T.C.
HALİÇ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ

KARAKTER TANIMA

YÜKSEK LİSANS TEZİ

Hazırlayan
Metin ŞAHİN

Tez Danışmanı
Yrd. Doç. Dr. Murat BEKEN

Nisan, 2007
İSTANBUL

T.C.
HALİÇ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ

KARAKTER TANIMA

YÜKSEK LİSANS TEZİ

Hazırlayan
Metin ŞAHİN

Tez Danışmanı
Yrd. Doç. Dr. Murat BEKEN

Nisan, 2007
İSTANBUL

HALİÇ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRLÜĞÜNE

Bilgisayar Mühendisliği Programı Yüksek Lisans öğrencisi Metin Şahin tarafından hazırlanan **"Yapay Sinir Ağlarını Kullanarak Karakter Tanıma"** adlı bu çalışma jürimizce Yüksek Lisans Tezi olarak Kabul Edilmiştir.

Tez (Savunma) Tarihi : 25.05.2007

(Jüri Üyesinin Ünvanı , Adı , Soyadı ve Kurumu) :	İmzası :
Jüri Üyesi: Prof.Dr.Ali OKATAN	.....
Jüri Üyesi : Prof.Dr.Bekir KARAOĞLU (Elektronik ve Hab.ABD Öğr.Üyesi)	.....
Jüri Üyesi :Yrd.Doç.Dr Murat BEKEN (Danışman)	.....

ÖNSÖZ

Bu tez çalışmasında; yapay sinir ağlarını kullanarak karakter algılama olgusu üzerinde durulmaya çalışılmıştır. Bu konunun oluşum aşaması, her ne kadar yeni olmasa da; yapay sinir ağları konusunu, bir uygulama alanına yönlendirmek olmuştur. Karakter algılamaya yönelik çalışmanın oluşturulması için ilk adım aldığım "Uygulamalı Yapay Sinir Ağları" dersi olmuştur. Bu aşamadan sonra tezimin konusu oluşmaya başlamıştır. Daha sonra karakter algılamaya yönelik ve C#'da hazırladığım programda yazılım aşamasında ki uygulama adımı olmuştur.

Tezin hazırlanışında ki amacım bilimsel çalışmalarda, üzerinde pek de fazla durulmamış bir konuda, çalışma yapmak ve yapılacak başka çalışmalara katkıda bulunmak olmak olmuştur…

Bilimsel gelişmeye ve yeni pozitif düşüncelere hızla ulaşmak dileği ile…

Bütün bu aşamalar için bana çalışma ortamı hazırlayan anneme, tezimin düşünsel olarak oluşmasını sağlayan Sayın Prof. Dr. Ali Okatan'a, tez çalışmamın düzenlenmesi ve belli bir yapıda oluşması için tavsiyelerinden dolayı Sayın Yrd. Doç. Dr. Murat Beken'e ve özellikle C#'da yazdığım programa değerli katkılarından dolayı Sayın Öğr. Gör. Oğuz Karan'a teşekkür ederim.

Metin Şahin

Nisan, 2007

İÇİNDEKİLER

ÜÇÜNCÜ BÖLÜM

ONBİRİNCİ BÖLÜM

TABLOLAR LİSTESİ

ŞEKİLLER LİSTESİ

T.C.
HALİÇ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ
YÜKSEK LİSANS TEZİ

KARAKTER TANIMA

Hazırlayan
Metin ŞAHİN

Tez Danışmanı
Yrd. Doç. Dr. Murat BEKEN

Nisan, 2007

ÖZET

Girişte genel olarak zeka kavramı üstünde durulmuştur. Yapay zeka kavramından yapay sinir ağları konusuna geçilmiştir. Yapay sinir ağlarının kullanım amaçları, avantajları ve dezavantajlarından bahsedilmiştir. Bu aşamada yapay sinir ağları kavramının oluşumunu gerçekleştiren biyolojik sinir ağından söz edilmiştir. Yapay sinir ağlarının kullanıldığı alanlar ve uygulamaları söz konusu olmuştur. Yapay sinir ağlarının tarihsel gelişimi ve yapılan çalışmalar da dikkate alınmıştır. Ayrıca yapay sinir ağları kavramını oluşturan matematiksel fonksiyonlar göz önüne alınmıştır. Adım adım bu fonksiyondan çıkan formüller ile program kodlaması için temel yapı meydana getirilmiştir.

Yapay sinir ağlarını oluşturan işlem elemanları, giriş katmanı, ara katman ve çıkış katmanı ayrıntılı bir şekilde göz önüne alınmıştır. Bu aşamada, ağırlıklardan ve kullanılan transfer fonksiyonlarından bahsedilmiştir.

Karakter tanımaya yönelik olarak geliştirilmiş ve C#'da kodlanan programın akış şeması verilmiştir. Programda kullanılan değişkenler ve matematiksel işlem yapısı da akış şemasında yer almıştır. Akış şemasının ardından yazılmış olan programın detayı alınmıştır. Programın çalışması için gerekli olan bazı değişkenler program içinde

tanımlanmış ve değerleri atanmıştır. Örneğin: ara katmandaki ve çıkış katmanındaki işlem elemanı sayısı gibi. İterasyon sayısı gibi bazı değişkenler ise programın çalışması aşamasında dışarıdan girilmiştir.

Daha önceden Türk Alfabesi'ndeki 29 büyük ve 29 küçük harfe karşılık gelen matris yapısı hazırlanmış ve dosya olarak programın içinde yer almıştır. Bu girişlere karşılık olarak gelen çıkış matrisi de (dosya) programda tanımlanarak yer almıştır. Programda işlem elemanları üstündeki ağırlıklar ilk önce rasgele (random) alınmıştır. İterasyon sayısına kadar giriş ve çıkışlar birlikte değerlendirilip ağırlıklara son hali verilmiştir. Ayrıca programın bu aşamasından sonra belli periyotlarla oluşan hata sayısal olarak gösterilip grafiği de verilmiştir. Tez çalışması boyunca kullanılan bilgisayarda iterasyon sayısı en fazla 100000 değeri almıştır. Giriş olarak verilen harfe ait matris, programda değerlendirilip programın çıktısı olan ilgili büyük yada küçük harf, çıkışta görüntülenmiştir.

Bütün bu işlemler boyunca kullanılan program konsol ortamında çalıştırılmıştır. Genel olarak yapay sinir ağları kavramı tekrar incelenip son aşamada karakter tanımanın hangi amaçlar için kullanılabileceği ve programın daha hızlı çalışabilmesi için bazı önerilerde bulunulmuştur.

Anahtar Kelimeler: Delta, Karakter, Matris, Dizi, Yapay.

T.C.
HALİÇ UNIVERSITY
INSTITUE OF TECHNOLOGY SCIENCES
FACULTY OF ENGINEERING
COMPUTER ENGINEERING
THESIS OF MASTER'S DEGREE

CHARACTER RECOGNITION

AUTHOR
Metin ŞAHİN

ADVISER OF THESIS
Ass. Prof. Murat BEKEN

April, 2007

SUMMARY

The entry of this thesis contains information about of intelligence. Then it explains artificial neural networks. It gives information of purposes of use , advantage and disadvantage.

At this stage it explains concept of about biological neurals. Fields of using and pratices of neural Networks are in question. It explains development of historical and operations of artificial neural networks. And then contains mathematical functions. Step by step basic informations for computer program comes into existence with formulas.

It contains constituent of procedure , input section , hidden section and exit section and it gives information about transfer functions and weights. And then it consists flowchart of computer program. This computer program wrote at C# programming language and wrote for character recognition equipment. Flowchart contains variable values and mathematical operations. After flowchart it consists code of computer program.

There are some values for running computer program in this section and some values are in the computer program. For example : counts of operation constituent of

input and output sections… And some values gives to computer program from outside during running. For example : count of step.

Computer program contains structure matrix of Turkish alphabet (29 capital letter and 29 lower case). It contains exit character matrix.

The weights take shape as random. And they get last condition after count of step.

And then it consists values of error and graphic of error. The maximum count of step is 5000 at this computer program. During running computer program entered 5000 fixed number.

Computer program takes inputs after operations it displays capital letter or lowercase. Computer program runs console environment during operations.

How can it use and what does it do? Answers of these questions explain (character recognition). And how can computer program more fast? It explains. The end of this thesis contains some summary informations.

Key Words: Delta, Character, Matrix, Series, Artificial.

GİRİŞ

İnsanlığın var oluşu ile birlikte pozitif bilimler oluşmaya, bir anlam kazanmaya ve gelişmeye başlamıştır. Bu basitçe hazırlanmış bir aletten, günümüzün vazgeçilmezlerinden olan bilgisayarlara kadar yol almıştır.

İnsanlığın gerçek amacı; daha rahat, daha barışçıl ve daha modern bir yaşam olgusuna ulaşmaktır. Bu amaca ulaşabilmek için mantıklı düşünce yapısını kullanmışlardır. Örneğin; doğadaki olayları ve yapıyı bir kural dahilinde açıklayabilmek için matematik, fizik ve kimya gibi bilimleri oluşturup geliştirmişlerdir.

Bu başlangıcın ardından adım adım daha ileri gidilmiştir. Hesaplanması zor olan ya da bir kural dahilinde çözülemeyen olgular, bilgisayar yardımı ile kolaylıkla çözülebilir hale gelmiştir. Elbette ki temel yapı, yine insan beyninin çözüm arama ve bu çözümü nerede gerçekleştirebileceği olgusudur.

Herhangi bir durumu açıklamak için oluşturulmuş karakterler sayısal ortamda daha kolay değerlendirilip açıklanarak bir sonuca bağlanabilir.

Belirli ölçütler dahilinde yazılmış karakterler, herhangi bir görüntü alma cihazı aracılığı ile bilgisayara aktarılabilir. Bilgisayarda daha önceden işlenmiş veriler aracılığı ile bu aktarılmış bilginin son şeklini yani hangi karakter olduğunun belirlenmesini sağlıklı bir şekilde gerçekleştirebilir.

Bunun için bu çalışmanın konusu olan yapay sinir ağlarını kullanarak karakter tanıma üzerinde durulmuştur, çeşitli kaynaklar incelenmiş ve değerlendirilmiştir.

BÖLÜM 1.

1. YAPAY ZEKA VE MAKİNE ÖĞRENMESİNE GENEL BAKIŞ

1. 1. AKIL VE ZEKA

Akıl kelimesi toplumda genellikle insanların zeka düzeyini ifade etmek amacıyla kullanılmaktadır. Sıklıkla akıl kavramı zekayla karıştırılmaktadır. Oysa akıl; düşünme, anlama, kavrama, idrak etme, karar verme ve önlem alma yetenekleridir. Akıl aynı zamanda muhakeme ve bilgi elde etme gücü olarak da tanımlanabilir. Zeka, gerçekleri algılama, yargılama ve sonuç çıkarma yeteneklerinin tamamıdır.

Akıl, genetik yoldan intikal eden sevgi, korku, kıskançlık, doğal savunma güdülerinin yanı sıra bulunduğumuz çevreden aldığımız etkileşimlerden ve toplumun şartlandırmalarından etkilenerek gelişmektedir. Dolayısıyla akıl sabit değil, aksine insan hayatının sonuna kadar artabilen ve gelişebilen bir yetenektir. Akıl, makine, bilgisayar, yazılım veya başka bir yolla taklit edilemez.

Her insan doğuştan belirli bir zekaya sahiptir. Zeka belirli bir konuda çalışılarak, öğretilerek, eğitilerek edinilen bilgi ve birikimlerle, deneyimlere dayalı becerilerle geliştirilebilir. İlk kez karşılaşılan ya da ani olarak gelişen bir olaya uyum sağlayabilme, anlama, öğrenme, analiz yeteneği, beş duyunun, dikkatin ve düşüncenin yoğunlaştırılması zeka ile gerçekleştirilebilmektedir. Zeka yazılım veya tümleşik yongalarla taklit edilebilmektedir. Bu durumda zeka "Yapay Zeka" olarak adlandırılmaktadır. (Elmas, 2003)

1. 2. YAPAY ZEKA

Yapay sinir ağları, yapay zeka biliminin altında araştırmacıların çok yoğun ilgi gösterdikleri bir araştırma alanıdır. Bilgisayarların öğrenmesine yönelik çalışmaları kapsamaktadır.

İnsan beyni dünyanın en karmaşık makinesi olarak kabul edilebilir. Sayısal bir işlemi birkaç dakikada yapabilmesine karşın, idrak etmeye yönelik olayları çok kısa bir sürede yapar. Örneğin yolda giden bir şoför, yolun kayganlık derecesini, önündeki tehlikeden ne kadar uzak olduğunu, sayısal olarak değerlendiremezse dahi geçmişte kazanmış olduğu tecrübeler sayesinde hızını azaltır. Çünkü o saniyelerle ölçülebilecek kadar kısa bir sürede tehlikeyi idrak etmiş ve ona karşı koyma gibi bir tepki vermiştir. Bu noktada akla gelen ilk soru şu olmaktadır. Acaba bir bilgisayar yardımı ile böyle bir zeka üretmek mümkün olabilir mi? Bilgisayarlar çok karmaşık sayısal işlemleri anında çözebilmelerine karşın, idrak etme ve deneyimlerle kazanılmış bilgileri kullanabilme noktasında çok yetersizdirler. Bu olayda insanı ya

da insan beynini üstün kılan temel özellik, sinirsel algılayıcılar aracılığı ile kazanılmış ve göreli olarak sınıflandırılmış bilgileri kullanabilmesidir. Uzman Sistemler (US), Bulanık Mantık (BM), Genetik Algoritma (GA) ve Yapay Sinir Ağları (YSA) gibi yapay zeka alt dalları özellikle son yıllarda, geniş bir araştırma alanı bulmaktadır. (Elmas, 2003)

1. 2. 1. Uzman Sistemler

Bir problemi o problemin uzmanlarının çözdüğü gibi çözebilen bilgisayar programları geliştiren teknolojidir. Uzmanlar problemleri çözerken bilgilerini ve deneyimlerini kullanırlar. Bu bilgi ve deneyimlerin bilgisayar tarafından anlaşılabilir olması ve bilgisayarda saklanması gerekmektedir. Bilgi tabanında saklanan bu bilgileri kullanarak insan karar verme sürecine benzer bir süreç ile problemlere çözümler üretirler. Bir uzman sistemin dört temel elemanı vardır.

1. 2. 1. 1. Bilginin Temin Edilmesi

Uzman sistemin uzmanlık alanı ile ilgili bilgilerin toplanması, derlenmesi ve bilgisayarın anlayacağı şekle dönüştürülmesi çalışmalarını kapsar.

1. 2. 1. 2. Bilgi Tabanı

Uzman sistemin uzmanlık alanı ile ilgili toplanan bilgilerin saklandığı yerdir. Bilgiler genellikle kurallar (EĞER … ise O ZAMAN … şeklinde), bilgi çerçeveleri, bilgi sınıfları ve prosedürlerden oluşur. Bu bilgiler ilgili uzmanlık alanı hakkında uzmanların bildiği ve belirlediği gerçeklere dayanmaktadır.

1. 2. 1. 3. Çıkarım Mekanizması

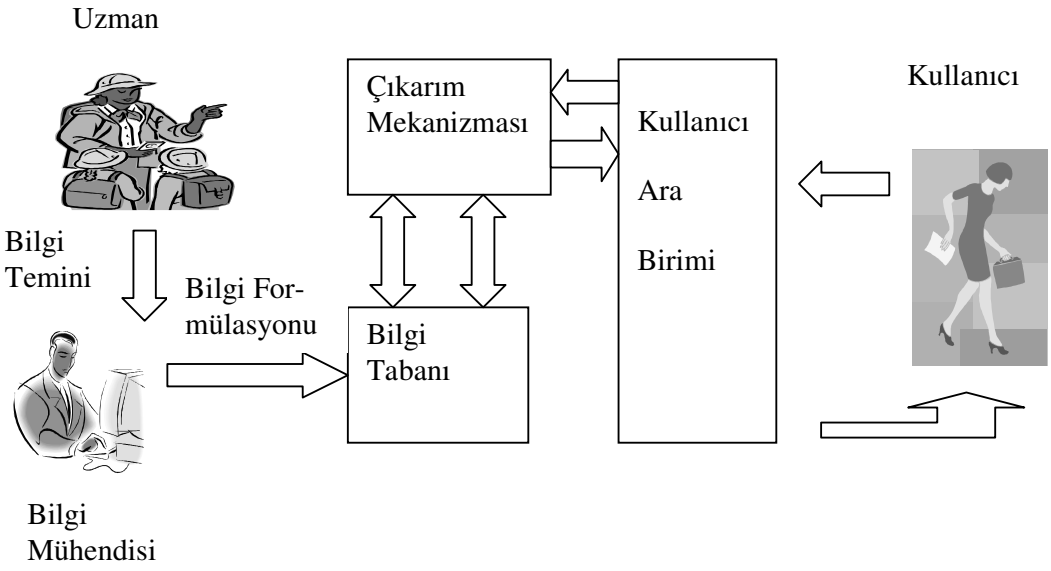
Bilgi tabanında bulunan bilgileri arayan, filtreleyen, yorumlayan ve sonuçlar çıkaran yani, çözüm üreten bir mekanizmadır. Genel olarak iki türlü çıkarım vardır: ileri doğru zincirleme ve geri doğru zincirleme.

1. 2. 1. 4. Kullanıcı Ara Birimi

Uzman sistemi kullanan kişiler ile uzman sistemin iletişimini sağlar. Problemlere üretilen sonuçların nasıl üretildiği ve neden o sonuçlara varıldığını açıklar. Uzman sistemin bir uzman gibi görülmesi, bu ara birimin ve açıklama yeteneğinin güçlü olmasına bağlar. Bir uzman sistemin çalışma ilkesi “Şekil 1. 1.” de gösterildiği gibidir.

Görüldüğü gibi uzman sistem elemanlarından birisi de uzmanın kendisidir. Bir uzman sistemi geliştirmek için en az bir uzmana ihtiyaç vardır. Bilgi mühendisi bilgi elde etme yöntemlerini kullanarak uzmandan bilgileri elde eder. Topladığı bilgileri derleyerek gereksiz bilgileri uzmanın yardımıyla ayıklar ve bilgileri bilgisayarın

anlayacağı biçime getirerek bilgi tabanına koyar. Kullanıcı uzman sisteme bir soru sorduğunda, çıkarım mekanizması bilgi tabanını arayarak sorulan sorunun yanıtını arar.



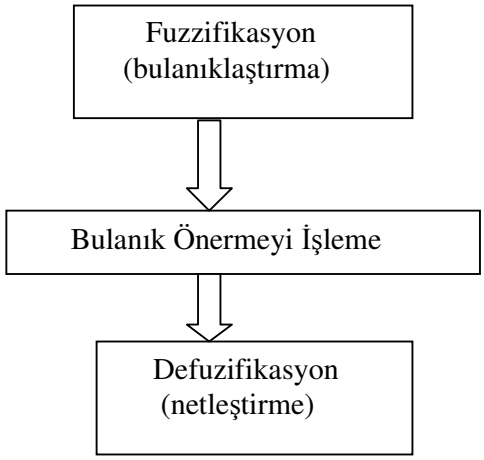
Şekil 1. 1. Uzman Sistemin Elemanları ve Bilgi Akışı

İlgili bilgileri belirleyip probleme çözüm ürettikten sonra kullanıcı arabirimi aracılığı ile kullanıcıya sorusunun yanıtı verilir. Bazı durumlarda problemleri çözecek bilgiler bilgi tabanında bulunmayabilir. Bu durumda bilgi tabanına yeni bilgi eklenmesi veya güncellenmesi olasıdır. Zaman içinde bilgi tabanında yeterli bilgileri toplamak söz konusu olabilir. Bilginin çoğalması ise çıkarım mekanizmasının problemlere çözümler üretme gücünü artırmaktadır. (Öztemel, 2003).

1. 2. 2. Bulanık Mantık

Günümüzdeki bir çok olay belirsiz koşullarda gerçekleşmektedir. Beklenmedik olaylar ortaya çıkmakta ve karar vermeyi etkilemektedir. Seyahate gitmek isteyen bir kişinin yağmur yağması ile planlarını değiştirmesi gibi belirsiz olaylar insanların kararlarını etkilemektedir. Kesin olarak bilinmeyen olaylar hakkında karar vermek

için uzmanlar normal, yüksek, düşük, yaklaşık gibi kavramlar kullanmaktadırlar. Hava sıcaklığının belirsiz olması durumunda sıcaklık 20 derece olunca şu işi yap demek yerine, sıcaklık normal olunca şu işi yap denilmektedir. Bulanık mantık teknolojisi bilgisayarın da bu gibi durumlarda karar verebilmesi için geliştirilmiş teknolojidir. "Normal" gibi kavramların bilgisayar tarafından anlaşılmasını sağlar. Bulanık mantık "Şekil 1. 2. "'de gösterilen sürece göre çalışmaktadır.

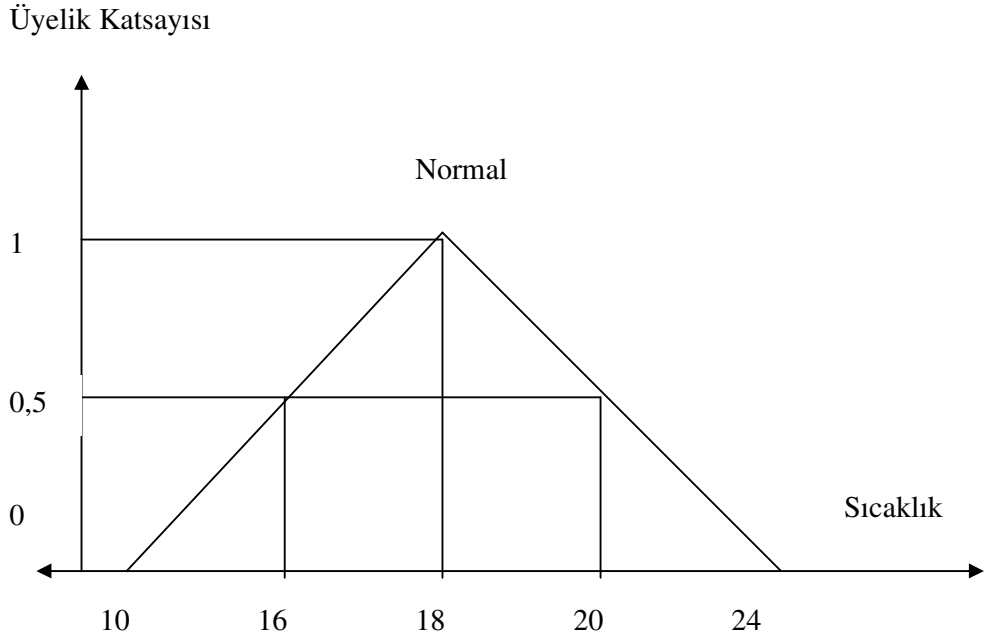


Şekil 1.2. Bulanık Önermeler Mantığının Elemanları ve Çalışması

1. 2. 2. 1. Bulanıklaştırma

Çözülecek problem ile ilgili bulanık önerme değişkenlerinin ve karar verme kurallarının belirlenmesi ve üyelik fonksiyonunun oluşturulması işlemidir. Örneğin, hava sıcaklığının "normal" olması durumunda değişkenin adı "normal" olabilir. Üyelik fonksiyonu ise herhangi bir sıcaklık değerinin "normal" olma üyeliğini gösterir.

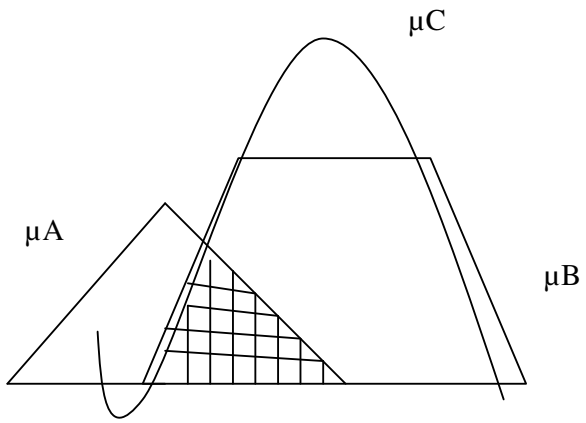
Üyelik değeri 0 – 1 arasında bir değerdir. 1 tam üyelik durumunu, 0 ise üye olmama durumunu gösterir. Üyelik değerinin bir olasılık değeri olmadığını belirtmek gerekmektedir. 18 derece sıcaklığın normal üyelik değeri 1 ise bu normal olma olasılığı % 100' dür demek değildir. Üyelik değerinin olasılık değeri yerine "mümküniyet değeri" olarak görmek gerekir. Normal değişkenin üyelik fonksiyonu "Şekil 1. 3" de gösterilmiştir. Görüldüğü gibi bu üyelik fonksiyonu -∞ ile +∞ arasındaki bütün sıcaklık değerlerinin üyelik katsayılarını bulmak mümkündür.



Şekil 1. 3. Üyelik Fonksiyonu Örneği

1. 2. 2. 2. Bulanık Önerme İşlemi

Belirlenen bulanık önerme değişkenlerinin kurallarını kullanarak problemin çözüm alanını belirleme işlemidir. Genellikle üyelik fonksiyonlarının üst üste konulması sonucu kurallara göre ortak alanın bulunmasıdır. Eğer kurallar (VE) bağlacı ile bağlanmış ise üyelik fonksiyonlarının küçük değeri, (VEYA) bağlacı ile bağlanmış ise o zaman üyelik fonksiyonlarının büyük değerleri alınarak alan oluşturulur. "Şekil 1. 4. " de oluşturulmuş bir çözüm alanı örneğini göstermektedir. Şekilde ki taralı alan da çözüm uzayını göstermektedir.



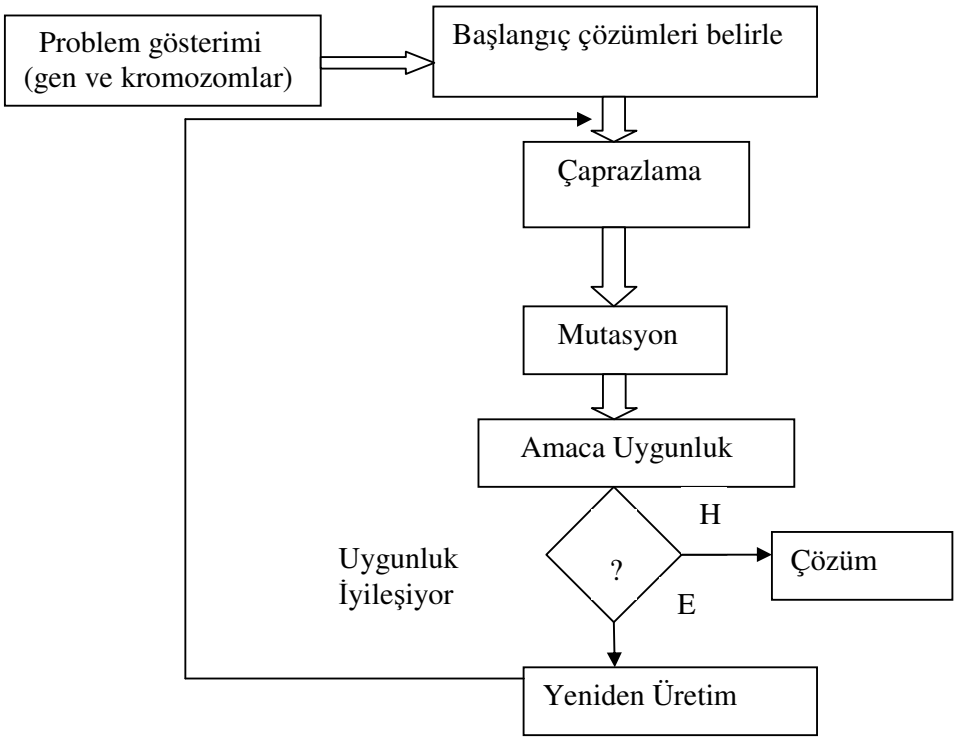
Şekil 1. 4. Bulanık Önermeler Çözüm Uzayı Örneği

1. 2. 2. 3. Netleştirme

Bulunan çözüm alanından tek bir değer elde edilmesi işlemidir. Genellikle üyelik değerinin en yüksek olduğu noktaya karşılık gelen değer problemin çözümü olan tek değerdir. Bu alandan böyle tek bir değer belirlenmemesi durumunda en yüksek değerlerin ortalaması veya oluşan çözüm alanının ağırlık merkezine karşı gelen çözüm değeri olarak alınır. (Öztemel, 2003)

1. 2. 3. Genetik Algoritma

Karmaşık optimizasyon problemlerinin çözülmesinde kullanılan bir teknolojidir. Bir problemi çözebilmek için öncelikle rasgele başlangıç çözümleri belirlenmektedir. Daha sonra bu çözümler birbirleri ile eşleştirilerek performansı yüksek (daha iyi) çözümler üretilmektedir. Bu şekilde sürekli çözümler birleştirilerek yeni çözümler aranmaktadır. Bu arama iyi sonuç üretilinceye kadar devam etmektedir. Genetik algoritmalar ile problemin çözülmesinde arzu edilen sonucu üretecek özelliklerin kalıtım yolu ile başlangıç çözümlerinden elde edilen yeni çözümlere onlardan da daha sonraki çözümlere geçtiği kabul edilmektedir. Bir genetik algoritmanın temel elemanları “Şekil 1. 5. “ de gösterilmiştir.



