Yapay Sinir Ağları Kullanılarak Santrifüj Pompalarda Performans Tayini

Sinan Ardıç

YÜKSEKLİSANSTEZİ

Makine Mühendisliği Anabilim Dalı

Ocak 2014

Determination of Performance of Centrifugal Pumps By Using Artificial Neural Networks

Sinan Ardıç

MASTER OF SCIENCE THESIS

Department of Mechanical Engineering

January 2014

Yapay Sinir Ağları Kullanılarak Santrifüj Pompalarda Performans Tayini

Sinan Ardıç

Eskişehir Osmangazi Üniversitesi
Fen Bilimleri Enstitüsü
Lisansüstü Yönetmeliği Uyarınca
Makine Mühendisliği Anabilim Dalı
Enerji Termodinamik Bilim Dalında
YÜKSEK LİSANS TEZİ
Olarak Hazırlanmıştır

Danışman: Prof. Dr. Yaşar Pancar

Ocak 2014

ONAY

Makine Mühendisliği Anabilim Dalı Yüksek Lisans öğrencisi Sinan Ardıç'ın YÜKSEK LİSANS tezi olarak hazırladığı "Yapay Sinir Ağları Kullanılarak Santrifüj Pompalarda Performans Tayini" başlıklı bu çalışma, jürimizce lisansüstü yönetmeliğin ilgili maddeleri uyarınca değerlendirilerek kabul edilmiştir.

Danışman **:** Prof. Dr. Yaşar PANCAR

İkinci Danışman **: -**

Yüksek Lisans Tez Savunma Jürisi:

Üye : Prof. Dr. Yaşar PANCAR

Üye : Doç. Dr. Necati MAHİR

Üye : Yrd. Doç. Dr. H. Sevil ERGÜR

Üye : Yrd. Doç. Dr. Mesut TEKKALMAZ

Üye : Yrd. Doç. Dr. Hakan GAŞAN

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ..... tarih ve.....sayılı kararıyla onaylanmıştır.

Prof. Dr. Nimetullah BURNAK

Enstitü Müdürü

ÖZET

Bu çalışmada ilk olarak yapay sinir ağları ve santrifüj pompalarla ilgili temel kavramlar ele alınarak, uygulama için hangi yapay sinir ağının uygun olacağı belirlenmiştir. Daha sonra T.Ş.F.A.Ş Eskişehir Şeker Makine Fabrikası'nda imal edilen pompaların karakteristik eğrilerinden alınan değerler ile yapay sinir ağı MATLAB aracılığı ile oluşturulmuş ve eğitilmiştir. Bu aşamadan sonra da ağ hiç görmediği örneklerle test edilmiştir. Bu test sonucu oluşan hatalar minimize edilerek bu işlemler sonucunda minimum hata %1,3 olup B 50-200 tip santrifüj pompaya ait olduğu bulunmuştur. Diğer pompalar için de bulunan hatalar çalışma içinde verilmiştir. Daha sonra yapay sinir ağı verileri ile verim eğrileri oluşturularak ara değerler için performans tayininin mümkün olduğu gösterilmiştir.

Anahtar Kelimeler: MATLAB, santrifüj pompa, yapay sinir ağı.

SUMMARY

In this thesis, the basic concepts of artificial neural networks and centrifugal pumps were investigated. The suitable neural network model was determined. The data set for neural network was taken from the characteristic curves of selected pumps'manufactured at TC Şeker Machine Factory. Then, the neural network model was formed by using MATLAB, after that, the neural network was trained. The neural network was tested by using non similar data. The aim of the test is to decrease the error value. In the end the minimum error value 1,3% was founded which belongs to B 50-200 centrifugal pump. Also the errors for the other type of pump were found. Then efficiency curves were drawn by the aid of neural network data. This demonstrated the possibility of detecting pump performance for non similar data.

Keywords: MATLAB, centrifugal pump, artificial neural netwok.

TEŞEKKÜR

Üniversite eğitim ve öğrenimim ve tez çalışmam boyunca göstermiş olduğu ilgi; görüş ve eleştirileri için danışman hocam Prof. Dr. Yaşar PANCAR'a, modelleme kısmında benden yardımlarını esirgemeyen Elektronik ve Haberleşme Mühendisi Salih YILDIRIM'a, tez çalışmamın başından sonuna kadar değerli görüş, eleştirileri ve manevi desteği ile hep yanımda olan Makine Mühendisi Emine UZUN'a ve son olarak T.Ş.F.A.Ş Eskişehir Şeker Makine Fabrikası Müdürlüğü'ne teşekkürü borç bilirim.

İÇİNDEKİLER

Sayfa

İÇİNDEKİLER (devam)

Sayfa

İÇİNDEKİLER (devam)

Sayfa

İÇİNDEKİLER (devam)

Sayfa

İÇİNDEKİLER (devam)

ŞEKİLLER DİZİNİ

Şekil **Sayfa**

ÇİZELGELER DİZİNİ

Çizelge **Sayfa**

SİMGELER VE KISALTMALAR DİZİNİ

Simgeler	Açıklamalar
A, w	Ağırlık
B	Beklenen çıktı
d	İki vektör arasındaki mesafe
D	Giriş çapı (mm)
E	Hata
F	Transfer fonksiyonu
g	Yerçekimi ivmesi (m/s^2)
H	Manometrik yükseklik (m)
J	Jacobian matris
mak	Maksimum
min	minimum
N	Güç (kW)
n	Devir sayısı (d/dk)
n_g	Özgül hız (d/dk)
NET	Net girdi değeri
P	Basınç (Pa)
P	Pompa motor gücü (kW)
P_m	Pompa tahrik motor gücü (kW)
Q	Debi (m^3/h)
sgn	Sigmoid fonksiyonu
t	Zaman (s)
TH	Toplam hata
V	Hız (m/s)
X, G	Girdi değeri
Y, Ç	Çıktı değeri
z	Referans düzleme olan mesafe (m)
α	Momentum katsayısı
β	Eşik değer ağırlığı

SİMGELER VE KISALTMALAR DİZİNİ (devam)

Simgeler	Açıklamalar
$\dot{m}$	Kütlesel debi (kg/s)
η_g	Genel verim
η_h	Hidrolik verim
η_m	Mekanik verim
η_v	Volumetrik verim
θ	Eşik değeri
ρ	Akışkan yoğunluğu (kg/m^3)

Kısaltmalar	Açıklamalar
ADALINE	Adaptive Linear Element (Adaptif Lineer Element)
ART	Adaptive Rezonance Theory (Adaptif Rezonans Teorisi)
ÇKA	Çok Katmanlı Algılayıcı
GRNN	General Regression Neural Networks (Genel Regrasyon Sinir Ağları)
KDH	Kısa Dönemli Hafıza
LM	Levenberg-Marguardt
LMS	Least Mean Square (En Küçük Kareler)
LVQ	Linear Vector Quantization (Doğrusal Vektör Nicelemesi)
MADALINE	Multiple Adaptive Linear Element (Çoklu Adaptif Lineer Element)
MATLAB	Matrix Laboratory (Matris Laboratuarı)
NPSH	Net Positive Suction Head (Pozitif Net Emme Yükü)
PNN	Probabilistic Neural Networks (Olasılıksal Sinir Ağları)
RBFN	Radial Basis Neural Networks (Radyal Tabanlı Sinir Ağları)
SOM	Self-Organization and Associative Memory (Öz Örgütlemeli Hafıza)
TKA	Tek Katmanlı Algılayıcı
UDH	Uzun Dönemli Hafıza
YSA	Yapay Sinir Ağı
YYM	Yeniden Yerleştirme Modülü

BÖLÜM 1

GİRİŞ

Çok eski zamanlardan beri insan beyninin nasıl çalıştığı merak edilmiştir. Dış dünyadan gelen etkilerin yorumlanılıp bir tepki oluşturulması ve bu tepkinin oluşturulurken insan beyninin nasıl çalıştığı incelenmek istenmiştir. İlk ilkel hesap makinelerinin de bu merak sonucu ortaya çıktığı söylenebilir.

Yerkürede bilgisayar ve bilgisayar sistemleri insanların hayatlarında büyük bir yer kaplamaktadır. Hemen hemen her alanda bilgisayarlardan yararlanılmaktadır. Bilgisayarların bu teknolojik gelişimi incelenirse, önceleri sadece elektronik veri aktarımı ve karmaşık hesaplamalar yapmak üzere geliştirilen bilgisayarlar gelişen süreç içerisinde verileri filtreleyip özetleyen, mevcut bilgileri kullanıp olaylar hakkında yorum yapan makineler haline gelmiştir.

Bilgisayarların bu teknolojik süreci içinde araştırmacılar insan davranışlarının modellenmesi için çalışmalar yapmışlardır. Bu gelişmeler sonucu 1950'li yıllarda yapay zeka kavramı ortaya çıkmıştır. Yapay zeka ile matematiksel formülasyonu kurulamayan karmaşık problemler, insan beyninin öğrenme yolunun benzerini kullanarak sezgisel olarak çözülmek istenilmiştir. Yapay zeka çalışmaları hızla gelişirken yapay sinir ağları oluşturulmuştur.

Yapay sinir ağları, olayların örneklerini inceleyip olaylar hakkında genelleme yapıp ve daha sonra görmediği örnekler hakkında mantıklı sonuçlar veren sistemlerdir. Yapay sinir ağları ile ilgili özellikle 1990'lı yıllar itibarıyla sayısız araştırma yapılmış ve çok sayıda yapay sinir ağı modeli oluşturulmuştur.

Günümüzde yapay sinir ağları, sanayideki uygulamalardan finansal uygulamalara, tıp uygulamalarından askeri sistemlere kadar geniş bir alanda kullanılmaktadır.

Bu çalışmada da belirlenen tip santrifüj pompalarının karakteristik eğrilerinden elde edilen verilerle yapay sinir ağı modellenmiştir. Ağın ürettiği çıktılarla pompa verim eğrileri oluşturulmuş ve gerçek eğriler ile karşılaştırılmıştır. Oluşturulan bu eğriler ile yapay sinir ağı kullanılarak performans tayininin yapılabilir olduğu gösterilmiştir.

BÖLÜM 2

YAPAY ZEKA

2.1 Yapay Zeka Kavramı

İnsan beyni dünyanın en karmaşık makinesi olarak kabul edilebilir. İnsan beyni 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 değerlendiremese dahi geçmişte kazanmış olduğu tecrübeler sayesinde aracın 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? (Elmas, 2007). Bu gibi problemlerden esinlenerek bilgisayarlar farklı bir yönde gelişmeye başlamıştır.

Günümüzde bilgisayarlar hem olaylar hakkında karar verebilmekte hem de olaylar arasında ilişkileri öğrenebilmektedir. Matematiksel olarak formülasyonu kurulamayan ve çözülmesi mümkün olmayan problemler sezgisel yöntemler kullanılarak bilgisayarlar tarafından çözülebilmektedir. Bilgisayarları bu özelliklerle donatan ve bu yeteneklerinin gelişmesini sağlayan çalışmalar "yapay zeka" olarak bilinmektedir. İlk defa 1950'li yıllarda ortaya atılan yapay zeka terimi zaman içinde oldukça yoğun ilgi gördüğünden 40-50 yıllık bir zaman diliminde hayatın vazgeçilmez parçası olan sistemlerin doğmasına neden olmuştur. Artık bilgisayarlar eskiden olduğu gibi sadece bilgi iletişiminin ve hesaplamaların otomasyonunu yapan sistemler olarak görülmemektedir. İnsan karar verme sürecine oldukça benzer bir karar verme sürecine kavuşmakta ve daha karmaşık fakat kullanışlı sistemler ortaya çıkmaktadır. Yapay zeka bilimine göre bu bilimin; bilginin organizasyonu, öğrenme, problem çözme, teorem ispatlama, bilimsel buluşların modellenmesi gibi birçok konu ile ilgilendiği

görülmektedir. Bu yetenekler ile donatılan bilgisayar sistemleri problemlere çözüm üretirken insanın problemleri çözme sürecini taklit etmektedir (Öztemel, 2006).

2.2 Yapay Zeka Teknolojileri

2.2.1 Uzman sistemler

Bir problemi o problemin uzmanlarının çözdüğü gibi çözebilen bilgisayar programları geliştiren teknolojidir. Uzman sistemlerde bilgi ve deneyimler bilgisayarda saklanır. Bilgisayar bilgi tabanında saklanan bu bilgileri kullanarak insan karar verme sürecine benzer bir süreçle problemlere çözüm üretir.

Bir uzman sistemin bilginin temin edilmesi, bilgi tabanı, çıkarım mekanizması ve kullanıcı ara birimi olmak üzere dört elemanı vardır (Öztemel, 2006).

2.2.2 Makine Öğrenmesi

Bilgisayarların olayları öğrenmesini sağlayan teknolojidir. Genellikle örnekler kullanılarak olayların girdi ve çıktıları arasında ilişkiler öğrenilir. Öğrenilen bilgiler ile benzer olaylar yorumlanılarak karar verilir veya problemler çözülür (Öztemel, 2006).

2.2.3 Genetik Algoritmalar

Karışık optimizasyon problemlerin çözülmesinde kullanılan bir teknolojidir. Bir problemi çözmek için önce rastgele başlangıç çözümleri belirlenmektedir. Daha sonra bu çözümler birbiri ile eşleştirilerek performansı yüksek çözümler üretilmektedir. Bu şekilde sürekli çözümler birleştirilerek yeni çözümler aranmaktadır.

Bir genetik algoritmanın temel elemanları şunlardır: Kromozom ve gen, çözüm havuzu, çaprazlama, mutasyon, uygunluk fonksiyonu, yeniden üretim elemanı.

2.2.4 Bulanık mantık

Bulanık mantık yaklaşımı, makinelere insanlara özel verilerini işleyebilme ve onların deneyimlerinden ve önsezilerinden yararlanarak çalışabilme yeteneği verir. Bu yeteneği kazandırırken sayısal ifadeler yerine sembolik ifadeler kullanır. Bu sembolik ifadelerin makinelere aktarılması matematiksel bir temele dayanır. Bu matematiksel temel bulanık mantık kümeler kuramı ve buna dayanan bulanık mantıktır (Elmas, 2007).

2.2.5 Zeki etmenler

Bağımsız karar verebilen bilgisayar sistemleridir. Hem donanım hem de yazılım olarak geliştirilebilmektedirler. Birden fazla yapay zeka teknolojisi kullanılabilinir. Algılama, kavrama (idrak) ve eylem olmak üzere üç ana elemana sahiptir.

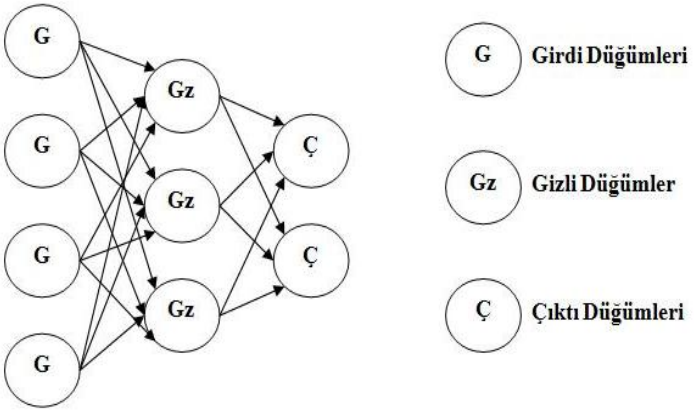
BÖLÜM 3

YAPAY SİNİR AĞLARI

3.1. Yapay Sinir Ağlarının Tanımı

Yapay sinir ağları (YSA), insan beyninden esinlenilerek geliştirilmiş, ağırlıklı bağıntılar aracılığıyla birbirine bağlanan ve her biri kendi bağına sahip işlem elemanlarından oluşan paralel ve dağıtılmış bilgi işleme yapılarıdır. Bir başka deyişle, biyolojik sinir ağlarını taklit eden bilgisayar programlarıdır. Yapay sinir ağları zaman zaman bağlantıcılık, paralel dağıtılmış işlem, sinirsel-işlem, doğal zekâ sistemleri ve makine öğrenme algoritmaları gibi isimlerle de anılmaktadır. Yapay sinir ağları bir programcının geleneksel yeteneklerini gerektirmeyen, kendi kendine öğrenme düzenekleridir. Bu ağlar öğrenmenin yanı sıra, ezberleme ve bilgiler arasında ilişkiler oluşturma yeteneğine de sahiptir (Elmas, 2007).

Yapay sinir ağları günümüzde birçok probleme çözüm üretebilecek yeteneğe sahiptirler. Tanımlarının birkaç ortak 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 birbirine bağlandıkları ve her bağlantının bir değerinin olduğu kabul edilmektedir (Öztemel, 2006). Yapay sinir ağı modeli basit olarak şekil 3.1'de gösterilmiştir.



Şekil-3.1. Bir yapay sinir ağı modeli

3.2. Yapay Sinir Ağlarının Genel Özellikleri

Yapay sinir ağları uygulanan ağ modeline göre çeşitli karakteristik özellikler gösterir. Buna karşın her ağ modeli için temel birkaç genel özellikten bahsedilebilir. Bunları aşağıdaki gibi sıralamak mümkündür:

• **Yapay sinir ağları makine öğrenmesini gerçekleştirirler:** Yapay sinir ağları bilgisayarların öğrenmesi üzerine yoğunlaşır. Öğrenilen olaylar arası benzerliklerden yararlanır.

• **Bilginin saklanması:** Yapay sinir ağlarının bağlantı değerleri bilginin önemini belirler ve bilgi bağlantılarda saklanır. Diğer programlar gibi veriler için bir veri tabanı kullanılmaz, bilgiler ağ üzerinde saklıdır.

• **Yapay sinir ağları örnekleri kullanarak öğrenirler:** Bir yapay sinir ağının olayları öğrenebilmesi için o olayla ilgili örnekler belirlenmesi gerekmektedir. Belirlenen örnekler kullanılarak ağın olay hakkında genelleme yapacak yeteneğe gelmesi sağlanır. Örnekler olmadan yapay sinir ağının eğitilmesi mümkün değildir.

• **Algılamaya yönelik olaylarda kullanılabilirler:** Ağlar bilgiye dayalı çözüm isteyen problemden ziyade algılamaya yönelik bilgileri işlemede kullanılırlar.

• **Örüntü ilişkilendirme ve sınıflandırma yapabilirler:** Genel olarak ağların çoğunun amacı kendisine örnekler halinde verilen örüntülerin kendisi veya diğerleri ile ilişkilendirilmesidir. Diğer bir amaç ise sınıflandırma yapmaktır. Örneklerin kümelendirilmesi ve belirli sınıflara ayrılarak daha sonra gelen bir örneğin hangi sınıfa gireceğine karar vermesi hedeflenir.

• **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.

• **Eksik bilgi ile çalışabilirler:** Yapay sinir ağları eğitildikten sonra eksik bilgiler ile çalışabilir ve gelen yeni örneklerde eksik bilgi olmasına rağmen sonuç üretebilirler. Bu, ağda performans düşüklüğüne neden olacağı anlamına gelmemelidir çünkü performans eksik olan bilginin önemine bağlıdır. Hangi bilginin önemli olduğunu ağ eğitim sırasında öğrenmektedir. Kullanıcının bu konuda bir fikri yoktur. Eğer performans düşüyorsa eksik bilginin önemli olduğu anlaşılır.

• **Hata toleransına sahiptirler:** Eksik bilgilerle çalışabilmek yapay sinir ağlarına hatalara karşı toleranslı olmalarını sağlamaktadır. Ağda bozulan bazı hücreler olsada ağ çalışmaya devam eder. Bozulan hücrenin sorumluluk değeri ağ performansında düşmeye neden olabilir.

• **Dereceli bozulma gösterirler:** Hatalara karşı toleranslı olan yapay sinir ağlarının bozulmaları dereceli olmaktadır. Ağ zaman içerisinde eksik bilgi veya bozuk hücre kaynaklı olarak yavaş yavaş bozulur.

• **Dağıtık belleğe sahiptirler:** Yapay sinir ağlarında bilgi ağa dağılmış durumdadır. Hücrelerin bağlantılarının değerleri ağın bilgisini gösterir. Ağın tamamı öğrendiği olayın bütününü karakterize ettiğinden bilgiler ağa dağılmış durumdadır.

• **Sadece nümerik bilgilerle çalışabilmektedirler:** Yapay sinir ağına girilen bilgiler nümerik olmak zorundadır. Semboller ya da resimler nümerik olarak ifade edildikten sonra ağa gösterilmelidir (Öztemel,2006; Ergezer vd., 2003).

3.3. Yapay Sinir Ağlarında Görülen Eksiklikler

Yapay sinir ağlarında görülen eksiklikleri şu şekilde sıralanabilir:

a) YSA'lar donanıma bağlı olarak çalışırlar ve ağ paralel işlemciler üzerinde çalışabilir. Günümüzdeki makinelerin çoğu seri biçimde çalışmaktadır ve aynı zamanda tek bir bilgiyi işleyebilmektedir. Paralel işlemleri seri makinelerde yapmak zaman kaybına neden olabilir.

b) Probleme uygun ağ yapısının belirlenmesinde kullanılan genel tekniğin deneme yanılma tekniği olması önemli bir eksikliktir. Dolayısıyla bulunan çözümün en iyi çözüm olduğunu garanti etme problemi ortaya çıkacağından YSA'lar kabul edilebilir sonuçlar üretir. Ancak en iyi çözümü garanti etmez.

c) Ağ oluşturulurken ağın parametre değerlerinin (öğrenme katsayısı, proses elemanı sayısı vb.) belirlenmesinde kullanılan bir kural olmaması önemli bir eksikliktir. Bu parametrelerin bulunması kullanıcı tecrübesine bağlıdır. Bu parametre değerleri için belirli standartların oluşturulması çok zor olduğundan her problem için ayrı ayrı değerlendirmeler yapılmalıdır.

d) YSA'ların sadece nümerik değerlerle çalışması problemin ağa gösterilmesinde önemli bir eksiklik oluşturmaktadır. Problemin nümerik gösterime dönüştürülmesi gerekir. Uygun bir gösterim mekanizmasının kurulamamış olması problemin çözümünü engelleyeceğinden düşük performanslı bir öğrenme elde edilecektir.

e) Ağ eğitimin ne kadar süreceğine karar vermek içinde herhangi bir yöntem belirlenmemiştir. Ağın hatayı belirli bir değerin altına indirmesi eğitimin tamamlanması için yeterli görülsede en iyi öğrenmenin gerçekleştiği anlamına gelmez.

f) En önemli eksiklik ise ağın davranışlarının açıklanamamasıdır. Bir probleme çözüm üretildiği zaman bunun nasıl ve neden üretildiğini bulmak mümkün değildir (Öztemel 2006; Şen 2004).

3.4. Yapay Sinir Ağlarının İşleyiş Prensibi

Yapay sinir ağları kendilerine gösterilen girdi setine karşılık gelebilecek uygun çıktı setini belirleyen mekanizmalardır. Ağ bu çıktıları sağlamak için uygun örneklerle eğitilir ve genelleme yapacak yeteneği kazanır. Eğitim genellikle ağa örnek girdi ve çıktı setleri ile gerçekleştirilir. Ağ bu eğitim sayesinde gelen girdilere karşılık beklenen çıktılar üretmeyi öğrenir. Genellemeden yola çıkarak benzer girdilere karşılık gelebilecek çıktıları ağ kendiliğinden belirleyebilir (Öztemel, 2006).

Yapay sinir ağlarının temel özellikleri mimari bölüm ve fonksiyonel bölüm olarak iki bölümde incelemek mümkündür. Mimari yapı ağın topolojisini belirler. Ağdaki nöron sayısı ve birbirleri ile olan bağlantıları bu mimari yapı tanımlar. Ağ çok sayıda, benzer karakteristik özelliklere sahip nöron yada diğer bir adlandırmayla proses elemanlarının birbirine bağlanması ile oluşur. Ağın öğrenmesi, öğrendiklerini hatırlaması, veriler arasında bağlantı kurması, yeni bilgilerle var olan bilgileri karşılaştırabilmesi, yeni bilgilerin sınıflandırılması ve eğer gerekli ise yeni sınıflandırmaların geliştirilmesi ağın fonksiyonel özellikleridir (Kartalopoulos, 1996). Ağın topolojisinin ayarlanması, ağırlık faktörleri, aktivasyon parametreleri ve diğer ağ parametreleri, ağa uygun eğitim metodu ile belirlenmelidir (Kalach, 2005).

Yapay sinir ağları geleneksel işlemcilerden farklı şekilde çalışırlar. Seri sistemlerden farklı olarak, her biri problemin bir parçası ile ilgilenen, çok sayıdaki basit

işlem elemanlarının paralel olarak çalışması ile problem çözülür. İşlem elemanları, girdiyi ağırlık kümesi ile ağırlıklandırır, doğrusal olmayan bir şekilde dönüşümünü sağlar ve bir çıktı değeri üretir. YSA'larda çoğu zaman benzer karakteristiğe sahip nöronlar tabakalar halinde yapılandırılır ve transfer fonksiyonları eş zamanlı çalıştırılırlar. Matematiksel fonksiyon ağın mimarisi tarafından belirlenir. Üretilen çıktılar belirli bir hata değerinin altına inene kadar ağırlık değerleri değiştirilerek gerekli ağırlık değerlerine ulaşılır. Ağ kendisine gösterilen örneklerle çıktılar arasındaki ilişkiyi ortaya çıkarır ve genelleme yapacak seviyeye gelerek eğitilmiş olur. Ağın çıktısı beklenen çıktı ile karşılaştırılarak hata payı elde edilir. Bu hata payı ağın performansını belirler. Geri yayılma (backpropagation) algoritması ile hata payını istenen değere getirmek amacıyla ağırlıkları ayarlama yoluna gidilebilir. Bu işlem, optimum çözüme ulaşana kadar tekrar edilebilir (Yurtoğlu, 2005).

3.5. Yapay Sinir Ağlarının Tarihçesi

Yapay sinir ağlarının tarihçesi nörobiyoloji konusuna insanların ilgi duyması ve elde ettikleri bilgileri bilgisayar bilimine uygulamaları ile başlamaktadır. 1970'li yıllardan sonra çalışmalar önemli gelişmeler göstermiş ve çok büyük ilerleme kaydedilmiştir. Bu ilerlemeleri kronolojik tanımı aşağıya çıkarılmıştır.

- •1890: İnsan beyninin yapısı ve işlemleri ile ilgili ilk yayının yazılması
- •1911: İnsan beyni bileşenlerinin sinir hücreleri olan nöronlardan oluşması fikri
- •1943: Yapay sinir hücrelerine dayalı hesaplama teorisinin ortaya atılması ve eşik değerli mantıksal devrelerin geliştirilmesi
- •1949: Mümkün olan biyolojik öğrenme prosedürünün bilgisayarlar tarafından gerçekleştirilebilecek şekilde geliştirilmesi
- •1956: 1962 – ADALINE ve Widrow öğrenme algoritmasının geliştirilmesi, tek katmanlı algılayıcı (perseptron) geliştirilmesi
- •1965: İlk makine öğrenmesi kitabının yayımlanması
- •1967-1969:Bazı gelişmiş öğrenme algoritmalarının geliştirilmesi

•1969:Tek katmanlı algılayıcıların problemleri çözme yeteneklerinin olmadığının gösterilmesi

•1969 – 1972: Doğrusal ilişkilendiricilerin geliştirilmesi

•1972: Korelasyon Matris belleğinin geliştirilmesi

•1974: Geriye yayılım modelinin ve danışmansız öğrenmenin geliştirilmesi

•1978: ART modelinin geliştirilmesi

•1982: Kohonen öğrenmesi ve SOM modelinin geliştirilmesi, Hopfield ağları ve çok katmanlı algılayıcının geliştirilmesi

•1984: Boltzman makinesinin geliştirimesi

•1988: RBFN ve PNN modellemelerinin geliştirilmesi

•1991: GRNN modellemesinin geliştirilmesi

1991'den günümüze kadar olan sayısız çalışma yapay sinir ağlarının oldukça yol kat ettiğini göstermektedir (Öztemel, 2006; Elmas, 2007; Mehrotra vd.,1997).

3.6. Yapay Sinir Ağlarının Kullanım Alanları

Temel olarak yapay sinir ağı (YSA) uygulamalarının çoğu aşağıdaki sınıflardan birine girmektedir:

- Öngörü
- Sınıflandırma
- Veri birleştirme
- Veri kavramlaştırılması
- Veri süzülmesi
- Resim veya görüntü işleme (Elmas, 2007)

Bunlar dışında teorik uygulamaların ötesinde günlük hayatta kullanılan finansal konulardan mühendisliğe ve tıp bilimine kadar birçok uygulamadan bahsetmek mümkündür. Bu uygulamaların bazılarını şöyle sıralanabilir:

• 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 parçalarının ömürlerinin ve kırılmalarının tahmin edilmesi

• Kalite kontrolü

• İletişim kanallarındaki geçersiz ekoların filtrelenmesi

• Radar ve sonar sinyalleri sınıflandırılması

• Kan hücreleri reaksiyonları ve kan analizleri sınıflandırma

• Beyin modellenmesi çalışmaları (Öztemel, 2006)

BÖLÜM 4

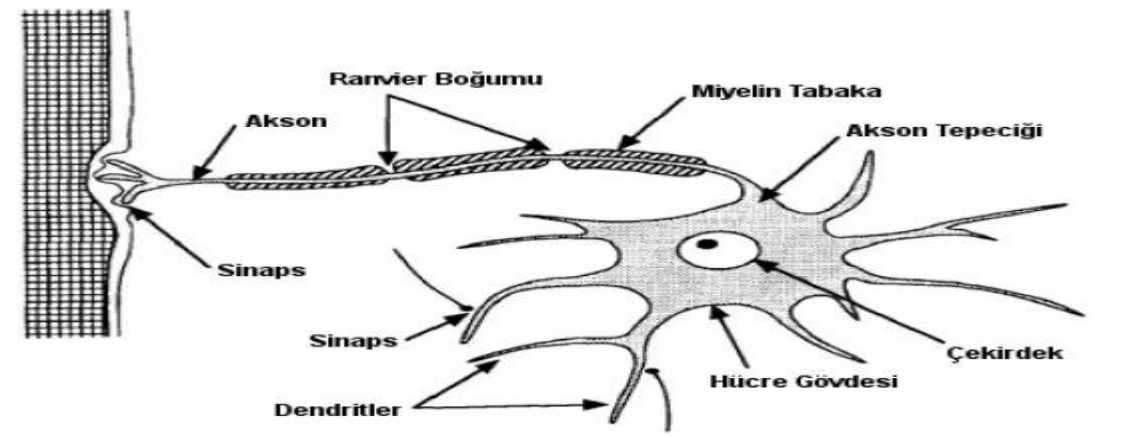
YAPAY SİNİR AĞLARININ YAPISI VE TEMEL ELEMANLARI

4.1. Biyolojik Sinir Hücresi

Yapay sinir ağlarını daha iyi anlayabilmek için biyolojik sinir hücresini ve bu hücrelerin oluşturduğu sinir ağlarının iyi anlaşılması gerekir.

İnsan beyninin en temel parçası, hatırlama, düşünme, her harekette daha önceki deneyimlere başvurma yeteneğini sağlayan kendine özgü sinir hücreleridir. Sinir hücrelerine nöron da denilmektedir. İnsan beyninde yaklaşık 10^{11} sinir hücresi vardır. Her bir biyolojik sinir hücresinin yaklaşık 10000 kadar komşu bağlantısı vardır ve bu sinirlerden uyarı alır. İnsan beyninin çalışma frekansı 100 Hz'dir. İnsan beyninin yetişkin bir insanda ağırlığı yaklaşık 1,3 kg'dır (Elmas, 2007).

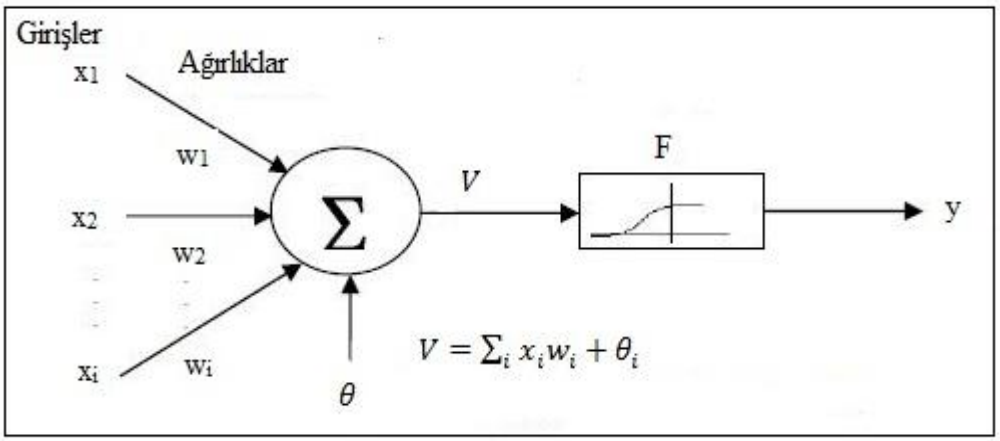
Bir sinir hücresi sinapslar, hücre gövdesi (soma), çekirdek, akson ve dentritlerden oluşur (Şekil 4.1). Sinapslar sinir hücrelerinin birbiri arasındaki bağıntılar olarak görülebilir. Bunlar fiziksel bağlantılar olmayıp bir hücreden diğerine elektrik sinyallerinin geçmesini sağlayan yapılardır. Bu sinyaller somaya gider. Çekirdek bunları işleme tabi tutar. Sinir hücresi kendi elektrik sinyalini oluşturur ve akson aracılığı ile dentritlere gönderir. Dentritlerde bu sinyalleri sinapslara göndererek diğer hücrelere iletilir. Bu özellikte milyarlarca sinir hücresi bir araya gelerek sinir sistemini oluşturmaktadır. Yapay sinir ağları biyolojik hücrelerin bu özelliklerinden yararlanarak geliştirilebilir (Öztemel, 2006; Kulkarni, 1994).



Şekil 4.1. Biyolojik sinir hücresinin yapısı (Freeman and Skapura, 1991)

4.2. Yapay Sinir Hücresi

Yapay sinir ağları birbirine bağlı çok sayıda işlem elemanlarından oluşmuş, genellikle paralel işleyen yapılardır. YSA'lardaki yapay sinir elemanları ve diğer deyişle düğümler basit sinirler olarak adlandırılır (şekil 4.2) Bir YSA birbiriyle bağlantılı çok sayıda düğümden oluşur (Elmas, 2007).



