

$\delta_{kj}^{2}, \delta_{ji}^{1}, \delta_{j}^{1}$ ve δ_{k}^{2} yapay sinir ağının bağlantı anahtarlarıdır.

δ_{kj}^{2} =1 k. çıktının j. gizlenmiş hücreyle bağlantılı olduğunu gösterir.

δ_{kj}^{2} =0 k. çıktının j. gizlenmiş hücreyle bağlantılı olmadığını gösterir.

δ_{ji}^{1} =1 j. gizlenmiş hücrenin i. girdi ile bağlantılı olduğunu gösterir.

δ_{ji}^{1} =0 j. gizlenmiş hücrenin i. girdi ile bağlantılı olmadığını gösterir.

δ_{j}^{1} =1 ve δ_{k}^{2} =1 polarma ağırlılarının gizlenmiş ve çıktı tabakalarına bağlantılı olduğunu gösterir.

δ_{j}^{1} =0 ve δ_{k}^{2} =0 polarma ağırlılarının gizlenmiş ve çıktı tabakalarına bağlantılı olmadığını gösterir.

4.2.1. Kromozomların Belirlenmesi

GA'da problemin olası çözümleri kromozomlara kodlanır. Kodlama istemi seçilen problemin yapısına uygun olarak seçilir. YSA'nın hem ağ yapısını hem de parametrelerini ayarlamak için "gerçek kodlama" kullanılır. Gerçek kodlamada her bir kromozom karar ağacının vektörüyle aynı uzunlukta olmak üzere floating sayıların bir vektörü olarak deşifre edilir [31]. Gerçek kodlamalı GA hem daha hassas hem de bilgisayar belleğinde daha az yer kaplar [23]. Bu tez çalışmasında ağırlık ve polarmalar için gerçek kodlama, bağlantı anahtarları için de binary kodlama kullanılmıştır.

$$P = \left\{ w1_{11}.....w1_{kj}, b1_{1}.....b1_{j}, w2_{11}.....w_{ji}, b2_{1}.....b2_{k}, \delta_{11}^{1}.....\delta_{ji}^{1}, \delta_{1}^{1}.....\delta_{j}^{1}, \delta_{11}^{2}.....\delta_{kj}^{2}, \delta_{11}^{2}.....\delta_{k}^{2} \right\}$$

4.2.2. Başlangıç Popülasyonu

GA'nın çalışabilmesi için başlangıç popülasyonu oluşturulur. Popülasyonun belirlenmesinde iki önemli nokta vardır. Bunlardan birincisi hangi büyüklükte olacağı ikincisi ise başlangıç değerlerinin nasıl seçileceğidir. Genetik algoritmalar tasarlanırken popülasyon büyük seçilirse, genetik algoritmanın istenen çözüme ulaşması daha uzun zaman alır. Tersine popülasyon çok küçük seçilirse bu durumda popülasyon içerisindeki kromozomların çeşitliliği düşeceği için lokal minimumda kalmasına neden olacaktır [32].

Başlangıç popülasyonu oluştururken birinci adım popülasyonu oluşturacak olan kromozomları ve kromozom sayısını belirlemektir. Bu çalışmada popülasyonu; ağırlıklar, polarma değerleri ve ağırlıklar ile gizli hücreler arasındaki bağlantının olup olmadığını simgeleyen bağlantı anahtarları oluşturdu.

$$P_1^N = \left\{ w1_{11}.....w1_{kj}, b1_1.....b1_j, w2_{11}.....w_{ji}, b2_1.....b2_k, \delta_{11}^1.....\delta_{ji}^1, \delta_1^1.....\delta_j^1, \delta_{11}^2.....\delta_{kj}^2, \delta_{11}^2.....\delta_k^2 \right\}$$

Burada

P : Popülasyonu oluşturan kromozomlar.

N : Popülasyondaki kromozom sayısı.

Başlangıç popülasyonu oluştururken ağırlıklar ve polarma değerleri rastgele sayılardan seçildi. İlk popülasyonda ağ yapısı tam bağlantılı olarak alındı. Bu nedenle bağlantı anahtarlarının tümü 1 olarak seçildi. Başlangıç popülasyonu 50 olarak belirlendi.

N_{ipop} =50

4.2.3. Değerlendirme

Uygunluk fonksiyonu her bir kromozomun bir sonraki jenerasyonda bulunup bulunmayacağını gösteren parametredir. Çözümün optimal olması, uygunluk değeri kriterine bağlıdır. Yüksek uygunluk değerine sahip olan kromozomun hayatta kalabilme olasılığı daha yüksektir [33].

Çalışmamızda uygunluk fonksiyonu oluşturmak için hata diye nitelendirdiğimiz sistem çıkışı ile ağ çıkışının mutlak değer farklarının ortalamasını kullandık. Her kromozomun uygunluk değeri Denklem 4.3 ve 4.4 deki formüllere göre hesaplandı.

$$hata = \frac{\sum_{1}^{N} \left| y_{hedef} - y \right|}{N} \tag{4.3}$$

$$uygunluk \;\; fonksiyonu = \frac{1}{hata} \tag{4.4}$$

Burada:

y_{hedef} : Referans sinyal verilerek sistemden elde edilen çıkış değeri.

y : Ağın çıkış değeri.

N : Örnek sayısını ifade eder.

4.2.4. Doğal Seçim

Başlangıç popülasyonunun fazla olduğu durumlarda bazı iteratif adımlarla en uygun değere sahip olan kromozomlar seçilir. Böylece en iyi olan kromozomlar bir sonraki jenerasyona katılırken, kötü olan kromozomlar elenmiş olur.

Çalışmamızda başlangıç popülasyonu 50 kromozomdan oluşmaktadır. Her kromozom için uygunluk fonksiyonu kullanılarak uygunluk değeri hesaplanır. Kromozomlar elden edilen uygunluk değerlerine göre büyükten küçüğe sıralanır. En iyi değere sahip olan 24 kromozom alınır (N_{pop}), kalan kromozomlar elenir. Seçilen N_{pop} içerisinden, uygunluk değeri yüksek olan ilk 12 kromozom N_{iyi} olarak, kalan 12 kromozom da $N_{kötü}$ olarak belirlenir. Bundan sonraki her bir iterasyonda en iyi değere sahip olan 24 kromozom elde tutulur. Bu kromozomlar artık yeni nesli oluşturacak olan ebeveynler olarak eşleme havuzuna alınır. Tablo 4.1'de N_{pop}, N_{iyi}, $N_{kötü}$ kromozomlarının nasıl oluşturulduğu gösterilmektedir.

Tablo 4.1. Kromozomların uygunluk değerlerine göre sıraya dizilişi

Kromozomlar	Uygunluk Değerleri
N_{ipop} { N_{iyi} { P_1	2.0064
P_2	1.7664
P_3	.....
P_4	.....
.	.....
.	.....
P_{12}	1.5077
$N_{kötü}$ { P_{13}	1.4583 } N_{pop}
P_{14}	.....
P_{15}	.....
P_{16}	.....
.	1.3723
.	1.3416
P_{24}	1.3413
P_{25}	1.3303
P_{26}	.....
P_{27}	.....
P_{28}	. ....
.	1.2348
.	1.2290
P_{49}	1.2048
P_{50}	1.1990
	1.1881

4.2.5. Eşleme

İki adet yeni nesil üretmek için N_{iyi} kromozomların bulunduğu eşleştirme havuzundan iki tane kromozom seçilir. Eşleştirme, seçilen kromozomlar arasında gerçekleştirilir.

Kromozomların eşlenmesinde rastgele sayı üreteci kullanılır. Kromozomlar 1'den başlanarak N_{iyi}'ye kadar sıralanır. Eşlenecek olan kromozomları bulmak için 1 ile 12 arasında rastgele sayı üretilir. Elde edilen sayı indislerine sahip kromozomlar eşleştirilir. Örneğin bu sayılardan ilk dördü 2, 10, 3, 7 olsun. Bu durumda 2. kromozomla 10. kromozom, 3. kromozomla 7. kromozom eşleşir.

4.2.6. Çaprazlama

İki adet yeni nesil elde etmek için kromozomların bulunduğu eşleme havuzundan iki adet kromozom seçilir. Eşleme sürecinde, seçilen kromozomlardan bir ve birden fazla yeni nesil oluşturma olayına "çaprazlama" denir. En yaygın olarak kullanılan iki kromozomdan iki tane yeni nesil elde edilmesidir.

Çaprazlama işleminin uygulanması için ilk önce bu işleme tabii olacak kromozom sayısı, çaprazlama oranına göre seçilmelidir. Bu oran popülasyondaki toplam kromozom sayısıyla çarpılarak, eşlenecek olan kromozom sayısı bulunur.

Çalışmamızda çaprazlama oranı 0.5 olarak alındı. Buna göre çaprazlama işlemine göre 24*0.5=12 kromozom katıldı. İlk olarak iki kesme noktası oluşturuldu. Birinci kesme noktası ağırlık ve polarma değerlerinin kromozomda oluşturduğu bölüm için, diğeri ise bağlantı anahtarları değerleri için kullanıldı.

Örneğin iki anne baba $\Theta_1 = (x_1, x_2 .....x_p, \delta_{a1}, \delta_{a2} .....\delta_{ap})$ ve $\Theta_2 = (y_1, y_2 .....y_p, \delta_{b1}, \delta_{b2} .....\delta_{bp})$ olsun. *k.* ve *m.* pozisyondan sonra çaprazlama olursa;

$\Theta_1^{'} = (x_1, x_2 .....x_k^{'}, y_{k+1}, y_{k+2} ....y_p, \delta_{a1}, \delta_{a2} .....\delta_{bm}, \delta_{b(m+1)} .....\delta_{bp})$

$\Theta_2^{'} = (y_1, y_2 .....y_k^{'}, x_{k+1}, x_{k+2} ....x_p, \delta_{b1}, \delta_{b2} .....\delta_{am}, \delta_{a(m+1)} .....\delta_{ap})$ kromozomları elde edilir. Kesme noktalarındaki $x_k^{'}$ ve $y_k^{'}$ değişkenlerinin yeni değerleri hesaplanırken Denklem 4.5 kullanılır.

$$x_k^{'} = \beta x_k - (1-\beta) y_k$$
$$y_k^{'} = \beta y_k - (1-\beta) y_k \qquad (4.5)$$

Burada

$x_k^{'}$: Anne kromozomun k. değişkenini,

$y_k^{'}$: Baba kromozomun k. değişkenini,

m : Bağlantı anahtarı değerleri için seçilen kesme noktasını,

k : Ağırlık ve polarmalar için rastgele seçilen kesme noktasını,

β : 0-1 arasında rastgele üretilen bir sayıyı temsil eder.

4.2.7. Mutasyon

Mutasyonun rolü, arama algoritmasının bir yerel optimuma takılmamasının garantisini sağlamaktır. Seçim sırası ve çaprazlama operatörleri herhangi bir homojen çözüm kümesinde durgunlaşabilir. Böyle şartlar altında, tüm kromozomlar özdeştir ve bu yüzden popülasyonun ortalama uygunluğu geliştirilemeyebilir. Çözüm sadece optimal (veya yerel olarak oldukça optimal) olmak görünebilir. Çünkü arama algoritması daha fazla ilerlemeyebilir. Mutasyon rastgele bir aramaya eşdeğerdir ve genetik farklılıkların kaybının korunmasında bize yardım eder [19].

Çaprazlama sonucu elde edilen kromozomlara popülasyondaki çeşitliliği artırmak için mutasyon uygulanarak yeni popülasyonun oluşturulur. Mutasyon işlemine girecek olan kromozom sayısı, mutasyon oranına göre belirlenir.

Gerçek değerli kodlamada mutasyon işlemi seçilen gene küçük bir sayı eklemek ya da çıkarmakla gerçekleşir. İkili kodlamada ise 0'lar 1'e, 1'ler ise 0'a dönüştürülür.

Çalışmamızda belirlenen mutasyon oranına göre rastgele seçilen genler mutasyon işlemine tabi tutuldu. Bu işleme geçilmeden önce tıpkı çaprazlama işleminde yapıldığı gibi 2 kesme noktası oluşturuldu. Birinci kesme noktası, ağırlık ve polarma değerlerinin oluşturduğu kısım için, diğer kesim noktası ise bağlantı anahtarlarını gösteren kısım için seçildi.

Örneğin $P_1 = (x_1, x_2 .....x_k^{'}, x_{k+1}, x_{k+2} ....x_n, \delta_1, \delta_2 .....\delta_m^{'}, \delta_{(m+1)} .....\delta_n)$ kromozomu mutasyon işlemine tabi tutulsun.

$$a = \ random \in [-0.5, 0.5]$$

$$x_k^{'} = a + x_k^{'} \tag{4.6}$$

$\delta_m^{'} = 1$ ise 0, 0 ise 1 olur.

