BULANIK MANTIK VE YAPAY SİNİR AĞLARI İÇİN EĞİTİM YAZILIMI GELİŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

Utku KÖSE

DANIŞMAN
Yrd. Doç. Dr. Ömer DEPERLİOĞLU

BİLGİSAYAR ANABİLİM DALI

HAZİRAN 2010

AFYON KOCATEPE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

BULANIK MANTIK VE YAPAY SİNİR AĞLARI İÇİN EĞİTİM YAZILIMI GELİŞTİRİLMESİ

Utku KÖSE

DANIŞMAN
Yrd. Doç. Dr. Ömer DEPERLİOĞLU

BİLGİSAYAR ANABİLİM DALI

HAZİRAN 2010

ONAY SAYFASI

Yrd. Doç. Dr. Ömer DEPERLİOĞLU danışmanlığında,

Utku KÖSE tarafından hazırlanan

"Bulanık Mantık ve Yapay Sinir Ağları için Eğitim Yazılımı Geliştirilmesi"

başlıklı bu çalışma lisansüstü eğitim ve öğretim yönetmeliğinin ilgili maddeleri uyarınca

...../...../.....

tarihinde aşağıdaki jüri tarafından

Bilgisayar Anabilim Dalında

Yüksek Lisans tezi olarak oybirliği/oy çokluğu ile kabul edilmiştir.

	Ünvanı, Adı, SOYADI	İmza
Başkan	Doç. Dr. Muhammet YÜRÜSOY	
Üye (Danışman)	Yrd. Doç. Dr. Ömer DEPERLİOĞLU	
Üye	Yrd. Doç. Dr. Yüksel OĞUZ	

Afyon Kocatepe Üniversitesi

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun

...../...../..... tarih ve

……. sayılı kararıyla onaylanmıştır.

Doç. Dr. Rıdvan ÜNAL

Enstitü Müdürü

ÖZET

Yüksek Lisans Tezi

BULANIK MANTIK VE YAPAY SİNİR AĞLARI İÇİN EĞİTİM YAZILIMI GELİŞTİRİLMESİ

Utku KÖSE

Afyon Kocatepe Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Anabilim Dalı

Danışman: Yrd. Doç. Dr. Ömer DEPERLİOĞLU

Günümüzde, fiziksel dünyayla özdeşleşmiş problemlerin, insan düşünce yapısıyla çözülmesini amaçlayan, birçok farklı yapay zekâ tekniği bulunmaktadır. Genetik algoritmalar, bulanık mantık ve yapay sinir ağları, yaygın kullanım alanlarına sahip, popüler yapay zekâ tekniklerinden bazılarıdır. Bu teknikler, bilgisayar teknolojisinin de yardımıyla, farklı alanlardaki problemlerin çözümünde kolayca kullanılabilmektedir.

Bu tez çalışmasında, bulanık mantık ve yapay sinir ağları tekniklerinin eğitiminde ve bu tekniklerle ilgili araştırma çalışmalarında kullanılabilen, etkili bir uygulama yazılımı geliştirilmiştir. Çalışma sayesinde, ilgili tekniklerin öğretiminde kullanılabilen, örnek çalışmaların yürütülebildiği, güçlü bir "eğitim yazılımı" sunulması amaçlanmaktadır. Yazılım yapısı, C# programlama dili aracılığıyla, nesneye yönelik programlama tekniklerinin avantajları bir araya getirilerek oluşturulmuştur. Ayrıca bu yazılım, ilgili alanda gerçekleştirilmiş olan, Türkçe arayüzlü, ender çalışmalardan birisidir.

2010, 106 sayfa

Anahtar Kelimeler: Bulanık Mantık, Bulanık Kontrol Sistemleri, Yapay Sinir Ağları, Yapay Zekâ, Eğitim Yazılımı.

ABSTRACT

M. Sc. Thesis

DEVELOPING EDUCATION SOFTWARE
FOR FUZZY LOGIC AND ARTIFICIAL NEURAL NETWORKS

Utku KÖSE

Afyon Kocatepe University
Graduate School of Natural and Applied Sciences
Department of Computer

Supervisor: Asst. Prof. Dr. Ömer DEPERLİOĞLU

Nowadays, there are many different artificial intelligence techniques that aim to solve real-world problems with the human reasoning structure. Genetic algorithms, fuzzy logic and artificial neural networks are some popular artificial intelligence techniques, which have wide usage ranges. With the support of the computer technology, these techniques can be used easily to solve problems in different fields.

In this thesis study, an effective application, which can be used in education of fuzzy logic and artificial neural networks techniques and related research studies, has been developed. With the study, it is aimed to provide a strong "education software", which can be used for teaching the related techniques and performing sample works about them. The software structure has been developed by combining advantages of object oriented programming techniques via C# programming language. Additionally, this software is one of the uncommon works, which provide Turkish interface.

2010, 106 pages

Keywords: Fuzzy Logic, Fuzzy Control Systems, Artificial Neural Networks, Artificial Intelligence, Education Software.

TEŞEKKÜR

Bu tez çalışmasının gerçekleştirilmesinde, gerekli bütün yardım, tavsiye ve yönlendirmeleri yapan, karşılaştığım problemlerin çözümünde deneyimlerinden yararlandığım, sayın hocam Yrd. Doç. Dr. Ömer DEPERLİOĞLU'na katkılarından dolayı teşekkür ederim. Ayrıca, gösterdikleri özveri ve desteklerinden dolayı aileme de teşekkürü bir borç bilirim.

Utku KÖSE

İÇİNDEKİLER

	Sayfa No
ÖZET	**i**
ABSTRACT	**ii**
TEŞEKKÜR	**iii**
İÇİNDEKİLER	**iv**
SİMGELER VE KISALTMALAR DİZİNİ	**vii**
ŞEKİLLER DİZİNİ	**x**
RESİMLER DİZİNİ	**xi**
ÇİZELGELER DİZİNİ	**xiii**
1. GİRİŞ	**1**
2. BULANIK MANTIK VE YAPAY SİNİR AĞLARINDA UYGULAMA YAZILIMLARI VE EĞİTİMSEL ÖZELLİKLER	**5**
2.1 Bulanık Mantık	5
2.1.1 Bulanık Küme Teorisi	6
2.1.2 Bulanık Kontrol Sistemleri	11
2.1.3 Bulanık Kontrol Kuralları	12
2.1.4 Bulanık Mantıkta Karar Verme Mantığı	13
2.1.5 Bulanık Çıkarım Yöntemleri	15
2.1.6 Durulaştırma Yöntemleri	17
2.2 Yapay Sinir Ağları	19
2.2.1 Yapay Sinir Ağlarının Temel Yapısı	20
2.2.2 Yapay Sinir Ağlarında Mimariler	22
2.2.3 Yapay Sinir Ağlarında Öğrenme ve Genel Öğrenme Kuralları	23
2.2.4 İleri Beslemeli Çok Katmanlı YSA için Geri Yayılım Algoritması ve Öğrenme	25
2.3 Bulanık Mantıkta Uygulama Yazılımları ve Eğitimsel Özellikler	28
2.3.1 Fuzzy TECH	29
2.3.2 Dot Fuzzy	30
2.3.3 MATLAB Fuzzy Logic Toolbox	31

İÇİNDEKİLER

	Sayfa No
2.3.4 NEFCLASS	32
2.3.5 jFuzzy Logic	33
2.3.6 Kütüphane Niteliğindeki Yazılımlar	34
2.4 Yapay Sinir Ağlarında Uygulama Yazılımları ve Eğitimsel Özellikler	35
2.4.1 MATLAB – Neural Networks Toolbox	35
2.4.2 Neuro Solutions	36
2.4.3 Statistica – Neural Networks	37
2.4.4 Mathematica – Neural Networks	38
2.4.5 Netmaker	39
2.4.6 JOONE	40
2.4.7 Fuzzy COPE	41
2.4.8 Kütüphane Niteliğindeki Yazılımlar	41
3. MATERYAL VE METOT	**43**
3.1 Programlama ve Tasarım Faktörleri	43
3.1.1 C# Programlama Dili	43
3.1.2 Diğer Programlama ve Tasarım Faktörleri	45
3.2 Nesneye Yönelik Programlama Yaklaşımı	46
4. BULANIK MANTIK VE YAPAY SİNİR AĞLARI İÇİN EĞİTİM YAZILIMI GELİŞTİRİLMESİ	**48**
4.1 Kaynak Kod Özellikleri	48
4.2 Yazılımın Temel Kullanım Özellikleri ve İşlevleri	52
4.3 Yazılım Arayüzleri	53
4.3.1 Giriş Arayüzü	54
4.3.2 Bulanık Mantık Arayüzü	54
4.3.3 Yapay Sinir Ağları Arayüzü	55
4.4 Bulanık Mantık Uygulama ve Eğitim Yazılımı ile Çalışmak	56
4.4.1 Bulanık Kontrol Sistemlerinin Tasarlanması ve Düzenlenmesi	56

İÇİNDEKİLER

	Sayfa No
4.4.2 Bulanık Kontrol Sistemleri Üzerinde Yapılabilecek Çalışmalar	67
4.4.3 Yazılım Bünyesindeki Diğer Özellik ve İşlevler	72
4.4.4 Bulanık Mantık Uygulama ve Eğitim Yazılımı ile Örnek Çalışmalar	74
4.5 Yapay Sinir Ağları Uygulama ve Eğitim Yazılımı ile Çalışmak	77
4.5.1 Yapay Sinir Ağlarının Tasarlanması ve Düzenlenmesi	78
4.5.2 Yapay Sinir Ağlarının Eğitilmesi ve Test Edilmesi	81
4.5.3 Yazılım Bünyesindeki Diğer Özellik ve İşlevler	89
4.5.4 Yapay Sinir Ağları Uygulama ve Eğitim Yazılımı ile Örnek Çalışmalar	91
5. SONUÇ VE TARTIŞMA	**97**
6. KAYNAKLAR	**100**
6.1 İnternet Kaynakları	105
ÖZGEÇMİŞ	**106**

SİMGELER VE KISALTMALAR DİZİNİ

1. Simgeler

A, B,…Z	Kümeler
a, b, c, d	Bulanık mantıkta üyelik fonksiyonları için parametreler
u	Bir X kümesinin elemanı (Bir X kümesi kapsamında değer)
μ	Bulanık mantıkta üyelik fonksiyonu
U, V, W	Bir X kümesinin tanımlı olduğu evren
$\cup$	Birleşim
$\cap$	Kesişim
x, y, z	Bulanık kurallarda dilsel değişkenler
R	Bulanık kural
$\rightarrow$	İse
Λ	Ve
V	Veya
α	Bulanık kont. sistemlerinde ateşleme kuv. – Yapay sinir ağlarında momentum
f	Fonksiyon
z	Bulanık kontrol sistemlerinde kontrol hareketi
y	Bulanık kontrol sistemlerinde bulanık kontrol hareketi
$\sum$	Toplam sembolü
w	Bulanık mantıkta kural etki faktörü – Yapay sinir ağlarında ağırlık
x	Yapay sinir ağlarında giriş değeri
C	Yapay sinir ağlarında kural sonuç değeri
θ	Yapay sinir ağlarında kesin eşik değeri
e	Yapay sinir ağlarında hata
o	Yapay sinir ağlarında çıkış
L	Yapay sinir ağlarında katman
Δ	Delta (değişim miktarı)
n	Yapay sinir ağlarında iterasyon
η	Yapay sinir ağlarında öğrenme katsayısı

SİMGELER VE KISALTMALAR DİZİNİ

2. Kısaltmalar

NORM	Normalization (Normalizasyon)
CON	Concentration (Konsantrasyon)
DIL	Dilatation (Genişletme)
INT	Intensification (Şiddetlendirme)
MISO	Multi Input Single Output (Çoklu Giriş Tek Çıkış)
CoG	Center of Gravity (Ağırlık Merkezi)
CoA	Center of Area (Alan Merkezi)
MoM	Mean of Maximum (Maksimumların Ortalaması)
M-P	McCulloch-Pitts
MLP	Multi Layer Perceptron (Çok Katmanlı Perceptron)
ADALINE	Adaptive Linear Neuron ya da Adaptive Linear Element (Uyarlamalı Doğrusal Sinir Hücresi ya da Uyarlamalı Doğrusal Unsur)
FIS	Fuzzy Inference System (Bulanık Çıkarım Sistemi)
MATLAB	Matrix Laboratory (*Özel İsim* – Matris Laboratuarı)
FFLL	Free Fuzzy Logic Library (*Özel İsim* – Ücretsiz Bulanık Mantık Kütüphanesi)
CERN	(*Fransızca*) Conseil Europeen pour la Recherche Nucleaire Avrupa Nükleer Araştırma Merkezi

SİMGELER VE KISALTMALAR DİZİNİ

JOONE	Java Object Oriented Neural Engine (*Özel İsim* – Java Nesneye Yönelik Sinirsel Motor)
FANN	Fast Artificial Neural Network (*Özel İsim* – Hızlı Yapay Sinir Ağı)
ANSI	American National Standards Institute (Amerikan Ulusal Standartlar Enstitüsü)
ECMA	European Computer Manufacturers Association (Avrupa Bilgisayar Üreticileri Birliği)
IDE	Integrated Development Environment (Tümleşik Geliştirme Ortamı)
XML	Extensible Markup Language (Genişletilebilir İşaretleme Dili)
W3C	World Wide Web Consortium (Dünya Çapında Ağ Birliği)
HTML	Hyper Text Markup Language (Zengin Metin İşaretleme Dili)
XOR	Exclusive or (Özel veya)

ŞEKİLLER DİZİNİ

	Sayfa No
Şekil 2.1 Üçgen üyelik fonksiyonu ve ilgili parametreler.	6
Şekil 2.2 Yamuk üyelik fonksiyonu ve ilgili parametreler.	7
Şekil 2.3 Çan eğrisi üyelik fonksiyonu ve ilgili parametreler.	8
Şekil 2.4 Singleton üyelik fonksiyonu ve ilgili parametreler.	8
Şekil 2.5 Yedi adet üyelik fonksiyonundan oluşan, örnek bir bulanık küme.	9
Şekil 2.6 Bir bulanık kontrol sisteminin temel yapısı.	11
Şekil 2.7 Mamdani tipi bulanık çıkarım (Deperlioğlu 2001).	16
Şekil 2.8 Takagi-Sugeno tipi bulanık çıkarım (Deperlioğlu 2001).	17
Şekil 2.9 McCulloch ve Pitts sinirinin temel yapısı.	20
Şekil 2.10 İleri beslemeli çok katmanlı bir yapay sinir ağı yapısı.	22
Şekil 2.11 İleri beslemeli çok katmanlı örnek bir yapay sinir ağı yapısı (Öz *vd.* 2002).	26
Şekil 4.1 Geliştirilen yazılımın programlama yapısı.	49

RESİMLER DİZİNİ

Sayfa No

Resim 4.1 Geliştirilen yazılımın giriş arayüzü. 54
Resim 4.2 Bulanık mantık arayüzü. 55
Resim 4.3 Yapay sinir ağları arayüzü. 56
Resim 4.4 Bulanık Mantık Sistem Özeti panelinde, Takagi-Sugeno model şeması. 57
Resim 4.5 Bulanık Mantık Sistem Tanımı panelinden bir ekran görüntüsü. 58
Resim 4.6 Sistem Değişkenleri penceresinden bir ekran görüntüsü. 59
Resim 4.7 Yeni değişken ekleme penceresi. 60
Resim 4.8 Yeni üyelik fonksiyonu ekleme penceresi. 62
Resim 4.9 Seçili değişken için üyelik fonksiyonlarının düzenlenebildiği panel. 63
Resim 4.10 Tanımlanabilecek üyelik fonksiyonlarının görsel karşılıkları. 64
Resim 4.11 Farklı tanım aralığı ve özelliklerinde, çıkış değişkeni fonksiyonları. 65
Resim 4.12 Kural Düzenleyici penceresinden bir ekran görüntüsü. 66
Resim 4.13 Kural Uygulayıcı penceresinden bir ekran görüntüsü. 67
Resim 4.14 Aktif Sistem – Çıkış Listesi penceresinden bir ekran görüntüsü. 68
Resim 4.15 Aktif Sistem Bilgileri penceresi. 69
Resim 4.16 Matris Giriş penceresinden bir ekran görüntüsü. 71
Resim 4.17 Bir kontrol süreci sonrası elde edilen grafiklerden örnek ekran görüntüleri. 72
Resim 4.18 Yazılım Ayarları penceresi. 73
Resim 4.19 Bulanık Mantık E-Bilgi – HTML sayfasından bir ekran görüntüsü. 74
Resim 4.20 Uygulama – kontrol süreci sonucunda elde edilen grafikler. 77
Resim 4.21 Yapay sinir ağları arayüzünde "YSA Özellikleri" paneli. 78
Resim 4.22 Yapay sinir ağları arayüzünde katman ekleme (düzenleme) penceresi. 80
Resim 4.23 YSA Sistem Tanımı paneli altında örnek bir yapay sinir ağı sistemi. 81
Resim 4.24 Veri Dosyası Yükleme / Düzenleme penceresi. 82
Resim 4.25 YSA İşlem Adımları panelinden bir ekran görüntüsü. 84
Resim 4.26 YSA İşlem Süreci paneli ve YSA – Nöron Ağırlık Bilgileri penceresi. 85
Resim 4.27 Örnek hata ve çıkış değişim grafikleri. 86
Resim 4.28 Yazılım bünyesinde sunulmuş test penceresinden bir ekran görüntüsü. 87

RESİMLER DİZİNİ

	Sayfa No
Resim 4.29 Performans Ölçümü penceresi.	88
Resim 4.30 Örnek bir performans ölçüm grafiği.	89
Resim 4.31 Yazılım Ayarları penceresi ve ilgili sekmeler.	90
Resim 4.32 YSA E-Bilgi – HTML sayfasından bir ekran görüntüsü.	91
Resim 4.33 XOR uygulaması için “giriş” ve “beklenen çıkış” veri dosyaları.	92
Resim 4.34 XOR uygulamasında elde edilen hata ve çıkış değişim grafikleri.	93
Resim 4.35 Uygulama sonucu elde edilen hata ve çıkış değişim grafikleri.	95
Resim 4.36 Uygulama kapsamında elde edilen performans ölçüm grafiği.	96

ÇİZELGELER DİZİNİ

	Sayfa No
Çizelge 2.1 Bulanık kümede bazı işlemler (Elmas 2003a).	10
Çizelge 2.2 Bulanık içerme kuralları (Elmas 2003a).	14
Çizelge 4.1 Bulanık mantık için oluşturulan, bazı kaynak kod dosyaları.	51
Çizelge 4.2 Yapay sinir ağları için oluşturulan, bazı kaynak kod dosyaları.	51
Çizelge 4.3 Uygulama için tanımlanan dilsel kurallar.	75
Çizelge 4.4 Uygulama kapsamında, farklı yazılımlar ile elde edilen sonuçlar.	76
Çizelge 4.5 XOR mantık kapısının doğruluk tablosu.	92
Çizelge 4.6 XOR uygulaması kapsamında elde edilen test sonuçları.	94

1. GİRİŞ

Yapay zekâ alanı, insan düşünce ve davranış şeklini taklit etmeyi amaç edinen ve bu yolla fiziksel dünyayla özdeşleşmiş problemlere çözüm üreten farklı yaklaşım ve teknikleri bir araya geldiği, günümüzün popüler Bilgisayar Bilimleri alanlarından birisidir. Daha kısa anlamda yapay zekâ alanına "zeki davranışların otomasyonu ile ilgili Bilgisayar Bilimleri dalı" demek de mümkündür (Russell and Norvig 1995, Nilsson 1998, Luger 2002, Uğur ve Kınacı 2005). Yapay zekâ içerisinde yer alan yaklaşım ve teknikler, özellikleri ve uygulanabilirlik düzeyleri itibariyle, birçok farklı alanda kullanılabilmekte ve yeni alanlara da kolaylıkla adapte edilebilmektedir. Günümüzde, yapay zekâ tekniklerine dayalı uygulamalara elektronik, biyoloji, fizik, tıp, makine endüstrisi ve askeriye gibi alanlar ile birlikte birçok mühendislik biliminde rastlamak mümkündür (Whitby 2003, Nabiyev 2005, Çetin *vd.* 2006, Castillo *et al.* 2006, Uğur ve Kınacı 2006). Yapay zekâ kapsamında incelenebilecek birçok farklı teknik var olmasına rağmen, ilgili alanlarda yalnızca birkaç tanesi yaygın bir kullanım alanı bulabilmektedir. Söz konusu bu alanlarda gerçekleştirilmiş çalışmalar değerlendirildiği zaman, makine öğrenmesi, genetik algoritmalar, bulanık mantık ve yapay sinir ağları gibi tekniklerin daha popüler bir konumda olduğu ortaya çıkmaktadır. Bu teknikler, bilgisayar teknolojisinin de yardımıyla, farklı alanlardaki problemlerin çözülmesi amacıyla kolay ve etkili bir şekilde kullanılabilmekte, karşılaşılan farklı türdeki problemlere de kısa sürede uyarlanabilmektedir. Bu bağlamda, özellikle bulanık mantık ve yapay sinir ağları teknikleri için kullanılan, büyük ve küçük ölçekte birçok uygulama yazılımı, geniş bir kullanıcı kitlesi tarafından kullanılmaktadır. Bu yazılımlar, bulanık mantık denetleyicileri ve yapay sinir ağları sistemleri üzerinde çalışan birçok araştırmacı ve bilim adamına, teknik anlamda kolaylıklar sunmaktadır.

Diğer yapay zekâ tekniklerinde de olduğu gibi, bulanık mantık ve yapay sinir ağları tekniklerinin önemli özelliklerinden birisi, yoğun miktarda matematiksel – mantıksal teorem ve işlevleri beraberlerinde getirmeleridir. Bu durum, tekniklerin farklı problemlere uygulanabilmesi ve sürekli geliştirilebilir bir yapıda olmaları açısından önemli avantajlar sunmasına rağmen, yine ilgili tekniklerin öğrenilmesi aşamasında çeşitli dezavantajları da gün yüzüne çıkarmaktadır. Bulanık mantık ve yapay sinir ağları

tekniklerinin temel prensiplerini ve uygulama şekillerini öğrenme aşamasında birçok kişi, teknik ve matematiksel bilgiler üzerine yoğunlaşmak durumunda kalmaktadır. Kuşkusuz ki, ilgili tekniklerin öğrenilmesi aşamasında, bu bilgilerin büyük ölçüde özümsenmesi gereklidir. Ancak öğrenme aşamasında izlenen yolun yeterliliği tartışılabilir niteliktedir. Bulanık mantık ve yapay sinir ağları gibi teknik içerikli, uygulamaya dönük konuların, sadece teorik yaklaşımlarla öğrenilmeye çalışılması ne denli yetersizse, yine bu konuların, önemli ölçüde teorik temelden yoksun bir konumdayken, ileri düzeyde bilgilerden yararlanarak öğrenilmeye çalışılması, büyük ihtimalle teknikleri anlaşılamaz bir düzeye taşıyacaktır. Konu bu açıdan ele alındığı zaman, piyasada bulunan bulanık mantık ve yapay sinir ağları uygulama yazılımlarının da eğitimsel yönünün değerlendirilmesi gerekmektedir. Daha önce de belirtildiği üzere bu yazılımlar, bulanık mantık denetleyicileri ve yapay sinir ağları sistemleri üzerinde çalışan kişilere, teknik anlamda birçok kolaylık sunmaktadır. Ancak bu duruma ek olarak, ilgili yazılımların eğitimsel yönden de çeşitli avantajları vardır. En basit anlamda, farklı bulanık mantık denetleyici sistemlerinin, bilgisayar ortamında hızlı ve duyarlı bir şekilde gerçekleştirilebilmesi, sistemlerle ilgili parametre ve özelliklerin detaylı bir şekilde düzenlenebilir – değiştirilebilir olması, ilgili sistemlerinin çalışma şeklinin öğrenilmesi açısından önemlidir. Aslına bakılırsa, söz konusu yazılımlar üzerinde sürekli uygulamalar geliştirilmesi, bulanık mantık ve yapay sinir ağlarının, dolaylı yönden de olsa öğrenilmesine imkân sağlamaktadır. Bütün bu etmenler göz önüne alındığı takdirde, bulanık mantık ve yapay sinir ağlarına dayalı olarak geliştirilmiş uygulama yazılımlarının, eğitim yazılımı niteliğinde değerlendirilmesi mümkündür.

Günümüzde, bulanık mantık denetleyiciler ve yapay sinir ağları sistemleri üzerinde çalışmalar gerçekleştirilmesine imkân tanıyan, farklı uygulama yazılımları bulunmaktadır. Bu yazılımların bir kısmı, ilgili yapay zekâ teknikleriyle bağlantılı, belirli uygulama alanlarını hedef alırken, bir kısmı da büyük ölçekte, farklı nitelikte uygulamaların gerçekleştirilmesine de imkân tanımaktadır. Bu noktada, yaygın kullanım alanına sahip olanlar, daha çok farklı nitelikte uygulama alanlarında da kullanılabilen yazılımlar olmaktadır. Söz konusu uygulama yazılımları, eğitimsel açıdan değerlendirilebildiği için, günümüzde popüler olan bulanık mantık ve yapay sinir ağları

uygulama yazılımlarını da eğitimsel yapılarda görmek mümkündür. Ancak bu yazılımların, eğitim yazılımı niteliğinde birtakım özellik ve işlevler taşımasına rağmen, belirli bir miktarda karmaşıklık ve teknik ön bilgi gerektiren bazı özellik ve işlevlere sahip oldukları da gözlemlenmektedir. Uygulama yazılımları bu açıdan incelendiği takdirde, eğitimsel kaygılar göz önünde bulundurularak tasarlanmış ve kullanım özellikleri ile işlevleri de bu kapsamda daha basit düzeyde organize edilmiş bir uygulama yazılımına ihtiyaç doğmaktadır. Bu amaçla tasarlanacak ve geliştirilecek bir yazılım, bulanık mantık ve yapay sinir ağları ile ilgili uygulama temellerini, ortalama bir düzeyde, etkili bir şekilde sunacak, ayrıca etkileşimli ve görsel faktörler yardımıyla, daha etkili ve verimli bir çalışma ortamı ortaya çıkarmalıdır. Ayrıca, bu kıstaslar göz önünde bulundurularak geliştirilecek bir uygulama yazılımı, hem temel düzeydeki kullanıcılara hem de deneyimli kullanıcılara hitap edecek bir nitelikte olmalıdır.

Bu tez çalışmasında, bulanık mantık ve yapay sinir ağları tekniklerinin eğitiminde ve bu tekniklerle ilgili araştırma çalışmalarında kullanılabilen, etkili bir uygulama yazılımı geliştirilmiştir. Söz konusu yazılım, sunduğu farklı özellik ve işlevler sayesinde, bulanık mantık ve yapay sinir ağları kavramlarının ve uygulamalarının öğretiminde olduğu kadar, yüksek nitelikte çalışmaların yürütülmesinde de kullanılabilen, Türkçe arayüzlü, güçlü bir yazılım ortamı sunmaktadır. Gerçekleştirilen çalışma aracılığıyla, piyasada hazır durumda olan daha karmaşık yapıdaki yazılımların önemli özellikleri basit yapılarda sunulmuş, bu yolla, ilgili tekniklerle alakalı prensiplerin ve uygulama yapılarının daha kolay öğretilmesi amaçlanmıştır. Ayrıca yazılım, bünyesinde sunduğu hazır işlevler ve kontroller aracılığıyla, daha hızlı ve etkili bir kullanım tecrübesi de vaat etmektedir. Bazı işlev ve kontroller sayesinde, herhangi bir önbilgi sahibi olmaksızın, bulanık mantık ve yapay sinir ağları ile ilgili temel uygulamalar kolay ve hızlı bir şekilde tasarlanmakta ve kullanıma açılabilmektedir. Sunulan bazı işlev ve kontroller ise, ilgili yazılım ortamında, daha teknik düzeyde çalışmaların tasarlanmasına ve gerçekleştirilmesine olanak tanımaktadır. Bu bağlamda geliştirilen uygulama yazılımı, temel düzeyden ileri düzeye olmak üzere, geniş bir kullanıcı kitlesini kapsam içerisine almaktadır. Bu noktada önemli olan bir başka faktör de, geliştirilen yazılımın, teknik düzeydeki İngilizce yazılımları kullanmakta güçlük çeken kullanıcıların, Türkçe hazırlanmış bir ortamda daha kolay çalışmasına imkân sağlamasıdır.

Çalışmanın geriye kalan bölümleri şu şekilde düzenlenmiştir: 2. Bölüm'de, bulanık mantık ve yapay sinir ağları tekniklerinin temel prensipleri ve kullanım şekilleri açıklanmış, ilgili teknikler için geliştirilmiş olan benzer uygulama – eğitim yazılımları incelenerek, bu yazılımların, çalışma kapsamında değerlendirilmesi gerekli görülen, çeşitli özelliklerine değinilmiştir. 3. Bölüm'de, çalışma kapsamında geliştirilmiş olan uygulama yazılımının programlanması ve tasarlanması aşamasında izlenen metodlar ve kullanılan temel materyaller açıklanmıştır. 4. Bölüm'de, geliştirilen uygulama yazılımının kullanım özellikleri ve işlevlerine detaylı bir şekilde değinilmiştir. Son bölüm olan 5. Bölüm'de ise, ilgili çalışmanın önemi, işlevi ve bu bağlamda elde edilmiş olan sonuçlardan bahsedilmiştir.

2. BULANIK MANTIK VE YAPAY SİNİR AĞLARINDA UYGULAMA YAZILIMLARI VE EĞİTİMSEL ÖZELLİKLER

Çalışma kapsamında geliştirilen uygulama – eğitim yazılımının avantajlarını ve sunduğu etkili özellik ya da işlevleri değerlendirmek için, piyasada bulunan ya da çeşitli araştırma çalışmaları kapsamında gerçekleştirilmiş olan, benzer nitelikteki yazılımların temel özelliklerini ve çeşitli avantaj – dezavantajlarını incelemekte fayda vardır. Ancak, söz konusu incelemeden önce, bulanık mantık ve yapay sinir ağları tekniklerinin temel prensiplerini ve kullanım şekillerini kısaca açıklamak, ilgili yazılımları ve çalışma kapsamında ortaya konulan uygulama – eğitim yazılımı anlamak açısından yerinde olacaktır.