Şekil 4.2. Yapay sinir hücresinin yapısı

Yapay sinir hücresinin yapısı Şekil 4.2'de gösterilmiştir. Şekilde girişler x, ağırlıklar w, toplama fonksiyonu ve θ değeri, aktivasyon fonsiyonu F ve bir çıktı olarak da y görülmektedir. Buradaki girişlerden her biri ağırlık (w) ile çarpılır. Basit olarak bu ürünler θ eşik değeri ile toplanır ve sonucu oluşturmak için aktivasyon fonksiyonu ile işlem yapılır ve y çıktısı alınır. Bir yapay sinir düğümün en basit çalışma biçimi temel olarak bu şekildedir.

4.2.1. Girişler

Girişler ya da diğer adıyla girdiler çevreden alınan bilgilerdir. Bu bilgiler ağa kendinden önceki sinirlerden veya dış dünyadan gelir. Girdiler genellikle ağın öğrenmesi istenen örnekler tarafından belirlenir (Öztemel, 2006).

4.2.2. Ağırlıklar

Ağırlıklar alınan girdilerin yapay sinir ağı üzerindeki etkisini belirleyen katsayılar olarak tanımlanabilir (Elmas, 2007). Her giriş kendine ait bir ağırlığa sahiptir. Yani x_1 girdisi w_1 ağırlığına sahiptir. Ağırlıklar negatif ve pozitif olabileceği gibi sıfırda olabilir.Ağırlıklar girdilerin ağa bağlanma derecelerine göre yani o ağa bağlanmalarının güçlü yada zayıf olması, ağ için önemli yada önemsiz olması ile değişken veya sabit değerler alabilirler.

4.2.3. Toplama fonksiyonu

Toplama fonksiyonu bir hücreye gelen net girdiyi hesaplar. Bunun için değişik fonksiyonlar kullanılmaktadır. En yaygın olanı ağırlıklı toplamı bulmaktır (Öztemel, 2006). Bu yöntemde gelen her girdi kendi ağırlığı ile çarpılarak toplanır. Böylece ağa gelen net girdi bulunmuş olur. Formülize edilmiş hali şu şekildedir:

$$NET = \sum_{i}^{n} x_i w_i \tag{4.1}$$

Bu denklemde daha önce bahsedildiği gibi x girdileri w ağırlıkları n ise bir hücreye gelen toplam girdi sayısını göstermektedir. Fakat yapay sinir ağlarında her zaman bu formülün kullanılması şart değildir.Uygulan yapay sinir ağı modellerinden bazıları kullanılacak toplama fonksiyonunu kendi belirlemektedir. Kullanılan değişik toplama fonksiyonu örnekleri Çizelge 4.1'de verilmiştir. Çizelgede görülebileceği üzere bazı durumlarda gelen girdilerin değeri dikkate alınırken bazı durumlarda ise gelen girdilerin sayısı önemlidir. Bir problem için en uygun toplama fonksiyonunu belirlemek için bulunmuş bir formül yoktur. Genellikle deneme yanılma yolu ile toplama fonksiyonu belirlenmektedir (Öztemel, 2006). Çeşitli toplama fonksiyonları çizelge 4.1'de gösterilmiştir.

Çizelge 4.1. Toplama fonksiyonları örneği (Öztemel, 2006)

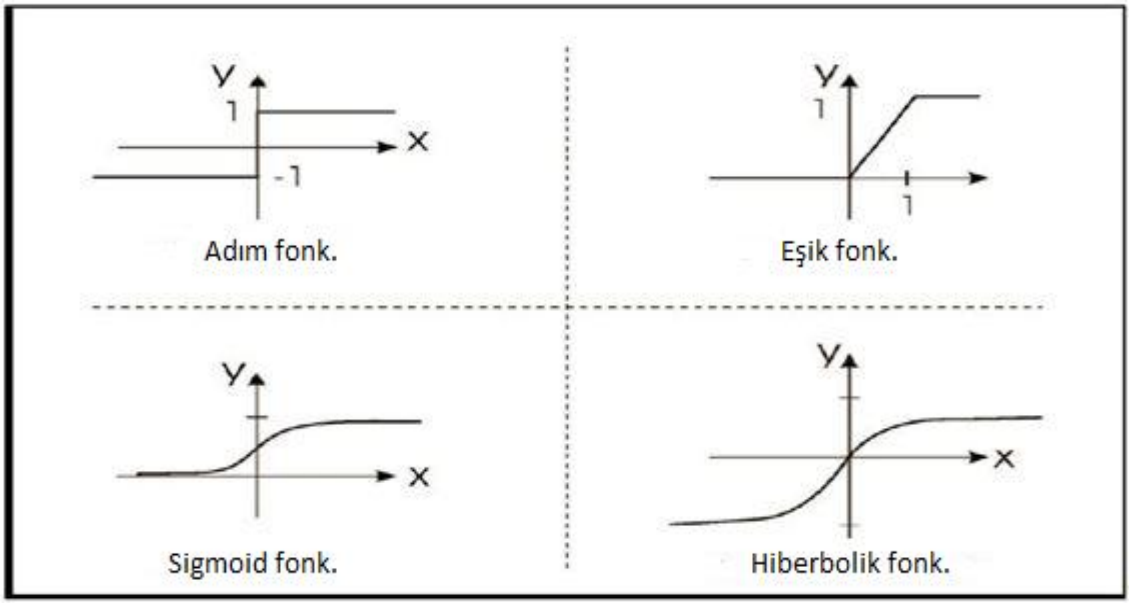
Net Giriş	**Açıklama**
Çarpım $NET = \prod_i x_i w_i$	Ağırlık değerleri girdiler ile çarpılır, bulunan değerler birbiri ile çarpılarak net değerler bulunur.
Maksimum $NET = Mak(x_i w_i)$ i= 1…N	N adet girdiden ağırlıklar ile çarpıldıktan sonra en büyüğü YSA net girdisi kabul edilir.
Minimum $NET = Min(x_i w_i)$ i= 1…N	N adet girdiden ağırlıklar ile çarpıldıktan sonra en küçüğü YSA net girdisi kabul edilir.
Çoğunluk $NET = \sum_i sgn(x_i w_i)$	N adet girdiden ağırlıklar ile çarpıldıktan sonra pozitif ve negatif olanların sayısı bulunur. Büyük olan sayı net girdi olarak kabul edilir.
Kumilatif Toplam $NET = Net(eski) + \sum_i (x_i w_i)$	Hücreye gelen bilgiler ağırlıklı olarak toplanır ve daha önceki bilgilere eklenerek hücrenin net girdisi bulunur.

4.2.4. Aktivasyon (Transfer) fonksiyonu

Aktivasyon yada diğer adıyla transfer fonksiyonu hücreye gelen net girdiyi işleyerek hücrenin bu girdiye karşılık üreteceği çıktıyı belirleyen fonksiyon olup toplama fonksiyonunda olduğu gibi transfer fonksiyonu da çıktıyı hesaplamak için formül kullanır. Bir problem için seçilecek en uygun fonksiyon, tasarımı yapan kişinin denemeleri sonucunda belirlenir. Günümüzde en çok kullanılan 'çok katmanlı algılayıcı' modelidir (Öztemel, 2006). Genel olarak transfer fonksiyonu olarak da sigmoid fonksiyonu kullanılır ve

$$F(NET) = \frac{1}{1+e^{-NET}} \tag{4.2}$$

şeklinde ifade edilir. Bu denklemde NET ifadesi proses elemanına gelen NET girdi değerini göstermektedir.



Şekil 4.3. Çeşitli transfer fonksiyonları

Çeşitli transfer fonksiyonları mevcuttur (Şekil 4.3). Eşik veya basamak işlevleri transfer fonksiyonunun nasıl çalıştığını basit bir biçimde açıklamaktadır. Sinir, etkinlik

işlevinin eşik seviyesinin altında çıkış üretmez. Sinir, etkinlik işlevinin eşik seviyesinin üzerinde çıkış üretir (Elmas, 2007). Bir diğer fonksiyon olan hiberbolik tanjant fonksiyonunda gelen NET girdi değeri tanjant fonksiyonundan geçirilmesi ile hesaplanır (Öztemel, 2006).

4.2.5. Hücre çıktısı

Hücre çıktısı transfer fonksiyonu tarafından belirlenen çıktı değeridir. Bu çıktı dış dünyaya veya başka bir hücreye gönderilir. Hücre kendi çıktısını kendisine girdi olarak da gönderebilir. Bir sinirin bir tek çıkışı vardır (Öztemel, 2006; Elmas, 2007).

4.2.6. Öğrenme

Öğrenme kuralı, Hebbian öğrenme kuralı denilen modelden çıkarılır. Bu modelin temeli, iki düğüm aynı zamanda etkinse aralarındaki bağ gücünün artacağı şeklindedir. Öğrenmenin amacı düğüm girişlerindeki bağlantı ağırlıklarını derlemektir. İstenen sonuçları elde edebilmek içim giriş bağlantılarının ağırlıklarının değiştirme işlemi öğrenme olarak adlandırılabilir (Elmas, 2007).

Danışmanlı ve danışmansız olmak üzere iki eğitim türü vardır. Danışmanlı yada öğretmenli öğrenmede isminden anlaşılabileceği gibi bir öğretmene ihtiyaç vardır. Öğretmenden kasıt ağa gösterilen bir veri alıştırma kümesi ya da ağ sonuçlarının performansını derecelendiren bir gözlemcidir. Belli başlı öğrenme kuralları şunlardır:

• **Hebb Kuralı:** İlk ve en iyi bilinen bu öğrenme kuralı Donald Hebb tarafından 1949'da yazdığı 'The Organization of Behaviour' adlı kitabında tanıtılmıştır. Buradaki temel kural eğer bir sinir başka bir sinirden bir giriş alırsa ve her ikisi de aktif ise(matematiksel olarak aynı işaretli) sinirler arasındaki boyut kuvvetlenir (Elmas,

2007). Yani bir hücre kendisi aktif ise bağlı olduğu hücreyi aktif yapmaya pasif ise pasif yapmaya çalışır (Sağıroğlu vd., 2003).

• **Hopfield Kuralı:** Bu kural Hebb kuralı ile benzerdir. Farklı olarak sadece burada kuvvetlendirme veya zayıflandırmanın genliği belirlenebilmektedir. Buradaki temel kural da şudur: Eğer istenilen çıkış ve girişlerin her ikisi de aktif veya her ikisi de pasif ise, bağlantı boyutlarını öğrenme oranı kadar arttırılır, aksi halde bağlantı boyutu öğrenme oranı kadar azaltılır (Elmas, 2007). Görüldüğü gibi ağırlıkların kuvvetlendirilmesi veya zayıflatılması öğrenme katsayısı yardımı ile gerçekleştirilir. Bu katsayı kullanıcı tarafından atanan, genellikle 0-1 arasında pozitif ve sabit bir sayıdır (Öztemel, 2006).

• **Delta Kuralı:** En çok kullanılan kurallardan biri olan Delta kuralı, Hebb kuralının daha geliştirilmiş halidir. Bu kural bir sinirin gerçek çıkışı ile istenilen çıkış değeri arasındaki farkı azaltmak için giriş bağlantı ağırlıklarının sürekli geliştirme fikrine dayanır ve ağ hatasının karesini minimize etmek için bağlantı boyutlarını değiştirir. Hata bir önceki katmana geri çoğaltılır. Her bir zaman dilimi için bir hata şeklinde bu geri çoğaltma işlemi ilk katmana ulaşılıncaya kadar devam eder. Bu tip ağ 'ileri beslemeli ağ' olarak adlandırılır. Geri yayılım adını bu hata terimini toplama yönteminden türetir. Bu kural ayrıca Windrow-Hoff öğrenme ve en küçük ortalamalar karesi (Lean Mean Square) kuralı olarak da adlandırılır (Elmas, 2007).

• **Kohenen Kuralı:** Bu kuralda ağın hücreleri ağırlıklarını değiştirmek için birbiri ile yarışır. En büyük çıktıyı üreten sinir kazanan sinir olmakta ve bağlantı ağırlıkları değiştirilmektedir. Bu, o sinirin yakınındaki sinirlere karşı daha kuvvetli hale gelmesi demektir. Hem kazanan elemanların hem de komşuları sayılan elemanların ağırlıklarını değiştirmesine izin verilmektedir (Öztemel, 2006).

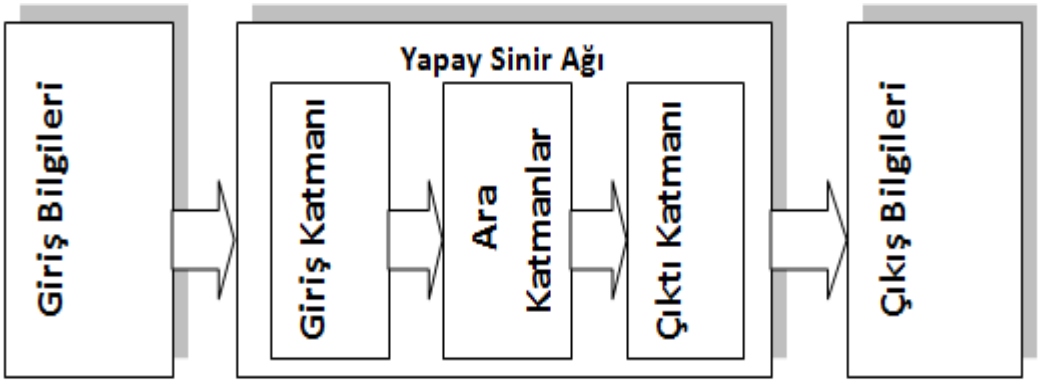
4.3. Yapay Sinir Ağının Yapısı

Yapay sinir hücreleri bir araya gelerek yapay sinir ağlarını oluştururlar. Sinir hücrelerinin bir araya gelmesi rastgele değildir. Genel olarak hücreler 3 katman halinde (şekil 4.4) ve her katman içinde paralel olarak bir araya gelerek ağı oluştururlar (Öztemel, 2006). Bu katmanlar şunlardır:

• **Giriş katmanı:** Dış dünyadan gelen bilgilerinin alınarak ara katmanlara transfer edildiği yerdir.

• **Ara katmanlar:** Giriş katmanından gelen bilgiler burada işlenilerek çıktı katmanına aktarılır. Bir yapay sinir ağının birden çok katmanı olabilir

•**Çıktı katmanı:** Bu katmanda ara katmandan gelen bilgiler işlenerek sunulan girdi seti için üretilmesi gerekilen çıktı üretilir. Çıktı dış dünyaya aktarılır



Şekil 4.4. Yapay sinir ağı katmanları

BÖLÜM 5

YAPAY SİNİR AĞLARININ SINIFLANDIRILMASI

Yapay sinir ağlarını gerek kullanılan ağ mimarisi (ağ topolojisi) gerek kullanılan öğrenme algoritması ve metotlarına göre kendi aralarında sınıflandırmak mümkündür.

5.1. Yapay Sinir Ağlarının Mimarilerine Göre Sınıflandırılması

5.1.1. İleri beslemeli ağlar

1958 yılında Rosenbalt tarafından geliştirilen algılayıcı (perseptron) modeli yapar sinir ağı için ilk mimari yapıdır (Gurney, 1997). Bu yapı basit olarak bir giriş bir çıkış ve bir yada daha fazla ara katmandan oluşur. İleri beslemeli ağlar bilgi akışının girişten çıkışa doğru tek yönde olduğu, birbirine bağlı nöronlardan oluşur. Bu ağlarda, ileri yönde olan bilgi akışı başlangıç noktasına geri gönderilmez ve böyle bir bilgi akış döngüsü oluşturulmaz (Dreyfus, 2005). Bu ağlar girdi ile çıktıyı birleştiren bilginin ileriye doğru aktığı ve herhangi bir geri beslemenin olmadığı ağlardır.

İleri beslemeli ağın en tipik modeli ardışık olarak nöronların bir araya getirilmesi ile oluşur. İleri beslemeli yapay sinir ağlarında hücreler katmanlar şeklinde düzenlenir ve bir katmandaki hücrelerin çıkışları bir sonraki katmana ağırlıklar üzerinden giriş olarak verilir. Bilgi ara katmanda işlenir ve ağ çıkışı belirlenir. Bu yapısı ile ileri beslemeli ağlar doğrusal olmayan statik bir işlevi gerçekleştirir. İleri beslemeli 3 katmanlı YSA'nın, orta katmanında yeterli sayıda hücre olmak kaydıyla, herhangi bir sürekli fonksiyonu istenilen doğrulukta yaklaştırılabilir (Fırat ve Güngör, 2004).

İleri beslemeli ağlarda, ağın toplam davranışındaki nonlineerliği, giriş ve çıkış katmanları arasındaki gizli katmanlardaki nöronların doğrusal olmayan davranışları belirler. Giriş ve çıkış katmanındaki nöron sayıları ele alınan probleme göre belirlenir fakat gizli katmanlardaki nöron sayısını belirlemek için herhangi bir aritmetik yöntem yoktur. Bu sayı deneme yanılma yöntemi ile belirlenmelidir (Efe ve Kaynak, 2000).

5.1.2. Geri beslemeli ağlar

Geri beslemeli yapay sinir ağları, çıktı ve ara katmanlardaki bilgilerin bir önceki ara katmanlara veya girişe yönlendirilerek geri besleme yapan yapay sinir ağlarıdır. Yapılan bu geri beslemeden dolayı bu ağlar ileri beslemeli ağların aksine dinamik bir hal içindedir.

Geri beslemeli ağlarda bir döngü vardır. Bu döngü ileri yönlü bilgilerden en az birini başlangıç hücresine yönlendirir. Çıktı hücresi kendinin fonksiyonu olmadığında bu tür yapılar zaman fonksiyonunun net olarak hesaba katılmasını sağlar. Hücrenin çıktısı aynı zamanda kendisinin fonksiyonu olamaz ancak eksi değerinin fonksiyonu olabilir. Geri beslemeli ağlarda her bağlantıya gecikme atanır. Her gecikme başlangıç zamanının çok katlı türevidir. Döngünün sınırlarındaki gecikmelerin toplamı sıfır olmalıdır. Süreksiz zaman yinelemeli sinir ağları, hücrelerin fonksiyonlarının birleşimi ve zaman gecikmelerini bağlantılara birleştiren doğrusal olmayan süreksiz zaman yineleme denklemlerine göre çalışır (Dreyfus, 2005).

5.2.Yapay Sinir Ağlarının Öğrenme Metotlarına Göre Sınıflandırılması

5.2.1. Danışmanlı öğrenme

Danışmanlı öğrenmede sistemin olayı öğrenebilmesi için bir öğreticiye yani danışmana ihtiyaç vardır. Bu öğretici ile sisteme, öğretilmesi istenilen olay ile ilgili

örnekler girdi-çıktı seti olarak verilir. Yani sisteme her örnek için hem girdiler hem de o girdiler karşılığında oluşturulması gereken çıktılar gösterilir. Ağın görevi girdilerin öğreticinin belirlediği çıktılara haritalamaktır. Bu sayede olayın girdileri ile çıktıları arasındaki ilişkiler öğrenilmektedir (Öztemel, 2006).

Birçok uygulamada, ağa gerçek veriler uygulanmak zorundadır. Bu eğitim safhası uzun zaman alabilir. Sinir ağı, belirli bir sıralamadaki girişler için istenen istatistiksel doğruluğu elde ettiği zaman eğitme işlemi tamamlanmış kabul edilir ve eğitme işlemi bitirilir. Öğrenim aşaması tamamlandıktan sonra ağ kullanılmaya başlanıldığında, bulunan ağırlıkların değeri sabit olarak alınır ve bir daha değiştirilmezler. Bazı ağ yapılarında ağ çalışırken çok düşük oranda eğitmeye izin verilir. Bu işlem ağların değişen koşullara uyum sağlamasına yardımcı olur (Elmas, 2007).

5.2.2. Danışmansız öğrenme

Danışmansız öğrenmede sistemin öğrenmesine yardımcı olan bir öğretici yoktur. Sistemde sadece girdiler gösterilir. Örnekler arasındaki parametreleri sistemin kendi kendisine öğrenmesi beklenir. Bu daha çok sınıflandırma problemleri için kullanılan bir yöntemdir. Fakat sistemin öğrenmesi bittikten sonra çıktıların ne anlama geldiğini gösteren etiketlendirmenin kullanıcı tarafından yapılması gerekmektedir (Öztemel, 2006).

Danışmansız öğrenmede ağ istenen dış verilerle değil girilen bilgilerle çalışır. Bu tür öğrenmede gizli sinirler dışarıdan yardım almaksızın kendilerini örgütlemek için bir yol bulmalıdırlar. Bu yaklaşımda, verilen giriş için önceden bilinebilen performansını ölçebilecek ağ için hiçbir çıkış örneği sağlanmaz, yani ağ yaparak öğrenir. Danışmansız öğrenmeye Hebbian öğrenme kuralı, Grossberg öğrenme kuralı ve Kohonen'in özörgütlemeli harita ağı örnek olarak verilebilir. Kohonen'in özörgütlemeli harita ağında sinirler öğrenmek için elverişli durum yada ölçülerini

güncellemek için yarışırlar. En büyük çıktı ile işlenilen sinir, kazananı belirler ve komşularına bağlantı boyutlarını güncellemek için izin verir ve güncellemeler bu şekilde devam eder (Elmas, 2007).

5.2.3. Karma öğrenme

Kısmen danışmanlı veya kısmen danışmansız olarak öğrenme yapan ağlardır. Radyal tabanlı yapay sinir ağları (RBN) ve olasılık tabanlı ağlar (PBN) bunlara örnek gösterilebilir (Öztemel, 2006)

5.3.Yapay Sinir Ağlarında Öğrenmenin Uygulamaya Göre Sınıflandırılması

5.3.1. Çevrimiçi öğrenme

Gerçek zamanlı çalışabilen sistemlerdir. Bu sistemde, gerçek zamanda çalışırken bir taraftan fonksiyonlarını yerine getirmekte diğer taraftan öğrenmeye devam etmektedir. ART ağı ve Kohonen ağının öğrenmesi buna örnektir(Öztemel. 2006).

5.3.2. Çevrimdışı öğrenme

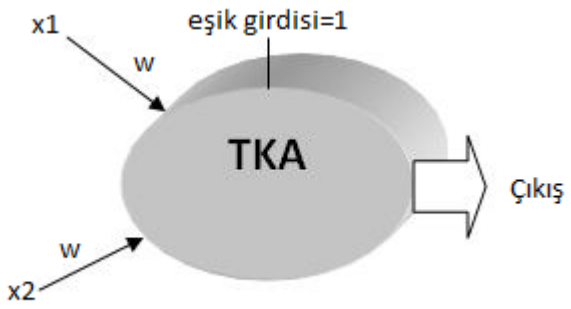
Çevrimdışı öğrenmede sistemler kullanılmaya alınmadan önce örnekler üzerinden eğitilirler. Ağ eğitildikten sonra gerçek hayatta kullanıma alındığında artık öğrenme olmamaktadır. Ağın öğrenmesi gereken yeni bilgiler söz konusu olduğunda kullanımdan çıkarılmakta ve çevrimdışı olarak yeniden eğitilmektedir. Eğitim tamamlanınca ağ tekrardan kullanıma alınır. Yapay sinir ağlarında yaygın olarak kullanılan delta öğrenme kuralı bu tür öğrenmeye örnek olarak verilebilir (Öztemel, 2006).

BÖLÜM 6

YAPAY SİNİR AĞLARINDA KULLANILAN MODELLER

6.1 Tek Katmanlı Algılayıcılar

Tek katmanlı yapay sinir ağları (TKA) sadece giriş ve çıkış katmanlarından oluşur. Her ağın bir yada daha fazla girdi ve çıktısı vardır. Çıktı üniteleri bütün giriş ünitelerine bağlıdırlar. Her bağlantının bir ağırlığı vardır ve ağlarda proses elemanlarının ve ağın çıktısının sıfır olmasını önleyen bir de eşik değeri vardır (Öztemel, 2006).



Şekil 6.1. İki girdi ve bir çıktıdan oluşan TKA modeli

Ağın çıktısı, ağırlıklandırılmış girdi değerlerinin eşik değeri ile toplanması sonucu bulunur (Şekil 6.1). Bu, giriş bir aktivasyon fonksiyonundan geçirilir ve ağın çıktısı elde edilir ve,

$$Ç = f[\sum_i^n w_i x_i + \theta] \qquad (6.1)$$

şeklinde ifade edilir. Tek katmanlı algılayıcılarda çıkış fonksiyonu doğrusal fonksiyondur. Dolayısıyla ağa gönderilen örnekler iki sınıfa paylaştırılır ve iki sınıfı birbirinden ayıran doğru bulunmaya çalışılır. Bu yüzden de eşik değer fonksiyonu kullanılmaktadır. Ağın çıktısı 1 veya -1 değerini alır ve bu 1 ve -1 değerleri sınıfları temsil eder.

$$f(x) = \begin{cases} 1 & eğerÇ < 0 \\ -1, & aksitakdirde \end{cases} \tag{6.2}$$

ile tanımlanır. 6.2'ye göre ağa gelen toplam girdi pozitif ise ağa sunulan örnek 1 sınıfına negatif olması durumunda -1 sınıfında olacaktır. Sıfır olması durumunda bu sınıf ayrımı tasarıcının kabulüne bağlıdır. Sınıf ayracı olarak adlandırılan doğru Şekil 6.2'de gösterilmiştir. Bu doğrunun denklemi

$$w_1.x_1 + w_2.x_2 + \theta = 0 \tag{6.3}$$

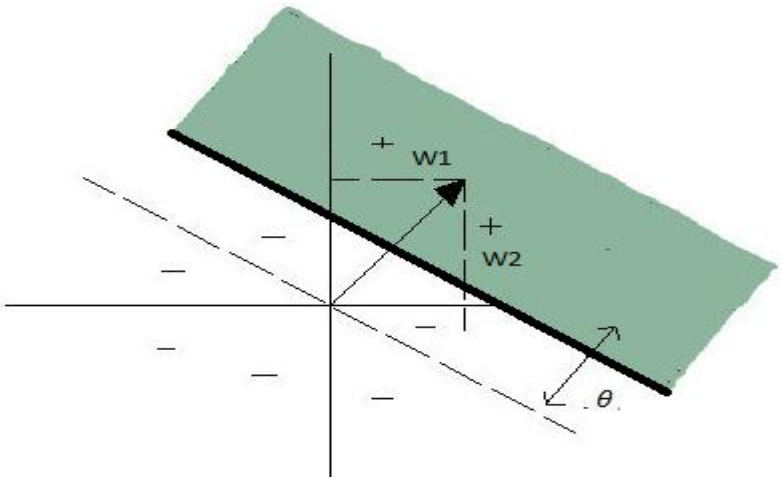
şeklindedir. Bağıntıdaki $x_2$6.4'de olduğu üzere formüle edilebilir.

$$x_2 = -\left({}^{w_1}/_{w_2}\right)x_1 - {}^{\theta}/_{w_2} \tag{6.4}$$

6.3'deki x_1 değeri çekilirse,

$$x_1 = -\left({}^{w_2}/_{w_1}\right)x_2 - {}^{\theta}/_{w_2} \tag{6.5}$$

elde edilir. Bu iki formülden sınıf ayracı doğrusu çizilebilir ve oluşturulan ağda sınıf ayracı denilen doğruyu (şekil 6.2) her iki gurubu en iyi şekilde ayıracak şekilde belirlemek gerekir.



Şekil 6.2. Sınıf ayracı ve ağırlıkların geometrik gösterimi

Sınıf ayracının en iyi şekilde ayarlanmasını şu şekilde daha iyi açıklanabilir. Zaman biriminde ağırlık Δw kadar değiştirilirse şu ifadeye ulaşılır:

$$w_i(t+1) = w_i(t) + \Delta w_i(t) \tag{6.6}$$

Öğrenme sırasında bu değişim her iterasyonda gerçekleştirilerek sınıf ayracının en doğru pozisyonu bulunmaya çalışılır (Öztemel, 2006). Ağırlıkların değiştirilmesi doğrunun eğiminin değiştirilmesi demektir. Bu yeterli olmazsa eşik değerini de değiştirmek gerekebilir. Böylece doğrunun sınıflar arasında kayması sağlanabilir ve transfer fonksiyonunun konumu belirlenebilir. Bu durumda t anında eşik değerinin şu şekilde değiştirilmesi gerekmektedir:

$$\theta(t+1) = \theta(t) + \Delta\theta \tag{6.7}$$

Öğrenme sırasında eşik değeri de ağırlıkların olduğu gibi her iterasyonda $\Delta\theta$ kadar değiştirilir.

6.1.1. Perseptron

1958 yılında Rosenblatt tarafından örüntü yada diğer deyişle şekil sınıflandırma amacı ile geliştirilmiştir (Rosenblatt, 1958). Perseptron basit algılayıcı modelidir ve bir sinir hücresinin birden fazla girdiyi alarak bir çıktı üretmesi prensibine dayanmaktadır. Ağın çıktısı bir veya sıfırdan oluşan mantıksal bir değerdir (Öztemel, 2006).

Perseptron eğitilebilen tek bir yapay sinir hücresinden oluşur. Eğitilebilirin anlamı ağırlıkların değiştirilebilir olmasıdır. Girdiler hücreye gösterilir ve her girdi setine karşılık gelen çıktı seti de ağa gösterilip ağın öğrenme kuralına göre çıktı değeri hesaplanır. Eğer çıktı değeri olması gereken değer değilse ağırlıklar ve eşik değerler değiştirilir. Bu değişimin nasıl yapılacağı kullanılan öğrenme kuralına bağlıdır. Girdilere karşılık gelen çıktı değerleri bir veya sıfırdan oluşur (Öztemel, 2006; Kasabov 1998). Perseptronun öğrenme kuralı şu şekildedir:

İlk olarak ağa girdi seti ve ona karşılık gelen istenen çıktı gösterilir. Girdi değeri X_1, X_2,…,X_N gibi birden fazla değer olabilir. Çıktı değeri ise 1 ve 0 değerlerinden birisini alır. Perseptrona gelen net girdi,

$$NET = \sum_{i=1}^{n} w_i x_i \tag{6.8}$$

şeklindedir. Daha sonra perseptronun çıktısı hesaplanır. Net girdinin eşik değerinden büyük olup olmamasına göre çıktı değeri 1 veya 0 değerlerinden birini alır.

$$Ç = \begin{cases} 1 & eğer NET > \theta \\ 0 & eğer NET \leq \theta \end{cases} \tag{6.9}$$

Eğer hesaplanan çıktı ile beklenen çıktı aynı olursa ağırlıklarda herhangi bir değişiklik olmaz. Ağ, beklenmeyen bir çıktı üretir ise iki durum söz konusudur:

a) Ağın beklenen çıktı değeri 0 değeridir fakat NET girdi eşik değerinin üstündedir ve alınan çıktı 1 değeridir. Bu durumda ağırlık değerleri azaltılmaktadır. Ağırlıkların değişim oranı girdi değerlerinin belli bir oranı kadardır ve vektörel olarak,

$$W_n = W_0 - \lambda X \tag{6.10}$$

6.10'daki gibi ifade edilebilir. Burada λ öğrenme katsayısıdır. Ağırlıkların değişim miktarlarını belirlemekte ve sabit bir değer olarak alınmaktadır.

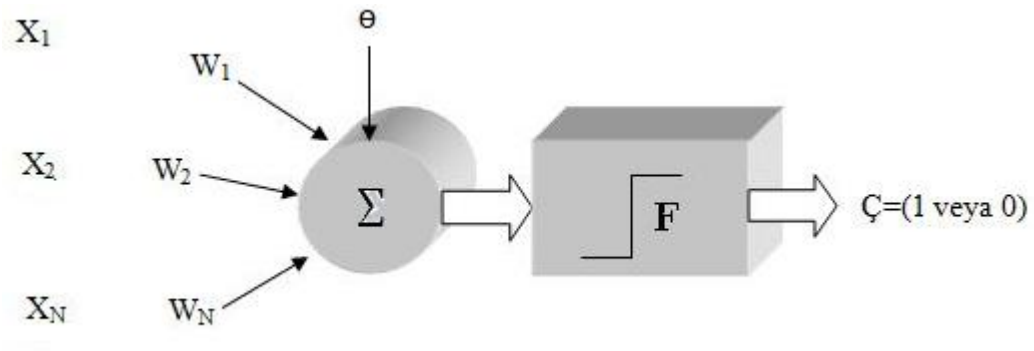
b) Ağın beklenen çıktı değerinin 1 olması ve ağın gerçek çıktısının 0 olması durumunda net girdi eşik değerinin altındadır. Bu durumda ağırlık değerlerinin arttırılması gereklidir ve bunun için aşağıdaki formül kullanılır:

$$W_n = W_0 + \lambda X \tag{6.11}$$

Perseptron bu anlatılan öğrenme aşamalarını bütün girdi setindeki örnekler için doğru sınıflandırmalar yapana kadar tekrarlamaktadır (Öztemel, 2006).

6.1.2 ADALINE modeli

Bu model 1959 yılında geliştirilmiş olup adaptif doğrusal eleman (Adaptive Linear Element) ağının kısaltılmış şeklidir. Genel olarak bir proses elemanından oluşan bir ağdır (Widrow and Hoff, 1960). Bu ağ modeli en küçük ortalamaların karesi (least mean square, LMS) yöntemine dayanır. Öğrenme kuralına delta kuralı da denir. Bu kurala göre ağın çıktısın beklenen çıktı değerine göre hatasını en aza indirecek şekilde ağırlıkların değiştirilmesi yoluna gidilir (Öztemel, 2006)..