Şekil 1. 5. Genetik Algoritmanın Elemanları ve Çalışması

Bu elemanlar kısaca şöyle tanımlanmaktadır.

1. 2. 3. 1. Kromozom ve Gen

Genetik algoritmanın çözmesi istenen problemin her bir çözümünü göstermektedir. Bir problem için N adet çözüm olabilir. Genetik algoritmanın bunların arasından en iyisini arayıp bulması istenmektedir. Kromozomlar ise bu çözümleri gösterirler.

Başlangıçta rasgele atanan çözümler daha sonra genetik algoritmanın çalışma ilkesine göre iyileştirilmektedir. Bir kromozomun elemanlarından her birisi çözümün bir özelliğini göstermektedir. Bunlara da gen denilmektedir.

1. 2. 3. 2. Çözüm Havuzu

Problemin en iyi çözümünü aramak için kullanılan ve rasgele belirlenmiş başlangıç çözüm setidir. Probleme göre değişen sayıda kromozom (başlangıç çözümü) belirlenebilir. Buna havuzun büyüklüğü denilmektedir.

1. 2. 3. 3. Çaprazlama

Problem çözüm havuzunda bulunan çözümleri (kromozomları) ikişer ikişer birleştirerek yeni çözümler üretmektedir. İki kromozomdan iki adet yeni kromozom üretilmektedir. Bir problemin çözüm uzayında kaç adet kromozomun çaprazlanacağı çaprazlama oranına göre belirlenmektedir.

1. 2. 3. 4. Mutasyon

Çaprazlama sonucunda farklı çözümlere ulaşmak bazen zor olmaktadır. Yeni çözüm aramanın kolaylaştırılması ve aramanın yönünü değiştirmek amacı ile bir kromozomun bir elemanının (bir genin) değiştirilmesi işlemidir. Bir problem havuzu içinden kaç kromozomun mutasyona uğratılacağına mutasyon oranına göre karar verilmektedir.

1. 2. 3. 5. Uygunluk Fonksiyonu

Belirlenen çözümlerin uygunluk derecelerinin ölçülmesini sağlayan bir fonksiyondur. Her problem için bir uygunluk fonksiyonunun belirlenmesi gerekmektedir.

Bu fonksiyon probleme göre değişmektedir. Örneğin, bir iş çizelgesi (sıralanması) probleminde belirlenen her sıraya göre, işlerin tamamının belirtilmesi zamanı uygunluk ölçütü olarak belirlenebilir. En kısa zamanda işleri bitiren iş sırasının belirlenmesi istenmektedir. İşlerin toplam bitirilme zamanı azaltıldığı sürece daha fazla iş sıralarının aranmasına devam eder. Bu süre daha fazla azaltılamayacağı bir noktada en iyi çözüm bulunmuş olacaktır.

1. 2. 3. 6. Yeniden Üretim

Çözüm havuzundaki kromozomlar çaprazlama ve mutasyon sonucunda üretilen yeni kromozomlar nedeni ile çoğalacaktır. Bunların arasından problem havuz büyüklüğüne göre kromozomlar seçilerek diğerleri atılır. Seçilenler ise bir sonraki nesil çözümü olarak yeniden çaprazlanıp gelecek çözümleri üretirler. Yeni üretim için değişik yöntemler geliştirilmiştir. Bu yöntemlerden bir tanesi, havuz üyeleri rulet mantığı ile seçilmektedir. En iyi çözümlerin bir sonraki havuz içinde seçilme olasılıkları daha fazladır. Kötü çözümler de seçilebilirler. Bunun nedeni, belki bu kötü çözümlerin sonraları iyi çözümlere yol açabileceklerindendir. (Öztemel, 2003)

1. 2. 4. Yapay Sinir Ağları

Bilgisayarların olayları öğrenmesini sağlayan teknolojidir. Genellikle örnekler kullanılarak olayların girdi ve çıktıları arasındaki ilişkileri öğrenilir. Öğrenilen bilgiler ile benzer olaylar yorumlanarak kararlar verilir veya problemler çözülür.

BÖLÜM 2.

2. YAPAY SİNİR AĞLARI

2. 1. YAPAY SİNİR AĞLARININ GENEL TANIMI

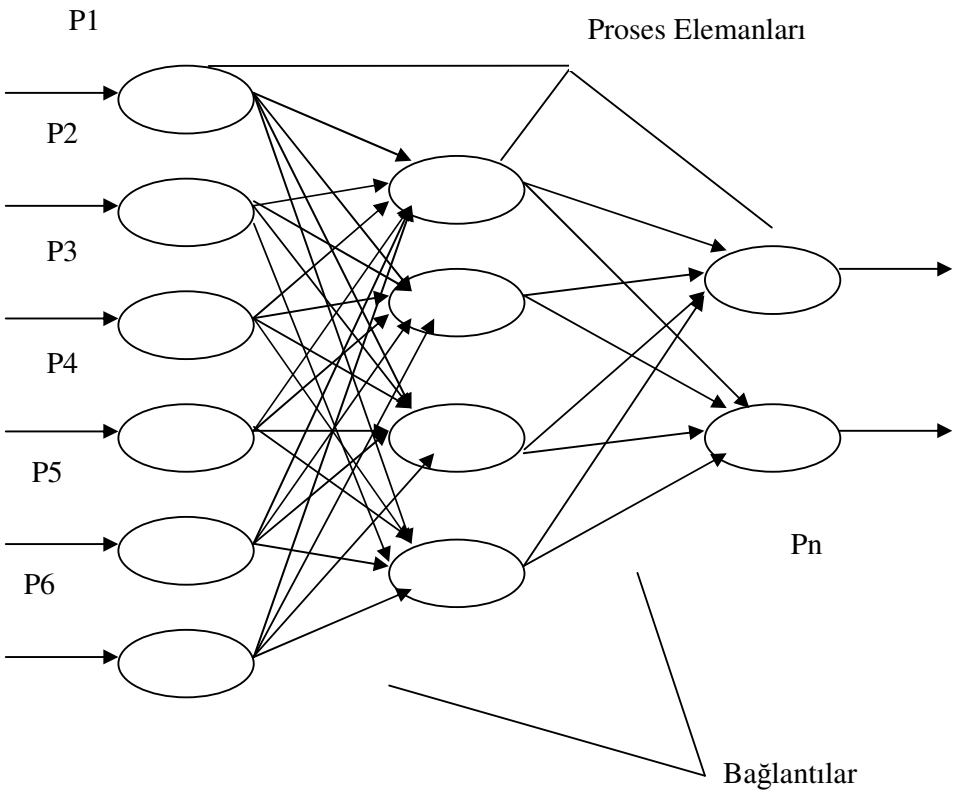
Yapay sinir ağları, insan beyninin özelliklerinden olan öğrenme yolu ile yeni bilgiler türetebilme, yeni bilgiler oluşturabilme ve keşfedebilme gibi yetenekleri herhangi bir yardım almadan otomatik olarak gerçekleştirmek amacı ile geliştirilen bilgisayar sistemleridir. Bu yetenekleri geleneksel programlama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

2. 2. YAPAY SİNİR AĞI TANIMI VE EN TEMEL GÖREVİ

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

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

gibi konularda başarılı bir şekilde uygulanmaktadır. Örneklerden elde ettikleri bilgiler ile kendi deneyimlerini oluşturur ve daha sonra benzer konularda benzer kararları verirler. Yapay sinir ağları günümüzde bir çok probleme çözüm üretebilecek yeteneğe sahiptir. Değişik şekillerde tanımlanmaktadır. Tanımların ortak birkaç noktası vardır. Bunların en başında yapay sinir ağlarının birbirine hiyerarşik olarak bağlı ve paralel olarak çalışabilen yapay hücrelerden oluşmaları gelmektedir. Proses elemanları da denilen bu hücrelerin birbirlerine bağlandıkları ve her bağlantının bir değerinin olduğu kabul edilmektedir. Bilginin öğrenme yolu ile elde edildiği ve proses elemanlarının bağlantı değerlerinde saklandığı dolayısıyla dağıtık bir hafızanın söz konusu olduğu ortak noktaları oluşturmaktadır. Proses elemanlarının birbirleri ile bağlanmaları sonucu oluşan ağa yapay sinir ağı denilmektedir. Bir yapay sinir ağı modeli “Şekil 2. 1. “ de gösterilmektedir.



P1, P2, P3, P4, P5 … Pn : Proses Elemanları

Şekil 2. 1. Bir Yapay Sinir Ağı Örneği

Teknik olarak da, bir yapay sinir ağının en temel görevi, kendisine gösterilen bir girdi setine karşılık gelebilecek bir çıktı seti belirlemektir. Bunu yapabilmesi için ağ, ilgili olayın örnekleri ile eğitilerek (öğrenme) genelleme yapabilecek yeteneğe kavuşturulur. Bu genelleme ile benzer olaylara karşılık gelen çıktı setleri belirlenir.

Ağı oluşturan proses elemanları, bunların bilgileri işleme yetenekleri, birbirleri ile bağlantılarının şekilleri değişik modelleri oluşturmaktadır.

Yapay sinir ağı aynı zamanda, "bağlantılı ağlar", "paralel ağlar", "nuroformik sistemler" olarak da adlandırılmaktadır. Yapay sinir ağları bilgisayar bilimine de bazı yenilikler getirmiştir. Algoritmik olmayan, paralel programlama, dağıtılmış programlama vb. gibi tekniklerin gelişmesine katkıda bulunmuştur. Bilgisayarların da öğrenebileceğini göstermiştir. Özellikle olaylar hakkında bilgilerin olmadığı

fakat örneklerin bulunduğu durumlarda çok etkin olarak kullanılabilecek bir karar verme aracı ve hesaplama yöntemi olarak görülebilir.

2. 3. YAPAY SİNİR AĞLARININ GENEL ÖZELLİKLERİ

Yapay sinir ağlarının karakteristik özellikleri, uygulanan ağ modeline göre değişmektedir.

2. 3. 1. Yapay Sinir Ağları Makine Öğrenmesi Gerçekleştirirler

Yapay sinir ağlarının temel işlevi bilgisayarların öğrenmesini sağlamaktır. Olayları öğrenerek benzer olaylar karşısında benzer kararlar vermeye çalışır.

2. 3. 2. Programları Çalıştırma Stili Bilinen Programlama Yöntemlerine Benzememektedir

Geleneksel programlama ve yapay zeka yöntemlerinin uygulandığı bilgi işleme yöntemlerinden tamamen farklı bir bilgi işleme yöntemi vardır.

2. 3. 3. Bilginin Saklanması

Yapay sinir ağlarında bilgi, ağın bağlantılarının değerleri ile ölçülmekte ve bağlantılarda saklanmaktadır. Diğer programlarda olduğu gibi veriler bir veri tabanında veya programın içinde gömülü değildir. Bilgiler ağın üzerinde saklı olup ortaya çıkartılması ve yorumlanması zordur.

2. 3. 4. Yapay Sinir Ağları Örnekleri Kullanarak Öğrenirler

Yapay sinir ağlarının olayları öğrenebilmesi için o olay ile ilgili örneklerin belirlenmesi gerekmektedir. Örnekleri kullanarak ilgili olay hakkında genelleme yapabilecek yeteneğe kavuşurlar (adaptif öğrenme). Örnek bulunmuyorsa ve yok ise yapay sinir ağının eğitilmesi mümkün değildir. Örnekler ise gerçekleşmiş olan olaylardır.

Örneğin bir doktor hastasına bazı sorular sorar ve aldığı cevaplara göre teşhis ederek ilaç yazar. Sorulan sorular ve verilen cevaplar ile konulan teşhis bir örnek olarak nitelendirilir. Bir doktorun belirli bir zaman içinde hastaları ile yaptığı görüşmeler ve koyduğu teşhisler not edilerek örnek alınırsa yapay sinir ağı benzer hastalıklara benzer teşhisi koyabilir. Elde edilen örneklerin olayı tamamı ile gösterebilmesi çok önemlidir. Ağa, olay bütün yönleri ile gösterilemez ve ilgili örnekler sunulmaz ise başarılı sonuçlar elde edilemez. Bu ağın sorunlu olduğundan değil, olayın ağa iyi gösterilemediğindendir. O nedenle örneklerin oluşturulması ve toplanması yapay sinir ağı biliminde özel bir öneme sahiptir.

2. 3. 5. Yapay Sinir Ağlarının Güvenle Çalıştırılabilmesi İçin Önce Eğitilmeleri ve Performanslarının Test Edilmesi Gerekmektedir

Yapay sinir ağlarının eğitilmesi demek, mevcut örneklerin tek tek ağa gösterilmesi ve ağın kendi mekanizmalarını çalıştırarak örnekteki olaylar arasındaki ilişkileri belirlemesidir. Her ağı eğitmek için elde bulunan örnekler iki ayrı sete bölünürler.

Birincisi ağı eğitmek için (eğitim seti) diğeri ise ağın performansını sınamak için (eğitim seti) kullanılır. Her ağ önce eğitim seti ile eğitilir. Ağ bütün örneklere doğru yanıtlar vermeye başlayınca eğitim işi tamamlanmış kabul edilir. Daha sonra ağın hiç görmediği test setindeki örnekler ağa gösterilerek ağın verdiği yanıtlara bakılır. Eğer ağ hiç görmediği örneklere kabul edilebilir bir doğrulukta yanıt veriyor ise o zaman ağın performansı iyi kabul edilir ve ağ kullanıma alınarak gerekirse çevrim içi kullanılır. Eğer ağın performansı yetersiz olursa o zaman yeniden eğitmek veya yeni örnekler ile eğitmek gibi bir çözüme gidilir. Bu işlem ağın performansı kabul edilebilir bir düzeye gelinceye kadar devam eder.

2. 3. 6. Görülmemiş Örnekler Hakkında Bilgi Üretebilirler

Ağ kendisine gösterilen örneklerden genellemeler yaparak görmediği örnekler hakkında bilgiler üretebilirler.

2. 3. 7. Algılamaya Yönelik Olaylarda Kullanılabilirler

Ağlar daha çok algılamaya yönelik bilgileri işlemede kullanılırlar. Bu konuda başarılı oldukları yapılan uygulamalarda görülmektedir. Bilgiye dayalı çözümlerde uzman sistemler kullanılmaktadır. Bazı durumlarda yapay sinir ağı ve uzman sistemleri birleştirmek daha başarılı sistemler oluşturmaya neden olmaktadır.

2. 3. 8. Şekil (Örüntü) İlişkilendirme ve Sınıflandırma Yapabilirler

Genel olarak ağların çoğunun amacı kendisine örnekler halinde verilen örüntülerin kendisi veya diğerleri ile ilişkilendirilmesidir. Diğer bir amaç ise sınıflandırma yapmaktır. Verilen örneklerin kümelendirilmesi ve belirli sınıflara ayrıştırılarak daha sonra gelen bir örneğin hangi sınıfa gireceğine karar vermesi hedeflenmektedir.

2. 3. 9. Örüntü Tamamlama Gerçekleştirebilirler

Bazı durumlarda ağa eksik bilgileri içeren bir örüntü veya bir şekil verilir. Ağın bu eksik bilgileri bulması istenir. Örneğin yırtık bir resmin kime ait olduğunun belirlemesi ve tam resmi vermesi gibi bir sorumluluk ağdan istenebilmektedir. Bu tür olaylarda yapay sinir ağlarının çok etkin çözümler ürettiği bilinmektedir.

2. 3. 10. Kendi Kendini Organize Etme ve Öğrenebilme Yetenekleri Vardır

Yapay sinir ağlarının örnekler ile kendisine gösterilen yeni durumlara adapte olması ve sürekli yeni olayları öğrenebilmesi mümkündür.

2. 3. 11. Eksik Bilgi İle Çalışabilmektedirler

Yapay sinir ağları kendileri eğitildikten sonra eksik bilgiler ile çalışabilirler ve gelen yeni örneklerde eksik bilgi olmasına karşın sonuç üretebilirler. Oysa geleneksel sistemler, bilgi eksik olunca çalışmazlar. Yapay sinir ağlarının eksik bilgiler ile çalışması performanslarının düşeceği anlamına gelmez. Performansın düşmesi eksik olan bilginin önemine bağlıdır. Hangi bilginin önemli olduğunu ağ kendisi eğitim sırasında öğrenmektedir. Kullanıcıların bu konuda bir fikri yoktur. Ağın performansı düşük olunca, kayıp olan bilginin önemli olduğu kararına varılır. Eğer ağın performansı düşmez ise eksik olan bilginin önemli olmadığı anlaşılır.

2. 3. 12. Hata Toleransına Sahiptirler

Yapay sinir ağlarının eksik bilgilerle çalışabilme yetenekleri hatalara karşı toleranslı olmalarını sağlamaktadır. Ağın bazı hücrelerinin bozulması ve çalışamaz duruma düşmesi halinde ağ çalışmaya devam eder. Ağın bozuk olan hücrelerinin sorumluluklarının önemine göre ağın performansında düşmeler görülebilir. Hangi hücrelerin sorumluluklarının önemli olduğuna da yine ağ eğitim sırasında kendisi karar verir. Bunu kullanıcı bilmemektedir. Ağın bilgisinin yorumlanamamasının nedeni de budur.

2. 3. 13. Belirsiz, Tam Olmayan Bilgileri İşleyebilmektedirler

Yapay sinir ağlarının belirsiz bilgileri işleyebilme yetenekleri vardır. Olayları öğrendikten sonra belirsizlikler altında ağlar öğrendikleri olaylar ile ilgili ilişkiler kurarak kararlar verebilirler.

2. 3. 14. Dereceli Bozulma Gösterirler

Yapay sinir ağlarının hatalara karşı toleranslı olmaları bozulmalarının da dereceli (göreceli) olmasına neden olmaktadır. Bir ağ, zaman içerisinde yavaş yavaş ve zarif bir şekilde bozulur. Bu eksik olan bilgiden veya hücrelerin bozulmasından kaynaklanır. Ağlar, herhangi bir problem ortaya çıktığında hemen anında bozulmazlar.

2. 3. 15. Dağıtık Belleğe Sahiptirler

Yapay sinir ağlarında bilgi ağa yayılmış durumdadır. Hücrelerin birbirleri ile bağlantılarının değerleri ağın bilgisini gösterir. Tek bir bağlantının bir anlamı yoktur.

Bu ağlarda ağın tamamı, öğrendiği olayın bütününü karakterize etmektedir. O nedenle bilgiler ağa dağıtılmış durumdadır. Bu ise dağıtık bir belleğin doğmasına neden olmaktadır.

2. 3. 16. Sadece Nümerik Bilgiler İle Çalışabilmektedirler

Yapay sinir ağları sadece nümerik bilgiler ile çalışırlar. Sembolik ifadeler ile gösterilen bilgilerin nümerik gösterime çevrilmeleri gerekmektedir. Sembolik bilgilerin nümerik değerler ile ifade edilmesinde bilgilerin yorumlanması ve kararların (üretilen çözümlerin) açıklanmasını zorlaştırmaktadır. (Öztemel, 2003), (Zurada, 1992).

2. 4. YAPAY SİNİR AĞLARININ ÖNEMLİ DEZAVANTAJLARI

Yapay sinir ağlarının bir çok avantajlı özelliklerinin yanı sıra bazı dezavantajları da vardır.

- Yapay sinir ağlarının donanım bağımlı çalışmaları önemli bir sorun olarak görülebilir. Ağların temel var oluş nedenlerinden birisi de paralel işlemciler üzerinde çalışabilmeleridir. Ağların, özellikle gerçek zamanlı bilgi işleyebilmeleri paralel çalışabilen işlemcilerin varlığına bağlıdır. Günümüzdeki makinelerin çoğu seri şekilde çalışabilmekte ve aynı zamanda sadece tek bir bilgiyi işleyebilmektedir. Paralel işlemleri seri makinelerde yapmak ise zaman kaybına yol açmaktadır. Bunun yanı sıra, bir ağın nasıl oluşturulması gerektiğini belirleyecek kuralların olmaması da başka bir dezavantajdır. Her problem farklı sayıda işlemci gerektirebilir. Bazı problemleri çözebilmek için gerekli olan paralel işlemcilerin tamamını bir arada (paralel olarak) çalıştırmak mümkün olmayabilir.

- Probleme uygun ağ yapısının belirlenmesi genellikle deneme yanılma yolu ile yapılmaktadır. Bu ise önemli bir problemdir. Çünkü eğer problem için uygun bir ağ oluşturulamaz ise çözümü olan bir problemin çözülememesi veya performansı düşük çözümlerin elde edilmesi söz konusu olabilir. Bu aynı zamanda bulunan çözümün en iyi çözüm olduğunu da garanti etmez. Yani yapay sinir ağları, kabul edilebilir çözümler üretebilir. En iyi çözümü garanti etmez.

- Bazı ağlarda ağın parametre değerlerinin (örneğin öğrenme katsayısı, her katmanda olması gereken proses eleman sayısı, katman sayısı vb.) belirlenmesinde de bir kural olmaması diğer bir problemdir. Bu, iyi çözümler bulmayı zor durumda bırakan bir etken olarak görülebilir. Bu parametrelerin belirlenmesi de kullanıcının tecrübesine bağlıdır. Her problem için ayrı faktörleri dikkate almayı

gerektirmektedir. Bu parametre değerleri için belirli standartların oluşturulması çok zor olduğundan her problem için ayrı ayrı değerlendirmeler yapılması gerekmektedir. Bu da önemli bir dezavantaj olarak görülebilir.

- Ağın öğreneceği problemin ağa gösterimi de çok önemli bir problemdir. Yapay sinir ağları sadece nümerik bilgiler ile çalışmaktadırlar. Problemin nümerik gösterime dönüştürülmesi gerekir. Bu kullanıcının becerisine bağlıdır. Uygun bir gösterim mekanizmasının kurulamamış olması problemin çözümünü engelleyebilir veya düşük performanslı bir öğrenme (çözüm) elde edilebilir. Problemin nümerik gösterimi mümkün olsa bile bunun ağa gösteriliş şekli problemin başarılı bir şekilde çözülmesini yakından etkiler. Örneğin bir olay hem ikili (ayrık) hem de sürekli değerler ile gösterilebilir. Bunun hangisinin daha başarılı bir öğrenme gerçekleştireceği ise bilinmemektedir. Bu konuda, kullanıcının tecrübesi de yeterli olmayabilir. Bu günümüzde bir çok olayın yapay sinir ağları ile çözülememesinin en önemli nedenlerinden birisidir.
- Ağın eğitiminin ne zaman bitirileceğine karar vermek içinde geliştirilmiş bir yöntem yoktur. Ağın örnekler üzerindeki hatasının belirli bir değerin altına indirilmesi eğitimin tamamlanması için yeterli görülmektedir. Fakat sonuçta en iyi öğrenmenin gerçekleştiği söylenememektedir. Sadece "iyi çözümler üretebilen bir ağ oluştu" denilmektedir. En iyi sonucu veren bir mekanizma henüz geliştirilememiştir. Bu konuda oldukça önemli olup çözülmesi için araştırmalar yapılması gerekmektedir.
- Bir diğer sorun ise, belki de en önemlisi, ağın davranışlarının açıklanamamasıdır. Bir probleme çözüm üretildiği zaman bunun nasıl ve neden üretildiği konusunda bir bilgi bulmak mümkün değildir. Bu ise ağın sonucuna olan güveni azaltmaktadır.

2. 5. YAPAY SİNİR AĞLARI İLE NELER YAPILABİLİR?

Yapay sinir ağları günümüzde geliştirilmiş en güncel ve en mükemmel örüntü tanıyıcı ve sınıflandırıcılardan sayılabilirler. Bu ağları bu kadar güncel yapan da, belirtildiği gibi, eksik bilgiler ile çalışabilme ve normal olmayan verileri işleyebilme yetenekleridir. Özellikle çok sayıda veriyi işleme gerektiren (radar verileri gibi) işlerde çok avantajlı sonuçlar üretebilmektedirler. Günümüzde bir çok problem aslında şekil tanıma problemi haline getirilmekte ve ondan sonra çözülmektedir. Bu nedenle, yapay sinir ağlarının kullanılabileceği bir çok alan vardır. Endüstriyel ve sosyal hayatta görülen binlerce örnek ile başarılı oldukları gözlemlenmiştir. Fakat her

problemi yapay sinir ağları ile çözmek mantıklı olmayabilir. Eğer herhangi bir problemin çözümü için yeterli etkinlikte ve verimlilikte çözüm yöntemi söz konusu ise yapay sinir ağının kullanılmasının bir anlamı yoktur. İlgili olay hakkında örneklerin olmayışı (bulunamayışı) da bu ağları kullanmamak için önemli bir nedendir. Bir problemin yapay sinir ağları ile çözülmesi için şu koşullardan birini sağlaması gerekir.

- Sadece yapay sinir ağları ile problemlere pratik çözümler üretebilme durumunun söz konusu olması gerekir.
- Başka çözüm yolları olmasına karşın yapay sinir ağlarının daha kolay ve daha etkin çözümler üretebilmesinin sağlanması gerekir. Başarılı uygulamalar incelendiğinde yapay sinir ağlarının, doğrusal olmayan, çok boyutlu, gürültülü, karmaşık, kesin olmayan, eksik, kusurlu, hata olasılığı yüksek sensör verilerinin olması ve problemin çözümü için özellikle bir matematik modelin ve algoritmanın bulunmaması hallerinde yaygın olarak kullanıldıkları görülmektedir. Bu amaçla geliştirilmiş ağlar genel olarak şu fonksiyonları yerine getirmektedir:
  - Probalistik fonksiyon kestirimleri
  - Örüntü tanıma
  - Sınıflandırma
  - Doğrusal olmayan sinyal işleme
  - İlişkilendirme veya örüntü eşleştirme
  - Doğrusal olmayan sistem modelleme
  - Zaman serisi analizleri
  - Optimizasyon
  - Sinyal filtreleme
  - Zeki ve doğrusal olmayan kontrol
  - Veri sıkıştırma

Belirtilen teorik uygulamaların ötesinde günlük yaşamda kullanılan finansal konulardan mühendisliğe ve tıp bilimine kadar bir çok uygulamadan bahsetmek mümkündür. Bunlardan bazıları:

- Veri madenciliği
- Optik karakter tanıma ve çek okuma
- Bankalardan kredi isteyen müracaatları değerlendirme
- Ürünün pazardaki performansını tahmin etme

- Kredi kartı hilelerini saptama
- Zeki araçlar ve robotlar için optimum rota belirleme
- Güvenlik sistemlerinde konuşma ve parmak izi tanıma
- Robot hareket mekanizmalarının kontrol edilmesi
- Mekanik parçaların ömürlerinin ve kırılmalarının tahmin edilmesi
- Kalite kontrolü
- İş çizelgeleme ve iş sıralaması
- İletişim kanallarında ki trafik yoğunluğunu kontrol etme ve anahtarlama
- Radar ve sonar sinyallerini sınıflandırma
- Üretim planlama ve çizelgeleme
- Kan hücreleri reaksiyonları ve kan analizlerini sınıflandırma
- Kanserin saptanması ve kalp krizlerinin tedavisi
- Beyin modellemesi çalışmaları

2. 6. YAPAY SİNİR AĞLARININ KISA BİR TARİHÇESİ

İnsanoğlu tarih boyunca insan beyninin nasıl çalıştığını merak etmiştir. Bilgisayarların doğması da aslında bu merakın bir sonucudur. İlk hesap makinelerinden günümüzdeki çok karmaşık bilgisayar sistemlerine geçişin temelinde, bu merak ve arayışın rolünü unutmamak gerekir. Bilgisayarlar başlangıçta sadece aritmetik işlemler yapmak amacı ile geliştirilmiş iken, bugün olayları öğrenmeleri ve çevre koşullarına göre karar vermeleri istenmektedir. Yapay sinir ağları günümüzde bu gelişmeyi tetikleyen bilim dallarından birisidir. Gelecekte de yine en önemli bilim dallarından birisi olacaktır.

Yapay sinir ağlarının tarihçesi nörobiyoloji konusuna insanların ilgi duyması ve elde ettikleri bilgileri bilgisayar bilimine uygulamaları ile başlamaktadır. Yapay sinir ağları ile ilgili çalışmalar 1970 öncesi ve sonrası diye ikiye ayırmak gerekmektedir. Çünkü, 1970 yılında bu bilimin tarihinde önemli bir dönüm noktası başlamış ve o zamana kadar olmaz diye düşünülen bir çok sorun çözülmüş ve yeni gelişmeler başlamıştır. Her şey bitti derken yapay sinir ağları yeniden doğmuştur.