2.1 Bulanık Mantık

Bulanık mantığın temel mekanizması, bir insanın herhangi bir sistemi denetlemedeki düşünce ve sezgilerine bağlı olan çeşitli davranışlarının, dilsel niteleyiciler kullanarak, esnek yapıda denetim mekanizması geliştirilmesinde kullanılmasına dayanmaktadır (Deperlioğlu 2001, Elmas 2003a). Bu bağlamda, insan normal olarak bir sistemi denetlerken, ilgili o sistemin matematiksel modeli hakkında bilgi sahibi olmamaktadır. Ancak bu noktada, geliştirilen sistemi aktif haldeki durumundan, istenilen bir duruma götürmek için, düşünce, sezgi ve daha önemlisi deneyimlerine bağlı olarak ortaya çıkarılan bir kontrol yaklaşımı uygulayabilmektedir. Örneğin, bir odada, sıcaklığı kontrol eden bir klimanın motoru, otomatik yaklaşımlarla değil de, bir kişi tarafından denetlendiği zaman, eğer oda sıcaklığı biraz arttıysa klima motoru hızı ilgili kişi tarafından "biraz" azaltacaktır. Eğer oda sıcaklığı çok düştüyse, bu durumda kişi klima motorunun hızını "çok" artıracaktır. İşte bu örnekte kullanılan "biraz" ve "çok" terimleri birer dilsel terim olmakta ve bulanık mantık tekniği kapsamında bulanık değişkenler olarak adlandırılmaktadır. Bu tarz kontrol problemleri için geliştirilecek bir bulanık kontrol sisteminde, dilsel olarak tanımlanmış kontrol yaklaşımları, uzmana dayalı "otomatik kontrol" mekanizmasına çevrilmektedir. Bulanık mantık tekniğinin temel prensiplerini ve işlevlerini daha iyi anlamak adına, "bulanık küme teorisi" olarak adlandırılan konuyu daha detaylı incelemek, bu noktada yararlı olacaktır.

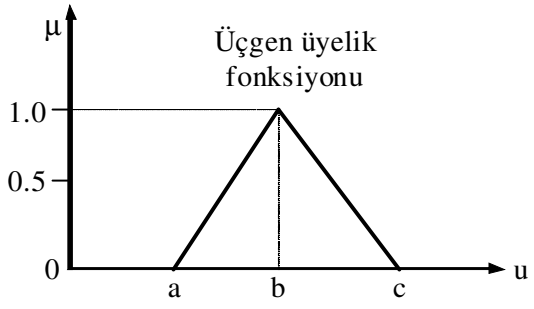
2.1.1 Bulanık Küme Teorisi

Klasik küme teorisindeki yaklaşıma göre bir eleman, seçilmiş bir kümenin ya elemanı olmakta ya da olmamaktadır. Yani bu noktada, ilgili eleman için kısmi bir üyelikten bahsetmek mümkün değildir. Klasik küme teorisinde, ele alınan her eleman, bir kümeyle alakalı üyelik durumları için 0 ("üyesi değil" veya "elemanı değil") ya da 1 ("üyesi" veya "elemanı") değerlerinden birisini alabilmektedir.

Bulanık küme teorisinde, klasik kümeden farklı olarak, kapsam dâhilindeki elemanlar çeşitli üyelik dereceleri ile anılmaktadır. Bu durumda her elemanın üyelik derecesi [0, 1] (kapalı) aralığında sonsuz sayıda değişmektedir. Böylece, klasik kümelerdeki kısa-uzun, yavaş-hızlı, soğuk-sıcak, karanlık-aydınlık gibi ikili durumlar, bulanık küme teorisi kapsamında biraz kısa, biraz sıcak, biraz aydınlık gibi, daha esnek niteleyicilerle, gerçek dünyadaki durumlara benzetilmektedir.

Bulanık küme teorisi kapsamında birçok farklı üyelik fonksiyonu olmasına rağmen, günümüzde genellikle "üçgen", "yamuk", "çan eğrisi" ve "singleton" tipi üyelik fonksiyonları kullanılmaktadır.

Üçgen üyelik fonksiyonunda temel anlamda üç farklı parametre kullanılmaktadır. Şekil 2.1'de tipik bir üçgen üyelik fonksiyonu ve ilgili parametreler gösterilmektedir.

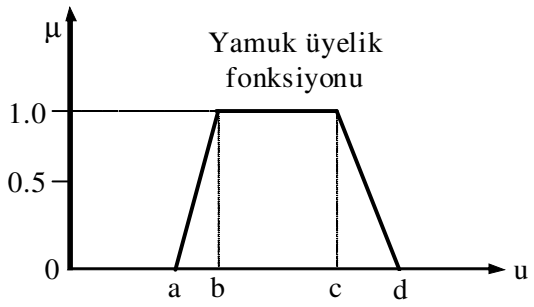


Şekil 2.1 Üçgen üyelik fonksiyonu ve ilgili parametreler.

İlgili parametrelerin durumlarına göre, üçgen tipi üyelik fonksiyonunda elde edilecek sonuçlar, Eşitlik 2.1'de sunulduğu gibi olacaktır.

$$T(u;a,b,c)=\begin{cases}0 & u<a \ \ için\\ (u-a)/(b-a) & a\le u\le b \ \ için\\ (c-u)/(c-b) & b\le u\le c \ \ için\\ 0 & u>c \ \ için\end{cases} \qquad (2.1)$$

Yamuk tipi üyelik fonksiyonunda dört farklı parametre kullanılmaktadır. Şekil 2.2, tipik bir yamuk üyelik fonksiyonu ve ilgili parametreleri göstermektedir.

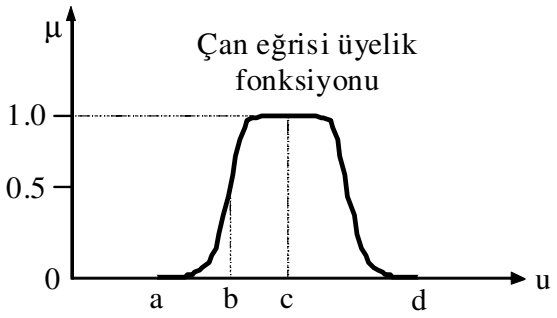


Şekil 2.2 Yamuk üyelik fonksiyonu ve ilgili parametreler.

Parametrelerin durumlarına göre, yamuk tipi üyelik fonksiyonu ile elde edilecek sonuçlar, Eşitlik 2.2'de sunulmuştur.

$$T(u;a,b,c,d)=\begin{cases}0 & u<a \ \ için\\ (u-a)/(b-a) & a\le u\le b \ \ için\\ 1 & b\le u\le c \ \ için\\ (d-u)/(d-c) & c\le u\le d \ \ için\\ 0 & u>d \ \ için\end{cases} \qquad (2.2)$$

Çan eğrisi tipi üyelik fonksiyonunda üç farklı parametre söz konusudur. Tipik bir çan eğrisi üyelik fonksiyonu ve ilgili parametreleri Şekil 2.3'de gösterilmektedir.

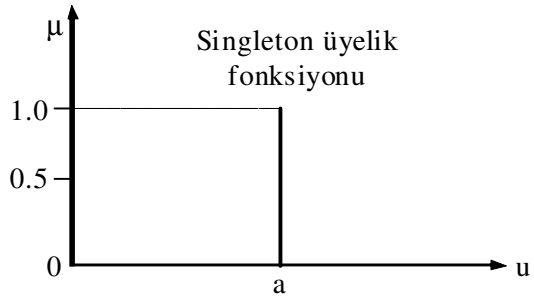


Şekil 2.3 Çan eğrisi üyelik fonksiyonu ve ilgili parametreler.

Verilen parametre durumlarına göre, çan eğrisi tipi üyelik fonksiyonu ile elde edilecek sonuçlar, Eşitlik 2.3'de sunulmuştur.

$$G(u;a,b,c)=\begin{cases} 0 & u<a \ \ için \\ \dfrac{1}{1+\left|\dfrac{u-c}{a}\right|^{2b}} & a\le u\le d \ \ için \\ 0 & u>d \ \ için \end{cases} \qquad (2.3)$$

Singleton üyelik fonksiyonunda tek parametre söz konusudur. Şekil 2.4'de tipik bir singleton üyelik fonksiyonu ve parametresi sunulmaktadır.

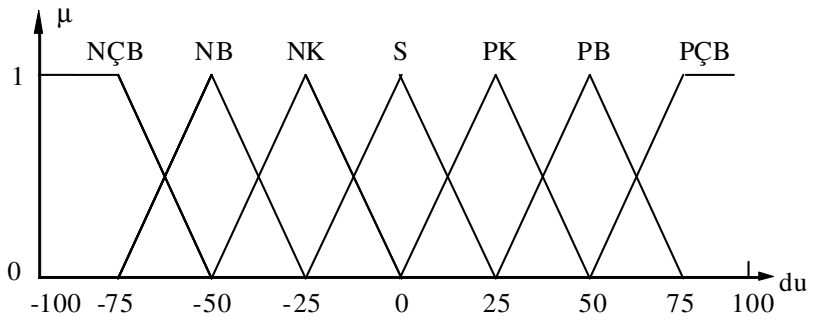


Şekil 2.4 Singleton üyelik fonksiyonu ve ilgili parametreler.

Parametrenin durumuna göre, singleton tipi üyelik fonksiyonunda elde edilecek sonuçlar, Eşitlik 2.4'de sunulduğu gibidir.

$$S(u;a)=\begin{cases} a & u=a \ \text{için} \\ 0 & \text{diger durumlar için} \end{cases} \qquad (2.4)$$

Üyelik fonksiyonları tanımlanırken genellikle "küçük", "orta", "büyük" gibi terimlerle etiketlenmektedir. Üyelik fonksiyonu sayısı olarak da genellikle tek sayı (1, 3, 5…vb.) değerleri kullanılmaktadır. Şekil 2.5'de, pozitif çok büyük (PÇB), Negatif Küçük (NK) gibi toplam 7 adet dilsel etiketli üyelik fonksiyonlarından oluşan, örnek bir bulanık küme gösterilmektedir.



Şekil 2.5 Yedi adet üyelik fonksiyonundan oluşan, örnek bir bulanık küme.

Sonuç olarak bir bulanık küme, o kümenin ilgili elemanları ve elemanların üyelik dereceleri ile oluşturulabilmektedir. Bir A bulanık kümesi, Eşitlik 2.5'te olduğu gibi tanımlanabilmektedir.

$$A=\{u / \mu A(u) \mid u \in U\} \qquad (2.5)$$

Eşitlik 2.5'te ifade edilen u, A kümesinin bir elemanı, μA(u) üyelik fonksiyonu, U ise A kümesinin tanımlandığı evren durumundadır.

Üyelik fonksiyonu, bir kümenin elemanlarının, o kümeye hangi üyelik derecesi ile ait olduğunu gösteren ve [0,1] (kapalı) aralığında değer alabilen bir fonksiyon olduğuna göre, bu durum Eşitlik 2.6'da olduğu gibi ifade edilebilmektedir.

$\mu A(u) : U \rightarrow [0,1]; \mu A(u) : \in [0,1]$ (2.6)

Eşitlik 2.6'da A bir bulanık küme, U, A kümesinin üzerinde tanımlandığı evren ve $\mu A(u)$ ifadesi ise üyelik fonksiyonu durumundadır.

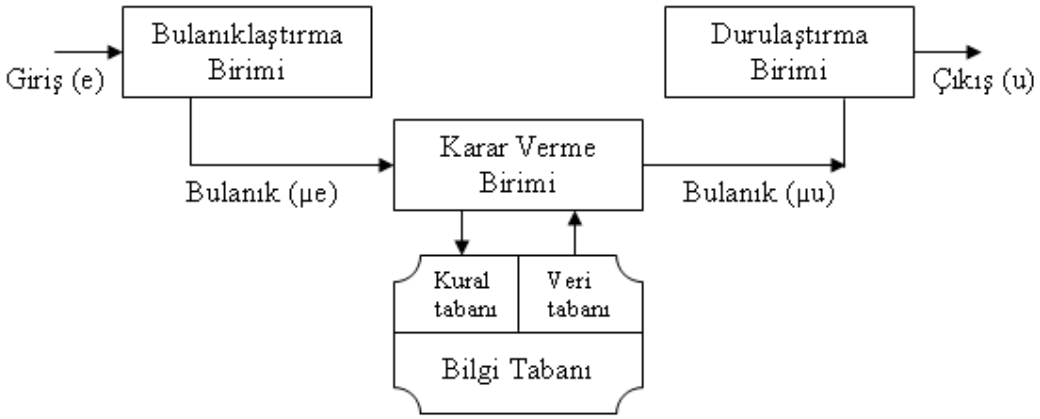
Bilindiği üzere, klasik kümelerde "birleşme", "kesişme", "tümleme" gibi çeşitli işlemler tanımlanmıştır. Söz konusu bu işlemlere karşılık gelen bulanık işlemler de yine bulanık küme teorisi kapsamında ifade edilmektedir. Bulanık küme işlemleri üyelik fonksiyonları kullanılarak sağlanmaktadır. A ve B, U'daki kümeler ve μA ile μB'nin bu kümelerin üyelik fonksiyonları olduğu kabul edildiğinde, tanımlanabilecek bazı bulanık işlemler, Çizelge 2.1'de özet şeklinde verilmiştir.

Çizelge 2.1 Bulanık kümede bazı işlemler (Elmas 2003a).

İşlem	**İşlem İfadesi**	
Eşitlik	$\mu_A(u)=\mu_B(u)$	$u \in U$
Birleşme	$\mu_{A\cup B}(u)= \max\{\mu_A(u), \mu_B(u)\}$	her $u \in U$
Kesişme	$\mu_{A\cap B}(u)= \min\{\mu_A(u), \mu_B(u)\}$	her $u \in U$
Tümleme	$\mu_A(u)=1-\mu_A(u)$	$u \in U$
Normalizasyon	$\mu_{NORM(A)}(u)= \mu_A(u)/\max(\mu_A(u))$	$u \in U$
Konsantrasyon	$\mu_{CON(A)}(u)= (\mu_A(u))^2$	$u \in U$
Genişletme	$\mu_{DIL(A)}(u)= (\mu_A(u))^{0,5}$	$u \in U$
Şiddetlendirme	$\mu_{INT(A)}(u) = \begin{cases} 2(\mu_A(u))^2 \\ 1-2(1-\mu_A(u))^2 \end{cases}$	$0 \le \mu_{A(u)} \le 0.5$ $0.5 \le \mu_{A(u)} \le 1$
Aritmetik ürün	$\mu_{A\bullet B}(u)= \mu_A(u) \bullet \mu_B(u)$	her $u \in U$
Sınırlı toplam	$\mu_{A\oplus B}(u)= \min\{1, \mu_A(u) + \mu_B(u)\}$	her $u \in U$
Sınırlı ürün	$\mu_{A\oplus B}(u)= \max\{o,\mu_A(u) + \mu_B(u)-1\}$	her $u \in U$
Şiddetli ürün	$\mu_{A\otimes B}(u) = \begin{cases} \mu_A(u) \\ \mu_B(u) \\ 0 \end{cases}$	$\mu_B(u)=1$ için $\mu_A(u)=1$ için $\mu_B(u),\mu_B(u)<1$ için

2.1.2 Bulanık Kontrol Sistemleri

Bulanık kontrol sistemlerinin çalışma şekli, insanın karar verme ve çıkarım yapma yeteneğinin taklit edilmesi üzerine kurulu bir yapıda oluşturulmakta ve geliştirilmektedir. Bir bulanık kontrol sistemi, kullanılan tasarım nesnelerine bağlı olarak birçok farklı şekillerde yapılandırılabilmektedir. Ancak ilgili sistemlerin, temel olarak "bulanıklaştırma birimi", "bilgi tabanı", "karar verme birimi" ve "durulaştırma birimi" olmak üzere dört farklı bölümden oluştuğu ifade edilebilmektedir. Bu bağlamda, bir bulanık kontrol sisteminin temel yapısı Şekil 2.6'da gösterilmektedir.



Şekil 2.6 Bir bulanık kontrol sisteminin temel yapısı.

Bir bulanık kontrol sisteminde, denetlenen sistemden ölçülen "e" giriş değişkeni ve sistemi denetim için kullanılan, "u" çıkış değişkeni olmak üzere iki çeşit sistem değişkeni kullanılmaktadır. Bulanıklaştırma birimi, en son ölçülen verinin uygun dilsel değerlere dönüştürülmesini sağlamaktadır. Bilgi tabanı kısmı ise, veritabanı ve kural tabanı olmak üzere bilginin iki ana tipini kapsamaktadır. Bu kapsamda veri tabanı, her bir sistem değişkeninin değerleri gibi, kullanılan bulanık kümelerin üyelik fonksiyonlarını tanımlamakta, kural tabanı ise giriş bulanık değerlerin, çıkış bulanık değerlerine tam olarak eşlenmesini temsil etmektedir. Karar verme birimi, bulanık kontrol sisteminin özü durumunda olan bir yapıdır. Bu birim, arzu edilen kontrol stratejisine erişmek için, yaklaşık çıkarım sağlaması ile insan gibi karar verme

yeteneğine sahip olan bir kısım olarak görülmektedir. Sistemin son kısmı, durulama birimi ise karar verme biriminden gelen bulanık bilgileri, gerçek değerlere dönüştürerek, sistemin tanıyabileceği kontrol hareketi haline gelmesini sağlayan bir yapıdır (Deperlioğlu 2001, Elmas 2003a).

2.1.3 Bulanık Kontrol Kuralları

Bulanık mantık uygulamalarında uzman bilgileri, dilsel ifadelerle, geliştirilen sistemlere tanıtılmaktadır. Bu kurallar genellikle "Eğer sistem şu durumda ise o halde şöyle bir denetim uygula" şeklinde bir cümlesel yapı halinde kurulmaktadır. Bu noktada örnek vermek gerekirse, kurallar:

EĞER durum = x ise O HALDE denetim =y ya da *IF durum = x THEN denetim = y*

şeklinde kurulup sisteme programlama platformu üzerinden iletilebilmektedir.

Kurulan bir bulanık kontrol kuralı "bir neden" ve "bir sonuç" kısmından oluşmaktadır. Kontrol kurallarındaki nedenler ve sonuçlar birden fazla olabilmektedir. Örnek olarak çok girişli, bir çıkışlı (MISO) yapıda hazırlanmış bir bulanık kontrol sisteminin kontrol kuralları, Eşitlik 2.7'de olduğu gibi ifade edilebilmektedir.

$$R^i : Eğer\ x = A_i\ \ ve\ \ y = B_i\ \ ise\ \ O Halde\ \ z = C_i \qquad i = 1, 2, \cdots, n \tag{2.7}$$

Eşitlik 2.7'deki x, y ve z dilsel değişkenlerdir ve sistem durum değişkenleri ile kontrol değişkenini temsil etmektedirler. A_i, B_i ve C_i ise sırasıyla U, V ve W uzaylarında tanımlanmış x, y ve z dilsel değişkenlerinin, dilsel değişken değerleri olarak ifade edilmektedir. Bu durumda $x \in U$, $y \in V$ ve $z \in W$ durumundadır. i=1,2,...n olarak verilen ifade kural indisidir. Kurallar kontrol değişkeni ya da sonuç sistem durum değişkenlerinin fonksiyonu olarak tanımlanmasıyla da yapılabilmektedir. Eşitlik 2.8'de bu duruma dair bir örnek sunulmuştur.

$$R_i : Eğer\ x = A_i\ \ ve\ \ y = B_i\ \ ise\ \ O Halde\ \ z = f_i(x,..., y) \tag{2.8}$$

Eşitlik 2.7 ve Eşitlik 2.8'deki bulanık kontrol kuralları t zamanındaki sistem değişkenlerini hesaplayarak değerlendirmekte ve durum değişkenlerinin fonksiyonu ile denetim hareketine karar vermektedir. "Eğer $x=A_1$ ve $y=B_1$ ise O Halde $z=C_1$" bulanık kontrol kuralı Eşitlik 2.9'da olduğu gibi tanımlanabilmektedir.

$$\mu R_1=[\mu A_1(u) \text{ ve } \mu B_1(v)]\Rightarrow\mu C_1(w) \tag{2.9}$$

A_1 ve B_1 bulanık kümelerdir ve $A_1 \times B_1 \in U \times V$'dir. $R_1= (A_1 \text{ ve } B_1) \rightarrow C_1$ bulanık bağıntı, $R_1 \in U \times V \times W$'dir ve bulanık bağıntı fonksiyonunu göstermektedir. Bu noktada, daha önce Çizelge 2.1'de sunulan birçok biçim tanımlanabilmektedir.

Bulanık kontrol kuralları sayısal değerlerden ziyade, dilsel terimler olarak daha iyi eşitliğe dökülebilmektedir. Bir bulanık mantık sisteminde ve dolayısıyla kurallarda kullanılan değişkenlerin seçiminde, deneyimlerin ve mühendislik bilgisinin oldukça önemli bir rolü bulunmaktadır (Deperlioğlu 2001, Elmas 2003a).

Bulanık kontrol sistemlerinin en temel birimi olan bilgi tabanındaki bulanık kontrol kurallarının oluşturulmasında genel olarak dört farklı yaklaşım uygulanmaktadır. Ancak söz konusu bu yaklaşımlar birbirinden tamamen bağımsız bir yapıda değillerdir. Çoğu durumda, bu yolların sentezi aracılığıyla dilsel kurallar türetilmektedir. İlgili yaklaşımlar, aşağıdaki gibi sıralanabilmektedir (Deperlioğlu 2001, Elmas 2003a):

- Uzman deneyimleri ve mühendislik bilgilerine dayanarak oluşturma,
- Sistemin bulanık modelinin kullanılmasına bağlı olarak oluşturma,
- Operatörün sistem üzerinde yaptığı kontrol işlemlerine dayanarak oluşturma,
- Öğrenme sürecine dayanarak oluşturma.

2.1.4 Bulanık Mantıkta Karar Verme Mantığı

Daha önce de belirtildiği üzere, bir bulanık kontrol sisteminin çalışma şekli, insanın karar verme ve çıkarım yapma yeteneğinin taklit edilmesi üzerine kurulu bir yapıdadır. Bu çalışma şekli, bulanık içerme, bileşke kural çıkarımları ve cümle bağlayıcıları ile

ilgili olmaktadır. Genelde bir bulanık kontrol kuralı bir bulanık ilişkidir ve bulanık içerme ile açıklanabilmektedir. Bu noktada, bulanık içermeyi tanımlamanın birçok yolu bulunmakta ve bir bulanık kontrol sisteminde bu yollarından hangisinin kullanılacağı daha çok "sezgisel olarak" belirlenmektedir (Deperlioğlu 2001, Elmas 2003a).

Literatür incelendiği takdirde, birçok farklı bulanık içerme fonksiyonunun tanımlanmış durumda olduğu görülmektedir. Bir bulanık kontrol kuralı: "eğer x=A ise y=B" bulanık içerme fonksiyonu ile gösterilen A ve B, sırasıyla U, V uzaylarında tanımlanmış bulanık kümelerdir. μA ve μB ise bu kümelerin üyelik fonksiyonları olmaktadır. Buna göre, günümüzde yaygın olarak kullanılan bulanık içermeler Çizelge 2.2'de sunulmaktadır.

Çizelge 2.2 Bulanık içerme kuralları (Elmas 2003a).

Bulanık İçerme Kuralı	**İçerme Formülü**	**Bulanık İçerme**
R_c: min işlemi [Mamdani]	$a \rightarrow b = a \wedge b$	$= \mu_A(u) \wedge \mu_B(v)$
R_p: ürün işlemi [Larsen]	$a \rightarrow b = a \bullet b$	$= \mu_A(u) \bullet \mu_B(v)$
R_{bp}: sınırlı ürün	$a \rightarrow b = 0 \vee (a+b-1)$	$= 0 \vee [\mu_A(u) + \mu_B(v) - 1]$
R_{dp}: şiddetli ürün	$a \rightarrow b = \begin{cases} a, & b = 1 \\ b, & a = 1 \\ 0, & a, b < 1 \end{cases}$	$= \begin{cases} \mu_A(u), & \mu_B(v) = 1 \\ \mu_B(v), & \mu_A(u) = 1 \\ 0, & \mu_A(u), \mu_B(v) < 1 \end{cases}$
R_a: aritmetik ürün [Zadeh]	$a \rightarrow b = 1 \wedge (1-a+b)$	$= 1 \wedge (1- \mu_A(u) + \mu_B(v))$
R_m: max-min kuralı [Zadeh]	$a \rightarrow b = (a \wedge b) \vee (1-a)$	$= (\mu_A(u) \wedge \mu_B(v)) \vee (1- \mu_A(u))$
R_s: standart düzen	$a \rightarrow b = \begin{cases} 1, & a \le b \\ 0, & a > b \end{cases}$	$= \begin{cases} 1, & \mu_A(u) \le \mu_B(v) \\ 0, & \mu_A(u) > \mu_B(v) \end{cases}$
R_b: Boolean	$a \rightarrow b = (1-a) \vee b$	$= (1- \mu_A(u)) \vee \mu_B(v)$
R_g: Gödelian mantık	$a \rightarrow b = \begin{cases} 1, & a \le b \\ b, & a > b \end{cases}$	$= \begin{cases} 1, & \mu_A(u) \le \mu_B(v) \\ \mu_B(v), & \mu_A(u) > \mu_B(v) \end{cases}$
R_Δ: Goguen	$a \rightarrow b = \begin{cases} 1, & a \le b \\ b / a, & a > b \end{cases}$	$= \begin{cases} 1, & \mu_A(u) \le \mu_B(v) \\ \mu_A(u) / \mu_B(v), & \mu_A(u) > \mu_B(v) \end{cases}$

Çizelgede sunulan bulanık içerme türlerinden Mamdani'nin bulanık içermesi, max-min kompozisyonu ile birleştirilmesi suretiyle, bulanık kontrol sistemlerinde yaygın bir şekilde kullanılmaktadır.

2.1.5 Bulanık Çıkarım Yöntemleri

Çevrim içi işlemlerdeki kontrol sistemlerinin durumları, kontrol hareketlerinde önemli bir rol oynamaktadır. Bu noktada, girişler genellikle algılayıcılardan ölçülen değerler olmakta ve söz konusu bu değerler de en yeni değerler olarak değerlendirilmektedir. Bazı durumlarda, alınan giriş verisini bulanık kümelere dönüştürmek uygun olmayabilmekte ancak elverişli bir çare olarak değerlendirilmektedir. Aslına bakılacak olursa, genellikle en yeni değer bulanık singleton olarak davranmaktadır. Buna göre birinci ve ikinci kuralın ateşleme kuvvetleri α_1 ve α_2, Eşitlik 2.10'da verildiği gibi ifade edilebilmektedir.

$$\begin{aligned} \alpha_1 &= \mu_{A_1}(x_0) \wedge \mu_{B_1}(x_0) \\ \alpha_2 &= \mu_{A_2}(x_0) \wedge \mu_{B_2}(x_0) \end{aligned} \tag{2.10}$$

Eşitlikte ifade edilen $\mu_{A_2}(x_0)$ ve $\mu_{B_2}(x_0)$, kullanıcı tarafından verilen veri ve kural tabanında yer alan veri arasındaki kısmi ilişkinin derece değerinin belirtilmesini sağlamaktadır. Söz konusu bu ilişki, bulanık kontrol uygulamalarında kullanılan bulanık çıkarım yöntemlerinde merkezi bir rol oynamaktadır.

Günümüzde genel olarak, Mamdani ve Takagi-Sugeno çıkarım türleri yaygın bir şekilde kullanılmaktadır. Söz konusu bu yaklaşımlara ek olarak, Larsen'in ürün veya cebirsel çarpım işlemi bulanık içerme kuralı fonksiyonu ve Tsukamato'nun tekdüze üyelik fonksiyonları gibi dilsel terimlerle uygulanan yöntemi olmak üzere iki adet daha çıkarım türü de ilgili çalışmalarda kullanılabilmektedir (Lee 1990, Yan *et al.* 1994, Fuller 1999, Wierman 2008).

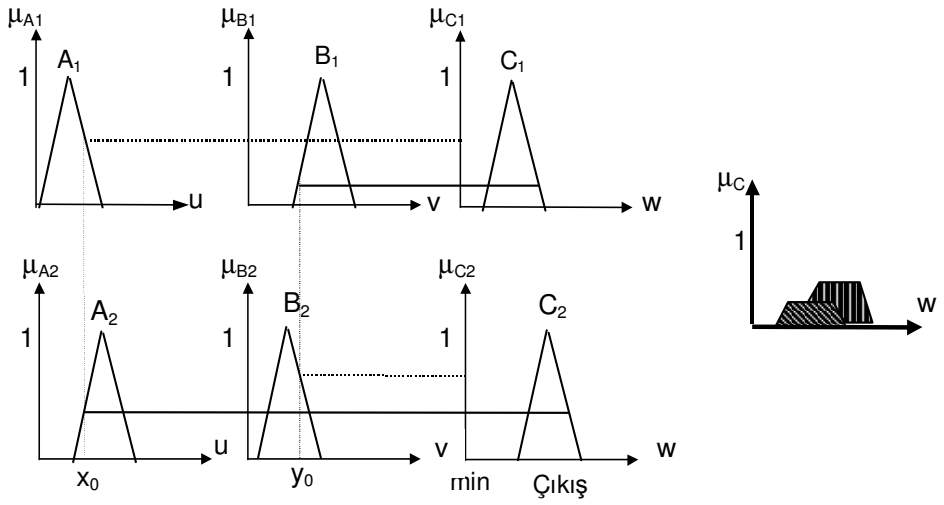
2.1.5.1 Mamdani Tipi Bulanık Çıkarım

Mamdani tipi bulanık çıkarım, Mamdani'nin minimum bulanık içerme kuralı fonksiyonudur. Bu tipte R_c bulanık içerme fonksiyonu olarak seçilmektedir. Bu tip çıkarımda (i.) kuralın toplam kontrol hareketindeki etkisi $\mu_{C_i'}(w) = \alpha_i \wedge \mu_{C_i}(w)$

bağıntısı ile ilgili olmaktadır. İki kuralın olduğu varsayıldığı takdirde, sonuçta elde edilen kontrol hareketi Eşitlik 2.11'deki gibi ifade edilebilmektedir.

$$\mu_C(w) = \mu_{C_1'} \vee \mu_{C_2'} = [\alpha_1 \cdot \mu_{C_1}(w)] \vee [\alpha_2 \cdot \mu_{C_2}(w)] \tag{2.11}$$

Mamdani tipi bulanık çıkarım Şekil 2.7'de olduğu gibi ifade edilebilmektedir.