Şekil 6.3. ADALINE Ünitesi

Şekil 6.3'te ADALINE ünitesi gösterilmiştir. Girdiler (X_1, $X_{2,\ldots,}X_N$) olup her girdiye karşılık gelen ağırlıklar (W_1, W_2,…,W_N) ve çıktının sıfırdan farklı bir değer olmasını sağlayan eşik değeri θ şekil üzerinde görülmektedir. ADALINE ağının öğrenme kuralı şu şekilde izah edilebilir:

Bu ağ modelinde en küçük kareler yöntemi kullanılarak öğrenme gerçekleştirilir. Perseptron algoritmasına çok benzer. Öğrenme kuralı yapay sinir ağlarında genel öğrenme prensibine göre çalışır. Girdilere göre çıktılar hesaplanır ve ağırlıklar çıktıya göre değiştirilir. Burada net girdi şu şekilde hesaplanmaktadır:

$$NET = \sum_{i=1}^{n} w_i x_i + \theta \tag{6.12}$$

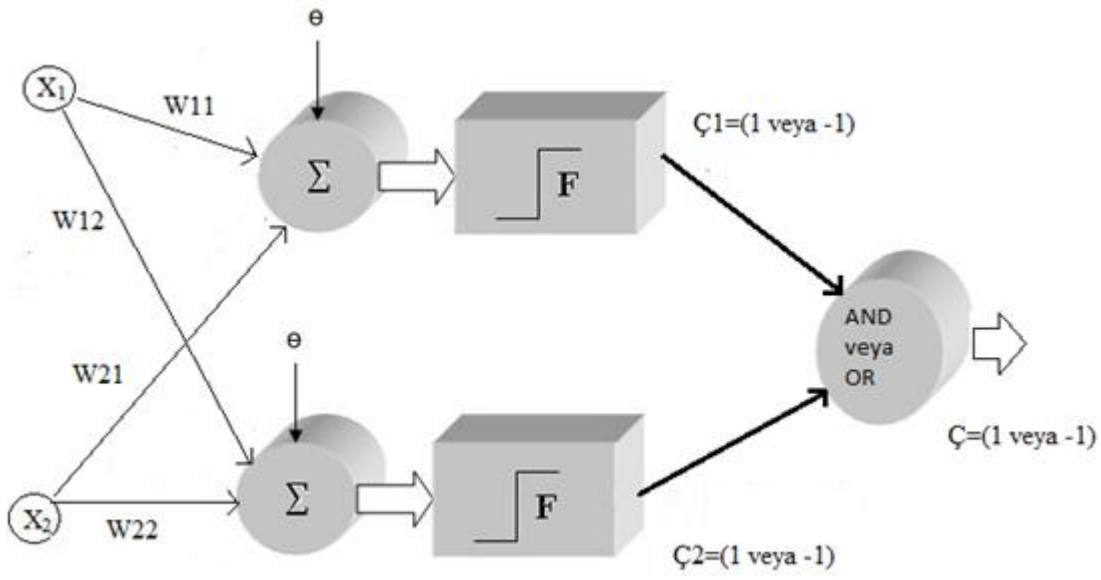
Buradan eğer net değer sıfır ve sıfırdan büyükse çıktı 1 aksi halde -1 değerini almaktadır. Beklenen değer ile hatanın ürettiği değer arasındaki fark hatayı vermektedir. Amaç bu hatayı en aza indirmektir. Bunun içinde her seferinde ağa farklı örnekler gösterilip ağırlıklar hatayı azaltacak şekilde ayarlanır. Zaman içinde hata olması gereken en küçük değere düşmektedir. Bu ayarlamayı bir t anı için şu şekilde formüle etmek mümkündür:

$$W_i(t) = W_i(t-1) + a \times E \times X_i \tag{6.13}$$

6.13'deki bağıntıda $W_i(t)$ ağırlıkların t zamanındaki yeni değerlerini, $W_i(t-1)$ ağırlıkların değişmeden önceki (t-1) zamanındaki değerlerini, a öğrenme katsayısını, E beklenen çıktı ile gerçek değerin arasındaki farktan oluşan hatayı ve X'de girdileri göstermektedir. Benzer şekilde eşik değeri de ayarlanılabilmektedir (Öztemel, 2006).

6.1.3. MADALINE modeli

MADALINE ağları birden fazla bir araya gelmiş ADALINE ağlarıdır. Genel olarak iki katmandan oluşurlar (Şekil 6.4). Her katmanda değişik sayıda ADALINE ünitesi bulunur. Ağ çıktısı 1 ve -1 değerleri ile gösterilir. Her biri bir sınıfı temsil eder (Öztemel, 2006).MADALINE ağlarında öğrenme kuralı ADALINE ağındaki gibidir. Burada sadece son kısımda AND ve OR sonlandırıcısı vardır. AND sonlandırıcısı olması durumunda bütün ADALINE ünitelerinin 1 üretmesi sonucu MADALINE ağının çıktısı 1 olur aksi takdirde -1 değerini alır. OR sonlandırıcı varsa ADALINE ünitelerinden birisinin 1 değerini üretmesi MADALINE ağının çıktısının 1 olması için yeterlidir.



Şekil 6.4. İki ADALINE ağından meydana gelmiş MADALINE ağı

6.2. Çok Katmanlı Algılayıcılar (ÇKA)

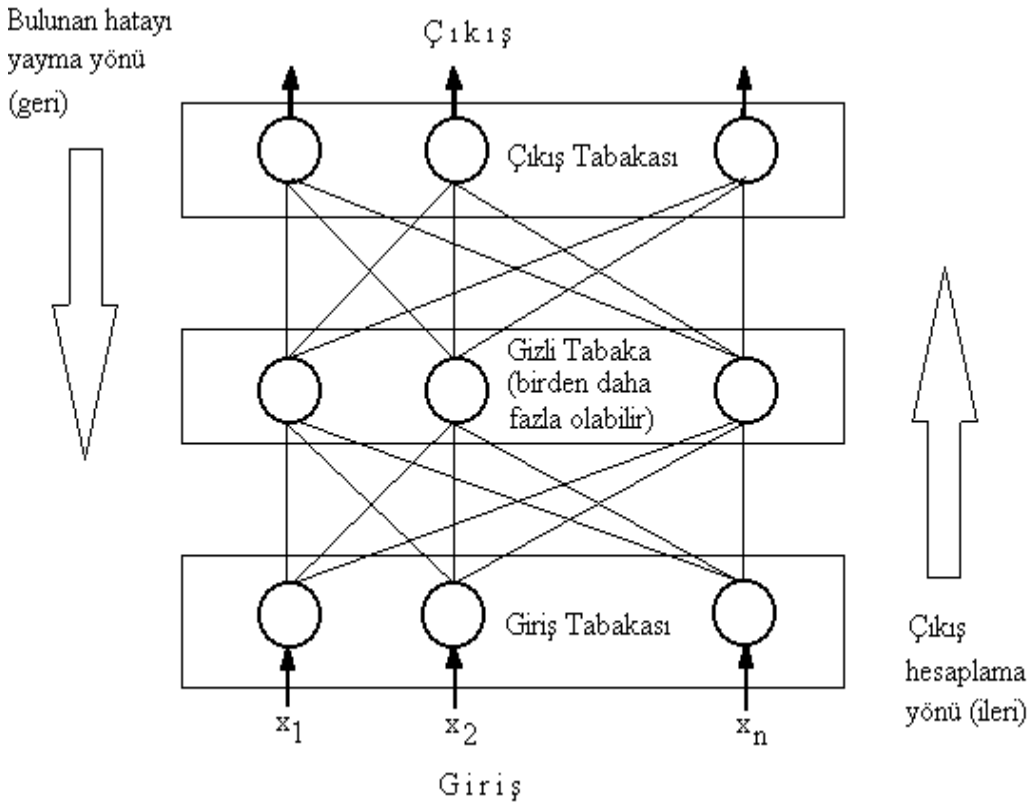
Yapay sinir ağlarında girdi ve çıktılar arasında doğrusal olmayan bir ilişki varsa öğrenmenin mümkün olması için çok katmanlı algılayıcı modeline ihtiyaç vardır. Bu modelin gelişiminde XOR problemi büyük rol oynamıştır. (Çizelge 6.1)

Çizelge 6.1. XOR problemi

Girdi 1	Girdi 2	Çıktı
0	0	0
0	1	1
1	0	1
1	1	0

Doğrusal olmayan XOR probleminin çözümü 1986 yılında Rumelhart ve arkadaşları tarafından geliştirilen hata yayma modeli veya diğer adıyla geriye yayma modeli (backpropogation network) ile çözülmüştür (Rumelhart, et al., 1986).

Çok katmanlı algılayıcı modeli ile yapay sinir ağlarında büyük bir gelişme yaşanmıştır ve bu model günümüzde mühendislik problemlerinin tümüne çözüm sunar hale gelmiştir. Özellikle sınıflandırma, tanıma ve genelleme yapmayı gerektiren problemler için önemli bir çözüm aracıdır. Modelde delta öğrenme kuralı kullanılır. Temel amaç ağın beklenen çıktısı ile üretilen çıktı arasındaki hatayı en aza indirmektir. Bunu da hatayı ağa geri yayarak gerçekleştirir (Öztemel, 2006).



Şekil 6.5. Çok katmanlı algılayıcı yapısı

Çok katmanlı algılayıcı (ÇKA), girdi, ara ve çıkış katmanlarından meydana gelmektedir (Şekil 6.5). Girdi katmanı dış dünyadan gelen bilgilerin ara katmana gönderildiği katmandır. Burada herhangi bir işlem meydana gelmez.

Ara katmanda girdi katmanından gelen bilgiler işlenir ve bir sonraki katmana gönderilir. Birden fazla ara katman kullanılabilir. Çıktı katmanında ara katmandan gelen bilgiler işlenilerek çıktı üretilir. Bu çıktılar üretilirken ağ kendisine sunulan örneklerden genelleme yaparak problem uzayını temsil eden bir çözüm uzayı üretir sonra gösterilen benzer örnekler için bu çözüm uzayı sonuçlar ve çözümler üretebilmektedir (Öztemel, 2006).

6.2.1. ÇKA ağının öğrenme kuralı

ÇKA ağları danışmanlı öğrenme stratejisine göre çalışırlar. Bu ağlara öğrenme sırasında hem girdiler hem de o girdilere karşılık gelmesi beklenen çıktılar gösterilir. ÇKA ağının öğrenme kuralı Delta öğrenme kuralının genelleştirilmiş halidir. Öğrenmenin gerçekleşebilmesi için eğitim seti ve örneklerden oluşan bir sete ihtiyaç vardır. Bu setlerde girdiler ve girdilere karşılık gelen çıktılar belirtilmiş haldedir (Öztemel, 2006).

Genelleştirilmiş Delta kuralı, ağın çıktısını hesaplama safhası olan ileriye doğru hesaplama ve ağırlıkları değiştirme safhası olan geriye doğru hesaplama olmak üzere iki safhadan oluşur.

İleriye doğru hesaplama safhası eğitim setindeki örneğin girdi katmanına gösterilmesi ile başlar. Girdi katmanında herhangi bir işlem olmaz. Gelen girdiler ara katmana hiçbir değişiklik olmadan yönlendirilir. Yani, girdi katmanında k. proses elemanının çıktısı,

$$Ç_k^i = G_k \tag{6.14}$$

şeklindedir. Daha sonra ara katmana gelen net girdi,

$$NET_j^a = \sum_{k=1}^{n} A_{kj}\, Ç_k^i \tag{6.15}$$

ile hesaplanabilir. A_{kj} , k. girdi elemanını j. ara katman elamanına bağlayan bağlantının ağırlık değeridir ve j. ara katman elemanının çıktısı net girdinin transfer fonksiyonundan geçirilmesi ile hesaplanır. Transfer fonksiyonu olarak türevi alınabilen bir fonksiyon kullanılmalıdır. Genelde sigmoid fonksiyonu tercih edilir (Öztemel, 2006). Sigmoid fonksiyonu kullanılması halinde çıktı,

$$Ç_j^a = \frac{1}{1+e^{-(NET_j^a + \beta_j^a)}} \quad (6.16)$$

ile bulunur. β_j ara katmanda bulunan j. elemana bağlanan eşik değer ağırlığını göstermektedir. Bu eşik değer ünitesinin çıktısı sabit olup 1'e eşittir. Ağırlık değerinin konulmasının nedeni Sigmoid fonksiyonunun oryantasyonunu belirlemektir.Ara katmandaki ve çıkış katmanındaki hücreler için kendilerine gelen NET girdinin hesaplanması aynı şekilde devam eder. Çıktı katmanındaki çıktı değerleri alınınca ağın ileri hesaplama işlemi tamamlanmış olur.Geriye doğru hesaplama safhasında ise ağın ürettiği çıktılar ile beklenen çıktılar (B_1, B_2,..,B_N) karşılaştırılır. Aradaki fark hata olarak kabul edilir ve bu hata azaltılmalıdır. Bu yüzden de hata, ağırlık değerlerine dağıtılıp sonraki iterasyonlarda bu hatanın azaltılması sağlanır. Çıktı katmanındaki m. proses elemanı için oluşan hata,

$$E_m = B_m - Ç_m \quad (6.17)$$

şeklinde formüle edilebilir. Bir hücre için bulunan bu hata, çıktı katmanındaki toplam hatayı (TH) bulmak için toplanır. Negatif değerlerden kurtulmak için kareleri alınıp sonra karekök alınmalıdır.

$$TH = {}^{1}/_{2} \sum E_m^2 \quad (6.18)$$

Toplam hatayı minimuma indirmek için bu hatanın kendisine neden olan hücrelere dağıtılması gereklidir. Buda ağırlıkların değiştirilmesi anlamına gelmektedir. Ağdaki ağırlıklar, ara katman ve çıktı katmanı arasındakiler, ara katmanlar ve ara katman ile girdi katmanı arasındaki ağırlıklardır (Öztemel, 2006; Elmas, 2007).

Ara katman ile çıktı katmanı ele alırsa, ara katmandaki j. eleman ve çıktı katmanındaki m. eleman için ağırlık değişim miktarı ΔA^a'nın t zamanı için değişim miktarı şudur:

$$\Delta A_{jm}^{a}(t) = \lambda \delta_m \text{Ç}_j^{a} + \alpha \Delta A_{jm}^{a}(t-1) \quad (6.19)$$

λöğrenme katsayısı, α momentum katsayısıdır. Öğrenme katsayısı ağırlıkların değişim miktarıdır. Momentum katsayısı ise ÇKA ağının öğrenme sırasında yerel bir optimum noktaya takılıp kalmaması için ağırlık değişim değerinin belirli bir oranda bir sonraki değişime eklenmesini sağlar. δ_m ise m. çıktı ünitesinin hatasıdır (Öztemel, 2006). Bu hata terimi şu şekilde hesaplanır:

$$\delta_m = f'(NET).E_m \quad (6.20)$$

Sigmoid fonksiyonu kullanıldığı takdirde şu şekli alır:

$$\delta_m = \text{Ç}_m(1 - \text{Ç}_m).E_m \quad 6.21)$$

Değişim miktarını da hesapladıktan sonra ağırlıkların t. iterasyondaki yeni değeri şu hale gelir:

$$A_{jm}^{a}(t) = A_{jm}^{a}(t-1) + \Delta A_{jm}^{a}(t) \quad (6.22)$$

Eşik değer ünitesinin de ağırlıklarını değiştirmek gereklidir. Bunun için de öncelikle değişim miktarının hesaplanması gerekir. Çıktı katmanındaki hücrelerin ağırlıkları $\beta^{\text{ç}}$ ve bu ünite çıktısının sabit ve 1 olması nedeni ile değişim miktarı,

$$\Delta\beta_m^{\text{ç}}(t) = \lambda \delta_m + \alpha \Delta\beta_m^{\text{ç}}(t-1) \quad (6.23)$$

ile ifade edilir. Eşik değerinin t. iterasyondaki ağırlığının yeni değeri de şu şekli alır:

$$\beta_m^{\text{ç}}(t) = \beta_m^{\text{ç}}(t-1) + \Delta\beta_m^{\text{ç}}(t) \quad (6.24)$$

Ara katmanlar arası veya ara katman ve girdi katmanı arasındaki ağırlıkların değişiminde, ağırlık değişimi ΔA^{i} ile ifade edilip şu formül yazılabilir:

$$\Delta A_{kj}^{i}(t) = \lambda \delta_{j}^{a} Ç_{k}^{i} + \alpha \Delta A_{kj}^{i}(t-1) \quad (6.25)$$

δ^{a} hata terimi şu şekilde hesaplanır:

$$\delta_{j}^{a} = f'(NET) \sum_{m} \delta_{m} A_{jm}^{a} \quad (6.26)$$

Yine Sigmoid fonksiyonu kullanıldığı varsayılırsa hata değeri,

$$\delta_{j}^{a} = Ç_{j}^{a} \left(1 - Ç_{j}^{a}\right) \sum_{m} \delta_{m} A_{jm}^{a} \quad (6.27)$$

şeklini alır.Ağırlıkların yeni değeri de şu hali alır:

$$A_{kj}^{i}(t) = A_{kj}^{i}(t-1) + \Delta A_{kj}^{i}(t) \quad (6.28)$$

Ara katman eşik değer ağırlıkları β^{a} ile gösterildiği taktirde değişim miktarı şu olur:

$$\Delta \beta_{j}^{a}(t) = \lambda \delta_{j}^{a} + \alpha \Delta \beta_{j}^{a}(t-1) \quad (6.29)$$

Ağırlıkların t. iterasyondaki yeni değeri de şu hali alır:

$$\beta_{j}^{a}(t) = \beta_{j}^{a}(t-1) + \Delta \beta_{j}^{a}(t) \quad (6.30)$$

6.2.2 ÇKA ağının performansının ölçülmesi

Yapay sinir ağlarının performansı öğrenme yeteneğinin ölçülmesidir. Ağın eğitim sırasında kendisine gösterilen bütün örneklere doğru cevap üretmesi performansının iyi olması anlamına gelmez. Bu yüzden öğrenen ağların daha önce görmedikleri örnekler karşısında da beklenen performansı gösterip göstermedikleri

ölçülmelidir. Bunun için genellikle ağın eğitildiği problem üzerinden hem eğitimde kullanılacak hem de test esnasında kullanılacak örnekler seçilir. Eğitim sırasında ağa sadece eğitim setindeki örnekler gösterilir. Ağ eğitimi tamamlayınca ağa hiç görmediği test setindeki örnekler gösterilir (Öztemel, 2006). Ağın performansı bu görmediği örneklere ürettiği cevaplar üzerinden ölçülür,

$$P = \frac{D}{T} \times 100 \tag{6.31}$$

ile tayin edilir. Burada D test setinden doğru olarak cevaplandırılan örnek sayısını, T test setinde bulunan toplam örnek sayısını, P ise performans oranını göstermektedir. Eğer performans oranı istenilen düzeyde veya kabul edilebilir bir değerde değilse, ağ eğitim setindeki tüm örnekleri doğru cevaplasa bile iyi öğrenmiş denilemez. O zaman eğitime biraz daha devam etmek gerekebilir. Eğitim iterasyonlarının arttırılmasına rağmen hala performans düzelmiyorsa örneklerin problem uzayını iyi temsil etmedikleri veya ağın parametrelerinin ve topolojisinin iyi seçilmediği anlaşılır (Öztemel, 2006).

6.3. LVQ (Linear Vector Quantization) Modeli

6.3.1. LVQ ağının yapısı ve özellikleri

LVQ ağı Kohonen tarafında 1984 yılında geliştirilen bir ağdır. n boyutlu bir vektörü bir vektörler setine haritalama temel çalışma prensibidir. Bu ağ ile bir vektörün belirli sayıda vektörle gösterilmesi amaçlanmaktadır. Öğrenme ile de girdi vektörünün hangi vektör seti tarafından temsil edilmesi gerektiğinin bulunması istenilir.. Bu vektör setine referans vektörü seti denilirse LVQ ağının görevi öğrenme yolu ile bu referans vektörleri belirlemektir. Yani girdi vektörlerinin üyesi olabilecekleri vektör sınıfını belirlemektir (Öztemel, 2006).

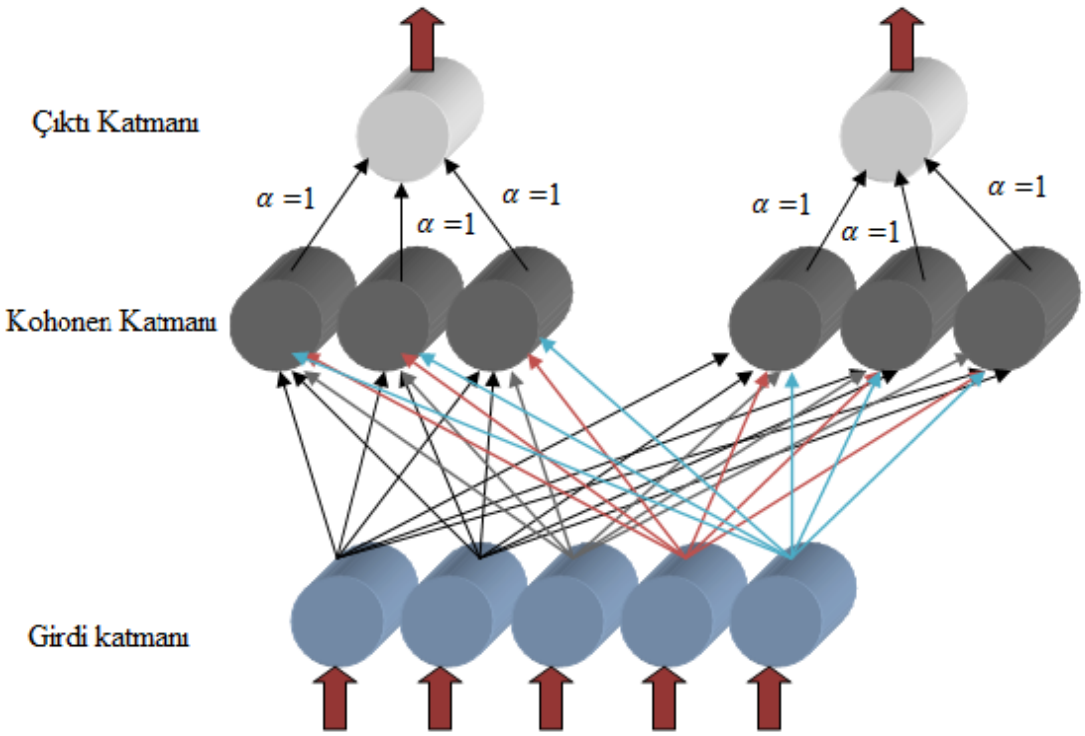
LVQ ağları genellikle sınıflandırma problemlerinin çözümünde kullanılmaktadır. Çıktılardan sadece birisi 1 diğerleri 0 değerini alır. Çıktının 1 olması

girdinin ilgili çıktının temsil ettiği sınıfa ait olduğu demektir.Eğitim sırasında girdinin sınıflara ayrılması en yakın komşu kuralına göre gerçekleştirilmektedir. Girdi vektörleri ile referans vektörleri arasındaki en kısa mesafe aranmakta ve girdi vektörünün en kısa mesafede bulunan vektör gurubuna ait olduğu varsayılır.

Ağın ağırlıklarını değiştirme yolu ile girdileri doğru sınıflara ayıracak referans vektörleri belirlenmektedir. Kullanılan öğrenme stratejisi destekleyici öğrenmedir. Çıktı değerinin belirlenmesinde ise 'kazanan her şeyi alır' (winner takes all) stratejisi uygulanmaktadır. Ağ eğitilirken her iterasyonda ağın ürettiği çıktının değeri yerine sadece doğru olup olmadığı belirtilir. Sadece girdi değerine en yakın vektörün değerleri değiştirilir. LVQ ağının öğrenme hızı ÇKA gibi ağlara göre yüksektir yani bu ağlar bir olayı ÇKA ağlarından daha hızlı öğrenebilmektedirler (Öztemel, 2006).

LVQ ağı girdi, Kohonen ve çıktı katmanlarından oluşmaktadır (Şekil 6.6). Girdi katmanı diğer ağlarda olduğu gibi dış dünyadan alınan örneklerin ağa gösterildiği katmandır ve burada herhangi bir işlem yapılmaz. Gelen bilgiler girdi vektörünü oluşturur. Kohonen katmanı bir ara katmandır. Burada girdi setine en yakın olan ağırlık vektörü belirlenir. Bu katmanda her eleman bir referans vektörünü gösterir. Girdi vektörü, girdi katmanı ile Kohonen katmanı arasındaki ağırlıkların oluşturduğu referans vektörüne haritalanır. Çıktı katmanında ise girdinin ait olduğu sınıf belirlenir.

LVQ ağları girdi katmanı ile Kohonen katmanı arasında tam bağlantılı, Kohonen ile çıktı katmanı arasında ise kısmı bağlantılıdır. Kohonen katmanı ile çıktı katmanı arasındaki ağırlıklar (α) sabit olup 1'e eşittir. Değişen ağırlıklar girdi katmanı ile Kohonen katmanı arasındaki ağırlıklardır.(Şekil 6.6)



Şekil 6.6. LVQ ağının mimari yapısı

Öğrenme bu ağırlıklar üzerinden gerçekleştirilir. Kohonen katmanında kaç tane proses elemanı var ise o kadar referans vektörü oluşacaktır. Referans vektörü girdi değerlerini Kohonen katmanındaki proses elemanlarına bağlayan bağlantıların ağırlık değerlerinden oluşur. Dolayısıyla referans vektörünün eleman sayısı girdi katmanındaki eleman sayısı kadardır.Kohonen ve çıktı katmanlarındaki proses elemanlarının çıktıları ikili değerler olup sadece bir proses elemanın çıktısı 1 diğerlerininki ise 0 değerini alır. Kohonen katmanındaki proses elemanları birbiri ile yarışırlar. Yarışı kazanan proses elemanının çıktısı 1 değerini alır ve diğerlerinin çıktısı 0 olur. Hangi proses elemanının yarışı kazandığına öğrenme kuralına göre karar verilir. Kohonen katmanında hangi proses elemanının çıktısı 1 olursa onun bağlı olduğu çıktı katmanındaki proses elemanının çıktısı 1 değerini alır diğerlerinin çıktısı da 0 olur. Böylece ağa sunulan bir girdi için çıktı katmanında sadece bir proses elemanının çıktısı 1 olmakta ve girdi vektörü o çıktının gösterdiği sınıfın üyesi kabul edilmektedir. Öğrenme ile girdi için doğru sınıfın belirlenmesi sağlanmaktadır (Öztemel, 2006).

6.3.2. LVQ ağının öğrenme kuralı

LVQ ağının öğrenme kuralı, Kohonen tabakasındaki proses elemanlarının birbiri ile yarışması ilkesine dayanır ve buna Kohonen öğrenme kuralı denilmektedir (Öztemel, 2006). Yarışma girdi vektörü ile ağırlık (referans) vektörü arasındaki Öklid mesafesinin hesaplanmasına dayanmaktadır. Mesafesi en kısa olan proses elemanı yarışmayı kazanmaktadır. i. proses elemanı için mesafe,

$$d_i = \|A_i - X\| = \sqrt{\sum_j (A_{ij} - x_j)^2} \tag{6.32}$$

ile hesaplanır. Burada $x$girdi vektörünü, A referans vektörünü, d aradaki mesafeyi ifade etmektedir. A_{ij}vex_j ağırlık vektörü ve girdi vektörünün j. değerlerini göstermektedir. Bu mesafeler tek tek hesaplandıktan sonra proses elemanının referans vektörü girdi vektörüne en yakın olan yarışmayı kazanmaktadır.Kazanan proses elemanı doğru sınıfın üyesi ise ağırlıklar girdi vektörüne biraz daha yakınlaştırılır. Bu örnek ağa tekrar gösterildiğinde yine aynı proses elemanının kazanması için yapılır. Bu durumda ağırlıkların değiştirilmesi,

$$A_y = A_e + \lambda(x - A_e) \tag{6.33}$$

ile tanımlanır. Öğrenme katsayısı olanλ zaman içerisinde 0 değerini alacak şekilde monoton olarak azaltılır. Nedeni ise girdi vektörünün referans vektörüne çok yakınlaştığında durması ve aksi yönde tekrar uzaklaşmaması içindir. Kazanan proses elemanı yanlış sınıftan ise ağırlık vektörü girdi vektöründen uzaklaştırılır. Bu, bir daha aynısı geldiğinde eleman kazanmasın diye yapılır. Ağırlık değişimi şu şekilde olur:

$$A_y = A_e - \lambda(x - A_e) \tag{6.34}$$

6.3.3. LVQ2 ağı

LVQ2 ağı Kohonen tarafından standart LVQ modelinin uygulanması sonucu elde edilen çözümün iyileştirilmesi amacı ile geliştirilmiştir (Kohonen, et al., 1987).Özellikle sınır değerlerindeki yanlış sınıflandırmaları önlemek için kullanılır. Standart LVQ ağının aksine bu ağ eğitim sırasında aynı anda iki referans vektörünün ağırlıklarını değiştirmeyi önermektedir (Öztemel, 2006). Bu iki vektöre A_1 ve A_2 denirse, bunların ağırlıklarının değiştirilmesi için iki koşulun sağlanması gerekmektedir. Bunlar:

a)A_1 girdi vektörüne en yakın ağırlık vektörü, A_2 ise ondan sonraki en yakın ağırlık vektörüdür. A_1 yanlış, A_2 ise doğru sınıftadır, b)Girdi vektörü A_1 ve A_2 vektörlerinin arasında merkezi olarak belirlenmiş bir aralık içerisinde kalır.

Bu iki durumun sağlanması halinde A_1 ve A_2 vektörlerinin her ikisinde ağırlıkları değiştirilir. Bu değiştirilen değerler,

$$A_{1y} = A_{1e} - \lambda(X - A_{1e}) \tag{6.35}$$

$$A_{2y} = A_{2e} + \lambda(X - A_{2e}) \tag{6.36}$$

ile hesaplanır. Burada A_{1e} ve A_{2e} ağırlık vektörlerinin değişmeden önceki değerleridir. Girdi vektörünün iki ağırlık vektörü arasında belirlenen bir aralıkta olup olmadığına

$$Min(d_1/d_2\,, d_2/d_1) > s \; ve \; s = (1-w)/(1+w) \tag{6.37}$$

ile karar verilir. 6.37'deki denklemler sağlanıyorsa o zaman girdi vektörü iki vektörün arasında belirlenen aralıkta demektir. Burada d_1, A_1 ağırlık vektörünün girdi vektörüne olan mesafesi, d_2 ise A_2 ağırlık vektörünün girdi vektörüne olan mesafesidir (Öztemel, 2006).

6.3.4. Cezalandırma mekanizmalı LVQ ağı

Standart LVQ modelinde görülen bazı ağırlık vektörlerinin çok sık üst üstte kazanması, diğer ağırlık vektörleri referans olma niteliklerini kaybetmesi ve belirlenen her ağırlık vektörünü girdinin bir yönünü göstermesi gerekirken sadece bir tanesi bütün sorumluluğu üstlenmesi gibi problemlerden kurtulmak için Desiono tarafından önerilen cezalandırma mekanizması ile ağın eğitimi sırasında sürekli kazanan ağırlık vektörü cezalandırılmakta ve sürekli ardarda kazanması önlenmek istenilmiştir (Desiono, 1987). Cezalandırma mekanizması ağırlık vektörün girdi vektöründen olan mesafesine b kadar değerin eklenmesi ile gerçekleşmektedir. Eklenecek b değeri ilgili ağırlık vektörünün yarışmayı kaç defa kazandığına bağlı olarak belirlenmektedir (Öztemel, 2006). i. proses elemanına ait ağırlık vektörünün girdi vektöründen uzaklık mesafesine eklenecek miktar şu şekilde hesaplanır:

$$b_i = C\left(p_i + \frac{1}{N}\right) \tag{6.38}$$

C sabit bir değeri gösterip kullanıcı tarafından belirlenir. N Kohonen katmanındaki proses elemanı sayısını göstermektedir ve p_i'de i. proses elemanının yarışmayı kazanma olasılığıdır. Bu olasılık başta 1/N olarak belirlenip daha sonra,

$$p_i^y = p_i^e + B(y_i - p_i^e) \tag{6.39}$$

şeklinde güncellenir. Burada p^y proses elemanının kazanma olasılığının yeni değerini, p^e ise proses elemanının değişmeden önceki kazanma olasılık değerini göstermektedir. B kullanıcı tarafından atanan, verilerdeki dalgalanmaları önlemek amacı ile geliştirilmiş sabit bir sayıdır (Öztemel, 2006). Cezalandırma mekanizmalı LVQ ağında referans vektörüne eklenecek ilave b değerleri bütün proses elemanları için belirlendikten sonra girdi vektörü ile ağırlık vektörü arasındaki hesaplanan mesafelere bu b değeri eklenmektedir ve

$$d_i^y = d_i^e + b_i \tag{6.40}$$

ile tanımlanır. d^e girdi vektörü ile ağırlık vektörü arasındaki gerçek mesafeyi, d^y ise kazanma olasılığına dayalı yeni mesafe değerini gösterir. Yeni mesafeler hesaplandıktan sonra Kohonen katmanındaki proses elemanları arasındaki yarışma bu yeni değerler dikkate alınarak yapılmaktadır. Böylece fazla kazananproses elemanının mesafesi daha fazla arttırılıp cezalandırılıp sürekli kazanması önlenir ve diğerlerine de zaman içinde kazanma şansı tanınabilir (Öztemel, 2006).

6.3.5. LVQ-X ağı

LVQ-X ağı, öğrenme kuralına göre eğitim sırasında her iterasyonda çoğunlukla iki ağırlık vektörünün ağırlıkları değiştirilen ve bu sayede hem öğrenme hızını arttıran ve öğrenme süresini kısaltan hem de ağın genelleme yeteneğini iyileştiren bir ağdır. Bu ağ modeli Öztemel tarafından geliştirilmiş olup, global kazanan ve yerel kazanan adında her iterasyonda yarışmayı kazanan iki tane proses elemanı belirlemektedir (Öztemel, 2006). Global kazanan, girdi vektörüne en yakın ağırlık vektörüne sahip olan proses elemanı, yerel kazanan ise doğru sınıf içerisinden girdi vektörüne en yakın olan ağırlık vektörüne sahip proses elemanıdır. LVQ-X ağında eğer global kazanan proses elemanı doğru sınıf içerisinde değil ise onun ağırlık vektörü o girdi vektöründen uzaklaştırılır. Aynı zamanda doğru sınıf içindeki en yakın olan proses elemanının (yerel kazanan) ağırlık vektörü de girdi vektörüne yaklaştırılır. Bu doğru proses elemanının daha sonraki iterasyonda kazanmasına şans vermektedir. Eğer global kazanan ve yerel kazanan aynı proses elemanı ise o zaman sadece tek bir ağırlık vektörü değiştirilmekte ve kazanan proses elemanının ağırlık vektörü girdi vektörüne yaklaştırılmaktadır. Ağırlık vektörü değerleri şu şekilde değiştirilmektedir:

a) Global kazanan ve yerel kazanan aynı proses elemanı ise o proses elemanına ait referans vektörü için formül şudur:

$$A_y = A_e + \lambda(X - A_e) \tag{6.41}$$

b) Global kazanan ve yerel kazanan farklı proses elemanları ise yerel kazanan proses elemanının referans vektörü için denklem (6.40) kullanılır. Global kazanan proses elemanının referans vektörü için ise,

$$A_y = A_e - \lambda(X - A_e) \tag{6.42}$$

den yararlanılır.