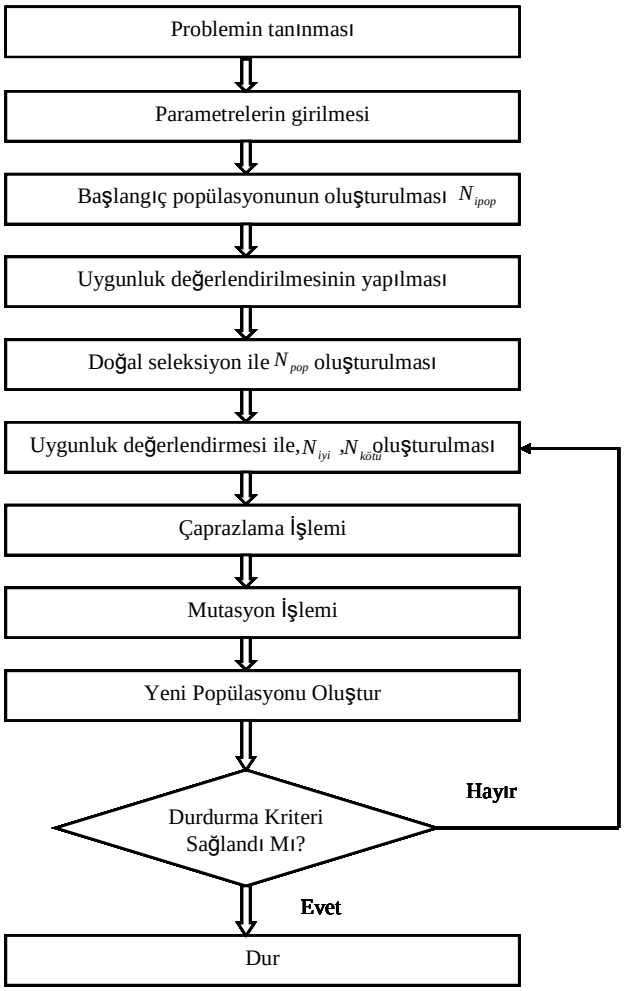
Kesme noktası göre seçilen ağırlık ya da polarma değerine [-0.5,0.5] arasında rastgele seçilen bir değer eklenir. Bağlantı anahtarlarının değerinde ise mutasyon işlemi yapılırken anahtar değeri 1 ise 0'a, 0 ise 1'e çevrilir.

4.2.8. Gelecek Nesil

İterasyon sonucunda, çaprazlamaya ve mutasyona uğramış kromozomların uygunluk değerleri hesaplanarak yüksel uygunluk değerine sahip olanlarla yeni nesil elde oluşturulur. Yeni nesildeki kromozomlar bir sonraki adımda uygunluk değerlerine göre yeniden sıralanır ve eşleştirme havuzuna konur. Eşleştirme havuzunda kromozomların sadece 12 tanesi tutulmaktadır. Tekrar çaprazlama, mutasyon ve sıralamaya tabi tutulan kromozomlar bir diğer nesli oluşturur. Bu işlem istenilen uygunluk değeri elde edilinceye kadar ya da iterasyon sayısı tamamlanıncaya kadar devam eder.

4.2.9. GYSA Algoritması

Şekil 4.2'de GYSA algoritmasının akış şeması görülmektedir.



Şekil 4.2. GYSA algoritmasının akış diyagramı

İşlem Basmakları

1. Adım : Sisteme verilen referans sinyallerle y_{hedef} değerinin elde edilir.

2. Adım : Ağırlıklar ve polarma değerleri için rastgele değer atanarak parametreler girilir.

3. Adım : Başlangıç popülasyonun oluşturulur. N_{ipop}

4. Adım : Her bir kromozom için uygunluk değeri hesaplanır.

5. Adım : Uygunluk değerlerine göre kromozomlar büyükten küçüğe doğru sıraya dizilir. Doğal seleksiyon ile sıralanan N_{ipop}'taki kromozomların ilk 24'ü N_{pop}'u oluşturur. Bu adımdan sonra her iterasyonda popülasyon sayısı N_{pop} kadar olacaktır.

6. Adım : N_{pop} sayısının ilk 12 kromozomu N_{iyi}, kalan 12 kromozomu $N_{kötü}$ olarak nitelendirilir.

7. Adım : N_{iyi} kromozomları yeni nesil oluşturması için bir sonraki adıma taşınır, $N_{kötü}$ kromozomlar popülasyondan atılır.

8. Adım : N_{iyi} kromozomları rastgele seçim işlemine tabi tutularak çaprazlamaya girecek olan çiftler seçilir.

9. Adım : Çaprazlama oranına göre çaprazlama işlemi yapılır.

10. Adım : Mutasyon oranına göre mutasyon işlemi yapılır.

11. Adım : Çaprazlama ve mutasyon sonrası elde edilen 12 kromozom popülasyona katılır. Böylece popülasyon sayısı tekrar 24'e yükselir. Yeni N_{pop} oluşturulmuş olur.

12. Adım : Yeni popülasyonun uygunluk değeri hesaplanır.

13. Adım : Durdurma kriterinin sağlanıp sağlanmadığına bakılır. Eğer hata istenilen düzeyde azalmışsa ya da iterasyon sayısı tamamlanmışsa Adım 14'e geçilir. Aksi halde 5. adıma dönülür.

14. Adım : Sistemin çıkışını ve ağın çıkışını çizilir.

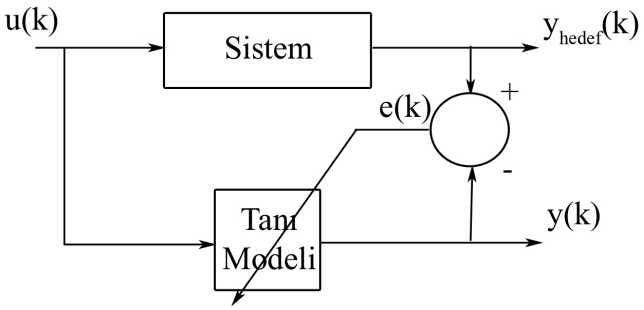
15. Adım : Optimum kromozomları ve sistemin hatasını gösterilir.

16. Adım : Dur.

5. GYSA İLE SİSTEM MODELLEME

5.1. Sistem Modelleme

Sistem modelleme, deneysel veya matematiksel yolla elde edilmiş verilerden faydalanarak sistemlerin modellerinin elde edilmesidir. Sistem modellemede amaç, bilinmeyen bir sistemin transfer fonksiyonu adı verilen geçiş eğrisinin yani içyapısının belirlenmesidir. Sistemin etkin bir şekilde kontrol edilebilmesi, çıkış değeri hakkında her zaman bilgi sahibi olunabilmesi ve geliştirilebilmesi açısından sistem modelinin ve tahmin edilen parametre değerlerinin mümkün olan en az hata ile belirlenmesi gerekir. Şekil 5.1'de sistem modelleme gösterilmiştir.



Şekil 5.1. Sistem Modelleme

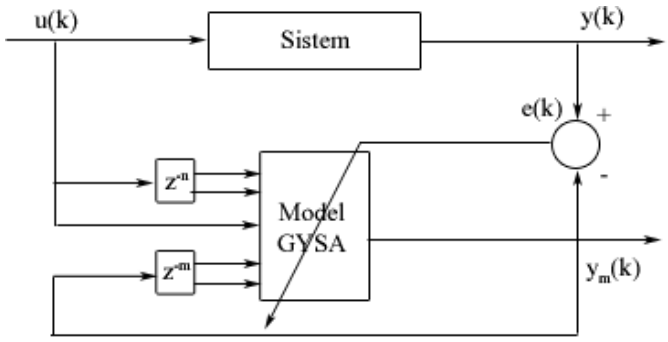
Modelleme işleminin temel basamakları aşağıdaki gibi verilebilir:

- Sistemin girişine herhangi bir işaret (darbe, basamak, sinüs veya rastgele işaretler) uygulanarak, sistemin bu işarete verdiği cevabı çıkış işareti olarak kaydedilir.
- Sisteme uygun bir model yapısı tespit edilir. Bu model yapısı doğrusal olabileceği gibi, çoğu fiziksel sistemin davranışını gösteren doğrusal olmayan model yapısı da kullanılabilir.
- Elde edilen modelin parametreleri, bazı istatistikî veya tahmini yöntemlerle belirlenir. Modelleme işleminin en önemli aşaması, bu parametrelerin doğru şekilde belirlenmesidir.
- Parametreleri belirlenen modelin girişine, sisteme uygulanmış olan giriş işareti uygulanıp, modelden alınan çıkış işareti ile sistemin gerçek çıkışı arasındaki fark bulunur.

Eğer fark büyükse, başka bir model yapısı veya yeni bir parametre tespit yöntemi belirlenir. Eğer fark çok küçükse bu model sistemi tanımlamak ve kontrol etmek için kullanılabilir [34].

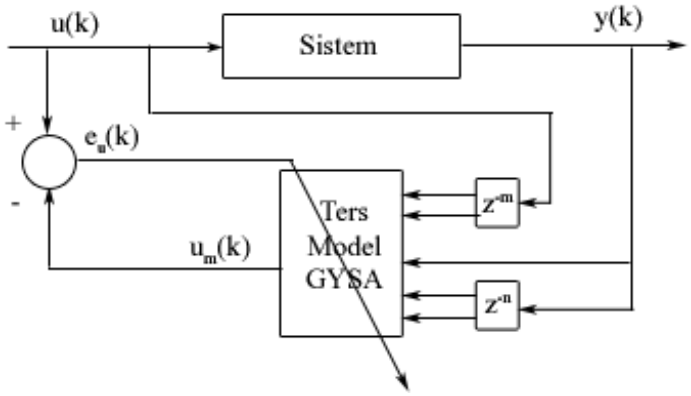
5.1.1. Düz Modelleme

Bir sistemin düz dinamiklerini gösterecek şekilde, YSA tanı modelinin seçilmesi ve eğitilmesi düz modelleme olarak söylenir. Bir sistemin düz yön dinamikleri matematiksel olarak, $y(k) = f(y(k-i), u(k-j))$ ile ifade edilir. Şekil 5.2'de NARX ya da seri paralel model yapısına göre bir sistemin YSA tanı modeli verilmiştir.



Şekil 5.2. GYSA ile düz modelleme, NARX modeli

Sistem tanılamada NARX model, kararlılık analizlerinin yapılabilir olması ve YSA modellerin statik geriye yayılım algoritması ile daha kolay ve hızlı eğitilebilmesi nedeni ile tercih edilir. Sistemlerin uyarlamalı denetiminde, denetim sisteminin bozucu girişlere karşı davranışının iyi olabilmesi için denetlene sistemin dinamik davranışlarının, model ve denetleyici YSA'da tam olarak temsil edilmesi gerekir. Buda ancak sistem dinamiklerinin YSA yapısında tam doğru olarak tanılanması ile sağlanabilir. Bu nedenle, özellikle gürültülü sistemlerde geri beslemeli YSA tanı modellerine gerek duyulur ve NOE modeli olarak bilinen bir sistemin geri beslemeli düz YSA modeli Şekil 5.3'te verilmiştir.



Şekil 5.3. GYSA ile düz modelleme, NOE modeli

Gürültülü sistemlerde, gerçek sistem çıkışı üzerindeki gürültünün neden olduğu polarma etkisinin önlemesi ve sistem dinamiklerini doğru olarak temsil etmesi nedeniyle NOE modelin kullanılması gerekebilir. Doğrusal sistemlerin modellenmesinde bile geri beslemeli modellerin (OE) parametrelerinin uyarlanmasında kararlılık şartını kesin olarak belirlemek oldukça zordur. Bu nedenle NOE modellerde, model parametrelerinin doğru değerlerinin yakınsayacağı kesin olarak söylenemez [14].

5.2. GYSA ile Sistem Modelleme

Bu bölümde 3 farklı sistem üzerinde yapay sinir ağlarının yapı ve parametre optimizasyonu, genetik algoritma ile sağlanmaya çalışıldı. Her sistem, tam bağlantılı 5 ve 10 gizli hücreye sahip olan sinir ağıyla eğitime tabi tutuldu.

Eğitim esnasında sisteme referans olarak verilen değerlerle elde edilen çıkış “ y_{hedef} ” olarak, GYSA ile eğitim esnasında elde edilen çıkış “ y ” olarak nitelendirildi. Denklem 5.1’e göre sistemin eğitim hata oranı elde edildi.

$$hata = \frac{\sum_{1}^{N} \left| y_{hedef} - y \right|}{N} \tag{5.1}$$

Eğitim sonunda elde edilen parametreler kullanarak, ağı test etmek amacıyla farklı referans sinyaller sisteme verildi. GA ile eğitimi tamamlanan sistemlerin, verilen referans sinyallere göre genellemesi grafiklerle gösterildi.