Şekil 2.7 Mamdani tipi bulanık çıkarım (Deperlioğlu 2001).

2.1.5.2 Takagi-Sugeno Tipi Bulanık Çıkarım

Takagi ve Sugeno tarafından 1983'te ortaya atılan Takagi-Sugeno tipi bulanık çıkarıma göre, bir kuralda elde edilen sonuç, dilsel değişkenlerden oluşan girişlerin bir fonksiyonu olmaktadır. Bu türdeki kurallar, Eşitlik 2.8'de belirtilmiş olan yapı içerisinde olmaktadır. Söz konusu iki kural, bu durum dâhilinde yeniden yazılırsa:

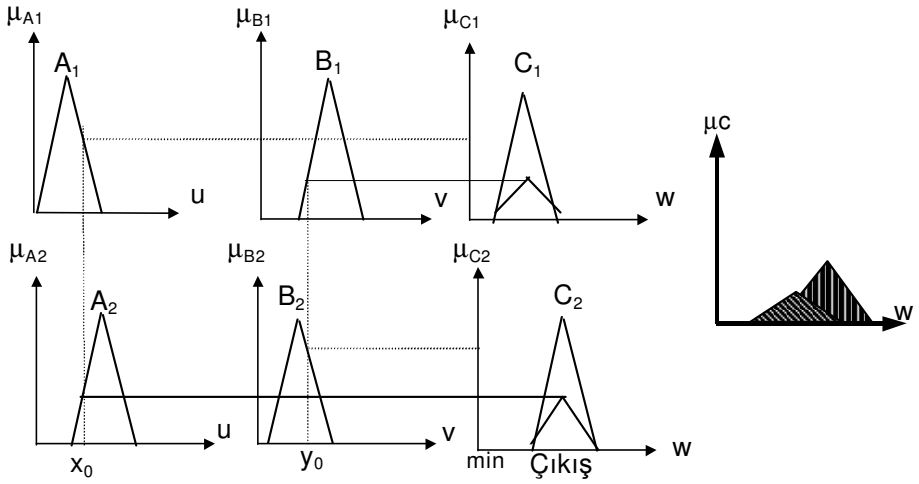
$$R_1 : E\breve{g}er\, x = A_1 \;\; ve \;\; y = B_1 \;\; ise \;\; OHalde \;\; z = f_1(x, y)$$
$$R_2 : E\breve{g}er\, x = A_2 \;\; ve \;\; y = B_2 \;\; ise \;\; OHalde \;\; z = f_2(x, y)$$

şeklinde ifade edilmektedir.

Birinci kuraldan çıkarılan kontrol hareketi değeri $\alpha_1 f_1(x_0, y_0)$, ikinci kuraldan çıkarılan kontrol hareketi değeri $\alpha_2 f_2(x_0, y_0)$ olduğu takdirde, sonuçta elde edilen en yeni kontrol hareketi iki kurallı bir durum olmaktadır. İlgili kontrol hareketi Eşitlik 2.12'deki gibi ifade edilebilmektedir. İlgili Eşitlik'te α_i i. kuralın ateşleme kuvvetini ifade etmektedir.

$$z_0 = \frac{\alpha_1 f_1(x_0, y_0) + \alpha_2 f_2(x_0, y_0)}{\alpha_1 + \alpha_2} \tag{2.12}$$

Söz konusu çıkarım yaklaşımı sayesinde Sugeno ve Nishida, kıvrımlı bir hatta bir model otomobili yavaşça hareket ettirmeyi başarmışlardır. Yine Sugeno ve Murakami tarafından, ilgili yaklaşım sayesinde, bir model otomobilin belirli bir güzergâhta hareket edip, daha sonra garaja kendi kendine park etmesi sağlanmıştır (Lee 1990, Lin and George Lee 1996). Takagi-Sugeno tipi bulanık çıkarım Şekil 2.8'de olduğu gibi ifade edilebilmektedir.



Şekil 2.8 Takagi-Sugeno tipi bulanık çıkarım (Deperlioğlu 2001).

2.1.6 Durulaştırma Yöntemleri

Bulanık mantıkta durulaştırma (defuzzification), sonuç olarak elde edilen bulanık kontrol hareketinin, kesin nitelikteki bir kontrol hareketine dönüştürülmesi işlemine

verilen genel bir addır. Durulaştırma yönteminin temel amacı, elde edilen bulanık kontrol hareketinin, mümkün olan dağılımını en iyi yansıtan kesin bir kontrol hareketini bulmaktır (Lee 1990, Yan *et al.* 1994, Lin and George Lee 1996, Kıyak ve Kahvecioğlu 2003). Durulaştırma, Eşitlik 2.13'de olduğu gibi ifade edilebilmektedir. Söz konusu eşitlikte, y bulanık kontrol hareketini, y_0 en son kontrol hareketini, durulaştırma ise durulaştırma operatörünü göstermektedir.

$$y_0 = \text{durulaştırma (y)} \tag{2.13}$$

Bulanık mantık kapsamında, Zadeh'in çalışmalarını takiben birçok farklı durulaştırma yönteminden bahsedilmiştir. Bu bağlamda farklı özellik ve işlevlerde, çeşitli durulaştırma yöntemlerinden bahsetmek mümkün olmaktadır (Zhao and Govind 1991, Passino and Yurkovich 1997, Jantzen 1998, Elmas 2003a). Günümüzde yaygın bir şekilde kullanılan durulaştırma yöntemleri ise "max kriteri", "ağırlık merkezi" ve "maksimumların ortalaması" yöntemleridir.

2.1.6.1 Max Kriteri Yöntemi

Max kriteri yöntemi ile sonuçta elde edilen bulanık kontrol hareketini temsil eden bulanık kümenin maksimum dereceli elemanı en son kontrol hareketi olarak alınmaktadır. Örneğin, sonuçta elde edilen bulanık kontrol hareketi üçgen üyelik fonksiyonuna sahip bir bulanık küme ise, bu vektör üçgenin tepe noktası olmaktadır.

2.1.6.2 Ağırlık Merkezi Yöntemi

Durulaştırma operatörü olarak oldukça yaygın bir kullanım şekline sahip olan ağırlık merkezi yöntemi (Center of Gravity – CoG), alan merkez yöntemi (Center of Area – CoA) olarak da adlandırılmaktadır. Bu yöntemle elde edilen bulanık kontrol hareketinin mümkün olan dağılımının ağırlık merkezi, en son kontrol hareketi olarak belirlenmektedir. Ağırlık merkezi yöntemi, Eşitlik 2.14'de olduğu gibi ifade edilebilmektedir.

$$z = \frac{\sum_{i=1}^{n} w_i C_i}{\sum_{i=1}^{n} w_i} \tag{2.14}$$

Eşitlik 2.14'de w_i i. kuralın etkili faktörü, C_i i. kuralın sonuç değeri, z ise gerçek sonuç değeri veya kontrol hareketini ifade etmektedir.

2.1.6.3 Maksimumların Ortalaması Yöntemi

Maksimumların ortalaması yöntemi (Mean of Maximum – MoM), eğer birden fazla maksimumlara ulaşan nokta varsa, bunların ortalamasının en son kontrol hareketi olarak alındığı bir durulaştırma yöntemidir. Örneğin, n tane üyelik fonksiyonu $\mu_z(w_i)$ maksimum değere ulaşan w_i noktası varsa, süreksiz evrenin bu durumundaki sistemin kontrol hareketi Eşitlik 2.15'de olduğu gibi hesaplanmaktadır.

$$z_0 = \sum_{i=1}^{n} (w_i / n) \tag{2.15}$$

Söz konusu durulaştırma yöntemleri değerlendirildiği zaman, her birinin farklı açıdan bir takım avantajlara sahip olduğu gözlemlenmiştir. Ancak günümüzde en çok kullanılan yöntem olarak, "ağırlıkların merkezi" yöntemi ön plana çıkmaktadır.

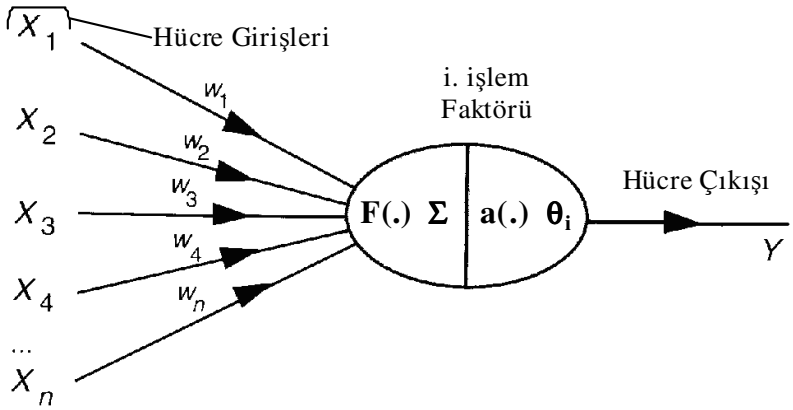
2.2 Yapay Sinir Ağları

Yapay sinir ağları (artificial neural networks), insan beyninin temel çalışma işlevlerine benzeyen birtakım özellikleri kullanarak geliştirilen ve yapılandırılan, günümüzde yaygın kullanım alanına sahip olan yapay zekâ sistemleridir. Bu sistemler, insan beyninin özelliklerinden olan öğrenme yolu ile yeni bilgiler türetebilme, yeni bilgiler oluşturabilme ve keşfedebilme gibi eylemleri otomatikman gerçekleştirebilmektedirler (Öztemel 2006). Yapay sinir ağları, genel anlamda yeni nesil bilgi işleme sistemlerini temsil etmektedirler. Geliştirilen yapay sinir ağı sistemleri genellikle optimizasyon,

fonksiyon tahmini, ilişkilendirme, model sınıflandırılması, veri sınıflandırılması ve kontrol gibi çeşitli işlemlerde kullanılmaktadır (Basheer and Hajmeer 2000, Deperlioğlu 2001, Elmas 2003b, Yegnanarayana 2006). Yapay sinir ağları ile ilgili olarak, McCulloch ve Pitts'in (1990) 1943 yılında ilk ortaya attığı sinir modelinden sonra birçok yenilik ve gelişmeler ortaya çıkmış ve günümüzde farklı şekillerde kullanılan birçok yapay sinir ağı modelleri ve işlevleri ortaya çıkmıştır.

2.2.1 Yapay Sinir Ağlarının Temel Yapısı

Tipik bir yapay sinir ağı sistemi, farklı yapılarda organize edilmiş, paralel işleyen ve tamamen birbirine bağlı olan, çok sayıda işlem elemanlarından oluşmaktadır. Bu işlem elemanları yapay sinir hücreleri olarak adlandırılmaktadır. Söz konusu yapay sinir ağları hücreleri de, sinaptik düğüm adı verilen çeşitli yapılarla birbirlerine bağlı durumda olmaktadır. Bütün bu yapı, beyindeki gerçek sinirlerin ağ şekillerinden esinlenerek oluşturulmuştur. Sonuç olarak bu yapı ile birlikte yapay sinir ağları, insan beyni ile ortak nitelikte olan öğrenme yeteneği, hatırlama ve tecrübelerden genelleştirme gibi yetenekleri sergilemektedir. Biyolojik sinir yapısının matematiksel bir modeli, McCulloch ve Pitts tarafından 1943 yılında ortaya atılmış ve kısaca M-P siniri olarak adlandırılmıştır. Şekil 2.10, söz konusu matematiksel modeli kısaca göstermektedir.



Şekil 2.9 McCulloch ve Pitts sinirinin temel yapısı.

McCulloch ve Pitts sinirindeki i. işlem faktörü, Eşitlik 2.16'da olduğu gibi girişlerinin ağırlıklı toplamını hesaplamakta ve çıkışlar, ateşleme varsa $y_i=1$, yoksa $y_i=0$ olarak, kesin eşik değeri θ_i'nin altında ya da üzerinde olan, söz konusu ağırlıklı giriş toplamına ayarlanmaktadır.

$$y_i(t+1) = a\left(\sum_{j=1}^{n} w_{ij} x_j(t) + \theta \right) \tag{2.16}$$

Yine söz konusu yapıda gösterilen aktivasyon fonksiyonu a(f) ünite adım fonksiyonudur ve Eşitlik 2.17'de olduğu gibi ifade edilmektedir.

$$a(f) = \begin{cases} 1 & f \geq 0 \ için \\ 0 & diger \ durumlar \ için \end{cases} \tag{2.17}$$

Yapıda kendisine yer verilen w_{ij} ifadesi, kaynak sinir j'den hedef sinir i'ye bağlanan ve sinaptik bağlantı olarak adlandırılan yapının kuvvetini temsil etmektedir. Bu noktada, pozitif nitelikte ağırlık, tahrik edici sinaptik bağlantıya, negatif nitelikte ağırlık ise engelleyici sinaptik yapıya karşılık gelmektedir. Şayet $w_{ij}=0$ durumu söz konusu ise iki sinir arasında bağlantı yok demektir. Eşitlik 2.16, işleme birimi t ve (t+1) arasında geçen zaman anı olarak ele alınmıştır.

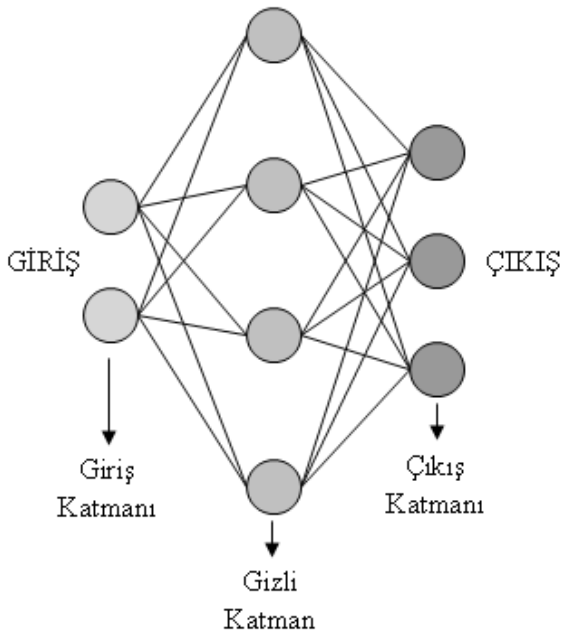
Yapay sinir ağlarında, problem niteliğine göre farklı sonuçlar üretmek amacıyla, farklı türde birçok aktivasyon fonksiyonu kullanılabilmektedir. Bu noktada bir yapay sinir ağında oluşturulan her yapay sinir hücresinin aynı aktivasyon fonksiyonunu kullanması şart değildir. Günümüzde yaygın olarak kullanılan aktivasyon fonksiyonları "lineer fonksiyon", "step fonksiyonu", "sinüs fonksiyonu", "eşik değer fonksiyonu" ve "hiperbolik tanjant fonksiyonu" olarak değerlendirilmektedir (Elmas 2003b, Öztemel 2006).

Bir biyolojik sinir hücresi, basit anlamda ikili eşik birimi olarak yapılanmasına rağmen, McCulloch-Pitts siniri oldukça zengin hesaplama potansiyeline sahip bir yapıda ortaya

çıkmaktadır. Nitekim M-P siniri yardımıyla, ağırlıklar ve eşikler uygun bir şekilde ayarlandığında "değil", "veya" ve "ve" gibi temel mantık işlemleri kolaylıkla yerine getirilebilmektedir. Genelleştirme yapılacak olursa bu durumun, bu türdeki sinir yapılarının eş zamanlı montajı sonucunda, sayısal bilgisayarlarda olduğu gibi genel hesaplamaları yerine getirebilme yeteneğini ortaya çıkardığı yorumu yapılabilmektedir (Kosko 1992, Brown and Harris 1995).

2.2.2 Yapay Sinir Ağlarında Mimariler

Yapay sinir hücreleri ve ilgili sinaptik bağlantılar, geliştirilen herhangi bir yapay sinir ağında farklı şekillerde bir araya getirilip organize edilebilmektedir. Buna bağlı olarak da farklı biçimlerde yapay sinir ağları meydana getirilebilmektedir. Bu farklı biçimlerdeki ağlar, farklı yapay sinir ağları mimarilerine karşılık gelmektedir. Söz konusu mimariler sahip oldukları biçim ve işlev özelliklerine göre farklı isimlerle anılmaktadır. Günümüzde sıkça kullanılan yapay sinir ağları mimarilerinden birisi de "ileri beslemeli çok katmanlı ağ" olarak adlandırılmaktadır. Şekil 2.11'de, ileri beslemeli çok katmanlı yapay sinir ağına dair bir örnek yapı gösterilmektedir.



Şekil 2.10 İleri beslemeli çok katmanlı bir yapay sinir ağı yapısı.

Bir ileri beslemeli çok katmanlı yapay sinir ağı mimarisinde aşağıdaki temel özelliklerden bahsetmek mümkündür:

- Ağ içerisindeki yapay sinir hücreleri katmanlar halinde mimariye yerleştirilmiştir.
- Her katmandaki yapay sinir hücrelerine, sadece önceki katmandaki yapay sinir hücrelerinden girişler söz konusudur.
- İlk katmandan (giriş katmanı) verilen bilgi, ağın içerisinden yapay sinir hücreleri aracılığıyla ileriye doğru yayılmakta ve bu nedenle yapay sinir hücrelerinin çıkış sinyalleri kesinlikle kendi giriş bilgilerini biçimlendirememektedir.
- Son katmandaki (çıkış katmanı) sinyaller ağın çıkışı olmaktadır.

İleri beslemeli ağlardan farklı olarak, geri beslemeli veya tekrarlanan ağlarda en azından bir yapay sinir hücresinin geriye doğru yayıldığı bir dönüş bağlantısı vardır ve bu sinyalin düzenlenmiş şekli aynı düğümün girişinde kullanılmaktadır. Tekrarlanan ağlar tamamen veya parçalı olarak geri besleme yollarına sahip olmakta ve bu nedenle, bu türden ağların tasarımları ve davranışları oldukça karmaşık olabilmektedir (Jang *et al.* 1997).

2.2.3 Yapay Sinir Ağlarında Öğrenme ve Genel Öğrenme Kuralları

Geliştirilen yapay sinir ağları, akıllı davranışı ve hedefe ulaşmadaki işlevselliği sağlamak için, uygun bir şekilde yapılandırılmalı ve organize edilmelidir. Bunun için de doğru sinaptik bağlantıların seçilmesi ve yine bu sinaptik bağlantıların uygun ağırlıkta organize edilmesi son derece önemlidir. Geliştirilen ağ yapısı da söz konusu bu şartları karşılayabilmek için problemle alakalı sistemin davranışlarını öğrenmek ya da kendi kendini organize etmek zorunda kalmaktadır. Bu noktada yapay sinir ağlarının eğitimi gerçekleştirilmeli ya da diğer bir deyişle yapay sinir ağlarının öğrenmesi sağlanmalıdır. Bu kapsam dâhilinde ele alındığında öğrenme kavramı, kalıcı yenilenmeler için gözlemleme ya da eğitim faaliyetlerinden elde edilen sonuçlar olarak tanımlanabilmektedir. Yapay sinir ağlarında eğitim için kullanılan öğrenme kuralları genellikle "danışmanlı öğrenme (supervised learning)", "danışmansız öğrenme

(unsupervised learning)" ve "yaparak ya da pekiştirerek öğrenme (reinforcement learning)" olmak üzere üç başlık altında incelenebilmektedir.

Danışmanlı öğrenmede, arzu edilen ağ çıkışının elde edilebilmesi için, çıkış hatasının düşürülmesi amacıyla ağırlıkların uyarlamalı hale getirilmesi gerekmektedir. Danışmanlı öğrenme kuralında uygulanan yaklaşım ne kadar etkiliyse, söz konusu yaklaşım bir o kadar da karmaşık hesaplama yapısı gerektirmektedir. Bu öğrenme kuralında geliştirilen bir ağın sinirsel değerlerinin her bir kümesi bir hata değerini ve bu kümelerin mümkün olanları da inişli çıkışlı hata yüzeyini tanımlamaktadır. Yapay sinir ağı bu noktada hata yüzeyindeki bir noktadan başlamakta ve yüzeyi en küçük hata noktasına taşımaya çalışmaktadır. Danışmanlı öğrenme kuralına çok katmanlı perceptron (MLP – multi layer perceptron), geri yayılım (backpropagation), üçgen kuralı (delta rule), Widrow-Hoff ya da en küçük ortalama kare (least mean square) ve karşılığı "uyarlamalı doğrusal eleman" anlamına gelen ADALINE örnek olarak verilebilmektedir (Kosko 1992, Lin and George Lee 1996, Jang *et al.* 1997, Kosko 1997, Drew and Monson 2000).

Danışmansız öğrenme, sadece ağa sunulan eğitim girişlerine dayanarak, ağın yapısını organize etmek amacıyla kullanılan bir öğrenme kuralıdır. Bu öğrenme kuralında bir danışman ya da öğretmen, yapay sinir ağına uygulanan girişin hangi veri parçası sınıfına ait olduğunu ya da söz konusu ağın hangi durumda iyi sonuç vereceğini açıklamamaktadır. Burada geliştirilen ağ, bütün veriyi, üyeleri birbirinin benzeri olan çeşitli gruplara ayırmaktadır. Danışmansız öğrenme, danışmanlı öğrenmeye göre daha hızlı bir öğrenme kuralı olarak değerlendirilmektedir. Ayrıca, danışmansız öğrenmedeki matematiksel algoritmalar daha basit bir yapıdadır ve bu öğrenme kuralı kapsamında verinin bir kez geçişi söz konusu olmaktadır. Danışmansız öğrenme kuralına örnek olarak "rekabete dayanan öğrenme (competetive learning)", Kohonen'in "kendi kendine organize (self-organizing) ağları", "Hebbian öğrenme" ve "Grossberg öğrenme" gibi kurallar ve yapılar örnek olarak verilebilmektedir (Kosko 1992, Jang *et al.* 1997).

Yaparak ya da pekiştirerek öğrenme kuralında, ağ doğrudan gerçek ağ çıkışını vermemekte, bu noktada ağ çıkışının "iyi" veya "kötü" olarak değerlendirilmesi

yapılmaktadır. Bu öğrenme kuralı kapsamında geliştirilen ağlarda performans bilgisi genellikle binary türünde olmakta ve denetim hareketleri sırasının başarısını göstermektedir (Brown and Harris, 1995).

2.2.4 İleri Beslemeli Çok Katmanlı Yapay Sinir Ağları için Geri Yayılım Algoritması ve Öğrenme

Geri yayılım algoritması, yapay sinir ağlarının danışmanlı öğrenme kuralına giren bir algoritma yapısıdır. Bu algoritma kapsamında, ağın girişleri ve çıkışları arasındaki hata sinyaline bağlı olarak, ağdaki ağırlıklar güncellenmektedir. Hata yani *e(k)*, arzu edilen çıkış (gerçek çıkış - *y(k)*) ile yapay sinir ağının çıkışı *o(k)* arasındaki fark olmaktadır. Söz konusu yaklaşım Eşitlik 2.18'de olduğu gibi ifade edilebilmektedir.

$$e(k) = y(k) - o(k) \tag{2.18}$$

Genel anlamda bir geri yayılım algoritması aşağıdaki işlem adımlarını izleyerek çalışmaktadır (Öz *vd.* 2002):

Adım 1- Ağ yapısı belirlenir;

- Giriş katmanındaki nöron sayısı, çıkış katmanındaki nöron sayısı ve gizli katman sayısı ile her gizli katmandaki nöron sayıları belirlenir.

Adım 2- Ağ'ın başlangıç parametreleri belirlenir;

- Başlangıç ağırlıkları, öğrenme katsayısı ve momentum katsayısı belirlenir.

Adım 3- Giriş ve çıkış verileri ağın değerlendirebileceği şekilde "normalize" edilir.

Adım 4- İleri beslemeli ağ yapısına göre ağ çıktıları hesaplanır.

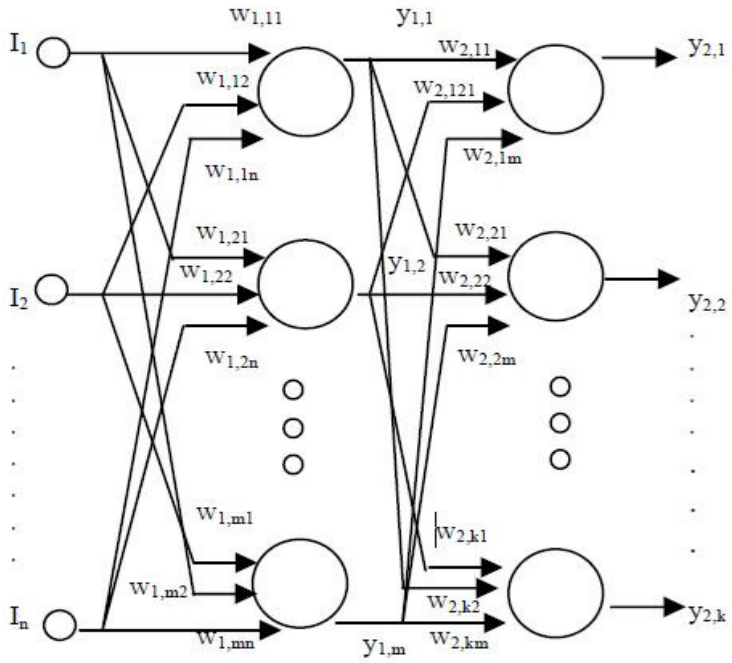
Adım 5- Ağ çıktısı ile gerçek çıktı arasındaki hata bulunur.

Adım 6- Hata minimum ise ağ problemi öğrenmiş demektir; öğrenme durdurulur.

Adım 7- Hata minimum değilse; geriye yayılım ağ yapısına göre, hatayı minimize edecek şekilde ağırlıklar hesaplanır.

Adım 8- Adım 3'e dönülerek işleme devam edilir.

Şekil 2.12'de, Öz *vd.* (2002) yaklaşımı kapsamında katmanlar girişten ileriye doğru L=0 (giriş katmanı), L=1 (Gizli katman), L=2 (çıkış Katmanı) olarak numaralandırılmıştır. Ağırlıklar ise giriş katmanı ile gizli katman arasındaki ağırlıklar $W_{1,ji}$, gizli katman ile çıkış katmanı arasındaki ağırlıklar $W_{2,tj}$ olarak gösterilmiştir. i=1,2,...n giriş nöronları, j=1,2,...m gizli katman nöronları, t=1,2,...k çıkış katmanı nöronları olmaktadır.



Şekil 2.11 İleri beslemeli çok katmanlı örnek bir yapay sinir ağı yapısı (Öz *vd.* 2002).

İlgili örnek kapsamında, basit bir yapay sinir hücresinin tanımından yararlanarak, her bir katman sonundaki çıkışlar aşağıdaki gibi yazılabilmektedir (Öz *vd.* 2002):

Gizli katman için,

$$y_{NET1,j} = \sum_{i=1}^{n} w_{1,ji} I_i$$
$$y_{1,j} = fj[y_{NET1,j}] \qquad j = 1,2,...m \tag{2.19}$$

Çıkış katmanı için,

$$y_{NET2,t} = \sum_{i=1}^{m} w_{2,tj} y_{1,j}$$
$$y_{2,k} = f_t[y_{NET2,t}] \qquad t = 1,2,...k \tag{2.20}$$

Eşitlik 2.19 ve Eşitlik 2.20'de Y_{NET} ifadeleri aktivasyon fonksiyonuna karşılık gelmektedir.

Belirtildiği üzere, geri yayılım algoritmasıyla bir yapay sinir ağını eğiterek, seçilen bir giriş veri setine karşılık olarak özel bir fonksiyonel karakteristiği veren çıkışları oluşturmak için, ilgili yapay sinir hücrelerinin ağırlıkları güncellenmektedir. Bu bağlamda, eğitim sürecinde kullanılması için, tasarlanmış bir yapay sinir ağı yapısına, giriş vektörü ile birlikte bu giriş vektörüne karşılık olan, arzu edilen çıkış vektörü verilmelidir. Söz konusu bu vektörlerin bütünü bir eğitim-öğrenme çiftini ifade etmektedir.

Tipik bir eğitim süreci sonucunda, tasarlanmış yapay sinir ağının "eğitilmiş" ya da "öğrenmiş" sayılabilmesi için, ağın çıkışı ile arzu edilen çıkış bilgisi eşit veya birbirine yakın değerde olmalıdır. Eşitlik olmadığı zaman, ilgili çıktılar arasındaki farka bakılarak "hata" değeri tespit edilmektedir.

İlgili hata ve çıkış katmanı ile gizli katman arasındaki ağırlıkların yenilenmesi ise (Öz *vd*. 2002);

$$\varepsilon_t = y_{2,t}^{H} - y_{2,t}^{A} \tag{2.21}$$

$$\delta_t = f^{'}(y_{NET2,t})\,\varepsilon_t \tag{2.22}$$

$$\Delta w_{2,jt}^{n} = \eta \delta_t y_{1,t} + \alpha \Delta w_{2,jt}^{n-1}$$
$$w_{2,jt}^{n} = w_{2,jt}^{n-1} + \Delta w_{2,jt}^{n} \qquad (2.23)$$

eşitlikleri aracılığıyla gerçekleştirilmektedir. Söz konusu eşitliklerde ifade edilen n iterasyon sayısını, η öğrenme katsayısını, α momentum katsayısını göstermektedir.

Diğer taraftan, gizli katman ile giriş katmanı arasındaki ağrılıkların yenilenmesi (Öz *vd.* 2002);

$$\delta_j = y'_{2,j} \sum_{t=1}^{k} \delta_t w_{2,jt} \qquad (2.24)$$

$$\Delta w_{1,ij}^{n} = \eta \delta_j y_{0,i} + \alpha \Delta w_{1,ij}^{n-1}$$
$$w_{1,ij}^{n} = w_{1,ij}^{n-1} + \Delta w_{1,ij}^{n} \qquad (2.25)$$

eşitlikleri ile yerine getirilmektedir.