6.4. ART (Adaptif Rezonans Teori) Modeli

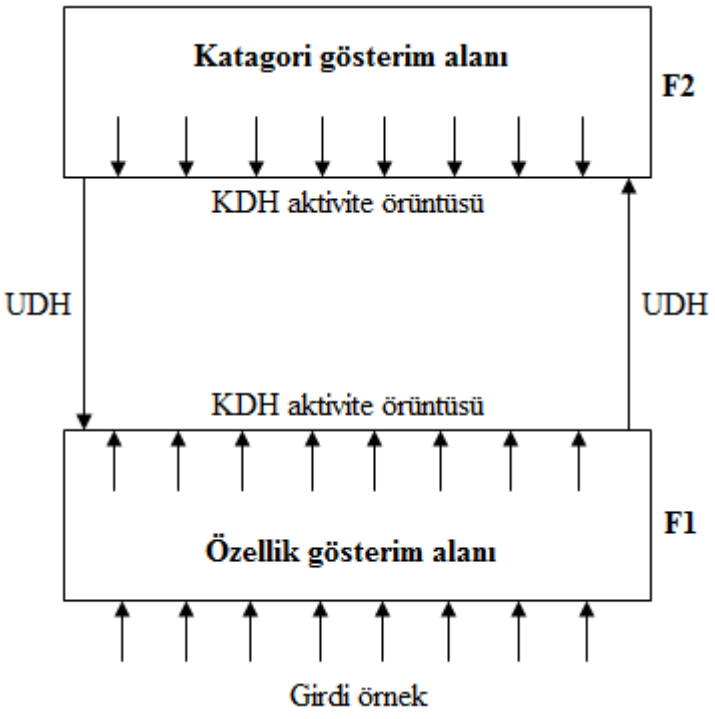
6.4.1. ART ağının yapısı ve özellikleri

ART ağları 1976 yılında Grossberg'in biyolojik beynin fonksiyonlarına yönelik yaptığı çalışmalar neticesinde ortaya çıkmıştır (Grossberg, 1976). ART ağları danışmansız öğrenmeye dayalı çalışan ağlardır ve ART ağları ile danışmansız öğrenme ile çalışan ağların geliştirilmesine temel olmuştur (Öztemel, 2006).ART ağlarının en temel özelliği sınıflandırma problemleri için geliştirilmiş olmasıdır. LVQ ağları da sınıflandırılma için kullanılsa da ART ağlarının bu ağlardan farkı, ağa yapılacak sınıflandırma ile ilgili herhangi bir bilgi vermeden ağın bu işi kendi başına yapmasıdır. Bunu da ağ, doğru bilgileri belirleyerek hafızaya alması ile gerçekleştirir. Bilgiler iki şekilde hafızada saklanır:

a) Kısa dönemli hafıza (KDH): Bilgilerin geçici olarak tutulduğu ve zaman içerisinde yok olup yerine başka bilgilerin saklandığı hafızadır (Öztemel, 2006).

b) Uzun dönemli hafıza (UDH): Bilgilerin sürekli tutulduğu ve kolay kolay unutulmadığı hafızadır ve silinmesi için çok uzun zamanın geçmesi gerekir (Öztemel, 2006).

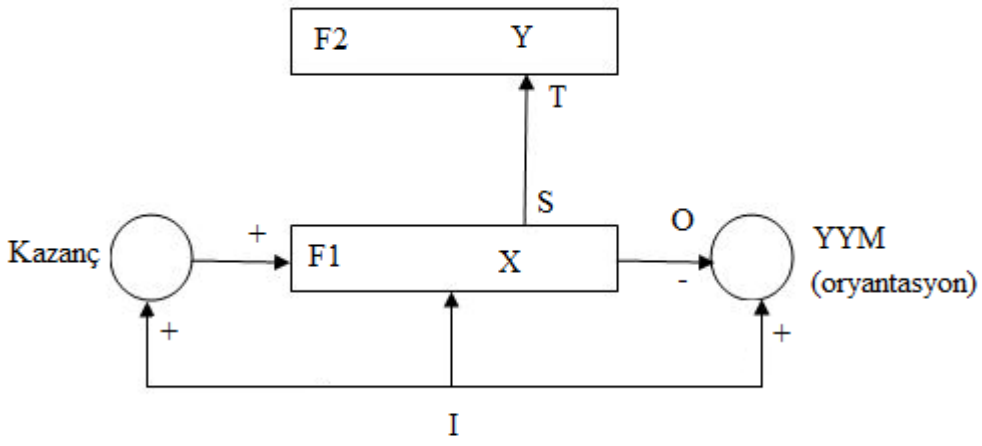
ART ağları genel olarak iki katmandan oluşmaktadır. Bu katmanlar F1 ve F2 olarak isimlendirilir (Şekil 6.7) F1 katmanı girdinin özelliklerini gösterirken F2 katmanı kategorileri (ayrıştırılmış sınıfları) göstermektedir. Bu iki katman birbirine UDH ile bağlanmaktadır. Girdi bilgileri F1 katmanından alınır ve sınıflandırma F2 katmanında yapılır (Öztemel, 2006). ART ağlarında girdiler direkt olarak sınıflandırılmaz. Öncelikle girdilerin özellikleri incelenerek F1 katmanının aktivasyonu belirlenir. UDH'daki bağlantı değerleri ile gelen bilgiler kategorilere ayrılarak F2 katmanına gönderilir. F2 katmanındaki sınıflandırma ile F1 katmanından gelen sınıflandırma birbiri ile eşleştirilerek, eğer örnek belirlenmiş bir sınıf uyuyorsa o kategoride gösterilir. Aksi takdirde ya yeni bir sınıf oluşturulur yada girdinin sınıflandırılması yapılmaz.



Şekil 6.7. ART ağının yapısı

ART ağları aşağıdan yukarı (F1'den F2'ye) bilgi işleme ve yukarıdan aşağı (F2'den F1'e) bilgi işleme şeklinde iki yönlü çalışmaktadır.

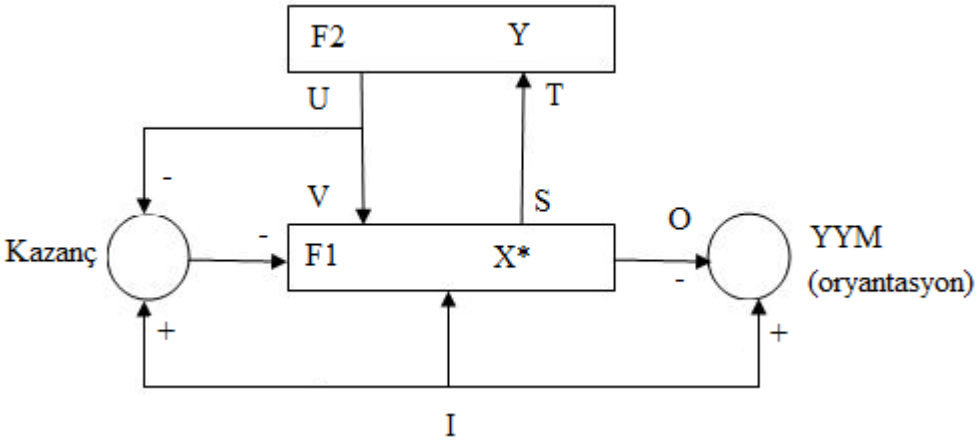
ART ağlarının aşağıdan yukarıya doğru bilgi işlemesi şematik olarak Şekil 6.8'de gösterilmiştir. Aşağıdan yukarıya doğru bilgi işlemede bir girdi örüntüsü (I) ağa gösterilir. Bu görüntü hem F1 katmanında KDH'da X aktivite örüntüsünü oluşturur hem de oryantasyon sistemini diğer bir deyişle yeniden yerleştirme modülünü (YYM) aktif etmek üzere işaret (sinyal) gönderir. Benzer şekilde oluşturulan X örüntüsü hem YYM'ye bir menedici (inhibitory) işaret (O) göndermekte hem de F1 katmanında bir çıktı örüntüsü (S) oluşmaktadır. S sinyali F2 katmanına giden bir girdi örüntüsüne (T) dönüştürülür. Bu girdi örüntüsü ise F2 katmanının çıktısı olan örüntüyü (Y) oluşturur. Bu aynı zamanda ağın da çıktısıdır. Bu şekilde aşağıdan yukarı bilgi işleme tamamlanmış olur.



Şekil 6.8. ART ağında aşağıdan yukarıya bilgi işleme süreci

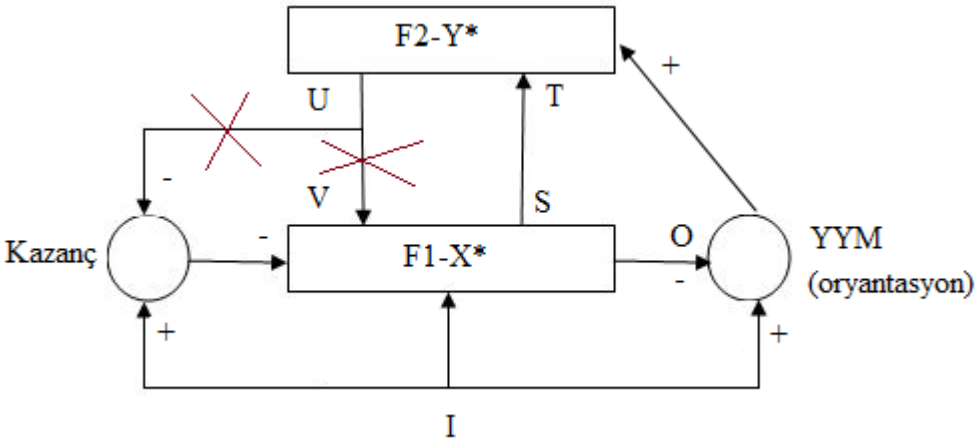
ART ağlarının yukarıdan aşağıya doğru bilgi işlemesi şematik olarak Şekil 6.9'da gösterilmiştir. Yukarıdan aşağıya doğru bilgi işlemede ise F2 katmanında oluşturulan çıktı örüntüsü yukarıdan aşağıya bir sinyal (U) gönderir. Bu sinyal daha sonra beklenen şablon örüntüye (V) dönüştürülür. Aynı zamanda kontrol faktörü

(kazanç) için menedici bir işaret üretilir. Gerçekleştirilen bu işlemlerden sonra şablon örüntünün girdi örüntüsü ile eşlenip eşlenemeyeceği sınanır. Eğer böyle bir eşleşme mümkün değilse o zaman F1 katmanında yeni bir KDH örüntüsü (X*) oluşturulur. Bu örüntü oryantasyon sistemindeki menedici işaretin etkisini azaltır. Oluşturulan X* sinyali oryantasyon sisteminde O işaretinin menedici etkisini azaltarak YYM'nin (oryantasyon modülünün) F2 katmanına bir sinyal göndermesini sağlar.



Şekil 6.9.ART ağında yukarıdan aşağıya bilgi işleme süreci

F' katmanından sinyal gönderildikten sonra bu oluşan gönderilen sinyal Y* örüntüsünü oluşturur ve girilen I örüntüsü için doğru sınıfı gösteren Y* çıktısı üretilir. (Şekil 6.10)

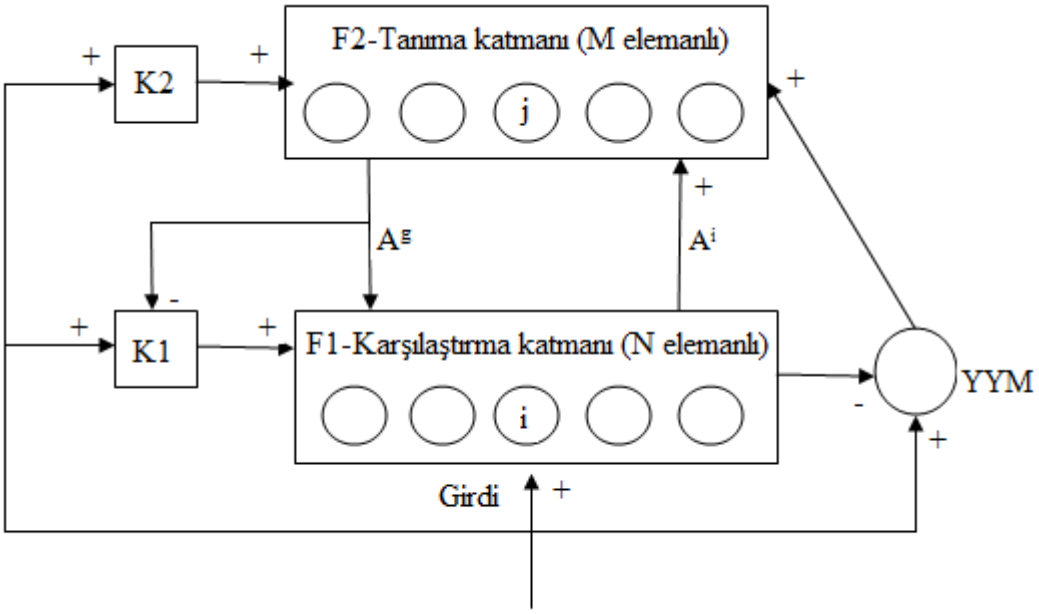


Şekil 6.10 ART ağında yeni sınıfın oluşum şeması

Üretilen V şablon örüntüsü ile girdi örüntüsü eşlendiği zaman sadece yukarıdan aşağı o girdinin sınıfını gösteren ağırlıklar değiştirilir. Bu değiştirme öğrenme kuralına göre yapılır. Her ART ağı modelinin öğrenme kuralı farklıdır (Öztemel, 2006).

6.4.2. ART1 ağı

ART1 ağı sadece ikili (binary) girdiler ile çalışan ve en basit ART ağı modelidir (Şekil 6.11). Karşılaştırma katmanı ismini alan F1 katmanı ve tanıma katmanı ismini alan F2 katmanı olmak üzere iki katmandan oluşur (Öztemel, 2006).



Şekil 6.11. ART1 ağının şematik gösterimi

F1 katmanındaki bütün proses elemanlar F2 katmanındaki proses elemanlarının tamamına bağlanmış şekildedir. Bu bağlantılar sürekli değerlerden oluşan UDH bağlantılarıdır ve A^i olarak isimlendirilmiştir. Bu bağlantılar ileriye doğrudur. Aynı zamanda F2 katmanından F1 katmanına geriye doğru ikili değerleri olan bağlantılar

vardır bunlarda A^g olarak isimlendirilmiştir. Modelde K1 ve K2 olmak üzere iki tane kazanç modülü ve bir adet yeniden yerleştirme modülü (YYM) vardır. Yeniden yerleştirme modülü oryantasyon modülü reset modülü gibi isimlerde almaktadır (Öztemel, 2006; Hristev, 1998).

Karşılaştırma katmanındaki her proses elemanın, girdi örneğinin bir elemanı (aşağıdan yukarı), geri besleme değeri (yukarıdan aşağı), kazanç değeri (K1'den gelen) olmak üzere üç girdisi vardır. F1 katmanındaki proses elemanı bu üç girdiden en az ikisinin aktif olması durumunda çıktı olarak 1 değerini oluşturur ve buna 2/3 kuralı denilir (Öztemel, 2006).F2 katmanındaki proses elemanları kendilerine gelen net girdiyi hesaplar. K2 modülü girdi örneğindeki bütün elemanlar arasında mantıksal 'veya' (OR) operatörüdür. Bu yeni bir sınıfın oluşturulmasında kullanılır. K1 modülü girdi örüntüsünün değerlerini hafızada bulunan şablon değerler ile eşleştirilmede kullanılır. Aşağıdan gelen değerler bu kazanç değerini aktif aşağıdan gelenler ise pasif yapar. YYM ise girdi vektörü ile F1 katmanı çıktısı arasındaki fark belirli bir değeri aşması durumunda F2 katmanına bir sinyal göndermekle yükümlüdür. İki vektörün arasındaki fark kullanıcı tarafından belirlenen benzerlik katsayısı denilen bir değer ile karşılaştırılır. Bu katsayı girdi vektörünün hafızada bulunan çıktı vektörlerine uygunluğunu belirler. Aradaki fark katsayıdan küçükse benzerlik yok anlamına gelir. Bu durumda girdi vektörü için yeni bir sınıf oluşturmak ve hafızaya koymak gerekir. ART1 ağının öğrenmesi bu şekilde gerçekleşir (Öztemel, 2006).

ART1 ağının öğrenme kuralı ise 6 madde halinde açıklanabilir:

a) Ağırlıklara başlangıç değerleri atanır. F1 katmanında ağa sunulan N tane örnek ve F2 katmanında M tane çıktı olduğu durumda M≥N olmalıdır. Çünkü N tane örneğin hepsinin birbirinden farklı olması durumunda ağın her biri için bir sınıf ayıracak durumda olması gerekir. F2 ve F1 arasındaki geriye doğru ağırlıkların bütün değerleri başlangıçta 1 değerini alır ve $A_{ji}^{g} = 1$'dir. F1 ve F2 arasındaki ileriye doğru ağırlıkların başlangıç değerleri, F1 katmanındaki proses elemanı n olmak üzere şu şekilde atanır:

$$A_{ij}^{i} = \frac{1}{1+n} \tag{6.43}$$

b) Benzerlik katsayısının (ρ) değeri belirlenir. Bu 0 ile 1 arasındadır. Bu katsayı ile iki vektörün aynı sınıfın elemanı sayılabilmesi için birbirine ne kadar benzemesi gerektiği belirlenir. Yani bu değerin 0,8 alındığı bir durumda benzerliğin %80 olması durumunda iki vektör aynı sınıfta yer alır.

c) Girdi setindeki örnek X (x1, x2,…,xn) vektörü olarak her bir elemanı x_i olarak tanımlanmış şekilde ağa gösterilir.

d) F2 katmanındaki proses elemanlarının çıktıları hesaplanır:

$$y_j^{'} = \sum_i^N A_{ij}^{i}\ (t)x_i\ \ ,\ \ j: 1,2,\dots,M \tag{6.44}$$

e) Seçilecek kazanan eleman maksimum çıktıya sahip proses elemanıdır. Bu elemanın sahip olduğu ağırlık vektörüne, en uygun sınıf gösterim vektörü denilir. Bu elemanın k. proses elemanı olduğu varsayılan elemanın hesaplanması,

$$y_k^{*} = \max(y_j^{'}) \tag{6.45}$$

ile yapılır.

f) Uygunluk testi yapılıp bu adımda kazanan elemanın ağırlık vektörünün ile girdi vektörünü temsil edip etmediğine karar verilir. Bunun için önce girdi vektöründe bulunan 1 sayısı (s1) belirlenir:

$$s1 = |X| = \sum x_i \tag{6.46}$$

Daha sonra sınıf gösterimi vektörü ile girdi vektörünün uyuştuğu 1 sayısı (s2) bulunur:

$$s2 = \left|A_k^g . X\right| = \sum A_{jk}^g x_i \quad (6.47)$$

$s2/s1 \geq \rho$durumunda iki vektör birbirinin benzeri kabul ediliyorsa sınıf gösterim vektörü girdi vektörünü temsil edebiliyor anlamındadır. Bu durumda ağırlıklar aşağıdaki şekilde değiştirilir:

$$A_{jk}^g (t+1) = A_{jk}^g (t) x_i \quad (6.48)$$

$$A_{ik}^i (t+1) = \left[A_{jk}^g (t) x_i\right] / \left[0{,}5 + \sum_i^N A_{jk}^g (t) x_i\right] \quad (6.49)$$

$s2/s1 < \rho$ durumunda maksimum çıktıyı veren proses elemanının çıktısı 0 olarak atanır ve ondan sonraki maksimum çıktıyı veren proses elemanı seçilerek dördüncü adım olan d adımından itibaren işlemler tekrar edilir. Girdi vektörü bu sınıfın elemanı olarak atanamaz ise üçüncü en büyük çıktıyı veren proses elemanının sınıf gösterim vektörü ile benzerliğine bakılır ve benzerlik olması durumunda ağırlıklar değiştirilir. Başlangıçta birinci örnek birinci sınıfın temsilcisi olarak atanır. İkinci örnek birinci örnek ile benzer ise birinci sınıfın elemanı sayılır. Aksi takdirde bu örnek ikinci sınıfın temsilcisi olur. Böylece örneklerin hepsi ya var olan sınıflardan birine girer ve ağırlıkları değiştirilir ya da yeni bir sınıfın temsilcisi olur. Bu şekilde bu işlemler bütün girdi vektörleri sınıflandırılıncaya kadar devam eder (Öztemel, 2006)

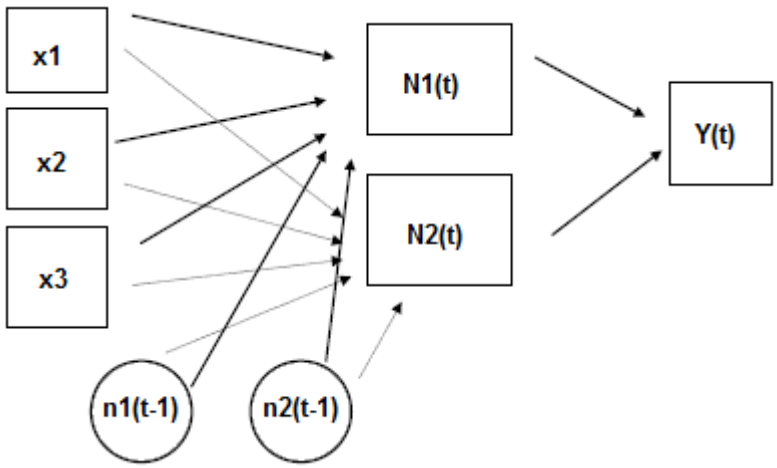
ART1 ağının yanı sıra 1976 yılından bugüne kadar ART2, ART3, ARTMAP, Fuzzy ART gibi birçok ART ağı tanımlanmıştır. Bu ağların hepsi temelde aynı felsefeye dayanmakta ve çok az farklılıklar içermektedir (Öztemel, 2006).

6.5 Elman Ağı

Elman ağı çok katmanlı algılayıcı ağının öğrenme kuralına göre öğrenen, girdi elemanları, ara katman elemanları, çıktı elemanları ve içerik elemanları olmak üzere

dört çeşit proses elemanına sahip yapay sinir ağıdır (Şekil 6.12). Girdi katmanı diğer ağlardaki gibi dış dünyadan bilgiyi alır ve başka işlem yapmaz. Çıktı ünitesinin bilgi işleme fonksiyonları doğrusaldır. Ara katmanlar doğrusal veya doğrusal olmayan transfer fonksiyonlarına sahip olabilirler. İçerik elemanları ara katman elemanlarının önceki aktivite değerlerini hatırlatmak için kullanılmaktadır. Bu elemanlar bir adım gecikmeyi içermektedir. Bir önceki iterasyondaki aktivasyon değerlerini bir sonraki iterasyona girdi olarak taşırlar (Öztemel, 2006).

Elman ağındaki herhangi bir t zamanında hem t zamanındaki girdi hem de ara katmanlardaki t-1 zamanındaki aktivite değerleri ağa girdi olarak verilir. Ağın girdileri belirlendikten sonra ağ artık ileri beslemeli çok katmanlı bir algılayıcıya dönüşür. Bu girdiler kullanılarak ileri doğru ağın çıktıları belirlenir. Bu ileri doğru hesaplamadan sonra ağın ara katmanların aktivasyon değerleri geriye doğru içerik elemanlarına girdi olarak gönderilir ve orada bir sonraki iterasyonda kullanılmak üzere saklanır.



Şekil 6.12. Elman ağının şematik gösterimi (McNelis, 2005)

Elman ağlarında başlangıçta ara katmanların aktivasyon değerleri bilinmediği için içerik elemanlarının başlangıç değerlerinin belirlenmesi gerekir. Bunun için genellikle bir ara katmanın alabileceği maksimum değerin yarısı içerik elemanlarının başlangıç girdi değerleri olarak atanır. Eğer Sigmoid fonksiyonu kullanılacak ise

genellikle 0,5 değeri, Hiperbolik Tanjant fonksiyonu kullanılacaksa 0 değeri başlangıç girdi değerleri olarak atanmaktadır (Öztemel,2006).

Elman ağı öğrenmesini genelleştirilmiş Delta kuralı kullanarak gerçekleştirmektedir. Ara katmanda bulunan elemanlara gelen net girdi değeri girdi katmanındaki elemanın girdi değeri (u) ile ağırlığının (A^g) çarpılıp toplanması sonucu bulunan değerlere içerik elemanından gelen ara katmanlarının (A^i) bir önceki aktivite değerleri ile çarpılıp eklenmesi sonucu bulunur. Herhangi bir t zaman diliminde kullanılan transfer fonksiyonunun Sigmoid fonksiyonu olması durumunda ara katman elemanlarının çıktıları şu şekilde hesaplanır:

$$x_i(t) = 1/\left[1 + e^{-NET_i(t)}\right] \tag{6.50}$$

Net girdi değerini,

$$NET(t) = A^g u(t) + A^i x(t-1) \tag{6.51}$$

formülüyle ifade etmek mümkündür. Bu formül daha açık yazılarak,

$$NET_i = \sum_{j=1}^{N} A_{ji}^g u(t) + \sum_{i=1}^{M} A_{ij}^i x(t-1) \tag{6.52}$$

ifadesine ulaşılır. 6.52'deki N girdi eleman sayısını, M ise ara katman elemanının sayısıdır. Çıktı katmanındaki proses elemanlarının transfer fonksiyonu doğrusal olduğundan herhangi bir t zaman diliminde çıktı katmanındaki çıktı elemanının değeri, A^a ağırlık değerleri ile ara katman elemanlarının çıktıları (x) kullanılarak,

$$y(t) = A^a(t)x(t) \tag{6.53}$$

elde edilirken t zaman dilimindeki çıktı katmanındaki j. elemanın çıktısı ise

$$y_j(t) = \sum_{i=1}^{M} A_i^a(t) X_i(t) \tag{6.54}$$

şeklinde olacaktır. Bu şekilde ağın ileri doğru bilgi işlemesi tamamlanmış olacaktır.Ağa sunulan her örnek için beklenen çıktı (B) belirli olduğundan t. zaman diliminde j. çıktı elemanında oluşan hata,

$$E_j = B_j(t) - y_i(t) \tag{6.55}$$

ile hesaplanır. Hata değeri bulunduktan sonra bu hata değeri ağın değişebilen ağırlıklarına dağıtılır. Bunun içinde ağırlıklara dağıtılacak hata oranları bulunur. Çıktı fonksiyonun Sigmoid olması durumunda bu hata oranı ise

$$\delta(t) = y(t) - [1 - y(t)]E(k) \tag{6.56}$$

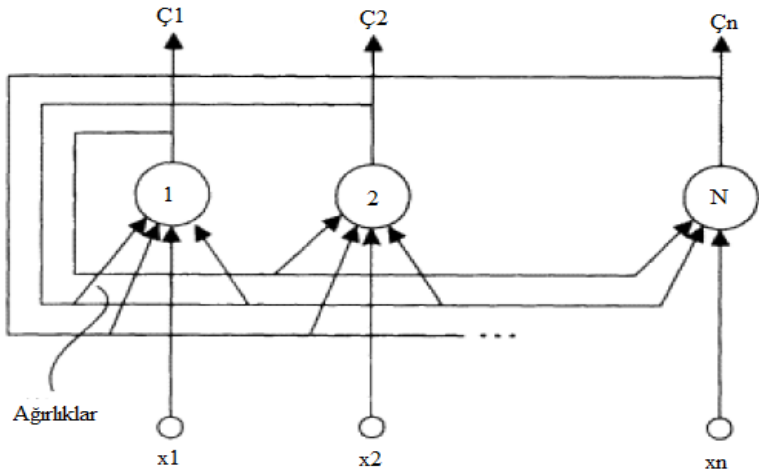
bağıntısıyla elde edilir. Böylelikle Elman ağının öğrenmesi tamamlanır. Burada dikkat edilecek husus geri dönüşüm ağırlıklarının sabit olduğu ve değiştirilmeyeceğidir. Yani ağırlıkların değerleri değiştirilirken geri dönüşüm ağları dikkate alınmaz.

6.6.Hopfield Ağı

Hopfield ağı tek katmanlı ve geri dönüşümlü bir ağdır. Proses elemanlarının tamamı hem girdi hem de çıktı elemanıdır. Ağın bağlantı değerleri bir enerji fonksiyonu olarak saklanmaktadır. Günümüzde kesikli ve sürekli olmak üzere iki Hopfield ağı vardır. Kesikli Hopfield ağları çağrışımlı bellek olarak kullanılırlar. Sürekli Hopfield ağları ise daha çok bileşik optimizasyon problemleri için kullanılmaktadırlar (Öztemel, 2006).

6.6.1. Kesikli Hopfield ağı

Bu ağdaki her hücrenin iki değeri vardır. Hücre on (+1) veya off (-1) olabilir. N proses elemandan oluşan kesikli Hopfield ağının şematik gösterimi Şekil 6.13'te gösterilmiştir.



Şekil 6.13. Kesikli Hopfield ağının şematik gösterimi

Bir proses elemanın t. zamandaki girdisi G(t) olarak ve sabit eşik değeri de θ olarak gösterilirse girdi,

$$G_k(t) = \sum_{j \neq k}^{N} A_{jk}\, Ç_j(t-1) - \theta_k \qquad (6.57)$$

ile hesaplanır.Aynı proses elemanının çıktısı olan Ç(t) ise,

$$Ç(t) = sgn(G(t)) \qquad (6.58)$$

kullanılarak tayin edilecektir. Hopfield ağının eğitilmesinde ağırlıkları belirleme ve saklama fazı ile bilgilere ulaşma fazı olmak üzere iki faz vardır. Ağırlıkları belirleme

fazı ağın öğrenme aşamasını gösterir. Ağ eğitimini bir defada öğrenme prensibine göre aşağıdaki formülü kullanarak gerçekleştirir:

$$A_{ij} = \begin{cases} \frac{1}{N}\sum_{p=1}^{M} \chi_i^p \chi_i^p \; eğer j \neq kise \\ 0 \qquad\qquad aksihalde \end{cases} \tag{6.59}$$

6.59'da M öğrenilecek diğer bir deyişle saklanılacak örnek sayısını ifade etmektedir. χbir örneğin i. ve j. elemanının değerlerini göstermektedir. Bu ağırlıklar hesaplandıktan sonra sabitlenir. Ağın kullanılabilmesi için durağan hale gelmiş olması gereklidir. Ağa daha önce görmediği yani eğitim setinde olmayan bir örnek gösterilir. Bu örnek eksik bilgiler içerebilir. Ağın görevi bu eksiklikleri belirlemek ve örneğin tamamını hafızadan bulmak olduğundan verilen örnek ağa sunulur ve ağın iterasyonlar yaparak durağan hale gelmesi beklenir. Ağ durağan gelince ürettiği çıktı ağın kendisine gösterilen girdiye ürettiği cevabı olarak görülür. Girdi X (X1, X2,…,XN) başlangıç değerlerine atanmak üzere ağın iterasyonları 6.58'e göre devam eder. Ağın durağan hale gelmesi bir enerji fonksiyonunun değerinin minimuma inmesi demektir. Bu enerji fonksiyonu,

$$E(t) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N} A_{ij}\, Ç_i(t) Ç_j(t) + \sum_{i=1}^{N} Ç_i(t)\theta_i \tag{6.60}$$

şeklindedir. Ağ çalışırken bu enerji fonksiyonu ya azalır yada değişmeyeceğinden zaman içinde ağın minimum hata düzeyine yani durağan hale gelmesi mümkün olur.

6.6.2. Sürekli Hopfield ağı

Bu ağların kesikli Hopfield ağlarında farkı bu ağlarda Signum fonksiyonu yerine Sigmoid fonksiyonu kullanılmasıdır. Bu yüzdende ağın çıktı değerleri 0 ila 1 arasında sürekli değerler olabilmektedir (Öztemel, 2006).

6.7.Diğer YSA Modelleri ve Birleşik YSA Modelleri

6.7.1. Counterpropogation ağı

Bu ağlar Robert Hect-Nielsen tarafından geliştirilen hızlı öğrenen ve özellikle uzun eğitim zamanlarına toleransı olmayan problemlerde etkin bir biçimde kullanılan ağlardır (Hect-Nielsen, 1988).Counterpropogation ağları Kohonen ve Grossberg algoritmalarının birleştirilmesi ile oluşturulur. Ağ eğitildikten sonra eksik ve kısmen yanlış değerler içeren girdiler için doğru çıktı vektörleri oluşturulabilir. Bu özelliği ile şekil tanıma problemlerinde etkin olarak kullanılan bir ağdır (Öztemel, 2006).

Counterpropogation ağı girdi katmanı, Kohonen katmanı ve Grossberg katmanı olmak üzere üç katmandan oluşur. Girdi katmanı diğer ağlarda olduğu gibi dış dünyadan gelen bilgileri ağa transfer eder ve burada herhangi bir işlem yapılmaz. Kohonen katmanında girdi vektörünün sınıflandırılması yapılır. Grossberg katmanı da bir çıktı katmanı olarak çalışır.

6.7.2. Cognitron ağı

Cognitron ağı Fukushima tarafından insan beyninin görsel sisteminin görevini üstlenmek amacı ile geliştirilen ve iki katmandan oluşan bir ağdır (Fukushima, 1975).Cognitron ağının birinci katmanında bağlantı alanları mevcuttur. Birbirine yakın ve komşu olan proses elemanları bir bağlantı alanının içinde olduğu varsayılır. İkinci katman ise yarışma alanlarına bölünmüştür. Yarışma alanının her birinde bulunan proses elemanları bağlantı elemanlarından gelen bilgilere göre yarışır. Her elemana bağlantı alanlarında bulunan proses elemanları bağlanmıştır (Öztemel, 2006). Bu ağda uyarıelemanları ve menedici elemanlar olmak üzere iki tür proses elemanı vardır. Bir elemanın çıktısı kendisine gelen uyarıcı ve menedici sinyallere göre belirlenir.

6.7.3. Neocognitron ağı

Fukushima ve arkadaşları tarafından Cognitron modeli daha güçlü ve etkin hale getirilerek bu model oluşturulmuştur (Fukushima, et al., 1982). Neocognitron ağı ile resimlerin rotasyonları, bozulmalar, aynı şeklin değişik pozisyonlarda gösterilmesi gibi durumlarda daha iyi tanıma yapılabilir (Öztemel, 2006).