5.2.1. Fark Denklemleri ile Tanımlanan Sistemlerin GYSA ile Modellenmesi

Denklem 5.2'de giriş-çıkış modeli olarak da söylenen doğrusal olmayan dinamik bir sistemin matematiksel modeli verilmiştir. YSA bu sistemi modellemek için kullanılmıştır. Sinir ağı kontrol tasarımı için kullanılmadan önce standart GY algoritmasını kullanan eğitim sürecinden geçebilir. Eğitim esnasında u_k kontrolü [-1.7 1.7] arasında rastgele seçilir [35]. Bu tez çalışmasında matematiksel modeli verilmiş olan sistem GYSA ile modellenecek ve bu sistem Sistem-1 olarak ifade edilecektir.

$$y_{k+1} = \frac{0.00375 y_k y_{k-1}}{1 + 0.0025 y_k^2 + 0.0025 y_{k-1}^2} + 0.7 \sin[0.025(y_k + y_{k-1})]\cos[0.025(y_k + y_{k-1})] + 1.2u(k) \tag{5.2}$$

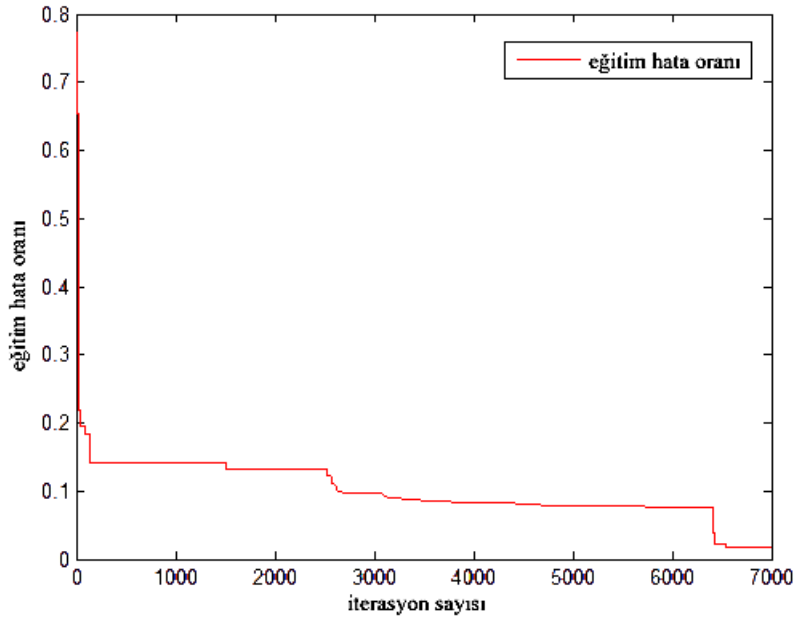
Denklem 5.2'de matematiksel modeli verilen Sistem-1'in, GYSA modeli ve eğitim parametreleri Tablo 5.1'de gösterilmektedir.

Tablo 5.1. GYSA model ve eğitim parametreleri

Model Yapısı	Narx
Orta Katmandaki Hücre Sayısı	5
Kullanılan Aktivasyon Çeşidi	Hiperbolik Tanjant Sigmoid
Sisteme Verilen Referans Girişi	[-1.7 1.7] aralığında rastgele atanan değerler.
Başlangıç Popülasyonu N_{ipop}	50
Mutasyon Oranı	0.01
Çaprazlama Oranı	0.5
İterasyon Sayısı	7000
Eğitime Başlarken Elde Edilen Hata	0.7736
Eğitim Sonunda Elde Edilen Hata	0.0168
Eğitime Başlarken Bağlantı Sayısı	26
Eğitim Sonunda Bağlantı Sayısı	7

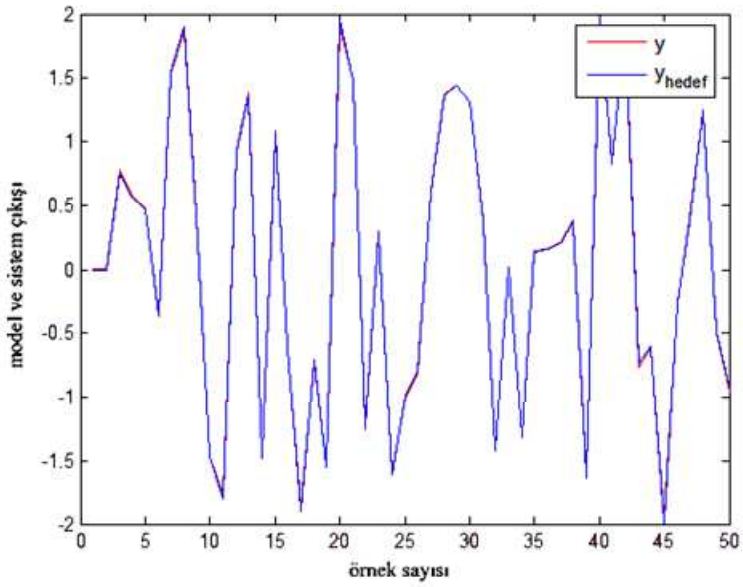
Eğitim hata oranı hesaplanırken birinci adımda N_{ipop} başlangıç popülasyonundaki 50 kromozomdan en yüksek uygunluk değerine sahip olan kromozomun eğitim hata oranı alınır. Bundan sonraki adımlarda N_{pop} popülasyonundaki kromozomlardan en yüksek uygunluk değerine sahip olan kromozomun eğitim hata oranı alınarak grafik çizdirilir.

Sistem-1 eğitilmeye başlandığında hata değeri 0.7736 iken 7000 iterasyon sonunda hata 0.0168'e düştü. Şekil 5.4'te eğitim esnasındaki hata oranının durumu gösterilmektedir.



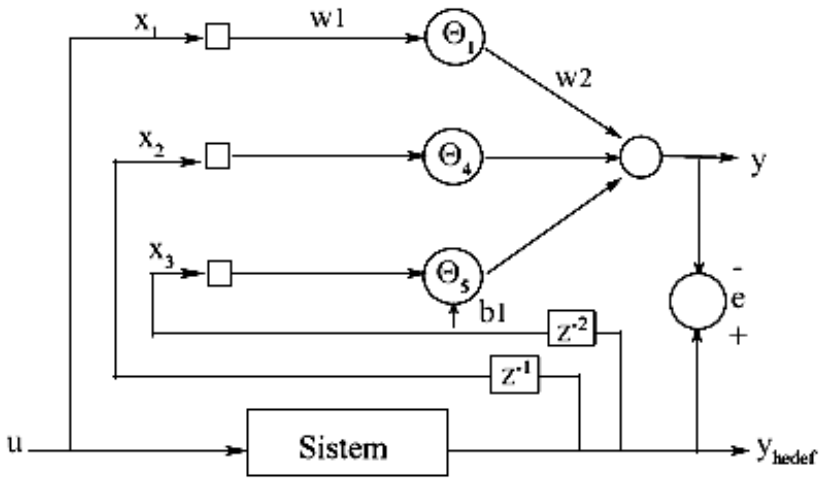
Şekil 5.4. GA kullanılarak eğitilen sinir ağının eğitim hata oranı

Eğitim sonunda istenilen değerler ile GYSA modelinin çıkış değerleri Şekil 5.5'te aynı şekil üzerine çizdirilerek modelin başarımı karşılaştırılmıştır. Buna göre modelin başarılı bir şekilde Sistem-1'i temsil ettiği görülmektedir.



Şekil 5.5. Modelin çıkış değeri ile sisteme ait çıkış değeri

Eğitime başlarken, orta katmanda 5 gizli hücreli, tam bağlantılı yapay sinir ağı kullanıldı. Eğitim sonunda elde edilen sinir ağı Şekil 5.6'daki gibi gizli katmanda 3 hücreye sahip ve bağlantı sayısı 7'dir.



Şekil 5.6. 5 gizli hücreyle eğitime başlanan Sistem-1'in, eğitim sonrası ağ yapısı

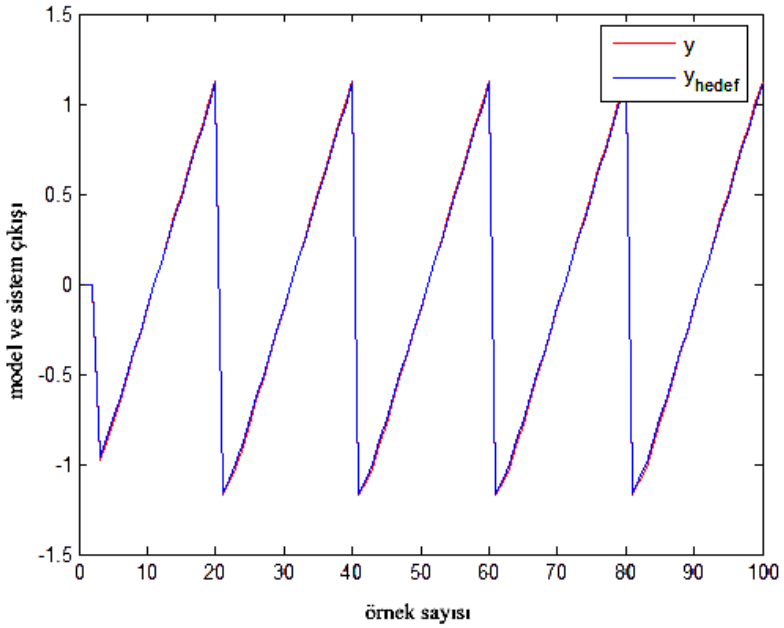
Eğitim sonunda elde edilen ağırlık ve polarma değerleri Tablo 5.2'de gösterilmektedir.

Tablo 5.2. 5 gizli hücreli GYSA modelin ağırlık ve polarma değerleri

w_1			b_1
0.2706	-	-	-
0.0056	-	-	-0.6056
-	-	-	0.0056
-	0.6737	-	-
-	-	0.3567	-0.0087

w_2					b_2
4.5917	-	-	0.0359	0.0462	-

GYSA'nın model parametrelerini kullanarak, ağı test etmek amacıyla üçgen dalga referans sinyal sisteme verildi. Eğitimi tamamlanan sinir ağının ve sistemin, üçgen dalga giriş sinyaline göre genellemesi Şekil 5.7'de gösterilmektedir. Buna göre eğitilen sinir ağının ezberleme yapmadığı ve öğrenmeyi gerçekleştirdiği görülmektedir.



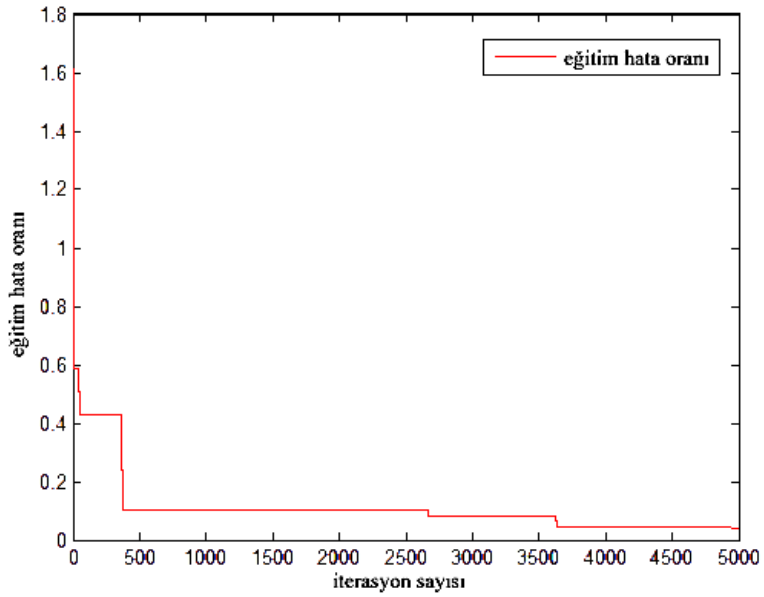
Şekil 5.7. Eğitilen sinir ağına, üçgen dalga referans giriş verilerek ağın test edilmesi

Denklem 5.2'de matematiksel modeli verilen Sistem-1'in, GYSA modeli ve eğitim parametreleri Tablo 5.3'te gösterilmektedir.

Tablo 5.3. GYSA model ve eğitim parametreleri

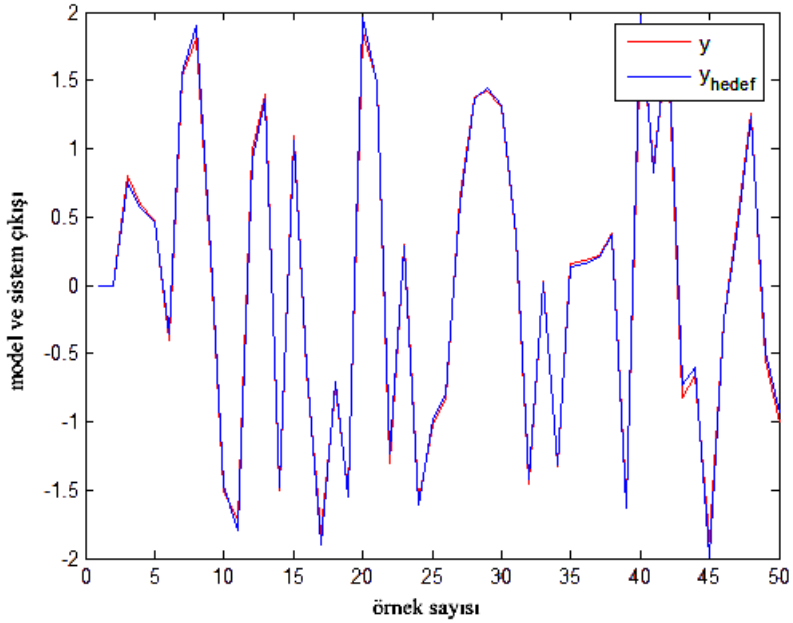
Model Yapısı	Noe
Orta Katmandaki Hücre Sayısı	5
Kullanılan Aktivasyon Çeşidi	Hiperbolik Tanjant Sigmoid
Sisteme Verilen Referans Girişi	[-1.7 1.7] aralığında rastgele atanan değerler.
Başlangıç Popülasyonu N_{ipop}	50
Mutasyon Oranı	0.01
Çaprazlama Oranı	0.5
İterasyon Sayısı	5000
Eğitime Başlarken Elde Edilen Hata	1.6236
Eğitim Sonunda Elde Edilen Hata	0.0397
Eğitime Başlarken Bağlantı Sayısı	26
Eğitim Sonunda Bağlantı Sayısı	9

Sistem eğitilmeye başlandığında hata değeri 1.6236 iken 5000 iterasyon sonunda hata 0.0397'ye düştü. Şekil 5.8'de eğitim esnasındaki hata oranının durumu gösterilmektedir.