Söz konusu hesaplamalar, yapay sinir ağındaki toplam hata değerinin, istenen belli bir sınır değerinin altına düşünceye kadar ya da belirlenen bir iterasyon sınırına ulaşılıncaya kadar (tekrar tekrar) gerçekleştirilmektedir. Şayet seçilen şart sağlanırsa, tasarlanmış olan yapay sinir ağı "eğitildi" ya da yine ilgili ağ belirlenen problemi "öğrendi" denilebilmektedir.

2.3 Bulanık Mantıkta Uygulama Yazılımları ve Eğitimsel Özellikler

Çalışma kapsamında incelenebilecek, piyasada bulunabilecek ya da çeşitli araştırma çalışmaları ışığında geliştirilmiş olan, farklı özellik ve işlevlerde, birçok bulanık mantık uygulama yazılımı bulunmaktadır. İlgili yazılımları, sundukları özellik ve işlevlerin yanında, eğitimsel yönden yaklaşımları konusunda incelemek, geliştirilen uygulama – eğitim yazılımının söz konusu alandaki yeri konusunda fikir sahibi olunmasını sağlayacaktır.

2.3.1 Fuzzy TECH

Fuzzy TECH yazılımı, Inform yazılım firması tarafından piyasaya sürülen, geniş çapta özellik ve işlevleriyle birlikte ön plana çıkan, günümüz popüler bulanık mantık uygulama yazılımlarından birisidir (İnt.Kyn.1). Yazılımın ilk göze çarpan özelliği, Windows Gezgini yapısında olduğu gibi, kullanıcıya, pencere ve ağaç yapısı mantığına dayalı bir kullanım şekli sunmasıdır. Bu sayede kullanıcı, herhangi bir bulanık kontrol sisteminin tasarımını ve geliştirilmesini kolay ve hızlı bir biçimde gerçekleştirebilmektedir. Yazılım kapsamında geliştirilen bir bulanık kontrol sisteminin giriş ve çıkış değişken(ler)i, değişkenler için tanımlanan üyelik fonksiyonları gibi özellikler, yazılımın arayüzünde, ağaç yapısı mantığına dayalı bir şekilde gözlemlenebilmektedir. Yazılım kapsamında oluşturulmuş olan bir sistem ayrıca yine arayüzde yer alan bir önizleme penceresinde, görsel olarak kullanıcılara sunulmaktadır. Geliştirilen sistemle alakalı bir özelliği düzenlemek isteyen kullanıcı, söz konusu bu pencere üzerinde yer alan görsel nesneleri de kullanabilmektedir.

Fuzzy TECH yazılımın önemli özelliklerinden bir diğeri de sunduğu görsel nitelikteki araçlardır. Yazılım bünyesinde yer alan kontroller aracılığıyla, tanımlanmış olan değişkenler ya da üyelik fonksiyonları gibi elementler, görsel yollarla incelenerek organize edilebilmekte ve sisteme eklenebilmektedir. Tasarlanmış olan bulanık kontrol sistemleriyle alakalı çıkış ve benzetim verileri, farklı nitelik ve kapsamdaki, renkli grafiklerle incelenebilmektedir. Yazılım, bünyesinde sunduğu araçlar sayesinde, geliştirilen bir bulanık kontrol sistemiyle alakalı benzetim uygulamaları gerçekleştirilmesine imkân vermektedir. Bulanık mantıktan farklı olarak, Fuzzy TECH kapsamında sunulan özellik ve işlevler sayesinde sinirsel – bulanık (neuro – fuzzy) sistemler de geliştirilebilmektedir.

Eğitimsel açıdan incelendiği takdirde, Fuzzy TECH programı birçok etkili özellik ve işlevi kullanıcılara sunmaktadır. Yazılım bünyesinde gerçekleştirilen işlemlerin görsel ve renkli kontrollerle desteklenmesi, kolay bir kullanım tecrübesi vaat etmektedir. Nitekim Windows Gezgini yapısına benzer bir şekilde oluşturulmuş, pencere ve ağaç yapısı kullanım yaklaşımları da, kullanıcıların yazılıma kullanım açısından yabancılık

çekmemesini sağlamaktadır. Yazılımın bahsedilmesi gereken dezavantajı ise, geliştirilebilecek ileri düzey çalışmalarda kullanıcılara artan düzeyde, karışık bir kullanım düzeyi sunması ve kullanım bağlamında yine belli bir miktar ön bilgi gerektiriyor olmasıdır.

2.3.2 Dot Fuzzy

Dot Fuzzy yazılımı, açık kaynak kodlu olarak, nesneye yönelik programlama yaklaşımları dâhilinde geliştirilmiş olan, ufak çaplı bir uygulama yazılımıdır. Yazılım, Michele Bertoli tarafından, tamamen C# programlama dili kullanılarak programlanmış ve geliştirilmiştir (İnt.Kyn.2). Yazılımın kullanımı, sunulan basit "grid" kontroller sayesinde oldukça kolaydır. Tanımlanan üyelik fonksiyonları, yine yazılım arayüzünde sunulmuş olan kontroller aracılığıyla, kullanıcılar için görsel bir hale getirilmektedir. Yazılım ayrıca, geliştirilen uygulamaların kaydedilmesi amacıyla XML dosya yapısından yararlanmaktadır. Bu sayede basit, hızlı ve hiyerarşik bir yapıda veri kaydı imkânı kullanıcılara sunulmaktadır.

Eğitimsel açıdan incelendiği zaman, yazılımın sunulan kontroller ve arayüz açısından basit bir kullanım tecrübesi vaat ettiği görülmektedir. Yazılım ayrıca ufak boyutuyla birlikte farklı bir çalışma ortamına kolaylıkla taşınabilmektedir. Ancak yazılımın kullanımı, bulanık mantık konusunda ciddi bir ön bilgi düzeyine sahip olmayı da gerektirmektedir. Çünkü yazılım arayüzlerinde, kullanım şekline bağlı olarak, kullanıcıyı yönlendirebilecek, yardımcı özellik ve işlevlere rastlanılmamaktadır. Yazılımın diğer önemli bir eksiği de, yeni bir sistem tasarımında, kullanıcıyı büyük ölçüde klavyeye bağlamasıdır. Giriş ve çıkış değişken(ler)inin ya da üyelik fonksiyonlarının tanımlanması aşamasında, kullanıcı zamanının büyük bir bölümünü fare ve klavye arasında geçişler yaparak geçirmektedir. Özellikle, geliştirilmiş olan bir sistem için tanımlanacak olan dilsel kuralların, kullanıcı tarafından, Dot Fuzzy yazılımına tek tek tanıtılması gerekmektedir. Yazılımın diğer bir eksiği de, sonuç ya da çıkış verilerinin incelenebilmesi için grafik seçenekleri ya da benzeri görsel faktörler sunmamasıdır.

Dot Fuzzy yazılımı teknik açıdan incelendiğinde, bazı önemli işlevlerin de son sürüm bünyesinde (16.03.2010 tarihi itibariyle) eksik durumda olduğu gözlemlenmektedir. Yazılımda sadece üçgen ve yamuk tipi üyelik fonksiyonları desteklenmekte, durulaştırma yöntemi olarak da sadece ağırlık merkezi yöntemi sunulmaktadır.

2.3.3 MATLAB Fuzzy Logic Toolbox

MATLAB yazılımı, MathWorks firması tarafından geliştirilmiş olan, teknik hesaplama dili ve platformu olarak kullanılabilen, bünyesinde birçok yardımcı araç ve yazılım ile birlikte gelen, günümüzün popüler, teknik uygulama yazılımlarından birisidir (İnt.Kyn.3). MATLAB sayesinde, birçok farklı alan için teknik özellikte ve işlevde ileri düzeyde hesaplamalar yapılabilmekte, geliştirilen uygulamaların, yine MATLAB ortamında benzetimleri gerçekleştirilebilmektedir. Bu bağlamda, farklı alanda uygulamalara hitap eden, birçok farklı eklenti yazılımı, MATLAB aracılığıyla kullanıcılara sunulmaktadır. Fuzzy Logic Toolbox yazılımı da, MATLAB ortamında, bulanık mantık tekniğine dayalı uygulamaların gerçekleştirilmesine imkân tanıyan bir uygulama yazılımı olarak ortaya çıkmaktadır.

Fuzzy Logic Toolbox yazılımı temel olarak beş ayrı arayüzden oluşmaktadır (İnt.Kyn.4). Kullanıcılar bir bulanık kontrol sisteminin tasarımı ve uygulanması işlemini söz konusu bu arayüzleri kullanarak kolaylıkla gerçekleştirebilmektedir. Fuzzy Logic Toolbox yazılımı ilk çalıştırıldığı anda kullanıcının karşısına çıkan arayüz "FIS Editor" olarak adlandırılmaktadır. Söz konusu bu arayüz kullanılarak bulanık kontrol sisteminin tasarlanması ve düzenlenmesi ile ilgili işlemler gerçekleştirilmektedir. Arayüz üzerinde yer alan kontroller yardımıyla bir bulanık kontrol sisteminin modeli, içerme türü, giriş ve çıkış değişken(ler)i gibi nitelikleri tanımlanabilmektedir. Söz konusu bu yazılım, özellikle son yıllarda akademik ve endüstriyel ortamlarda yaygın bir kullanım alanına sahip olmuştur. Yazılım, MATLAB altyapısının gücü ve esnekliği ile birlikte birçok farklı sistemin tasarlanmasına ve uygulanmasına izin vermektedir. Simulink ortamında, hazır olarak sunulan blok yapıları kullanılarak, sürükle – bırak yaklaşımına göre sistemler tasarlanabilmektedir. Bu noktada, Simulink ortamında tasarlanan bir bulanık

kontrol sisteminde, bulanık mantık yaklaşımı, Fuzzy Logic Toolbox yazılımdan yararlanarak uygulanmaktadır.

Fuzzy Logic Toolbox yazılımı eğitimsel açıdan değerlendirildiği zaman, birçok kullanım kolaylığını ve avantajını da beraberinde getirdiği gözlemlenmektedir. Yazılımın kullanım özellikleri basit yapılarda görsel kontrollerle desteklenmiş, bu yolla kullanıcıların yazılımı daha kolay ve etkili kullanması sağlanmıştır. Yazılım bünyesinde üyelik fonksiyonları tanımlanırken, sürükle – bırak yaklaşımları sayesinde ilgili fonksiyon parametrelerinin daha kolay bir şekilde düzenlenebilmesine olanak verilmiştir. Rule Viewer arayüzünde, etkileşimli ve görsel özellikler sayesinde, istenilen giriş değerlerine karşılık, geliştirilen sistemin ne tür cevaplar [çıkış(lar)] verdiği daha kolay ve anlaşılır bir şekilde görüntülenebilmektedir. Ayrıca Surface Viewer arayüzü aracılığıyla, farklı nitelikteki grafikler kullanılarak, sistemden elde edilen veriler daha etkili bir şekilde kullanıcılara sunulabilmektedir. Daha önce de belirtildiği üzere, Fuzzy Logic Toolbox yazılımının önemli bir avantajı da Simulink yazılımı ile bütünleşik kullanılabilmesidir. Bu sayede yazılımın uygulanabileceği problem ve uygulama kapsamı büyük ölçüde genişletilmektedir. Ayrıca Simulink sayesinde, ilgili kullanıcılar teknik çalışmalarını, kolay ve hızlı bir şekilde yerine getirebilmektedir. Bu avantajların yanında, Fuzzy Logic Toolbox ile ilgili değinilebilecek birkaç dezavantaj da öne sürülebilmektedir. Buna göre, ilgili yazılımın sunduğu teknik anlamdaki bazı özellik ve işlevlerin, daha çok ileri düzeyde kullanıcıları hedef alması, bulanık mantık tekniğine yeni başlayanlar için bir miktar dezavantaj sayılabilmektedir. Bunun yanında, yazılımın genel olarak kolay bir kullanım yapısı olmasına rağmen, sunulan bazı işlevlerin, daha etkili kontroller yardımıyla, daha basit bir düzeyde ortaya konulabileceği de öne sürülebilmektedir.

2.3.4 NEFCLASS

NEFCLASS yazılımı, bulanık kontrolden çok, sinirsel-bulanık sistemler için geliştirilmiş bir uygulama yazılımıdır (İnt.Kyn.5). Bu noktada yazılımın ilgi çeken özelliği, bazı görsel özelliklerle desteklenmiş, MS-DOS işletim sistemi tabanlı bir yazılım platformu sunmasıdır. Yazılım bünyesinde klavye ve / veya fare kontrolü ile

sinirsel-bulanık sistemler üzerine kurulu uygulamaların tasarlanması ve geliştirilmesi mümkündür. Yazılım ufak boyutlu bir yapıdadır ve dolayısıyla kolay bir şekilde taşınabilmektedir.

Yazılım eğitimsel açıdan incelendiğinde, öncelikli olarak sinirsel-bulanık sistemleri desteklemesi, sadece bulanık kontrol sistemlerin tasarlanması ve geliştirilmesine imkân sağlayan uygulama yazılımları açısından, NEFCLASS'ı farklı bir konuma yerleştirmektedir. Bunun dışında, yazılımın günümüzde popülerliği olmayan bir işletim sistemi üzerinde çalışması, önemli bir dezavantaj olarak ortaya çıkmaktadır. Bu sebepten ötürü, söz konusu yazılım Windows XP ve sonrası işletim sistemlerinde uyumsuzluk problemleri ortaya çıkarmaktadır. Yazılım her ne kadar MS-DOS işletim sisteminin elverdiği ölçüde bazı görsel özellikler ve bu bağlamda incelenebilecek işlevler sunsa da, söz konusu özellik ve işlevler, gerek genel kullanım, gerekse eğitimsel açıdan düşük düzeyde kalmaktadır.

2.3.5 jFuzzy Logic

jFuzzy Logic yazılımı, Java ortamında, Pablo Cingolani tarafından tasarlanmış ve geliştirilmiş olan, bir başka açık kaynak kodlu bulanık mantık uygulama yazılımıdır (İnt.Kyn.6). Yazılım, benzer uygulamalara göre teknik açıdan oldukça gelişmiş bir düzeyde uygulama platformu sunmaktadır. Bu bağlamda, uygulamalarda fazla kullanım alanı bulmayan üyelik fonksiyonları, durulaştırma yöntemleri gibi çeşitli özellik ve işlevleri desteklemektedir. Bunun dışında, Java ortamının sunduğu görsel ve etkileşimli programlama özellikleri sayesinde kullanımı kolay ve etkili bir yazılım platformu sunmaktadır.

jFuzzy Logic eğitimsel açıdan incelendiğinde, sunduğu görsel ve etkileşimli özellikler ve işlevler sayesinde önemli bir avantaja sahip olduğu söylenebilmektedir. Java ortamı ve nesneye yönelik programlama yaklaşımının sunduğu kolaylıklar sayesinde de hızlı, etkili ve görsel yönden nitelikli uygulama arayüzleri sunmaktadır. Yazılım ayrıca pencere mantığına göre çalışmakta, bu durum da tıpkı Fuzzy TECH yazılımında olduğu gibi, kullanıcıların aşina olduğu bir kullanım tecrübesi ortaya koymaktadır. Yazılımın

değinilebilecek tek dezavantajı, ortalama bir düzeyin üstünde bulanık kontrol sistemleri ile çalışılması için yeterli özellik ve işlevlere sahip olmamasıdır. Bu açıdan ele alındığında, jFuzzy Logic yazılımının, Fuzzy TECH ve MATLAB Fuzzy Logic Toolbox gibi yazılımlar karşısında çok daha basit bir düzeyde olduğu gözlemlenmektedir.

2.3.6 Kütüphane Niteliğindeki Yazılımlar

Bahsedilen uygulama yazılımlarına ek olarak, bulanık mantık tekniği ve bulanık kontrol yaklaşımı için, farklı platformlarda geliştirilmiş olan, birçok bulanık mantık kütüphane (library) yazılımı da bulunmaktadır. Söz konusu bu kütüphane yazılımlarının büyük bir kısmı ücretsiz ve / veya açık kaynak kodlu olarak kullanıcılara sunulmaktadır. Günümüzde popüler olarak kullanım alanı bulan bu kütüphanelerden birisi Free Fuzzy Logic Library (FFLL) yazılımıdır. Söz konusu yazılım açık kaynak kodlu olarak, nesneye yönelik programlama yaklaşımları çerçevesinde geliştirilmiştir (İnt.Kyn.7). Dünya çapında birçok geliştirici tarafından, ilgili kütüphane yazılımı için, uyumlu arayüzler tasarlanmakta ve Internet üzerinden paylaşılmaktadır. FFLL yazılımından farklı olarak, AForge.Net kütüphane yazılımı da bir başka bulanık mantık kütüphane yazılımı olarak değerlendirilmektedir. Bu yazılım da açık kaynak kodlu olarak, C# programlama dili kullanılarak programlanmış ve geliştirilmiştir. AForge.Net kütüphane yazılımının dikkat çekici diğer bir özelliği, bulanık mantık tekniğinin yanında, yapay sinir ağları tekniği, genetik algoritma tekniği, görüntü işleme, makine öğrenmesi ve robotik alanlarında çeşitli uygulamaların yapılmasına da imkân veren, ek kütüphane yazılımı kodlarını da bünyesinde barındırmasıdır (İnt.Kyn.8). Kütüphane yazılımlarını eğitimsel açıdan ele almak, belirli yapılarda kullanıcı arayüzleri sunmadıklarından dolayı mümkün olmamaktadır. Daha önce de değinildiği üzere, FFLL yazılımı için birçok arayüz tasarlanmasına rağmen, söz konusu bu arayüzlerin sürekliliği ve kapsamı tartışılabilir nitelikte olduğundan, ilgili yazılımı eğitimsel açıdan ele almak, değerlendirmenin geçerliliğini etkileyecektir.

2.4 Yapay Sinir Ağlarında Uygulama Yazılımları ve Eğitimsel Özellikler

Bulanık mantık tekniğinde olduğu gibi, yapay sinir ağları tekniği için de, çalışma kapsamında incelenebilecek, piyasada bulunabilecek ya da çeşitli araştırma çalışmaları bünyesinde geliştirilmiş olan, farklı özellik ve işlevlere sahip, birçok uygulama yazılımı bulunmaktadır. Bu yazılımların incelenmesinde izlenecek yaklaşım, bulanık mantık uygulama yazılımları için söz konusu olan yaklaşımla aynı yöndedir.

2.4.1 MATLAB – Neural Networks Toolbox

MATLAB ile birlikte gelen Neural Networks Toolbox uygulama yazılımı, MATLAB yazılımının da etkisi ve desteği ile birlikte, yapay sinir ağları üzerine oldukça geniş bir uygulama alanını destekleyen, teknik anlamda etkili kullanım özellikleri ve işlevleri sunan, popüler bir platform olarak kullanıcılara sunulmaktadır. Yazılım bünyesinde yer alan görsel arayüz sayesinde, yapay sinir ağları sistemleri hızlı bir şekilde tasarlanıp, uygulamaya dönüştürülebilmekte, yine görsel işlevlerle birlikte, gerçekleştirilen çalışmalarla alakalı veriler etkili bir şekilde gözlemlenebilmektedir (İnt.Kyn.9). Neural Networks Toolbox yazılımı yaygın bir şekilde kullanılan, çok katmanlı ağ ve Kohonen kendi kendine organize ağları gibi yaygın kullanım alanı bulan mimariler ile birlikte birçok farklı yapay sinir ağları mimarisini ve ilgili öğrenme kurallarını desteklemektedir (İnt.Kyn.10). Yazılım bünyesinde sunulan kontroller yardımıyla, üzerinde çalışılan sistemin kaynakları elverdiği ölçüde yapay sinir ağı yapıları tasarlanıp, düzenlenebilmektedir. Geliştirilen bir uygulama ile ilgili performans ve eğitim süreçleri, etkili görsel işlevlerle kullanıcılara sunulmaktadır. Tıpkı Fuzzy Logic Toolbox yazılımında olduğu gibi, yapay sinir ağları için geliştirilmiş olan bu yazılım da, yine MATLAB Simulink ortamı ile bütünleşik çalışma yaklaşımını desteklemektedir.

Yazılım eğitimsel açıdan incelendiğinde, sunduğu özellik ve işlevler bağlamında, oldukça geniş kapsamlı bir yapay sinir ağları uygulama ortamı sunduğu ifade edilebilmektedir. Yazılım ile birlikte gelen görsel grafik seçenekleri, kullanılan verilerin ve elde edilen sonuçların kolay bir şekilde gözlemlenmesine imkân sağlamaktadır. Yazılımın en önemli avantajı ise, bulanık mantıkta olduğu gibi, yapay sinir ağları için

de Simulink desteğinin var olmasıdır. Bu şekilde çok çeşitli kontrol problemleri ve uygulamaları, yapay sinir ağları tabanında, kolay ve hızlı bir şekilde geliştirilebilmektedir. Simulink ortamında kullanılan etkileşimli, görsel "blok" yapıları, ilgili kullanıcılara basit bir kullanım tecrübesi vaat etmektedir. Bahsedilen bütün bu avantajların yanında, ilgili yazılım için değinilebilecek tek dezavantaj, sunduğu teknik anlamdaki bazı özellik ve işlevlerin, daha çok ileri düzeyde kullanıcıları hedef alması sayılabilmektedir.

2.4.2 Neuro Solutions

Neuro Solutions, Neuro Dimension adlı bir yazılım firması tarafından geliştirilen, geniş kapsamlı bir uygulama alanına sahip, farklı özelliklere sahip bir yapay sinir ağları uygulama yazılımıdır. Yazılım bünyesinde yer alan çok sayıdaki simge, arayüz üzerinde yapay sinir ağlarına dayalı birçok farklı çalışmanın gerçekleştirilmesine ve düzenlenmesine imkân sağlamaktadır. Tasarımı yapılan bir yapay sinir ağı, arayüz üzerinde sunulan ilgili alan kapsamında, görsel elementler yardımıyla görüntülenmektedir. Geliştirilen yapay sinir ağı uygulamalarına bağlı olarak, yazılım bünyesinden farklı nitelikte grafikler ve görsel sonuçlar çıkarmak mümkündür. Neuro Solutions yazılımı, üç farklı tasarım sihirbazını da kolay yapay sinir ağı çalışmaları gerçekleştirilebilmesi için kullanıcılara sunmaktadır. Söz konusu sihirbazlara ek olarak, yazılım kapsamında sunulan yardımcı araçlar sayesinde, geliştirilmiş olan bir yapay sinir ağı yapısına ait çalışma bilgileri farklı formatlarda kaydedilebilmektedir (İnt.Kyn.11). Yazılım, birçok farklı yapay sinir ağı mimarisini desteklediği gibi ilgili öğrenme kurallarını ve algoritmaları da beraberinde getirmektedir. Bu bağlamda yazılımın, yapay sinir ağları ile alakalı geniş bir problem alanına uygulanabildiği gözlemlenmektedir.

Neuro Solutions yazılımı eğitimsel açıdan ele alındığında, sunduğu renkli, görsel özellik ve işlevlerin, kullanıcıların daha kolay ve etkili bir kullanım tecrübesi yaşamasına imkân verdiği değerlendirilebilmektedir. Yazılımın sunduğu simge tabanlı kullanım özellikleri ve işlevlerini de bu bağlamda değerlendirmek mümkündür. Ayrıca, daha önce bahsedilen tasarım sihirbazları da, yazılım ortamında hızlı ve kolay bir şekilde

çeşitli uygulamalar geliştirilmesine imkân vermektedir. Bu kapsam dâhilinde yazılım, kullanıcılara ellerinde bulunan veri dosyalarını kullanarak, sunulan ağ mimarilerinden birisini seçerek ya da seçtikleri problem alanlarını tanımlayarak, yeni yapay sinir ağları sistemlerini kolaylıkla oluşturabilmelerini sağlamaktadır. Diğer yandan, yine ifade edildiği üzere, geliştirilmiş olan yapay sinir ağları sistemleri ile ilgili çalışma bilgilerinin, farklı platformlarda değerlendirilmek üzere, farklı formatlarda kaydedilebilmesi de yazılımın bir başka avantajıdır. Neuro Solutions yazılımı ile ilgili değinilebilecek bir dezavantaj ise, yazılımın teknik anlamda belirli bir düzeye kadar yapay sinir ağları sistemleri ve dolayısıyla uygulamaları geliştirilmesine olanak vermesi, ileri seviyedeki çalışmalar için yetersiz kalmasıdır.

2.4.3 Statistica – Neural Networks

Statistica yazılımı, Statsoft firması tarafından geliştirilmiş olan, adından da anlaşılacağı üzere, istatistiksel çalışmaların ve uygulamaların gerçekleştirilebilmesine imkân sağlayan, gelişmiş kullanım özellikleri ve işlevlerine sahip bir yazılım platformudur. MATLAB yazılımı gibi, birçok eklenti yazılımını da bünyesinde toplayan Statistica yazılımı, yapay sinir ağları uygulamaları için de çözüm sunmaktadır. İlgili yapay sinir ağları yazılımı geniş çapta bir yapay sinir ağı mimarisini ve bu mimarilere bağlı öğrenme kurallarını kapsamaktadır. Buna bağlı olarak da birçok farklı problem türünün çözümünde teknik yapıda, gelişmiş bir uygulama yazılımı ortamı ortaya koymaktadır (İnt.Kyn.12). Yazılım özellikle Statistica uygulama yazılımının istatistiksel anlamdaki güçlü özellik ve işlevlerinden yararlanmakta, bu anlamda çeşitli görsel grafik özelliklerini ve kullanım işlevlerini de ilgili yazılım arayüzünde sunmaktadır.

Eğitimsel açıdan incelendiğinde, geniş kapsamda bir uygulama ve problem alanının hedef alınması, yazılım açısından bir avantaj olarak değerlendirilebilmektedir. Ayrıca yazılımın Statistica gibi gelişmiş bir yazılımın altyapısını kullanması da, teknik açıdan etkili bir uygulama yazılımının ortaya konulmasını da sağlamaktadır. Ayrıca yazılım bünyesinde sunulan, çok çeşitli renkli grafikler de kullanıcılara eldeki veriler ya da elde edilen sonuçlar üzerinde etkili gözlem yapma imkânı de vermektedir. Ancak bütün bu olumlu özellik ve işlevlere rağmen, yazılımın teknik kullanım özelliklerinin ön planda

olması ve bu kapsamda daha çok ileri düzeyde kullanıcılara hitap etmesi, eğitimsel açıdan bir dezavantaj olarak ifade edilebilmektedir.

2.4.4 Mathematica – Neural Networks

Mathematica yazılımı, Wolfram Research firması tarafından geliştirilmiş olan, ileri düzeyde matematiksel uygulama geliştirme ve hesaplama yapma özellik ile işlevlerini sunan, popüler bir yazılım platformudur (İnt.Kyn.13). MATLAB ve Statistica yazılımları gibi, Mathematica yazılımı da birçok farklı eklenti yazılımını beraberinde getirmektedir. Nitekim yazılımın bünyesinde yer alan eklenti yazılımlardan birisi de yapay sinir ağları sistemleri tasarlanması ve geliştirilmesi üzerine kurulu olarak geliştirilmiş bir uygulama yazılımıdır. Söz konusu yazılım incelendiğinde, birçok farklı yapay sinir ağları mimarisini ve ilgili öğrenme kurallarını, algoritmalarını desteklediği gözlemlenmektedir (İnt.Kyn.13). Yazılımın bu bağlamda basit ve hızlı bir kullanım tecrübesi vaat ettiği ifade edilebilmektedir. Bu özelliklere ek olarak, Mathematica yazılımının güçlü altyapısı, yapay sinir ağları uygulama platformunda gerçekleştirilebilecek işlem kapsamını da önemli ölçüde genişletmektedir.

Yazılım eğitimsel açıdan ele alındığında, sunduğu geniş kapsamlı yapay sinir ağı mimarileri ve ilgili öğrenme yaklaşımları, yazılımın avantajları arasında sayılabilmektedir. Bunun yanı sıra, Mathematica yazılımının güçlü altyapısı da, bir önceki paragrafta da ifade edildiği gibi, dolaylı yönden de olsa yapay sinir ağları yazılımının işlem kapsamını genişletmektedir. Ayrıca yazılım, gerçekleştirilen işlemlerin sonuçlarının ve ilgili verilerin gözlemlenebildiği, görsel araçları da kullanıcılara sunmaktadır. Ancak bütün bu avantajlarının yanında, –Mathematica yazılımının geniş kapsamlı, teknik özelliklerinden olsa gerek– yapay sinir ağları üzerine gerçekleştirilen uygulamalarda komut yapılarının kullanılması ve söz konusu uygulama işlemlerinin, benzeri uygulama yazılımlarına göre daha manüel yöntemlerle gerçekleştirilmesi, ilgili yazılıma eğitimsel açıdan çeşitli dezavantajlar kazandırmaktadır. Bu bağlamda ele alındığında, Mathematica – Neural Networks yazılımın daha çok ileri düzeydeki kullanıcılara hitap ettiğini söylemek yerinde bir değerlendirme olacaktır.

2.4.5 Netmaker

Netmaker, bünyesinde etkili özellik ve işlevler barındıran, görsel yönden oldukça güçlü bir yapıda çalışma ortamı sunan, ücretsiz nitelikte bir uygulama yazılımıdır. Yazılım tamamen C# programlama dili aracılığıyla tasarlanmış ve yazılmıştır. Netmaker ilk aşamada CERN'de gerçekleştirilen parçacık etkileşimi ve nötrino fizik deneylerine destek sağlamak amacıyla geliştirilmiş, ancak zaman içerisinde geniş kapsamlı, esnek yapıda bir yazılım platformu haline dönüşmüştür (İnt.Kyn.14). Söz konusu yazılım, benzer uygulama yazılımlarına göre daha büyük miktardaki verilerle çalışmaya imkân vermekte, bu kapsam dâhilinde de hızlı ve kolay bir kullanım tecrübesi sunmaktadır. Yazılım bünyesinde geliştirilen sistemlerle alakalı elde edilen veriler, farklı türde grafiklerle incelenebilmekte ve değerlendirilebilmektedir. Kullanıcılar yazılım bünyesinde sunulan özellik ve işlevler yardımıyla başta çok katmanlı ağ mimarileri olmak üzere, farklı yapıda birçok ağ mimarisinin kullanılıp düzenlenmesine imkân vermektedir. Sunulan öğrenme kuralları açısından incelendiğinde, yaygın bir şekilde kullanılan algoritmalar olduğu kadar, daha çok spesifik uygulamalara yönelik olarak geliştirilen, birçok farklı algoritma yapısı da Netmaker tarafından sunulmaktadır. Netmaker yazılımı, özellikle sınıflandırma uygulamalarında etkili bir kullanım tecrübesi vaat etmektedir.