2. 6. 1. 1970 Öncesi Çalışmalar

İnsan beyninin nasıl çalıştığı ve fonksiyonları uzun yıllar araştırılmıştır. 1980 yılında beynin fonksiyonları hakkında bilgi veren ilk eser yayınlanmıştır. 1940 dan önceki yıllarda bazı bilim adamlarının yapay sinir ağı kavramı üzerinde çalıştıkları bilinmektedir. Fakat bu çalışmaların mühendislik değerinin olduğu söylenemez.

1940'lı yıllardan sonra Hebb, McCulloch ve Pitts gibi bilim adamları yapılan araştırmaları mühendislik alanlarına kaydırmaya ve günümüzdeki yapay sinir ağlarının temellerini oluşturmaya başladılar. İlk yapay sinir hücresinin yapısını oluşturdular. Yapay sinir hücreleri ile her türlü mantıksal ifadeyi formülüze etmenin mümkün olduğunu gösterdiler. Hücrelerin birbirleri ile paralel çalışması gerektiği fikrini ortaya atarak öğrenme kurallarını belirlemeye başladılar. 1949 yılında Donald Hebb, yapay hücrelerden oluşan bir yapay sinir ağının değerlerini değiştiren bir öğrenme kuralı geliştirdi. "Hebbian öğrenme" kuralı denilen bu kural günümüzde de bir çok öğrenme kuralının temelini oluşturmaktadır. 1951 yılında ilk nuro-bilgisayar üretildi. Silikon teknolojinin geliştirilmesi ile bu çalışmalar 1960'lı yıllarda oldukça önemli gelişmelere neden oldu. Özellikle Rosenblatt tarafından geliştirilen algılayıcı model (perceptron) yapay sinir ağları tarihinde önemli bir gelişmeye öncülük etmiştir.

Çünkü bu model, daha sonraları geliştirilecek ve yapay sinir ağlarında devrim niteliğinde olacak olan çok katmanlı algılayıcıların temelini oluşturmaktadır.

Benzer şekilde Widrow ve Hoff ADALINE (ADaptive LInear NEuron) modelini ortaya attılar. Bu model, Rosenblatt'ın algılayıcı modeli ile aynı niteliklere sahip bir model olup sadece öğrenme algoritması daha da gelişmiş bir modeldir.

Bu arada bilim dünyasında başka gelişmeler olmuştur. 1956 yılında "Yapay Zeka" kavramı ortaya atılmış ve bilim dünyasında kabul görmüştür. İlk yapay zeka çalışmaları yapay sinir ağlarına pek değinmemiştir. Herkes ilgisini yapay zekaya çevirmiş ve nöro-bilgisayar ve yapay sinir ağları popülaritesini yitirmeye başlamıştır. Bu gidişata dur demek ve araştırmacıların ilgisini yapay sinir ağları üzerine tekrar çekmek için 1960'lı yıllarda Grosberg, Kahonen, Widrow, Fukushima vb. gibi bilim adamları tekrar konunun üzerine gitmeye başladılar.

1960'lı yılların sonunda yapay sinir ağı çalışmaları duraklama devrine girdi. Yapay sinir ağlarının tarihinde bir duraklama devrine neden olan ise Yapay Zeka biliminin o devirde önde gelen isimlerinden Misnky ve Pappert tarafından yazılan algılayıcılar başlıklı kitap oldu. Bu kitapta yazarlar özellikle yapay sinir ağlarına dayalı algılayıcıların bilimsel bir değerinin olmadığını ve doğrusal olmayan problemlere çözüm üretemediğini iddia ettiler. Tezlerini kanıtlamak için ise XOR probleminin çözülmemesini örnek gösterdiler. Bu örnek bir çok kişiyi tatmin etti ve çalışmalar kesildi. Yapay sinir ağı yapmak mümkün değildir inancı yükselmeye

başladı. Bu problem çözülünceye kadar dikkatleri yapay sinir ağına çekmek mümkün olmadı.

Bu zamana kadar yapılan çalışmaların bazıları kronolojik olarak aşağıdaki gibi listelenebilir:

1890 – İnsan beyninin yapısı ve fonksiyonları ile ilgili ilk yayının yazılması

1911 – İnsan beyninin bileşenlerinin belirli bir düzenek ile sinir hücrelerinden (nöronlar) oluştuğu fikrinin benimsenmesi

1943 – Yapay sinir hücrelerine dayalı hesaplama teorisinin ortaya atılması ve eşik değerli mantıksal devrelerin geliştirilmesi

1949 – Biyolojik olarak mümkün olabilen öğrenme prosedürünün bilgisayarlar tarafından gerçekleştirilecek biçimde geliştirilmesi

1956 – 1962 – ADALINE ve Widrow öğrenme algoritmasının geliştirilmesi

1957 – 1962 – Tek katmanlı algılayıcının geliştirilmesi

1965 – İlk makine öğrenmesi kitabının yayınlanması

1967 – 1969 – Bazı gelişmiş öğrenme algoritmalarının geliştirilmesi

1969 – Tek katmanlı algılayıcıların problemleri çözme yeteneklerinin olmadığının gösterilmesi

1969 – DARPA'nın (Amerika Birleşik Devletlerinde araştırma geliştirme çalışmalarını yürüten organizasyon) yapay sinir ağlarını desteklemeyi durdurup diğer yapay zeka çalışmalarına destek vermesi

2. 6. 2. 1970 Sonrası Çalışmalar

Çalışmaların 1969 yılında sekteye uğraması ve gerekli finansal desteklerin kesilmesine rağmen bazı bilim adamları çalışmalarına devam ettiler. Özellikle Amari, Anderson, Cooper, Fukushima, Grossberg, Kohonen ve Hopfield gibi araştırıcıların çalışmaları 1980'li yıllara gelindiğinde meyvelerini vermeye başladı ve yapay sinir ağları çalışmalarındaki sessizlik sona erdi. Elektrik mühendisi Kohonen ve nöropsikolojist Anderson "çağrışımlı bellek" konusunda hemen hemen birbirinin aynı çalışmalar yayınladılar. Bu çalışmalar daha sonraları geliştirilecek olan öğretmensiz öğrenme kurallarının temeli oldu. Carpenter ise Adaptif Rezonans Teorisini (ART) geliştirdi. Bu öğretmensiz öğrenme konusunda zamanının geliştirilmiş en karmaşık yapay sinir ağı oldu.

1970'lerin sonlarına doğru Fukushima görsel şekil ve örüntü tanıma amaçlı geliştirdiği NEOCOGNITRON modelini tanıttı. Bu model önceleri öğretmensiz öğrenme yapan bir model olacak şekilde geliştirilmesine rağmen daha sonraları

öğretmenli öğrenme yapacak hale getirilmiştir. Bu çalışmaların sonucunda daha çok mühendislik uygulamaları görülmeye başlandı. Bu model ara katmanlar kullanılarak öğrenme konusuna değiniyordu.

1982 ve 1984 yıllarında Hopfield tarafından yayınlanan çalışmalar ile yapay sinir ağlarının genelleştirilebileceği ve özellikle geleneksel bilgisayar programlama ile çözülmesi zor olan problemlere çözüm üretebileceğini gösterdi.

Aynı zamanlarda Rummelhart ve arkadaşları paralel programlama konularındaki çalışmalarını sonuçlandırıyor ve 2 ciltlik bir eser ortaya koyuyordu. Bu eserlerinde çok katmanlı algılayıcı modelinin temellerini atıyorlar ve daha sonra bu modeli geliştiriyorlardı. Çok katmanlı algılayıcıların bulunması yapay sinir ağlarının tarihsel gelişimi bakımından çok önemli bir adım oluyordu. Tek katmanlı algılayıcının çözemediği XOR problemi çok katmanlı algılayıcıların bulunması ile çözülmüş ve yapay sinir ağlarının çalışmadığını söyleyen bütün tezler çürütülmüştü.

Aynı zamanlarda Parker ve Werbos tarafından da çok katmanlı algılayıcı ile ilgili olarak bazı çalışmalarda yürütülüyordu. Çok katmanlı algılayıcı aynı zamanda Hopfield ve Boltzman makinelerinin sınırlamalarını da çözmüştü.

1988'de Broomhead ve Lowe Radyal tabanlı fonksiyonlar modelini geliştirdiler. Özellikle filtreleme problemlerine oldukça başarılı sonuçlar ürettiler. Daha sonra bu ağların daha gelişmiş şekli olan Probabilistik Ağlar ve Genel Regrasyon Ağları geliştirildi. 1970 yılından sonra yapılan çalışmaların bazıları kronolojik olarak aşağıdaki gibi listelenebilir:

1) 1969 – 1972 – Doğrusal ilişkilendiricilerin geliştirilmesi
2) 1972 – Korelasyon Matriks belleğinin geliştirilmesi
3) 1974 – Geriye yayılım modelinin (Çok katmanlı algılayıcının) ilk çalışmalarının geliştirilmesi
4) Öğretmensiz öğrenmenin geliştirilmesi
   - 1978 – ART modelinin geliştirilmesi
   - 1982 – Kohonen öğrenmesi ve SOM modeli geliştirilmesi
5) 1982 – Hopfield ağlarının geliştirilmesi
6) 1982 – Çok katmanlı algılayıcının geliştirilmesi
7) 1984 – Boltzman makinesinin geliştirilmesi
8) 1985 – Genelleştirilmiş Delta Kuralı
9) 1988 – RBF modelinin geliştirilmesi
10) 1988 – PNN modelinin geliştirilmesi

11) 1991 – GRNN modelinin geliştirilmesi

BÖLÜM 3.

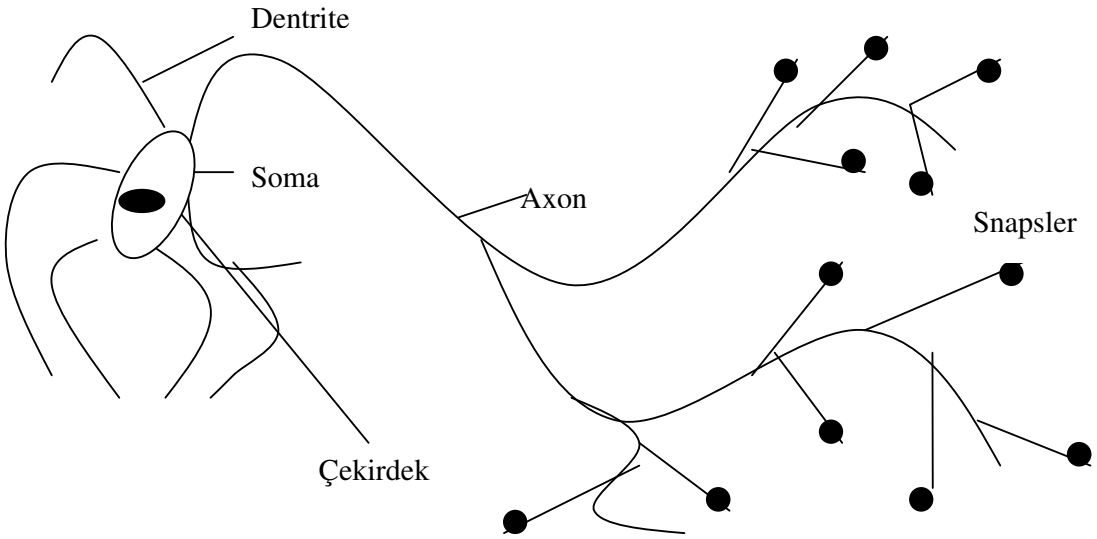
3. BİYOLOJİK ANLAMDA SİNİR AĞLARI

3. 1. BİYOLOJİK SİNİR HÜCRELERİ

Biyolojik sinir ağları, beynimizde bulunan bir çok sayıda sinir hücresinin bir kolleksiyonudur. Bir sinir ağı, milyarlarca sinir hücresinin bir araya gelmesi ile oluşmaktadır. Sinir hücreleri, birbirleri ile bağlanarak fonksiyonlarını yerine getirirler. Beynimizde 10^{10} adet sinir hücresi ve bunlarında $6x10^{13}$'ten fazla sayıda bağlantısının olduğu söylenmektedir. İnsan beyni, çok hızlı çalışabilen mükemmel bir bilgisayar gibi görülebilir. Bir grup insan resmi içinden tanıdık bir resmi 100-200 ms gibi kısa bir sürede fark edebilir. Halbuki geleneksel bilgisayarların böyle bir tanıma işlemini yapması çok daha uzun zamanlar alabilir. Bugün insan beyninin kapasitesinin çok küçük bir oranında kapasiteye sahip ve çalışabilen bir makine yapılsa olağanüstü bilgi işleme ve kontrol edebilme mekanizmaları geliştirmek ve mükemmel sonuçlar elde etmek mümkün olabilir. Biyolojik sinir ağlarının performansları küçümsenemeyecek kadar yüksek ve karmaşık olayları işleyebilecek yetenektedir. Yapay sinir ağları ile bu yeteneğin bilgisayarlara kazandırılması amaçlanmaktadır.

Biyolojik sinir ağları, insan beyninin çalışmasını sağlayan en temel taşlardan birisidir. İnsanın bütün davranışlarını ve çevresini algılamasını sağlar. Biyolojik sinir ağları, beş duyu organından gelen bilgiler ışığında geliştirdiği algılama ve anlama mekanizmalarını çalıştırarak olaylar arasındaki ilişkileri öğrenir. İnsan beyninin değişik bölgeleri değişik fonksiyonları yerine getirmektedir. Duyu organlarından gelen bilgiler (sinyaller), sinir sistemi sayesinde beyne taşınır ve beynin oluşturduğu kararları da yine sinir sistemi tarafından vücudun organlarına eylem olarak gönderilir. Bir sinir hücresi "Şekil 3. 1. "'de şematik olarak gösterilmektedir.

Şekilde gösterildiği gibi temel bir biyolojik sinir hücresi snapsler, soma, axon ve dentrite'lerden oluşmaktadır. Snapsler sinir hücreleri arasındaki bağlantılar olarak görülebilir. Bunlar fiziksel bağlantılar olmayıp bir hücreden diğerine elektrik sinyallerinin geçmesini sağlayan boşluklardır. Bu sinyaller somaya giderler. Soma bunları işleme tabi tutar, sinir hücresi kendi elektrik sinyalini oluşturur ve axon aracılığı ile dentrite'lere gönderir. Dentrit'ler ise bu sinyalleri snapslere göndererek diğer hücreler gönderilir. İki hücrenin birbirleri ile bilgi alış verişi snaptik bağlantılarda neurotransmitter'lar yolu ile sağlanmaktadır. Axon uçlarının her birisi başka bir hücre ile birleşmektedir.

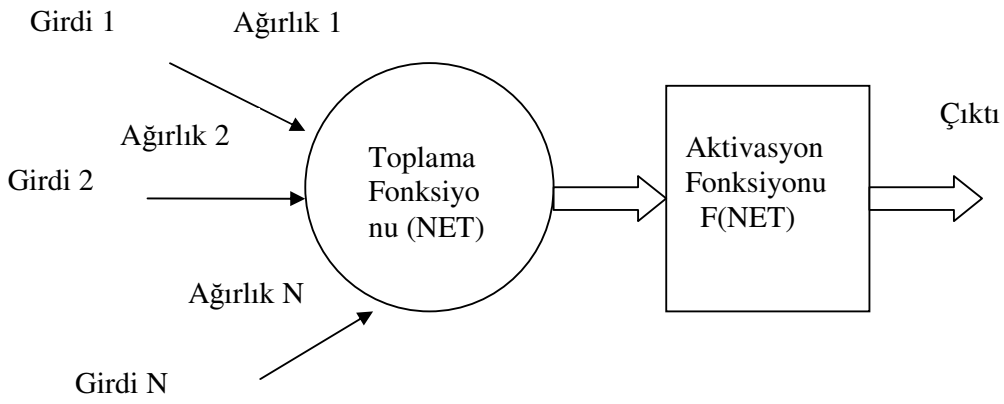


Şekil 3. 1. Bir Biyolojik Sinir Hücresinin Yapısı

Verilen özellikte milyarlarca sinir hücresi bir araya gelerek sinir sistemini oluşturmaktadır. Yapay sinir ağları biyolojik hücrelerin bu özelliklerinden yararlanarak geliştirilmiştir.

3. 2. YAPAY SİNİR HÜCRESİ (PROSES ELEMANI)

Biyolojik sinir ağlarının sinir hücreleri olduğu gibi yapay sinir ağlarının da yapay sinir hücreleri vardır. Yapay sinir hücreleri mühendislik biliminde proses elemanları olarak da adlandırılmaktadır. Her proses elemanının 5 temel elemanı vardır. “Şekil 3. 2. “



Şekil 3. 2. Yapay Sinir Hücresinin Yapısı

3. 2. 1. Girdiler

Bir yapay sinir hücresine (proses elemanına) dış dünyadan gelen bilgilerdir. Bunlar ağın öğrenmesi istenen örnekler tarafından belirlenir. Yapay sinir hücresine dış dünyadan olduğu gibi başka hücrelerden veya kendi kendisinden de bilgiler gelebilir.

3. 2. 2. Ağırlıklar

Ağırlıklar, bir yapay hücreye gelen bilginin önemini ve hücre üzerindeki etkisini gösterir. Şekildeki Ağırlık 1, Girdi 1'in hücre üzerindeki etkisini göstermektedir. Ağırlıkların büyük yada küçük olması önemli veya önemsiz olduğu anlamına gelmez. Bir ağırlığın değerinin sıfır olması o ağ için en önemli olay olabilir. Eksi değerler önemsiz demek değildir. O nedenle artı veya eksi olması etkisinin pozitif veya negatif olduğunu gösterir. Sıfır olması ise herhangi bir etkinin olmadığını gösterir. Ağırlıklar değişken veya sabit değerler olabilirler.

3. 2. 3. Toplama Fonksiyonu

Bu fonksiyon, bir hücreye gelen net girdiyi hesaplar. Bunun için değişik fonksiyonlar kullanılmaktadır. En yaygın olanı ağırlıklı toplamı bulmaktır. Burada her gelen girdi değeri kendi ağırlığı ile çarpılarak toplanır. Böylece ağa gelen net girdi bulunmuş olur. Bu şu şekilde formülüze edilmektedir.

$$NET = \sum_{i}^{n} G_i A_i \qquad (3.1)$$

Burada G girdiler, A ise ağırlıkları, n ise bir hücreye gelen toplam girdi (proses elemanı) sayısını göstermektedir. Yalnız yapay sinir ağlarında daima bu formülün kullanılması şart değildir. Uygulanan yapay sinir ağı modellerinden bazıları kullanılacak toplama fonksiyonunu belirleyebilmektedir. Bir problem için en uygun toplama fonksiyonunu belirlemek için bulunmuş bir formül yoktur. Genellikle deneme yanılma yolu ile toplama fonksiyonu belirlenmektedir. Bir yapay sinir ağında bulunan proses elemanlarının tamamının aynı toplama fonksiyonuna sahip olmaları gerekmez. Her proses elemanı bağımsız olarak farklı bir toplama fonksiyonuna sahip olabileceği gibi hepsi aynı fonksiyona da sahip olabilir. Hatta ağın bazı proses elemanları grup halinde aynı toplama fonksiyonuna sahip olabilir. Diğerleri ise farklı fonksiyonlar kullanabilirler. Bu tamamen tasarımcının kendi öngörüsüne dayanarak verdiği karara bağlıdır.

3. 2. 4. Aktivasyon Fonksiyonu

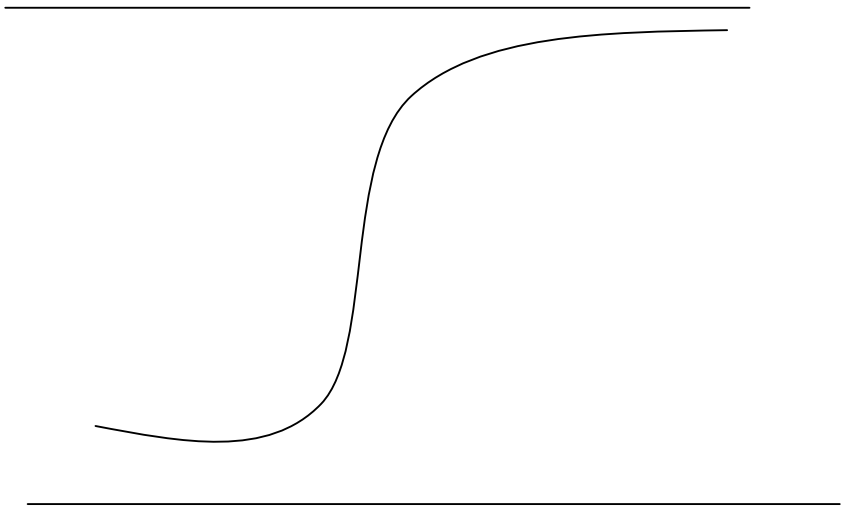
Bu fonksiyon, hücreye gelen net girdiyi işleyerek hücrenin bu girdiye karşılık üreteceği çıktıyı belirler. Toplama fonksiyonunda olduğu gibi aktivasyon fonksiyonu olarak da çıktıyı hesaplamak içinde değişik formüller kullanılmaktadır. Bazı modeller (örneğin çok katmanlı algılayıcı) bu fonksiyonun türevinin alınabilir bir fonksiyon olmasını şart koşmaktadır. Toplama fonksiyonunda olduğu gibi aktivasyon fonksiyonunda da ağın proses elemanlarının hepsinin aynı fonksiyonu kullanması gerekmez.

Bazı elemanlar aynı fonksiyonu diğerleri farklı fonksiyonları kullanabilirler. Bir problem için en uygun fonksiyonda yine tasarımcının denemeleri sonucunda belirleyebileceği bir durumdur. Uygun fonksiyonu gösteren bir formül bulunmuş değildir.

Günümüzde en yaygın olarak kullanılan Çok Katmanlı Algılayıcı modelinde genel olarak aktivasyon fonksiyonu olarak sigmoid fonksiyonu kullanılmaktadır. Bu fonksiyon şu formül ile gösterilmektedir.

$$F(NET) = \frac{1}{1 + e^{-NET}} \qquad (3.2)$$

Burada NET proses elemanına gelen NET girdi değerini göstermektedir. Bu değer toplama fonksiyonu kullanılarak belirlenmektedir. Sigmoid fonksiyonu şekilsel olarak da "Şekil 3. 3. "'de gösterilmiştir.



Şekil 3. 3. Sigmoid Fonksiyonunun Şekilsel Gösterimi

Aktivasyon fonksiyonu olarak kullanılacak olan diğer fonksiyonlara örnekler ise "Tablo 3. 1. "'de verilmiştir.

Aktivasyon Fonksiyonu	Açıklama
Lineer Fonksiyon F(NET)=NET	Gelen girdiler olduğu gibi hücrenin çıktısı olarak Kabul edilir.
Adım Fonksiyonu F(NET)= 1 eğer NET > eşik_değer 0 eğer NET <= eşik_değer	Gelen NET girdi değerinin belirlenen bir eşik değerinin Altında veya üstünde olmasına göre hücrenin çıktısı 1 veya 0 değerlerini alır.
Sinüs fonksiyonu F(NET) = Sin (NET)	Öğrenmesi düşünülen olayların sinüs fonksiyonuna Uygun dağılım gösterdiği durumlarda kullanılır.
Eşik değer fonksiyonu 0 eğer NET <= 0 F(NET) = NET eğer 0 < NET < 1 1 eğer NET >= 1	Gelen bilgileri 0 veya 1'den büyük veya küçük olmasına göre bir değerler alır. 0 ve 1 arasında değerler alabilir. Bunların dışında değerler alamaz.
Hiperbolik tanjant fonksiyonu $F(NET)=(e^{NET} + e^{-NET})/(e^{NET} - e^{-NET})$	Gelen NET girdi değerinin tanjant fonksiyonundan Geçirilmesi ile hesaplanır

Tablo 3. 1. Aktivasyon Fonksiyonu Örnekleri

3. 2. 5. Hücrenin Çıktısı

Aktivasyon fonksiyonu tarafından belirlenen çıktı değeridir. Üretilen çıktı dış dünyaya veya başka bir hücreye gönderilir. Hücre kendi çıktısını kendisine girdi olarak da gönderebilir. Bir proses elemanının birden fazla girdisi olmasına karşın sadece bir çıktısı olmaktadır. Ağ şeklinde gösterildiğinde bir proses elemanının birden fazla çıktısı varmış gibi görülmektedir. Aslında bir proses elemanından çıkan tek bir çıktı değeri vardır. Aynı değer birden fazla proses elemanına girdi olarak gitmektedir. (Öztemel, 2003)

3. 3. YAPAY SİNİR AĞININ YAPISI

Yapay sinir hücreleri bir araya gelerek yapay sinir ağını oluştururlar. Sinir hücrelerinin bir araya gelmesi rasgele olmaz. Genel olarak hücreler 3 katman halinde ve her katman içinde paralel olarak bir araya gelerek ağı oluştururlar. Bu katmanlar: şematik olarak "Şekil 3. 4. "de görülmektedir;

3. 3. 1. Girdi Katmanı

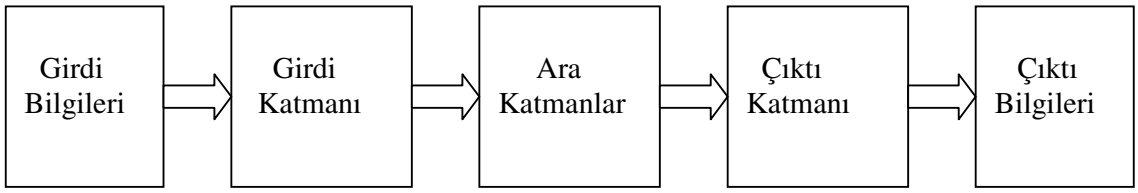
Bu katmandaki proses elemanları dış dünyadan bilgileri alarak ara katmanlara transfer etmekle sorumludurlar. Bazı ağlarda girdi katmanında herhangi bir bilgi işleme olmaz.

3. 3. 2. Ara Katmanlar

Girdi katmanından gelen bilgiler işlenerek çıktı katmanına gönderilir. Bu bilgilerin işlenmesi ara katmanlarda gerçekleştirilir. Bir ağ için birden fazla ara katman olabilir.

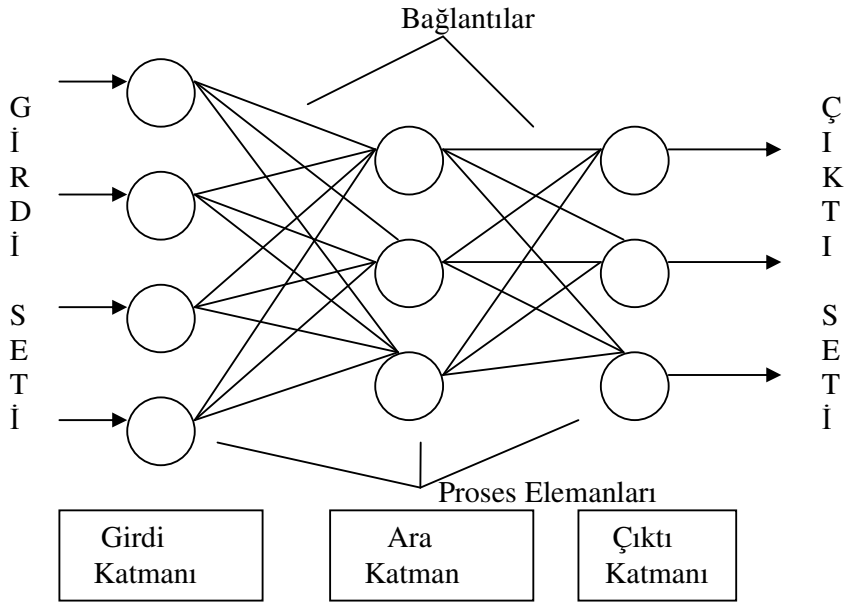
3. 3. 3. Çıktı Katmanı

Bu katmandaki proses elemanları ara katmandan gelen bilgileri işleyerek ağın girdi katmanından sunulan girdi seti (örnek) için üretmesi gereken çıktıyı üretirler. Üretilen çıktı dış dünyaya gönderilir.