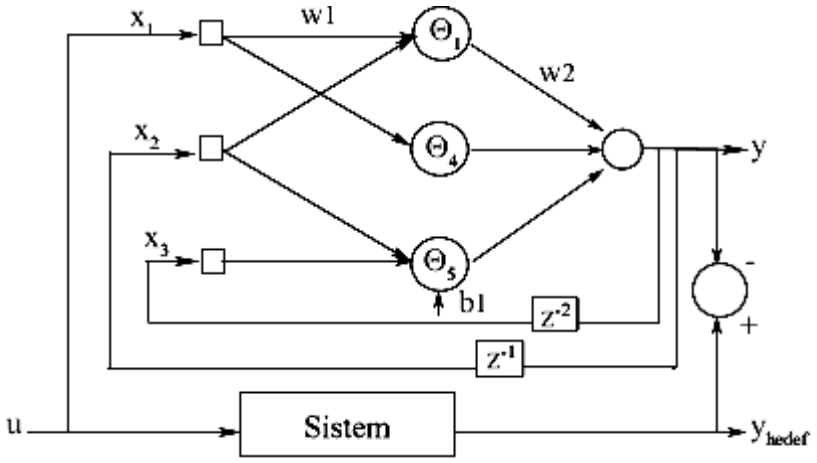
Şekil 5.8. GA kullanılarak eğitilen sinir ağının eğitim hata oranı

Eğitim sonunda istenilen değerler ile GYSA modelinin çıkış değerleri Şekil 5.9'da aynı şekil üzerine çizdirilerek modelin başarımı karşılaştırılmıştır. Buna göre modelin başarılı bir şekilde Sistem-1'i temsil ettiği görülmektedir.



Şekil 5.9. Modelin çıkış değeri ile sisteme ait çıkış değeri

Eğitime başlarken, orta katmanda 5 gizli hücreli, tam bağlantılı yapay sinir ağı kullanıldı. Eğitim sonunda elde edilen sinir ağı Şekil 5.10'daki gibi gizli katmanda 3 hücreye sahip ve bağlantı sayısı 9'dur.



Şekil 5.10. 5 gizli hücreyle eğitime başlanan Sistem-1'in, eğitim sonrası ağ yapısı

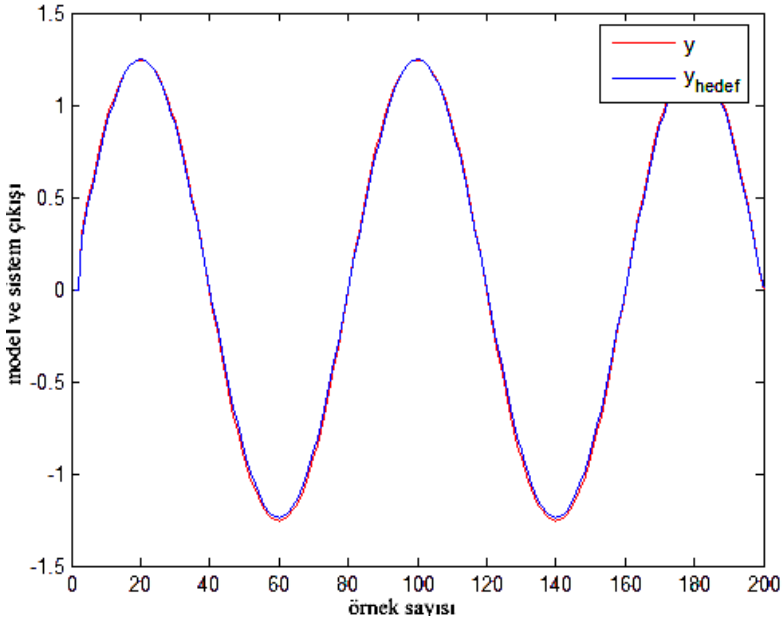
Eğitim sonunda elde edilen ağırlık ve polarma değerleri Tablo 5.4'te gösterilmektedir.

Tablo 5.4. 5 gizli hücreli GYSA modelin ağırlık ve polarma değerleri

w_1			b1
0.0176	0.4224	-	-
-	-	-	-
-	-0.3011	-	-
0.4224	-	-	-
-	0.4119	0.1156	-0.1869

w_2					b_2
0.0109	0.6770	-	3.0902	0.0393	-

GYSA'nın model parametrelerini kullanarak, ağı test etmek amacıyla sinüzoidal referans sinyal sisteme verildi. Eğitimi tamamlanan sinir ağının ve sistemin, sinüzoidal giriş sinyaline göre genellemesi Şekil 5.11'de gösterilmektedir. Buna göre eğitilen sinir ağının ezberleme yapmadığı ve öğrenmeyi gerçekleştirdiği görülmektedir.



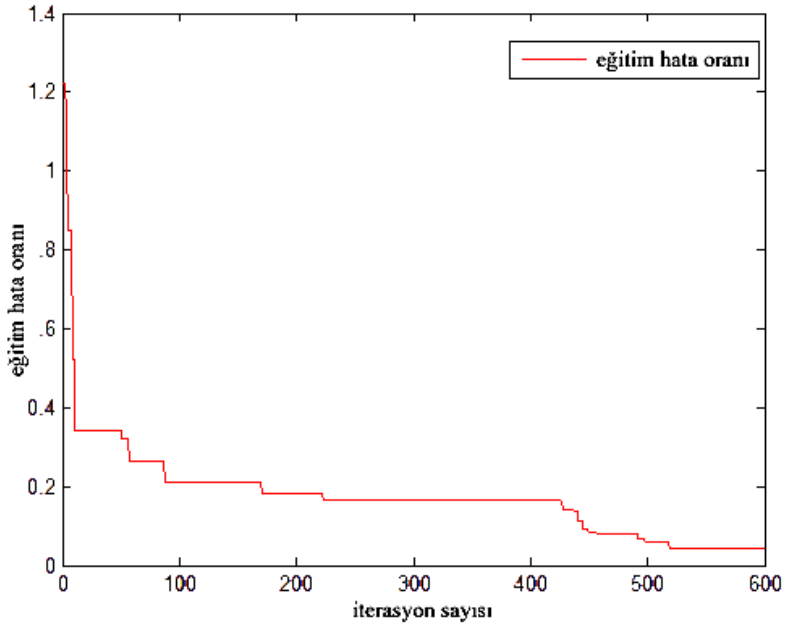
Şekil 5.11. Eğitilen sinir ağına sinüzoidal referans giriş verilerek ağın test edilmesi

Denklem 5.2'de matematiksel modeli verilen Sistem-1'in GYSA modeli ve eğitim parametreleri Tablo 5.5'te gösterilmektedir.

Tablo 5.5. GYSA model ve eğitim parametreleri

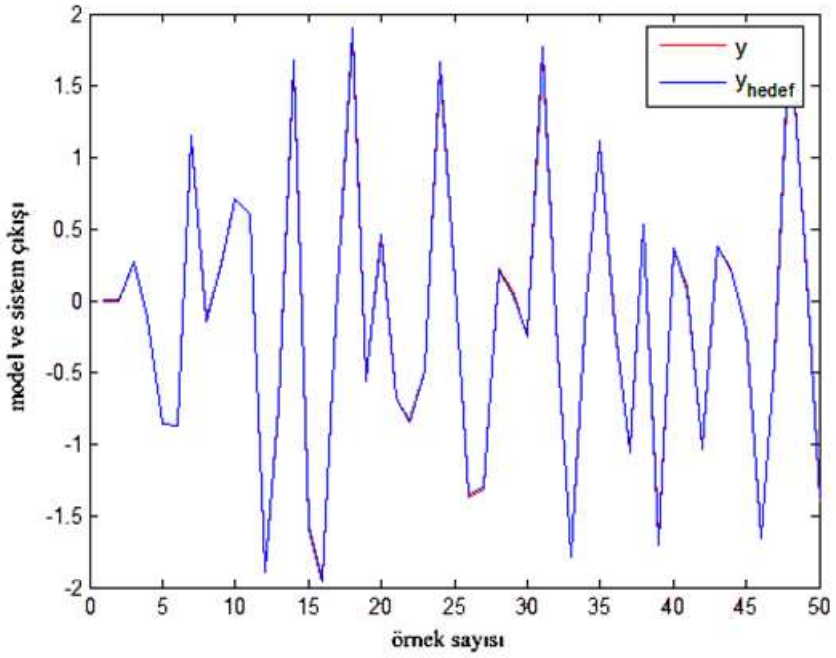
Model Yapısı	Narx
Orta Katmandaki Hücre Sayısı	10
Kullanılan Aktivasyon Çeşidi	Hiperbolik Tanjant Sigmoid
Sisteme Verilen Referans Girişi	[-1.7 1.7] aralığında rastgele atanan değerler.
Başlangıç Popülasyonu N_{ipop}	50
Mutasyon Oranı	0.005
Çaprazlama Oranı	0.5
İterasyon Sayısı	600
Eğitime Başlarken Elde Edilen Hata	1.2310
Eğitim Sonunda Elde Edilen Hata	0.0144
Eğitime Başlarken Bağlantı Sayısı	51
Eğitim Sonunda Bağlantı Sayısı	16

Sistem eğitilmeye başlandığında hata değeri 1.2310 iken 600 iterasyon sonunda hata 0.0144'e düştü. Şekil 5.12'de eğitim esnasındaki hata oranının durumu gösterilmektedir.



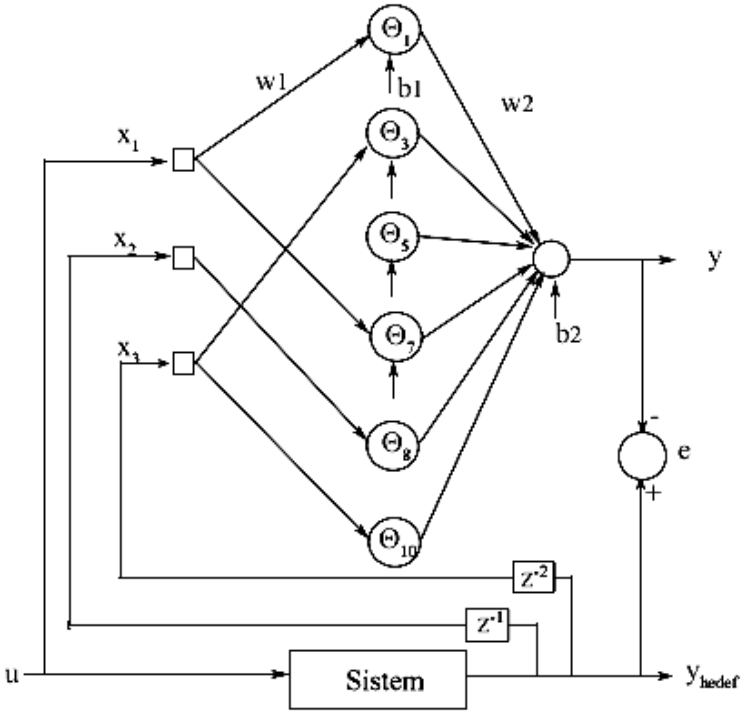
Şekil 5.12. GA kullanılarak eğitilen sinir ağının eğitim hata oranı

Eğitim sonunda istenilen değerler ile GYSA modelinin çıkış değerleri Şekil 5.13'te aynı şekil üzerine çizdirilerek modelin başarımı karşılaştırılmıştır. Buna göre modelin başarılı bir şekilde Sistem-1'i temsil ettiği görülmektedir.



Şekil 5.13. Modelin çıkış değeri ile sisteme ait çıkış değeri

Eğitime başlarken, orta katmanda 10 gizli hücreli, tam bağlantılı yapay sinir ağı kullanıldı. Eğitim sonunda elde edilen sinir ağı Şekil 5.14'teki gibi gizli katmanda 6 hücreye sahip ve bağlantı sayısı 16'tir.



Şekil 5.14. 10 gizli hücreyle eğitime başlanan sistemin, eğitim sonrası ağ yapısı

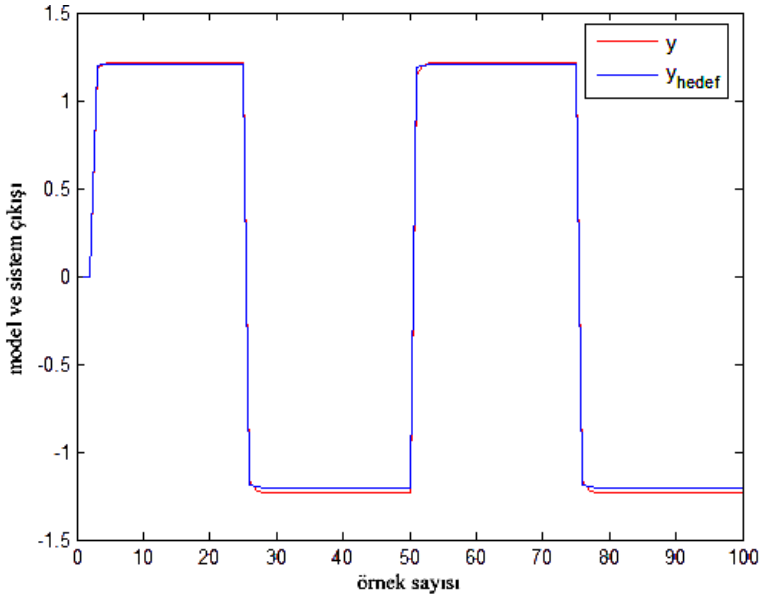
Eğitim sonunda elde edilen parametreler Tablo 5.6'da gösterilmektedir

Tablo 5.6. 10 gizli hücreli GYSA modelin ağırlık ve polarma değerleri

w_1			b_1
-0.2166	-	-	0.0115
-	-	-	-
-	-	-0.0062	1.2634
-	-	-	-0.1301
-	-	-	-2.3555
-	-	-	-
-0.3263	-	-	0.1374
-	0.0425	-	-
-	-0.7143	-	1.4239
-	-	-0.9794	-

w_2										b_2
-7.0080	1.1488	-0.1797	-	0.4429	-	0.9835	0.5363	-	-0.0064	0.5342

GYSA'nın model parametrelerini kullanarak, ağı test etmek amacıyla kare dalga referans sinyal sisteme verildi. Eğitimi tamamlanan sinir ağının ve sistemin, kare dalga giriş sinyaline göre genellemesi Şekil 5.15'te gösterilmektedir. Buna göre eğitilen sinir ağının ezberleme yapmadığı ve öğrenmeyi gerçekleştirdiği görülmektedir.