6.7.4. SOM ağı

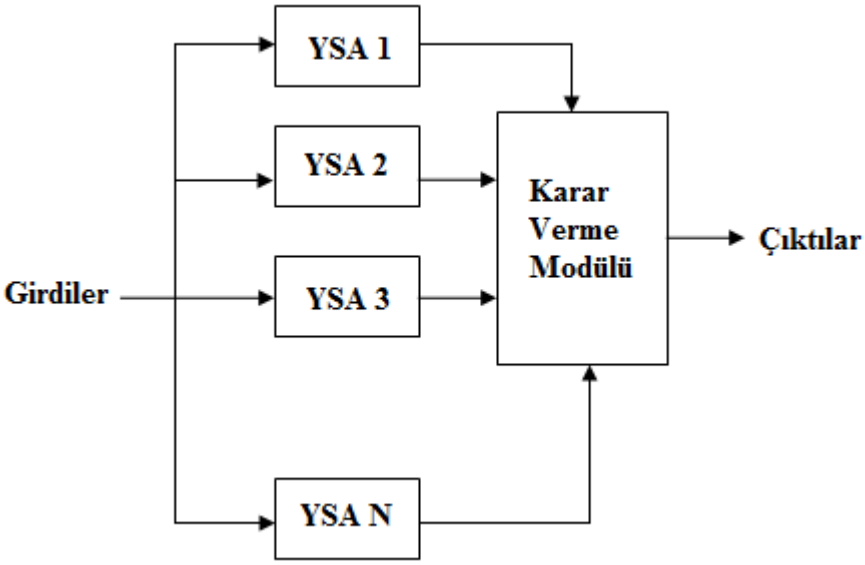
SOM (Self-Organization and Associative Memory) ağları Kohonen tarafından geliştirilmiştir. Genel olarak sınıflandırma için kullanılırlar. Bu ağların girdi vektörlerini sınıflandırmak ve girdi vektörlerinin dağılımını öğrenebilme yetenekleri çok yüksektir.Bu ağların en temel özelliği olayları öğrenmek için danışmana ihtiyaç duymamasıdır. Özellikle beklenen çıktıların belirlenemediği problemler için kullanımı uygundur. Ağ girdi ve çıktı katmanından oluşur. Çıktı katmanı iki boyutlu bir düzlemi gösterir ve proses elemanları bu düzlem üzerine dağılmış vektörleri gösterirler.

SOM ağları yarışmayı kazanma ve kazanan elemanın 1 diğerlerinin 0 değerini alması ilkesine dayanır. Bir girdi verildiğinde, çıktı uzayında yarışmayı kazanan ve onun etrafındaki komşuları, eğitim sırasında ağırlıklarını değiştirmektedir (Öztemel, 2006)

6.7.5. Birleşik YSA modelleri

Karşılaşılan problemlere daha iyi sonuçlar üretebilmek için birden fazla ağın eğitilerek birlikte kullanılması ve bunların bir sinerjisini oluşturarak çözüm üretmek için birden fazla ağın kullanıldığı sistemler geliştirilmiştir. Birden fazla ağın aynı probleme çözüm için geliştirilen bu sistemlere birleşik ağlar denilir (Öztemel, 2006).

Birleşik ağlarda (Şekil 6.14) birden fazla ağın her biri olayın farklı bir yönünü öğrenmekte ve hepsinin kararları bir araya getirilerek ortak bir karar oluşturulabilir.



Şekil 6.14. Birleşik sinir ağlarının şematik gösterimi

Birleşik ağ oluşturmak için an az iki ağın bir araya gelmesi gereklidir. Verilen kararların daha rahat ortak bir karara dönüştürülmesi için üç adet YSA ağının kullanılması önerilmekle beraber böyle bir zorunluluk yoktur (Öztemel, 2006). Bazı durumlarda ağa sunulan örnek bir ağ tarafından tanınmaz iken diğer ağ tarafından tanınmaktadır. Üçün bir ağın kararı bu durumda önemli olmaktadır. Bu üçüncü ağın kararı hangi ağa yakın ise o zaman birleşik ağın kararı o yönde olacaktır. Birleşik ağların eğitilmesi ve test edilmesinde herhangi bir özel algoritmaya gerek olmadığından her ağ birbirinden bağımsız olarak kendi öğrenme kuralına göre eğitilir ve test edilir. Birleşik ağlar eğitimini tamamlamış ve test edilmiş ağlardan oluşur.

Birleşik ağlarda nihai çözüm karar verme modülünde hesaplanır. Problemin girdileri bağımsız olarak her ağa gösterilir ve her ağdan çıktı alınır. Oluşan bu çıktılar bir araya getirilerek oy birliği anlayışına dayalı bir sistemle tek bir sonuca indirgenir.

BÖLÜM 7

SANTRİFÜJ POMPALAR

7.1. Santrifüj Pompalar

Pompalar, mekanik enerjiyi hidrolik enerjiye dönüştüren makinelerdir. Pompalar pozitif yer değiştirmeli pompalar ve rotadinamik pompalar olmak üzere iki ana gurupta toplanır. Volümetrik pompalarda, pompa içerisindeki akışkan hacmi değişmekte ve çalışma mekanik-statik kurallara bağlı kalmaktadır. Rotodinamik pompalarda ise akışkanın, içinden geçtiği bir çark bulunmaktadır. Bu türdeki pompalarda, akışkana çeşitli elemanlar (palet veya özel tasarımlı elemanlar, vb.) yardımıyla moment aktarılmaktadır. Kapalı hacim söz konusu değildir. Akışkan, açık kanallardan geçerken sahip olduğu momentumu arttırılarak, difüzör yardımı ile ulaşılan yüksek hız gerektiğinde basınca dönüştürülür (Pancar ve Ergür, 2007).

Santrifüj pompalar, rotodinamik pompalar gurubundadırlar. Bir santrifüj pompada akışkan çarkın emiş tarafında meydana gelen vakum nedeniyle çarkın kanatları arasına girer. Çark kanatları arasından geçen akışkan çarkın dönüş hareketi ile moment kazanır. Çark kanatları ile çarkın ön ve arka profili tarafından sınırlanan kanalları arasındaki akışkan çarkın çıkış tarafına doğru dönme hareketi esnasında meydana gelen santrifüj kuvvetler etkisiyle itilir. Bu oluşan hareket devamlı akışı ve pompanın emme tarafındaki emişini sağlar. Çark kanatlarını büyük bir momentle terk eden akışkanın içerdiği kinetik enerji, sabit difüzor kanatları arasında ve salyangoz boşluğunda basınç enerjisine çevrilir (Palgrave, 2003).

7.2.Pompalarda Temel Kavramlar

Pompalarda enerjinin nasıl aktarıldığı, akışkanın basılan miktarının ne kadar olduğu gibi genel ifadeleri hesaplamak için pompalardaki temel kavramları incelemek gereklidir.

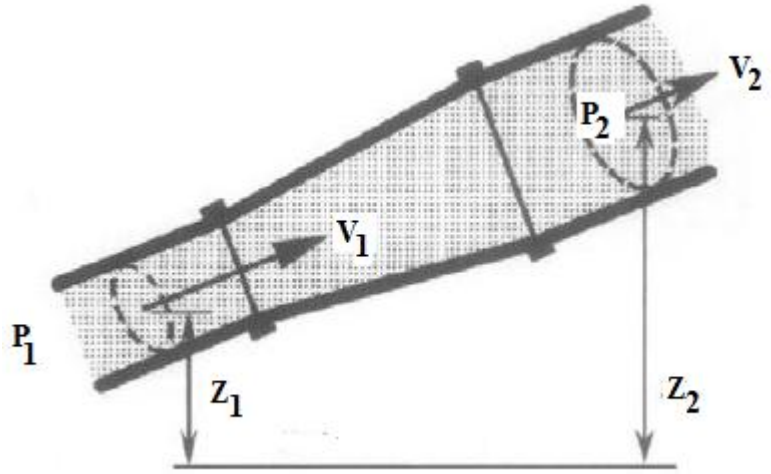
7.2.1. Debi

Pompa debisi, birim zamanda pompanın basma flanşından basılan akışkanın birim zamandaki hacmidir. Pompanın iç kaçakları, eksenel itme dengeleme sistemlerine ve salmastraya giden akışkan miktarı dikkate alınmaz (Şen, 2003). Hacimsel debi Q (m^3/s) şeklinde ifade edilir. Hacimsel debiye bağlı olarak da kütlesel debi şu şekilde ifade edilir. $\dot{m} = \rho . Q$ (kg/s), $\rho = akışkanyoğunluğu$ (kg/m^3)

7.2.2. Manometrik basma yüksekliği

Pompalarda manometrik yükseklik (H) basılan sıvının pompa giriş ve çıkış kesitleri arasındaki birim ağırlık başına kazandığı net enerji olarak tanımlanır. (Şen, 2003). Pompa çarkı vasıtasıyla akışkanın birim ağırlığının yaptığı iştir.

Manometrik yükseklik, pompa çıkışı ve girişinde ölçülen, Bernoulli denkleminde yer alan enerji bileşenlerinin toplamları arasındaki fark olup, emme ve basma borularına yerleştirilen vakummetre ve manometre ile ölçülür.



Şekil 7.1. Bernoulli denklemindeki ifadelerin şematik gösterimi (Nesbitt, 2006)

Sürekli akış halinde pompa girişi ve çıkışı arasında düşü artışı, H, sağlanmaktadır. Tüm kayıplar ihmal edildiğinde Bernoulli denklemi ile

$$H = \left(\frac{P_2}{\gamma} + \frac{V_2^2}{2g} + Z_2\right) - \left(\frac{P_1}{\gamma} + \frac{V_1^2}{2g} + Z_1\right) \tag{7.1}$$

ulaşılan bağıntı 7.1'deki γ ifadesi yoğunluk (kg/m^3) ile yerçekimi ivmesinin (m/s^2) çarpımıdır. P (Pa) basınç V (m/s) ise hızı ifade etmektedir. Z (m) ise referans düzleme olan mesafeyi göstermektedir. Pratikte emme ve basma borusu çapları birbirine eşit veya emme borusu çapı basma borusu çapından daha büyüktür. Emme ve basma borularının çapı eşit olduğunda emme ve basma hızı da birbirine eşit olur (Pancar ve Ergür, 2007). Yerleştirilecek ölçerlerin aynı düzlem üzerinde oldukları düşünülecek olursa 7.1'deki bağıntı aşağıdaki hale gelir:

$$H = (P_2 - P_1)/\gamma = \Delta P/\gamma \tag{7.2}$$

7.2.3. Pompa gücü

Pompa gücü pompa mil gücü ve pompa tahrik motoru gücü olmak üzere iki gurupta incelenebilir. Pompa mil gücü pompayı tahrik etmek için pompa miline uygulanması gereken güçtür ve η pompa verimi olmak üzere şu şekilde hesaplanır:

$$P = (\gamma \times Q \times H)/(102 \times \eta) \quad [kW] \tag{7.3}$$

Pompa tahrik motoru gücü ise, pompanın etiket değerlerinden daha büyük debilerle çalışabileceği düşünülerek pompa mil gücünden α katsayısı kadar büyük seçilir. α katsayıları Çizelge 7.1'de verilmiştir.

$$P_m = \alpha \times P \tag{7.4}$$

Çizelge 7.1. P- α değerleri

P (kW)	α
< 1,5	1,50-1,40
1,5-4	1,40-1,25
4-35	1,25-1,15
>35	1,15-1,10

7.2.4. Özgül hız

Özgül hız pompa çarkının geometrik biçimi belirleyen bir değerdir ve pompanın optimum noktadaki performansı için hesaplanır (Şen, 2003). $n$pompa devir sayısı (d/dk), Q_{opt} pompanın optimum debisi (m^3/s) ve H_{opt} optimum manometrik yükseklik (m) olmak üzere özgül hız aşağıdaki formülle ifade edilir:

$$n_q = n\sqrt{Q_{opt}}/H_{opt}^{\frac{3}{4}} \tag{7.5}$$

7.2.5. Pompa verimi

Pompalarda hidrolik verim, volumetrik verim ve mekanik verim olmak üzere üç çeşit verimle karşılaşılır. Bu verimlerin birbiri ile çarpımından sistemin genel verimi bulunur. Volumetrik verim Δq kaçak debi olmak üzere

$$\eta_v = (Q)/(Q + \Delta q) \tag{7.6}$$

ile hesaplanacaktır. Hidrolik verim (η_h) kayıp düşünün ulaşılan düşüye bölünmesi ile elde edilir. Mekanik verim ise P_f mekanik sürtünmenin neden olduğu güç kaybı olmak üzere

$$\eta_m = 1/(P_f + P_m) \tag{7.7}$$

olacaktır. Sistemin genel verimi ise bu üç verimin çarpımıyla bulunur:

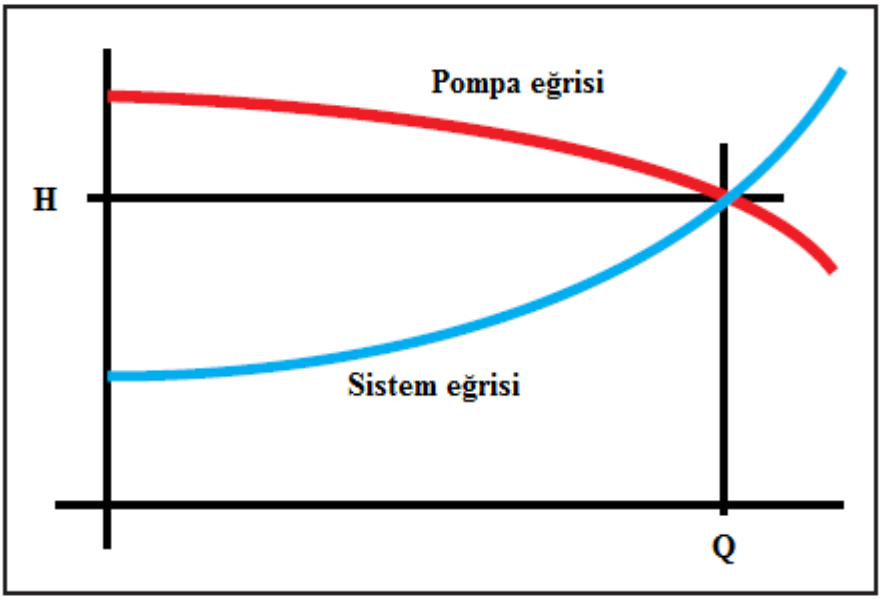
$$\eta_g = \eta_v \times \eta_h \times \eta_m \tag{7.8}$$

7.2.6. Net pozitif emme yüksekliği

Net pozitif emme yüksekliği, pompaların emme koşullarının incelenmesi için kullanılan bir ifadedir. Pompanın emme hattında suyu pompa çarkına taşıyan toplam yararlanılabilir enerji olarak ifade edilen net pozitif emme yüksekliği mutlak basınç olarak belirtilen genel emme yüksekliği ile buhar basıncının farkıdır. $NPSH_A$ pompanın ulaştığı, $NPSH_R$ pompaya gerekli olan net pozitif emme yüksekliğidir (Flach, 1999).

7.2.7. Pompa karakteristik eğrileri

Pompa karakteristik eğrileri, bir pompanın sabit bir devir sayısında su basması halinde manometrik yükseklik, pompa mil gücü, pompa verimi, NPSH gibi değerlerin debiye bağlı olarak değişimini gösteren eğrilerdir. Karakteristik eğriler çizilirken sistem eğrisi ile pompa eğrisinin birbirini kestiği nokta çalışma noktasını verir. Şekil 7.2'de çalışma noktasına denk gelen debi ve basma yüksekliği gösterilmiştir. Pompa çalışma noktasında değişiklik yapabilmek için bu eğrilerde değişiklik yapma yoluna gidilmelidir. Bunun için devir sayısı, çark çapı ve akışkan viskozitesi gibi değerler değiştirilebilir (Flach, 1999).



Şekil 7.2. Pompa eğrisi, sistem eğrisi ve çalışma noktası (Flach, 1999)

BÖLÜM 8

UYGULAMA

T.C. Şeker Fabrikaları A.Ş. Eskişehir Makine Fabrikası'ndan alınan iki adet B tipi (B 50-200 ve B 65-200) pompaya ait dört ayrı giriş çapında yapılan deneylerle elde edilen ve Ekler (Ek.1) bölümünde yer alan H=f(Q) ve N=f(Q) eğrilerinden gerekli veriler alınmıştır. Bu veriler her pompa için H (m) basma yüksekliği, Q (m^3/h) ve N (kW) güçtür. Alınan bu verilerle MATLAB aracılığı ile oluşturulan yapay sinir ağı eğitilmiş olup daha sonra ağın önceden görmediği ara değerler ile ağ test edilmiştir. Yapay sinir ağından elde edilen sonuçlarla bu iki pompaya ait verimler (η) bulunmuş verim eğrileri çizilip gerçek verimlerle karşılaştırılmıştır ve bu şekilde ağın hiç görmediği değerler ile pompa performans tayini yapılabilmiştir.

8.1. Giriş Verilerinin Elde Edilmesi

Ekler bölümünde (Ek.1) verilen B 50-200 ve B 65-200 tipi pompalar için olan H=f(Q) ve N=f(Q) pompa karakteristik eğrilerinden her pompa için n=3000 d/dk ve n=1500 d/dk'lık dönüş hızlarında ve 155 mm, 170 mm, 185 mm, 200 mm'lik giriş çapı için alınan debi (Q), basma yüksekliği (H) ve güç (N) değerleri okunmuştur. Bu değerler ve YSA'dan elde edilen sonuçlar Ekler bölümünde (Ek.6, Ek.7, Ek.8, Ek.9)'dir.

H, Q, n ve giriş çapı D yapay sinir ağında girdi ve N değeri çıktı olarak kullanılmış ve yapay sinir ağından güç (N) değerlerini bulması beklenmiştir. Ayrıca ağın görmediği ve pompa karakteristik eğrilerinden her pompa için 4 adet alınan ara değerler ile YSA test edilmiştir. Bu ara değerlere de Ekler bölümünde (Ek.6, Ek.7, Ek.8, Ek.9) yer verilmiştir.

8.2.Yapay Sinir Ağının Oluşturulması

Oluşturulan yapay sinir ağı 4 girdi değeri için 1 çıktı değeri oluşturan ve danışmanlı öğrenmeyi kullanan Levenberg-Marquardt algoritması ile öğrenen bir ağ modelidir. Girdi değerlerimiz her pompa için basma yüksekliği (H), devir sayısı (n), debi (Q) ve giriş çapıdır. Ağdan beklenen ise bu girdilere karşılık gelen güç (N) değeridir.

8.2.1. Levenberg-Marguardt algoritması

Levenberg- Marguardt algoritması minimumu araştırma algoritmalarından bir tanesidir. Her bir iterasyonda hata yüzeyine parabolik olarak yaklaşır ve parabolün minimumu o iterasyon için en uygun çözüm olur. Minimumu araştırmak için elimizde bir *E(x)* fonksiyonu olduğunu düşünülürse ve bu fonksiyon $\underline{x}$ parametresine göre minimize edilirse Newton metodunda bu 8.1'deki gibi olacaktır.

$$\Delta \underline{x} = -[\nabla^2 E(\underline{x})]^{-1} \nabla E(\underline{x}) \tag{8.1}$$

$\nabla^2 E(\underline{x})$ ifadesi Hessian matrisidir. $\nabla E(\underline{x})$ ise eğimdir. $E(\underline{x})$ fonksiyonunu ağ hatalarının karelerinin toplamı olduğu düşünülürse bu 8.2'deki gibi gösterilebilir:

$$E(\underline{x}) = \sum_{i=1}^{N} e_i^2(\underline{x}) \tag{8.2}$$

8.1'deki ifadeler açıldığında şu iki denkleme ulaşılır:

$$\nabla E(x) = J^T(\underline{x}) e(\underline{x}) \tag{8.3}$$

$$\nabla^2 E(x) = J^T(\underline{x}) J^T(\underline{x}) + s(\underline{x}) \tag{8.4}$$

Buradaki J(x) Jacobian matristir.

$$J(\underline{x}) = \begin{bmatrix} \frac{\partial e_1(\underline{x})}{\partial x_1} & \frac{\partial e_1(\underline{x})}{\partial x_2} & \cdots & \frac{\partial e_1(\underline{x})}{\partial x_n} \\ \frac{\partial e_2(\underline{x})}{\partial x_1} & \frac{\partial e_2(\underline{x})}{\partial x_2} & \cdots & \frac{\partial e_2(\underline{x})}{\partial x_n} \\ \cdots\cdots & \cdots & \cdots & \cdots\cdots \\ \frac{\partial e_N(\underline{x})}{\partial x_1} & \frac{\partial e_N(\underline{x})}{\partial x_2} & \cdots & \frac{\partial e_N(\underline{x})}{\partial x_n} \end{bmatrix} \tag{8.5}$$

8.4'teki $S(\underline{x})$ fonksiyonunun açılımı 8.5'te gösterilmiştir, bu fonksiyon Gauss-Newton metodunda 0 kabul edilir ve denklem 8.6'ya ulaşılır.

$$S(\underline{x}) = \sum_{i=1}^{N} e_i(\underline{x}) \nabla^2 e_i(\underline{x}) \tag{8.5}$$

$$\Delta \underline{x} = [J^T(\underline{x}) J(\underline{x})]^{-1} J^T(\underline{x}) e(\underline{x}) \tag{8.6}$$

Levenberg-Marquardt modifikasyonuyla Gauss-Newton metodu şu hali alır:

$$\Delta \underline{x} = [J^T(\underline{x}) J(\underline{x}) + \mu I]^{-1} J^T(\underline{x}) e(\underline{x}) \tag{8.7}$$

Buradaki I birim matris ve μMarquardt parametresidir. μparametresi skaler bir sayıdır. Her iterasyon sonrasında $E(\underline{x})$ arttıysa μbir faktörle (β) çarpılır, azaldıysa μ parametresi β'ya bölünür. μ parametresi büyük bir sayı ise yöntem küçük adımlı gradyen azalması, küçük bir sayı ise Gauss-Newton yöntemi haline gelir.

İleri beslemeli yapay sinir ağlarında Hessian matrisinin hesaplanması çok zordur. Bunun nedeni ikinci dereceden türevlerin olmasıdır. Bu yüzden LM

algoritmasında Hessian matrisi hesaplanmayıp, bu matrisin yaklaşık değeri kullanılmaktadır. Hessian matrisinin yaklaşık değeri 8.8'deki gibidir:

$$H = [J^T(\underline{x})J(\underline{x}) + \mu I]^{-1} \tag{8.8}$$

Hessian matrisinin yaklaşık değerini bulmak için ilk önce Jacobian matrisinin hesaplanması gerekir. Bu yüzden ağ bütün eğitim girişleri için eğitilerek 8.9'daki gibi hata vektörü hesaplanır:

$$v^T = [v_1, v_2 ... v_N] = [e_{1,1}, e_{2,1} ... e_{S^M+1}, e_{1,2} ... e_{S^M,Q}] \tag{8.9}$$

YSA da ki bütün ağırlıklar ve eşik (bias) değerlerinden aşağıdaki gibi bir vektör oluşturulur.

$$x^T = [x_1, x_2 ... x_N] = [w^1_{1,1}, w^1_{1,2} ..., w^1_{S^1,R}, b^1_1 ... b^1_{s^1}, w^2_{1,1} ... b^M_{S^M}] \tag{8.10}$$

Çıkış katmanı için 8.11 ve gizli katman için 8.12'deki denklemler kullanılarak yerel gradyenler hesaplanır. Bulunan bu gradyenler çıkış katmanı için ayrı, gizli katman için ayrı olmak şartıyla iki matris oluşturulur.

$$S^{-m}_q = -F^m(n^m_q) \tag{8.11}$$

$$S^{-m}_q = -F^m(n^m_q).(W^{m+1})^T.S^{-m+1}_q \tag{8.12}$$

Jacobian matris elamanları ağırlıklar için 8.13, eşik değerler için 8.14 kullanılarak hesaplanır:

$$[J_{h,1}] = \frac{\partial v_h}{\partial x_i} = \frac{\partial e_{k,q}}{\partial w^m_{i,j}} = \frac{\partial e_{k,q}}{\partial n^m_{i,q}} x \frac{\partial n^m_{i,q}}{\partial w^m_{i,j}} = S^{-m}_{i,h} x \frac{\partial n^m_{i,q}}{\partial w^m_{i,j}} = S^{-m}_{i,h} x a^{m-1}_{Jq} \tag{8.13}$$

$$[J_{h,1}] = \frac{\partial v_h}{\partial x_i} = \frac{\partial e_{k,q}}{\partial b^m} = \frac{\partial e_{k,q}}{\partial n_{i,q}^m} x \frac{\partial n_{i,q}^m}{\partial b_i^m} = S_{i,h}^{-m} x \frac{\partial n_{i,q}^m}{\partial b_i^m} = S_{i,h}^{-m} \quad (8.14)$$

Jacobian matrisinin boyutu hata vektörü uzunluğu (N) ile x vektörünün uzunluğu (n) çarpımında bir matristir. Jacobian matrisi hesaplandıktan sonra 8.8'deki ifadeden Hessian matrisi hesaplanır. Hessian matrisi NxN boyutunda simetrik kare matristir. Hessian matrisi sayesinde bağlantı ağırlıkları 8.15'e göre güncellenir.

$$x(k+1) = x(k) - [J^T(k)J(k) + \mu I]^{-1} J^T(k)e(k) \quad (8.15)$$

Levenberg-Marquardt algoritması ağdaki hangi bağlantı ağırlığı sonucu daha çok etkiliyorsa onu direk olarak değiştirmektedir. Bu yüzden çözüme hızlı ulaşmaktadır. Bununla birlikte kusur olarak çok fazla hafıza gerekmektedir.

8.2.2. Ağda kullanılan transfer fonksiyonları

Oluşturulan yapay sinir ağında giriş ve ara katmanlarda hiberbolik tanjant ve çıkış katmanı için lineer bir transfer fonksiyonu olan purelin fonksiyonu kullanılmıştır. Teorik kısımda bu iki fonksiyondan ayrıntılı biçimde bahsedilmiştir.

8.2.2. Ağın MATLAB programında modellenmesi

Uygulamada kullanılan yapay sinir ağı MATLAB programında oluşturulmuştur. MATLAB temel olarak sayısal hesaplama, grafiksel veri gösterimi ve programlamayı içeren teknik ve bilimsel hesaplamalar için yazılmış yüksek performansa sahip bir yazılımdır(İnan, 2007). MATLAB; kontrol, görüntü işleme, istatistik, bulanık mantık, sinir ağları, sayısal işaretleme vb. gibi birçok alanda güvenli bir şekilde kullanılabilecek araç kutuları (toolbox) içerir. Dolayısıyla MATLAB içerisinde yapay sinir ağları için

tasarlanmış araç kutusu (nn toolbox) bulunmaktadır. Fakat uygulamada, kullanılan algoritma ve yapılan işlemlerin daha açık görülmesi için MATLAB içerisinde m-file denilen dosyalarda çalışılmıştır. Kullanılan öğrenme algoritması daha önceki bölümlerde bahsedilen Levenberg-Marquardt algoritmasıdır. Ekler bölümünde (Ek.2, Ek.3, Ek.4, Ek.5) B 50-200 ve B 65-200 tipi pompalar için 1500 d/dk ve 3000 d/dk ve 155 mm giriş çapı için tasarlanan ağın MATLAB kodları verilmiştir. Diğer giriş çapı için aynı kodlar kullanılmış olup sadece giriş çapı değiştirilmiştir.

Yapay sinir ağı kodlarını içeren yukarıda belirtilen ekler incelenecek olursa ağ için ara katmanda 100 nöron kullanıldığı görülmüştür. Sayının bu kadar fazla olması her pompa için kullanılan girişin dörtlü takımlar halinde 10 ila 13 arasında değişmesinden dolayıdır. Giriş için kullanılan veri sayısı çoğaltılırsa nöron sayısı düşürülebilir. Ayrıca algoritmanın sonuna beklenen değerler ile ağın bulduğu değerlerin kıyaslandığı grafik kodları da eklenmiştir. Test için kullanılan ağın görmediği ara değerler yine pompa karakteristik eğrilerinden alınmıştır. Belirtilen eklerde gösterilen test için seçilen girdi değerleri ağa sunulmuş olup ağdan çıktılar alınmıştır. Bu çıktılar ağa görmediği beklenen çıktı ile karşılaştırılmıştır.

BÖLÜM 9

SONUÇ VE ÖNERİLER

9.1. Sonuçlar

Yapılan uygulama sonucu yapay sinir ağlarında kullanılan ağ modeli, eğitim algoritması, ara katman sayısı, bu katmanlarda bulunan nöron sayısı, ağa sunulan örnek sayısı ağın performansına etkisi olan faktörler olarak saptanmıştır.

H=f(Q) ve N=f(Q) pompa eğrilerinden alınan veriler MATLAB programı aracılığı ile Levenberg-Marquardt (LM) öğrenme algoritmasıyla oluşturulan tek katmanlı (giriş ve çıkış katmanı dahil edilirse üç katmanlı) yapay sinir ağından beklenen çıktılara yakın çıktılar alınmak istenmiştir. Yapılan uygulamada görülmüştür ki LM algoritması ile çok çabuk öğrenme gerçekleşip, istenilen sonuçlara kısa bir sürede ulaşılmaktadır. Yapılan ayarlamalar ile nöron sayısı 100 adet seçilmiştir. Bu nöron sayısı ağa gösterilen örnek sayısı çoğaltılarak düşürülebilir. Ağdan alınan çıktılar (N güç değerleri) ile ağın genel hatası hata toplamları olarak her pompa, devir sayısı ve giriş çapı için ayrı ayrı belirlenmiştir. Ayrıca ağın görmediği ara değerler ağa sunularak ağın test edilmesi sağlanmış ve bu girdilere karşılık ağın verdiği çıktılardan da hata toplamları hesaplanıp bu çalışmaya göre yapay sinir ağının hangi tip pompa, devir sayısı ve giriş çapı için en iyi sonucu verdiği tespit edilmiştir. Bu çalışmaya göre uygulamada en iyi sonuç genel öğrenmede %1,5 hata oranı ile 1500 d/dk ve 200 mm giriş çapındaki B 65-200 tip pompanın olsa da ağın görmediği ara değerlere verilen sonuçlar bazında en performanslı ağın %1,3 hata oranı ile 1500 d/dk ve 200 mm giriş çapı ile B 50-200 tip pompaya ait olduğu görülmüştür. Tüm pompalara ait genel hata değerleri ve ara değerlerle ağın sınanması ile bulunan hata değerleri Ek.10'da ayrıntılı biçimde verilmiştir.

Santrifüj pompalar bölümünde geçen 7.3 nolu bağıntı ile pompa verimleri hem gerçek değerler ile hem de YSA sonuçlarına göre hesaplanmış ve performans (verim) eğrileri oluşturulmuştur. Ayrıca bu eğriler karşılaştırmalı grafikler biçiminde Ek.11'de verilmiştir. Bu grafiklerden de görülebilir ki uygun ağ modeli ve öğrenme algoritması kullanıldığında, ağ hiç görmediği değerlere karşılık beklenen çıktıya çok yakın değerler bulup bu değerlerden gerçeğe uygun pompa performans tahmini yapılabilmektedir.

9.2 Öneriler

Yapay sinir ağları kullanılarak değişik tip pompalarda güç, verim, basma yüksekliği gibi değerler için çeşitli çalışmalar yapılabilir. Ayrıca veri aralığı genişletilip farklı yapay sinir ağı modelleri tasarlanabilir.

KAYNAKLAR DİZİNİ

Desiono, D., 1987, Adding a conscience to competitive learning, Proc Of The International Conference On Neural Networks, San Diego, California, Vol 1, 117-124.

Dreyfus, G., 2005, Neural networks methodology and appplications, Springer-Verlag, Berlin, Heidelberg, 497 p.

Efe, M. ve Kaynak, O., 2000, Yapay sinir ağları ve uygulamaları, Boğaziçi Üniversitesi Yayınları, 148 s.

Elmas, Ç., 2007, Yapay zeka uygulamaları, Seçkin Yayıncılık San ve Tic Aş, 425 s.

Ergezer, H., Dikmen, M. ve Özdemir, E., 2003, Yapay sinir ağları ve tanıma sistemleri, Pivolka, 6, 14-17.

Fırat, M. ve Güngör, M., 2004, Askı madde konsantrasyonu ve miktarının yapay sinir ağları ile belirlenmesi, İMO Teknik Dergi, 3267-3282.

Flach, P., 1999, Pump handbook: pumps and systems: centrifugal pumps, Elsevier Science and Technology Books, 259 p.

Freeman, J.A. and Skapura, D.M., 1991, Neural networks algorithm, applications and programming techniques, Addison-Wesley Publishing Company. 401 p.

Fukushima, K., 1975, Cognitron-a self organizing multi-layered neural network, Biological Cybernetics, 20, 121-136.

Fukushima, K. and Miyake S., 1982, Neocognitron: a new algorithm for pattern recogniton tolerant of defermations and shifts in positions, Pattern Recognition, 15, 455-469.

Grossberg, S., 1976, Adaptive pattern classification and universal recoding 1: parallel devolopment and coding of neural feature detectors, Biological Cybernetics, 23, 121-134.

Gurney, K., 1997, An introduction to neural networks, UCL Press Limited, Londra, 234 p.

Hect-Nielsen, R., 1988, Aplications of counterpropogation networks, Neural Networks, 1, 131-139.

KAYNAKLAR DİZİNİ (devam ediyor)

Hristev, R.M., 1998, The ANN book, GNU General Public Press, 371 p.

İnan, A., 2007, MATLAB kılavuzu grafik programlama, matematik ve mühendislik uygulamaları, Papatya Yayıncılık, 699 s.

Kalach, A.V., 2005, Application of neural networks in multitouch-sensitive systems for the detection of nitrohydrocarbons in the air, Sensors Journal, 5, 97-102.

Kartalapoulos, S.V., 1996, Understanding neural networks and fuzzy logic basic concepts and applications, IEEE Pres, 205 p.

Kasabov, N.K., 1998, Foundation of neural networks, fuzzy systems and knowledge engineering, The MIT Press, England, 550 p.

Kohonen, T., Barna, G. and Chrisley, R., 1987, Statistical pattern recognition with neural networks: benchmarking studies, Proc. Of The International Joint Conference On Neural Networks, San Diago, California, Vol 1, 66-68.

Kulkarni, A.D., 1994, Artificial neural networks for image understanding, Van Nostrand Reinhold, New York, 210 p.

McNelis, P.D., 2005, Neural network in finance: gaining predictive edge in the market, Elsevier Academic Press, London, UK, 233 p.

Mehrotra, K., Mohan, C.K. and Ranka, S., 1997, Elements of artificial neural networks, The MIT Press, London, 344 p.

Nesbitt, B., 2006, Handbook of pumps and pumping, Elsevier Science and Technology Books, 470 p.

Öztemel, E., 2006, Yapay sinir ağları, Papatya Yayıncılık Eğitim Bilgisayar Sis. San. Ve Tic. A.Ş., 232 s.