Netmaker eğitimsel açıdan incelendiğinde, sunduğu etkileşimli özellik ve işlevler sayesinde basit ve etkili bir kullanım şekli sunmaktadır. Yazılım bünyesinde sunulan renkli araçlar ve sürükle-bırak gibi kolay kullanım işlevleri, Netmaker'ın benzeri uygulama yazılımlarına göre bir adım önde olmasını sağlamaktadır. Yazılım ayrıca farklı araçları yeni pencere yapıları içerisinde sunduğundan, Windows ortamına aşina kullanıcılara daha kolay bir kullanım tecrübesini de beraberinde getirmektedir. Ancak bütün bu kullanım kolaylıkları, yazılımın benzerlerine göre biraz daha ileri seviyede yapay sinir ağları çalışmalarını desteklemesi ve bu bağlamda kullanıcılardan bir miktar da ön bilgi talep etmesi gerçeğini değiştirmemektedir. Yine de yazılımın özellikleri ve işlevleri, desteklenen uygulamalar ve çalışmalar dışında incelendiğinde, birçok farklı avantajdan da söz etmek mümkün olmaktadır. Yazılım kapsamında sunulan çok çeşitli grafik seçenekleri, kullanıcıların tasarlayıp geliştirdikleri yapay sinir ağları sistemleri

hakkında daha fazla bilgi sahibi olmalarına imkân vermektedir. Yazılım, gerçekleştirilecek uygulamaların karmaşıklığına göre artan bir düzeyde kullanım zorluğu ortaya çıkarmaktadır. Ancak sunulan özellik ve işlevler, Netmaker yazılımının, benzer nitelikteki yazılımlara göre daha kolay bir şekilde kullanılmasına imkân sağlamaktadır.

2.4.6 JOONE

JOONE yazılımı, The Joone Team tarafından, Java ortamında tasarlanmış ve geliştirilmiş olan, açık kaynak kodlu bir uygulama yazılımıdır (İnt.Kyn.15). Yazılım, Java ortamında tasarlanmış olması nedeniyle etkili görsel özellik ve işlevler sunmaktadır. Yazılım yapısı temelde bir kütüphane yazılımı olarak programlanmış olmasına rağmen, görsel bir arayüz üzerinde kullanıcılara sunulmaktadır. Arayüzde sunulan araç kutusunu kullanarak, kolaylıkla yapay sinir ağı sistemleri tasarlanabilmekte ve hazır bir sistem üzerinde ilgili uygulama işlemlerini kolaylıkla gerçekleştirebilmektedir. Tasarlanan bir ağ sistemi, ilgili alan içerisinde görsel olarak görüntülenmekte, kullanıcılar söz konusu bu alanı kullanarak tasarlanmış olan bir yapay sinir ağını kolaylıkla düzenleyebilmektedir. Yazılımın kaynak kod yapısı, sunulan özellik ve işlevlerin, çeşitli modül yapıları aracılığıyla geliştirilmesine imkan vermektedir. Bu yüzden, yazılım standart olarak uyarlamalı (adaptive) sistemler ve geri yayılım algoritması gibi öğrenme yaklaşımlarına destek vermesine rağmen, söz konusu özellik ve işlevler kolaylıkla geliştirilebilmektedir.

Yazılım eğitimsel açıdan incelendiğinde, sunulan görsel nitelikteki kullanım özellikleri ve işlevlerin, kullanıcıya basit ve renkli bir kullanım tecrübesi vaat ettiği değerlendirilmektedir. Ancak yazılımın Java ortamında düzenlendiği hesaba katıldığında, görsel özelliklerin yoğunluğu ve etkinliğinin daha fazla olacak bir şekilde, yazılım bünyesinde sunulabileceği de dikkat çekmektedir. Her ne kadar yeni modüller yardımıyla yazılımın özellik ve işlevleri geliştirilebilse de, standart olarak desteklenen yapay sinir ağı model ve yaklaşımlarının, benzer uygulama yazılımlarına göre daha az miktarda olduğu ifade edilebilmektedir.

2.4.7 Fuzzy COPE

Fuzzy COPE yazılımı, farklı yapay sinir ağları yapıları üzerinde çalışmaya imkân veren, ücretsiz olarak geliştirilmiş bir uygulama yazılımıdır (Watts *et al.* 1999). Söz konusu yazılımın ilerleyen zaman içerisinde geliştirileceği ve yenileneceği belirtilmesine rağmen, Internet ortamında yeni sürümleri bulunamamaktadır. Fuzzy COPE, ilk olarak MS-DOS tabanlı olarak geliştirilmiş, daha sonraları Windows sürümleri de ilgili geliştiriciler tarafından sunulmuştur. Fuzzy COPE yazılımının Windows sürümü basit bir arayüz ile birlikte bazı görsel özellik ve işlevlerle desteklenmiş çeşitli araçları kullanıcılara sunmaktadır. Yazılım çok katmanlı ağların, Kohonen'in kendi kendine organize ağlarının ve sinirsel-bulanık sistemlerin geliştirilmesine olanak sağlamaktadır (Watts *et al.* 1999). Yazılım Windows tabanlı bir sürüm sunmasına rağmen, uygulamaların oluşturulması, düzenlenmesi, test edilmesi gibi aşamalarda komut satırı kullanımını desteklemektedir. Bu noktada kullanıcı, yazılım bünyesinde kullanılabilen bazı özel komutları ilgili işlemleri yerine getirmek amacıyla kullanabilmektedir.

Fuzzy COPE eğitimsel açıdan incelendiğinde, basit bir arayüz ve görsel özellik ya da işlevlerle desteklenmiş bir yapı sunduğu gözlemlenmektedir. Ancak yazılımın en önemli dezavantajı, kullanıcılara komut satırı üzerinden kullanım imkânı sunmasıdır. Komut satırı kullanımı, yazılım Windows tabanlı bir arayüz sunmasına rağmen, kullanıcılar için fazladan bir zorluk ortaya çıkarmaktadır.

2.4.8 Kütüphane Niteliğindeki Yazılımlar

Bulanık mantık tekniğinde olduğu gibi, yapay sinir ağları tekniği için de, farklı platformlarda geliştirilmiş olan, çeşitli yapay sinir ağları kütüphane yazılımları mevcuttur. Yine bu yazılımların büyük bir kısmı ücretsiz ve / veya açık kaynak kodlu olarak kullanıcılara sunulmaktadır. Bu bağlamda, günümüzde popüler olarak değerlendirilen kütüphane yazılımları AForge.Net, Encog ve Fast Artificial Neural Network Library (FANN) isimli kütüphane yazılımlarıdır. AForge.Net, bulanık mantık kütüphane yazılımları başlığı altında da değinildiği gibi, içerisinde yapay sinir ağlarının da bulunduğu, geniş bir alanı kapsayan kütüphane yazılımıdır. Encog ise, .NET

platformunun yanı sıra, Java ve Silverlight gibi farklı platformlarda da kullanımı destekleyen, geniş kullanım özelliklerine sahip yapay sinir ağları kütüphane yazılımıdır (İnt.Kyn.16). Diğer yandan FANN yazılımı ise, C programlama dili yardımıyla tasarlanmış ve geliştirilmiş, yaygın bir kullanıcı ve geliştirici ağı olan, popüler nitelikte bir kütüphane yazılımıdır (İnt.Kyn.17). Söz konusu FANN kütüphane yazılımı için, geliştiriciler tarafından birçok yeni arayüz tasarlanmakta ve diğer kullanıcılarla paylaşılmaktadır. Ayrıca eksik özelliklerini ve işlevlerini gidermek adına, ilgili kütüphane yazılımı için birçok eklenti de geliştirilmektedir. Bulanık mantık tekniği kapsamında değinilen kütüphane yazılımlarında olduğu gibi, yapay sinir ağları tekniği için geliştirilmiş olan kütüphane yazılımlarını da eğitimsel açıdan ele almak, belirli yapılarda kullanıcı arayüzleri sunmadıklarından dolayı mümkün olmamaktadır. FANN kütüphane yazılımı da arayüz ve eklenti kapsamında, belirli bir süreklilik ve standart sağlamadığı için, eğitimsel açıdan ele alınmamaktadır.

İncelemesi yapılan bulanık mantık ve yapay sinir ağları uygulama yazılımları, bu çalışma kapsamında tasarlanan ve geliştirilen uygulama – eğitim yazılımının başlıca özellik ve işlevlerini belirleme aşamasında oldukça etkili olmuştur. Geliştirilen yazılım bünyesinde, incelenen yazılımların farklı avantajları bir araya getirilmeye çalışılmış, eksik olduğu düşünülen çeşitli özellik ve işlevler de yine ilgili yazılıma eklenmiştir. Geliştirilen yazılımın kullanım özellikleri ve işlevleri, daha detaylı bir şekilde, 4. Bölüm'de ele alınacaktır. Ancak öncelikle, yazılımın tasarlanması ve geliştirilmesi safhasında yararlanılan başlıca materyal ve metotlara değinmek gerekmektedir.

3. MATERYAL VE METOT

Bu çalışma kapsamında geliştirilen yazılım, etkili bir uygulama ortamı sunmak amacıyla, çeşitli programlama ve tasarım yaklaşımları dikkate alınarak oluşturulmuştur. Bu bağlamda, günümüzde özellikle yazılım – proje geliştirme çalışmalarında yaygın bir şekilde kullanılan, popüler yazılım geliştirme ortamlarından ve araçlarından yararlanılmıştır. Söz konusu yaklaşımların ve ilgili araçlarının incelenmesi, geliştirilen yazılımın kullanım özellikleri ve işlevleri hakkında ön bilgi sahibi olmak adına oldukça önemlidir.

3.1 Programlama ve Tasarım Faktörleri

Bulanık mantık ve yapay sinir ağları tekniklerinin yapı taşlarını oluşturan özellik ve işlevler, bu tekniklerin bilgisayar ortamında gerçekleştirilmesi aşamasında, kullanım yönünden esnek ve hızlı bir programlama dilini gerekli kılmaktadır. Bu amaçla kullanılabilecek programlama dili, kuşkusuz nesneye yönelik programlama tekniklerini ve yaklaşımlarını benimseyen, gelişmiş özelliklere sahip bir dil olmak zorundadır. Günümüzde yaygın olarak kullanılan programlama dilleri bu kapsam dâhilinde incelendikten sonra, çalışmanın istek ve gerekliliklerine uygun, en ideal programlama dili olarak, C# programlama dili olarak seçilmiştir.

3.1.1 C# Programlama Dili

C# (C Sharp) programlama dili, Microsoft firması tarafından geliştirilmiş olan, geniş kullanım alanına sahip, yeni nesil bir programlama dilidir. Günümüzde, yine Microsoft tarafından geliştirilmiş olan .NET (Dot Net) teknolojisi ile birlikte bütünleşik olarak kullanılabilmektedir (İnt.Kyn.18, İnt.Kyn.19). C# programlama dilinin temelleri ANSI C ve C++ (C plus plus) programlama dillerine dayanmaktadır. Ancak geliştirilme safhasında, Java, Eiffel, Modula-3 ve Object Pascal gibi programlama dillerinden de etkilenmiştir (Wong 2002, Naugler 2007, Hejlsberg 2008). Dilin tasarlanması ve geliştirilmesine, yine Delphi, Pascal ve J++ gibi programlama dillerinin

geliştirilmesinde söz sahibi olmuş olan Anders Hejlsberg öncülük etmiştir (İnt.Kyn.18, İnt.Kyn.19).

C# programlama dilinin en büyük özelliği, nesneye yönelik bir programlama dili olmasıdır. Buna ek olarak, basit, modern, esnek, bileşen yönelimli ve kolayca geliştirilebilir, etkili bir programlama yapısı da ortaya koymaktadır (ECMA 2006). Genel anlamda C# programlama dilinin, programlama yöntem ve yaklaşımları açısından sunduğu kolaylık ve yenilikleri şu şekilde özetlemek mümkündür (İnt.Kyn.18, İnt.Kyn.19):

- C# programlama dili, nesneye yönelik programlama teknikleri ve yaklaşımlarını bünyesinde toplamaktadır.
- C# programlama dili ile yazılımlarda daha fazla güvenlik ve daha gelişmiş hata kontrol yaklaşımları ortaya konmaktadır.
- C# programlama dili yapısı, en basitinden en karmaşığına kadar her türlü programın yazımında kolaylıkla kullanılabilen, esnek bir yaklaşım sergilemektedir.
- C# programlama dili, dünya çapında oldukça geniş bir destek ve geliştirme ağına sahiptir. Bu bağlamda, Microsoft yazılım şirketinin de etkili girişimleri bulunmaktadır.
- C# programlama dili, performans açısından C ve Assembly programlama dilleriyle rekabet edecek düzeyde geliştirilmiştir.
- Sunulan kullanım özellikleri ve işlevleri açısından incelendiği takdirde, C# programlama dili hızlı bir yazılım tasarlama ve geliştirme platformu sunmaktadır.

C# programlama dili, bahsi geçen kolaylıklar ve etkili kullanım özellikleri sayesinde, bu çalışmanın amaçlarına uygun, güçlü bir yapıda uygulama ve eğitim yazılımı geliştirilmesine olanak tanımıştır. Geliştirilen yazılımın program yapısı oluşturulurken, yine Microsoft yazılım şirketinin bir ürünü olan ve C# programlama dili ile birlikte birçok farklı dili bünyesinde barındıran, Visual Studio 2008 adlı yazılım geliştirme ortamından yararlanılmıştır.

3.1.2 Diğer Programlama ve Tasarım Faktörleri

Bulanık mantık ve yapay sinir ağlarına dayalı çalışmalarda kullanılacak olan bir uygulama ve eğitim yazılımının, teknik anlamda bazı önemli işlevleri yerine getiren, birtakım temel özellikleri de beraberinde getirmesi gerekmektedir. Bu bağlamda, geliştirilen yazılım içerisinde, gerek bulanık mantık, gerekse yapay sinir ağları kapsamında gerçekleştirilen çalışmaların sonuçlarını daha iyi değerlendirebilmek adına, elde edilen sonuç verilerini kullanıcıya daha anlaşılır formatta sunan, grafiklerden yararlanılmıştır. Bu amaçla, Visual Studio 2008 ortamına kolayca eklenebilen ve C# programlama dili ile kullanılabilen, etkili bir grafik düzenleme ve geliştirme eklentisi olan, Zed Graph isimli kütüphaneden yararlanılmıştır.

Zed Graph, .NET teknolojisi kapsamında kullanılmak üzere geliştirilmiş, detaylı grafik düzenleme ve geliştirme işlevleri sunan, açık kaynak kodlu bir kütüphane yazılımıdır (İnt.Kyn.20). Açık kaynak kodlu bir uygulama olması sebebiyle, özellikleri sürekli geliştirilen bir yazılım olarak piyasadaki yerini korumaktadır. Temelde C# programlama dili ile yazılmış olan Zed Graph kütüphanesi, çizgi grafik, çubuk grafik ve pasta grafik gibi birçok farklı grafik çeşidini, eldeki veriler ışığında, hızlı ve etkili bir şekilde, görsel arayüzlerde oluşturulabilmektedir. Bu kapsam dâhilinde, geliştirilen bulanık mantık ve yapay sinir ağları eğitim yazılımındaki sonuç / çıkış grafikleri, Zed Graph tarafından sunulan ilgili kontroller yardımıyla, görsel ve etkileşimli bir şekilde sunulmuştur.

Geliştirilen uygulama ve eğitim yazılımı bünyesinde, gerçekleştirilen aktif çalışmaların bilgisayar ortamına kaydedilmesi veya önceden gerçekleştirilmiş / kaydedilmiş çalışmaların tekrar yazılım arayüzüne çağırılması gibi çeşitli işlevler de bulunmaktadır. Söz konusu çalışmalar, XML formatındaki dosyalar olarak kaydedilebilmekte ve tekrar çağırılabilmektedir. XML, dünya çapındaki web standartlarını belirleyen, bağımsız bir kuruluş olan W3C tarafından tasarlanan bir meta dildir. Meta dil özelliği sayesinde, XML aracılığıyla diğer dilleri açıklamak mümkün olmaktadır. XML, görünüş açısından HTML ile benzerlik gösterse de, temel anlamda bilgiyi biçimlendirmekten çok, onu tanımlamak gibi işlevlerle uğraşmaktadır (Stanek 2002). XML sayesinde, veri saklamanın yanında, farklı sistemler arasında veri alışverişi yapma imkânına da sahip

olunmaktadır (İnt.Kyn.21). XML dosyaları, yapı itibariyle ağaç yapısı (tree view) şeklinde oluşturulmaktadır. Bu bağlamda, farklı nitelikteki her veri, belirli etiket (tag) elemanları arasında belirtilmektedir. Çalışma kapsamında geliştirilen yazılımdaki kontroller yardımıyla da, aktif durumdaki bulanık mantık ve yapay sinir ağları çalışmalarının temel değer ve parametreleri, yazılımdan bağımsız olarak okunabilen, basit yapıda XML dosyaları içerisinde saklanmaktadır.

Daha önce de belirtildiği gibi, bulanık mantık ve yapay sinir ağlarının temelini oluşturan özellik ve işlevler, bilgisayar ortamında, günümüzün popüler programlama yaklaşımı nesneye yönelik programlama yardımıyla daha etkili ve hızlı bir biçimde uygulanabilmektedir. Bu sebeple, geliştirilen yazılımın altyapısını oluşturmak amacıyla kullanılan metotları anlamak adına, nesneye yönelik programlama yaklaşımının temel özelliklerinden bahsetmek ve yine geliştirilen yazılımın, bu yaklaşım dâhilinde oluşturulmuş olan kaynak kod yapılarını kısaca incelemek, yerinde bir tespit olacaktır.

3.2 Nesneye Yönelik Programlama Yaklaşımı

Nesneye yönelik programlama (OOP), bilgisayar programları ve geniş kapsamlı uygulamaları yazılımlarının geliştirilmesinde, "nesne (object)" adı verilen yapılara dayalı ilke ve yöntemlere göre çalışan, modern bir programlama yaklaşımıdır. Söz konusu yaklaşım için "nesne yönelimli programlama" ifadesi de kullanılmaktadır. Günümüzün çağdaş programlama dilleri tarafından da desteklenen nesneye yönelik programlama yaklaşımı, yazılımların geliştirilmesi aşamasında karşılaşılan karmaşıklık ve maliyet gibi bazı sorunların çözülmesi amacıyla, 1960'lı yılların sonlarına doğru ortaya atılmış, ancak 90'lı yılların başlarına kadar yaygın bir kullanım alanına sahip olamamıştır (İnt.Kyn.22, İnt.Kyn.23). Java, C++ ve C#, nesneye yönelik yaklaşım çerçevesinde geliştirilmiş programlama dillerinden bazılarıdır.

Belirtildiği üzere, nesne adı verilen yapılar, nesneye yönelik programlama yaklaşımının ve dolayısıyla ilgili programlama dillerinin temel unsurları olarak kabul edilmektedir. Bu yaklaşım kapsamındaki her nesne, çözümlenmesi gereken problemle alakalı çeşitli varlıklara karşılık gelmektedir. Bu noktada, her nesnenin kendisine özgü bazı özellikler

(properties) ve yine kendisine adanmış olan çeşitli işlevler – eylemler (method) bulunmaktadır. Söz konusu bu faktörler yardımıyla nesneler arasında etkileşim sağlanabilmekte, bu etkileşimler neticesinde de geliştirilen yazılım kapsamındaki problemler çözümlenebilmektedir. Bunlara ek olarak, benzer nitelikteki nesneleri tanımlamak ve bir araya toplamak adına, sınıf (class) adı verilen yapılar da yine nesneye yönelik yaklaşımlar kapsamında kullanılmaktadır. Örneğin, nesneye yönelik yaklaşım kapsamında, "araba" isimli sınıf altında yer alan farklı markadaki her araba, kendine has özellik veya işlevlere sahip olabilen birer nesne olarak kabul edilmekte, diğer yandan ise, söz konusu her nesne, aynı sınıf (araba) altında toplandığından, bu sınıfın ortak özellikleri ve işlevlerini de bünyelerinde taşımaktadır. Bu yapılara ek olarak, nesneye yönelik yaklaşım kapsamında sıkça kullanılan, "kalıtım (inheritence)", "kapsülleme (encapsulation)" ve "çok biçimlilik (polymorphism)" gibi birçok farklı yöntem de etkili nesneye yönelik yaklaşım ve işlevleri olarak kabul edilmektedir. Kalıtım, bir sınıfın başka bir sınıftaki özellik ve işlevlere sahip olmasını, kapsülleme, bir sınıfın nesne ve işlevlerinin, yanlış kullanım veya karmaşıklığa karşı saklanmasını ve çok biçimlilik ise aynı isimli işlevlerin, benzer yapıda olsalar bile farklı görevleri yerine getirebilmesini tanımlamaktadır (Wirfs-Brock *et al.* 1990, Svenk 2003, Pillay 2007).

4. BULANIK MANTIK VE YAPAY SİNİR AĞLARI İÇİN EĞİTİM YAZILIMI GELİŞTİRİLMESİ

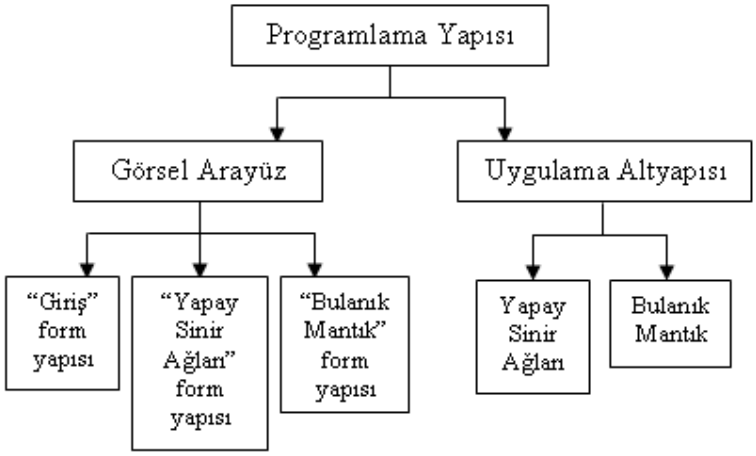
Bulanık mantık ve yapay sinir ağları tekniklerine dayalı çalışma ve uygulamaların gerçekleştirilmesine imkân tanıyan yazılım, bünyesinde taşıdığı özellik ve işlevlerle, daha önce incelemesi yapılmış benzer yazılımlara ek olarak, eğitimsel kaygılar da göz önüne alarak oluşturulmuş, yeni ve farklı bir uygulama – eğitim ortamı ortaya koymaktadır. Daha önce de belirtildiği üzere, geliştirilen yazılım, piyasada yer alan daha karmaşık ve teknik yapıdaki yazılımların önemli özelliklerini ve işlevlerini basit yapılarda sergilemekte, ek olarak etkili yeni özellik ve işlevler sunmakta, bütün bunların sonucu olarak da, bulanık mantık ve yapay sinir ağları teknikleriyle alakalı prensiplerin ve uygulama yapılarının daha kolay bir şekilde öğretilmesini sağlamaktadır. Yine geliştirilen yazılım, bulanık mantık ve yapay sinir ağları ile ilgili temel uygulamaların tasarlanmasına ve kullanılmasına olduğu kadar, daha teknik düzeyde çalışmaların tasarlanmasına ve gerçekleştirilmesine de olanak tanımaktadır. Bu açıdan bakıldığında geliştirilen yazılım, temel düzeyden ileri düzeye olmak üzere, geniş bir kullanıcı kitlesini kapsam içerisine almaktadır.

Geliştirilen yazılımın sağlıklı bir şekilde değerlendirilebilmesi için, temel kullanım özellikleri, işlevler ve ilgili arayüzler aracılığıyla, bulanık mantık ile yapay sinir ağları teknikleri kapsamında gerçekleştirilebilecek olan muhtemel çalışma şekillerinin incelenmesi gerekmektedir. Bu sayede ilgili yazılımın, çalışmanın amaçları yönünden etkinliği ve kapsam dâhilinde sağladığı katkılar daha iyi bir şekilde ifade edilecektir.

4.1 Kaynak Kod Özellikleri

Geliştirilen uygulama ve eğitim yazılımının programlama yapısı temel olarak iki kısım altında incelenebilmektedir. Bu noktada, yazılımın kullanıcı tarafında görüntülenen, görsel özelliklerle donatılmış, etkileşimli kısmı “Görsel Arayüz”, kullanılan bulanık mantık ve yapay sinir ağları tekniklerinin, bilgisayar ortamında uygulanabilmesi için gerekli olan özellik ve işlevlerini barındıran kısım ise “Uygulama Altyapısı” olarak adlandırılabilmektedir.

Görsel Arayüz kısmı altında, yazılımın görsel bütünlüğünü ve bu açıdan gerekli olan işlevselliğini sağlayan, bütün "form" yapıları bulunmaktadır. Söz konusu bu formları da, yazılımın giriş arayüzüne ait formlar, bulanık mantık platformuna ait formlar ve yapay sinir ağları platformuna ait formlar olmak üzere üçe ayırmak mümkündür. Bu yüzden, yazılımın görsel form tasarımları gerçekleştirilirken üç ayrı proje (solution) dosyası üzerinde çalışılmıştır. Bu şekilde, yazılımın tasarımı ve programlanması süreci de daha hızlı ve daha kolay bir şekilde tamamlanmıştır. Diğer yandan, Uygulama Altyapısı adı altındaki özellik ve işlevler de bulanık mantık ve yapay sinir ağları için ayrı ayrı olmak üzere iki proje dosyası altında programlanmıştır. Bu "altyapı" proje çalışmaları da daha sonra ilgili oldukları tekniğin proje dosyalarına eklenmiştir. Yazılım içerisindeki etkileşimi ve iletişimi sağlamak adına gerekli olan düzenlemelerin yapılması sonucunda da, uygulama yazılımı tek bir yapı haline getirilmiştir. Geliştirilen yazılımın programlama yapısı, genel hatlarıyla Şekil 4.1'de gösterilmektedir.



Şekil 4.1 Geliştirilen yazılımın programlama yapısı.

Bulanık mantık ve yapay sinir ağları için tasarlanmış ve geliştirilmiş olan özellik ve işlevler, her bir teknik için ayrı proje dosyaları içerisinde hazırlanmıştır. Buna göre, bulanık mantık altyapısını oluşturan proje dosyasında toplam 15 adet, yapay sinir ağları altyapısını oluşturan proje dosyasında ise toplam 22 adet kaynak kod dosyası (.cs) yer almaktadır. Yapay sinir ağlarında oluşturulan ağ sistemlerindeki yapay sinir hücrelerini, sinaptik bağlantıları ve oluşturulan katmanları birbirleriyle ilişkilendirmek amacıyla

daha fazla özellik ve işlev kullanılması, yapay sinir ağları altyapısı için hazırlanan kaynak kod dosyalarının, bulanık mantık için hazırlanan kaynak kod dosyalarına göre daha fazla sayıda olmasına neden olmuştur. Bu noktada belirtilmelidir ki, yapay sinir ağları için hazırlanmış olan söz konusu dosyalardan 7 tanesi arayüz (interface) niteliğinde dosyalar olarak oluşturulmuştur. Arayüzler, nesneye yönelik programlama kapsamında, karmaşık yapıda hazırlanmış kodlar üzerinde çalışmayı nispeten daha kolay hale getiren yapılar olarak kullanılmaktadır.

Bulanık mantık için oluşturulan altyapı kapsamında, yazılım arayüzünde oluşturulan her kontrol sistemi için gerekli olan giriş veya çıkış değişkenleri, üyelik fonksiyonları, dilsel ifadeler ve çıkarım yöntemleri gibi temel unsurlar, ayrı kaynak kod dosyaları içerisinde programlanmış ve gerekli işlevler doğrultusunda birbirleriyle ilişkilendirilmiştir. Ayrıca, çalışma dâhilinde geliştirilen bulanık mantık yazılımı Mamdani ve Takagi-Sugeno modellerini desteklediği için, ilgili modellere bağlı özellik ve işlevleri bünyelerinde barındıran, iki ayrı kaynak kod dosyası da programlanmıştır. Tasarlanan bir bulanık mantık sistemi, "FuzzySistem.cs" adlı dosyada tanımlanan yapılar yardımıyla kolaylıkla kontrol edilebilmektedir. Çizelge 4.1, bulanık mantık altyapısı için oluşturulan, önemli bazı kaynak kod dosyalarını, temel işlevleriyle birlikte kısaca tanıtmaktadır.

Yapay sinir ağları için oluşturulan altyapı kapsamında, geliştirilen yazılım içerisinde kullanılan çok katmanlı ileri beslemeli ağ mimarisine uygun bir biçimde, ilgili kaynak kod dosyaları oluşturulmuştur. Ayrıca, oluşturulan yapay sinir ağı sistemlerinin eğitimi amacıyla kullanılan geri yayılım algoritması için de ayrı kaynak kod dosyaları programlanmıştır. Tipik bir ağ sisteminin olmazsa olmazı olan katman yapıları, yapay sinir hücreleri ve sinaptik bağlantılar, ilgili özellik ve işlevleriyle birlikte yine ayrı kaynak kod dosyaları içerisinde oluşturulmuş ve birbirleriyle ilişkilendirilmiştir. Katmanlar için kullanılan çeşitli aktivasyon fonksiyonları da üç ayrı kaynak kod dosyası kapsamında düzenlenmiştir. Bulanık mantık altyapısında olduğu gibi, yapay sinir ağları altyapısında da, ağ sistemlerinin kullanımını kolaylaştırmak amacıyla, gerekli yapıların bir araya toplandığı, tek bir kaynak kod yapısı da ayrıca oluşturulmuştur. Çizelge 4.2, yapay sinir ağları altyapısı için oluşturulan, önemli bazı kaynak kod dosyalarını, temel işlevleriyle birlikte kısaca tanıtmaktadır.