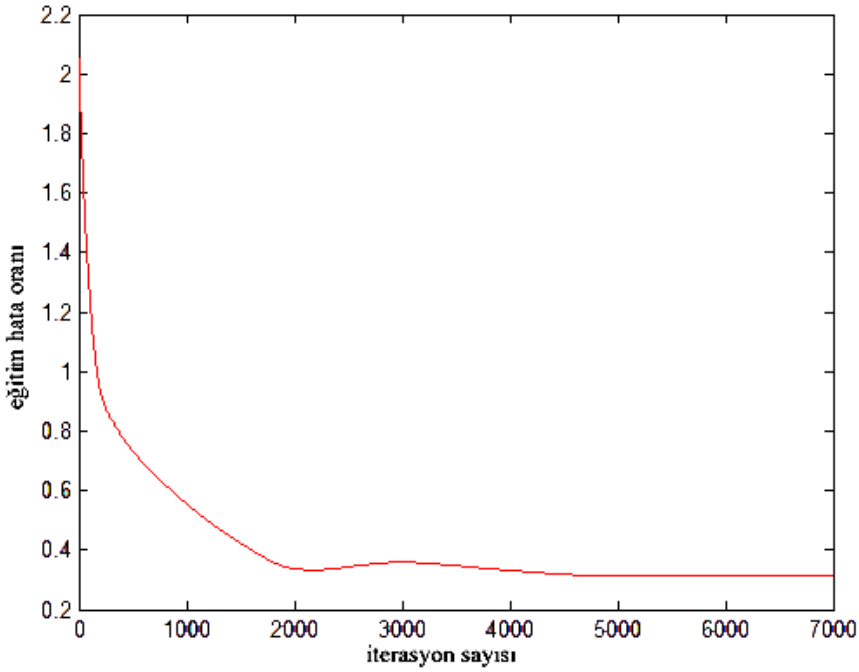
Şekil 5.15. Eğitilen sinir ağına, kare dalga referans giriş verilerek ağın test edilmesi

Denklem 5.2'de matematiksel modeli verilen Sistem-1'in Geriye Yayılım YSA modeli ve eğitim parametreleri Tablo 5.7'de gösterilmektedir.

Tablo 5.7. GYSA model ve eğitim parametreleri

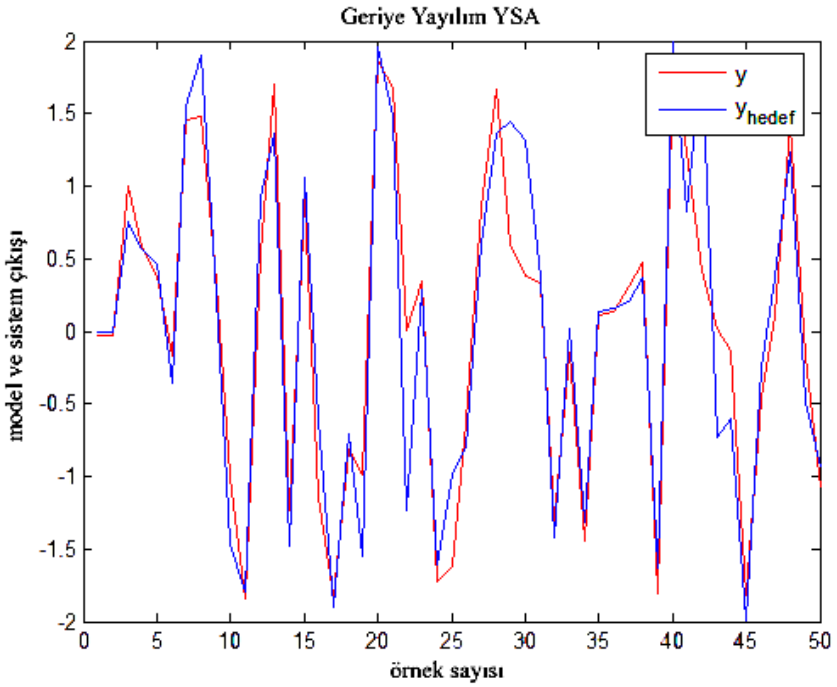
Model Yapısı	Narx
Orta Katmandaki Hücre Sayısı	5
Kullanılan Aktivasyon Çeşidi	Hiperbolik Tanjant Sigmoid
Sisteme Verilen Referans Girişi	[-1.7 1.7] aralığında rastgele atanan değerler.
İterasyon Sayısı	7000
Öğrenme Oranı	0.000052
Eğitime Başlarken Elde Edilen Hata	2.0107
Eğitim Sonunda Elde Edilen Hata	0.2873

Sistem eğitilmeye başladığında hata oranı 2.0107 iken 7000 iterasyon sonunda hata 0.283'e düşmüştür. Şekil 5.16'da GY algoritması ile eğitilen YSA'nın eğitim hata oranı görülmektedir.



Şekil 5.16 GY algoritmalı YSA'nın eğitim hata oranı

Denklem 5.2'de matematiksel modeli verilen Sistem-1 kullanılarak geriye yayılım algoritması ile eğitilen sinir ağının model çıkışı ve sistem çıkışı Şekil 5.17'de görülmektedir. Model sitem çıkışını düzgün olarak takip edememektedir.



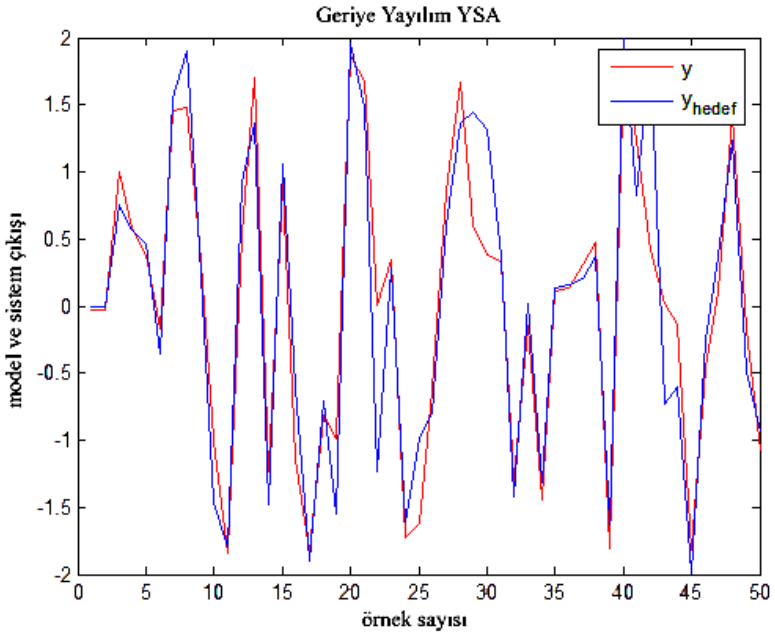
Şekil 5.17 Modelin çıkış değeri ile sisteme ait çıkış değeri

Sistem-1 aynı referans girişle hem GA ile hem de GY algoritması ile eğitime tabi tutuldu. Her iki modelleme türünde de 5 gizli hücreli, tam bağlantılı YSA yapısı ile eğitime başlandı. Modelleme sonunda elde edilen parametreler Tablo 5.8'de görülmektedir.

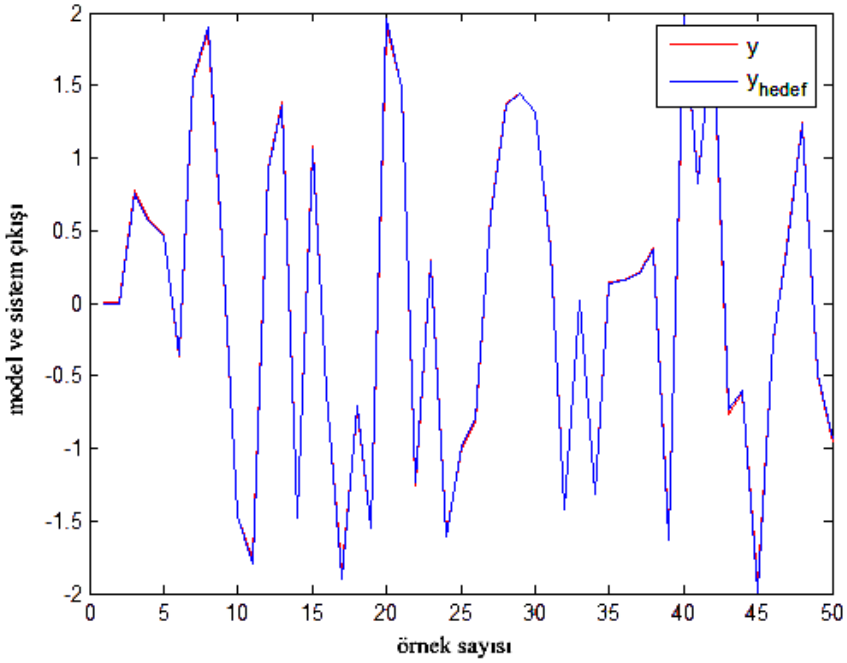
Tablo 5.8 GA ve GY ile modellenen sitemin parametreleri

	Genetik Algoritma	**Geriye Yayılım Algoritması**
Model Yapısı	Narx	Narx
Bağlantı Sayısı	7	26
Eğitim Sonundaki Hata Oranı	0.0168	0.2873
İterasyon Sayısı	7000	7000

Şekil 5.18'de aynı matematiksel denklem ve referans girişlerle modellenen geriye yayımlı YSA ile genetik algoritmalı YSA'nın sistem ve model çıkışı görülmektedir. Şekil 5.18 (a)'da GY algoritmalı YSA modelini, Şekil 5.18 (b)'de GA'lı YSA modeli göstermektedir. GY algoritmasının yerel minumumlara yakalanmış, böylece optimum sonuca ulaşamamıştır. Bu sonuca göre GA'nın sistem modellemede GY algoritmasına göre daha optimum sonuç verdiği görülür.



(a)



(b)

Şekil 5.18. Modelin çıkış değeri ile sisteme ait çıkış değeri
(a) Geriye yayılım algoritması ile eğitilen sinir ağı
(b) Genetik algoritma ile eğitilen sinir ağı

5.2.2. Durum Denklemi ile Tanımlanan Sistemlerin GYSA ile Modellenmesi

Denklem 5.3'te durum denklemleri ile tanımlanan sistemin matematiksel modeli verilmiştir. Bu sistem doğrusal olmayan durum fark denklemleriyle tanımlanan, bilinmeyen dinamikleri olan bir tek giriş ve tek çıkış sistemidir [36]. Bu tez çalışmasında Denklem 5.3'te matematiksel modeli verilen sistem, GYSA ile modellenecek ve Sistem-2 olarak ifade edilecektir.

$$
\begin{aligned}
x_1(k+1) &= x_2(k) \\
x_2(k+1) &= x_3(k) \\
x_3(k+1) &= g[x_1(k), x_2(k), x_3(k), x_4(k), u(k)] \\
x_4(k+1) &= u(k) \\
y(k) &= x_3(k)
\end{aligned}
\tag{5.3}
$$

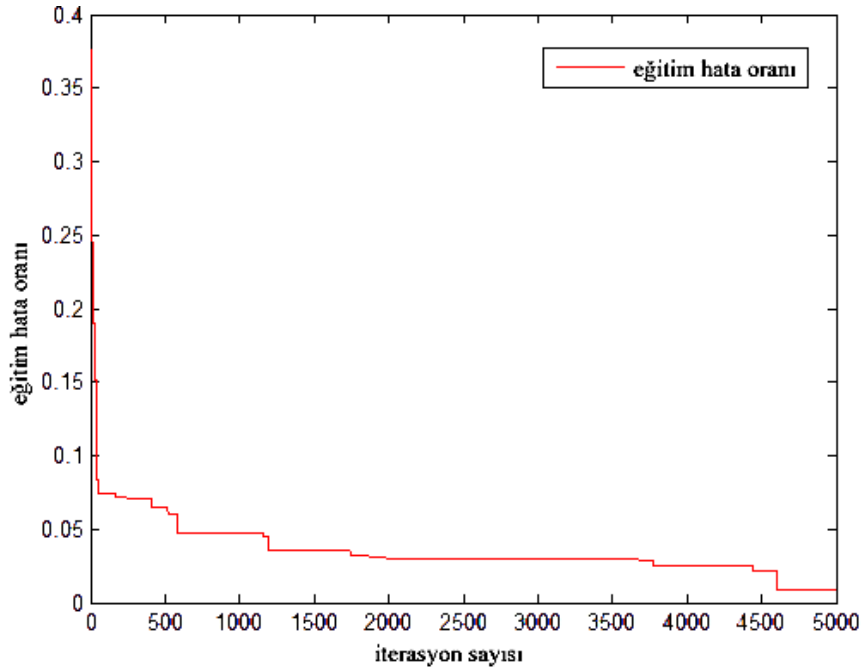
$$
g(x_1, x_2, x_3, x_4, u) = \frac{x_1 x_2 x_3 x_4 (x_3 - 1) + u}{1 + x_1^2 + x_2^2 + x_3^2}
$$

Denklem 5.3'te matematiksel modeli verilen Sistem-2'nin GYSA modeli ve eğitim parametreleri Tablo 5.9'da gösterilmektedir.