Palgrave, R., 2003, Troubleshooting centrifugal pumps and their systems, Elsevier Science and Technology Books, 280 s.

Pancar, Y. ve Ergür, H.S., 2007, Hidrolik makinalar ve uygulamaları, Birsen Yayınevi, 267 s.

KAYNAKLAR DİZİNİ (devam ediyor)

Rosenblatt, F., 1958, The perceptron: a probabilistic model for information stroge and organization in the brain, Pyschological Review, Volume 65, No 6, 386-408.

Rumelhart, D.E., Hinton, G.E. and Williams, R.J., 1986, Learning representations by backpropogation errors, Nature, Vol 323, 533-536.

Sağıroğlu, Ş., Beşdok, E. ve Erler, M., 2003, Mühendislikte yapay zeka uygulamaları-I:yapay sinir ağları, Ufuk Kitap Kırtasiye Yayıncılık Tic. Ltd. Şti., 423 s.

Şen, M., 2003, Santrifüj pompalar ve pompa tesisatları, Mas Pompa San. Aş., 249 s.

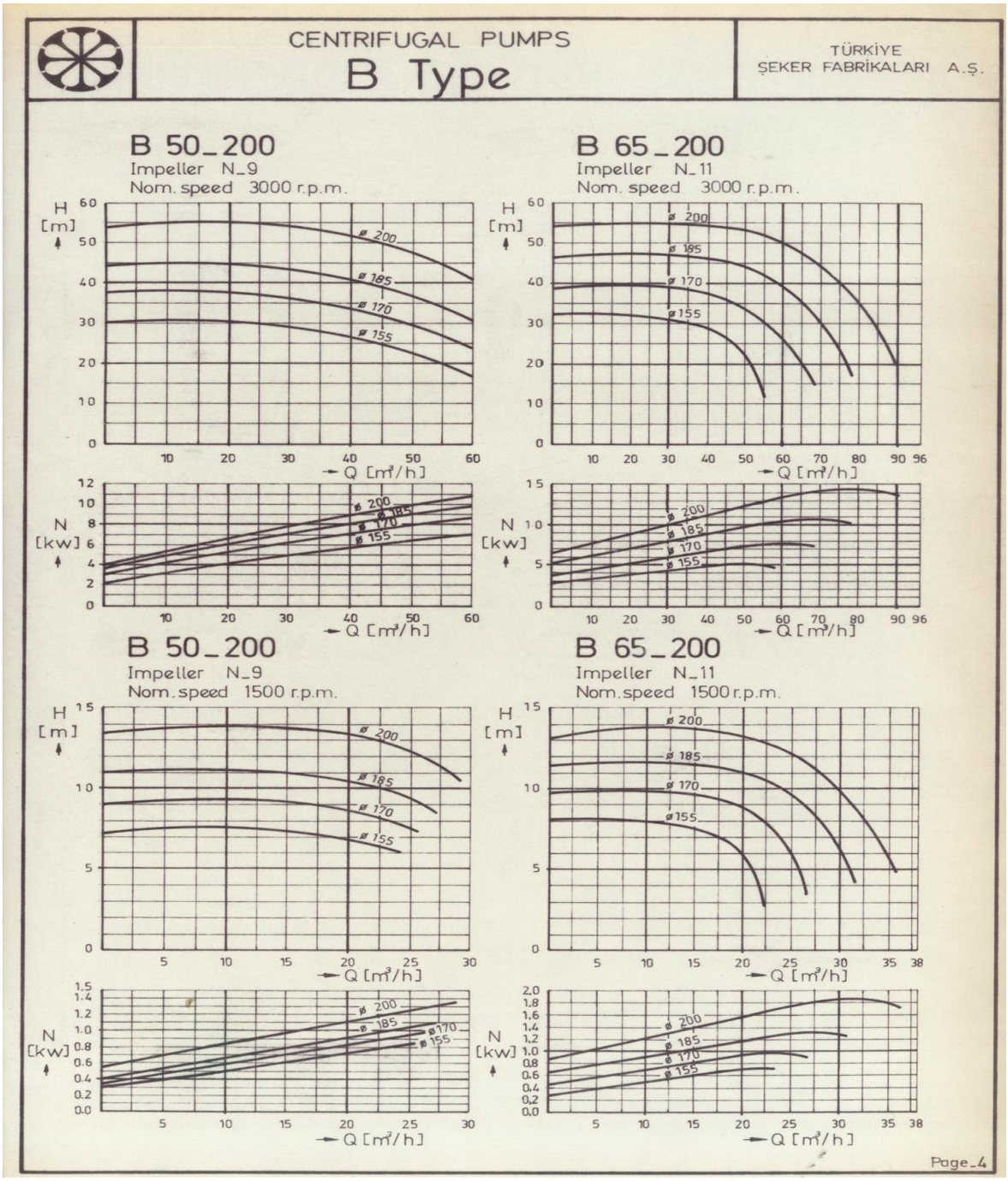
Şen, Z., 2004, Yapay sinir ağları ilkeleri, Su Vakfı yayınları, 183 s.

Widrow, B. and Hoff, M.E., 1960, Adaptive switching circuits, Ire Wescon Convection Records, Part 4, 96-104.

Yurtoğlu, H., 2005, Yapay sinir ağları metodolojisi ile öngörü modellemesi: bazı makroekonomik değişkenler için Türkiye örneği, Uzmanlık tezi, DPT Ekonomik Modeller ve Stratejik Araştırmalar Genel Müdürlüğü, 104 s., (yayımlanmamış).

EKLER

Ek. 1. Santrifüj pompa karakteristik eğrileri(T.Ş.F.A.Ş Eskişehir Şeker Makine Fabrikası)



Ek. 2. B 50-200 pompası için 1500 d/d ve 155 mm giriş çapı için oluşturulan YSA (3 sayfa)

```
x=[0 7.26 155 1500;2.5 7.5 155 1500;5 7.77 155 1500;7.5 7.8 155 1500;
    10 7.8 155 1500;12.5 7.54 155 1500;15 7.34 155 1500;17.5 7.15 155
1500;
    20 6.83 155 1500;22.5 6.5 155 1500];
yd=[0.3 0.36 0.4 0.46 0.51 0.57 0.61 0.68 0.73 0.8 ];




c1=max(max(x));
x=x/max(max(x));
c2=max(yd);
yd=yd./max(yd);
w1=rand(100,4);
w2=rand(1,100);
q1=rand(100,1);
q2=rand(1,1);
s22=[];
s11=[];
J=[];
I=eye(601,601);
m=.8;
E1=0;
maliyet=[];
for cok=1:100

    s22=[];
    s11=[];
for i=1:10
    x1=x(i,:);
    n1=w1*x1'+q1;
    a1=tanh(n1);
    n2=w2*a1+q2;
    a2(i)=purelin(n2);
    e(i)=yd(i)-a2(i);

    a2(i)=a2(i)*c2;
    s2=-1;
    s1=(sech(n1).*sech(n1)).*w2'*s2;
    s22=[s22 s2];
    s11=[s11 s1];
end
    bn=(e*c2)
    cn=bn.*bn
en=[1
        1
        1
        1
        1
```

```
        1
        1
        1
        1
        1]



    E=((cn*en)^(1/2))/10;
maliyet=[maliyet E];
for i=1:10
    s111=s11';
    j1=s111(i,:)'*x(i,:);
    j2=s111(i,:);
    j3=s22(i)*a1;
    j4=s22(i);
f=1;
for k=1:100
for z=1:4

         J(i,f)=j1(k,z);
f=f+1;
end
end

for k=1:100
      J(i,f)=j2(k);
f=f+1;
end
for k=1:100
       J(i,f)=[j3(k)];
f=f+1;
end
      J(i,f)=j4;
end
deltaw1=-(((J'*J)+(m*I))^(-1))*J'*e';
f=1;
for k=1:100
for z=1:4
    w1(k,z)=w1(k,z)+deltaw1(f);
f=f+1;
end
end

for k=1:100;
      q1(k)=q1(k)+deltaw1(f);
f=f+1;
end
for k=1:100;
      w2(k)=w2(k)+deltaw1(f);
f=f+1;
end
    q2=q2+deltaw1(f);
    E;
f=1;
```

```
if(E>E1 &E==E1)

m=m*1.1;
else
m=m/1.1;
end
  E1=E;
end


yd=yd*c2;

figure(1)
plot (yd,'o')
hold on;
plot(a2,'+')
title('Egitim sonuclari')
xlabel('+= Ag Cikisi,o= Gercek deger')




x2=[1.25 7.32 155 1500 ;8.75 7.8 155 1500 ;16.25 7.2 155 1500;21.25
6.68 155 1500 ];
  cc=[0.33 0.49 0.65 0.83 ];
  x2=x2/c1;
for i=1:4
    x3=x2(i,:);
    n11=w1*x3'+q1;
    a11=tanh(n11);
    n22=w2*a11+q2;
    a22(i)=purelin(n22);
    a22(i)=a22(i)*c2;
end
    figure(2)
   plot (a22,'+')
   hold on
   plot (cc,'o')
  title('Test sonuclari')
  xlabel('+= Ag Cikisi,o= Gercek deger')
```

Ek. 3. B 50-200 pompası için 3000 d/d ve 155 mm giriş çapı için oluşturulan YSA (3 sayfa)

```
x=[0 155 3000 30;5 155 3000 30.69;10 155 3000 30.93;
    15 155 3000 30.93;20 155 3000 30.35;25 155 3000 29.66;
    30 155 3000 28.62;35 155 3000 27.93;40 155 3000 26.55;
    45 155 3000 24.45;50 155 3000 22.03;55 155 3000 20;60 155 3000
16.55];
yd=[2 2.67 3.17 3.67 4.1 4.5 5 5.43 5.7 6 6.5 6.83 7];

c1=max(max(x));
x=x/max(max(x));
c2=max(yd);
yd=yd./max(yd);
w1=rand(100,4);
w2=rand(1,100);
q1=rand(100,1);
q2=rand(1,1);
s22=[];
s11=[];
J=[];
I=eye(601,601);
m=.9;
E1=0;
maliyet=[];
for cok=1:100

    s22=[];
    s11=[];
for i=1:13
    x1=x(i,:);
    n1=w1*x1'+q1;
    a1=tanh(n1);
    n2=w2*a1+q2;
    a2(i)=purelin(n2);
    e(i)=yd(i)-a2(i);

    a2(i)=a2(i)*c2;
    s2=-1;
    s1=(sech(n1).*sech(n1)).*w2'*s2;
    s22=[s22 s2];
    s11=[s11 s1];
end
    bn=(e*c2)
    cn=bn.*bn
en=[1
        1
        1
        1
        1
        1
        1
        1
        1
        1
```

```
        1
        1
        1]
    E=((cn*en)^(1/2))/13;
maliyet=[maliyet E];
for i=1:13
    s111=s11';
    j1=s111(i,:)'*x(i,:);
    j2=s111(i,:);
    j3=s22(i)*a1;
    j4=s22(i);
f=1;
for k=1:100
for z=1:4

         J(i,f)=j1(k,z);
f=f+1;
end
end

for k=1:100
      J(i,f)=j2(k);
f=f+1;
end
for k=1:100
       J(i,f)=[j3(k)];
f=f+1;
end
      J(i,f)=j4;
end
deltaw1=-(((J'*J)+(m*I))^(-1))*J'*e';
f=1;
for k=1:100
for z=1:4
    w1(k,z)=w1(k,z)+deltaw1(f);
f=f+1;
end
end

for k=1:100;
      q1(k)=q1(k)+deltaw1(f);
f=f+1;
end
for k=1:100;
      w2(k)=w2(k)+deltaw1(f);
f=f+1;
end
    q2=q2+deltaw1(f);
    E;
f=1;
if(E>E1 &E==E1)

m=m*1.1;
else
m=m/1.1;
end
  E1=E;
```

```
end


yd=yd*c2;

figure(1)
plot (yd,'+')
hold on;
plot(a2,'o')
title('Egitim sonuclari')
xlabel('+= Ag Cikisi,o= Gercek deger')


  x2=[22.5 155 3000 30;32.5 155 3000 28.28;37.5 155 3000 27.24;7.5 155
3000 30.8];
  cc=[4.3 5.2 5.55 2.9 ];
  x2=x2/c1;
for i=1:4
    x3=x2(i,:);
    n11=w1*x3'+q1;
    a11=tanh(n11);
    n22=w2*a11+q2;
    a22(i)=purelin(n22);
    a22(i)=a22(i)*c2;
end
    figure(2)
   plot (a22,'+')
   hold on
   plot (cc,'o')
  title('Test sonuclari')
  xlabel('+= Ag Cikisi,o= Gercek deger')
```

Ek. 4. B 65-200 pompası için 1500 d/d ve 155 mm giriş çapı için oluşturulan YSA (3 sayfa)

```
x=[0 8.01 155 1500;2.5 8.02 155 1500;5 8.02 155 1500;7.5 8.01 155
1500;
    10 8 155 1500;12.5 7.77 200 1500;15 7.51 155 1500;17.5 7 155 1500;
    20 6 155 1500;22.5 2.8 155 1500];
yd=[0.3 0.35 0.4 0.45 0.5 0.55 0.6 0.65 0.7 0.71];


c1=max(max(x));
x=x/max(max(x));
c2=max(yd);
yd=yd./max(yd);
w1=rand(100,4);
w2=rand(1,100);
q1=rand(100,1);
q2=rand(1,1);
s22=[];
s11=[];
J=[];
I=eye(601,601);
m=.8;
E1=0;
maliyet=[];
for cok=1:100

    s22=[];
    s11=[];
for i=1:10
    x1=x(i,:);
    n1=w1*x1'+q1;
    a1=tanh(n1);
    n2=w2*a1+q2;
    a2(i)=purelin(n2);
    e(i)=yd(i)-a2(i);

    a2(i)=a2(i)*c2;
    s2=-1;
    s1=(sech(n1).*sech(n1)).*w2'*s2;
    s22=[s22 s2];
    s11=[s11 s1];
end
    bn=(e*c2)
    cn=bn.*bn
en=[1
        1
        1
        1
        1
        1
        1
        1
        1
        1
```

```
]

    E=((cn*en)^(1/2))/10;
maliyet=[maliyet E];
for i=1:10
    s111=s11';
    j1=s111(i,:)'*x(i,:);
    j2=s111(i,:);
    j3=s22(i)*a1;
    j4=s22(i);
f=1;
for k=1:100
for z=1:4

         J(i,f)=j1(k,z);
f=f+1;
end
end

for k=1:100
      J(i,f)=j2(k);
f=f+1;
end
for k=1:100
       J(i,f)=[j3(k)];
f=f+1;
end
      J(i,f)=j4;
end
deltaw1=-(((J'*J)+(m*I))^(-1))*J'*e';
f=1;
for k=1:100
for z=1:4
    w1(k,z)=w1(k,z)+deltaw1(f);
f=f+1;
end
end

for k=1:100;
      q1(k)=q1(k)+deltaw1(f);
f=f+1;
end
for k=1:100;
      w2(k)=w2(k)+deltaw1(f);
f=f+1;
end
    q2=q2+deltaw1(f);
    E;
f=1;
if(E>E1 &E==E1)

m=m*1.1;
else
m=m/1.1;
end
```

```
  E1=E;
end


yd=yd*c2;

figure(1)
plot (yd,'o')
hold on;
plot(a2,'+')
title('Egitim sonuclari')
xlabel('+= Ag Cikisi,o= Gercek deger')


x2=[8.75 8 155 1500 ;13.75 7.64 155 1500 ;21.25 4.4 155 1500;16.25
7.26 155 1500 ];
  cc=[0.43 0.58 0.7 0.63];
  x2=x2/c1;
for i=1:4
    x3=x2(i,:);
    n11=w1*x3'+q1;
    a11=tanh(n11);
    n22=w2*a11+q2;
    a22(i)=purelin(n22);
    a22(i)=a22(i)*c2;
end
    figure(2)
   plot (a22,'+')
   hold on
   plot (cc,'o')
  title('Test sonuclari')
  xlabel('+= Ag Cikisi,o= Gercek deger')
```

Ek. 5. B 65-200 pompası için 3000 d/d ve 155 mm giriş çapı için oluşturulan YSA (3 sayfa)

```
x=[0 155 3000 32.5;5 155 3000 33;10 155 3000 33;
    15 155 3000 32.5;20 155 3000 31.75;25 155 3000 31.4;
    30 155 3000 30.7;35 155 3000 30;40 155 3000 28.84;
    45 155 3000 25.7;50 155 3000 21.4;55 155 3000 12.5];
yd=[2.9 3 3.33 3.5 3.8 4 4.2 4.5 4.8 5 5.1 4.97];


c1=max(max(x));
x=x/max(max(x));
c2=max(yd);
yd=yd./max(yd);
w1=rand(100,4);
w2=rand(1,100);
q1=rand(100,1);
q2=rand(1,1);
s22=[];
s11=[];
J=[];
I=eye(601,601);
m=.9;
E1=0;
maliyet=[];
for cok=1:100

    s22=[];
    s11=[];
for i=1:12
    x1=x(i,:);
    n1=w1*x1'+q1;
    a1=tanh(n1);
    n2=w2*a1+q2;
    a2(i)=purelin(n2);
    e(i)=yd(i)-a2(i);

    a2(i)=a2(i)*c2;
    s2=-1;
    s1=(sech(n1).*sech(n1)).*w2'*s2;
    s22=[s22 s2];
    s11=[s11 s1];
end
    bn=(e*c2)
    cn=bn.*bn
en=[1
        1
        1
        1
        1
        1
        1
        1
        1
        1
```

```
        1
        1]

    E=((cn*en)^(1/2))/12;
maliyet=[maliyet E];
for i=1:12
    s111=s11';
    j1=s111(i,:)'*x(i,:);
    j2=s111(i,:);
    j3=s22(i)*a1;
    j4=s22(i);
f=1;
for k=1:100
for z=1:4

         J(i,f)=j1(k,z);
f=f+1;
end
end

for k=1:100
      J(i,f)=j2(k);
f=f+1;
end
for k=1:100
       J(i,f)=[j3(k)];
f=f+1;
end
      J(i,f)=j4;
end
deltaw1=-(((J'*J)+(m*I))^(-1))*J'*e';
f=1;
for k=1:100
for z=1:4
    w1(k,z)=w1(k,z)+deltaw1(f);
f=f+1;
end
end

for k=1:100;
      q1(k)=q1(k)+deltaw1(f);
f=f+1;
end
for k=1:100;
      w2(k)=w2(k)+deltaw1(f);
f=f+1;
end
    q2=q2+deltaw1(f);
    E;
f=1;
if(E>E1 &E==E1)

m=m*1.1;
else
m=m/1.1;
```

```
end
  E1=E;
end


yd=yd*c2;

figure(1)
plot (yd,'o')
hold on;
plot(a2,'+')
title('Egitim sonuclari')
xlabel('+= Ag Cikisi,o= Gercek deger')

x2=[7.5 155 3000 33;22.5 155 3000 31.58;42.5 155 3000 27.27;47.5 155
3000 23.55];
  cc=[2.95 3.9 4.9 5.05];
  x2=x2/c1;
for i=1:4
    x3=x2(i,:);
    n11=w1*x3'+q1;
    a11=tanh(n11);
    n22=w2*a11+q2;
    a22(i)=purelin(n22);
    a22(i)=a22(i)*c2;
end
  figure(2)
   plot (a22,'+')
   hold on
   plot (cc,'o')
  title('Test sonuclari')
  xlabel('+= Ag Cikisi,o= Gercek deger')
```

Ek. 6. B 50-200 pompasının 1500 d/d için karakteristik eğrilerden alınan datalar ve YSA'nın ürettiği sonuçlar (2 sayfa)

Q[m³/h]	H [m]	n[d/d]	Ø[mm]	N [kW]	Nysa[kW]	Verim(ŋ)	VerimYSA(ŋ)
0	7,26	1500	155	0,300	0,300	0	0
2,5	7,50	1500	155	0,360	0,353	0,142	0,145
5,	7,77	1500	155	0,400	0,406	0,265	0,261
7,5	7,80	1500	155	0,460	0,460	0,346	0,347
10	7,80	1500	155	0,510	0,514	0,417	0,414
12,5	7,54	1500	155	0,570	0,569	0,450	0,451
15	7,34	1500	155	0,610	0,624	0,492	0,481
17,5	7,15	1500	155	0,680	0,679	0,501	0,502
20	6,83	1500	155	0,730	0,734	0,510	0,507
22,5	6,50	1500	155	0,800	0,790	0,498	0,504
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø[mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
1,25	7,32	1500	155	0,330	0,327	0,076	0,076
8,75	7,80	1500	155	0,490	0,486	0,379	0,382
16,25	7,20	1500	155	0,650	0,651	0,490	0,489
21,25	6,68	1500	155	0,830	0,762	0,466	0,507
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø[mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
0	9,00	1500	170	0,340	0,346	0	0
2,5	9,17	1500	170	0,400	0,408	0,156	0,153
5	9,26	1500	170	0,480	0,470	0,263	0,268
7,5	9,34	1500	170	0,540	0,531	0,353	0,359
10	9,34	1500	170	0,600	0,592	0,424	0,429
12,5	9,26	1500	170	0,650	0,653	0,485	0,483
15	9,17	1500	170	0,700	0,713	0,535	0,525
17,5	9,00	1500	170	0,780	0,773	0,550	0,555
20	8,68	1500	170	0,820	0,833	0,577	0,568
22,5	8,17	1500	170	0,900	0,891	0,556	0,562
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø[mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
3,75	9,22	1500	170	0,440	0,439	0,214	0,214
6,25	9,30	1500	170	0,510	0,501	0,310	0,316
11,25	9,32	1500	170	0,620	0,623	0,461	0,458
21,25	8,42	1500	170	0,920	0,862	0,530	0,566

Q[m³/h]	H [m]	n[d/d]	Ø [mm]	N [kW]	Nysa[kW]	Verim(ŋ)	VerimYSA(ŋ)
0	11,00	1500	185	0,400	0,402	0	0
2,5	11,10	1500	185	0,480	0,407	0,157	0,186
5	11,17	1500	185	0,520	0,532	0,292	0,286
7,5	11,17	1500	185	0,600	0,597	0,380	0,382
10	11,14	1500	185	0,660	0,662	0,460	0,459
12,5	11,08	1500	185	0,710	0,726	0,531	0,519
15	11,00	1500	185	0,800	0,791	0,562	0,568
17,5	10,83	1500	185	0,860	0,855	0,600	0,603
20	10,50	1500	185	0,920	0,919	0,622	0,622
22,5	10,00	1500	185	0,980	0,982	0,625	0,624
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
3,75	11,14	1500	185	0,500	0,500	0,228	0,228
6,25	11,17	1500	185	0,560	0,564	0,340	0,337
18,75	10,67	1500	185	0,900	0,887	0,605	0,614
21,25	10,25	1500	185	0,950	0,951	0,624	0,624
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
0	13,50	1500	200	0,550	0,547	0	0
2,5	13,68	1500	200	0,600	0,619	0,155	0,150
5	13,85	1500	200	0,700	0,691	0,269	0,273
7,5	13,94	1500	200	0,770	0,762	0,370	0,374
10	14,00	1500	200	0,840	0,833	0,454	0,458
12,5	13,94	1500	200	0,900	0,903	0,527	0,526
15	13,87	1500	200	0,970	0,972	0,584	0,583
17,5	13,60	1500	200	1,040	1,040	0,623	0,623
20	13,43	1500	200	1,100	1,109	0,665	0,660
22,5	13,00	1500	200	1,180	1,175	0,675	0,678
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
1,25	13,60	1500	200	0,580	0,583	0,080	0,079
16,25	13,74	1500	200	1,010	1,006	0,602	0,604
18,75	13,52	1500	200	1,070	1,074	0,645	0,643
21,25	13,22	1500	200	1,140	1,142	0,671	0,670

Ek .7. B 50-200 pompasının 3000 d/d için karakteristik eğrilerden alınan datalar ve YSA'nın ürettiği sonuçlar (2 sayfa)

Q [m³/h]	H [m]	n[d/d]	Ø [mm]	N [kW]	Nysa [kW]	Verim(ŋ)	VerimYSA(ŋ)
0	30,00	3000	155	2,000	2,081	0	0
5	30,69	3000	155	2,670	2,629	0,157	0,159
10	30,93	3000	155	3,170	3,143	0,266	0,268
15	30,93	3000	155	3,670	3,639	0,344	0,347
20	30,35	3000	155	4,100	4,091	0,403	0,404
25	29,66	3000	155	4,500	4,534	0,449	0,445
30	28,62	3000	155	5,000	4,950	0,468	0,472
35	27,93	3000	155	5,430	5,393	0,490	0,494
40	26,55	3000	155	5,700	5,783	0,507	0,500
45	24,45	3000	155	6,000	6,120	0,499	0,490
50	22,03	3000	155	6,500	6,433	0,461	0,466
55	20,00	3000	155	6,830	6,774	0,439	0,442
60	16,55	3000	155	7,000	7,010	0,386	0,386
Q [m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa [kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
7,5	30,80	3000	155	2,900	2,886	0,217	0,218
22,5	30,00	3000	155	4,300	4,312	0,427	0,426
32,5	28,28	3000	155	5,200	5,172	0,481	0,484
37,5	27,24	3000	155	5,550	5,588	0,501	0,498
Q [m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa [kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
0	37,58	3000	170	3,000	3,065	0	0
5	37,76	3000	170	3,670	3,634	0,140	0,141
10	38,11	3000	170	4,170	4,216	0,249	0,246
15	37,76	3000	170	4,830	4,744	0,319	0,325
20	37,58	3000	170	5,330	5,284	0,384	0,387
25	37,07	3000	170	5,770	5,799	0,437	0,435
30	36,38	3000	170	6,330	6,299	0,470	0,472
35	35,00	3000	170	6,700	6,745	0,498	0,495
40	33,62	3000	170	7,100	7,191	0,516	0,509
45	31,90	3000	170	7,660	7,611	0,510	0,514
50	29,66	3000	170	8,000	7,990	0,505	0,505
55	27,07	3000	170	8,330	8,342	0,487	0,486
60	23,80	3000	170	8,660	8,642	0,449	0,450
Q [m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa [kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
2,5	37,67	3000	170	3,330	3,350	0,077	0,077
12,5	38,11	3000	170	4,500	4,494	0,288	0,289
22,5	37,33	3000	170	5,550	5,542	0,412	0,413
47,5	30,05	3000	170	7,830	7,744	0,496	0,502

Q [m³/h]	H [m]	n[d/d]	Ø [mm]	N [kW]	Nysa[kW]	Verim(ŋ)	VerimYSA(ŋ)
0	44,31	3000	185	3,700	3,754	0	0
5	44,66	3000	185	4,330	4,344	0,140	0,140
10	45,00	3000	185	5,000	4,932	0,245	0,248
15	45,00	3000	185	5,550	5,501	0,331	0,334
20	44,66	3000	185	6,000	6,053	0,405	0,402
25	44,31	3000	185	6,660	6,603	0,453	0,457
30	43,62	3000	185	7,100	7,135	0,502	0,499
35	42,59	3000	185	7,660	7,648	0,530	0,531
40	40,69	3000	185	8,000	8,116	0,554	0,546
45	38,80	3000	185	8,660	8,584	0,549	0,554
50	36,73	3000	185	9,000	9,042	0,556	0,553
55	33,80	3000	185	9,500	9,454	0,533	0,535
60	30,69	3000	185	9,850	9,857	0,509	0,509
Q [m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
22,5	44,48	3000	185	6,330	6,328	0,431	0,431
32,5	43,10	3000	185	7,380	7,391	0,517	0,516
47,5	37,76	3000	185	8,830	8,812	0,553	0,554
57,5	32,24	3000	185	9,680	9,655	0,522	0,523
Q [m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
0	53,97	3000	200	4,000	4,111	0	0
5	54,31	3000	200	4,830	4,755	0,153	0,156
10	55,00	3000	200	5,480	5,417	0,273	0,277
15	55,00	3000	200	6,000	6,041	0,374	0,372
20	55,00	3000	200	6,700	6,664	0,447	0,450
25	54,66	3000	200	7,330	7,269	0,508	0,512
30	54,40	3000	200	7,830	7,877	0,568	0,564
35	53,28	3000	200	8,330	8,437	0,610	0,602
40	51,80	3000	200	9,000	8,978	0,627	0,628
45	50,00	3000	200	9,500	9,501	0,645	0,645
50	47,42	3000	200	10,000	9,981	0,646	0,647
55	44,66	3000	200	10,490	10,451	0,638	0,640
60	41,04	3000	200	10,850	10,874	0,618	0,617
Q [m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
2,5	54,14	3000	200	4,420	4,433	0,083	0,083
17,5	55	3000	200	6,350	6,353	0,413	0,413
32,5	53,84	3000	200	7,580	8,157	0,629	0,584
57,5	42,85	3000	200	10,670	10,663	0,629	0,629

Ek. 8. B 65-200 pompasının 1500 d/d için karakteristik eğrilerden alınan datalar ve YSA'nın ürettiği sonuçlar (2 sayfa)

Q[m³/h]	H [m]	n[d/d]	Ø [mm]	N [kW]	Nysa[kW]	Verim(ŋ)	VerimYSA(ŋ)
0	8,01	1500	155	0,300	0,298	0	0
2,5	8,02	1500	155	0,350	0,350	0,156	0,156
5	8,02	1500	155	0,400	0,401	0,273	0,272
7,5	8,01	1500	155	0,450	0,453	0,364	0,361
10	8,00	1500	155	0,500	0,505	0,436	0,432
12,5	7,77	1500	155	0,550	0,552	0,481	0,479
15	7,51	1500	155	0,600	0,603	0,511	0,508
17,5	7,00	1500	155	0,650	0,650	0,513	0,513
20	6,00	1500	155	0,700	0,693	0,467	0,472
22,5	2,80	1500	155	0,710	0,713	0,242	0,240
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
8,75	8,00	1500	155	0,430	0,479	0,443	0,398
13,75	7,64	1500	155	0,580	0,579	0,493	0,494
16,25	7,26	1500	155	0,630	0,627	0,510	0,512
21,25	4,40	1500	155	0,700	0,703	0,364	0,362
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
0	9,77	1500	170	0,450	0,435	0	0
2,5	9,85	1500	170	0,500	0,502	0,134	0,134
5	9,94	1500	170	0,560	0,568	0,242	0,238
7,5	9,94	1500	170	0,620	0,633	0,327	0,321
10	9,90	1500	170	0,700	0,698	0,385	0,387
12,5	9,85	1500	170	0,760	0,762	0,441	0,440
15	9,60	1500	170	0,830	0,824	0,472	0,476
17,5	9,26	1500	170	0,900	0,885	0,490	0,499
20	8,80	1500	170	0,960	0,944	0,499	0,508
22,5	8,00	1500	170	0,980	0,999	0,500	0,491
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
1,25	9,80	1500	170	0,480	0,468	0,070	0,071
16,25	9,43	1500	170	0,870	0,854	0,480	0,488
18,75	9,03	1500	170	0,930	0,971	0,496	0,475
21,25	8,40	1500	170	0,970	0,914	0,501	0,532

Q[m³/h]	H [m]	n[d/d]	Ø [mm]	N [kW]	Nysa[kW]	Verim(ŋ)	VerimYSA(ŋ)
0	11,50	1500	185	0,650	0,655	0	0
2,5	11,60	1500	185	0,720	0,719	0,110	0,110
5	11,68	1500	185	0,790	0,783	0,201	0,203
7,5	11,70	1500	185	0,850	0,848	0,281	0,282
10	11,70	1500	185	0,920	0,912	0,346	0,349
12,5	11,68	1500	185	0,970	0,976	0,410	0,407
15	11,60	1500	185	1,030	1,041	0,460	0,455
17,5	11,34	1500	185	1,100	1,105	0,491	0,489
20	11,00	1500	185	1,170	1,169	0,512	0,513
22,5	10,50	1500	185	1,240	1,233	0,519	0,522
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa [kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
3,75	11,64	1500	185	0,760	0,751	0,156	0,158
6,25	11,69	1500	185	0,820	0,815	0,243	0,244
8,75	11,70	1500	185	0,890	0,880	0,313	0,317
21,25	10,75	1500	185	1,210	1,201	0,514	0,518
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa [kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
0	13,01	1500	200	0,900	0,878	0	0
2,5	13,34	1500	200	0,960	0,963	0,095	0,094
5	13,65	1500	200	1,040	1,047	0,179	0,177
7,5	13,77	1500	200	1,120	1,133	0,251	0,248
10	13,87	1500	200	1,200	1,219	0,315	0,310
12,5	13,80	1500	200	1,300	1,306	0,361	0,360
15	13,77	1500	200	1,400	1,393	0,402	0,404
17,5	13,50	1500	200	1,500	1,482	0,429	0,434
20	13,17	1500	200	1,580	1,571	0,454	0,457
22,5	12,83	1500	200	1,650	1,660	0,476	0,474
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa [kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
1,25	13,18	1500	200	0,930	0,921	0,048	0,049
3,75	13,50	1500	200	1,000	1,005	0,138	0,137
18,75	13,34	1500	200	1,540	1,526	0,442	0,446
21,25	13,00	1500	200	1,620	1,615	0,464	0,466

Ek. 9. B 65-200 pompasının 3000 d/d için karakteristik eğrilerden alınan datalar ve YSA'nın ürettiği sonuçlar (2 sayfa)