Yapay Sinir Ağı

Şekil 3. 4. Yapay Sinir Ağı Katmanlarının Birbirleri İle İlişkileri



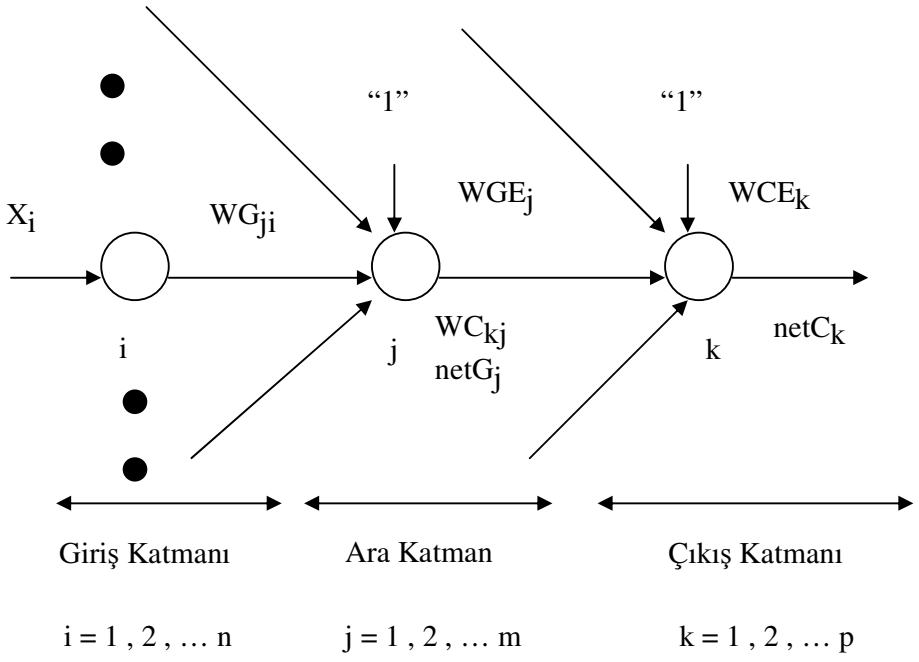
Şekil 3. 5. Bir Yapay Sinir Örneği

Bu üç katmanın her birinde bulunan proses elemanları ve katmanlar arası ilişkiler şematik olarak “Şekil 3. 5. “de gösterilmektedir. Şekildeki yuvarlak şekiller proses elemanlarını göstermektedir. Her katmanda birbirine paralel bağlı elemanlar söz konusudur. Proses elemanlarını birbirlerine bağlayan çizgiler ise ağın bağlantılarını göstermektedir. Proses elemanları ve bağlantıları bir yapay sinir ağını oluştururlar. Bu bağlantıların ağırlık değerleri öğrenme sırasında belirlenmektedir.
(Öztemel, 2003).

BÖLÜM 4.

4. MATEMATİKSEL VE ŞEKİLSEL OLARAK YAPAY SİNİR AĞI

4. 1. ŞEKİLSEL OLARAK YAPAY SİNİR AĞI



Şekil 4. 1. Öz Olarak Belirtilmiş Bir Yapay Sinir Ağı

4. 2. MATEMATİKSEL OLARAK BİR YAPAY SİNİR AĞI MODELİ

Matematiksel modelde kullanılan değişkenler ile şekil arasındakilerin ilişkisi;

$$W_{ji} = WG_{ji}$$

$$net_j = netG_j$$

$$W_{kj} = WC_{kj}$$

$$net_k = netC_k$$

Giriş Katmanı i = 1 …… n

Ara Katman j = 1 …… m

Çıkış Katmanı k = 1 …… p

$$net_j = \sum_{i=1}^{n} W_{ji} \cdot X_i \qquad (4.1)$$

$$Y_j = f_j (net_j) \qquad (4.2)$$

$$net_k = \sum_{j=1}^{m} W_{kj} \cdot Y_j \qquad (4.3)$$

$$O_k = f_k (net_k) \quad (\text{Ağdan elde edilen çıkış}) \qquad (4.4)$$

d_k : İstenen çıkış

$$e_k = d_k - O_k \qquad (4.5)$$

$$E = \frac{1}{2} \sum_k (d_k - O_k)^2 \qquad (4.6)$$

$$\Delta W_{kj} = - \varepsilon . \frac{\partial E}{\partial W_{kj}} \qquad (4.7)$$

$$\frac{\partial E}{\partial Wkj} = \frac{\partial E}{\partial net_k} . \frac{\partial net_k}{\partial Wkj} \qquad (4.8)$$

$$\frac{\partial net_k}{\partial Wkj} = \frac{\partial}{\partial Wkj} . \sum_j W_{kj} \cdot Y_j = \sum_j \frac{\partial Wkj \cdot Y_j}{\partial Wkj} = Y_j \qquad (4.9)$$

$$\delta_0 = - \frac{\partial E}{\partial net_k} \qquad (4.10)$$

$$\frac{\partial E}{\partial Wkj} = - \delta_0 \cdot Y_j \qquad (4.11)$$

$$\Delta W_{kj} = \varepsilon \ \delta_0 \cdot Y_j \quad (\text{Ara Katman ile Çıkış Katmanı Arası}) \qquad (4.12)$$

$$\delta_0 = - \frac{\partial E}{\partial net_k} = - \frac{\partial E}{\partial O_k} . \frac{\partial O_k}{\partial net_k} \quad (4.13)$$

$$\frac{\partial E}{\partial O_k} = - (d_k - O_k) \quad (4.14)$$

$$\frac{\partial O_k}{\partial net_k} = f '_k (net_k) \quad (4.15)$$

$$\delta_0 = (d_k - O_k) . f '_k (net_k) \quad (4.16)$$

$$\Delta W_{kj} = \varepsilon (d_k - O_k) . f '_k (net_k) . Y_j \quad (4.17)$$

$$\Delta W_{ji} = - \varepsilon . \frac{\partial E}{\partial W_{ji}}$$

$$= - \varepsilon . \frac{\partial E}{\partial net_j} . \frac{\partial net_j}{\partial W_{ji}} = - \varepsilon . \frac{\partial E}{\partial net_j} . X_i$$

$$= - \varepsilon . \frac{- \partial E}{\partial Y_j} . \frac{\partial Y_j}{\partial net_j} . X_i$$

$$= \varepsilon . \frac{\partial E}{\partial Y_j} . f '_j (net_j) . X_i \quad (4.18)$$

$$\Delta W_{ji} = \varepsilon . \delta Y . X_i \quad (4.19)$$

$$\frac{\partial E}{\partial Y_j} = -\sum_k \frac{\partial E}{\partial net_k} \cdot \frac{\partial net_k}{\partial Y_j} = \sum_k \frac{-\partial E}{\partial net_k} \cdot \frac{\partial}{\partial Y_j} \cdot \sum_m W_{km} \cdot Y_j \quad (4.20)$$

$$\sum_k \frac{-\partial E}{\partial net_k} \cdot W_{kj} = \sum_k \delta_0 \cdot W_{kj} \quad (4.21)$$

$$\delta Y = f\,'_j (net_j) \cdot \sum_k \delta_0 \cdot W_{kj} \quad (4.22)$$

$$\delta Y = (d_y - O_y) \cdot f\,'_j (net_j) \quad (4.23)$$

Sigmoid fonksiyonu kullanılıyor ise

$$f\ (net_j) = Y_j = \frac{1}{1 + e^{-netj}} \quad (4.24)$$

$$f\,' (net_j) = \frac{1}{1 + e^{-netj}} \cdot \frac{1 + e^{-netj} - 1}{1 + e^{-netj}} \quad (4.25)$$

$$\frac{\partial Y_j}{\partial net_j} = Y_j \cdot (1 - Y_j) \quad (4.26)$$

$$\frac{\delta O_k}{\partial net_k} = f\,'_k\ (net_k) = O_k \cdot (1 - O_k) \quad (4.27)$$

$$\delta_0 = (d_k - O_k) \cdot O_k \cdot (1 - O_k) \quad (4.28)$$

$$\delta Y = Y_j \cdot (1 - Y_j) \cdot \sum_k \delta_0 \cdot W_{kj} \quad (4.29)$$

δ_0 : Çıkış Katmanı

δY : Ara (Gizli) Katman

$$\Delta W_{ji} (n+1) = \varepsilon \cdot \delta Y \cdot X_i + \alpha \cdot \Delta W_{ji} (n) \quad (4.30)$$

$$\Delta W_{kj}(n+1) = \varepsilon \cdot \delta_0 \cdot Y_j + \alpha \cdot \Delta W_{kj}(n) \quad (4.31)$$

ε : Öğrenme Sayısı

α : Momentum Sayısı

n : İterasyon Sayısı

Oluşan formülleri "Şekil 4. 1. "'e uyguladığımızda ;

Ara (Gizli) katman ile çıkış katmanı arası için

$$\Delta WCE_k(r+1) = \varepsilon \cdot \delta_0 + \alpha \cdot \Delta WCE_k(r) \quad (4.32)$$

$$WCE_k(r+1) = WCE_k(r) + \Delta WCE_k(r+1) \quad (4.33)$$

$$\Delta WC_{kj}(r+1) = \varepsilon \cdot \delta_0 \cdot Y_j + \alpha \Delta WC_{kj}(r) \quad (4.34)$$

$$WC_{kj}(r+1) = WC_{kj}(r) + \Delta WC_{kj}(r+1) \quad (4.35)$$

Giriş katmanı ile ara (gizli) katman arası için

$$\Delta WGE_j(r+1) = \varepsilon \cdot \delta Y + \alpha \cdot \Delta WGE_k(r) \quad (4.36)$$

$$WGE_j(r+1) = WGE_j(r) + \Delta WGE_j(r+1) \quad (4.37)$$

$$\Delta WG_{ji}(r+1) = \varepsilon \cdot \delta Y \cdot X_i + \alpha \cdot \Delta WG_{ji}(r) \quad (4.38)$$

$$WG_{ji}(r+1) = WG_{ji}(r) + \Delta WG_{ji}(r+1) \quad (4.39)$$

(Okatan, 2005).

BÖLÜM 5.

5. YAPAY SİNİR AĞLARINI KULLANARAK KARAKTER ALGILAMA

5. 1. ALGILAMA YÖNTEMİ

Elle yazılmış veya basılmış karakterlerin öz işler (otomatik) olarak algılanması sorunu uzun zamandır üzerinde çalışılan bir konudur. Karakter algılanması konusunda birçok teknik ve ticari gelişmeler sağlanmış durumdadır. Bankalar çek üzerindeki sayısal bilgiyi ve kredi kartlarını karakter algılayıcılar kullanarak yapmaktadırlar. Karakter algılama işlemini birçok klasik algoritma ile yapılabilmekle beraber, yapay sinir ağlarını kullanarak da gerçekleştirmek mümkündür. Yapay sinir ağlarının klasik yöntemlere göre üstün yanları bulunmaktadır.

Klasik yöntemlerde karakter algılama olayı gerçekleştirilmekte fakat bazı durumlar da birçok zorluk ortaya çıkmaktadır. Örnek olarak düzgün alfa nümerik şekillerin algılanmasını göz önüne alalım. Böyle bir şekli algılayabilmek için ilk akla gelen şey karakteri bir matris halinde temsil etmektir. Matris üzerinde dolu bulunan her piksel " 1 "değerini diğer yerler ise " 0 " değerini almak üzere ikili sayı olarak kodlayabilmek mümkündür. Bu bir vektör olarak yazılırsa 35 bitlik ikili sayı veya buna karşılık gelen onluk sayıya göre bir karakter tablosundan arama yapılarak ASCII (American Standard Code for Information Interchange) kodu elde edilebilir.

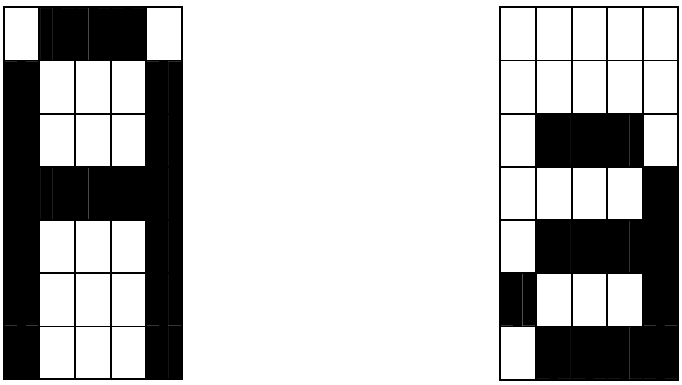
Bu yaklaşım her ne kadar hızlı, makul ve kolay bir yöntem ise de gerçek sistemlerde bu yöntemin işletilemeyeceği bazı durumlar vardır. Örneğin daha gerçekçi bir ortamda aynı şeklin ASCII dönüştürücü işlemine geldiğini düşünelim. Gürültü nedeni ile görüntüde meydana gelen değişimler yukarıda anlatılan statik yöntemin çalışmasını engelleyecektir. Giriş ve hedef arasındaki uyumun tam olması gerekliliğinden yanlış bir karakter veya olumsuz bir sonuç üretilecektir.

Burada görüntüdeki bütün pikselleri paralel olarak inceleyebilecek yeni bir işleme sistemine ihtiyaç duyulacaktır. Sistem, örnek kümesindeki ilişkiyi öğrenerek uyarlayabilecek ve bu öğrendiği ilişkiyi yeni giriş karakterlerine de uygulayabilir yetenekte olacaktır. Bu sistem, örneğin bilinen karakterlere benzeyen fakat gürültü nedeni ile bozuk olan pikseller gibi, önceden görülen şekillere benzeyen herhangi bir girişin özelliklerine odaklanabilmeli ve aynı zamanda gürültüyü tamamen ihmal edebilmelidir. İşte bu özelliklerin gerçekleştirilebileceği bir sistem yapay sinir ağları olmaktadır. (Elmas, 2003)

5. 2. KARAKTERLERİN GÖSTERİMİ (MATRİS VE DİZİ OLARAK)

Bu çalışmada Türk Alfabesi'ndeki 29 harf temel alınmıştır. Büyük ve küçük harflerin toplam sayısı 58 olmaktadır. 58 harf için 58*35 boyutlarında bir matris oluşturulmuştur.

Bu aşamadan önce her bir karakter 7x5 'lik matris başka bir deyiş ile 35 elemanlı bir dizi haline getirilmiştir. Fotoğrafı alınan karakter örneğin MATLAB gibi bir program kullanılarak belirtilen boyutlara getirilebilir.



Şekil 5. 1.
(Matris Yapısına Uyarlanmış İki Adet Harf)

01110	00000
10001	00000
10001	01110
11111	00001
10001	01111
10001	10001
10001	01111

Diğer karakterlerin matris yapısının tamamı (Ek – 1) 'de verilmiştir.

Çıkış ise yine 58 harften oluşacaktır. Bu yüzden çıkış bilgisi matris yapısında ve boyutu 58*58 olacaktır.

Yapay sinir ağındaki giriş işlem elemanı sayısı 35 adet olup, çıkış da ki işlem elemanı sayısı 58 olarak tasarlanmıştır. Bu çalışmada, tek gizli katmanlı ileri beslemeli geri yayılımlı yapay sinir ağı kullanılmıştır. Yapay sinir ağının öğrenme yöntemi olarak ise genelleştirilmiş Delta Kuralı'ndan faydalanılmıştır. (Kılıç, 1998), (Wilson, 1992).

BÖLÜM 6.

6. EĞİTİM İÇİN GİRİŞ İŞLEMİ DOSYASI

Dosya Adı : “C:\dosya01.txt”

(BÜYÜK HARFLER İÇİN)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
A	0	1	1	1	0	1	0	0	0	1	1	0	0	0	1	1	1	1	1	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1
B	1	1	1	1	0	1	0	0	0	1	1	0	0	0	1	1	1	1	1	0	1	0	0	0	1	1	0	0	0	1	1	1	1	1	0
C	0	1	1	1	1	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	1	1	1
Ç	0	1	1	1	1	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	1	1	1	0	0	1	0	0
D	1	1	1	0	0	1	0	0	1	0	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	1	0	1	1	1	0	0
E	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	1	1	1	1	0	1	0	0	0	0	1	0	0	0	0	1	1	1	1	1
F	1	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
G	0	1	1	1	0	1	0	0	0	1	1	0	0	0	0	1	1	1	1	0	1	0	0	0	1	1	0	0	0	1	0	1	1	1	0
Ğ	0	1	1	1	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	1	1	1	1	0	1	0	0	0	1	0	1	1	1	0
H	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	1	1	1	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1
I	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0
İ	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0
J	0	1	1	1	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	1	0	1	0	0	0	1	0	0	0
K	1	0	0	0	1	1	0	0	1	0	1	0	1	0	0	1	1	0	0	0	1	0	1	0	0	1	0	0	1	0	1	0	0	0	1
L	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	1	1	1	1
M	1	1	0	1	1	1	0	1	0	1	1	0	1	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1
N	1	0	0	0	1	1	1	0	0	1	1	0	1	0	1	1	0	1	0	1	1	0	1	0	1	1	0	0	1	1	1	0	0	0	1
O	0	1	1	1	0	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	0	1	1	1	0
Ö	0	1	0	1	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	0	1	1	1	0
P	1	1	1	1	0	1	0	0	0	1	1	0	0	0	1	1	1	1	1	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
R	1	1	1	1	0	1	0	0	0	1	1	0	0	0	1	1	1	1	1	0	1	0	1	0	0	1	0	0	1	0	1	0	0	0	1
S	0	1	1	1	0	1	0	0	0	1	1	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	0	0	0	1	0	1	1	1	0
Ş	0	1	1	1	0	1	0	0	0	0	1	0	0	0	0	0	1	1	1	0	0	0	0	0	1	0	1	1	1	0	0	0	1	0	0
T	1	1	1	1	1	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0
U	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	0	1	1	1	0
Ü	0	1	0	1	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	0	1	1	1	0
V	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	0	1	0	1	0	0	0	1	0	0
Y	1	0	0	0	1	0	1	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0
Z	1	1	1	1	1	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	1	1	1	1	1

Tablo 6. 1. Büyük Harfler İçin (Giriş)

(KÜÇÜK HARFLER İÇİN)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
a	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	0	1	1	1	1	1	0	0	0	1	0	1	1	1	1
b	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	1	1	1	0	1	0	0	0	1	1	0	0	0	1	1	1	1	1	0
c	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	1	1	1
ç	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0	1	1	1	1	0	0	1	0	0
d	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	1	1	1	1	1	0	0	0	1	1	0	0	0	1	0	1	1	1	1
e	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0	1	1	1	1
f	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0
g	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	1	0	1	1	1	1	0	0	0	0	1	0	1	1	1	0
ğ	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	1	0	1	1	1	1	0	0	0	0	1	0	1	1	1	0
h	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	1	1	1	0	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1
ı	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0
i	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0
j	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	1	0	0	0
k	1	0	0	0	0	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0	1	1	0	0	0	1	0	1	0	0	1	0	0	1	0
l	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
m	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	0	1	0	1	1	0	1	0	1	1	0	0	0	1	1	0	0	0	1
n	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1
o	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	0	1	1	1	0
ö	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	1	0	1	1	1	0
p	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0	0	0	1	1	1	1	1	0	1	0	0	0	0	1	0	0	0	0
r	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
s	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	1	1	1	0	0	0	0	0	1	0	1	1	1	0
ş	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	1	1	1	0	0	0	0	0	1	0	1	1	1	0	0	0	1	0	0
t	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	1	1	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0
u	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	0	1	1	1	0
ü	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	1	0	1	1	1	0
v	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	0	1	0	1	0	0	0	1	0	0
y	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	1	0	1	1	1	1	0	0	0	0	1	0	1	1	1	0
z	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	1	0	0	0	0	1	1	1	1	1

Tablo 6. 2. Küçük Harfler İçin (Giriş)

Yukarıda adı geçen dosyada her iki tablo alt alta birbirlerinin devamı olarak yer almaktadır. Örnek olarak ilgili dosyada ilk sekiz harfin bilgisi yer almaktadır.

BÖLÜM 7.

7. EĞİTİM İÇİN ÇIKIŞ İŞLEMİ DOSYASI

Dosya Adı : "C:\dosya02.txt"

(BÜYÜK HARFLER İÇİN)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
A	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Ç	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
F	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Ğ	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
İ	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
J	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
K	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
O	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Ö	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
Ş	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
T	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
U	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Ü	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
Z	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Tablo 7. 1. Büyük Harfler İçin (Çıkış)

(KÜÇÜK HARFLER İÇİN)

	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58
a	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
b	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ç	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
d	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
f	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
g	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ğ	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
h	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ı	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
i	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
j	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
k	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
l	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
m	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
n	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
o	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
ö	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
p	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
r	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
s	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
ş	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
t	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
u	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
ü	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
v	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
z	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Tablo 7. 2. Küçük Harfler İçin (Çıkış)

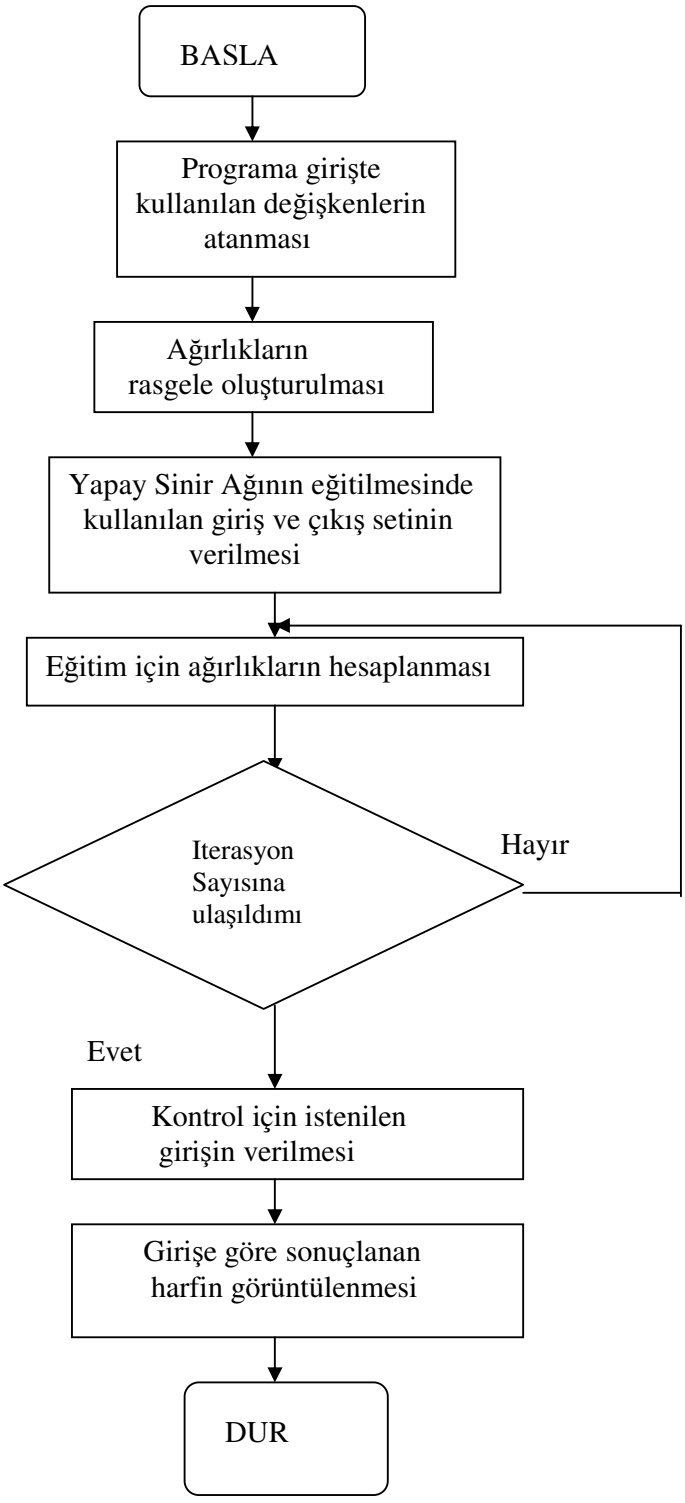
Yukarıda adı geçen dosyadaki her iki tablo alt alta birbirlerinin devamı olarak yer almaktadır. Örnek olarak ilgili dosyada ilk sekiz harfin bilgisi vardır.

Dosya: “C:\dosya03.txt” ‘ de ise test amacı ile çıkışı istenen harflerin bilgileri bulunmaktadır ve ilk sekiz harfin bilgisi söz konusudur.

BÖLÜM 8.

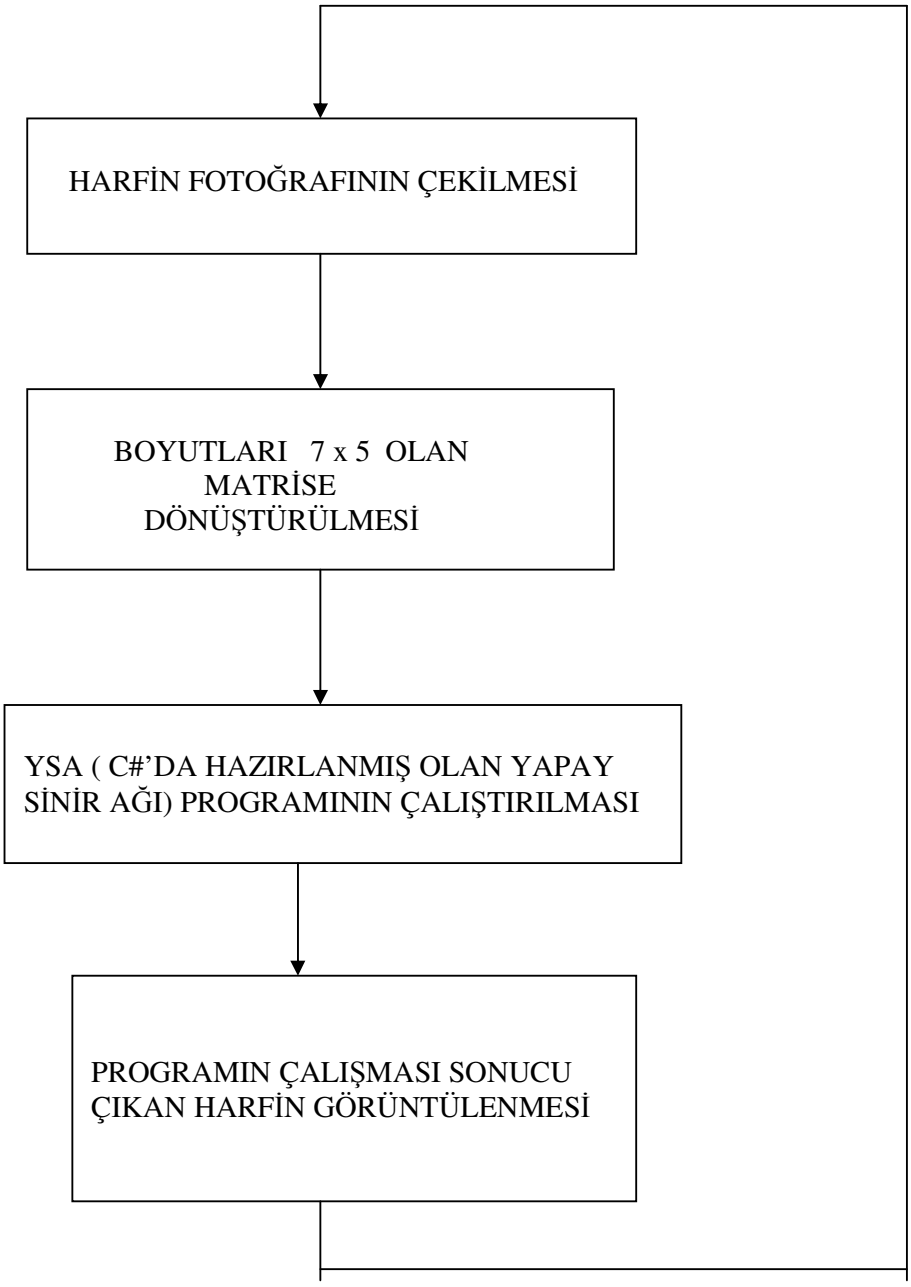
8. KARAKTER ALGILAMAYA YÖNELİK OLARAK HAZIRLANMIŞ OLAN YAPAY SİNİR AĞI PROGRAMININ AKIŞ ŞEMASI

8. 1. PROGRAMIN AKIŞ ŞEMASI



Burada verilen akış şeması ayrıntılı akış şemasının bir özeti olarak verilmiştir. Ayrıntılı akış şeması EK – 2 ‘de dir .

8. 2. GENEL ANLAMDA AKIŞ ŞEMASI



BÖLÜM 9.

9. C#'DA HAZIRLANMIŞ OLAN YAPAY SİNİR AĞI PROGRAMI

Program Microsoft Visual Studio 2005 ortamında hazırlanmış ve çalıştırılmıştır. Programda ağırlıklar ilk önce rasgele (random) alınmıştır. Ardından giriş ve çıkış seti birer dosya olarak verilmiştir. Bu setleri kullanan program iterasyon sayısı kadar döngüde kalarak ağırlıkları hesaplamıştır. İterasyon sayısına ulaşıldığında hesaplanmış olan son ağırlıklar, sonuç ağırlıklar olarak alınmıştır. Bu aşamadan sonra kontrol için istenilen harflere ait dosya yapısı girilerek çıktı alınıp doğruluk değerleri kontrol edilmiştir. Son aşamada, girişlere göre sonuçlanan harfler görüntülenir ve başka giriş yoksa programdan çıkılır. Ayrıca döngü sırasında toplam 20 adet hata değeri alınıp grafiği de çizdirilmiştir.

Programın ayrıntılı akışı EK – 3 'de verilmiştir.

BÖLÜM 10.