Tablo 5.9. GYSA model ve eğitim parametreleri

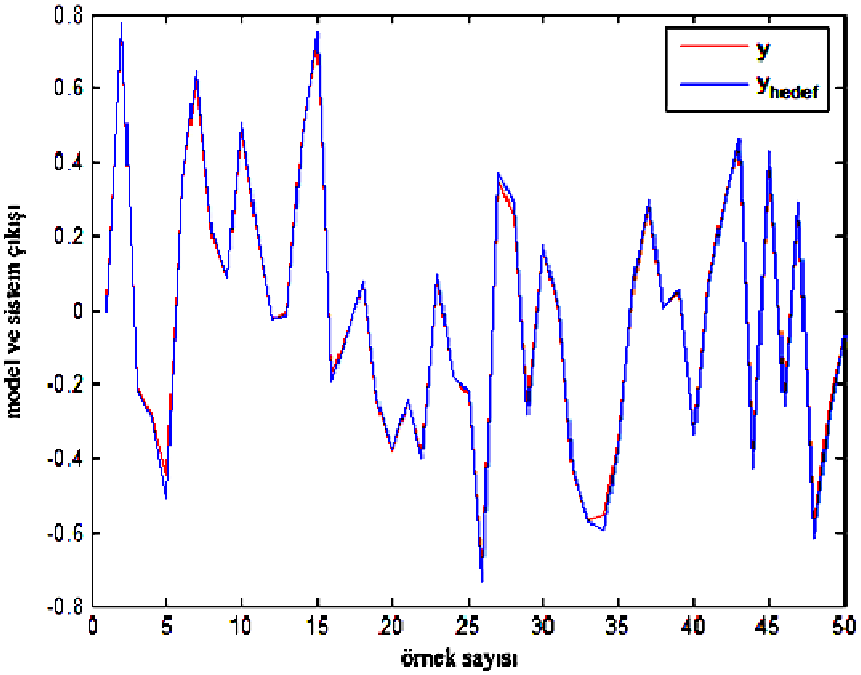
Model Yapısı	Narx
Orta Katmandaki Hücre Sayısı	5
Kullanılan Aktivasyon Çeşidi	Hiperbolik Tanjant Sigmoid
Sisteme Verilen Referans Girişi	[-1 1] Aralığında rastgele atanan değerler.
Başlangıç Popülasyonu N_{ipop}	50
Mutasyon Oranı	0.01
Çaprazlama Oranı	0.5
İterasyon Sayısı	5000
Eğitime Başlarken Elde Edilen Hata	0.3875
Eğitim Sonunda Elde Edilen Hata	0.0095
Eğitime Başlarken Bağlantı Sayısı	36
Eğitim Sonunda Bağlantı Sayısı	13

Sistem eğitilmeye başlandığında hata değeri 0.3875 iken 5000 iterasyon sonunda hata 0.0095'e düştü. Şekil 5.19'da eğitim esnasındaki hata oranının durumu gösterilmektedir.



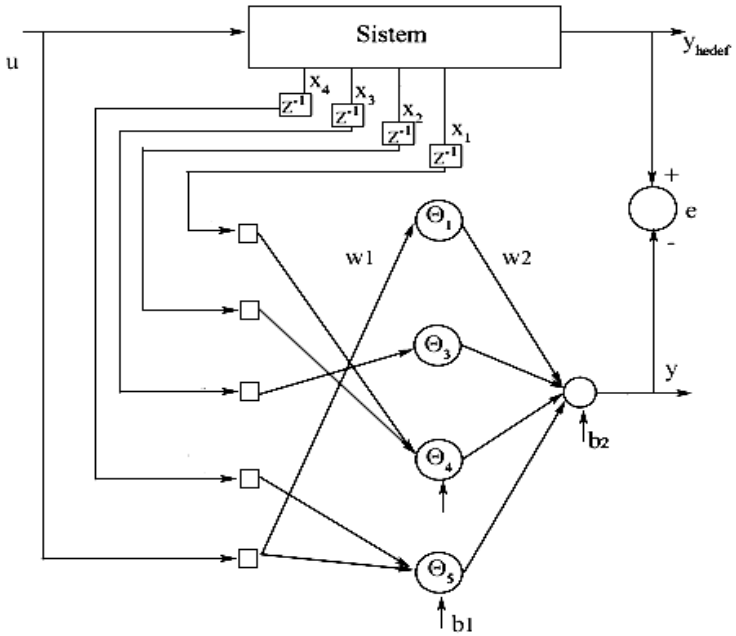
Şekil 5.19. GA kullanılarak eğitilen sinir ağının eğitim hata oranı

Eğitim sonunda istenilen değerler ile GYSA modelinin çıkış değerleri Şekil 5.20'de aynı şekil üzerine çizdirilerek modelin başarımı karşılaştırılmıştır. Buna göre GYSA modelinin sistemi başarılıyla temsil ettiği görülmektedir.



Şekil 5.20. Model çıkış değeri ile sisteme ait çıkış değeri

Sistem doğrusal olmayan durum uzay modelidir. x'ler durum değişkenleridir. Sistem modeline uygun olarak YSA'da durum değişkenleri giriş olarak verildi. Eğitime başlarken, orta katmanda 5 gizli hücreli, tam bağlantılı yapay sinir ağı kullanıldı. Eğitim sonunda elde edilen sinir ağı Şekil 5.21'deki gibi gizli katmanda 4 hücreye sahip ve bağlantı sayısı 13'tür.



Şekil 5.21. 5 gizli hücreyle eğitime başlanan Sistem-2'nin, eğitim sonrası sinir ağı

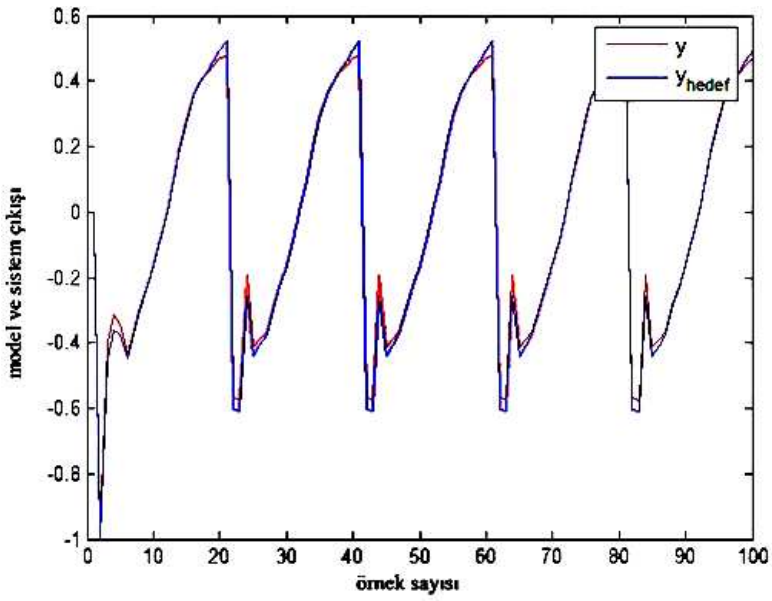
Eğitim sonunda elde edilen ağırlık ve polarma değerleri Tablo 5.10'da gösterilmektedir.

Tablo 5.10. 5 gizli hücreli GYSA modelin ağırlık ve polarma değerleri

w_1					b_1
-	-	-	-	-0.0102	-
-	-	-	1.0705	-	0.0045
-	-	0.4353	-	-	-
-0.0150	-0.4219	-	-	-	0.0684
-	-	-	-0.6656	0.0939	-0.0821

w_2					b_2
0.4151	-	2.6873	0.0102	0.2287	0.0185

GYSA'nın model parametrelerini kullanarak, ağı test etmek amacıyla üçgen dalga referans sinyal sisteme verildi. Eğitimi tamamlanan sinir ağının ve Sistem-2'nin, üçgen dalga giriş sinyaline göre genellemesi Şekil 5.22'de gösterilmektedir. Buna göre eğitilen sinir ağının ezberleme yapmadığı ve öğrenmeyi gerçekleştirdiği görülmektedir.



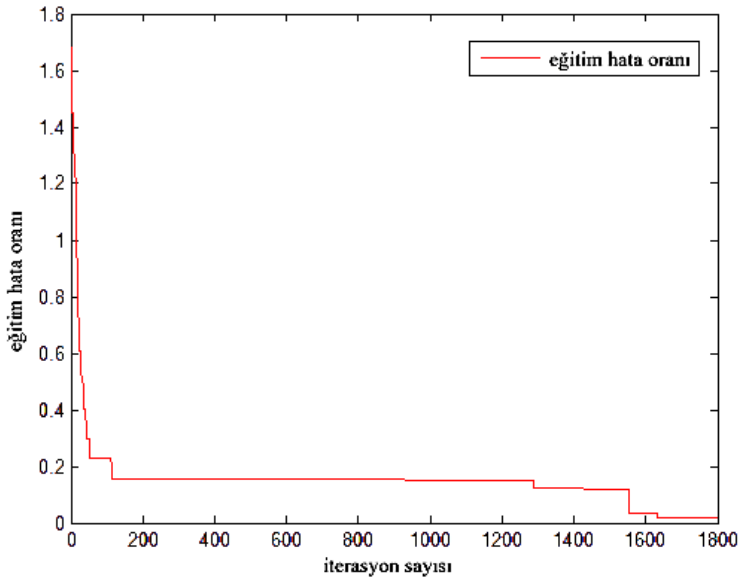
Şekil 5.22. Eğitilen sinir ağına, üçgen dalga giriş verilerek ağın test edilmesi

Denklem 5.3'te matematiksel modeli verilen Sistem-2'nin GYSA modeli ve eğitim parametreleri Tablo 5.11'de gösterilmektedir.

Tablo 5.11. GYSA model ve eğitim parametreleri

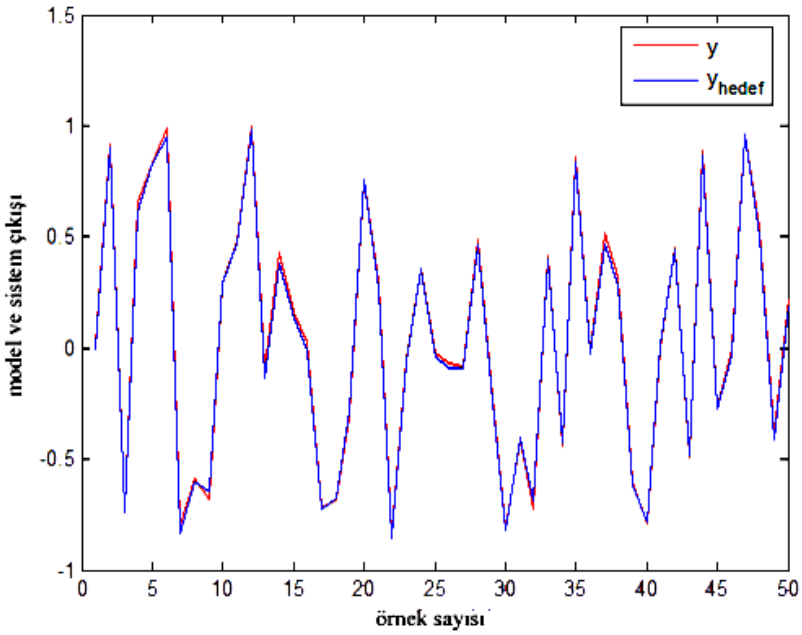
Model Yapısı	Narx
Orta Katmandaki Hücre Sayısı	10
Kullanılan Aktivasyon Çeşidi	Hiperbolik Tanjant Sigmoid
Sisteme Verilen Referans Girişi	[-1 1] aralığında rastgele atanan değerler.
Başlangıç Popülasyonu N_{ipop}	50
Mutasyon Oranı	0.005
Çaprazlama Oranı	0.5
İterasyon Sayısı	1800
Eğitime Başlarken Elde Edilen Hata	1.6818
Eğitim Sonunda Elde Edilen Hata	0.0192
Eğitime Başlarken Bağlantı Sayısı	71
Eğitim Sonunda Bağlantı Sayısı	24

Sistem eğitilmeye başlandığında hata değeri 1.6818 iken 1800 iterasyon sonunda hata 0.0192'e düştü. Şekil 5.23'de eğitim esnasındaki hata oranının durumu gösterilmektedir.