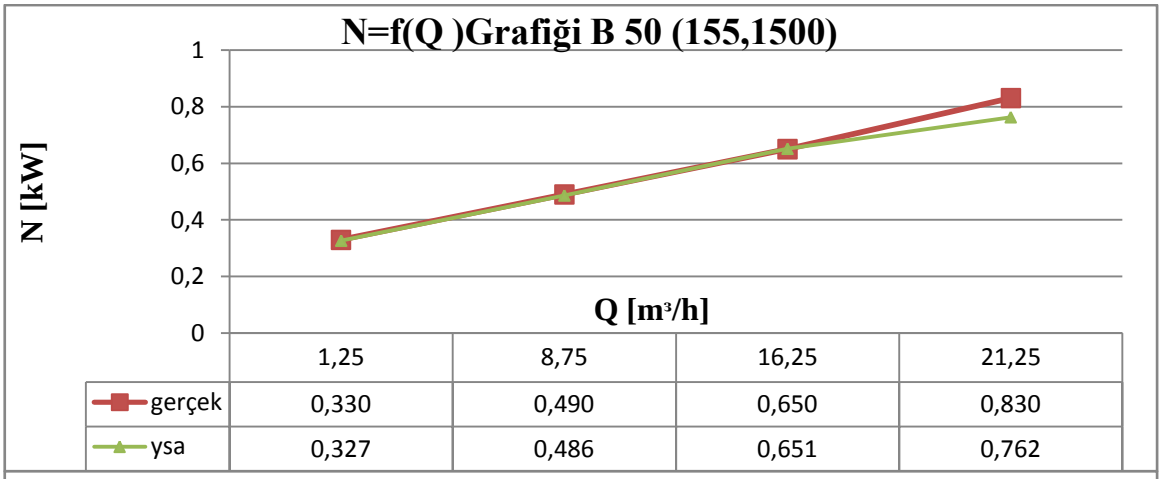
[m³/h]	H [m]	n[d/d]	Ø [mm]	N [kW]	Nysa[kW]	Verim(ŋ)	VerimYSA(ŋ)
0	32,50	3000	155	2,900	2,796	0	0
5	33,00	3000	155	3,000	3,070	0,150	0,146
10	33,00	3000	155	3,330	3,329	0,270	0,270
15	32,50	3000	155	3,500	3,572	0,379	0,372
20	31,75	3000	155	3,800	3,809	0,455	0,454
25	31,40	3000	155	4,000	4,058	0,534	0,527
30	30,70	3000	155	4,200	4,297	0,597	0,584
35	30,00	3000	155	4,500	4,536	0,635	0,630
40	28,84	3000	155	4,800	4,761	0,655	0,660
45	25,70	3000	155	5,000	4,925	0,630	0,640
50	21,40	3000	155	5,100	5,053	0,571	0,577
55	12,50	3000	155	4,970	5,039	0,377	0,372
[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
7,5	33,00	3000	155	2,950	3,200	0,228	0,211
22,5	31,58	3000	155	3,900	3,934	0,496	0,492
42,5	27,27	3000	155	4,900	4,844	0,644	0,652
47,5	23,55	3000	155	5,050	4,990	0,603	0,611
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
0	38,50	3000	170	3,670	3,654	0	0
5	39,50	3000	170	4,000	4,048	0,134	0,133
10	39,70	3000	170	4,500	4,429	0,240	0,244
15	39,90	3000	170	4,800	4,811	0,340	0,339
20	39,90	3000	170	5,200	5,188	0,418	0,419
25	39,70	3000	170	5,500	5,562	0,491	0,486
30	38,90	3000	170	5,900	5,927	0,539	0,536
35	38,85	3000	170	6,330	6,303	0,585	0,587
40	37,45	3000	170	6,700	6,657	0,609	0,613
45	35,07	3000	170	7,000	6,994	0,614	0,614
50	33,60	3000	170	7,330	7,347	0,624	0,623
55	30,00	3000	170	7,660	7,663	0,587	0,586
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
2,5	39,00	3000	170	3,840	3,851	0,069	0,069
17,5	39,90	3000	170	5,000	4,999	0,380	0,380
47,5	34,33	3000	170	7,170	7,170	0,619	0,619
52,5	31,80	3000	170	7,500	7,505	0,606	0,606

Q[m³/h]	H [m]	n[d/d]	Ø [mm]	N [kW]	Nysa[kW]	Verim(ŋ)	VerimYSA(ŋ)
0	46,40	3000	185	5,170	5,196	0	0
5	46,75	3000	185	5,700	5,666	0,112	0,112
10	47,10	3000	185	6,170	6,137	0,208	0,209
15	47,45	3000	185	6,660	6,609	0,291	0,293
20	47,45	3000	185	7,000	7,072	0,369	0,365
25	47,40	3000	185	7,500	7,535	0,430	0,428
30	46,75	3000	185	8,000	7,986	0,477	0,478
35	46,40	3000	185	8,350	8,443	0,530	0,524
40	46,05	3000	185	9,000	8,900	0,557	0,564
45	45,20	3000	185	9,330	9,346	0,594	0,593
50	43,85	3000	185	9,830	9,782	0,607	0,610
55	41,75	3000	185	10,170	10,203	0,615	0,613
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
2,5	46,57	3000	185	5,440	5,431	0,058	0,058
22,5	47,40	3000	185	7,250	7,303	0,401	0,398
37,5	46,57	3000	185	8,680	8,678	0,548	0,548
52,5	42,75	3000	185	10,000	9,992	0,611	0,612
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
0	54,0	3000	200	6,660	6,530	0	0
5	54,5	3000	200	7,000	7,099	0,106	0,105
10	54,8	3000	200	7,700	7,670	0,194	0,195
15	55,0	3000	200	8,170	8,244	0,275	0,273
20	55,0	3000	200	8,870	8,821	0,338	0,340
25	55,0	3000	200	9,300	9,398	0,403	0,398
30	54,9	3000	200	10,000	9,977	0,449	0,450
35	54,5	3000	200	10,500	10,561	0,495	0,492
40	54,0	3000	200	11,170	11,146	0,527	0,528
45	53,0	3000	200	11,830	11,738	0,549	0,553
50	52,5	3000	200	12,330	12,325	0,580	0,580
55	51,5	3000	200	12,900	12,919	0,598	0,597
Q[m³/h]	**H [m]**	**n[d/d]**	**Ø [mm]**	**N [kW]**	**Nysa[kW]**	**Verim(ŋ)**	**VerimYSA(ŋ)**
2,5	54,25	3000	200	6,830	6,816	0,054	0,054
22,5	55,00	3000	200	9,090	9,107	0,371	0,370
42,5	54,50	3000	200	11,500	11,427	0,549	0,552
52,5	52,00	3000	200	12,620	12,624	0,589	0,589

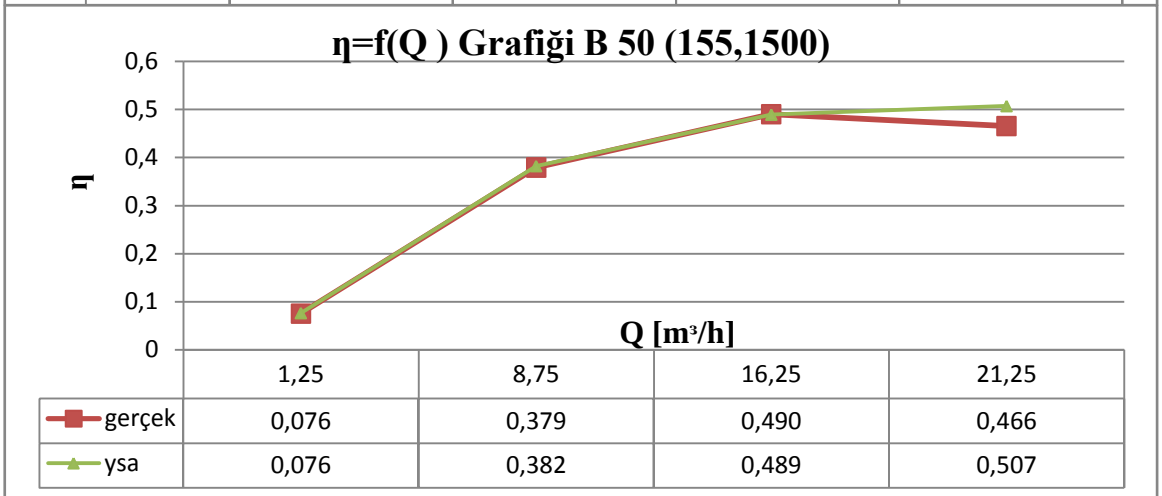
Ek. 10. Hata değerleri

Pompa Tipi	Devir sayısı (d/d)	Giriş çapı (mm)	%Hata toplamı (genel)	%Hata toplamı (ara değer)
B 50-200	1500	155	4,8	7,6
B 50-200	1500	170	8,6	7,2
B 50-200	1500	185	12,5	1,8
B 50-200	1500	200	6,5	1,3
B 50-200	3000	155	64,6	9,2
B 50-200	3000	170	56,4	11,9
B 50-200	3000	185	62,8	5,6
B 50-200	3000	200	64,6	60,1
B 65-200	1500	155	2,8	5,6
B 65-200	1500	170	9,8	12,4
B 65-200	1500	185	5,3	3,3
B 65-200	1500	200	1,5	3,3
B 65-200	3000	155	67,6	40,2
B 65-200	3000	170	34,2	1,7
B 65-200	3000	185	55,5	7,2
B 65-200	3000	200	70,2	10,8

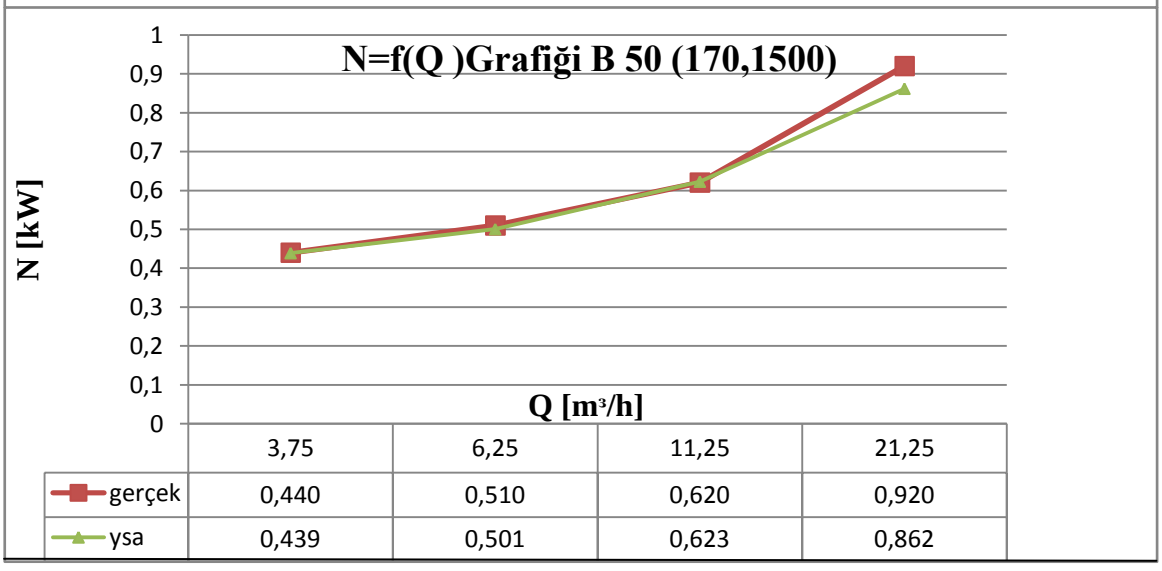
Ek. 11. Ara değerler için oluşturulan N=f(Q) ve η=f(Q) grafikleri (11 sayfa)



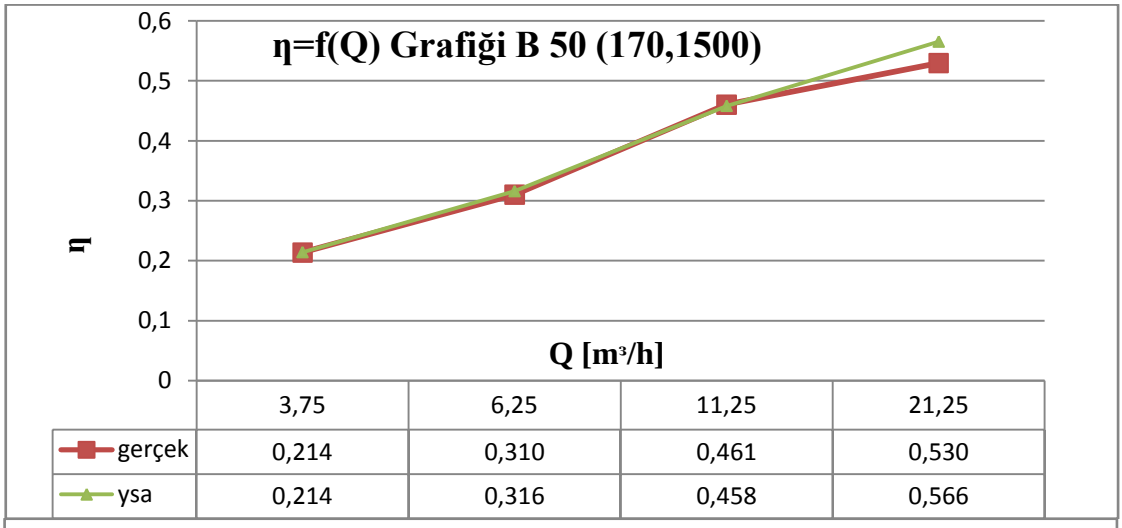
	1,25	8,75	16,25	21,25
gerçek	0,330	0,490	0,650	0,830
ysa	0,327	0,486	0,651	0,762



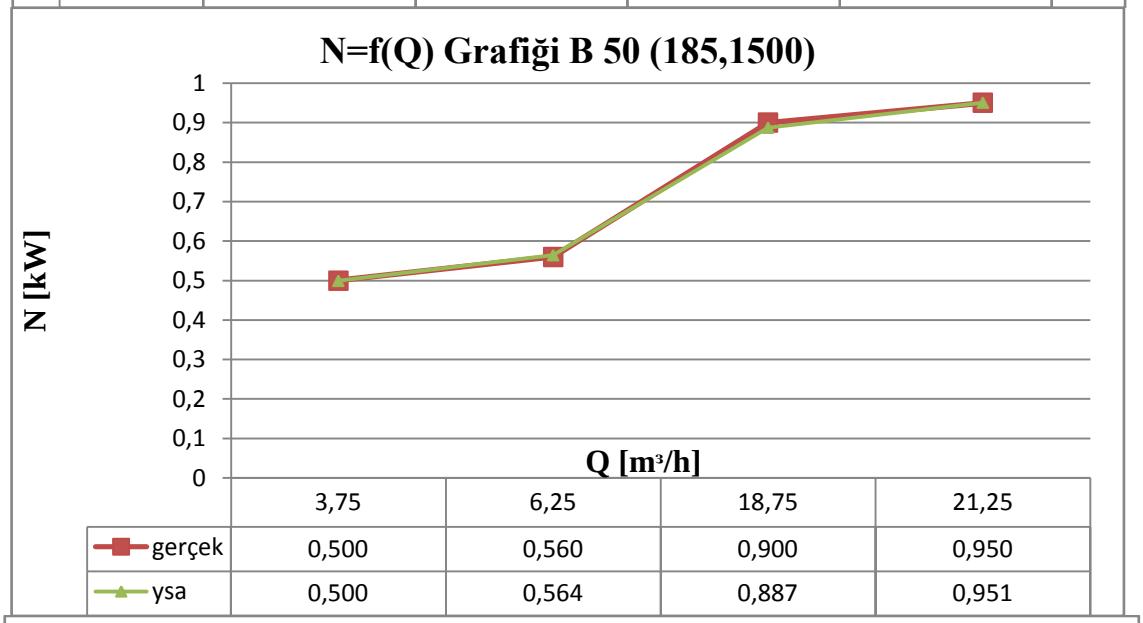
	1,25	8,75	16,25	21,25
gerçek	0,076	0,379	0,490	0,466
ysa	0,076	0,382	0,489	0,507



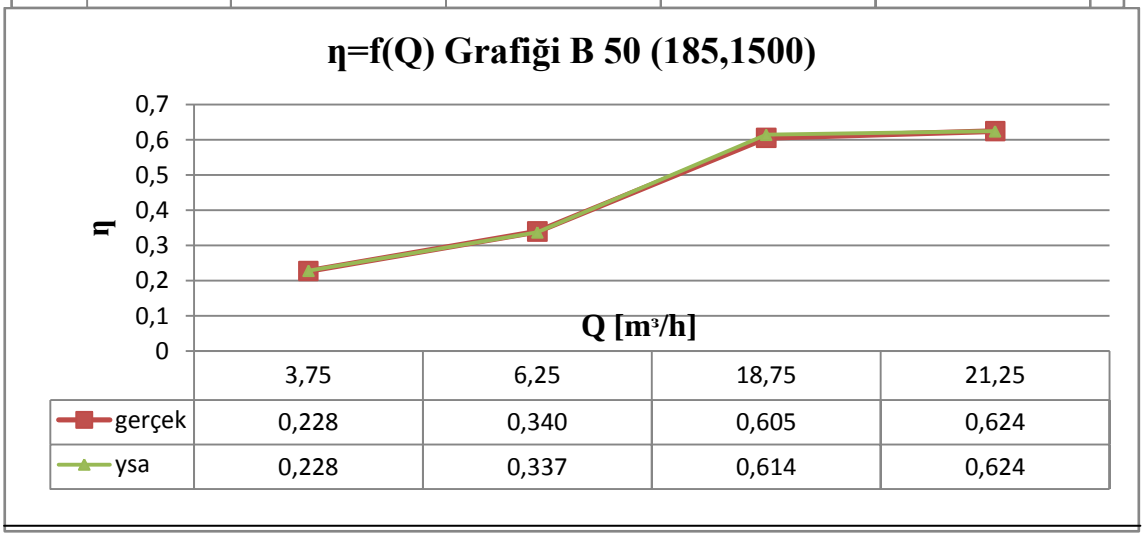
	3,75	6,25	11,25	21,25
gerçek	0,440	0,510	0,620	0,920
ysa	0,439	0,501	0,623	0,862



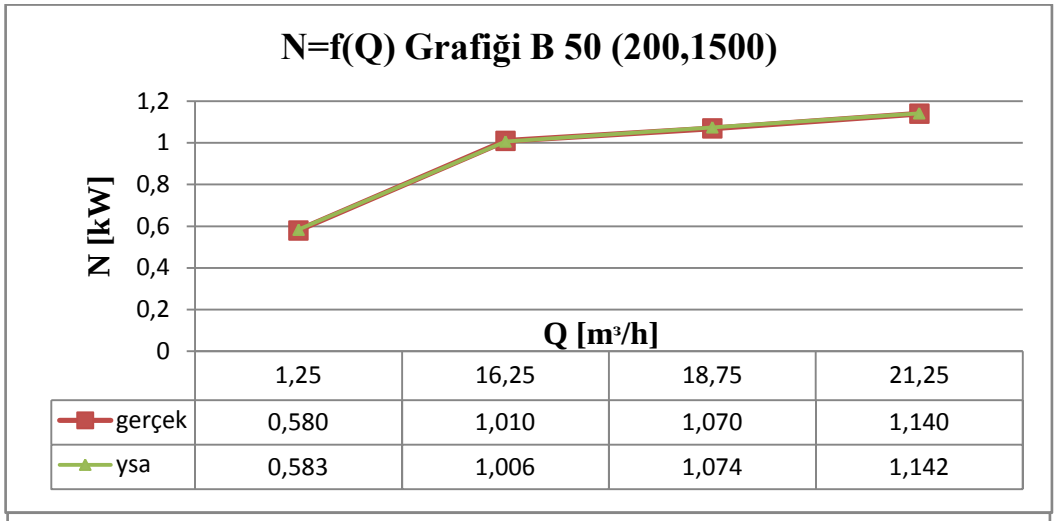
	3,75	6,25	11,25	21,25
gerçek	0,214	0,310	0,461	0,530
ysa	0,214	0,316	0,458	0,566



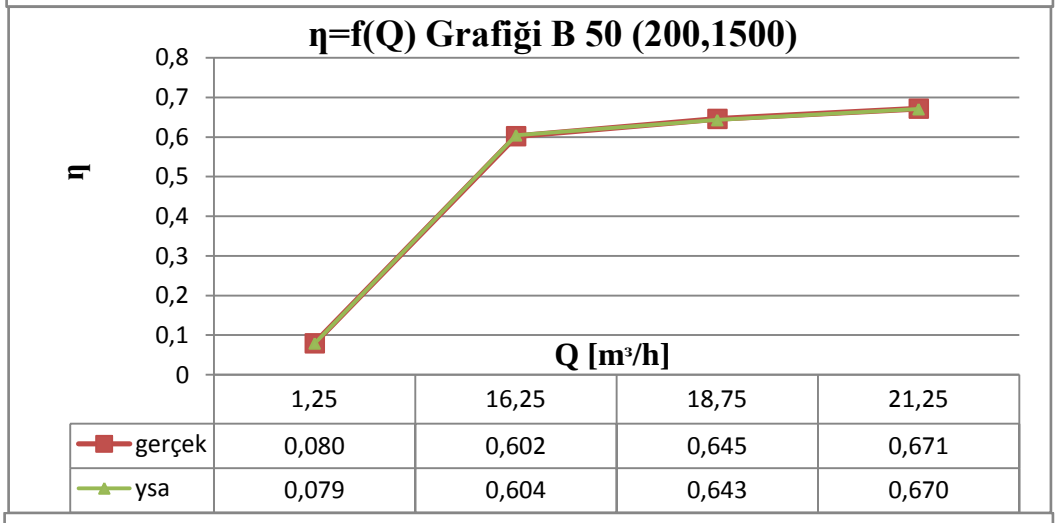
	3,75	6,25	18,75	21,25
gerçek	0,500	0,560	0,900	0,950
ysa	0,500	0,564	0,887	0,951



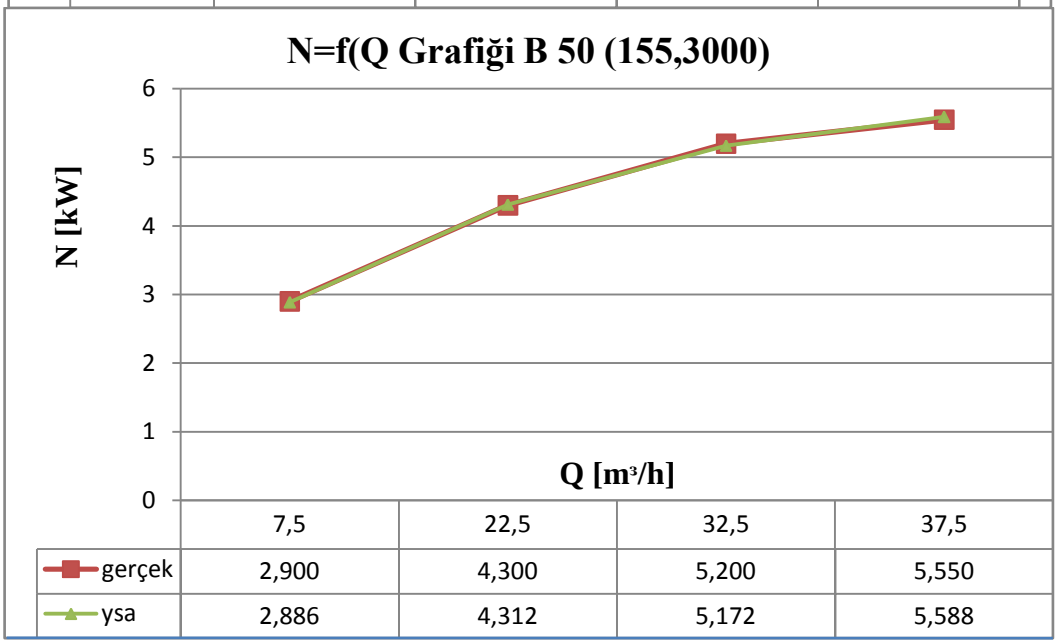
	3,75	6,25	18,75	21,25
gerçek	0,228	0,340	0,605	0,624
ysa	0,228	0,337	0,614	0,624



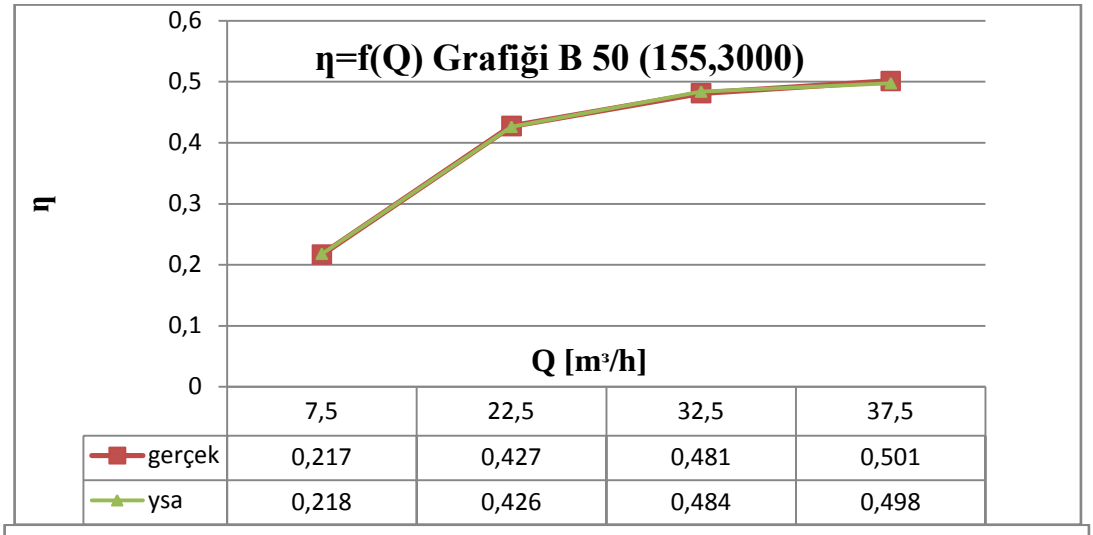
	1,25	16,25	18,75	21,25
gerçek	0,580	1,010	1,070	1,140
ysa	0,583	1,006	1,074	1,142



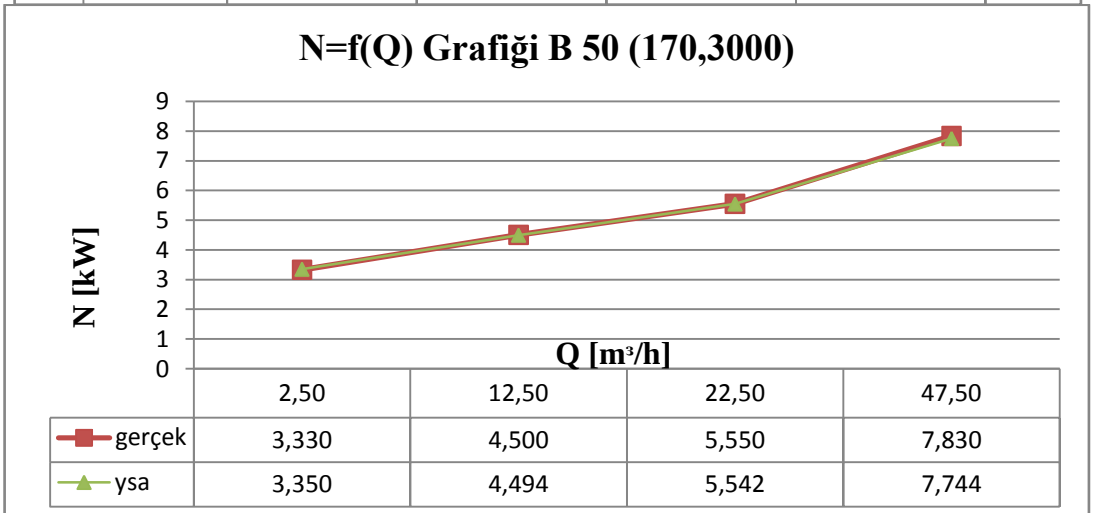
	1,25	16,25	18,75	21,25
gerçek	0,080	0,602	0,645	0,671
ysa	0,079	0,604	0,643	0,670



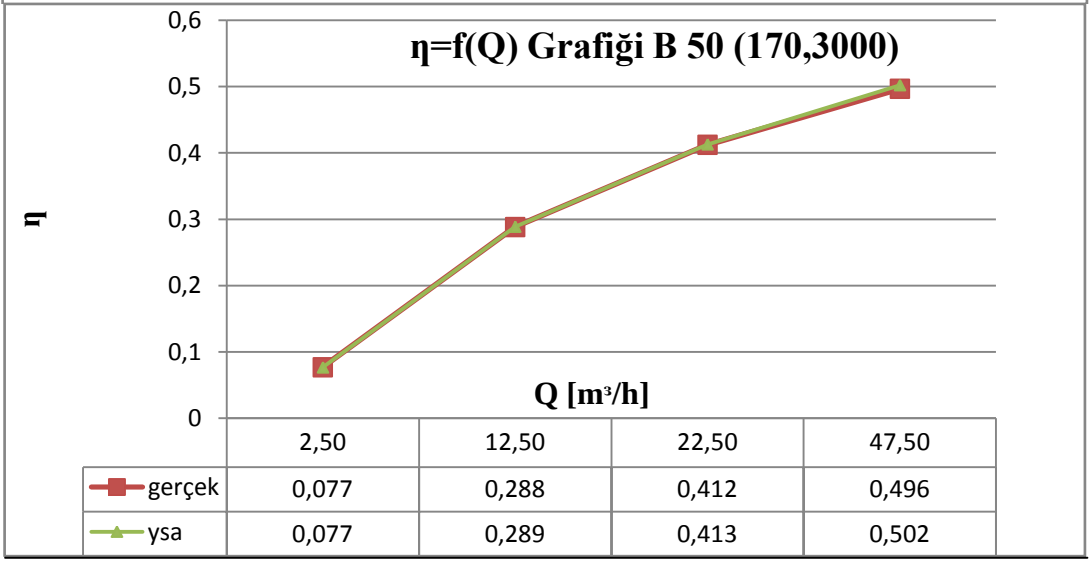
	7,5	22,5	32,5	37,5
gerçek	2,900	4,300	5,200	5,550
ysa	2,886	4,312	5,172	5,588



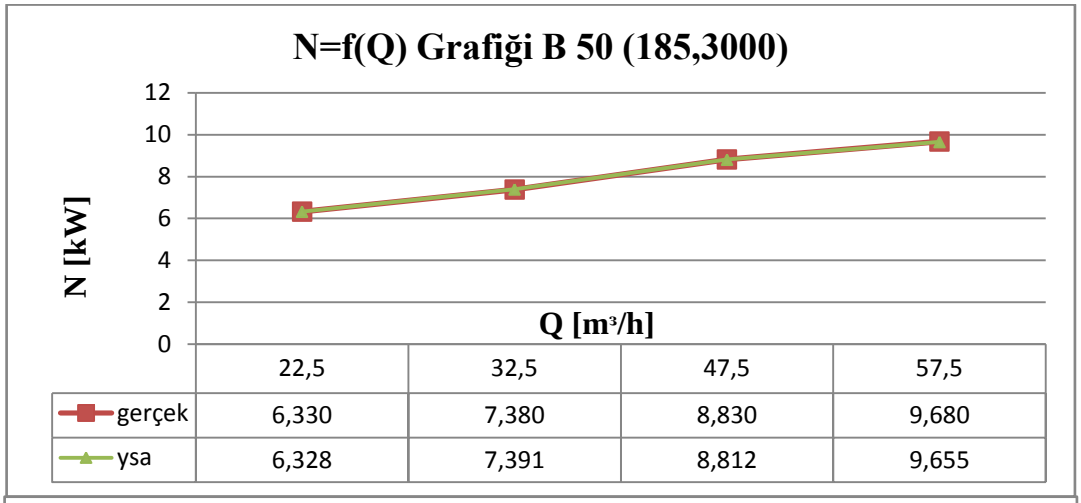
	7,5	22,5	32,5	37,5
gerçek	0,217	0,427	0,481	0,501
ysa	0,218	0,426	0,484	0,498



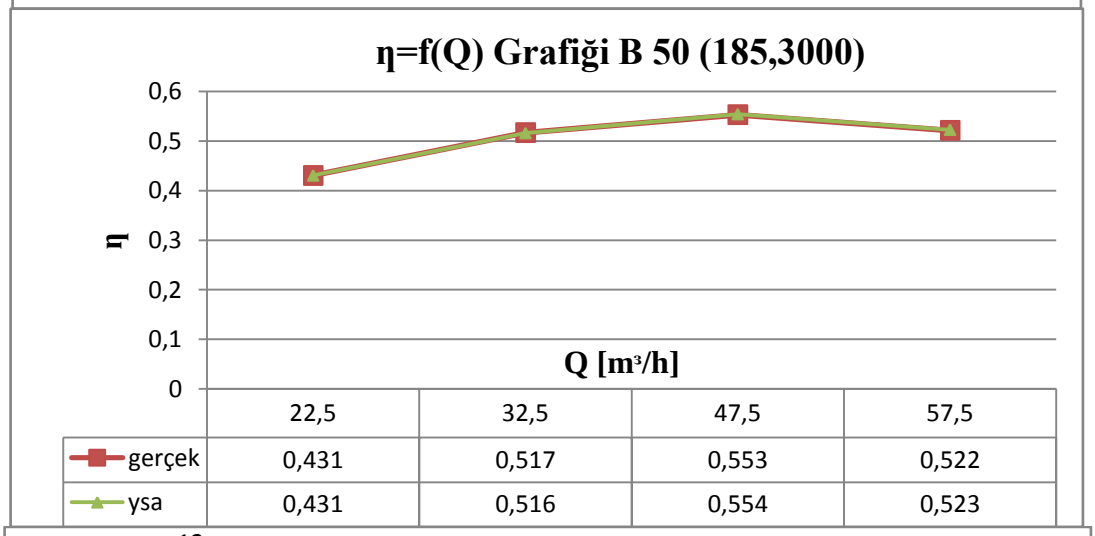
	2,50	12,50	22,50	47,50
gerçek	3,330	4,500	5,550	7,830
ysa	3,350	4,494	5,542	7,744



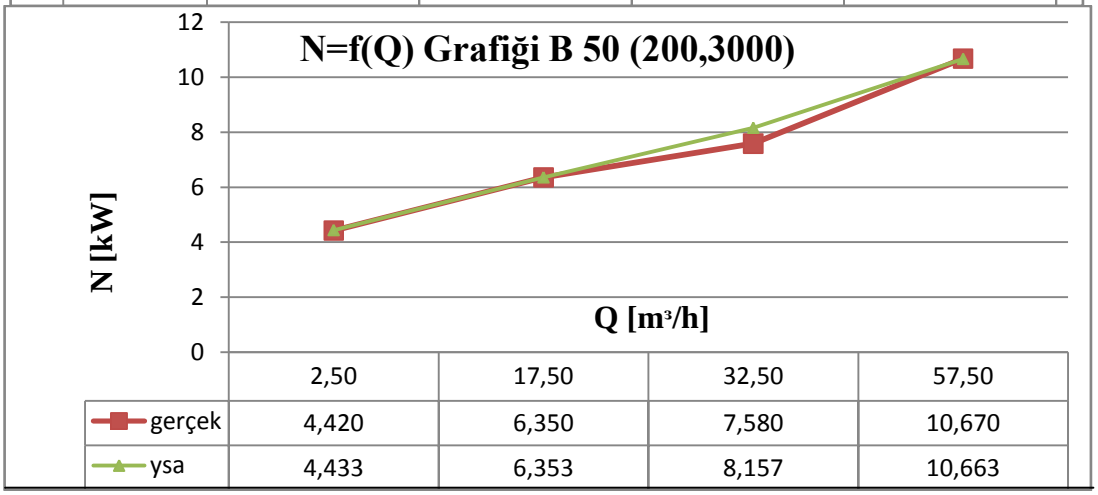
	2,50	12,50	22,50	47,50
gerçek	0,077	0,288	0,412	0,496
ysa	0,077	0,289	0,413	0,502