10. YAPAY SİNİR AĞI PROGRAMI ÇALIŞMASI BİLGİLERİ

10. 1. BİRİNCİ UYGULAMA GİRİŞLERİ VE SONUÇLARI

10. 1. 1. GİRİŞLER

Giriş Dosyası : c:\dosya01.txt

Çıkış Dosyası : c:\dosya02.txt

Sonucu İstenen Bilgilerin Bulunduğu Dosya : c:\dosya03.txt

Iterasyon Sayısı : 100000

Giriş Katmanındaki İşlem Elemanı Sayısı : 35

Toplam Boyut (Grup Sayısı) : 8

Ara Katmandaki İşlem Elemanı Sayısı : 10

Çıkış Katmanındaki İşlem Elemanı Sayısı : 8

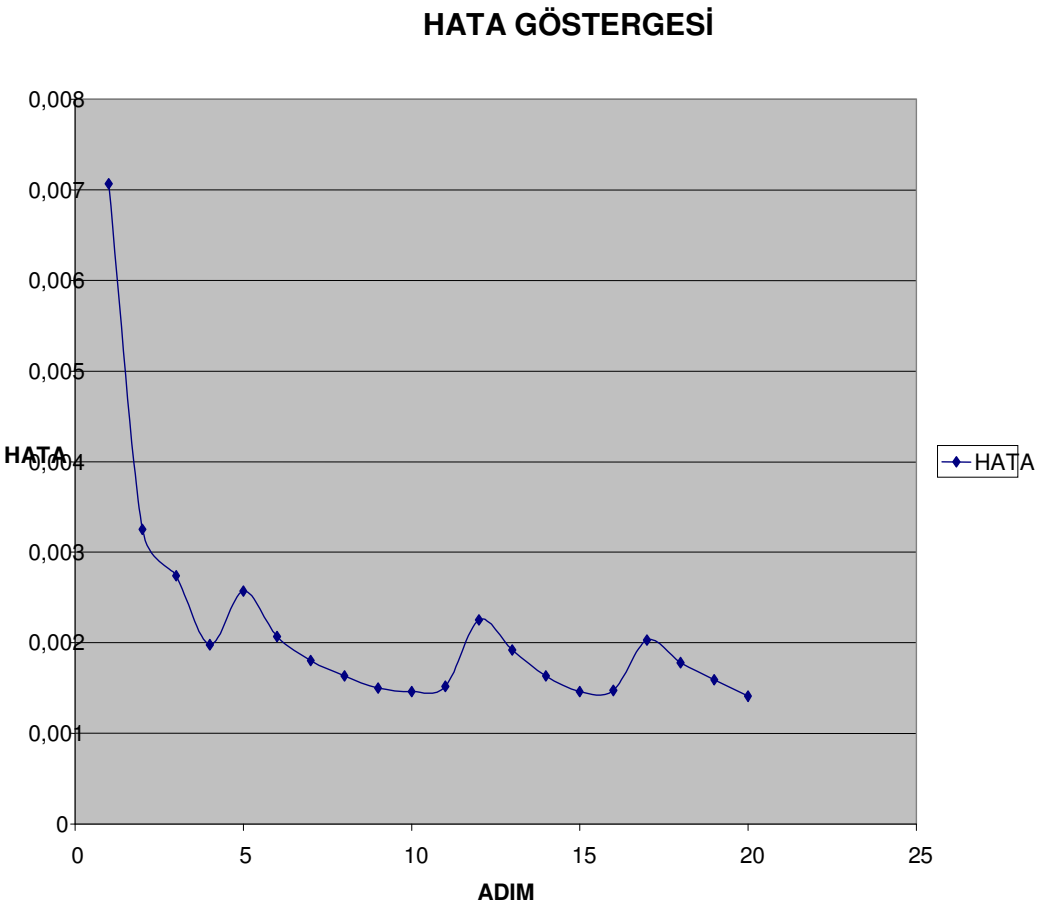
Öğrenme Sayısı : 0,5

Momentum : 0,8

10. 1. 2. SONUÇLAR

ADIM	HATA
1	0,00707
2	0,00325
3	0,00274
4	0,00198
5	0,00257
6	0,00207
7	0,0018
8	0,00163
9	0,0015
10	0,00146
11	0,00152
12	0,00225
13	0,00192
14	0,00163
15	0,00146
16	0,00147
17	0,00203
18	0,00178
19	0,00159
20	0,00141

Tablo 10. 1. Hata Bilgileri



Şekil 10. 1. Hata Bilgileri Grafiği

10. 2. İKİNCİ UYGULAMA GİRİŞLERİ VE SONUÇLARI

10. 2. 1. GİRİŞLER

Giriş Dosyası : c:\dosya01.txt

Çıkış Dosyası : c:\dosya02.txt

Sonucu İstenen Bilgilerin Bulunduğu Dosya : c:\dosya03.txt

Iterasyon Sayısı : 100000

Giriş Katmanındaki İşlem Elemanı Sayısı : 35

Toplam Boyut (Grup Sayısı) : 8

Ara Katmandaki İşlem Elemanı Sayısı : 10

Çıkış Katmanındaki İşlem Elemanı Sayısı : 8

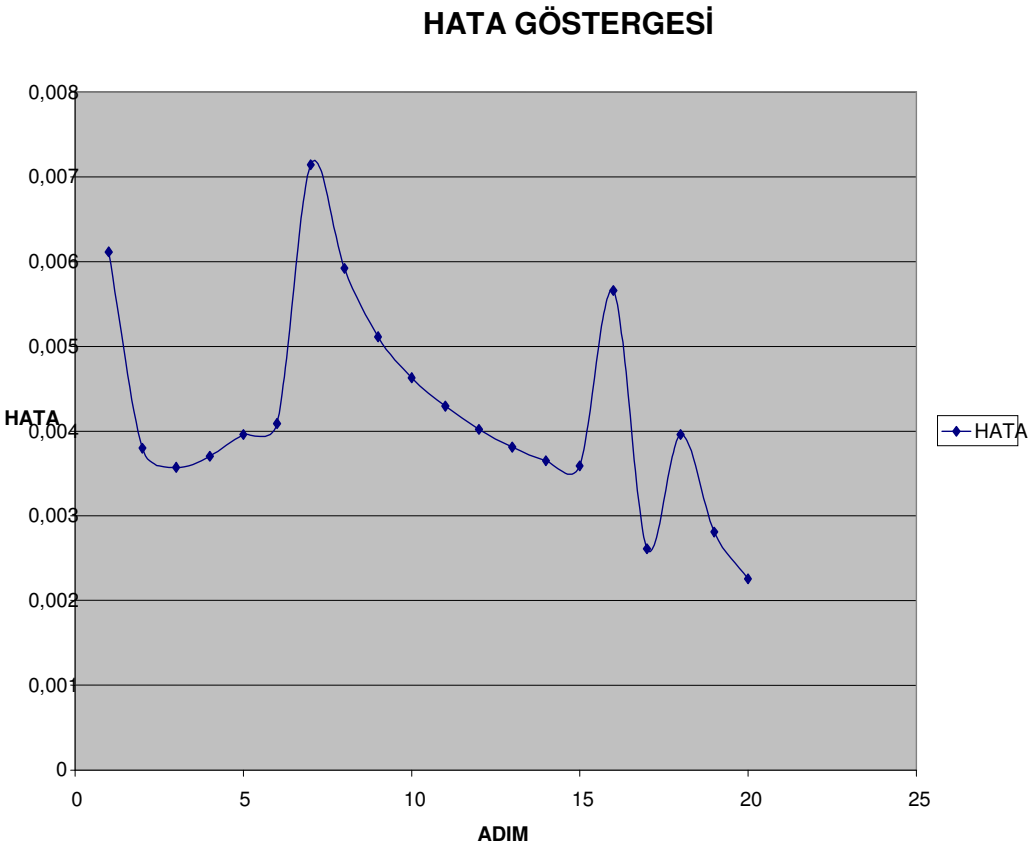
Öğrenme Sayısı : 0,5

Momentum : 0,8

10. 2. 2. SONUÇLAR

ADIM	HATA
1	0,00611
2	0,0038
3	0,00357
4	0,0037
5	0,00396
6	0,00409
7	0,00714
8	0,00592
9	0,00511
10	0,00463
11	0,00429
12	0,00402
13	0,00381
14	0,00365
15	0,00359
16	0,00566
17	0,00261
18	0,00396
19	0,00281
20	0,00226

Tablo 10. 2. Hata Bilgileri



Şekil 10. 2. Hata Bilgileri Grafiği

10. 3. ÜÇÜNCÜ UYGULAMA GİRİŞLERİ VE SONUÇLARI

10. 3. 1. GİRİŞLER

Giriş Dosyası : c:\dosya01.txt

Çıkış Dosyası : c:\dosya02.txt

Sonucu İstenen Bilgilerin Bulunduğu Dosya : c:\dosya03.txt

Iterasyon Sayısı : 100000

Giriş Katmanındaki İşlem Elemanı Sayısı : 35

Toplam Boyut (Grup Sayısı) : 8

Ara Katmandaki İşlem Elemanı Sayısı : 10

Çıkış Katmanındaki İşlem Elemanı Sayısı : 8

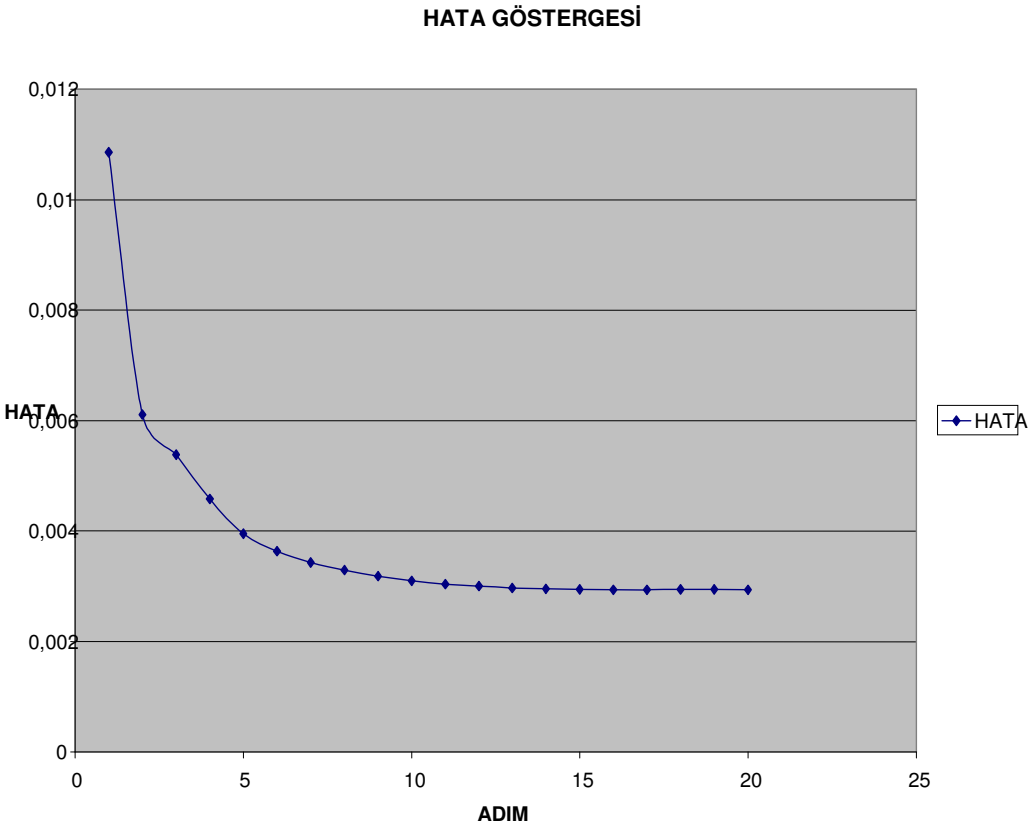
Öğrenme Sayısı : 0,5

Momentum : 0,8

10. 3. 2. SONUÇLAR

ADIM	HATA
1	0,01086
2	0,00611
3	0,00538
4	0,00458
5	0,00395
6	0,00363
7	0,00343
8	0,00329
9	0,00318
10	0,0031
11	0,00304
12	0,003
13	0,00297
14	0,00295
15	0,00294
16	0,00293
17	0,00293
18	0,00294
19	0,00294
20	0,00293

Tablo 10. 3. Hata Bilgileri



Şekil 10. 3. Hata Bilgileri Grafiği

BÖLÜM 11.

11. YAPAY SİNİR AĞLARINA ULAŞIM

11. 1. YAPAY SİNİR AĞI SİMULATÖRLERİ

Bu ana kadar yapay sinir ağları ile ilgili değişik modeller ve endüstriyel uygulamalar ile ilgili bilgiler verildi. Yapay sinir ağlarının bilgisayar programları yazılmış ve birçok modeli çözebilen hazır sistemler geliştirilmiştir. Yapay sinir ağı simulatörü denilebilen bu sistemler kullanıcıdan sadece örnek seti, test setini hazırlamasını ve ağın modelini belirleyerek ilgili modeller için gereken parametre ve sabit değerlerin (öğrenme katsayısı gibi) belirlemesini istemektedir. Böylece sistem problemi çözecek ağı otomatik olarak oluşturmakta ve ağı eğitebilmektedir. Bu sistemler aynı zamanda ağın hatasının dair grafiğini çizerek eğitimin zaman içindeki iyileşmesini de göstermektedir.

Bu anlamda sayısız yapay sinir ağı simulatörü bulmak mümkündür. Bazılarının ulaşım adresleri verilecektir. Sistemler hakkında ilgili web sayfalarından ayrıntılı bilgiler almak mümkün olabilir. Bazı sitelerde ise birden fazla simulatör tanıtılmaktadır.

Örneğin,

http://www.cs.cofc.edu/~manaris/ai-education-repository/neural-n-tools.html

adresinde birkaç simulatör hakkında bilgi bulunabilir…

SİMULATÖR ADI	WEB ADRESİ
neurosolutions	http://www.nd.com
Matlap-Neural Network Tool box	http://www.mathworks.com
Neur DS	ftp://gatekeeper.dec.com/pub/DEC
Mactivation	ftp://ftp.cs.colorado.edu/pub/cs/misc/.
Xerion	ftp://ftp.cs.toronto.edu/pub/xerion/
Brain Wave	http://www.psy.uq.edu.au/~brainwav/Manual/WhatIs.html
PDP++	http://www.cnbc.cmu.edu/PDP++/manual/pdp-user_20.html
Z-Solutions	http://www.zsolutions.com/software.htm
STATSOFT	www.statsoftinc.com

Tablo 11. 1. Örnek Yapay Sinir Ağları Simulatörleri ve Adresleri

11. 2. YAPAY SİNİR AĞLARI BİLGİ KAYNAKLARI

Yapay sinir ağları ile ilgili olarak her yıl çok sayıda sempozyumlar düzenlenmekte kitap ve makaleler yazılmaktadır. Bu konuya olan ilgi her geçen gün daha çok artmaktadır. Yapay sinir ağı hakkında bilgi almak isteyen ve en çok sorulan soruların ve yanıtlarının tutulduğu bir adres vardır. Orada sadece sorulara yanıt verilmemekte aynı zamanda yapılan sempozyumlar, yayınlanan kitaplar, geliştirilen bilgisayar programları vb. gibi konularda da sürekli bilgiler verilmekte ve site güncellemektedir. Bu bilgilere şu adresten ulaşmak mümkündür.

ftp://ftp.sas.com.pub/neural/FAQ.html

Ayrıca IEEE yapay sinir ağları Konsilinin web adresinden de
(http://www.ieeee.org/nnc/index.html) bilgiler almak mümkündür.

Bunların dışında web adreslerinin sayısı da oldukça fazladır. Bu iki adresten istenilen bilgileri içeren değişik web adreslerine ulaşmak mümkündür.

SONUÇ VE ÖNERİLER

Yapay sinir ağları yapısı kavramı kullanılarak bu tez çalışması hazırlanmıştır. Bir akış yapısı göz önüne alındığında;

1) C#'da yazılmış olan program çalıştırılır.
   (Microsoft Visual Studio 2005)'de …
   Programın çalışması aşamasında ağırlıklar rasgele üretilir. Ağ eğitilinceye kadar ağırlıklar hesaplanır ve belirlenmiş iterasyon sayısına ulaşıldığında programın çalışmasının ilk aşaması tamamlanmış olur.
2) Türk Alfabesi'n deki 29 büyük harf ve 29 küçük harf programa , programın içinde bir dosya olarak verilmiştir. Bu yapı dosya içinde bir matris olarak (58 x 35) yer almaktadır.
3) Örneğin herhangi bir harfin programdaki karşılığını almak için
   3.1) Harfin fotoğrafı çekilir.
   3.2) Fotoğraf bir matris yapısı haline getirilir. (7 x 5) olarak. Bunun için ilgili bir program kullanılabilir Matlab gibi…
   3.3) Programın çalışması aşamasında oluşturulan matris programa giriş dosyası olarak verilir.
   3.4) Program giriş matrisini aldıktan sonra ilgili adımların çalışmasının ardından karşılık gelen harf bulunur.

Programın çalışması başka bir deyiş ile ağın eğitilmesi ;

İterasyon sayısı = 5000 olarak alındığında (58 harf için) yaklaşık 10 dakika kadar sürmektedir. Daha hassas sonuçlara ulaşmak ve çözüme daha fazla yaklaşabilmek için iterasyon sayısının daha büyük alınması gerekir. Bu durumda programın çalışma süresi kuşkusuz daha da uzayacaktır. Bu aşamada daha hızlı CPU (Central processing unit – Merkezi işlem birimi) olan bir bilgisayarın kullanılması tavsiye edilir.

Karakter algılamaya yönelik olarak hazırlanmış bu program bir çok değişik amaç için kullanılabilir. Örneğin ;

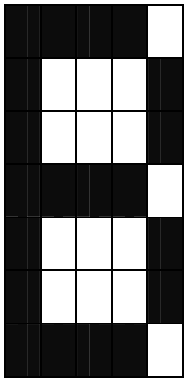
- Bildiğimiz karakterlerin tek tek algılanması
- Araba plakalarının algılanması ve tanınması
- Özellikle çocuklarda zeka gelişimine katkıda bulunabilmek için uygulanması
- Gizlilik gerektiren askeri veya sivil uygulamalarda eksik karakteri tanıma

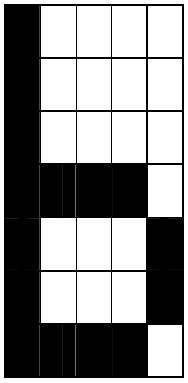
- Evrenin değişik koordinatlarından gelen sinyallerin, dünyadan uzaya yayılan sinyallerle bir veri seti olarak alınması, çıkış seti olarak da dünyanın referans olarak alınması ve dünya dışında başka gök cisimlerinde yaşam bulunup bulunmadığının araştırılması.
- Fizik bilimi konusu olarak göz önüne alındığında; değişik basınç altında çeşitli metallerin deneysel sonuçlarının girdi ve çıktı seti olarak alınması ve deneysel olarak sonuçları belli olmayan çözümlere ulaşılması.
- Daha önce yapılmış olan nükleer çalışmaların hangi etki alanlarında ne kadar tahribat yaptığının girdi ve çıktı seti olarak alınması ve daha yüksek etkili olarak yapılacak çalışmaların etki alanlarının ve etki büyüklüklerinin belirlenmesi.

Burada birkaç uygulama alanından çok basit olarak söz edilmiştir. Programa yapılacak ilave kodlar ile program daha da geliştirilerek örüntü tanır hale de getirilebilir. Bu durumda çok daha geniş kullanım amaçlı olarak bir çok alanda verimli ve bilimsel çalışmalar yapılabilir…

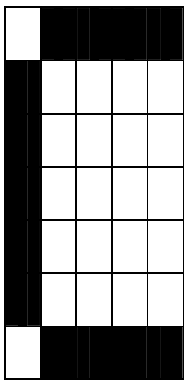
EK – 1

(KARAKTERLERİN MATRİS YAPISINA UYARLANMIŞ HALİ)
(TÜRK ALFABESİ'NDE Kİ BÜYÜK VE KÜÇÜK HARFLER)

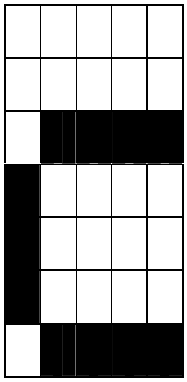




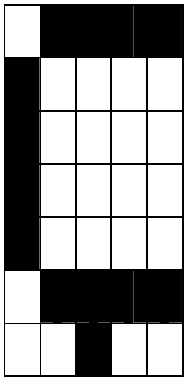
11110	10000
10001	10000
10001	10000
11110	11110
10001	10001
10001	10001
11110	11110



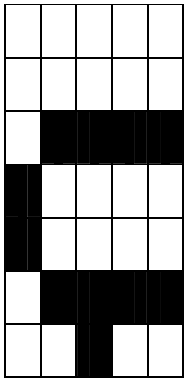
01111
10000
10000
10000
10000
10000
01111



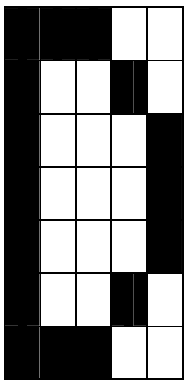
00000
00000
01111
10000
10000
10000
01111



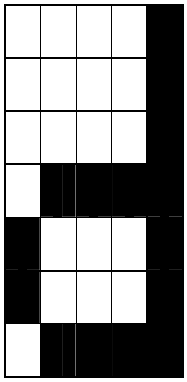
01111
10000
10000
10000
10000
01111
00100



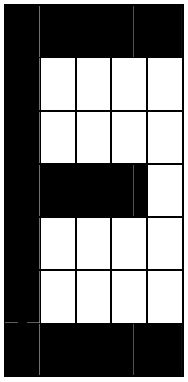
00000
00000
01111
10000
10000
01111
00100



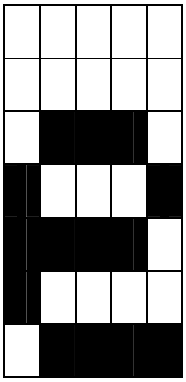
11100
10010
10001
10001
10001
10010
11100



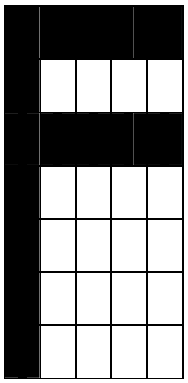
00001
00001
00001
01111
10001
10001
01111



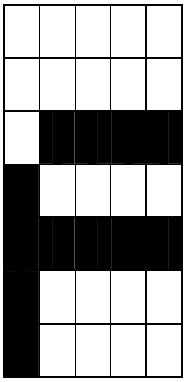
11111
10000
10000
11110
10000
10000
11111



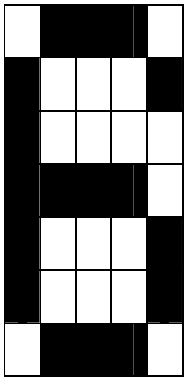
00000
00000
01110
10001
11110
10000
01111



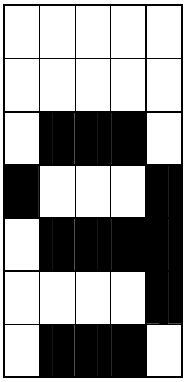
11111
10000
11111
10000
10000
10000
10000



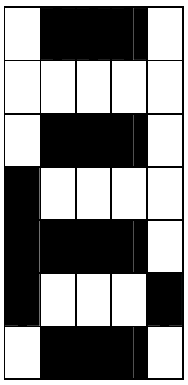
00000
00000
01111
10000
11111
10000
10000



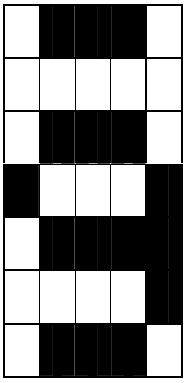
01110
10001
10000
11110
10001
10001
01110



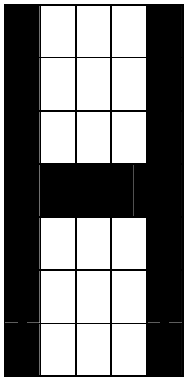
00000
00000
01110
10001
01111
00001
01110



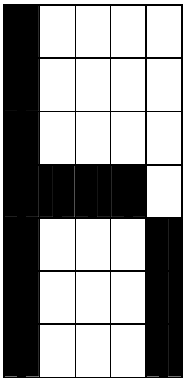
01110
00000
01110
10000
11110
10001
01110



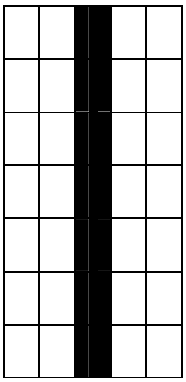
01110
00000
01110
10001
01111
00001
01110



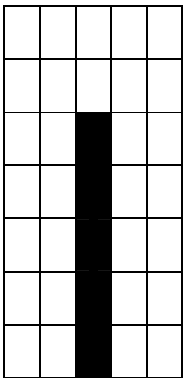
10001
10001
10001
11111
10001
10001
10001



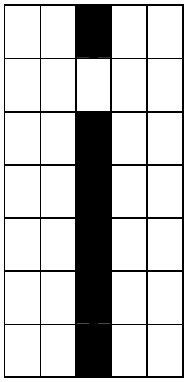
10000
10000
10000
11110
10001
10001
10001



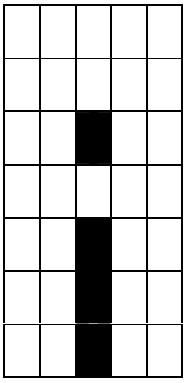
00100
00100
00100
00100
00100
00100
00100



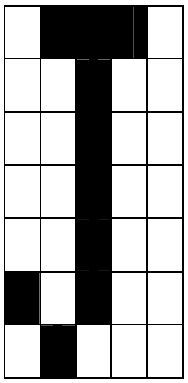
00000
00000
00100
00100
00100
00100
00100



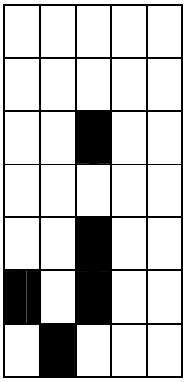
00100
00000
00100
00100
00100
00100
00100



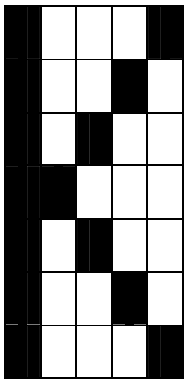
00000
00000
00100
00000
00100
00100
00100



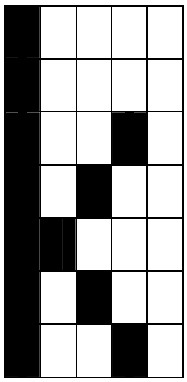
01110
00100
00100
00100
00100
10100
01000



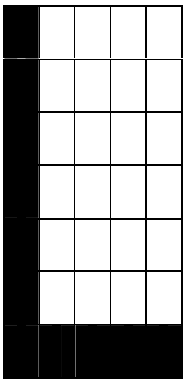
00000
00000
00100
00000
00100
10100
01000



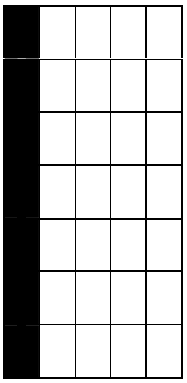
10001
10010
10100
11000
10100
10010
10001



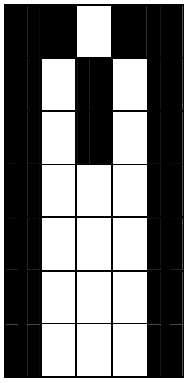
10000
10000
10010
10100
11000
10100
10010



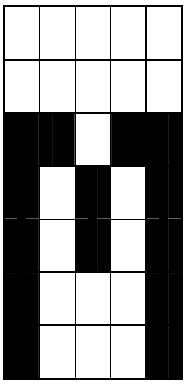
10000
10000
10000
10000
10000
10000
11111



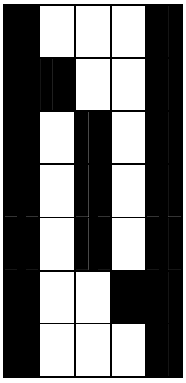
10000
10000
10000
10000
10000
10000
10000



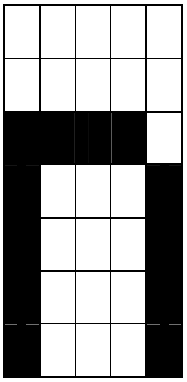
11011
10101
10101
10001
10001
10001
10001



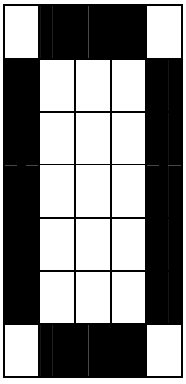
00000
00000
11011
10101
10101
10001
10001



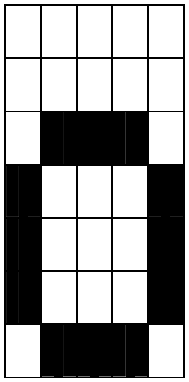
10001
11001
10101
10101
10101
10011
10001



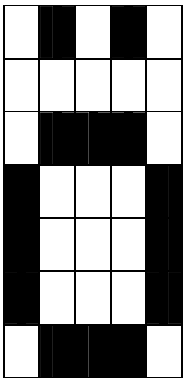
00000
00000
11110
10001
10001
10001
10001