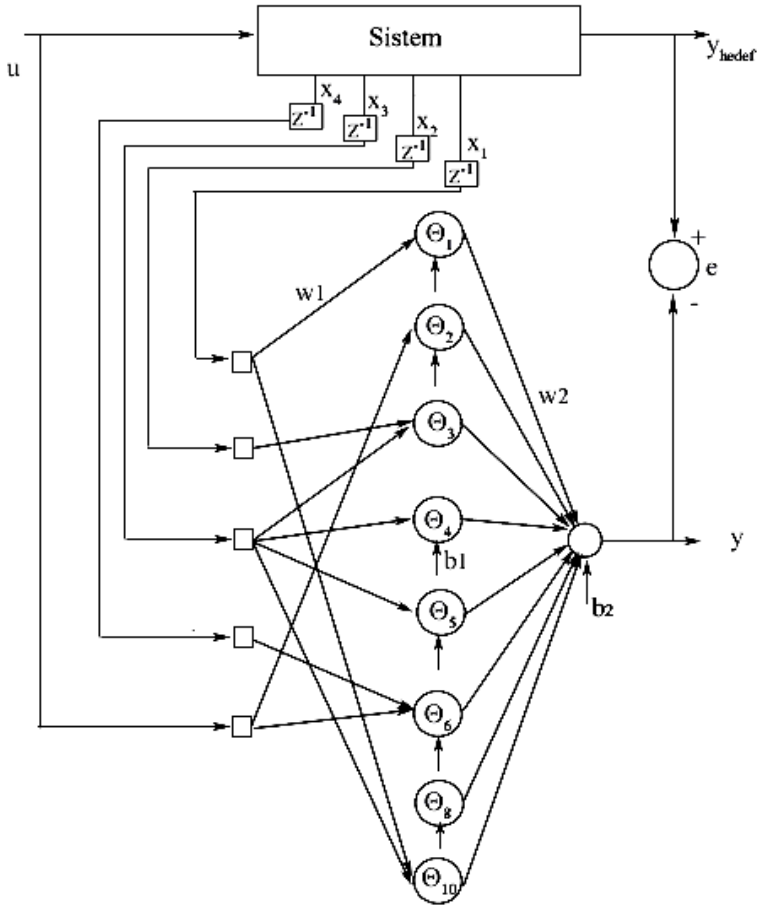
Şekil 5.23. GA kullanılarak eğitilen sinir ağının eğitim hata oranı

Eğitim sonunda istenilen değerler ile GYSA modelinin çıkış değerleri Şekil 5.24'te aynı şekil üzerine çizdirilerek modelin başarımı karşılaştırılmıştır. Buna göre modelin başarılı bir şekilde sistemi temsil ettiği görülmektedir.



Şekil 5.24. Modelin çıkış değeri ile sisteme ait çıkış değeri

Eğitime başlarken, orta katmanda 10 gizli hücreli, tam bağlantılı yapay sinir ağı kullanıldı. Eğitim sonunda elde edilen sinir ağı Şekil 5.25'deki gibi gizli katmanda 8 hücreye sahip ve bağlantı sayısı 24'tür.



Şekil 5.25. 10 gizli hücreyle eğitime başlanan Sistem-2'nin, eğitim sonrası ağ yapısı

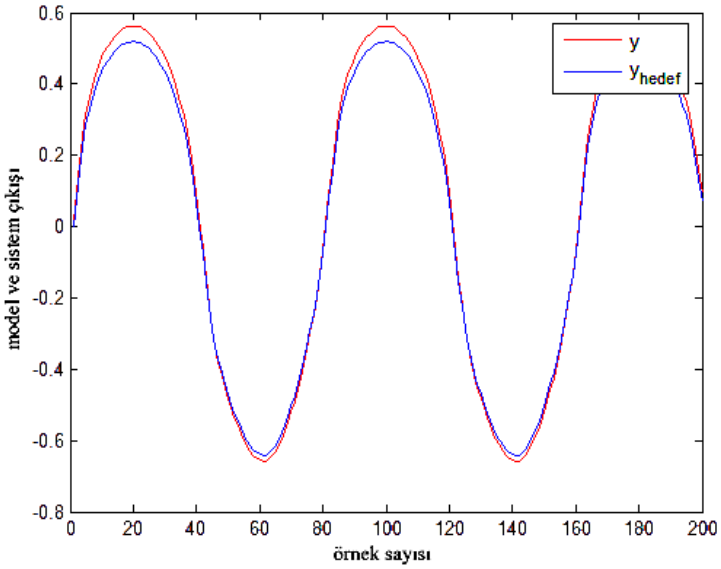
Eğitim sonunda elde edilen ağırlık ve polarma değerleri Tablo 5.12'de gösterilmektedir

Tablo 5.12. 10 gizli hücreli GYSA modelin ağırlık ve polarma değerleri

w_1					b_1
0.0734	-	-	-	-	0.0354
-	-	-	-	0.3796	0.0004
-	0.0194	0.3077	-	-	-
-	-	0.3542	-	-	0.0514
-	-	0.3153	-	-	-0.0900
-	-	-	0.0301	0.2818	1.5367
-	-	-	-	-	-
-	-	-	-	-	0.5870
-	0.2958	-	0.0137	-	-
0.3542	-	-0.1079	-	1.0910	

w_2										b_2
0.3834	0.0412	0.6004	2.3275	0.1092	-0.4063	0.3056	0.0999	-	-0.0020	0.2094

GYSA'nın model parametrelerini kullanarak, ağı test etmek amacıyla sinüzoidal referans sinyal sisteme verildi. Eğitimi tamamlanan sinir ağının ve Sistem-2'nin, sinüzoidal dalga giriş sinyaline göre genellemesi Şekil 5.26'da gösterilmektedir. Buna göre eğitilen sinir ağının ezberleme yapmadığı ve öğrenmeyi gerçekleştirdiği görülmektedir.



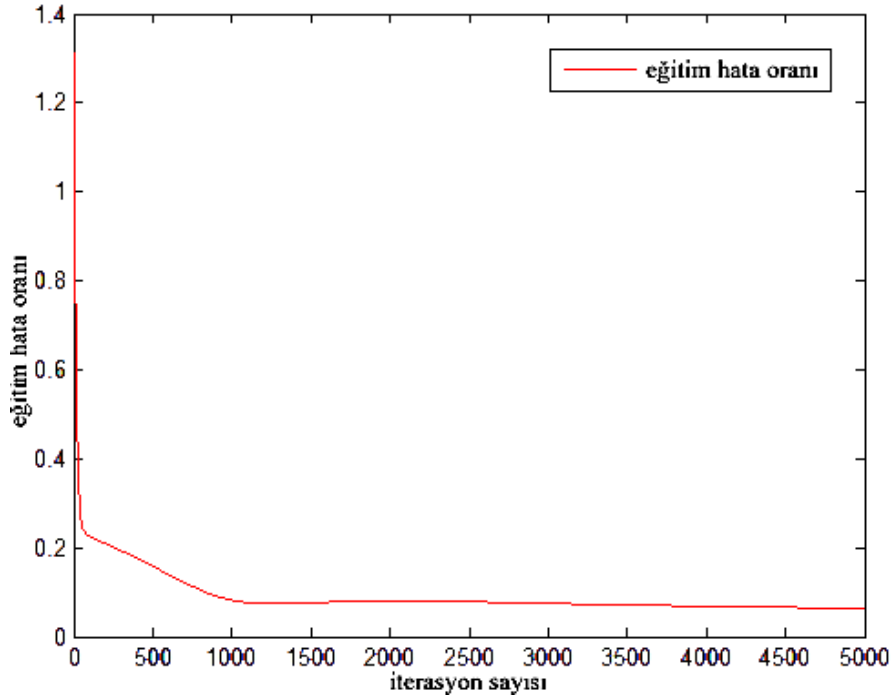
Şekil 5.26. Eğitilen sinir ağına, sinüzoidal referans giriş verilerek ağın test edilmesi

Denklem 5.3'te verilen sistemin Geriye Yayılım YSA modeli ve eğitim parametreleri Tablo 5.13'te gösterilmektedir.

Tablo 5.13. GYSA model ve eğitim parametreleri

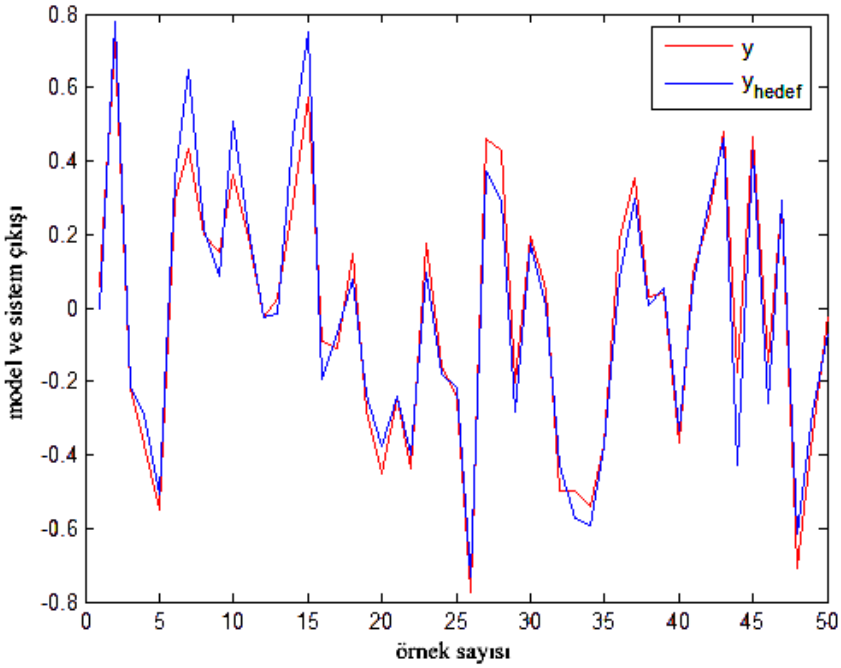
Model Yapısı	Narx
Orta Katmandaki Hücre Sayısı	5
Kullanılan Aktivasyon Çeşidi	Hiperbolik Tanjant Sigmoid
Sisteme Verilen Referans Girişi	[-1 1] aralığında rastgele atanan değerler.
İterasyon Sayısı	7000
Öğrenme Oranı	0.000052
Eğitime Başlarken Elde Edilen Hata	2.2110
Eğitim Sonunda Elde Edilen Hata	0.3273
Eğitime Kullanılan Bağlantı Yapısı	Tam bağlantılı YSA

Sistem eğitilmeye başladığında hata oranı 2.2110 iken 7000 iterasyon sonunda hata 0.3273'e düşmüştür. Şekil 5.27'de eğitim hata oranı görülmektedir.



Şekil 5.27 GY algoritmalı YSA'nın eğitim hata oranı

Sistem-2 kullanılarak geriye yayılım algoritması ile eğitilen sinir ağının model çıkışı ve sistem çıkışı Şekil 5.28'de görülmektedir. Modelin sitem çıkışını tam olarak takip edemediği görülmektedir.



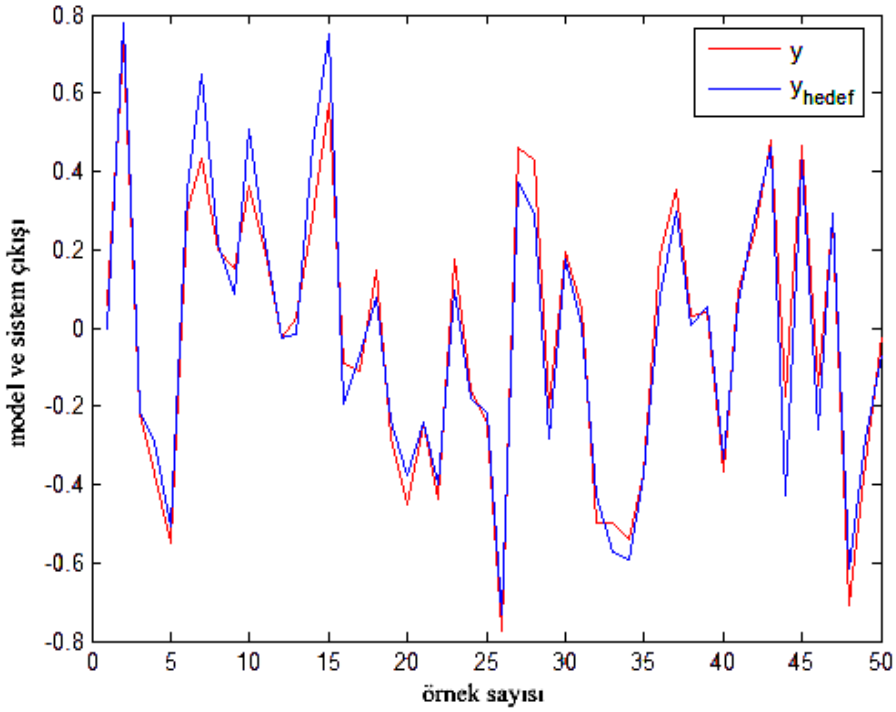
Şekil 5.28 Modelin çıkış değeri ile sisteme ait çıkış değeri

Denklem 5.3'te matematiksel modeli verilen Sistem-2, aynı referans girişle hem GA ile hem de GY algoritması ile eğitime tabi tutuldu. Her iki modelleme türünde de 5 gizli hücreli, tam bağlantılı YSA yapısı ile eğitime başlandı. Modelleme sonunda elde edilen parametreler Tablo 5.14'te görülmektedir.