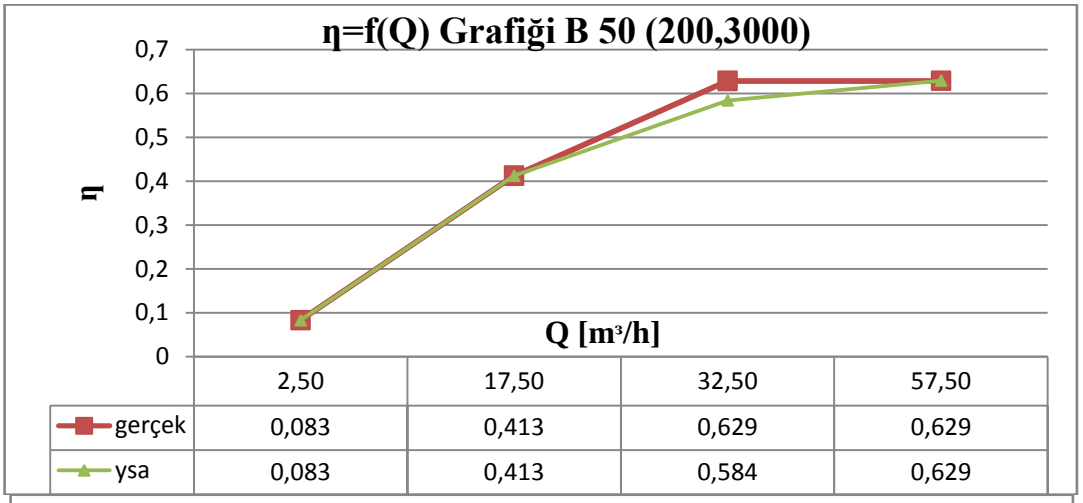
	22,5	32,5	47,5	57,5
gerçek	6,330	7,380	8,830	9,680
ysa	6,328	7,391	8,812	9,655



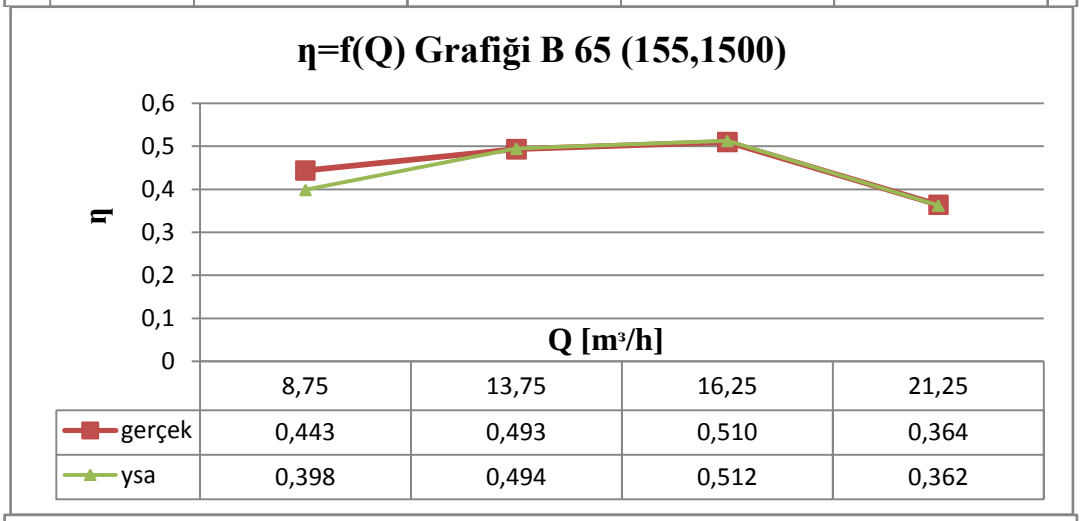
	22,5	32,5	47,5	57,5
gerçek	0,431	0,517	0,553	0,522
ysa	0,431	0,516	0,554	0,523



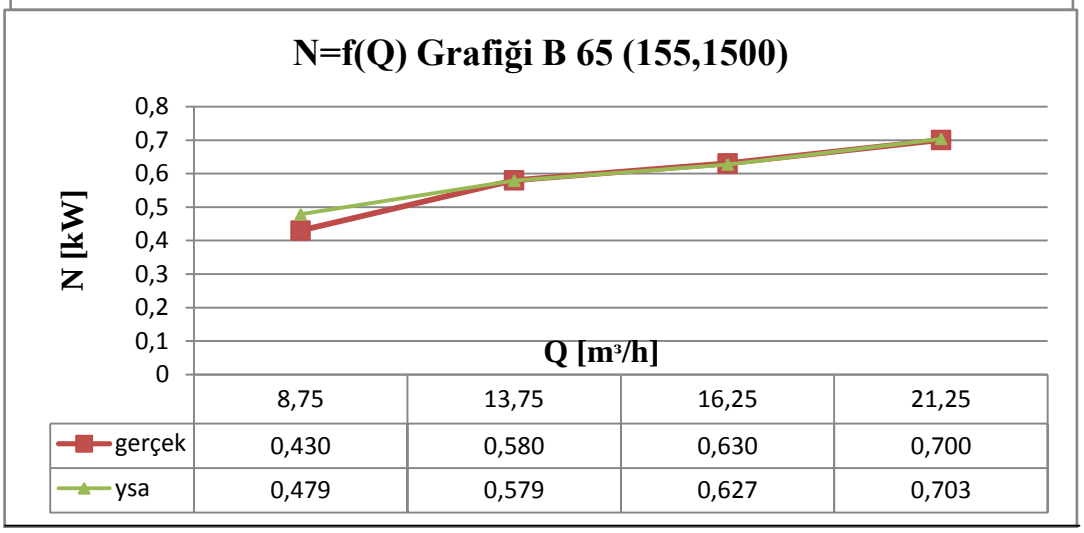
	2,50	17,50	32,50	57,50
gerçek	4,420	6,350	7,580	10,670
ysa	4,433	6,353	8,157	10,663



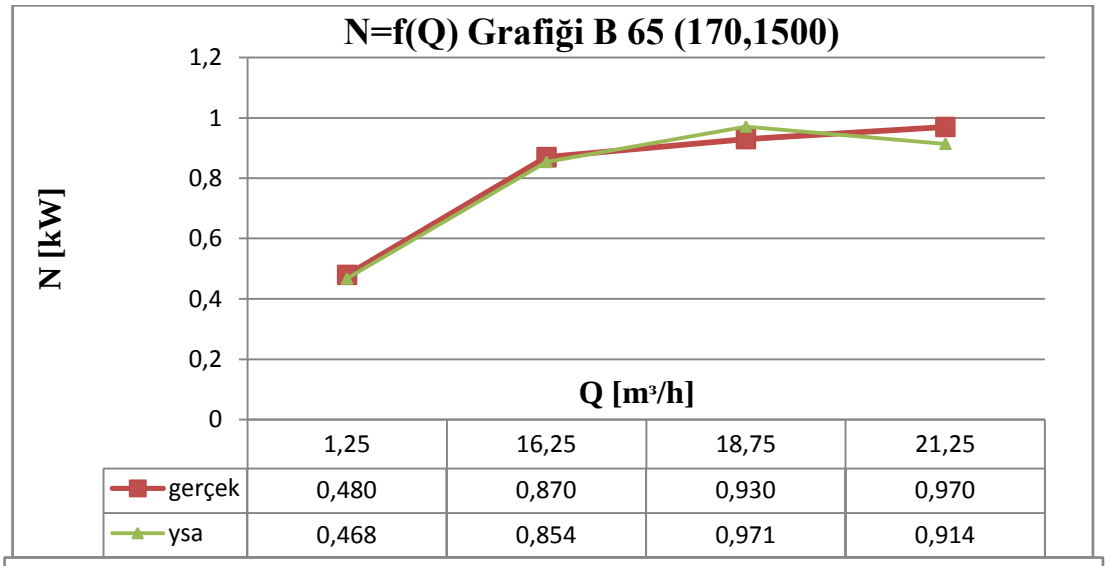
	2,50	17,50	32,50	57,50
gerçek	0,083	0,413	0,629	0,629
ysa	0,083	0,413	0,584	0,629



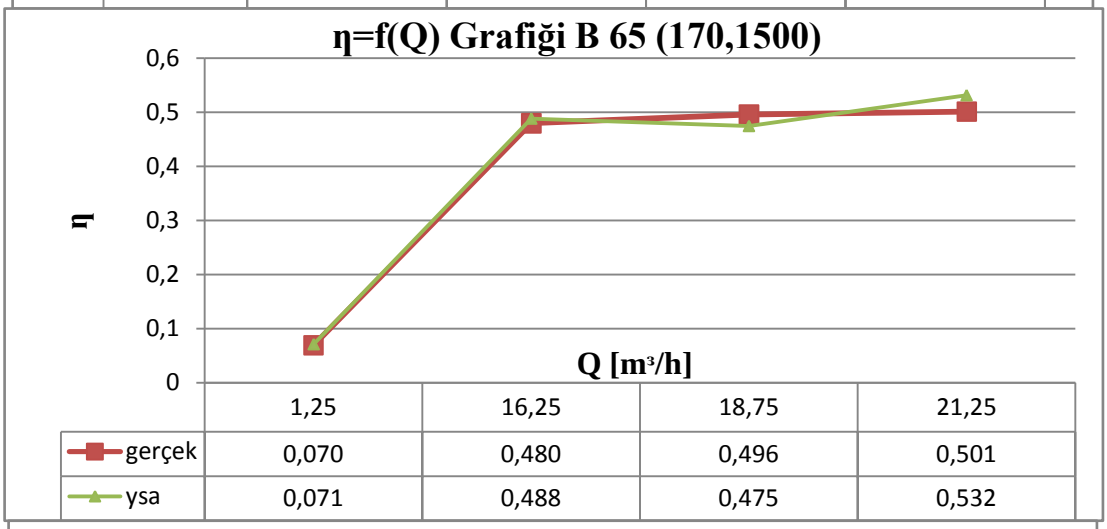
	8,75	13,75	16,25	21,25
gerçek	0,443	0,493	0,510	0,364
ysa	0,398	0,494	0,512	0,362



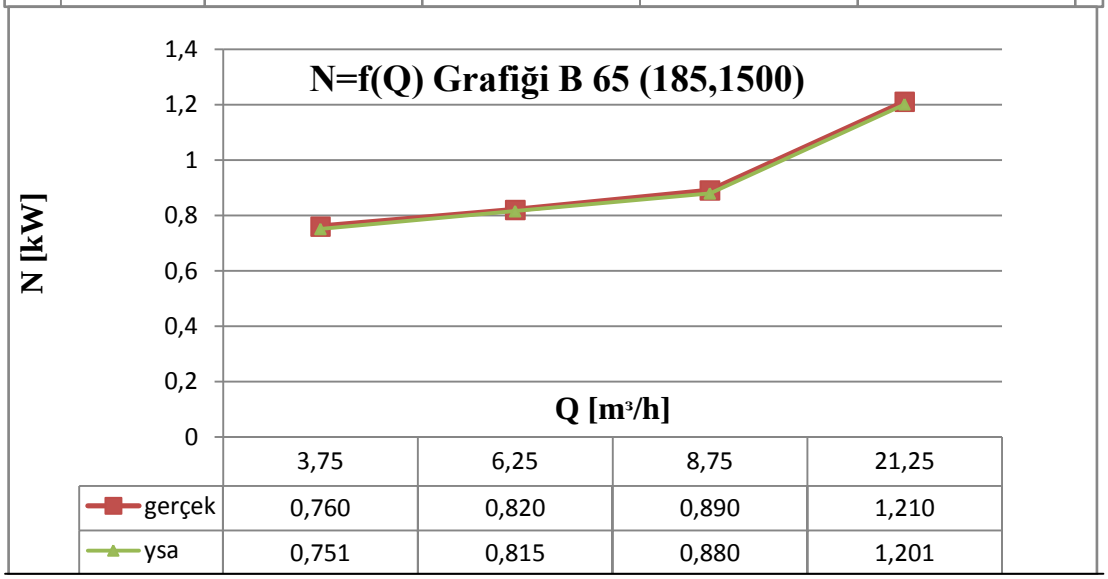
	8,75	13,75	16,25	21,25
gerçek	0,430	0,580	0,630	0,700
ysa	0,479	0,579	0,627	0,703



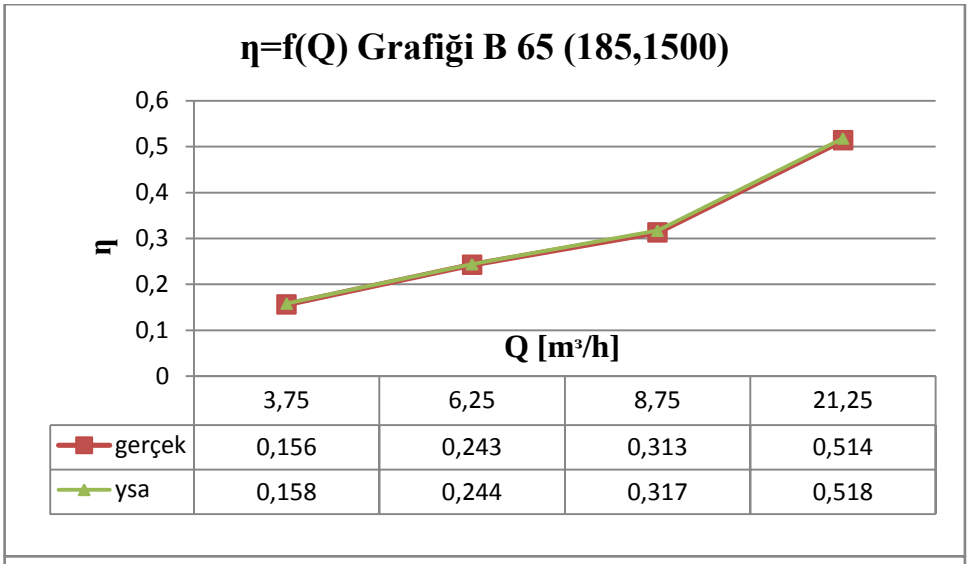
	1,25	16,25	18,75	21,25
gerçek	0,480	0,870	0,930	0,970
ysa	0,468	0,854	0,971	0,914



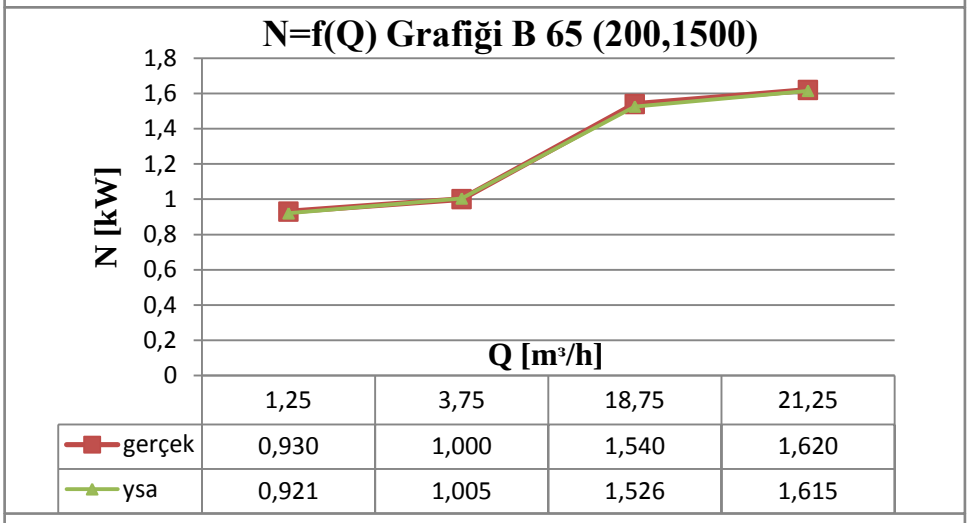
	1,25	16,25	18,75	21,25
gerçek	0,070	0,480	0,496	0,501
ysa	0,071	0,488	0,475	0,532



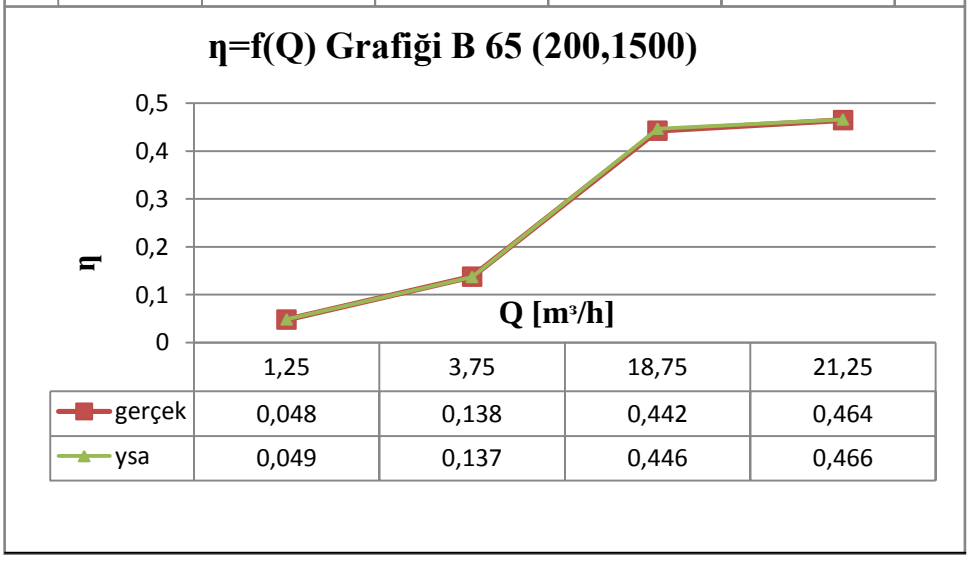
	3,75	6,25	8,75	21,25
gerçek	0,760	0,820	0,890	1,210
ysa	0,751	0,815	0,880	1,201



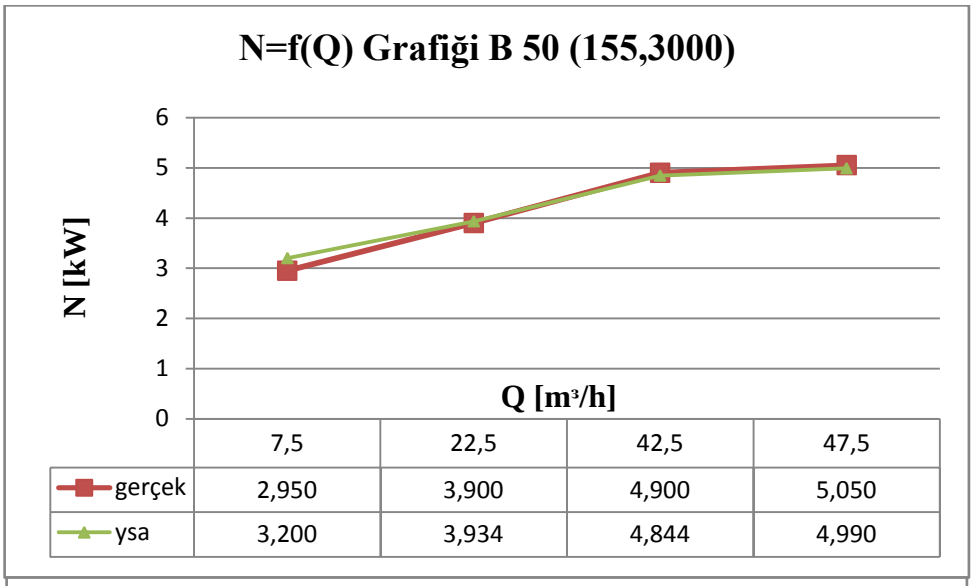
	3,75	6,25	8,75	21,25
gerçek	0,156	0,243	0,313	0,514
ysa	0,158	0,244	0,317	0,518



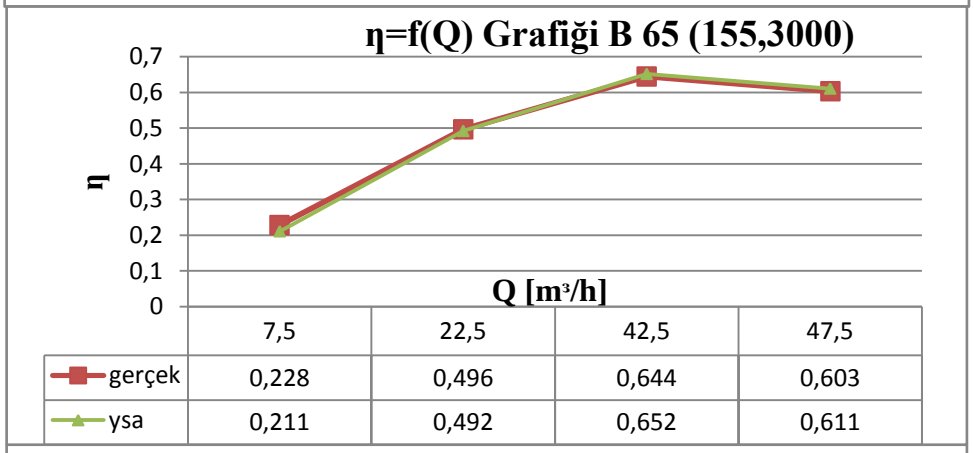
	1,25	3,75	18,75	21,25
gerçek	0,930	1,000	1,540	1,620
ysa	0,921	1,005	1,526	1,615



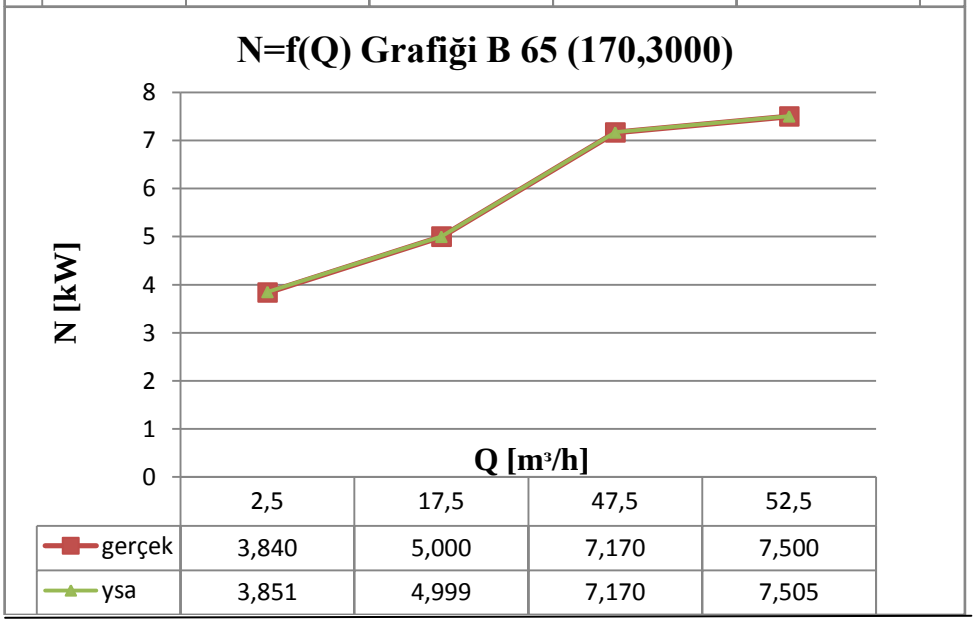
	1,25	3,75	18,75	21,25
gerçek	0,048	0,138	0,442	0,464
ysa	0,049	0,137	0,446	0,466



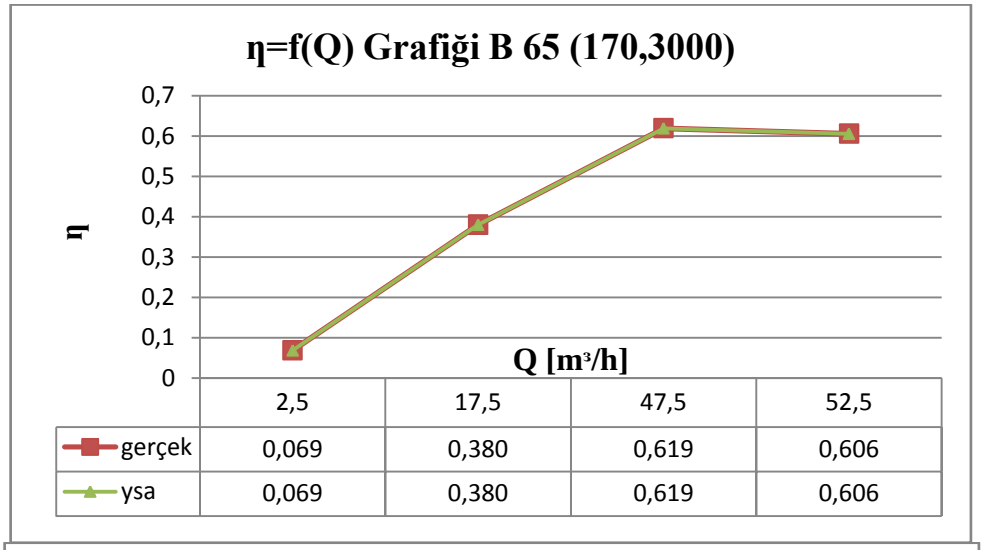
	7,5	22,5	42,5	47,5
gerçek	2,950	3,900	4,900	5,050
ysa	3,200	3,934	4,844	4,990



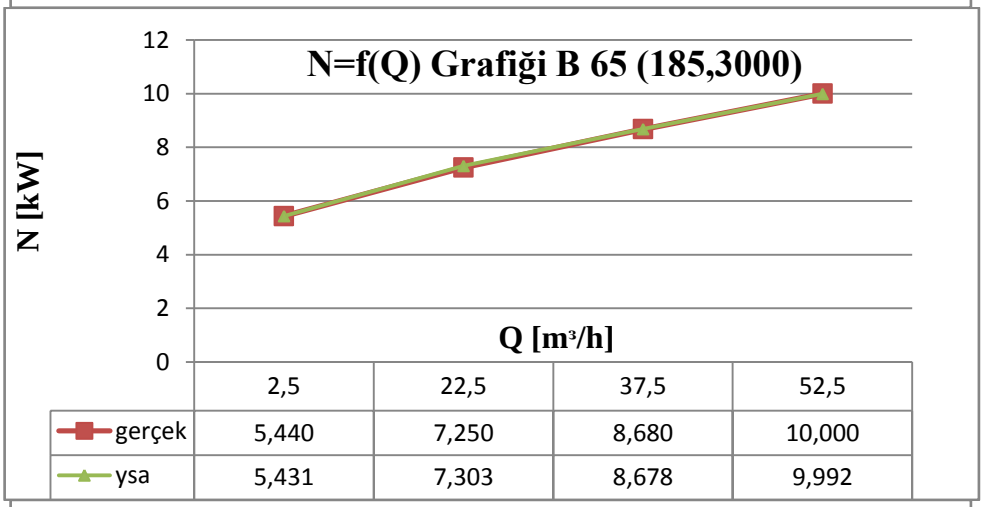
	7,5	22,5	42,5	47,5
gerçek	0,228	0,496	0,644	0,603
ysa	0,211	0,492	0,652	0,611



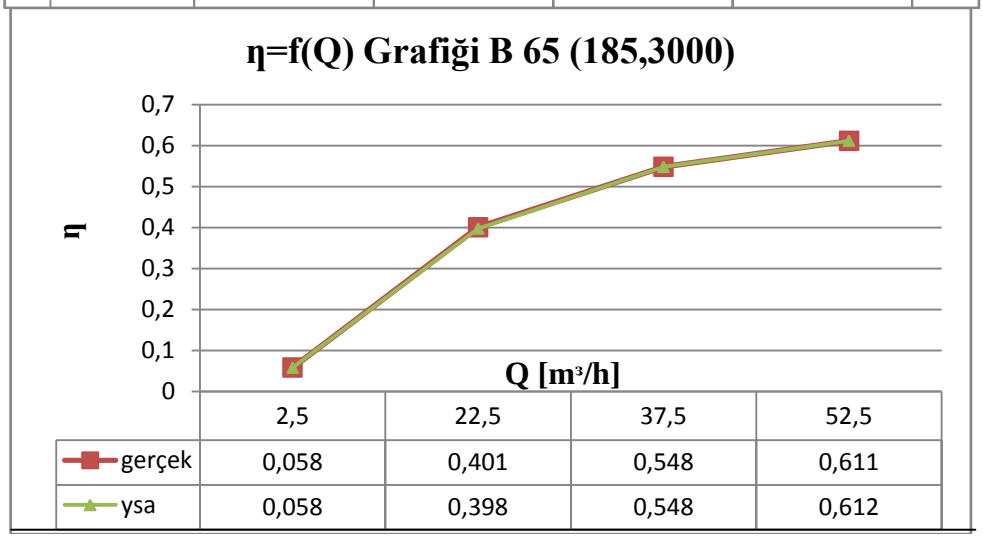
	2,5	17,5	47,5	52,5
gerçek	3,840	5,000	7,170	7,500
ysa	3,851	4,999	7,170	7,505



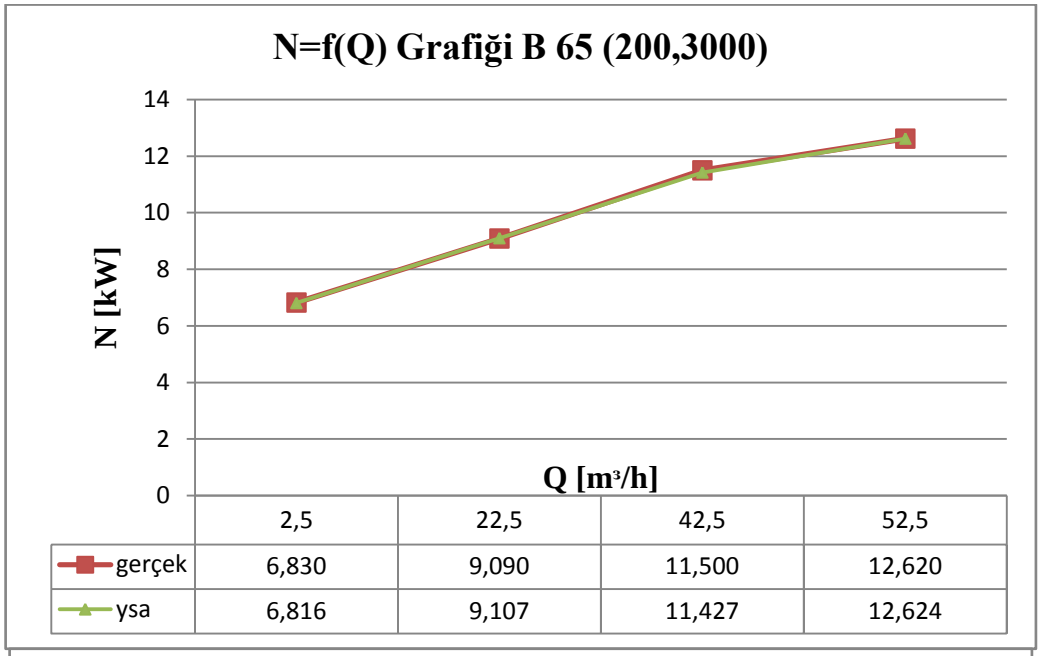
	2,5	17,5	47,5	52,5
gerçek	0,069	0,380	0,619	0,606
ysa	0,069	0,380	0,619	0,606



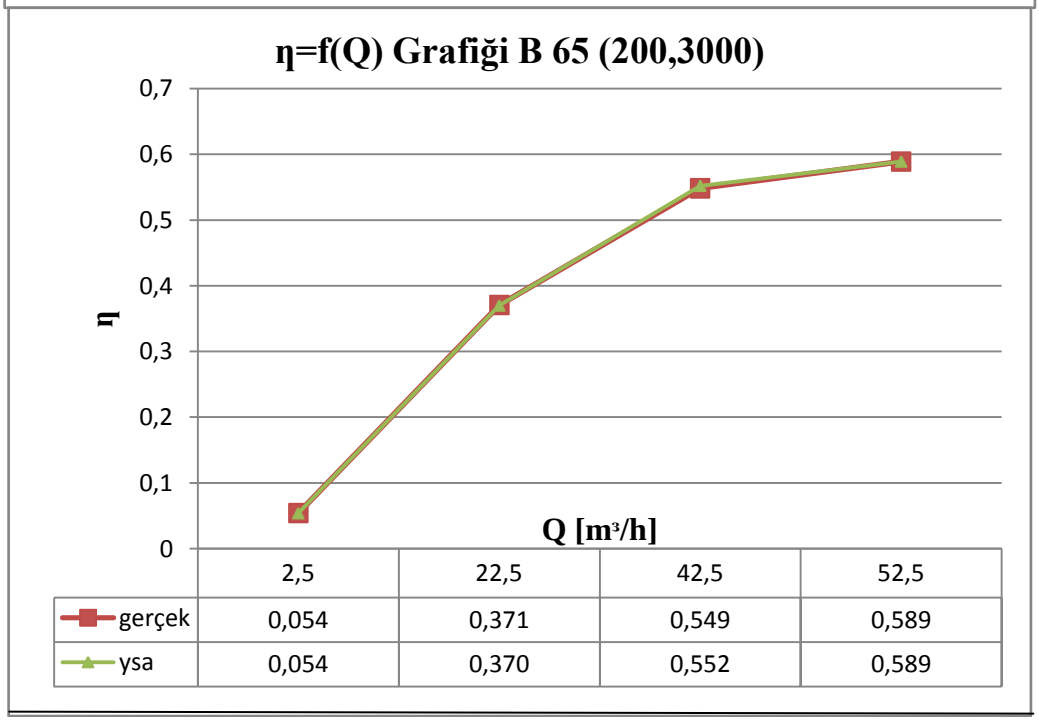
	2,5	22,5	37,5	52,5
gerçek	5,440	7,250	8,680	10,000
ysa	5,431	7,303	8,678	9,992



	2,5	22,5	37,5	52,5
gerçek	0,058	0,401	0,548	0,611
ysa	0,058	0,398	0,548	0,612



	2,5	22,5	42,5	52,5
gerçek	6,830	9,090	11,500	12,620
ysa	6,816	9,107	11,427	12,624



	2,5	22,5	42,5	52,5
gerçek	0,054	0,371	0,549	0,589
ysa	0,054	0,370	0,552	0,589