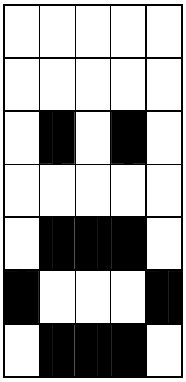
01110
10001
10001
10001
10001
10001
01110



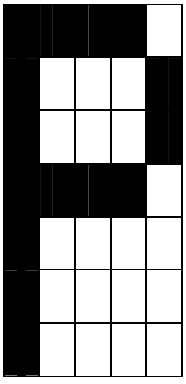
00000
00000
01110
10001
10001
10001
01110



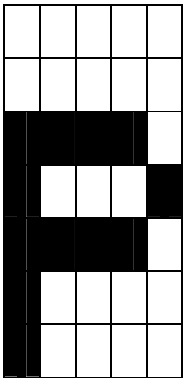
01010
00000
01110
10001
10001
10001
01110



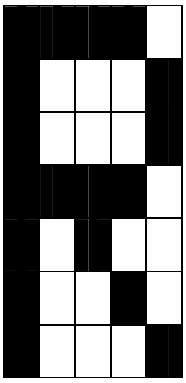
00000
00000
01010
00000
01110
10001
01110



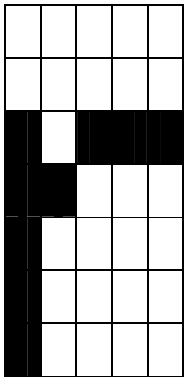
11110
10001
10001
11110
10000
10000
10000



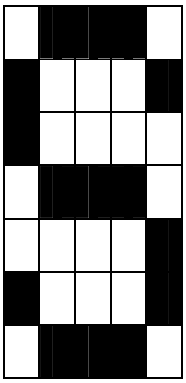
00000
00000
11110
10001
11110
10000
10000



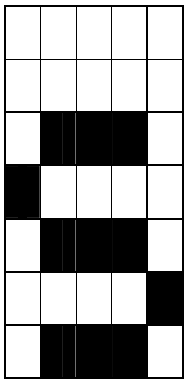
11110
10001
10001
11110
10100
10010
10001



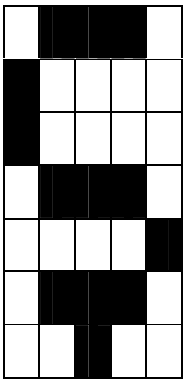
00000
00000
10111
11000
10000
10000
10000



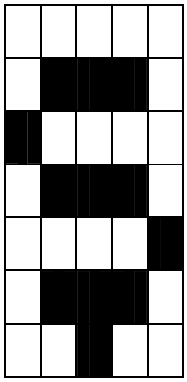
01110
10001
10000
01110
00001
10001
01110



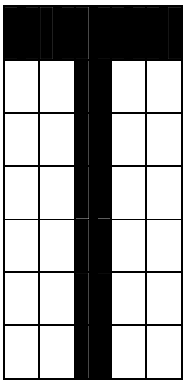
00000
00000
01110
10000
01110
00001
01110



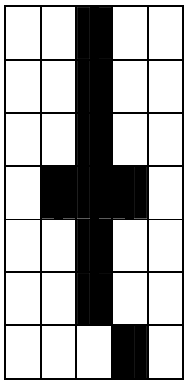
01110
10000
10000
01110
00001
01110
00100



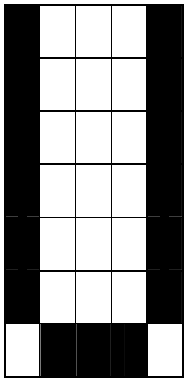
00000
01110
10000
01110
00001
01110
00100



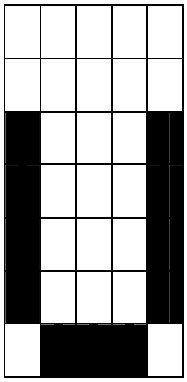
11111
00100
00100
00100
00100
00100
00100



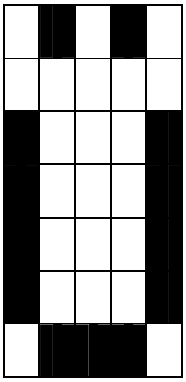
00100
00100
00100
01110
00100
00100
00010



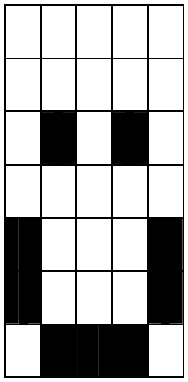
10001
10001
10001
10001
10001
10001
01110



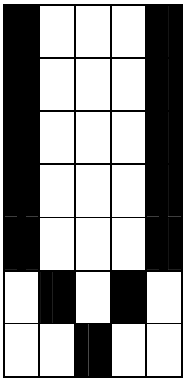
00000
00000
10001
10001
10001
10001
01110



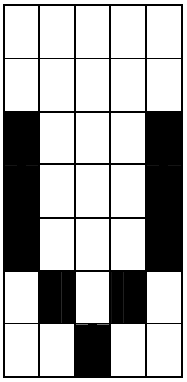
01010
00000
10001
10001
10001
10001
01110



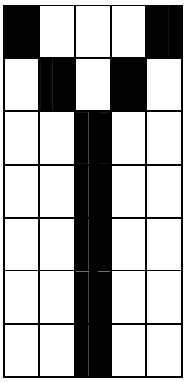
00000
00000
01010
00000
10001
10001
01110



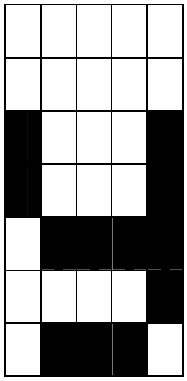
10001
10001
10001
10001
10001
01010
00100



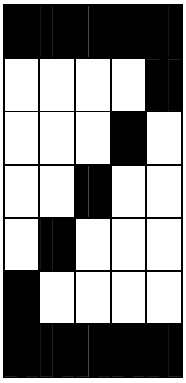
00000
00000
10001
10001
10001
01010
00100



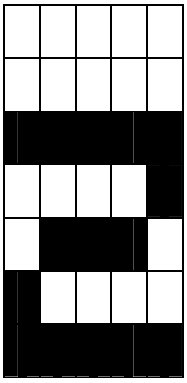
10001
01010
00100
00100
00100
00100
00100



00000
00000
10001
10001
01111
00001
01110



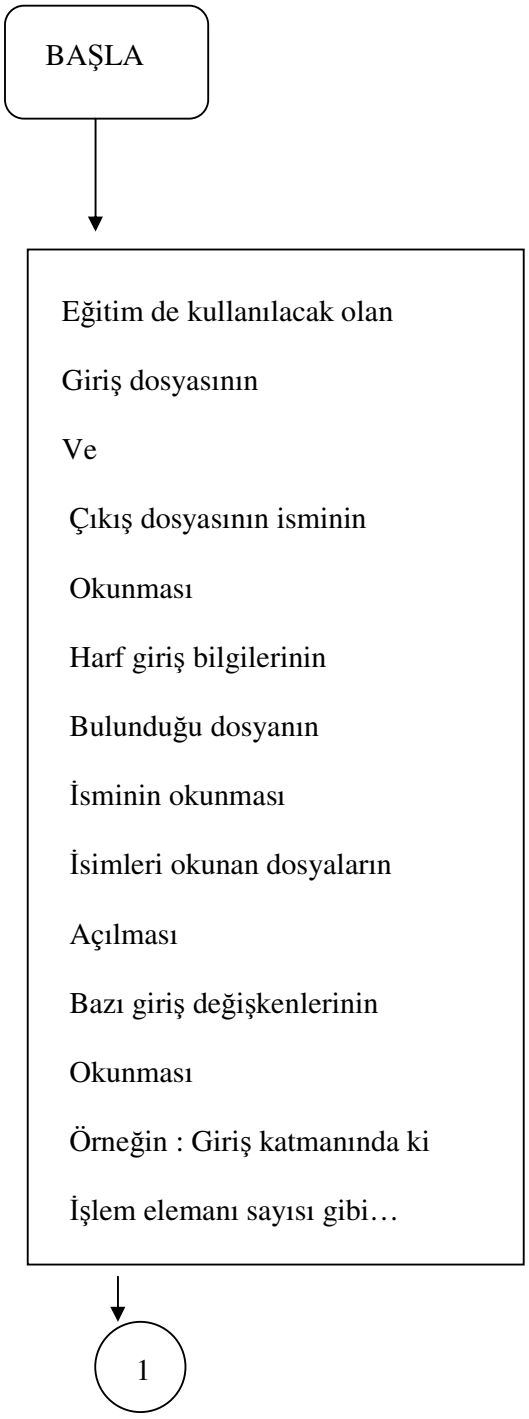
11111
00001
00010
00100
01000
10000
11111

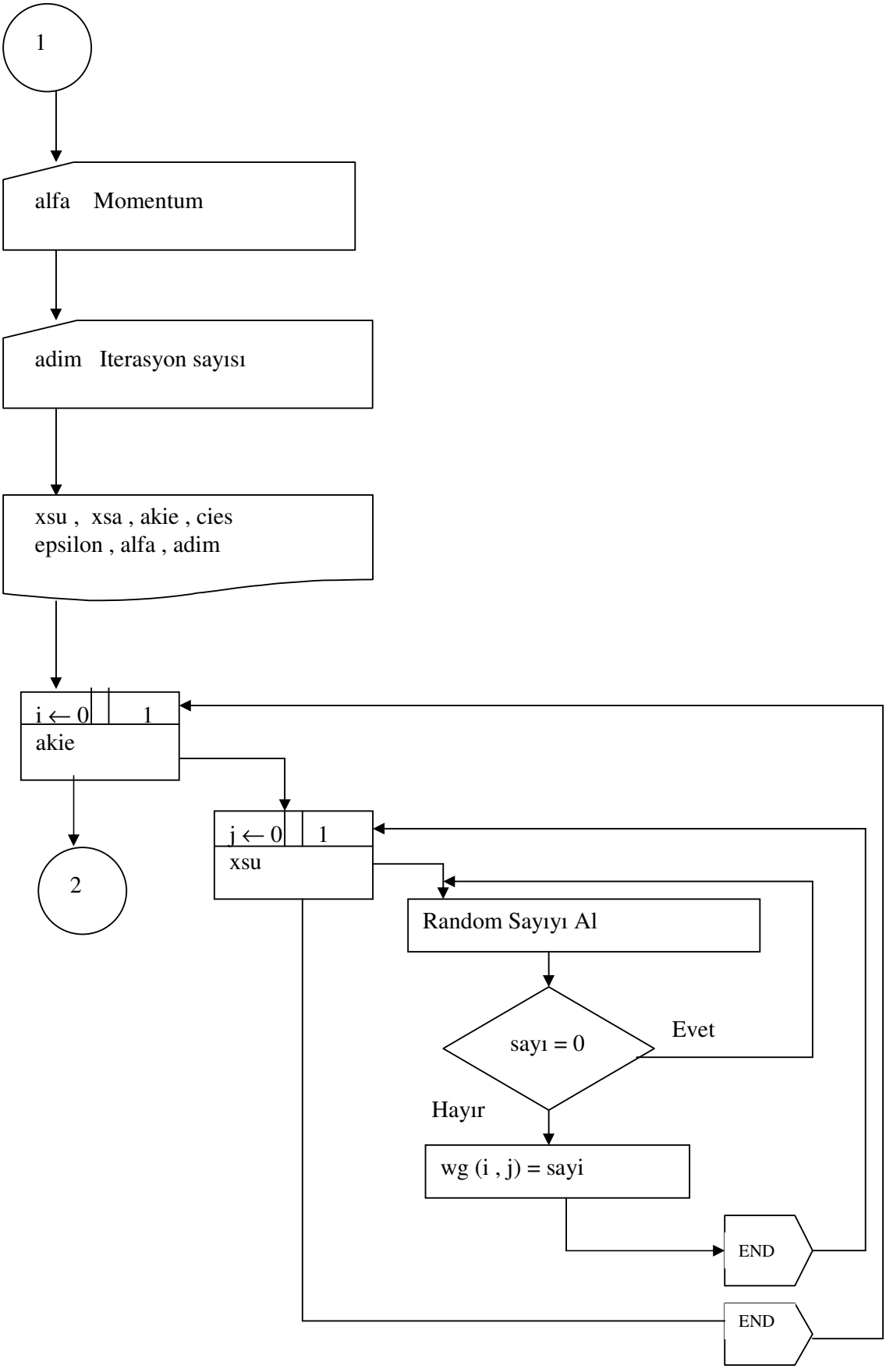


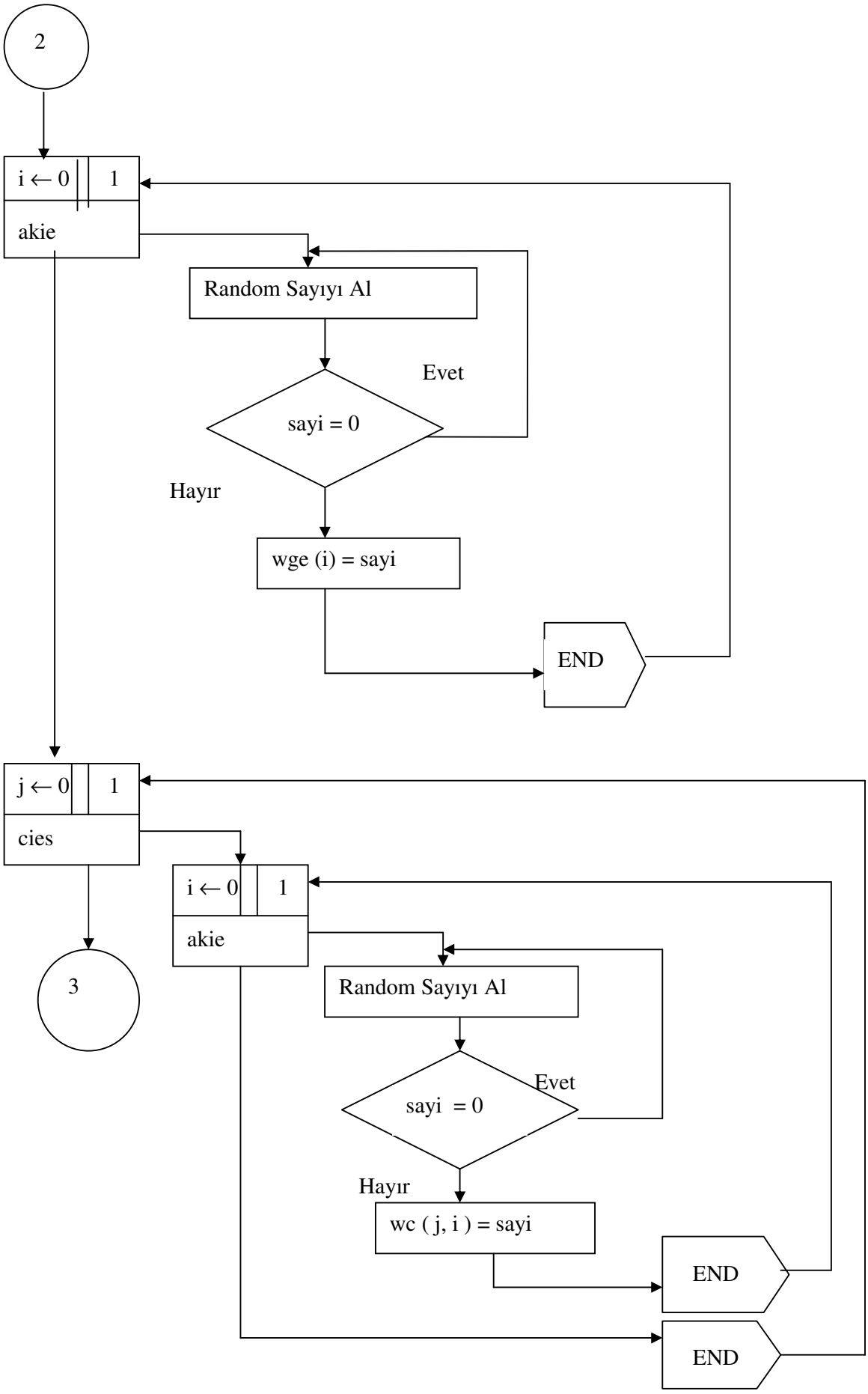
00000
00000
11111
00001
01110
10000
11111

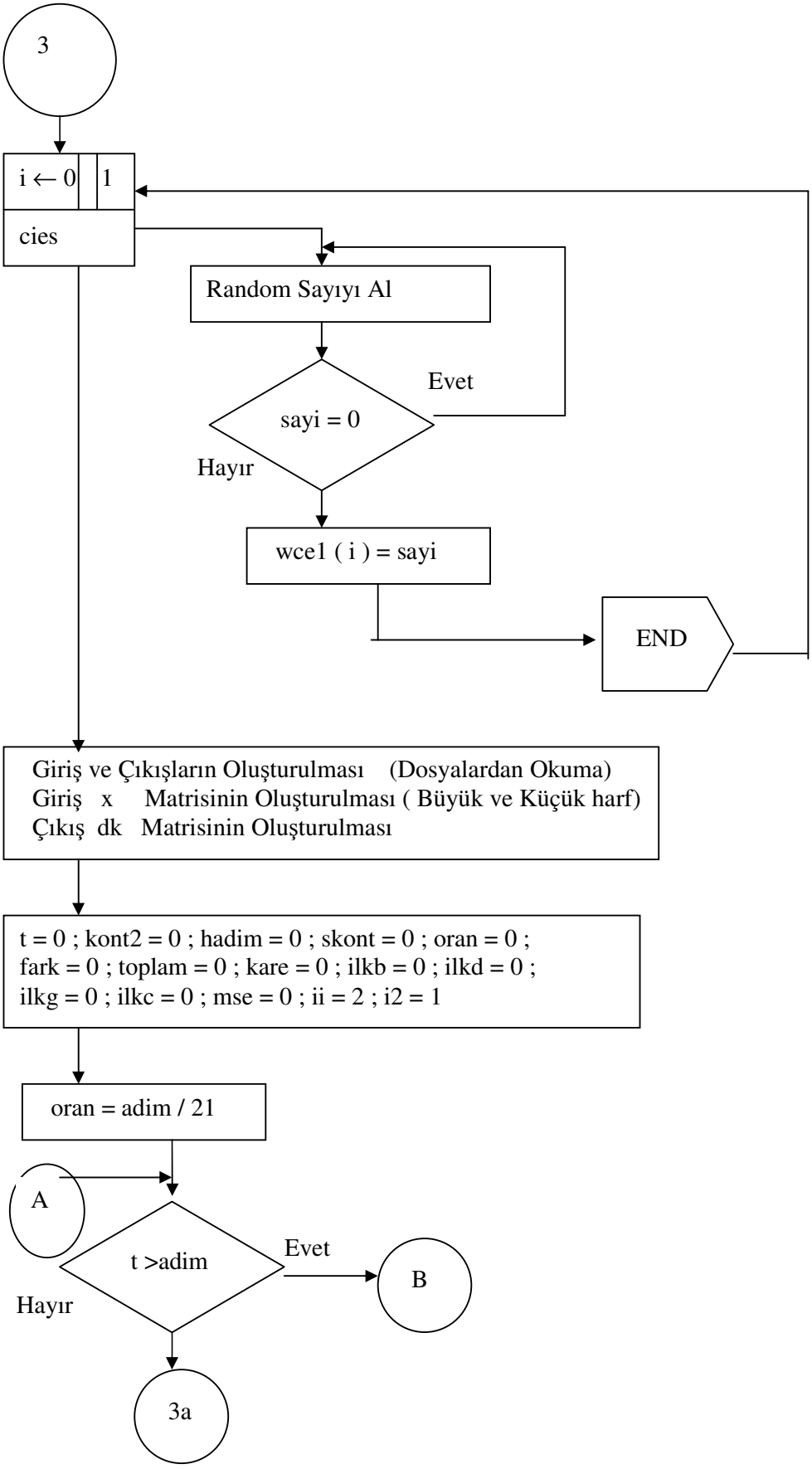
EK – 2

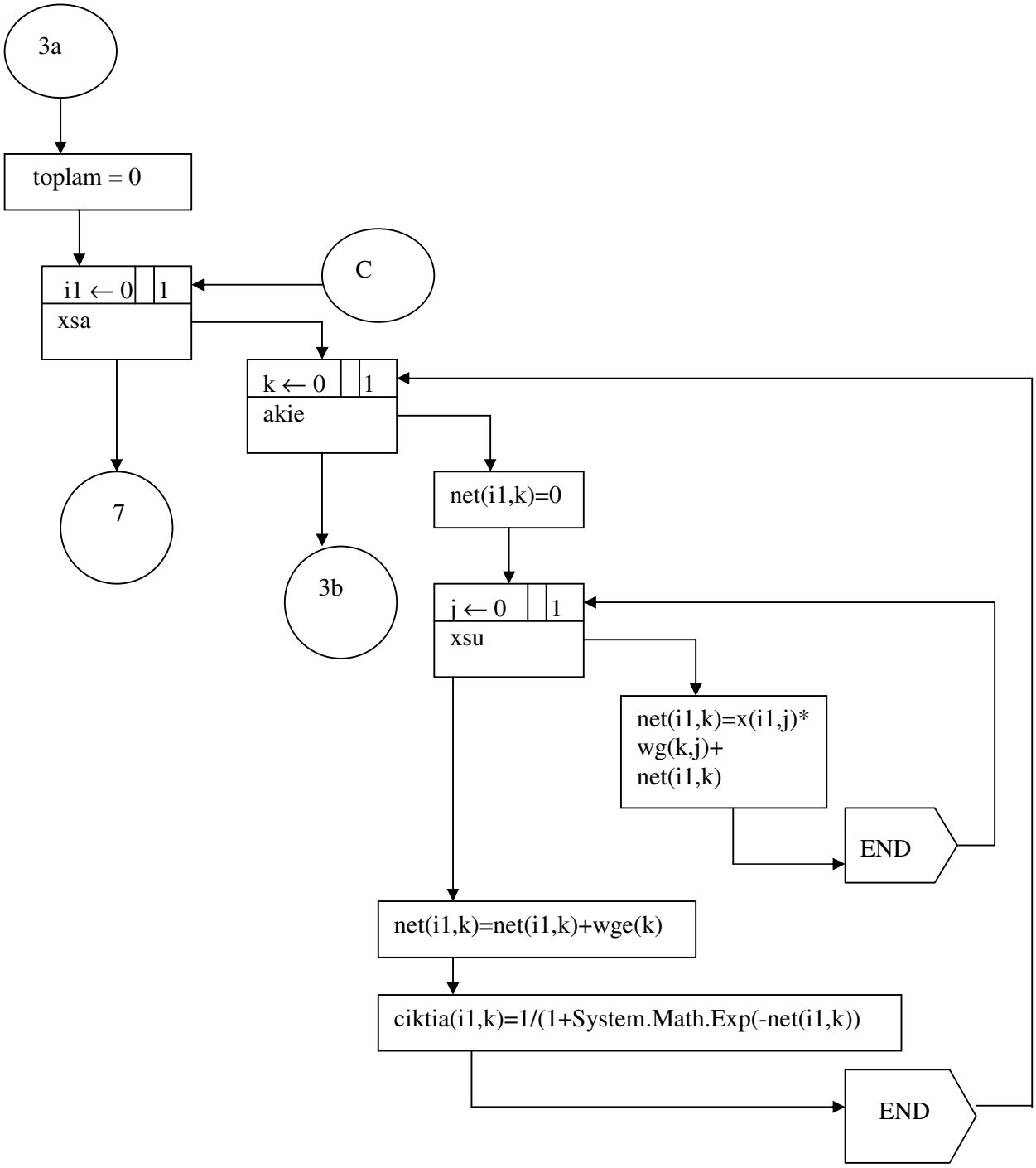
C#' DA YAZILMIŞ YAPAY SİNİR AĞI PROGRAMININ AYRINTILI AKIŞ ŞEMASI

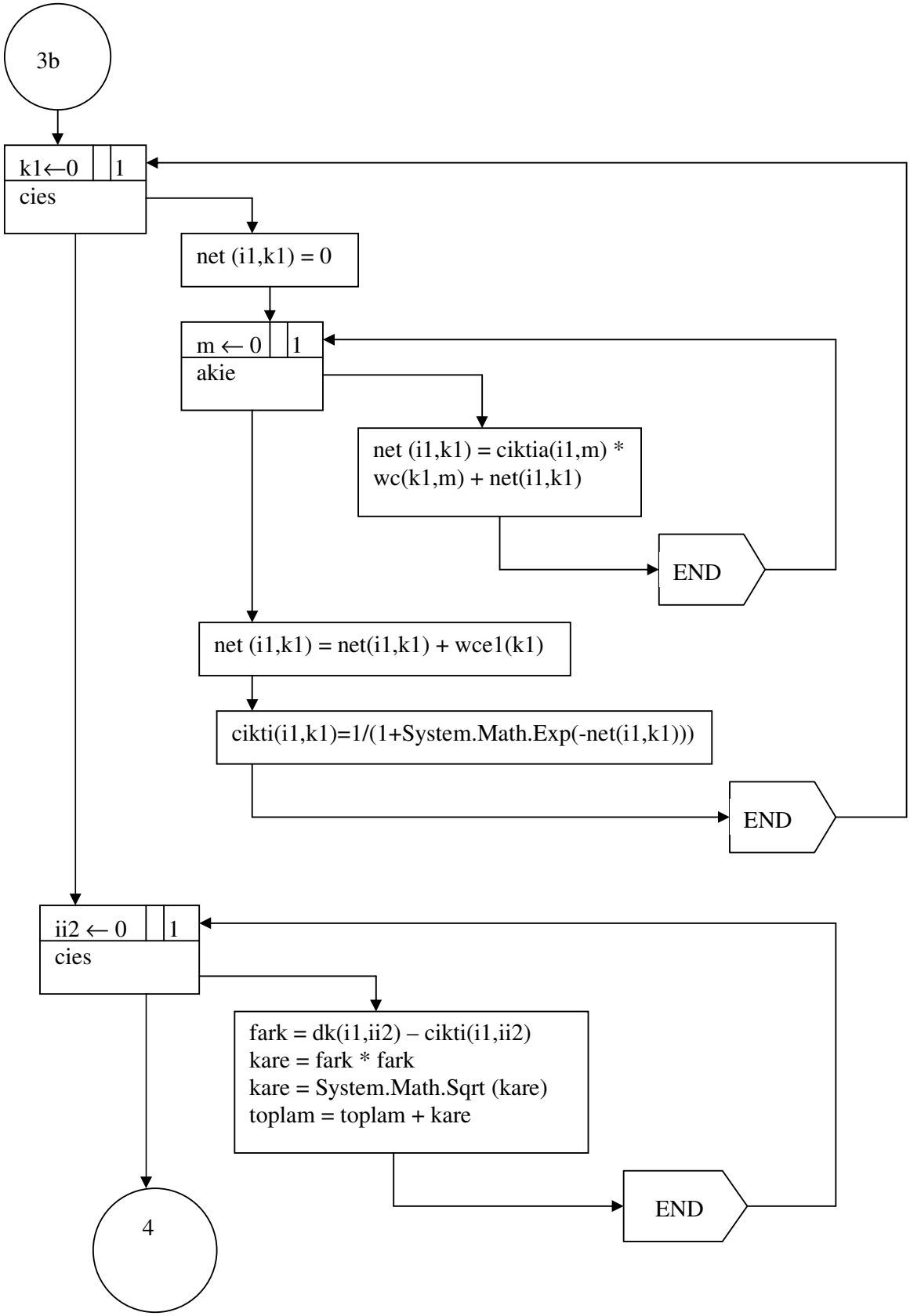


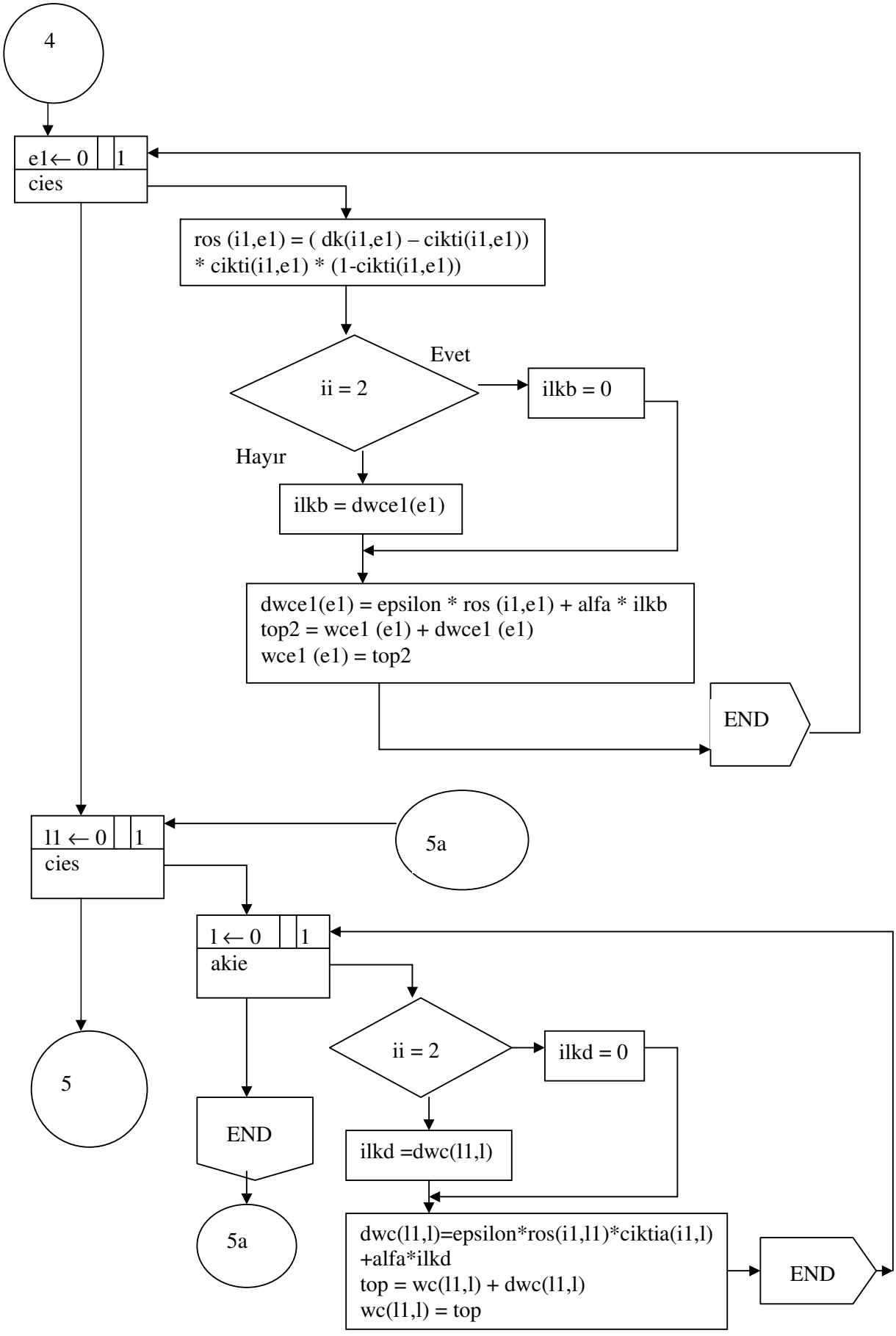

1
alfa Momentum
adim Iterasyon sayısı
xsu , xsa , akie , cies
epsilon , alfa , adim
i ← 0
1
akie
2
j ← 0
1
xsu
Random Sayıyı Al
sayı = 0
Evet
Hayır
wg (i , j) = sayi
END
END