Çizelge 4.1 Bulanık mantık için oluşturulan, bazı kaynak kod dosyaları.

Kaynak Kod Dosyası	**Temel İşlev**
FuzzySistem.cs	Bulanık mantık sistemlerinin oluşturulması, düzenlenmesi ve genel kontrolü.
MamdaniFuzzySistem.cs	Mamdani model bulanık mantık sistemlerinin düzenlenmesi ve kontrolü.
SugenoFuzzySistem.cs	Sugeno model bulanık mantık sistemlerinin düzenlenmesi ve kontrolü.
FuzzyDegisken.cs	Giriş ve çıkış değişkenlerinin oluşturulması ve düzenlenmesi.
UyelikFonksiyonu.cs	Üyelik fonksiyonlarının oluşturulması, düzenlenmesi ve bu fonksiyonlar kapsamındaki matematiksel işlevler.
FuzzyKural.cs	Tasarlanan bulanık mantık sistemleri için kuralların oluşturulması ve düzenlenmesi.
KuralAyristirici.cs	Form arayüzünde dilsel ifadelerle girilen kuralların, sistem tarafından ayrıştırılması ve analizi.
DurulamaYontemi.cs	Tasarlanan sistemler için kullanılan, farklı durulama yöntemleri kapsamındaki matematiksel işlevler.

Çizelge 4.2 Yapay sinir ağları için oluşturulan, bazı kaynak kod dosyaları.

Kaynak Kod Dosyası	**Temel İşlev**
Ag.cs	Ağ sistemlerinin oluşturulması, düzenlenmesi ve genel kontrolü.
Katman.cs	Giriş katmanı, çıkış katmanı ve gizli katmanların oluşturulması, düzenlenmesi ve kontrolü.
Noron.cs	Yapay sinir hücrelerinin düzenlenmesi ve kontrolü.
Baglayici.cs	Sinaptik bağlantıların oluşturulması, düzenlenmesi ve kontrolü.
GeriYayilimEgitim.cs	Geri yayılım algoritmasının işlem adımları, gerekli matematiksel işlevler.
EgitimSet.cs	Geri yayılım algoritması kapsamında, eğitim verilerinin değerlendirilmesi, düzenlenmesi ve kullanımı.
AktivasyonFKontrol.cs	Aktivasyon fonksiyonlarının genel kontrolü.

Bu çalışma kapsamında geliştirilen uygulama ve eğitim yazılımı, açıklanan materyal ve metotlar ışığında, bulanık mantık ve yapay sinir ağları tekniklerinin eğitimi ve bu tekniklere dayalı çalışmaların gerçekleştirilmesi hedeflerine uygun yapıda, etkili bir platform sunmaktadır.

4.2 Yazılımın Temel Kullanım Özellikleri ve İşlevleri

Geliştirilen yazılım, genel olarak kullanıcılara basit ve etkili bir kullanım tecrübesi vaat eden, birçok farklı kullanım özellikleri ve işlevleri ile donatılmıştır. Yazılımın görsel arayüzleri, Visual Studio 2008 ortamında yer alan, mümkün olan en basit yapıdaki "form kontrolleri" kullanılarak tasarlanmıştır. Bu bağlamda kullanılan kontroller, kullanıcıların genellikle Windows tabanlı yazılımlardan aşina oldukları kontroller olarak değerlendirilebilmektedir. Kullanıcıların yazılımı kolay bir şekilde kullanabilmesi için, bulanık mantık ve yapay sinir ağları ile ilgili uygulamalar üzerinde çalışırken ihtiyaç duyulan bütün işlevler, mümkün olan en az sayıda form penceresi üzerinde toplanmıştır. Böylece kullanıcılar, karmaşık bir kullanım yapısından ziyade daha sade ve basit bir yapıyla karşı karşıya kalmaktadır. Gerek bulanık mantık yazılım arayüzünde, gerekse yapay sinir ağları arayüzünde kullanılan benzer nitelikteki özellik ve işlevler, yine benzer yapıdaki form kontrolleri altında kullanıcılara sunulmuştur. Yine bu özellik ve işlevler, daha etkili kullanım tecrübeleri kazandırmak adına çeşitli görsel özelliklerle desteklenmiştir. Görsel anlamda, yazılım bünyesinde kullanılan renklerin birbirleriyle uyumlu ve kullanıcıyı rahatsız etmeyecek bir şekilde dağıtılmasına özen gösterilmiştir. Diğer yandan, kullanıcıların yerine getirdikleri işlevleri daha iyi anlamaları ve dolaylı yönden öğrenmeleri amacıyla, çeşitli etiket ve simge nesneleri kullanılarak, ilgili işlevlerin kısa ve öz tanıtımları yazılım arayüzleri bünyesinde yapılmıştır. Yazılım içerisinde kullanılan bazı işlevler, uygulama tasarlarken veya geliştirirken vakit alan bazı işlemlerin daha hızlı yapılmasını sağlamaktadır. Kullanıcılar, yazılımı fare ile olduğu kadar klavye ile de kolay bir şekilde kontrol edebilmektedir. Yazılımın sunduğu diğer bir önemli kullanım işlevi ise, kullanıcı kaynaklı ortaya çıkabilecek yazılım hatalarını en aza indirgemesidir. Bu bağlamda, kullanıcıların gerçekleştirdiği her işlem, yazılımın amacına uygun bir şekilde çalışmasını amaçlayan, çeşitli kısıtlama ve

denetimlerden geçirilmekte, böylece kullanıcılar, gerçekleşen ya da ileride gerçekleşebilecek olası herhangi bir hatalı işlem için, yazılım tarafından uyarılmaktadır.

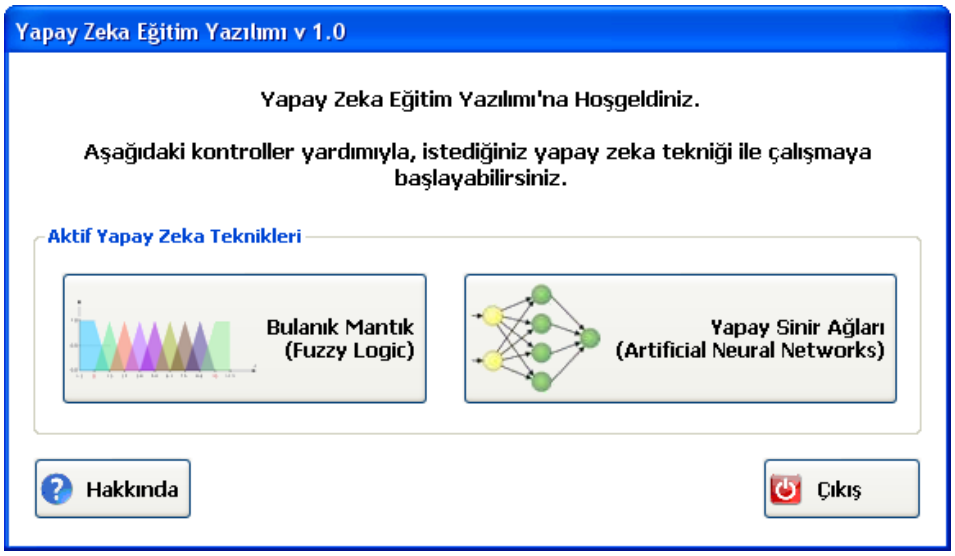
Bulanık mantık ve yapay sinir ağları teknikleriyle ilgili çalışma ve uygulamalar, bahsedilen temel kullanım özellikleri ve işlevleri ışığında, ilgili arayüzler kullanılarak kolaylıkla yerine getirilebilmektedir. Söz konusu çalışma ve uygulamalar ilerleyen bölümlerde detaylı bir şekilde ele alınacaktır. Anlaşılacağı üzere, yazılım kapsamında gerçekleştirilen çalışmalar, arayüzlerde kullanılan farklı form kontrolleri üzerinden sağlanmaktadır. Bu kapsamda, anlatım şeklini, programlama yaklaşımlarıyla ilgili kavramlardan soyutlamak ve daha anlaşılır bir düzeye taşımak için, yazılım kapsamında gözlemlenen bazı form kontrollerinin gerçek isimleri yerine daha genel ve Türkçe ifadeler kullanılacaktır. Bu amaçla, benzer görevdeki kontrollerin bir arada toplandığı form kontrolü olan "groupBox" yerine "panel" ifadesi, "label" kontrolü yerine "etiket" ifadesi, veri girişi için kullanılan "textBox" kontrolü yerine "veri giriş alanı" ifadesi ve "button" kontrolü yerine "düğme" ifadesinin kullanımı tercih edilmektedir. Yine ilgili yazılımlar bünyesinde sunulan farklı "alt arayüzler" için de standart "pencere" ifadesi kullanılmaktadır.

4.3 Yazılım Arayüzleri

Daha önce de belirtildiği gibi, geliştirilen yazılım, görsel tasarım ve programlama açısından üç farklı proje dosyası üzerinden oluşturulmuştur. Yazılım bu yönden incelendiğinde, üç farklı arayüzden bahsetmek mümkün olmaktadır. Bulanık mantık ve yapay sinir ağları ile ilgili arayüzler üzerinde gerçekleştirilebilecek çalışma şekillerinden bahsetmeden önce, ilgili arayüzlere kısaca değinmek, yazılımın görsel nitelikleri konusunda ön bilgi sahibi olmak adına önemlidir. Ancak bu noktada, bulanık mantık ve yapay sinir ağlarına ait "ilk" arayüzlere değinilecek, kullanılan diğer alt arayüzler ise, anlatımda akıcılığı sağlamak adına, ilgili tekniklerle alakalı çalışma şekillerinden bahsederken açıklanacaktır.

4.3.1 Giriş Arayüzü

Yazılımın çalıştırılması ile birlikte ekranda ilk görüntülenen arayüz "giriş arayüzü" olarak adlandırılabilmektedir. Söz konusu arayüz açılmadan hemen önce bir açılış görüntüsü de (splash screen) kısa süreli olarak ekrana getirilmektedir. Bu noktada giriş arayüzü, kullanıcıları yazılım bünyesindeki bulanık mantık ve yapay sinir ağları uygulama ortamlarına yönlendiren, bir nevi geçiş platformu özelliğindedir. İlgili arayüz üzerinde yer alan düğmeler yardımıyla kullanıcılar istedikleri yapay zekâ tekniğiyle bağlantılı yazılım arayüzünü kolaylıkla açabilmektedir. Yine bu arayüzde yer alan "Hakkında" düğmesi ile geliştirilen uygulama – eğitim yazılımı ile alakalı bazı kısa bilgilere ulaşılabilmektedir. Geliştirilen yazılımın giriş arayüzü, Resim 4.1 altında gösterilmektedir.

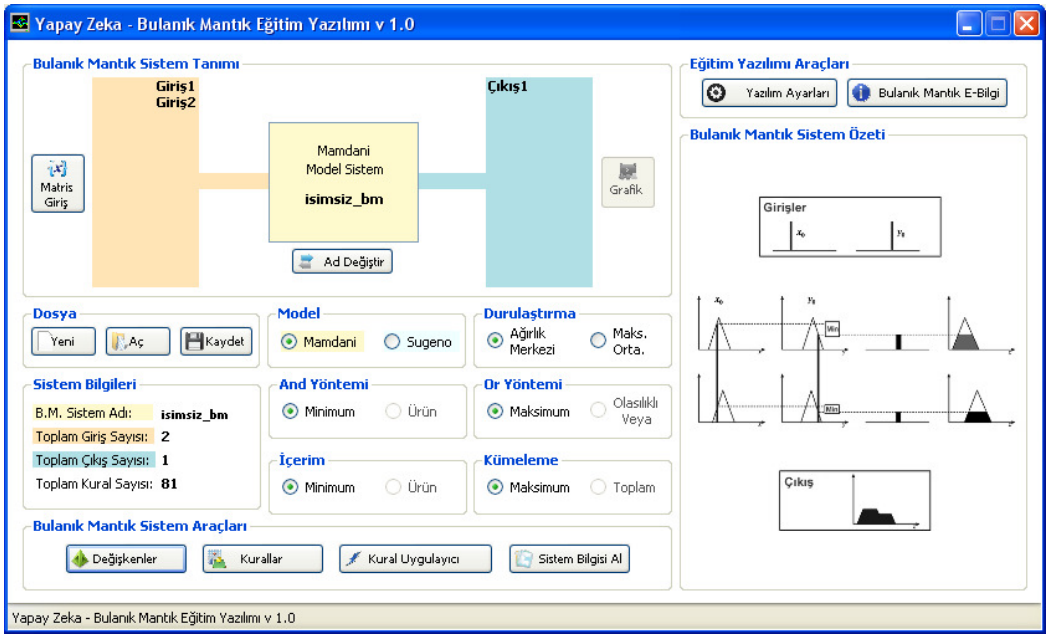


Resim 4.1 Geliştirilen yazılımın giriş arayüzü.

4.3.2 Bulanık Mantık Arayüzü

Giriş arayüzünde bulunan "Bulanık Mantık (Fuzzy Logic)" başlıklı düğmeye tıklayarak, bulanık mantık tekniğiyle alakalı çalışma ve uygulamaların gerçekleştirilebildiği ilgili arayüz ekrana getirilebilmektedir. Bu arayüzde, ortak nitelikteki özellik ve işlevler aynı

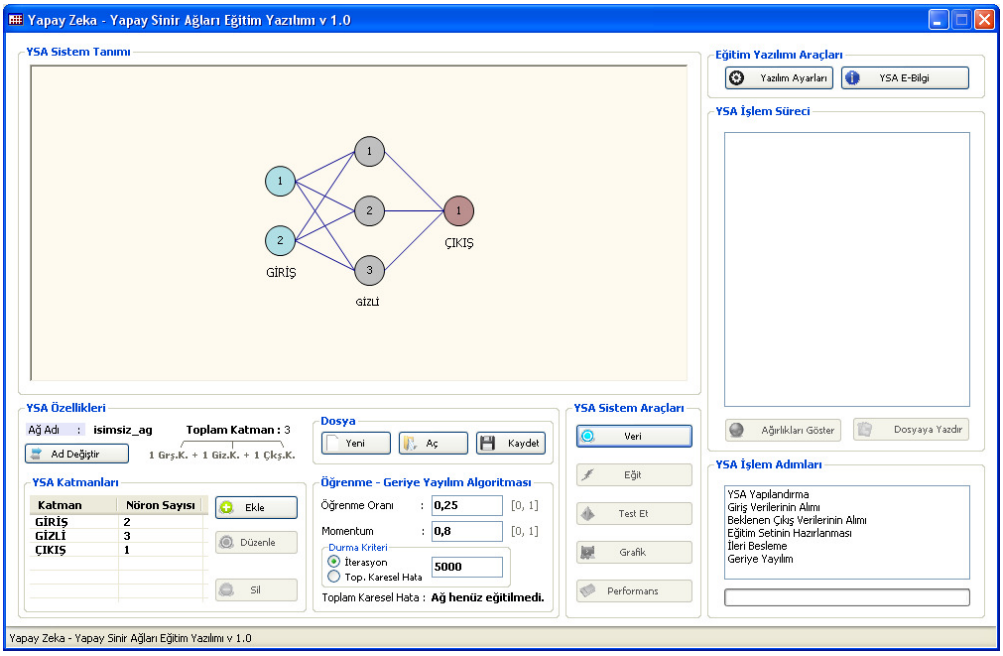
paneller altında toplanmış, sunulan kontroller ve görsel elementler, bulanık mantık tekniğiyle ilgili temel prensip ve uygulama yapılarını yansıtacak yapıda sunulmuştur. Söz konusu bulanık mantık arayüzü Resim 4.2'de gösterilmektedir.



Resim 4.2 Bulanık mantık arayüzü.

4.3.3 Yapay Sinir Ağları Arayüzü

Yapay sinir ağları ile alakalı çalışma ve uygulamaların gerçekleştirilebildiği arayüz, yine giriş arayüzünde sunulmuş olan, "Yapay Sinir Ağları (Artificial Neural Networks)" başlıklı düğmeye tıklayarak ekrana getirilebilmektedir. Yapay sinir ağları için hazırlanan arayüz de genel tasarım yönünden bulanık mantığın arayüzüne benzer niteliklerde hazırlanmıştır. Nitekim bu arayüzde de ortak nitelikteki özellik ve işlevler aynı paneller altında toplanmış, sunulan kontroller ve görsel elementler de, yapay sinir ağları tekniğiyle bağlantılı temel prensip ve uygulama yapılarını yansıtacak bir yapıda sunulmuştur. Yapay sinir ağları arayüzü Resim 4.3'de gösterilmektedir.



Resim 4.3 Yapay sinir ağları arayüzü.

4.4 Bulanık Mantık Uygulama ve Eğitim Yazılımı ile Çalışmak

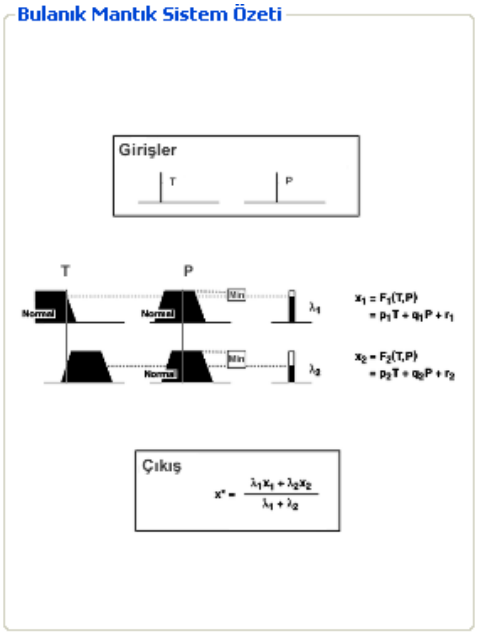
Bulanık mantık uygulama ve eğitim yazılımı, bünyesinde sunduğu, kullanımı kolay ve etkileşimli kontroller aracılığıyla, kullanıcılara basit yapıda bir kullanım tecrübesi vaat etmektedir. Yazılımın sade bir yapı içerisinde sunulması ve dolaylı yönden anlaşılabilirliğinin yüksek düzeyde olması amacıyla, söz konusu ortamda gerçekleştirilen çalışmalar, bazı kullanım şekillerinden ve görevlerinden oluşacak bir biçimde kullanıcılara sunulmuştur.

4.4.1 Bulanık Kontrol Sistemlerinin Tasarlanması ve Düzenlenmesi

Bulanık mantık yazılım arayüzünde sunulan çeşitli paneller aracılığıyla, ilgili ortam üzerinde bulanık kontrol sistemlerini tasarlamak ya da tasarlanmış durumda olan sistemleri düzenlemek, benzeri uygulama yazılımlarına göre oldukça kolay bir hale gelmektedir. Arayüzde yer alan "Dosya" başlıklı panel, bünyesinde sunduğu düğmeler yardımıyla, kullanıcının yazılım ortamında yeni bir çalışmanın açabilmesini ("Yeni"), aktif durumdaki çalışmayı XML dosya formatında kaydedebilmesini ("Kaydet") ya da

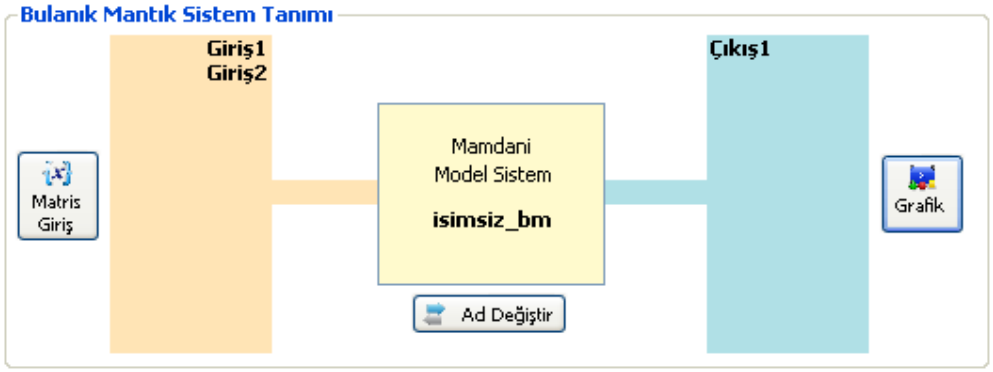
önceden üzerinde çalışılmış bir sistemi, ilgili yazılım sistemine yükleyebilmesini (“Aç”) sağlamaktadır. Yazılım arayüzünde aktif olarak yer alan bir bulanık kontrol sistemi, yine bu arayüzde sunulmuş olan diğer panellerin ve kontrollerin kullanımı sonucunda düzenlenebilmekte ve çalıştırılabilmektedir.

Daha önce de belirtildiği üzere, geliştirilen bulanık mantık uygulama ve eğitim yazılımı, hem Mamdani, hem de Takagi-Sugeno bulanık mantık modellerini desteklemektedir. Bu yüzden, ilgili modellerle alakalı sunulan farklı prensipler, yine bulanık mantık arayüzünde kullanıcılara tanıtılmaktadır. Bu bağlamda kullanıcı, “Model” paneli altındaki kontrolleri kullanarak, aktif sistemin modelini değiştirebilmekte, söz konusu model değiştiği anda da, “And Yöntemi”, “Or Yöntemi”, “İçerim” ve “Kümeleme” başlıklı paneller altından, seçili durumdaki modelin, sistem kapsamında kullandığı temel prensipler hakkında bilgi sahibi olmaktadır. Bu paneller arasında yer alan “Durulaştırma” başlıklı panel ise kullanılacak durulaştırma yönteminin belirlenmesini sağlamaktadır. Model seçimi gerçekleştiğinde, “Bulanık Mantık Sistem Özeti” paneli altında, seçili modelin çalışma şekli hakkında bilgi veren, özet nitelikte bir şema görüntülenmektedir. Resim 4.4, Takagi-Sugeno modeli ile ilgili şemayı göstermektedir.



Resim 4.4 Bulanık Mantık Sistem Özeti panelinde, Takagi-Sugeno model şeması.

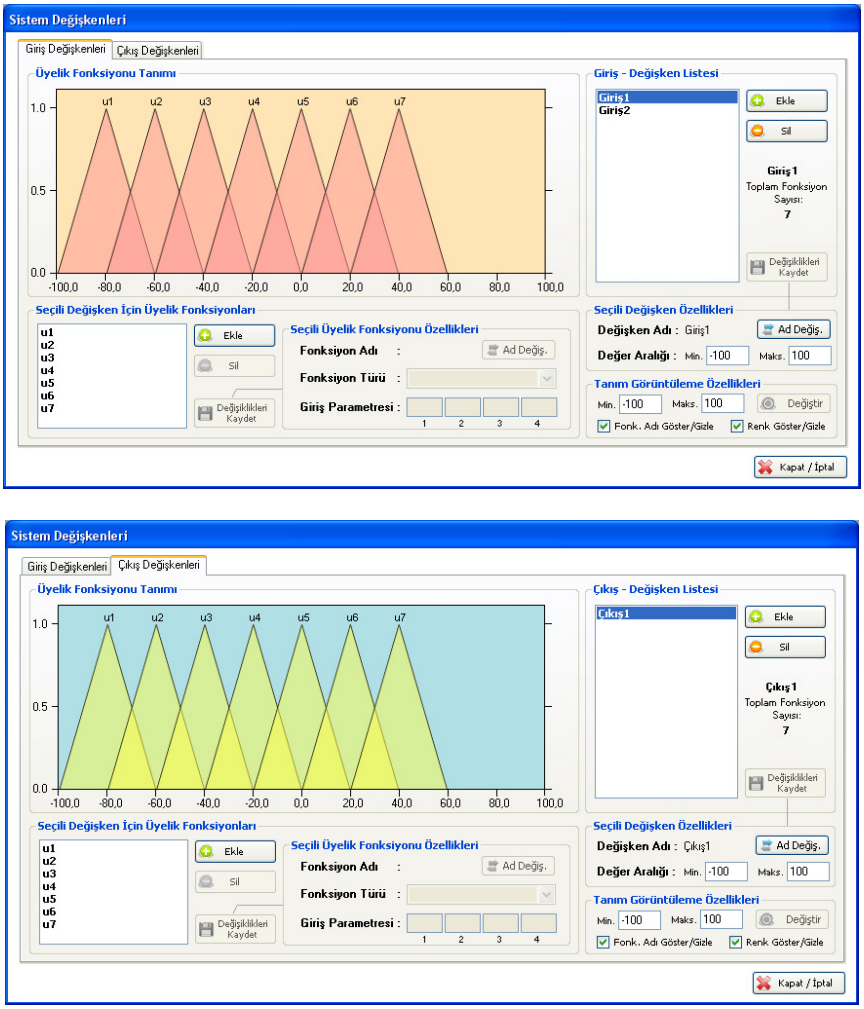
Bahsi geçen paneller arasında yer alan Bulanık mantık arayüzünde sunulan bir diğer panel olan "Sistem Bilgileri" paneli altında, aktif durumdaki sistemin (çalışmanın) adı, söz konusu sistem kapsamında tanımlanmış durumda olan giriş ve çıkış değişkenlerinin sayısı ve yine tanımlanmış olan toplam kural sayısı bilgileri, kullanıcılara sunulmaktadır. Arayüzde sunulan, en dikkat çekici panel ise "Bulanık Mantık Sistem Tanımı" başlığı altında sunulmuş olan, görsel modeldir. Söz konusu bu model, aktif durumdaki sistemin modeli ve bünyesinde taşıdığı giriş ile çıkış değişkenleri hakkında, görsel bir yapı içerisinde bilgi vermektedir. Modelin orta kısmında, aktif durumdaki sistemin (çalışmanın) adı ve yine bu sistem için seçilmiş olan model türü bilgileri görüntülenmektedir. İlgili kısım, seçili olan modelin türüne göre farklı bir renkte sergilenmektedir. Yine bu kısmın altında yer alan düğme kullanılarak, aktif sistemin (çalışmanın) adı değiştirilebilmektedir. Diğer yandan modelin solunda ve sağında sunulan sütunlar içerisinde, sistem için tanımlanmış olan giriş ile çıkış değişkenlerinin isimleri, listelenmiş bir şekilde görüntülenmektedir. Bahsedilen bu özelliklere ve işlevlere ek olarak, kullanıcıların bulanık kontrol sistemine matris formunda giriş yapmasını ve elde edilen grafikleri görüntülemesini sağlayan, iki adet düğme de yine "Bulanık Mantık Sistem Tanımı" paneli altında sunulmaktadır. Resim 4.5 altında, "Bulanık Mantık Sistem Tanımı" panelinden bir ekran görüntüsü sunulmuştur.



Resim 4.5 Bulanık Mantık Sistem Tanımı panelinden bir ekran görüntüsü.

Açıklanan özellik ve işlevler dışında, tipik bir bulanık kontrol sisteminin tasarlanması, düzenlemesi ve aynı zamanda kullanılması amacıyla sunulan, en önemli sistem paneli, yine yazılım arayüzünde sunulmakta olan, "Bulanık Mantık Sistem Araçları" başlıklı

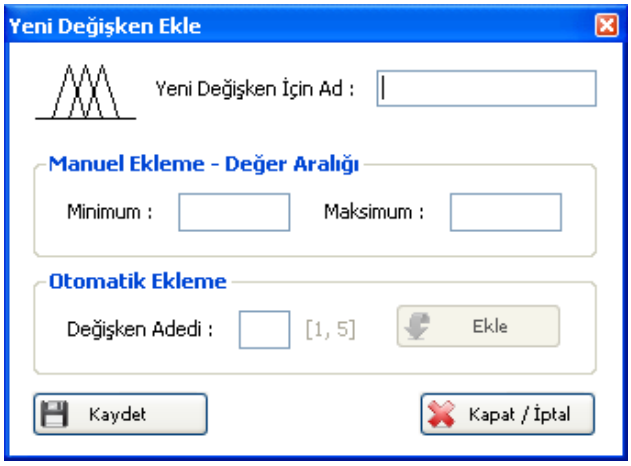
paneldir. Bu panel altında yer alan dört farklı düğme, aktif sistem ile ilgili farklı özellik ve işlevlerin yerine getirilmesini sağlamaktadır. Sırasıyla "Değişkenler", "Kurallar", "Kural Uygulayıcı" ve "Sistem Bilgisi Al" başlıkları altında sunulmuş olan bu düğmelerden birisi olan "Değişkenler" düğmesi, geliştirilen bulanık mantık yazılımının kalbini oluşturan ve değişkenler ile bağlı üyelik fonksiyonlarının tanımlanıp düzenlenmesine imkân veren, "Sistem Değişkenleri" penceresinin açılmasını sağlamaktadır. Sistem Değişkenleri penceresi, aktif durumdaki sistemin giriş ve çıkış değişkenlerinin tanımlanabilmesine imkân tanıyan, iki farklı sekmeden oluşmaktadır. "Sistem Değişkenleri" penceresi ve ilgili sekmelerden ekran görüntüleri, Resim 4.6 altında gösterilmiştir.



Resim 4.6 Sistem Değişkenleri penceresinden bir ekran görüntüsü.

"Sistem Değişkenleri" penceresi kapsamındaki sekmeler altında sunulan yapılar, birbirine benzer nitelikte olmakla beraber, giriş ile çıkış değişkenleri ayrımından dolayı, görsel ve dilsel yönden bazı ufak farklılıklar da taşımaktadır. İlgili sekmeler altında kullanıcılara sunulmuş olan beş farklı panel, gerekli düzenlemelerin yapılmasına imkân veren birçok kontrolü beraberinde getirmektedir.

Pencerede sunulmuş olan "Giriş – Değişken Listesi" ya da "Çıkış – Değişken Listesi" panelleri, sundukları liste kutusu kontrolleri ile aktif sistem içerisindeki giriş ya da çıkış değişkenlerinin bir listesini kullanıcılara göstermektedir. Bu liste kutuları kullanılarak, herhangi bir değişkene ait üyelik fonksiyonlarının görüntülenmesi sağlanabilmekte, değişkenle ilgili bazı parametreler ayarlanabilmekte ve yine sistemden kaldırılması istenen bir değişkenin silinmesi işlemi yerine getirilebilmektedir. Panel altında sunulmuş olan "Ekle" düğmesine tıklayarak, giriş ya da çıkış değişkeninin sisteme eklenmesini sağlayan, "Yeni Değişken Ekle" başlıklı pencere ekrana getirilebilmektedir. Söz konusu pencere Resim 4.7 altında gösterilmektedir.