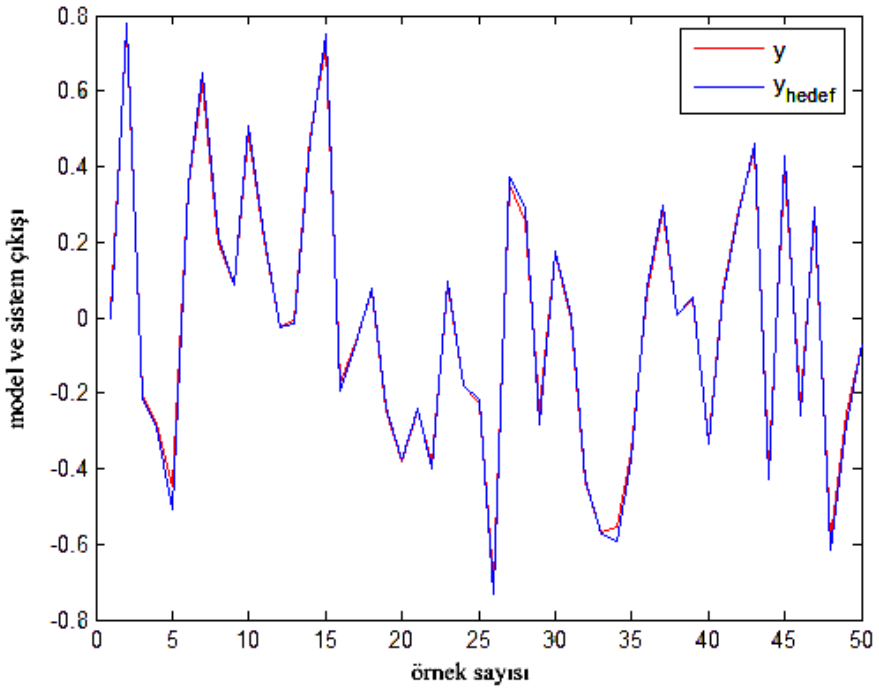
Tablo 5.14 GA ve GY ile modellenen sitemin parametreleri

	Genetik Algoritma	**Geriye Yayılım Algoritması**
Model Yapısı	Narx	Narx
Bağlantı Sayısı	13	26
Eğitim Sonundaki Hata Oranı	0.0095	0.1321
İterasyon Sayısı	5000	5000

Şekil 5.29'da aynı matematiksel denklem ve referans girişlerle modellenen geriye yayımlı YSA ile genetik algoritmalı YSA'nın sistem ve model çıkışı görülmektedir. Şekil 5.29 (a)'da GY algoritmalı YSA modelini, Şekil 5.29 (b)'de GA'lı YSA modeli göstermektedir. GY algoritmasının yerel minumumlara yakalanmış, böylece optimum sonuca ulaşamamıştır. Bu sonuca göre GA'nın sistem modellemede GY algoritmasına göre daha optimum sonuç verdiği görülür.



(a)



(b)

Şekil 5.29. Modelin çıkış değeri ile sisteme ait çıkış değeri
(a) Geriye yayılım algoritması ile eğitilen sinir ağı
(b) Genetik algoritma ile eğitilen sinir ağı

6. SONUÇ

Bu tez çalışmasında, yapay sinir ağlarının yapısının ve parametrelerinin genetik algoritma (GA) ile ayarlanması incelenmiştir. Bu amaçla, ileri beslemeli YSA'nın optimizasyonu için kullanılan GA da gerçek kodlama tekniği kullanıldı. Her bir kromozom YSA'nın ağırlıklarını, polarmalarını ve bu ağırlık ve polarmaların bağlantı anahtarlarını ihtiva eden bir vektör olarak kodlandı. Fark denklemleri ve durum denklemleri ile tanımlanan sistemler, GYSA ile modellendi. Eğitime tam bağlantılı ileri beslemeli ağ yapısıyla başlandı. GYSA ile parametreleri belirlenen modelin girişine, sisteme uygulanmış olan giriş işareti uygulanıp, modelden alınan çıkış işareti ile sistemin gerçek çıkışı arasındaki fark bulundu. Sinir ağının öğrenmesi, ağın sonuç değeri ile hedeflenen sonuçlar arasındaki (farkın) hata ölçüsünün minimum yapılması esasına dayandırıldı. Bu işlem gerçekleştirilirken ağ yapısı da minimum düzeyde tutulmaya çalışıldı. Etkisiz ağırlıklar ve hücreler elimine edilerek optimum ağ yapısı ve optimum parametre değerleri elde edildi.

Sistem modellemede GA ile GY algoritmasının başarısını kıyaslamak için aynı fark denklemine ve referans giriş sinyaline sahip sistem, her iki algoritma ile eğitime tabi tutuldu. Eğitim sonunda GA ile modellenen sistemin etkisiz ağırlıları ve hücreleri elime ettiği, modelden alınan çıkış işareti ile sistemin gerçek çıkışı arasındaki farkın daha düşük olduğu görüldü. GY algoritmasında ise ağ yapısında değişiklik olmadığı ve model çıkışı ile sistem çıkışı arasındaki farkın GA'yla modellemeye göre daha fazla olduğu görüldü.

Bu tez çalışmasıyla doğrusal olmayan dinamik sistemlerin modellenmesinden elde edilen sonuçlar, optimize edilmiş GYSA'nın iyi bir modelleme performansına sahip olduğunu göstermiştir. GA'nın YSA'yı optimize etmede ve sistem modellemede GY algoritmasına göre daha başarılı olduğu görülmüştür.

KAYNAKLAR

[1] **Elmas Ç.,** 2003. Yapay Sinir Ağları (Kuantum, Mimari, Eğitim, Uygulama). Seçkin Yayıncılık, Ankara.

[2] **Karadeniz M., Yüncü S., ve Aydemir M. T.,** 2001. Asenkron motorlarda stator direncinin yapay sinir ağları ile tahmini. Kocaeli.

[3] **Kalınlı A.,** 2003. Elman ağının benzetilmiş tavlama algoritması kullanarak, eğitilmesi,*Erciyes Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, **19** (1-2), 28-37.

[4] **Narendra, K.S., ve Parthasarathy, K.,** 1990. Identification and control of dynamical systems using neural networks, *IEEE Transactions on Neural Networks*, **1** (1), 4-27.

[5] **Tang, C.Z., ve Kwan, H.K.,** 1992. Convergence and generalization properties of multilayer feedforward neural networks, in proc. *IEEE Int. Symp. Circuits Syst.*, San Diego, CA, **1**, 65-68.

[6] **Li L., ve Niu B.,** 2008. A hybrid evolutionary system for designing artifical neural networks, *International Conference on Computer Science and Software Engineering*, Shenzhen, China, 859-862.

[7] **Eğrioğlu E., ve Aladağ Ç. H.,** 2005. Yapay sinir ağları ve arıma modellerinin melez yaklaşımı ile zaman serilerinde öngörü, *7. Ulusal Ekonometri ve İstatistik Sempozyumu*, İstanbul Üniversitesi, İstanbul, 26 - 27 Mayıs.

[8] **Srinivas M., ve Patnaik, L. M.,** 1991. Learning neural network weights using genetic algorithms-improvingperformance by search-space reduction, *Neural Networks, 1991 IEEE International Joint Conference*, Singapore, 18-21 Nov, **3**, p. 2331-2336.

[9] **Hägg J.,** 2007. Artificial neural network with genetic algorithm learning.

[10] **Karaboğa N., ve Çetinkaya B.,** 2004. Performance comparison of genetic algorithm based design methods of digital filters with optimal magnitude response and minimum phase, *IEEE*, p. 644- 647.

[11] **Koç, M. L. Balas, C. E.,** 2004. Taş dolgu dalgakıranların genetik algoritma ile güvenirlik analizi, *17. Teknik Kongre ve Sergisi*, Yıldız Teknik Üniversitesi, İstanbul, 15-17 Nisan.

[12] Yapay Sinir Ağları Ders Notları, http://www.bilinen.net/program-anlatimlari/yapay-sinir-aglari-3804/

[13] Saraç T., 2004. Yapay Sinir Ağları, *Seminer Projesi*, Gazi Üniversitesi Endüstri Mühendisliği Bölümü, Ankara.

[14] Gökbulut M., Yapay Sinir Ağları Ders Notları.

[15] Yurtoğlu, H., 2005. Yapay sinir ağları metodolojisi ile öngörü modellemesi: Bazı makroekonomik değişkenler için Türkiye örneği, *Uzmanlık Tezi*, Ekonomik Modeller ve Atratejik Araştırmalar Genel Müdürlüğü, Ankara.

[16] Çavuşoğlu, M. A., FPGA ile yapay sinir ağı eğitiminin donanımsal gerçeklenmesi. http://www.alicavuslu.com/images/tez.pdf

[17] Çetin M., Uğur A., ve Bayzan Ş., 2006. ileri beslemeli yapay sinir ağlarında backpropagation (geriye yayılım) algoritmasının sezgisel yaklaşımı, *Akademik Bilişim 2006 Konferansı*, Pamukkale Üniversitesi, Denizli, 9-11 Şubat.

[18] Koehn P., 1994. Combining genetic algorithms and neural networks: the encoding problem, *Yüksek Lisans Tezi*, The University of Tennessee, Knoxville.

[19] Çunkaş M., 2006. Genetik Algoritmalar ve Uygulamaları Ders Notları, Selçuk Üniversitesi, Teknik Eğitim Fakültesi, Elektronik-Bilgisayar Eğitimi, Konya.

[20] Çontar E., 2007. Mikrofon dizilerinde ses kaynağının yerinin genetik algoritma kullanılarak bulunması, *Yüksek Lisans Tezi*, Gazi Üniversitesi, Fen Bilimleri Enstitüsü, Ankara.

[21] Paksoy S., 2007. Genetik Algoritma ile Proje Çizelgeleme, *Doktora Tezi*, Çukurova Üniversitesi, Sosyal Bilimler Enstitüsü, Adana.

[22] Avcı E., İleri Yapay Zeka Ders Notları.

[23] Haupt R. L., ve Haupt S. E., 2004. Practical Genetic Algorithms. John Wiley & Sons, Inc., Hoboken, New Jersey, Canada.

[24] Özbilen A., Genetik Algoritma Roting Seminer. http://www.ozbilen.net/makaleler/GA_Roting_seminer_i2.pdf

[25] Kocamaz M., ve Çiçekli U. G., 2010. Paralel makinelerin genetik algoritma ile çizelgelenmesinde mutasyon oranının etkinliği, *Ege Akademik Bakış Dergisi*, **10** (1), 199-210.

[26] Yüksel A., ve Korürek M., 2008. EMG işaretlerinin genetik algoritmalar ve çok katmanlı yapay sinir ağı ile sınıflandırılması, *ELECO'2008 Elektrik - Elektronik - Bilgisayar Mühendisliği Sempozyumu ve Fuarı*, TMMOB Elektrik Mühendisleri Odası Bursa Şubesi, Bursa, 26 - 30 Kasım.

[27] Egeli B., Yapay Sinir Ağları. http://www.mis.boun.edu.tr/kutlu/iu/yapay4.ppt

[28] Seiffert U., 2001.Multiple layer perceptron training using genetic algorithms. *ESANN'2001 European Symposium on Artificial Neural Networks*, Belgium, 25-27 April.

[29] Bozkurt F., Düzenli G., Bozkurt M. R., ve Yumuşak N., 2008. Genetik algoritma yöntemini kullanarak yarı iletken diyoda ait parametre çıkarımı, *ELECO'2008 Elektrik - Elektronik - Bilgisayar Mühendisliği Sempozyumu ve Fuarı*, TMMOB Elektrik Mühendisleri Odası Bursa Şubesi, Bursa, 26 - 30 Kasım.

[30] Kurnaz M. N., ve Ölmez T., 2008. Artımsal yapay sinir ağları kullanılarak ultrasonik görüntülerin bölütlenmesi, *İTU Dergisi*, **7** (2), 17-28.

[31] Chou J. H., ve Tsai J. T., 2006. Tuning the structure and parameters of a neural network by using hybrid Taguchi-genetic algorithm, *IEEE Transactions on Neural Networks*, **17** (1), 69-80.

[32] Üstün O., ve Yıldız İ., 2009. Geri-yayılmalı öğrenme algoritmasındaki öğrenme parametrelerinin genetik algoritma ile belirlenmesi, *SDU International Technologic Science*, **1** (2), 61-73.

[33] Erkanlı S., Günel T., ve Kurnaz S., 2004. Optimum radar parametrelerinin sürekli genetik algoritma yardımıyla karıştırma ortamında radar menzilinin maksimize edilmesi için belirlenmesi geri-yayılmalı öğrenme algoritmasındaki öğrenme parametrelerinin genetik algoritma ile belirlenmesi, *Havacılık Ve Uzay Teknolojileri Dergisi*, **1** (4), 41-46.

[34] URL-1. http://tr.wikipedia.org/wiki/Yapay_sinir_aglari

[35] Chen C., ve Khalil H. K., 1990. Adaptive control of nonlinear systems using neural networks, *IEEE*, **55**(6), 1299-1317.

[36] Jin K., Nikiforuk P. N., ve Gupta M. M., 1994. Adaptive Control of Discrete-time Nonlinear Systems Using Recurrent Neural Networks. Academic Press, London, UK.

ÖZGEÇMİŞ

1983 yılında Elazığ'da doğdu. İlk, orta ve lise öğrenimini Elazığ'da tamamladıktan sonra 2002 yılında Fırat Üniversitesi Teknik Eğitim Fakültesi Elektronik Bilgisayar Eğitim Bölüm Bilgisayar Öğretmenliği Bölümüne girdi. 2006 yılında bu bölümden mezun oldu. Aynı yıl Elazığ'ın Karakoçan ilçesi Yatılı İlköğretim Bölge Okulu'nda Bilişim Teknolojileri öğretmeni olarak göreve başladı. 2007 yılında Elazığ İsmet Paşa İlköğretim Okulu'na Bilişim Teknolojileri öğretmeni olarak atandı ve hala bu okulda görev yapmaktadır.