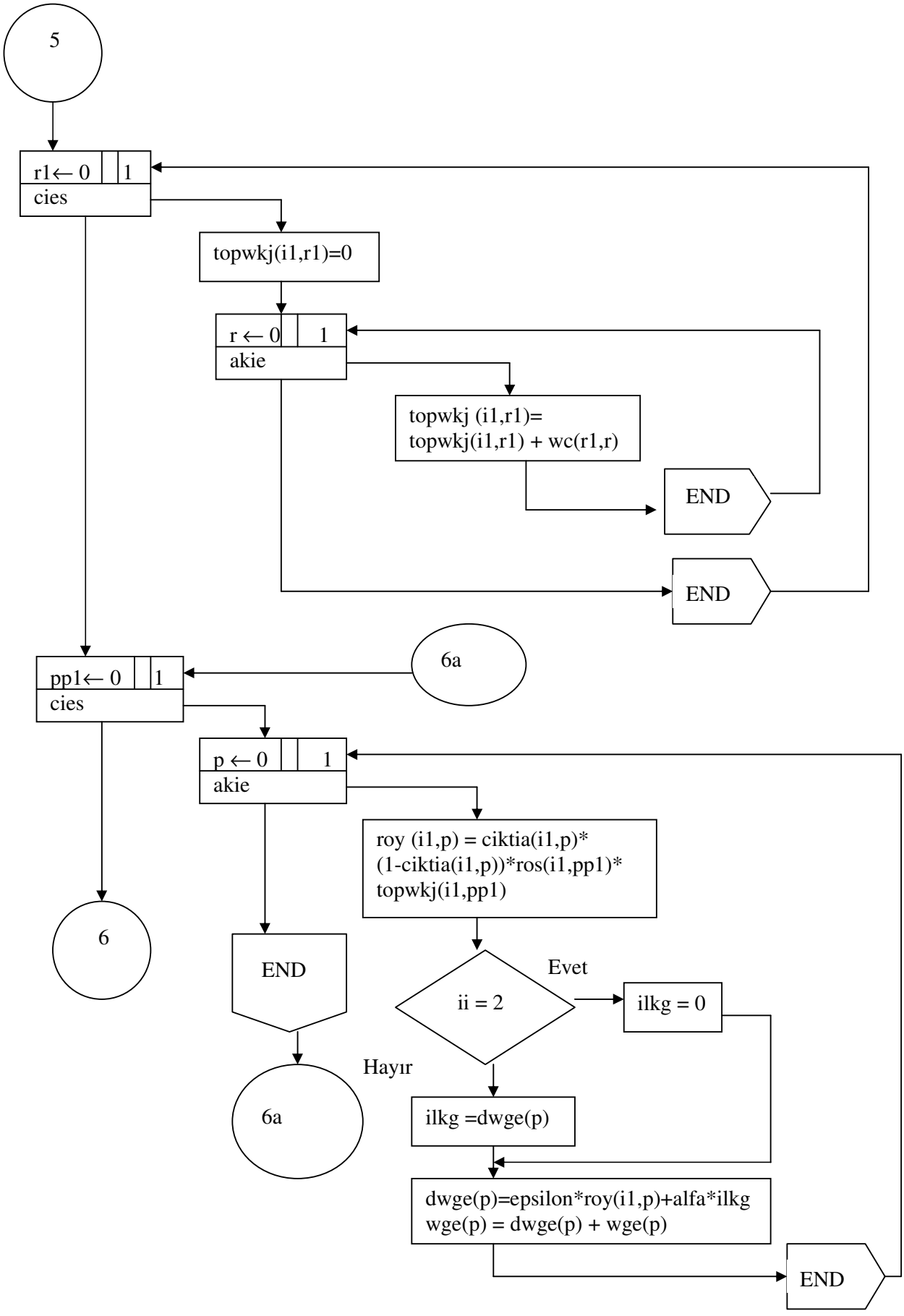

2
i ← 0
1
akie
Random Sayıyı Al
Evet
sayi = 0
Hayır
wge (i) = sayi
END
j ← 0
1
cies
i ← 0
1
akie
3
Random Sayıyı Al
Evet
sayi = 0
Hayır
wc (j, i) = sayi
END
END

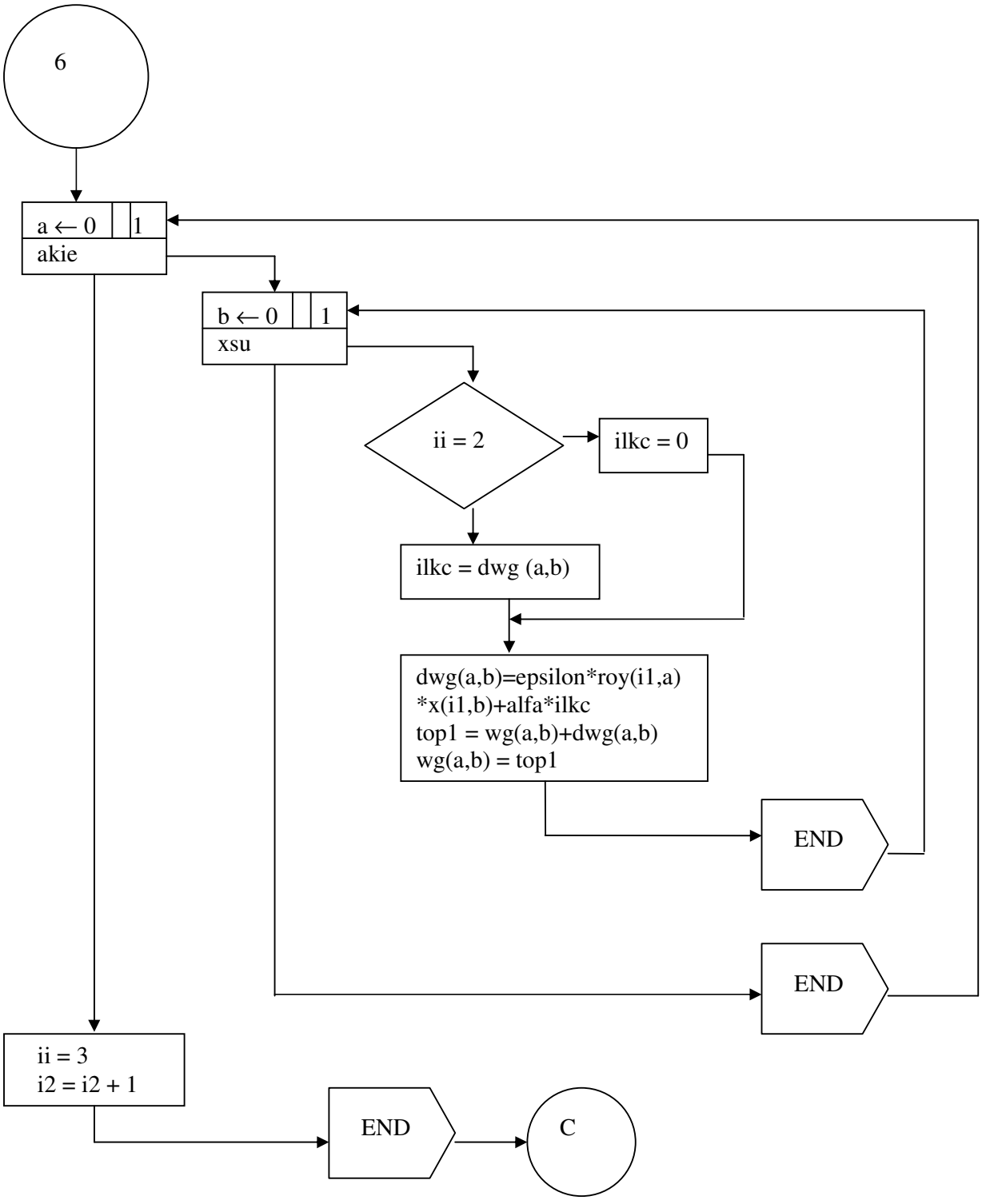

3
i ← 0
1
cies
Random Sayıyı Al
sayi = 0
Evet
Hayır
wce1 (i) = sayi
END
Giriş ve Çıkışların Oluşturulması (Dosyalardan Okuma)
Giriş x Matrisinin Oluşturulması (Büyük ve Küçük harf)
Çıkış dk Matrisinin Oluşturulması
t = 0 ; kont2 = 0 ; hadim = 0 ; skont = 0 ; oran = 0 ;
fark = 0 ; toplam = 0 ; kare = 0 ; ilkb = 0 ; ilkd = 0 ;
ilkg = 0 ; ilkc = 0 ; mse = 0 ; ii = 2 ; i2 = 1
oran = adim / 21
A
t >adim
Evet
Hayır
B
3a

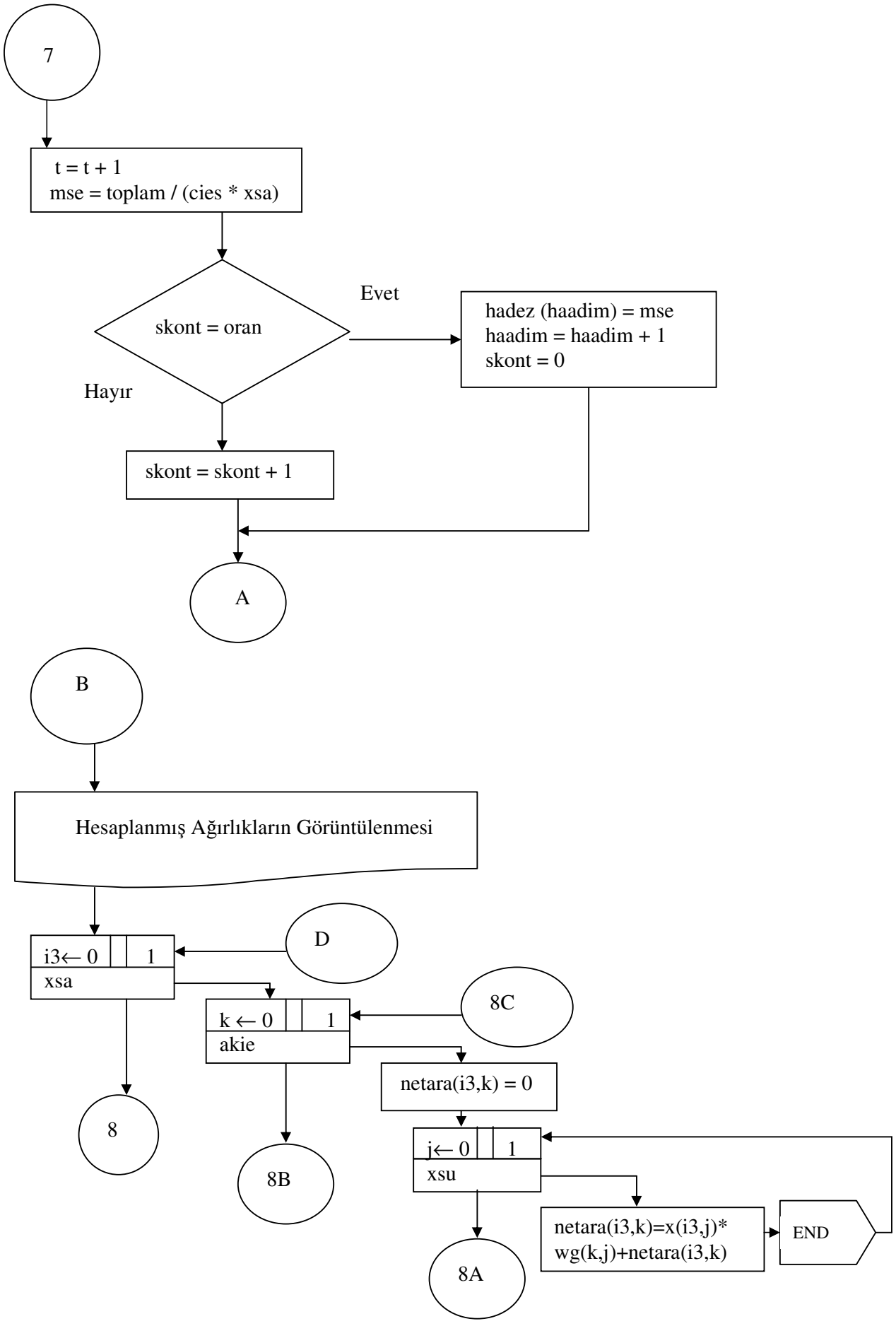

3a
toplam = 0
C
i1 ← 0
1
xsa
k ← 0
1
akie
7
3b
net(i1,k)=0
j ← 0
1
xsu
net(i1,k)=x(i1,j)*
wg(k,j)+
net(i1,k)
END
net(i1,k)=net(i1,k)+wge(k)
ciktia(i1,k)=1/(1+System.Math.Exp(-net(i1,k))
END

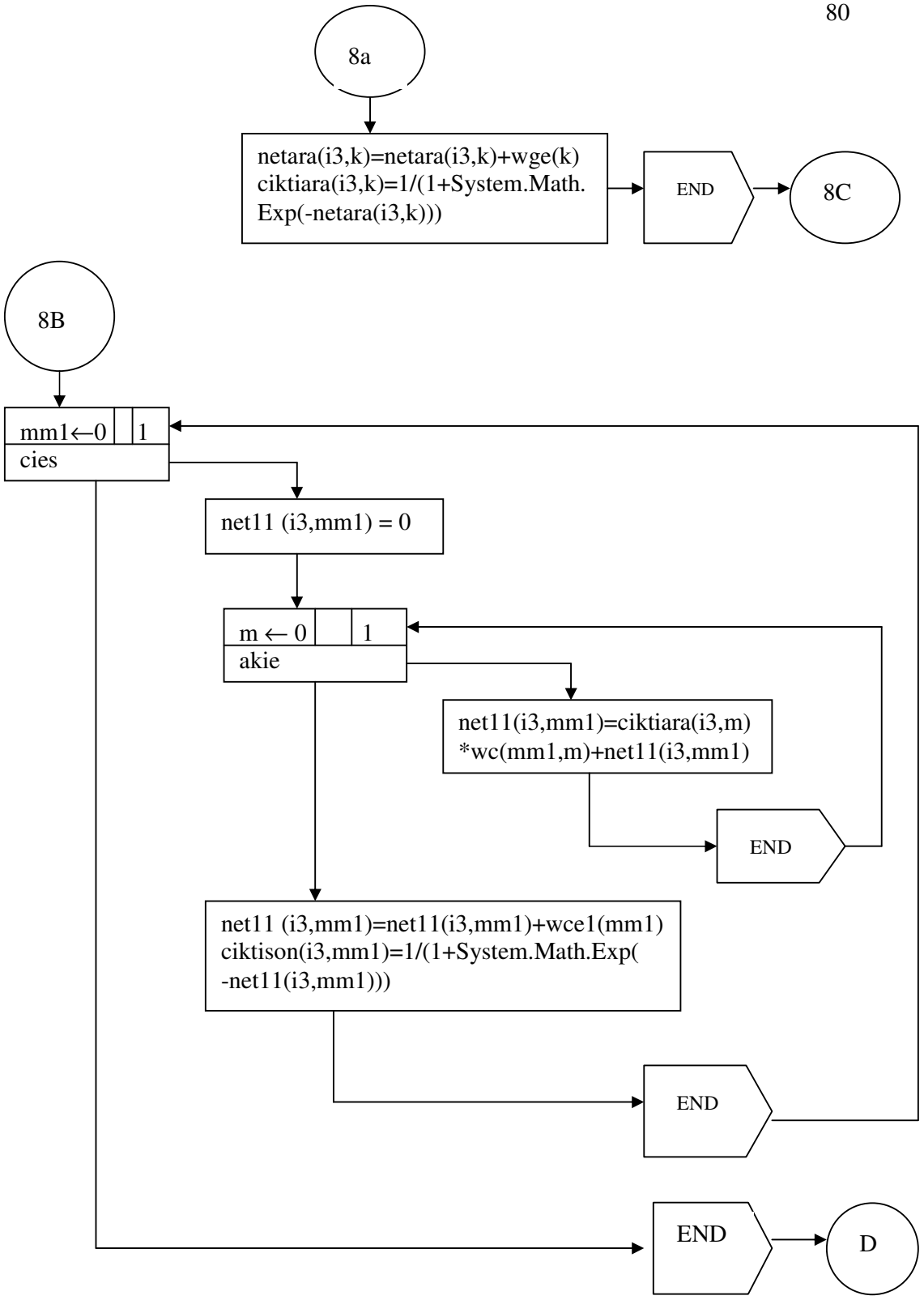

3b
k1←0
1
cies
net (i1,k1) = 0
m ← 0
1
akie
net (i1,k1) = ciktia(i1,m) *
wc(k1,m) + net(i1,k1)
END
net (i1,k1) = net(i1,k1) + wce1(k1)
cikti(i1,k1)=1/(1+System.Math.Exp(-net(i1,k1)))
END
ii2 ← 0
1
cies
fark = dk(i1,ii2) – cikti(i1,ii2)
kare = fark * fark
kare = System.Math.Sqrt (kare)
toplam = toplam + kare
END
4

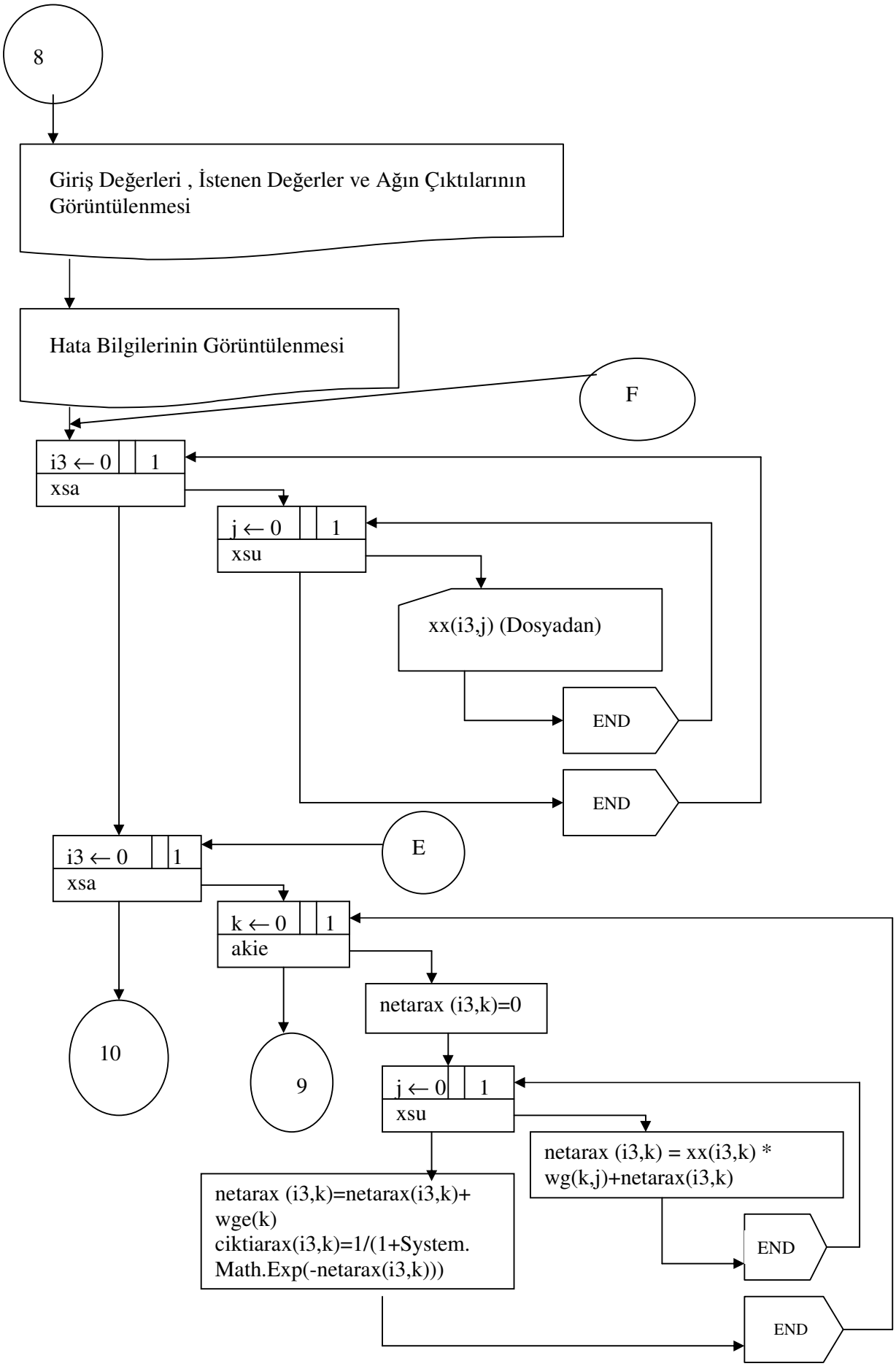

4
e1← 0
1
cies
ros (i1,e1) = (dk(i1,e1) – cikti(i1,e1))
* cikti(i1,e1) * (1-cikti(i1,e1))
ii = 2
Evet
ilkb = 0
Hayır
ilkb = dwce1(e1)
dwce1(e1) = epsilon * ros (i1,e1) + alfa * ilkb
top2 = wce1 (e1) + dwce1 (e1)
wce1 (e1) = top2
END
l1 ← 0
1
cies
5a
l ← 0
1
akie
5
END
ii = 2
ilkd = 0
ilkd =dwc(l1,l)
5a
dwc(l1,l)=epsilon*ros(i1,l1)*ciktia(i1,l)
+alfa*ilkd
top = wc(l1,l) + dwc(l1,l)
wc(l1,l) = top
END

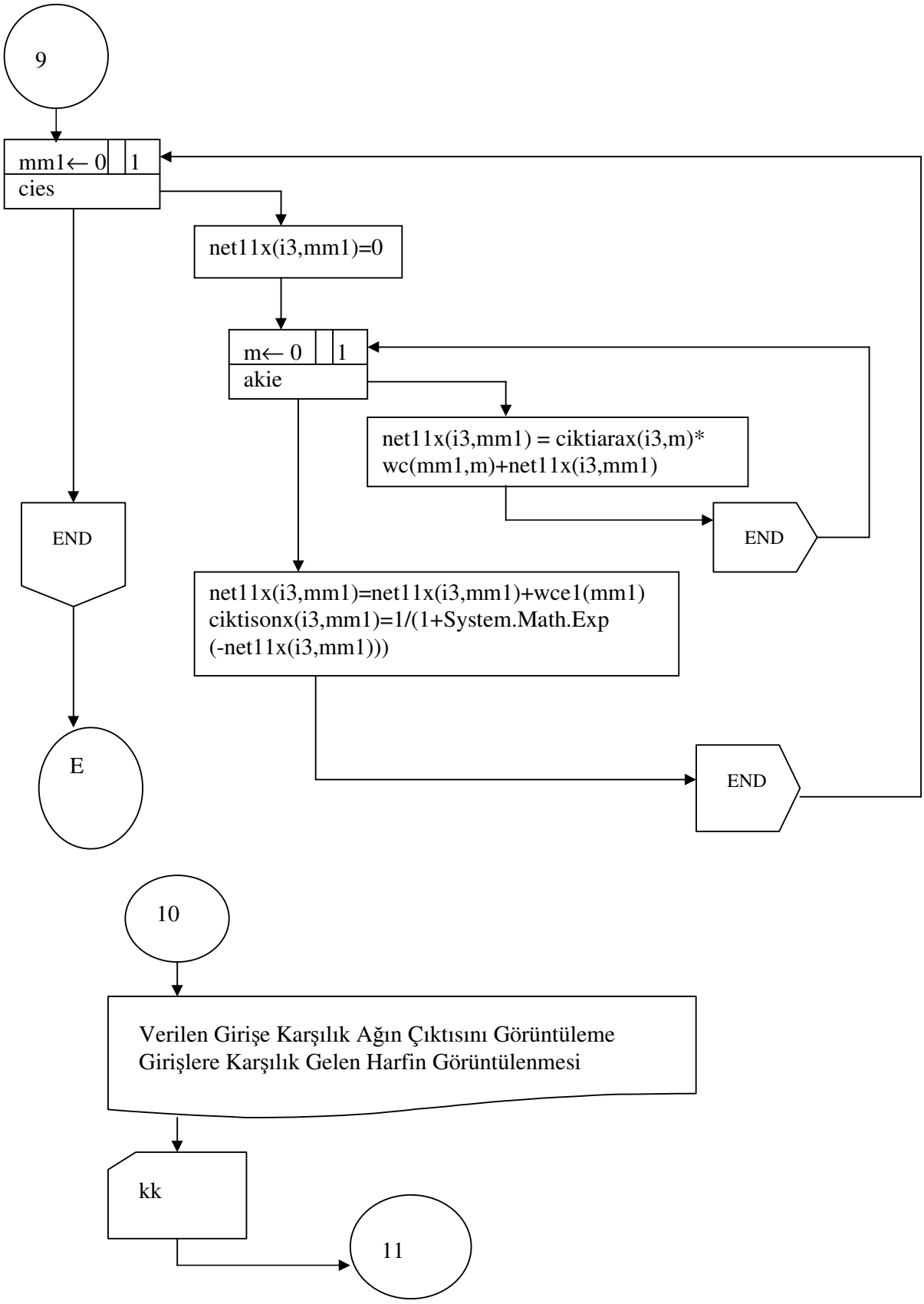

5
r1← 0
1
cies
topwkj(i1,r1)=0
r ← 0
1
akie
topwkj (i1,r1)=
topwkj(i1,r1) + wc(r1,r)
END
END
6a
pp1← 0
1
cies
p ← 0
1
akie
roy (i1,p) = ciktia(i1,p)*
(1-ciktia(i1,p))*ros(i1,pp1)*
topwkj(i1,pp1)
6
END
Evet
ii = 2
ilkg = 0
Hayır
6a
ilkg =dwge(p)
dwge(p)=epsilon*roy(i1,p)+alfa*ilkg
wge(p) = dwge(p) + wge(p)
END