Resim 4.7 Yeni değişken ekleme penceresi.

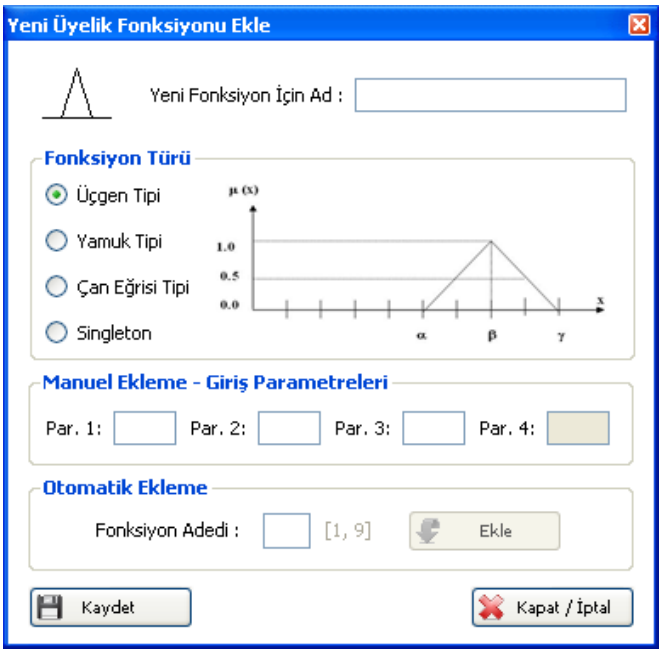
Yeni değişken ekleme penceresi üzerinde yer alan kontroller kullanılarak, eklenecek değişkenin adı ve değer aralığı (minimum ve maksimum değerleri) belirtilebilmekte ve "Kaydet" düğmesi aracılığıyla değişken ekleme işlemi sona erdirilmektedir. Söz konusu bu işlem, manuel yolla değişken ekleme olarak ifade edilebilmektedir. Bu noktada,

bulanık mantık uygulama ve eğitim yazılımın sunduğu önemli bir yenilik, otomatik olarak da değişkenlerin eklenmesine izin vermesidir. Söz konusu işlev, yeni değişken ekleme penceresi bünyesinde yer alan, "Otomatik Ekleme" paneli kullanılarak gerçekleştirilmektedir. Bu panelde yer alan, "Değişken Adedi" etiketli veri giriş alanına, 1 ile 5 arasında, eklenmesi istenen değişken adedi yazılmakta ve "Ekle" düğmesi kullanılarak işlem tamamlanmaktadır. Varsayılan olarak, eklenen yeni değişkenler, yazılım sistemi tarafından Giriş1, Giriş2…GirişN ya da Çıkış1, Çıkış2…ÇıkışN şeklinde adlandırılmakta, yine eklenen her değişken için değer aralığı, otomatik olarak [-100…100] aralığı içerisinde tanımlanmaktadır.

Yeni değişkenlerin eklenmesi ile birlikte, "Sistem Değişkenleri" penceresinde sunulmuş olan "Seçili Değişken İçin Üyelik Fonksiyonları" başlıklı panel de aktif duruma gelmektedir. Söz konusu bu panel de, değişken listesinden seçilmiş olan herhangi bir giriş ya da çıkış değişkeni için, tanımlanmış olan üyelik fonksiyonlarının bir listesini ve gerekli diğer düzenleme kontrollerini kullanıcılara sunmaktadır. Üyelik fonksiyonların listelenmesi, değişkenlere benzer olarak, yine liste kutusu kontrolü yardımıyla gerçekleştirilmektedir. Liste kutusunun yanında sunulan düğmeler yeni üyelik fonksiyonu eklemek, seçili olan bir üyelik fonksiyonunu silmek ya da seçili bir üyelik fonksiyonu ile alakalı yapılmış olan yeni düzenlemeleri kaydetmek amacıyla kullanılmaktadır. "Ekle" düğmesine tıkladıktan sonra, "Yeni Üyelik Fonksiyonu Ekle" başlıklı pencere ekrana otomatikman gelmektedir.

Yeni üyelik fonksiyonu ekleme penceresinde sunulan kontroller yardımıyla, seçili sistem değişkenine eklenmek istenen üyelik fonksiyonunun adı, türü ve türüne göre değişiklik gösteren parametre değerleri kullanıcı tarafından belirtilebilmektedir. Söz konusu değerler ayarlandıktan sonra "Kaydet" tuşu kullanılarak ekleme işlemi yerine getirilebilmektedir. Tıpkı sistem değişkenlerinin eklenmesinde olduğu gibi, bu pencere bünyesinde yer alan "Otomatik Ekleme" paneli kullanılarak da, istenilen sayıda ve istenilen türde üyelik fonksiyonunu seçili değişkene eklemek mümkün olmaktadır. Yazılım sistemi, otomatik olarak eklenen her yeni üyelik fonksiyonunu, seçili değişkende yer alan, parametre açısından en yüksek değerlikteki üyelik fonksiyonu ile karşılaştırmakta ve eklenecek yeni fonksiyonun parametrelerini ona göre

ayarlamaktadır. Otomatik ekleme işlemi, ilgili pencerede, eklenmesi istenen üyelik fonksiyonu ya da fonksiyonlarının türü seçildikten sonra, yine ilgili panel altında yer alan, "Fonksiyon Adedi" etiketli veri giriş alanına istenilen sayının belirtilip, "Ekle" düğmesi tıklanması suretiyle yerine getirilmektedir. Yeni üyelik fonksiyonu eklemek amacıyla kullanılan ilgili pencere, Resim 4.8 altında gösterilmektedir.



Resim 4.8 Yeni üyelik fonksiyonu ekleme penceresi.

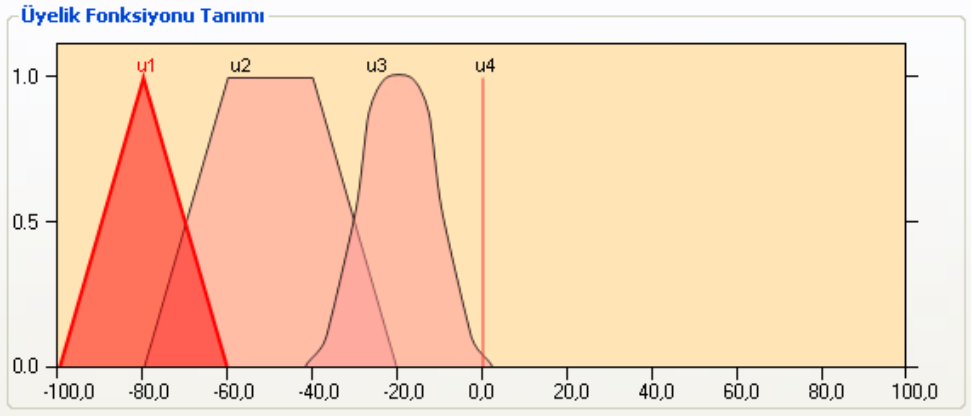
Seçili değişkene istenilen üyelik fonksiyonları eklendikten sonra, kullanıcı isterse tanımlı durumdaki bir üyelik fonksiyonunu seçip, ad, tür ve parametre değerleri gibi özelliklerini değiştirebilmektedir. Söz konusu işlev, üyelik fonksiyonu liste kutusundan herhangi bir fonksiyon seçildikten sonra aktif hale gelen ve "Seçili Değişken İçin Üyelik Fonksiyonları" başlıklı panel altında sunulmuş olan, "Seçili Üyelik Fonksiyonu Özellikleri" başlıklı panel kullanılarak yerine getirilebilmektedir. Bu paneldeki kontroller kullanılarak, seçili üyelik fonksiyonunun adı, türü ve türüne bağlı olarak da ilgili parametreleri kolaylıkla değiştirilebilmektedir. Yapılan değişiklikler, panelin hemen solunda yer alan, "Değişiklikleri Kaydet" düğmesiyle yazılım sistemine bildirilmektedir.

Resim 4.9, genel anlamda açıklaması yapılmış olan "Seçili Değişkenler İçin Üyelik Fonksiyonları" başlıklı paneli ve bünyesinde taşıdığı diğer kontrolleri göstermektedir.

Resim 4.9 Seçili değişken için üyelik fonksiyonlarının düzenlenebildiği panel.

Bu noktada, "Sistem Değişkenleri" penceresi kapsamında sunulmuş olan, önemli bir panelden söz etmek gerekmektedir. "Üyelik Fonksiyonu Tanımı" başlığı altında sunulan ve pencerenin büyük bir bölümünü kapsayan panel, görsel ve etkileşimli özellikleri sayesinde, kullanıcıların tanımlı durumdaki üyelik fonksiyonlarını görsel olarak gözlemleyebilmesini sağlayan etkili bir yazılım işlevi olarak kullanıcılara sunulmaktadır. Seçili durumdaki bir değişken içerisinde yer alan üyelik fonksiyonları, söz konusu bu panel altında görsel olarak kullanıcılara sunulmakta, böylece kullanıcıların gerçekleştirmekte oldukları düzenleme işlemleri konusunda daha etkili bir şekilde fikir sahibi olmaları sağlanmaktadır. Söz konusu panel bünyesinde, seçili durumdaki değişkenin değer aralığına bağlı olarak belirlenen, belli başlı değerler görüntülenmekte, ilgili değişkende yer alan üyelik fonksiyonları da, sahip oldukları parametre değerleri ve değişken değer aralığı kapsamında, adlarıyla birlikte, görsel olarak sunulan bu yapıya otomatik olarak yerleştirilmektedir. Açıklaması yapılmış olan üyelik fonksiyonları düzenleme panelinde, ilgili liste kutusundan bir üyelik fonksiyonu seçildiğinde, "Üyelik Fonksiyonu Tanımı" başlıklı panel altında sunulan görsel şekil de kırmızı renkte seçili bir duruma gelmektedir. Yine açıklaması yapılmış olan kontroller yardımıyla, seçili bir üyelik fonksiyonunun özellikleri (adı, türü ve parametreleri) değiştirildiğinde, yapılan değişiklikler anında söz konusu paneldeki görsel alana yansıtılmaktadır. Resim 4.10, tanımlanabilecek bütün üyelik fonksiyonu türlerinin, varsayılan parametreler kapsamında oluşturulan, görsel alandaki karşılıklarını

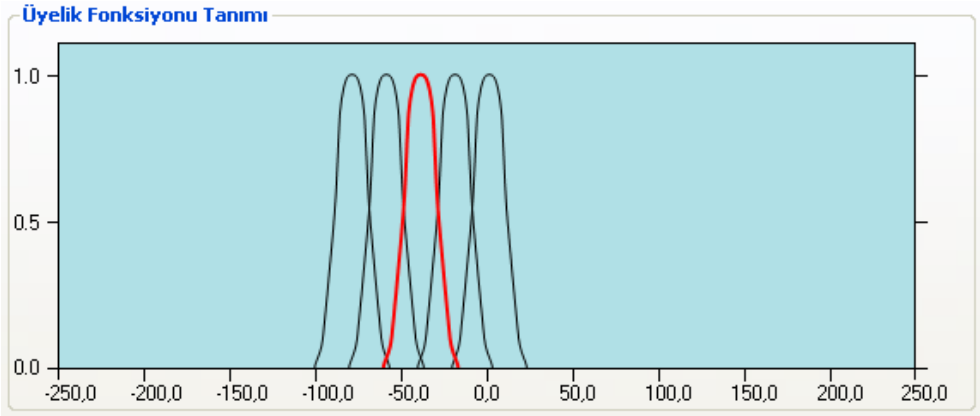
göstermektedir. Bu resim kapsamında sunulmuş olan üyelik fonksiyonlarından "üçgen" türünde olan seçili durumda görüntülenmiştir.



Resim 4.10 Tanımlanabilecek üyelik fonksiyonlarının görsel karşılıkları.

"Sistem Değişkenleri" penceresinde yer alan diğer paneller de "Seçili Değişken Özellikleri" ve "Tanım Görüntüleme Özellikleri" başlıkları altında sunulmaktadır. Anlaşılacağı üzere, "Seçili Değişken Özellikleri" paneli, daha önce açıklanmış olan değişken listesinde seçili durumda olan sistem değişkeninin adı ve değer aralığı gibi özelliklerinin, kullanıcılar tarafından değiştirilmesi amacıyla kullanılmaktadır. Yapılan değişiklikler, yine bu panelin üst kısmına doğru konumlandırılmış, "Değişiklikleri Kaydet" başlıklı düğme yardımıyla sisteme tanıtılmaktadır. Yapılan değişiklikler, "Üyelik Fonksiyonu Tanımı" paneline anında yansımaktadır. Diğer yandan, yine pencere üzerinde konumlanmış olan "Tanım Görüntüleme Özellikleri" paneli, seçili durumdaki değişkende görüntülenen üyelik fonksiyonlarının, daha geniş değer aralıkları yardımıyla, rahat bir şekilde görüntülenmesini sağlamaktadır. Bu noktada önemli olan, ilgili panelde yapılan değişiklikler, sadece "Üyelik Fonksiyonu Tanımı" panelinde sunulan görsel grafiğin aralıklarını değiştirmekte, buna karşın, tanımlı olan değişkenin gerçek değer aralıkları üzerinde herhangi bir değişiklik yapmamaktadır. Yine bu panel üzerinde sunulmuş olan kontroller yardımıyla, "Üyelik Fonksiyonu Tanımı" panelinde gösterilen üyelik fonksiyonlarının sadece çizgiler halinde görüntülenip–görüntülenmemesi ve/veya üyelik fonksiyonları adlarının görüntülenip–görüntülenmemesi işlevleri ayarlanabilmektedir. Örnek olması açısından, varsayılan

değer aralığı [-100…100] olan bir çıkış değişkenine ait üyelik fonksiyonlarının, ilgili panel üzerindeki kontroller yardımıyla [-250…250] aralığına ayarlanmış ve ayrıca çizgiler halinde, adları da gizlenecek şekilde görüntülenmiş hali, Resim 4.11'de gösterilmektedir.

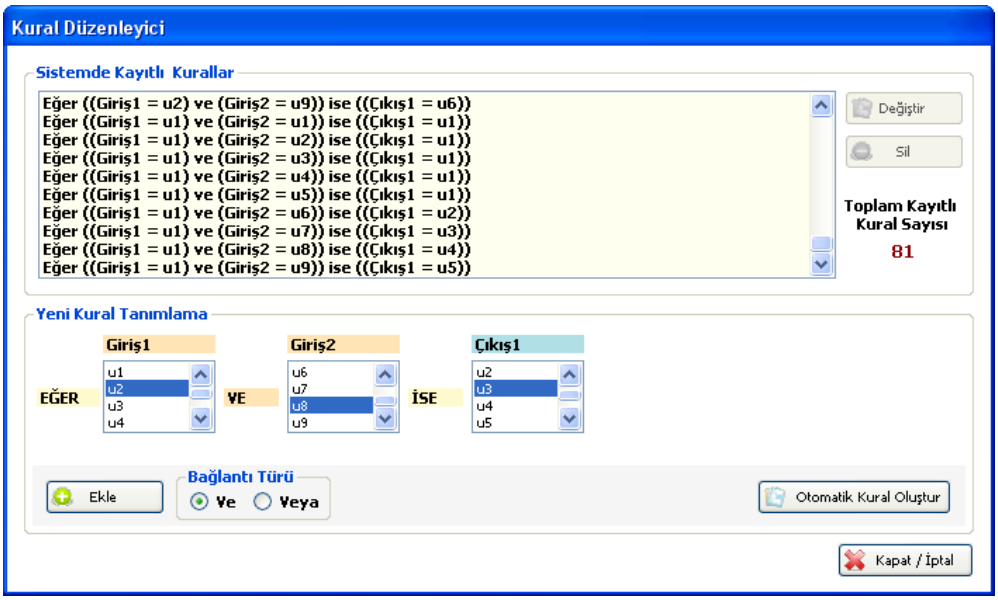


Resim 4.11 Farklı tanım aralığı ve özelliklerinde, çıkış değişkeni fonksiyonları.

"Sistem Değişkenleri" penceresinde gerekli tanımlama ve düzenlemeler yapıldıktan sonra, bulanık mantık yazılımı arayüzündeki "Bulanık Mantık Sistem Araçları" panelinde yer alan diğer düğmeler de kullanıcılar tarafından, farklı sistem işlevlerini yerine getirmek için kullanılabilmektedir. Tipik bir bulanık kontrol sisteminin değişkenleri ve üyelik fonksiyonları tanımlandıktan sonra yerine getirilmesi gereken diğer önemli bir görev de dilsel kuralların tanımlanmasıdır. Söz konusu bu görev, "Bulanık Mantık Sistem Araçları" paneli altında sunulmuş olan "Kurallar" düğmesi yardımıyla yerine getirilebilmektedir. İlgili düğmeye tıkladıktan sonra görüntülenen "Kural Düzenleyici" penceresinde, tanımlı durumda olan giriş ve çıkış değişkenlerine ve ilgili üyelik fonksiyonlarına bağlı olarak çeşitli dilsel kurallar tanımlamak mümkündür. Bu bağlamda, yazılım sistemi, aktif durumdaki giriş ve çıkış değişkenlerini, üyelik fonksiyonlarıyla birlikte, pencere ortamına otomatik olarak yerleştirmektedir. Bu noktada, sistem değişkenleri içerisinde yer alan üyelik fonksiyonlarının listelenmesi için, değişken sayısı kadar liste kutusu kontrolü ilgili pencere ortamında oluşturulmaktadır. Kullanıcılar, sunulan bu liste kutularından istedikleri üyelik fonksiyonlarını seçebilmekte, yine pencerede sunulan "Bağlantı Türü"

panelinden “ve” ya da “veya” bağlantı şeklini de ayarladıktan sonra, tanımlanan dilsel kuralı “Ekle” düğmesiyle sisteme tanıtabilmektedir. Sistem bünyesinde tanıtılmış olan dilsel kurallar, pencerenin üst alanında yer alan büyük liste kutusu içerisinde, alt alta listelenmektedir. Listeden seçilen bir dilsel kuralı değiştirmek ya da silmek için, yine ilgili liste kutusunun yan tarafında sunulmuş olan, “Değiştir” ve “Sil” düğmeleri kullanılabilmektedir.

“Kural Düzenleyici” penceresi bünyesinde, yazılım tarafından sunulmuş olan diğer önemli bir işlev de, otomatik kural oluşturmadır. Bu işlev, varsayılan olarak eşit sayıda üyelik fonksiyonuna sahip, 2 adet giriş ve 1 adet çıkış değişkenine sahip sistemler üzerinde kullanılabilmektedir. Buna göre, söz konusu şartları sağlayan bir sistem için, “Kural Düzenleyici” penceresinde yer alan “Otomatik Kural Oluştur” düğmesi aktif hale gelmektedir. Kullanıcı, bu düğmeye tıkladıktan sonra, dilsel kurallar simetrik tablo yapısı mantığına göre otomatik bir şekilde çıkartılmakta ve sisteme tanıtılmaktadır. Çıkartılan bütün dilsel kurallar bu esnada ilgili liste kutusuna da eklenmektedir. Resim 4.12, “Kural Düzenleyici” penceresinden bir ekran görüntüsü sunmaktadır. Bu resimde sunulan ve ekran görüntüsü alınan sistemin dilsel kuralları otomatik olarak oluşturulmuştur.

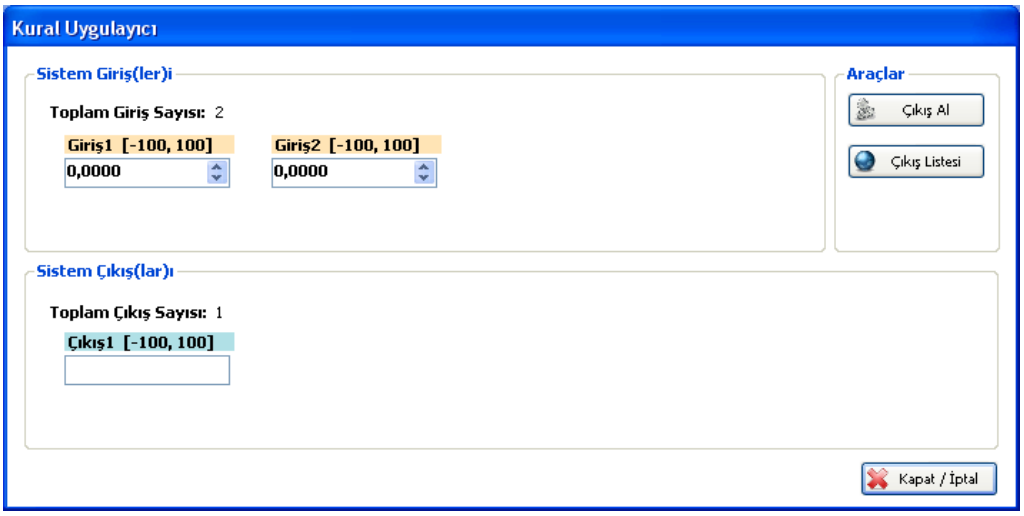


Resim 4.12 Kural Düzenleyici penceresinden bir ekran görüntüsü.

İstenilen bulanık kontrol sisteminin değişken ve üyelik fonksiyonları yapısı tasarlanıp, gerekli dilsel kurallar da tanımlandıktan sonra, kullanıcılar yine yazılım bünyesinde yer alan çeşitli kontrolleri kullanarak, uygulama ya da problemle alakalı çalışmalarını yürütebilmektedir. Söz konusu bu çalışmalar 4.4.2 başlığı altında verilmiştir.

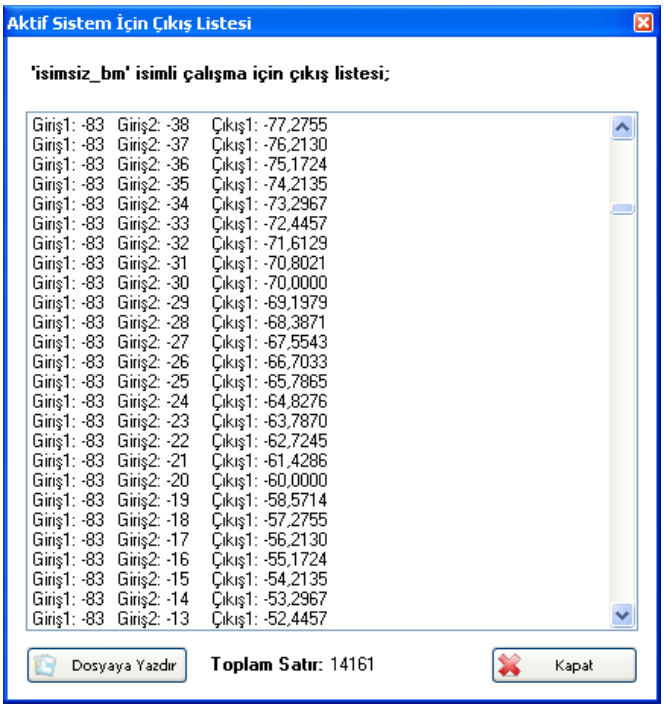
4.4.2 Bulanık Kontrol Sistemleri Üzerinde Yapılabilecek Çalışmalar

"Bulanık Mantık Sistem Araçları" paneli altında sunulmuş olan "Kural Uygulayıcı" düğmesi, aktif durumdaki sistemin farklı giriş değerlerine nasıl yanıt verdiğini gözlemlemek amacıyla kullanılan, ilgili pencerenin görüntülenmesi amacıyla kullanılmaktadır. Tıpkı "Kural Düzenleyici" penceresinde olduğu gibi, "Kural Uygulayıcı" penceresinde de, giriş ve çıkış değişkenlerinin sayısına bağlı olarak, çeşitli kontroller otomatik olarak oluşturulmaktadır. Buna göre kullanıcılar, pencere ortamında sunulan, giriş değişkenlerine bağlı olarak oluşturulan veri giriş alanlarını kullanarak, sisteme istedikleri giriş verilerini vermekte, ardından "Çıkış Al" düğmesine tıklayarak, çıkış değişkenlerine bağlı olarak oluşturulan kontrollerde, sistem tarafından gelen yanıtları gözlemleyebilmektedir. "Kural Uygulayıcı" penceresinden bir ekran görüntüsü, Resim 4.13 altında sunulmaktadır.



Resim 4.13 Kural Uygulayıcı penceresinden bir ekran görüntüsü.

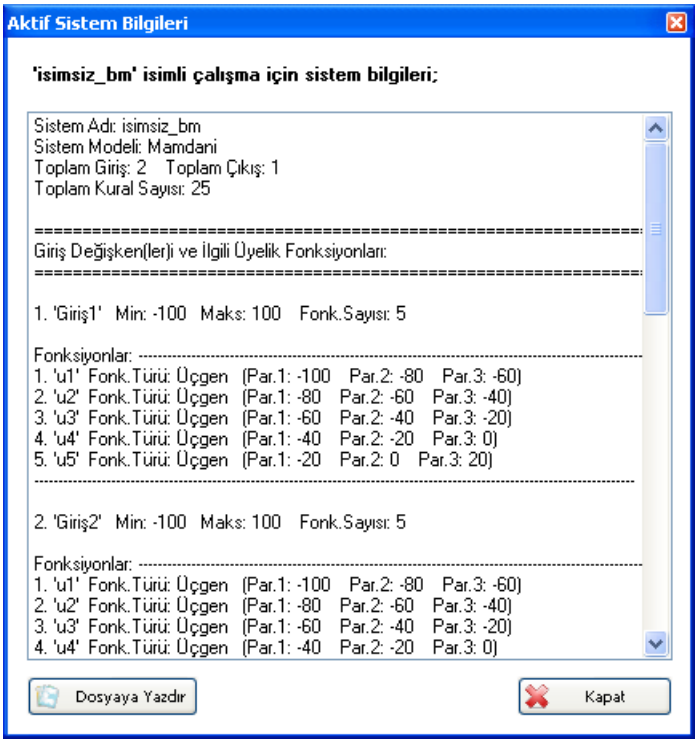
"Kural Uygulayıcı" penceresinde sunulmuş olan diğer bir düğme olan, "Çıkış Listesi" düğmesi kullanılarak, aktif sistemdeki bütün giriş ihtimallerinin ve bu girişlerin değerlendirilmesi sonucunda sistemden elde edilen yanıt(lar)ın [çıkış(lar)ın] listelendiği bir pencere ekrana getirilmektedir. "Aktif Sistem İçin Çıkış Listesi" başlığı altında sunulmuş olan bu pencerede, giriş ve çıkış değişkenlerinin sayılarına bağlı olarak en fazla 5000 ihtimal olmak üzere, çeşitli aralıklarda elde edilen değerler, kullanıcının talimatlarına göre listelenmektedir. Listeleme işlemi, kullanıcı tarafından, ekrana gelen mesaj kutusuna verilen yanıtlar doğrultusunda durdurulup–devam ettirilebilmektedir. Elde edilen listedeki veriler, istendiği takdirde kullanıcı tarafından .TXT dosya formatında kaydedilebilmekte ve farklı bulanık mantık uygulama yazılımlarında farklı türde hesaplamalar yapmak için kullanılabilmektedir. Söz konusu pencereden bir ekran görüntüsü Resim 4.14 altında gösterilmektedir.



Resim 4.14 Aktif Sistem – Çıkış Listesi penceresinden bir ekran görüntüsü.

"Bulanık Mantık Sistem Araçları" paneli bünyesinde yer alan son düğme olan "Sistem Bilgisi Al" düğmesi, aktif durumda olan bulanık kontrol sistemiyle alakalı, ihtiyaç duyulan bütün bilgilerin listesini sunan, "Aktif Sistem Bilgileri" başlıklı pencerenin

görüntülenmesini sağlamaktadır. Söz konusu pencere kapsamında, aktif sistem ile alakalı ad, model türü, toplam giriş ve çıkış değişkenleri sayısı ve her değişkende yer alan üyelik fonksiyonlarıyla bağlantılı bütün bilgiler (fonksiyon adı, türü ve parametreleri) sunulmaktadır. Kullanıcı dilerse, yine bu pencerede yer alan "Dosyaya Yazdır" düğmesini kullanarak, söz konusu bilgileri .TXT dosya formatında kaydedilmesini sağlayabilmektedir. "Aktif Sistem Bilgileri" penceresi, Resim 4.15 altında gösterilmektedir.



Resim 4.15 Aktif Sistem Bilgileri penceresi.