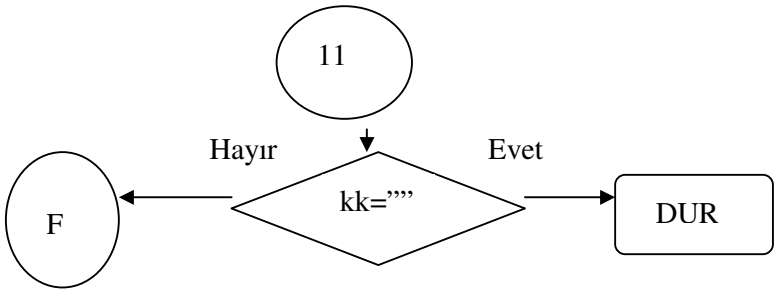

6
a ← 0
1
akie
b ← 0
1
xsu
ii = 2
ilkc = 0
ilkc = dwg (a,b)
dwg(a,b)=epsilon*roy(i1,a)
*x(i1,b)+alfa*ilkc
top1 = wg(a,b)+dwg(a,b)
wg(a,b) = top1
END
END
ii = 3
i2 = i2 + 1
END
C


7
t = t + 1
mse = toplam / (cies * xsa)
skont = oran
Evet
hadez (haadim) = mse
haadim = haadim + 1
skont = 0
Hayır
skont = skont + 1
A
B
Hesaplanmış Ağırlıkların Görüntülenmesi
i3← 0
1
xsa
D
k ← 0
1
akie
8C
netara(i3,k) = 0
j← 0
1
xsu
8
8B
8A
netara(i3,k)=x(i3,j)*
wg(k,j)+netara(i3,k)
END


8a
netara(i3,k)=netara(i3,k)+wge(k)
ciktiara(i3,k)=1/(1+System.Math.
Exp(-netara(i3,k)))
END
8C
8B
mm1←0
1
cies
net11 (i3,mm1) = 0
m ← 0
1
akie
net11(i3,mm1)=ciktiara(i3,m)
*wc(mm1,m)+net11(i3,mm1)
END
net11 (i3,mm1)=net11(i3,mm1)+wce1(mm1)
ciktison(i3,mm1)=1/(1+System.Math.Exp(
-net11(i3,mm1)))
END
END
D


8
Giriş Değerleri , İstenen Değerler ve Ağın Çıktılarının Görüntülenmesi
Hata Bilgilerinin Görüntülenmesi
F
i3 ← 0
1
xsa
j ← 0
1
xsu
xx(i3,j) (Dosyadan)
END
END
E
i3 ← 0
1
xsa
k ← 0
1
akie
netarax (i3,k)=0
10
9
j ← 0
1
xsu
netarax (i3,k) = xx(i3,k) * wg(k,j)+netarax(i3,k)
END
netarax (i3,k)=netarax(i3,k)+ wge(k)
ciktiarax(i3,k)=1/(1+System. Math.Exp(-netarax(i3,k)))
END


9
mm1← 0
1
cies
net11x(i3,mm1)=0
m← 0
1
akie
net11x(i3,mm1) = ciktiarax(i3,m)*
wc(mm1,m)+net11x(i3,mm1)
END
END
net11x(i3,mm1)=net11x(i3,mm1)+wce1(mm1)
ciktisonx(i3,mm1)=1/(1+System.Math.Exp
(-net11x(i3,mm1)))
END
E
10
Verilen Girişe Karşılık Ağın Çıktısını Görüntüleme
Girişlere Karşılık Gelen Harfin Görüntülenmesi
kk
11


11
Hayır
Evet
F
kk=""
DUR

EK – 3

(C#' DA YAZILMIŞ OLAN YAPAY SİNİR AĞI PROGRAMI)

```
using System;
using System.IO;
class app
{
   public static int getShapeNumber()
   {
      return ms_r.Next(-100, 100);
   }
   public static System.Random ms_r=new System.Random();
   public static void Main()
   {
      // ------------   YAPAY SİNİR AĞI PROGRAMI   ------------------------
      // ----------------------  T.C.  ----------------------------------------------------
      // ------------------- HALİÇ ÜNİVERSİTESİ  --------------------------------
      // -------- BİLGİSAYAR MÜHENDİSLİĞİ - YÜKSEK LİSANS --------
      // ------------------  METİN ŞAHİN  ----------------------------------------
      // -----------  02  NİSAN 2007 --- PAZARTESİ ----------------------------
System.Console.WriteLine("_____
_____");
      System.Console.WriteLine("  Eğitim seti için girişlerin bulunduğu dosyanın ismini
giriniz       ");
      string dosegigiris1 = @"C:\dosya01.txt";
      System.Console.WriteLine();
      System.Console.WriteLine(" Dosya Adı                  : " + dosegigiris1);
      System.Console.Write    (" Dosya Adı (ENTER.....Devam) : ");
      string dos1 = System.Console.ReadLine();
      string dosegigiris;
      if (dos1 == "") dosegigiris =dosegigiris1;
      else
      {
         dosegigiris =@dos1;
```

```
        }

System.Console.WriteLine("_____
_____");
        System.Console.WriteLine("  Eğitim seti için çıkışların bulunduğu dosyanın ismini
giriniz         ");
        string dosegicikis1 = @"C:\dosya02.txt";
        System.Console.WriteLine();
        System.Console.WriteLine(" Dosya Adı                  : " + dosegicikis1);
        System.Console.Write    (" Dosya Adı (ENTER.....Devam) : ");
        string dos2 = System.Console.ReadLine();
        string dosegicikis;
        if (dos2 == "") dosegicikis =dosegicikis1;
        else
        {
           dosegicikis = @dos2;
        }

System.Console.WriteLine("_____
_____");
        System.Console.WriteLine("      Giriş bilgilerinin bulunduğu dosyanın ismini
giriniz              ");
        string dosbilgiris1 = @"C:\dosya03.txt";
        System.Console.WriteLine();
        System.Console.WriteLine(" Dosya Adı                  : " + dosbilgiris1);
        System.Console.Write    (" Dosya Adı (ENTER.....Devam) : ");
        string dos3 = System.Console.ReadLine();
        string dosbilgiris;
        if (dos3 == "") dosbilgiris = dosbilgiris1;
        else
        {
           dosbilgiris =@dos3;
        }
```

```
System.Console.WriteLine("_____
_____");

        FileStream fs = new FileStream(dosegigiris, FileMode.Open);
        FileStream fs2 = new FileStream(dosegicikis, FileMode.Open);
        FileStream fs3 = new FileStream(dosbilgiris, FileMode.Open);

        StreamReader sr = new StreamReader(fs);
        StreamReader sr2 = new StreamReader(fs2);
        StreamReader sr3 = new StreamReader(fs3);

        string Line;
        string Line2;
        string Line3;

        //System.Console.Write("Sinir ağında kaç tane giriş var : ");
        //int xsu = int.Parse(System.Console.ReadLine());
        int xsu = 35;
        //System.Console.Write("Girişlerden kaç adet var (Grup Sayısı) : ");
        //int xsa = int.Parse(System.Console.ReadLine());
        int xsa = 8;
        //System.Console.Write("Ara katmanda kaç adet işlem elemanı var : ");
        //int akie = int.Parse(System.Console.ReadLine());
        int akie = 10;
        //System.Console.Write("Çıkışta kaç adet işlem elemanı var : ");
        //int cies = int.Parse(System.Console.ReadLine());
        int cies = 8;
        //System.Console.Write("Öğrenme katsayısı   : ");
        //double epsilon = double.Parse(System.Console.ReadLine());
        double epsilon = 0.5;
        //System.Console.Write("Momentum : ");
        //double alfa = double.Parse(System.Console.ReadLine());
        double alfa = 0.8;
```

```
System.Console.Write    ("      Iterasyon Sayısı              : ");
int adim = int.Parse(System.Console.ReadLine());
// ALINMIŞ DEĞERLERİN KONTROL İÇİN GÖRÜNTÜLENMESİ
System.Console.WriteLine("      Girişdeki İşlem Elemanı Sayısı  : " + xsu);
System.Console.WriteLine("      Toplam boyut                  : " + xsa);
System.Console.WriteLine("      Ara katman işlem elemanı sayısı : " + akie);
System.Console.WriteLine("      Çıkışta ki işlem elemanı sayısı : " + cies);
System.Console.WriteLine("      Öğrenme                       : " + epsilon);
System.Console.WriteLine("      Momentum                      : " + alfa);
System.Console.WriteLine("      Iterasyon                     : " + adim);

System.Console.WriteLine("_____________________________________________________
_____________________________");
System.Console.WriteLine();
// GİRİŞ İLE ARA KATMAN ARASINDAKİ AĞIRLIKLARIN ATANMASI
double [ , ]  wg = new double [akie,xsu];
for (int i=0 ; i<akie ; i++)
   for(int j=0 ;j<xsu ; j++)
   {
   TEKRAR1:
      double number = getShapeNumber();
      double sayi = number / 100;
      if (sayi == 0) goto TEKRAR1;
      wg[i, j] = sayi;
   }
// ARA KATMANDAKİ AĞIRLIKLARIN ATANMASI
double[] wge = new double[akie];
for (int i = 0; i < akie; i++)
{
TEKRAR2:
   double number = getShapeNumber();
   double sayi = number / 100;
   if (sayi == 0) goto TEKRAR2;
   wge[i] = sayi;
```

```
        }
        // ARA KATMAN İLE ÇIKTI KATMANI ARASINDAKİ AĞIRLIKLARIN
ATANMASI
        double[ , ] wc = new double[cies,akie];
        for (int j = 0; j < cies; j++)
        {
           for (int i = 0; i < akie; i++)
           {
           TEKRAR3:
              double number = getShapeNumber();
              double sayi = number / 100;
              if (sayi == 0) goto TEKRAR3;
              wc[j, i] = sayi;
           }
        }
        // ÇIKIŞ KATMANINDAKİ AĞIRLIKLARIN ATANMASI
        double[] wce1 = new double[cies];
        for (int i = 0; i < cies; i++)
        {
        TEKRAR4:
           double number = getShapeNumber();
           double sayi = number / 100;
           if (sayi == 0) goto TEKRAR4;
           wce1[i] = sayi;
        }
        // GİRİŞLERİN VE ÇIKTILARIN OKUNMASI
        double[,] x = new double[xsa, xsu];
        double hasi1;
        double hasi2;
        double hasi3;
        int xsatır = 0;
        int xsutun = 0;
        // EĞİTİM İÇİN GİRİŞ DOSYANIN OKUNMASI
        char[] ayırıcı = { ' ' };
```

```
while ((Line = sr.ReadLine()) != null)
{
   xsutun = 0;
   //System.Console.WriteLine("satır = "+xsatır+"---"+Line);
   string[] sayılar = Line.Split(ayırıcı);
   foreach (string si in sayılar)
   {
      hasi1 = Convert.ToDouble(si);
      x[xsatır, xsutun] = hasi1;
      xsutun = xsutun + 1;
     //  System.Console.WriteLine("satır : " + xsatır + "---" + "sutun : " + xsutun);
   }
   xsatır = xsatır + 1;
}
fs.Close();
System.Console.WriteLine();
System.Console.WriteLine("          EĞİTİM İÇİN GİRİŞ DOSYASINI OKUDU");
// EĞİTİM İÇİN ÇIKIŞ DOSYASININ OKUNMASI
double[,] dk = new double[xsa,cies];
int dsatır = 0;
int dsutun = 0;
// EĞİTİM İÇİN ÇIKIŞ DOSYANIN OKUNMASI
char[] ayırıcı2 = { ' ' };
while ((Line2 = sr2.ReadLine()) != null)
{
   dsutun = 0;
   string[] sayılar2 = Line2.Split(ayırıcı2);
   foreach (string ya2i in sayılar2)
   {
      hasi2 = Convert.ToDouble(ya2i);
      dk[dsatır, dsutun] = hasi2;
      dsutun = dsutun + 1;
   }
   dsatır = dsatır + 1;
```

```
}
fs2.Close();
System.Console.WriteLine();
System.Console.WriteLine("          EĞİTİM İÇİN ÇIKIŞ DOSYASINI OKUDU");
string[] harf = new string[xsa];
harf[0] = "A"; harf[1] = "B"; harf[2] = "C"; harf[3] = "Ç";
harf[4] = "D"; harf[5] = "E"; harf[6] = "F"; harf[7] = "G";
int t = 0;
double buyuk = 0;
int adres = 0;
int kont2 = 0;
int haadim = 0;
int skont = 0;
int oran = 0;
double fark = 0;
double toplam = 0;
double kare = 0;
double ilkb = 0;
double ilkd = 0;
double ilkg = 0;
double ilkc = 0;
double mse = 0;
int ii = 2;
int i2 = 1;
double top, top1 , top2;
double[] hadez = new double[21];
double[,] topwkj = new double[xsa,cies];
double[,] net = new double[xsa,akie];
double[,] ciktia = new double[xsa,akie];
double[,] net1 = new double[xsa,cies];
double[,] net11 = new double[xsa,cies];
double[,] net11x = new double[50,cies];
double[,] cikti = new double[xsa,cies];
double[,] ros = new double[xsa,cies];
```

```
        double[,] dwc = new double[cies,akie];
        double [] dwce1 = new double [cies];
        double[,] roy = new double[xsa,akie];
        double[] dwge = new double[akie];
        double[,] dwg = new double[akie,xsu];
        double[,] ciktison = new double[xsa,cies];
        double[,] ciktisonx = new double[50,cies];
        double[,] netara = new double[xsa,akie];
        double[,] netarax = new double[50, akie];
        double[,] ciktiara = new double[xsa,akie];
        double[,] ciktiarax = new double[50, akie];
        System.Console.WriteLine();
        System.Console.WriteLine();
        System.Console.WriteLine("         ŞU ANDA AĞIRLIKLARI HESAPLIYOR
..................  ");
        oran = adim / 21;
        ILKBAS:
        if (t > adim) goto SONYAZ;
        toplam = 0;
        for (int i1 = 0; i1 < xsa; i1++)
        {
           // ARA KATMANDAKİ NETLERİN HESAPLANMASI
           for (int k = 0; k < akie; k++)
           {
              net[i1,k] = 0;
              for (int j = 0; j < xsu; j++)
              {
                 net[i1,k] = x[i1, j] * wg[k, j] + net[i1,k];
              }
              net[i1,k] = net[i1,k] + wge[k];
              ciktia[i1,k] = 1 / (1 + System.Math.Exp(-net[i1,k]));
           }
           // ÇIKTI KATMANINDAKİ NETLERİN HESAPLANMASI
           for (int k1 = 0; k1 < cies; k1++)
```

```
{
   net1[i1,k1] = 0;
   for (int m = 0; m < akie; m++)
   {
      net1[i1,k1] = ciktia[i1,m] * wc[k1, m] + net1[i1,k1];
   }
   net1[i1,k1] = net1[i1,k1] + wce1[k1];
   cikti[i1,k1] = 1 / (1 + System.Math.Exp(-net1[i1,k1]));
}
// ITERASYONDA Kİ KONTROL
   for (int ii2 = 0; ii2 < cies ; ii2++)
   {
     fark = dk[i1,ii2] - cikti[i1,ii2];
     kare = fark * fark;
     kare = System.Math.Sqrt(kare);
     toplam = toplam + kare;
   }

for (int e1 = 0; e1 < cies; e1++)
{
   ros[i1,e1] = (dk[i1, e1] - cikti[i1,e1]) * cikti[i1,e1] * (1 - cikti[i1,e1]);
   if (ii == 2) ilkb = 0;
   else
   {
      ilkb = dwce1[e1];
   }
   dwce1[e1] = epsilon * ros[i1,e1] + alfa * ilkb;
   top2 = wce1[e1] + dwce1[e1];
   wce1[e1] = top2;
}
for (int l1 = 0; l1 < cies; l1++)
{
   for (int l = 0; l < akie; l++)
   {
```

```
        if (ii == 2) ilkd = 0;
        else
        {
           ilkd = dwc[l1, l];
        }
        dwc[l1, l] = epsilon * ros[i1,l1] * ciktia[i1,l] + alfa * ilkd;
        top = wc[l1, l] + dwc[l1, l];
        wc[l1, l] = top;
     }
  }
     for (int r1 = 0 ; r1 < cies ; r1++)
     {
      topwkj[i1,r1] = 0;
           for (int r = 0; r < akie; r++)
           {
             topwkj[i1,r1] = topwkj[i1,r1] + wc[r1, r];
           }
     }
     for (int pp1 = 0; pp1 < cies; pp1++)
     {
        for (int p = 0; p < akie; p++)
        {
           roy[i1,p] = ciktia[i1,p] * (1 - ciktia[i1,p]) * ros[i1,pp1] * topwkj[i1,pp1];
           if (ii == 2) ilkg = 0;
           else
           {
              ilkg = dwge[p];
           }
           dwge[p] = epsilon * roy[i1,p] + alfa * ilkg;
           wge[p] = dwge[p] + wge[p];
        }
     }
  for (int a = 0; a < akie; a++)
  {
```

```
            for (int b = 0; b < xsu; b++)
            {
               if (ii == 2) ilkc = 0;
               else
               {
                  ilkc = dwg[a, b];
               }
               dwg[a, b] = epsilon * roy[i1,a] * x[i1, b] + alfa * ilkc;
               top1 = wg[a, b] + dwg[a, b];
               wg[a, b] = top1;
            }
         }
         ii = 3;
         i2 = i2 + 1;
         }
      t = t + 1;
      //mse = toplam / t;
      mse = toplam / (cies * xsa);
      // HATALARI DİZİYE AKTARMA
      if (skont == oran)
      {
         hadez[haadim] = mse;
         haadim = haadim + 1;
         skont = 0;
      }
      else
      {
         skont = skont + 1;
      }

      goto ILKBAS;
   SONYAZ:
      // YENİ AĞIRLIKLAR İLE ARA KATMAN ÇIKIŞLARIN YENİDEN
HESAPLANMASI
```

```
for (int i3 = 0; i3 < xsa; i3++)
{
   for (int k = 0; k < akie; k++)
   {
      netara[i3,k] = 0;
      for (int j = 0; j < xsu; j++)
      {
         netara[i3,k] = x[i3, j] * wg[k, j] + netara[i3,k];
      }
      netara[i3,k] = netara[i3,k] + wge[k];
      ciktiara[i3,k] = 1 / (1 + System.Math.Exp(-netara[i3,k]));
   }
   // YENİ AĞIRLIKLAR İLE ÇIKTI KATMANINDA Kİ ÇIKIŞIN YENİDEN
HESAPLANMASI
   for (int mm1 = 0; mm1 < cies; mm1++)
   {
      net11[i3,mm1] = 0;
      for (int m = 0; m < akie; m++)
      {
         net11[i3,mm1] = ciktiara[i3,m] * wc[mm1, m] + net11[i3,mm1];
      }
      net11[i3,mm1] = net11[i3,mm1] + wce1[mm1];
      ciktison[i3,mm1] = 1 / (1 + System.Math.Exp(-net11[i3,mm1]));
   }
}
   // System.Console.WriteLine
   //("----------------------------------------------------------------");
for (int i3 = 0; i3 < xsa; i3++)
{
   //System.Console.Write("     "+"GİRİŞ DEĞERLERİ  : ");
   for (int i4 = 0; i4 < xsu; i4++)
   {
    //  System.Console.Write(x[i3, i4] + " ");
   }
```

```
   //System.Console.WriteLine();
   //System.Console.Write("     "+"İSTENEN DEĞERLER : ");
   for (int i6 = 0; i6 < cies; i6++)
   {
    // System.Console.Write(dk[i3, i6]+ " ") ;
   }
   //System.Console.WriteLine();
   //System.Console.Write("     "+"AĞIN ÇIKTILARI   : ");
   for (int i7 = 0; i7 < cies; i7++)
   {
    //  System.Console.Write(ciktison[i3,i7] + " ");
   }
   //System.Console.WriteLine();
   //System.Console.WriteLine
   //("------------------------------------------------------------------");
}
System.Console.WriteLine();
System.Console.Write("          Devam için  (ENTER)' a basın : ");
string kk1 = System.Console.ReadLine();
System.Console.WriteLine("    HATA BİLGİLERİ GÖRÜNTÜLEME ");
System.Console.WriteLine("_____________________________________________");
for (int i3 = 0; i3 < 20; i3++)
{
   System.Console.WriteLine("  HATA " + i3 + " --> " + hadez[i3]);
}
System.Console.WriteLine("_____________________________________________");
System.Console.Write("    Devam için  (ENTER)' a basın : ");
string kk2 = System.Console.ReadLine();
DEVADIM:
// DOSYADAN ALINAN GİRİŞ DEĞERLERİNE GÖRE ÇIKTI HESABI
double[,] xx = new double[50, xsu];
int x1satır = 0;
int x1sutun = 0;
char[] ayırıcı3 = { ' ' };
```

```
while ((Line3 = sr3.ReadLine()) != null)
{
   x1sutun = 0;
   string[] sayılar3 = Line3.Split(ayırıcı3);
   foreach (string ya3i in sayılar3)
   {
      hasi3 = Convert.ToDouble(ya3i);
      xx[x1satır, x1sutun] = hasi3;
      x1sutun = x1sutun + 1;
   }
   x1satır = x1satır + 1;
}
fs3.Close();
System.Console.WriteLine("    GİRİŞ DEĞERLERİNİ OKUDU");
xsa = x1satır;
for (int i3 = 0; i3 < xsa; i3++)
{
   for (int k = 0; k < akie; k++)
   {
      netarax[i3,k] = 0;
      for (int j = 0; j < xsu; j++)
      {
         netarax[i3,k] = xx[i3, j] * wg[k, j] + netarax[i3,k];
      }
      netarax[i3,k] = netarax[i3,k] + wge[k];
      ciktiarax[i3,k] = 1 / (1 + System.Math.Exp(-netarax[i3,k]));
   }
   // YENİ AĞIRLIKLAR İLE ÇIKTI KATMANINDA Kİ ÇIKIŞIN YENİDEN
HESAPLANMASI
   for (int mm1 = 0 ; mm1 < cies ; mm1++)
   {
   net11x[i3,mm1] = 0;
   for (int m = 0; m < akie; m++)
      {
```

```
            net11x[i3,mm1] = ciktiarax[i3,m] * wc[mm1, m] + net11x[i3,mm1];
          }
        net11x[i3,mm1] = net11x[i3,mm1] + wce1[mm1];
        ciktisonx[i3,mm1] = 1 / (1 + System.Math.Exp(-net11x[i3,mm1]));
        }
      }
      // BELİRLENMİŞ HARFİN GÖRÜNTÜLENMESİ İŞLEMİ
      int xsyazi;
      xsa = x1satır;
      for (int i3 = 0; i3 < xsa; i3++)
      {
        buyuk = ciktisonx[i3, 0];
        adres = 0;
        for (int i7 = 1; i7 < cies; i7++)
        {
          if (ciktisonx[i3, i7] > buyuk)
          {
            buyuk = ciktisonx[i3, i7];
            adres = i7;
          }
        }
        xsyazi = i3 + 1;

System.Console.WriteLine("_____________________________________________");
        System.Console.WriteLine("|  "+xsyazi+" .KARŞILIK GELEN HARF : " +
harf[adres]+"        |");

System.Console.WriteLine("_____________________________________________");
        //System.Console.WriteLine("Karşılık Gelen Harf Bulunamadı");
      }
      //CIK01:
      System.Console.WriteLine();
      System.Console.WriteLine();
      System.Console.Write("      Programdan çıkmak için (ENTER) : ");
```

```
        string kk = System.Console.ReadLine();
        if (kk == "") goto CIKIS;
        else
        {
           goto DEVADIM;
        }
        CIKIS:
        System.Console.WriteLine("    ***** PROGRAMDAN ÇIKIŞ *****    ");
    }
}
```

KAYNAKLAR

[1] : AKÇAY, B. ; Yapay Sinir Ağları, Elektrik Müh. 6. Ulusal Kongresi, Eylül, 1995

[2] : ALGAN, Sefer; Her Yönüyle C# , Pusula Yayıncılık, İstanbul, Mayıs, 2006

[3] : ALLAHVERDİ, Nevroz; Uzman Sistemler ve Yapay Sinir Ağları, Mart, 2002

[4] : ALTINTAŞ, Ergin; Yapay Sinir Ağları ve Tanıma Sistemleri, Nisan, 2005

[5] : ARCHER, Tom; C#'ı Kavramak, Arkadaş Yayınları, Ankara, 2004

[6] : ARKIN, Ethem; Hacettepe Üni. Bilgisayar Müh. Yapay Sinir Ağları

[7] : ÇAVUŞOĞLU, Abdullah; Yapay Sinir Ağları

[8] : ÇETİN, Müjdat; Yapay Sinir Ağları

[9] : ÇÖL, Muhterem; Fatih Üniversitesi, Müh. Fak. Yapay Sinir Ağları

[10] : ELMAS, Çetin; Yapay Sinir Ağları, Seçkin Yayıncılık, Ankara, Nisan, 2003

[11] : ERDEN, Sema; Yapay Sinir Ağları İle Zamanlı Verilerin Analizi

[12] : FIRAT, M. ; Pamukkale Üniversitesi, Yapay Sinir Ağları

[13] : GELİR, Ethem; Yapay Sinir Ağları, Gazi Üniversitesi 1994

[14] : GÖKÇE, N. ; Muğla Üniversitesi, Fen Bilimleri Enst. Yapay Sinir Ağları Modelleri

[15] : KARAKUZU, Cihan; Yapay Sinir Ağları İle Kontrol

[16] : KILIÇ, H. B. ; Yapay Sinir Ağları İle Karakter Algılama, Gazi Üni. Ankara, 1998

[17] : KIRBAŞ, İsmail; Yapay Sinir Ağları

[18] : KIYMIK, M. Kemal; Yapay Sinir Ağları

[19] : OKATAN, Ali; Yapay Sinir Ağları Ders Notları, İstanbul, Aralık, 2005

[20] : ÖLMEZ, Taner; Yapay Sinir Ağları Algılama

[21] : ÖZBAYOĞLU, Murat; Özellik Tanıma ve Görüntü, Mayıs, 1996

[22] : ÖZTEMEL, Ercan; Yapay Sinir Ağları, Papatya Yayıncılık, İstanbul, Ağustos, 2003

[23] : ÖZTÜRK, S. ; Yapay Sinir Ağları Temelli Resim Netlik Kıstası

[24] : SAĞIROĞLU, Şeref; Yapay Sinir Ağları ve Mühendislik Uygulamaları

[25] : ŞEKER, S. ; Yapay Sinir Ağları ve Uzman Sistemler

[26] : ŞEN, Zekai; Yapay Sinir Ağları İlkeleri

[27] : TAINN – Türk Yapay Zeka ve Yapay Sinir Ağları Sempozyumları Düzenleme Kuralları

[28] : TAVŞANOĞLU, Vedat; Yapay Sinir Ağları İle Görüntü İşleme, Aralık, 2002

[29] : TÜRKER, İpek ; Yapay Sinir Ağları İle Modelleme

[30] : UĞUR, Aybars; Yapay Sinir Ağları Ders Notları

[31] : WILSON, R. L. ; Neural Networks, 1992

[32] : YALÇIN, Mutsak Erhan; Yapay Sinir Ağları

[33] : YAZICI, Rıfat; Yapay Sinir Ağları

[34] : YÜKSEL, İbrahim; Matlab ile Mühendislik Sistemlerinin Analizi, Nobel Yayın , Ankara, 2004

[35] : ZURADA, J. M. ; Introduction to Artificial Neural Systems, West Yayın, New York, 1992

[36] : http : // www.backpropagation.netfirms.com

[37] : http : // www.gazi.edu.tr/journal/2005_1/13_20.pdf

[38] : http : // www.yapay-zeka.org/modules/icontent/index.php?page=47–30k

ÖZGEÇMİŞ

Kişisel Bilgiler

Doğum Tarihi : 20 – 06 – 1965

Doğum Yeri : Elazığ

Eğitim

Yüksek Lisans : Haliç Üniversitesi – Bilgisayar Mühendisliği (2005 – 2007)

Lisans : Yıldız Üniversitesi

Fakülte : Mühendislik Fakültesi

Bölüm : Bilgisayar Bilimleri Mühendisliği (1982 – 1986)

Deneyim

- Kale Kilit ve Kalıp San. A.Ş. ; Nixdorf bilgisayar sistemlerini kullanarak bordro programı yazımı , bu programın muhasebe ile entegre hale getirilmesi ve depo takibi yazılımı konularında görev aldım. (1987- 1988)
- PB – TSB (Parsons Brinckerhoff International Inc. – Teknik Servisler Bürosu) ; Ulaşım planlaması , metro ve tüp geçit işletme planması konularında görev aldım. Bu konular ile ilgili mühendislik programları yazılımında (Örneğin : Metro da çalışan trenlerin trafolardan çektikleri elektrik güçlerinin hesaplanması ve grafiksel olarak bu güç dağılımının yazıcıda dökümü, yolculuk sayılarının bulunması ve daha değişik alternatif çözümlerin üretilmesi…) (1986 – 1987)
- Lisans Bitirme Projesi (Hafızalı Osiloskop) ; ADC (Analog Dijital Çevirici) , DAC (Dijital Analog Çevirici) ve Z-80 CPU kullanarak analog bir sinyalin (bir insanın sesi veya kalp atış sinyalleri gibi) belleğe alınması ve ardından bu sinyallerin osiloskopta izlenmesi yada bir hat yardımı ile istenilen bir ortama gönderilmesi konusunda bir çalışma gerçekleştirdim. (1986)