Bulanık mantık uygulama ve eğitim yazılımı içerisinde sunulan en önemli işlev, tasarlanan kontrol sistemlerinin, matris formatında girilen matematiksel fonksiyonlar üzerinden işletilebilmesidir. Bu bağlamda, söz konusu yazılım bünyesinde, 2 adet giriş ve 1 adet çıkış değişkenlerine sahip sistemlerin kontrolü sağlanabilmektedir. Tasarlanan bir bulanık kontrol sisteminin matematiksel fonksiyonunun tanımlanması ve gerekli kontrol sürecinin gerçekleştirilebilmesi için, "Bulanık Mantık Sistem Tanımı" panelinde sunulmuş olan, "Matris Giriş" düğmesi kullanılmaktadır. Bu noktada, geliştirilen

yazılım, matematiksel fonksiyonların sisteme durum-uzay analiz yöntemleri kapsamında, matris formları halinde tanıtılmasına imkân vermektedir. Buna göre, aktif durumdaki bir bulanık kontrol sisteminin durum-uzay eşitliği aşağıdaki gibi tanımlanmaktadır:

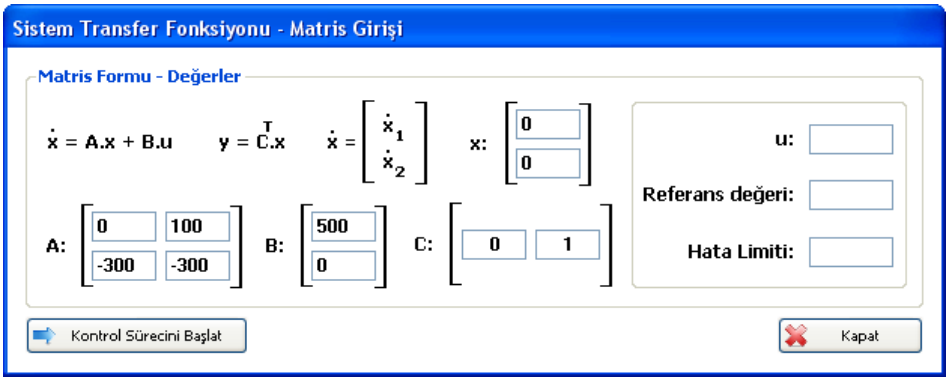
$$\dot{x} = A.x + B.u \qquad y = C^T.x \qquad ve \qquad \dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} \tag{4.1}$$

"Matris Giriş" düğmesine tıklanmasıyla beraber ekrana gelen pencerede, kullanıcıların tasarladıkları bulanık kontrol sistemleri için, söz konusu Eşitlik 4.1 kapsamında da ifade edilmiş olan, A, B, C ve x matris değerlerini tanımlaması, ilgili fonksiyon kapsamındaki u ve referans (başvuru) değerlerini belirtmesi ve son olarak, söz konusu kontrol süreci için gereken hata limiti değerini, ilgili veri giriş alanlarına yazması mümkündür. Söz konusu değerler belirtildikten sonra, yine aynı pencere bünyesinde yer alan, "Kontrol Sürecini Başlat" düğmesine tıklayarak, tasarlanan sistemle alakalı kontrol işleminin başlatılması sağlanmaktadır. Bu noktada, bulanık kontrol sistemi bünyesindeki kontrol işlemleri, kısaca şu şekilde yerine getirilmektedir:

- Başlangıç aşamasında belirtilen değerler, A.x + B.u eşitliğinde işleme konulmakta, sonuç olarak $\dot{x}$ matrisi elde edilmektedir. Elde edilen bu matris, ardından $y = C^T.x$ eşitliğinde işleme konulmakta ve sonuç olarak da y değeri elde edilmektedir.
- Elde edilen y değeri, referans (başvuru) değeri ile karşılaştırılmakta, elde edilen sonuç "hata (e)" değeri olarak ele alınmaktadır. Söz konusu bu hata değeri de, bir önceki iterasyonda elde edilen hata değeri (1. adımda 0 değerindedir) ile karşılaştırılmakta, bu noktada elde edilen değer de, "hatadaki değişim (de)" değeri olarak değerlendirilmektedir.
- Hata (e) ve hatadaki değişim (de) değerleri, bulanık kontrol sisteminde karşılıkları olan girişlerden, tasarlanan sisteme verilmekte ve ilgili yaklaşımlar ve dilsel kurallar yardımıyla sistem çıkışı (du) elde edilmektedir. Bu noktada, elde edilen çıkış "artırım miktarı" olarak da anılmaktadır.

- Elde edilmiş olan du değeri, bir önceki iterasyonda elde edilen u değerine (1. adımda 0 değerindedir) eklenmekte ve yeni elde edilen u değeri, A.x + B.u eşitliğinde kullanılmak suretiyle, ilk işlem adımlarına tekrar dönülmektedir. Söz konusu işlemler, belirtilen hata limiti şartının sağlanıp sağlanmamasına göre tekrar edilmektedir.

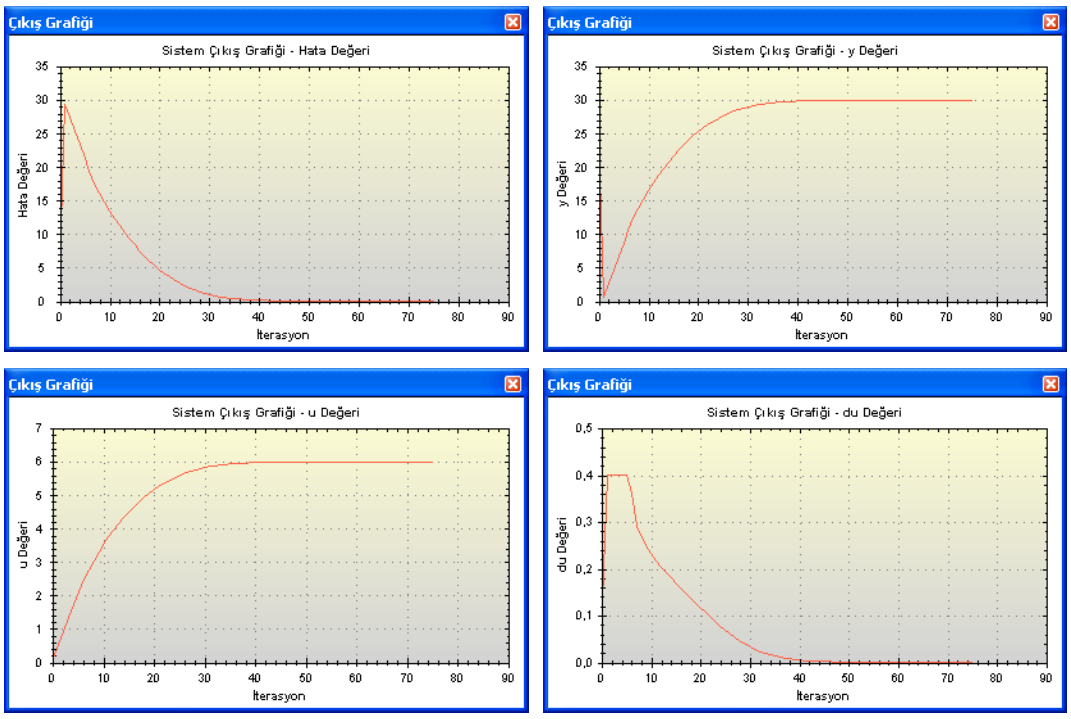
Resim 4.16, "Matris Giriş" penceresinden bir ekran görüntüsü sunmaktadır.



Resim 4.16 Matris Giriş penceresinden bir ekran görüntüsü.

İlgili pencerede bir kontrol süreci tamamlandıktan sonra, söz konusu süreç kapsamında ulaşılan iterasyon sayısı ve elde edilen son u değeri, kullanıcıya bir mesaj kutusu aracılığıyla bildirilmektedir. Buna ek olarak, süreç esnasında, hata değeri, y değeri, u değeri ve aktif sistem çıkışında (du değeri) gerçekleşen değişimler, dört farklı grafik bünyesinde, otomatik olarak görüntülenmektedir. Bu grafikler, kontrol süreci sırasında meydana gelen, ilgili değerlerdeki değişimleri, görsel anlamda daha etkili ve anlaşılır bir biçimde kullanıcıya sunmaktadır. Söz konusu her grafik, vektörel yapıda olmalarından dolayı, kullanıcının yakınlaştırma / uzaklaştırma ve büyütme / küçültme gibi işlevleri kullanması yoluyla, daha detaylı bir şekilde incelenebilmektedir. Bu noktada ayrıca, ilgili grafiklerin yazıcı çıktısı alınabilmekte, hatta farklı resim dosya formatları halinde kaydedilmelerine de imkân verilmektedir. Kullanıcılar bütün bu işlevlere, grafikler üzerine sağ tuş tıklandıktan sonra ekrana gelen menüler aracılığıyla ulaşabilmektedir. Elde edilen grafiklerden herhangi birisi ya da tümü kapatıldıktan sonra, istendiği takdirde tekrardan, "Bulanık Mantık Sistem Tanımı" panelinde yer alan

"Grafik" düğmesi aracılığıyla görüntülenebilen, "Grafik Görüntüle" penceresi üzerinden açılabilmektedir. Resim 4.17, bir kontrol süreci sonrası elde edilen grafiklerden örnek ekran görüntüleri sunmaktadır.



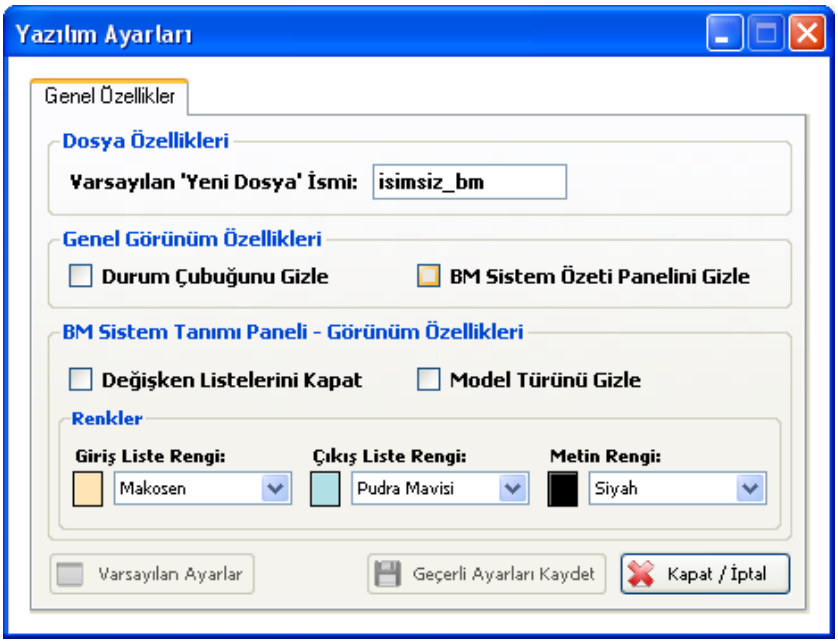
Resim 4.17 Bir kontrol süreci sonrası elde edilen grafiklerden örnek ekran görüntüleri.

Açıklaması yapılan söz konusu kullanım özellikleri ve işlevleri, bulanık mantık tabanlı çalışma ya da uygulamaların, bulanık mantık uygulama ve eğitim yazılımı arayüzünde yerine getirilmesi yönleriyle ifade edilmiştir. İlgili özellik ve işlevlerin dışında, değinilmesi gereken birtakım ek özellik ve işlevler de yine söz konusu yazılım bünyesinde kullanıcılara sunulmuştur.

4.4.3 Yazılım Bünyesindeki Diğer Özellik ve İşlevler

Anlatılan kullanım özelliklerinin yanında, yazılımın bazı özellik ve işlevleri, yine kullanıcı tarafından ilgili kontroller yardımıyla ayarlanabilmektedir. Bulanık mantık uygulama ve eğitim yazılımının arayüzünde sunulmuş olan, "Eğitim Yazılımı Araçları"

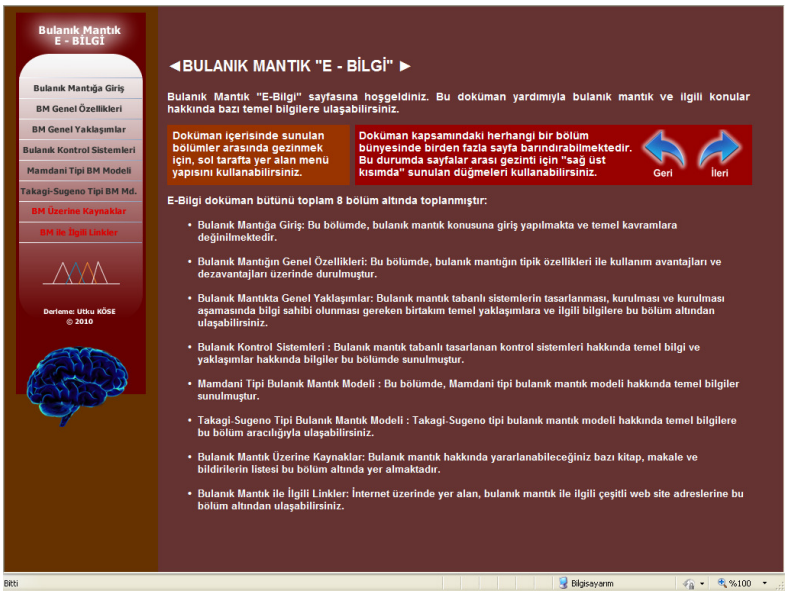
panelindeki "Yazılım Ayarları" düğmesi yardımıyla, ilgili düzenlemelerin yapılabileceği bir pencere ekrana getirilebilmektedir. "Yazılım Ayarları" başlıklı pencerede sunulmuş olan kontroller yardımıyla, yazılımda kullanılan varsayılan dosya ismi, bazı panellerin gizlenip / gizlenmeyeceği, "Bulanık Mantık Sistem Paneli" ile bağlantılı görüntüleme ayarları ve görsel anlamda, giriş ile çıkış değişkenlerinin renk özelliklerinin ne olacağı gibi birtakım düzenlemeler kolaylıkla gerçekleştirilebilmektedir. Söz konusu bu ayarlar kullanıcılara, kendi seçimlerine bağlı olarak oluşturulabilecek bir tasarım ve geliştirme ortamında, ilgili çalışmalarını yerine getirme imkânı vermektedir. Daha önce değiştirilmiş olan ayarlar, istenildiği takdirde, "Varsayılan Ayarlar" düğmesi kullanılarak, sistemin ilk aşamada sahip olduğu varsayılan durumlarına tekrar döndürülebilmektedir. "Yazılım Ayarları" penceresi Resim 4.18'de gösterilmektedir.



Resim 4.18 Yazılım Ayarları penceresi.

"Eğitim Yazılımı Araçları" panelinde sunulmuş olan, "Bulanık Mantık E-Bilgi" düğmesi ile birlikte, bulanık mantık tekniği ile alakalı temel prensiplerin ve uygulama yaklaşımlarının kısa bir şekilde anlatıldığı, ayrıca bulanık mantık tekniği ile ilgili ek kaynakların verildiği, web sayfası formatında sunulan, "Bulanık Mantık E-Bilgi" –

HTML sayfası ekrana getirilebilmektedir. Söz konusu sayfa, kullanıcıların bulanık mantık uygulama ve eğitim yazılımını kullanırken, ilgili teknikle alakalı birtakım bilgilere kolayca ulaşmasını ve akıllarına takılan ya da öğrenmek istedikleri noktalarda, yine sunulan sayfadan bir başvuru kaynağı niteliğinde yararlanmalarını amaç edinmektedir. Bulanık Mantık E-Bilgi sayfasından örnek bir ekran görüntüsü, Resim 4.19'da gösterilmektedir.



Resim 4.19 Bulanık Mantık E-Bilgi – HTML sayfasından bir ekran görüntüsü.

Gerçekleştirilebilecek çalışma şekilleri açıklanan bulanık mantık uygulama ve eğitim yazılımının etkinliği ve doğruluğu noktalarında daha fazla fikir sahibi olmak için, söz konusu yazılımı kullanarak gerçekleştirilen örnek bir uygulama çalışması ile elde edilen sonuçların değerlendirilmesi mümkündür. Bu kapsamda, ilgili yazılım ortamında iki farklı uygulama ele alınmıştır.

4.4.4 Bulanık Mantık Uygulama ve Eğitim Yazılımı ile Örnek Çalışmalar

Yazılım ortamında ele alınan ilk uygulama, 2 adet giriş değişkeni ve 1 adet çıkış değişkeninden meydana gelen, temel yapıda bir uygulama olmuştur. Söz konusu uygulama kapsamında, farklı düzeydeki "hizmet kalitesi" ve "sunulan gıda niteliği"

özellikleri giriş değişkenleri olarak ele alınmış, bulanık değerlendirme sonucunda elde edilen, "sunulan toplam hizmetin başarı düzeyi" değeri ise sistemin çıkış değişkeni olarak değerlendirilmiştir. Sistem, Mamdani bulanık mantık modeline göre tasarlanmıştır. Bu kapsamda ele alınan hizmet kalitesi ve sunulan gıda niteliği giriş değişkenleri [0…10] değer aralığında, "sonuç" adı altında ele alınan çıkış değişkeni ise [0…30] değer aralığında tanımlanmıştır. Hizmet kalitesi giriş değişkeni için toplam 3 adet üçgen tipi üyelik fonksiyonu, sırasıyla "zayıf", "iyi" ve "mükemmel" etiketleri altında ele alınmıştır. Bu üyelik fonksiyonları için yine sırasıyla [-5, 0, 5], [0, 5, 10] ve [5, 10, 15] parametre değerleri kullanılmıştır. Diğer taraftan, sunulan gıda niteliği giriş değişkeni için toplam 3 adet yamuk tipi üyelik fonksiyonu, sırasıyla "kötü", "orta" ve "lezzetli" etiketleri altında, [0, 0, 1, 3], [1, 4, 6, 9] ve [7, 9, 10, 10] parametre değerleri ile tanımlanmıştır. Son olarak, sonuç çıkış değişkeni ise toplam 3 adet üçgen tipi üyelik fonksiyonunda, "ucuz", "orta" ve "yüksek" etiketleri altında tanımlanmıştır. Söz konusu bu üyelik fonksiyonları için de sırasıyla [0, 5, 10], [10, 15, 20] ve [20, 25, 30] parametre değerleri kullanılmıştır.

Uygulama kapsamındaki sistem değişkenleri ve üyelik fonksiyonları tanımlandıktan sonra, dilsel kurallar, simetrik tablo yapısı halinde, Çizelge 4.3'de gösterildiği şekilde sisteme tanıtılmıştır.

Çizelge 4.3 Uygulama için tanımlanan dilsel kurallar.

No:	**Dilsel Kural:**
1	Eğer (hizmet = mukemmel) ve (gida = kotu) ise (sonuc = orta)
2	Eğer (hizmet = mukemmel) ve (gida = lezzetli) ise (sonuc = yuksek)
3	Eğer (hizmet = mukemmel) ve (gida = orta) ise (sonuc = yuksek)
4	Eğer (hizmet = iyi) ve (gida = kotu) ise (sonuc = ucuz)
5	Eğer (hizmet = iyi) ve (gida = lezzetli) ise (sonuc = orta)
6	Eğer (hizmet = iyi) ve (gida = orta) ise (sonuc = yuksek)
7	Eğer (hizmet = zayif) ve (gida = kotu) ise (sonuc = ucuz)
8	Eğer (hizmet = zayif) ve (gida = lezzetli) ise (sonuc = ucuz)
9	Eğer (hizmet = zayif) ve (gida = orta) ise (sonuc = orta)

Kural tanımlama sonrasında, yazılımdaki "Kural Uygulayıcı" penceresi yardımıyla, çeşitli giriş değerlerine karşılık, sisteminin yanıtları (çıkışları) gözlemlenmiştir. Yazılımın ürettiği sonuçların doğruluğunu onaylamak adına, söz konusu uygulama MATLAB Fuzzy Logic Toolbox yazılım ortamında da oluşturulmuş ve bu yazılımın "Rule Viewer" penceresinde elde edilen sonuçlar ile "Kural Uygulayıcı" penceresindeki değerler karşılaştırılmıştır. Sonuçlardan bazıları, Çizelge 4.4'de sunulmuştur.

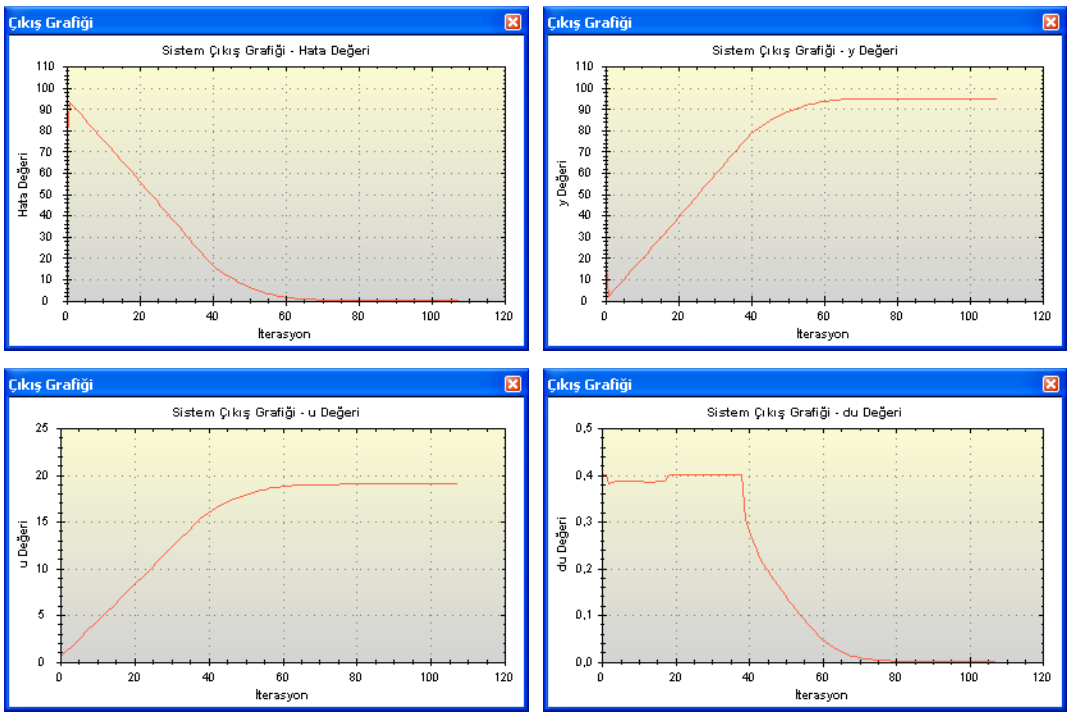
Çizelge 4.4 Uygulama kapsamında, farklı yazılımlar ile elde edilen sonuçlar.

Giriş 1: Hizmet	**Giriş 2: Gıda**	**Geliştirilen yazılım ile elde edilen sonuç**	**MATLAB yazılımı ile elde edilen sonuç**
5,0000	8,0000	19,2551	19,3000
2,0000	8,0000	13,9978	14,0000
4,6000	3,2000	23,5806	23,6000
9,6540	1,4560	16,0781	16,1000
6,6700	3,5670	25,0010	25,0000

İkinci uygulama olarak, geliştirilen yazılımda uygulanabilen, matematiksel fonksiyonu A.x + B.u formatında ifade edilebilen bir sistemin kontrolü gerçekleştirilmiştir. Oluşturulan sistem, dokuz adet üçgen tipi üyelik fonksiyonuna sahip, [-100…100] değer aralığında, iki adet giriş değişkenine ve yine dokuz adet üçgen tipi üyelik fonksiyonuna sahip, [-100…100] değer aralığında, bir adet çıkış değişkenine sahip olarak tasarlanmıştır. Söz konusu sistemin matematiksel fonksiyon–matris yapısı aşağıda belirtildiği gibi sisteme tanıtılmıştır:

$$A = \begin{bmatrix} 0 & 10 \\ -5 & -5 \end{bmatrix}, \qquad B = \begin{bmatrix} 5 \\ 0 \end{bmatrix}; \qquad C = [0 \;\; 1] \tag{4.2}$$

Gerçekleştirilecek kontrol süreci için hata limiti olarak 10^{-4} değeri belirtilmiş ve ilgili süreç, belirtilen diğer değerler kapsamında, 109. iterasyon sonrası sona ermiştir. Süreç sonucunda elde edilen grafikler, Resim 4.20'de sunulmuştur.



Resim 4.20 Uygulama – kontrol süreci sonucunda elde edilen grafikler.

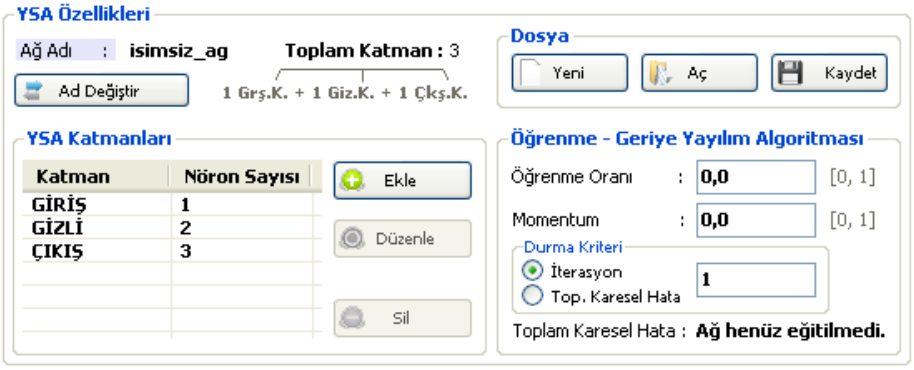
Gerçekleştirilen uygulamalar sonucunda elde edilen veriler göstermektedir ki, söz konusu uygulamalar bünyesinde ele alınan problemler, geliştirilen bulanık mantık uygulama ve eğitim yazılımı bünyesinde doğru ve tutarlı bir biçimde değerlendirilmektedir. Bu bağlamda ilgili yazılımın, bulanık mantık tekniğinin, ele alınan konu kapsamında gerçekleştirilmesi kapsamında, etkin ve doğru ölçüde bir uygulama ortamı sağladığı ifade edilebilmektedir.

4.5 Yapay Sinir Ağları Uygulama ve Eğitim Yazılımı ile Çalışmak

Yapay sinir ağları uygulama ve eğitim yazılımı ortamında çalışmak ve uygulama geliştirmek, düzenli ve sade bir yapıda sunulan kontroller sayesinde, benzer uygulamalara göre daha anlaşılır ve basit bir şekilde yerine getirilebilmektedir. Söz konusu ortamda bir yapay sinir ağı sistemi geliştirmek ve bu ağ üzerinde gerekli çalışmaları gerçekleştirmek, adım adım yerine getirilebilen birtakım kullanım şekillerinden ve görevlerinden oluşmaktadır.

4.5.1 Yapay Sinir Ağlarının Tasarlanması ve Düzenlenmesi

Yapay sinir ağları yazılım arayüzünde sunulan, "YSA Özellikleri" başlıklı panel, aktif durumdaki ağ yapısıyla alakalı çeşitli özelliklerin düzenlenebilmesine imkân tanıyan, bazı kontrolleri bünyesinde toplamaktadır. İlgili panelden bir ekran görüntüsü, Resim 4.21 altında sunulmuştur.



Resim 4.21 Yapay sinir ağları arayüzünde "YSA Özellikleri" paneli.

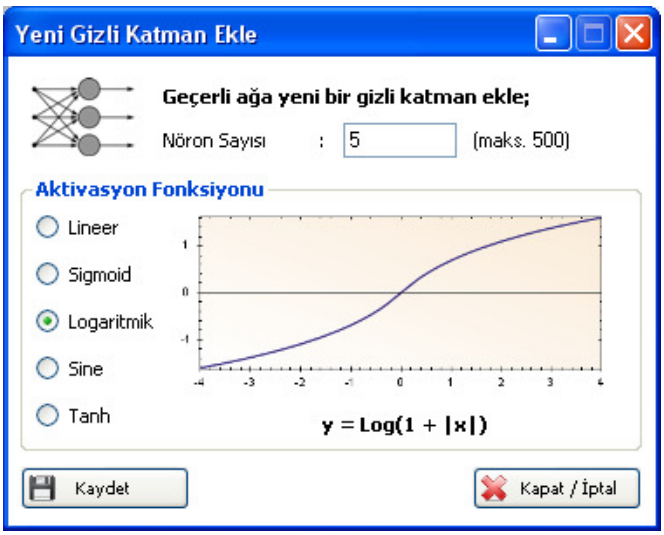
"YSA Özellikleri" panelinde yer alan, "Dosya" başlıklı panelde sunulmuş olan düğmeler kullanılarak, yeni bir çalışma ortamının açılması ("Yeni"), aktif durumdaki ağ çalışmasının XML dosya formatında kaydedilmesi ("Kaydet") ve daha önce yine XML dosya formatında kaydedilmiş olan bir çalışmanın sisteme yüklenmesi ("Aç") gerçekleştirilebilmektedir. Yine "YSA Özellikleri" panelinde yer alan bazı kontroller sayesinde, aktif durumdaki çalışmanın ismini değiştirmek ve yine aktif durumdaki ağda yer alan katman sayıları hakkında özet bilgiler almak mümkündür. İlgili panel altında sunulmuş olan, "Öğrenme – Geriye Yayılım Algoritması" başlıklı panel kullanılarak, tasarlanmış ağın eğitimine başlamadan önce, eğitim sürecini etkileyen, birtakım parametreleri ayarlamak mümkün olmaktadır. Söz konusu bu parametreler, ilerleyen bölümlerde daha detaylı bir şekilde ele alınacaktır. Bu noktada, yeni bir "çok katmanlı yapay sinir ağı sistemi" tasarlamak ya da tasarlanmış olan bir ağın üzerinde gerekli düzenlemeleri yapmak, yapay sinir ağları uygulama ve eğitim yazılımı üzerinde çalışmaya başladıktan sonra ele alınması gereken öncelikli görevlerden birisi olarak

değerlendirilmektedir. Söz konusu bu görevler, yine "YSA Özellikleri" paneli altında sunulmuş olan, "YSA Katmanları" başlıklı panel aracılığıyla yerine getirilebilmektedir.

"YSA Katmanları" panelinde sunulmuş olan kontrol, iki sütun altında ("Katman" ve "Nöron Sayısı"), aktif durumdaki ağ sisteminin katmanları ve ilgili katmanlardaki yapay sinir hücrelerinin sayısı hakkında birtakım bilgiler sunmaktadır. Söz konusu bu kontrolün yanında yer alan düğmeler ise, aktif sisteme yeni bir "gizli" katmanın eklenmesi ("Ekle"), kontrol üzerinde seçilmiş olan bir katmanın özelliklerinin düzenlenmesi ("Düzenle") ve yine kontrol üzerinde seçilmiş olan bir katmanın silinmesi ("Sil") işlemlerinin yerine getirilmesini sağlamaktadır. Bilindiği üzere, tipik bir "çok katmanlı yapay sinir ağı mimarisinde" en az bir giriş katmanı, bir gizli katman ve yine bir çıkış katmanı bulunmaktadır. Bu nedenle, yapay sinir ağları uygulama ve eğitim yazılımı, giriş ve çıkış katmanlarının silinmesine izin vermemekte, "Düzenle" düğmesi aracılığıyla bu katmanların sadece özelliklerinin ayarlanmasına izin vermektedir. Benzer şekilde, aktif durumdaki ağ sisteminde bir adet gizli katman kalmışsa, yazılım yine bu katmanın silinmesine izin vermemektedir.

Anlaşılacağı üzere, aktif durumdaki bir ağ sistemine, "Ekle" düğmesi yardımıyla yeni bir katman eklenmesi demek, söz konusu sisteme yeni bir "gizli" katman eklemek anlamına gelmektedir. Aktif sisteme eklenen yeni gizli katmanlar, katmanların düzenlendiği kontrol üzerinde, "GİZLİ (2), GİZLİ (3)…GİZLİ (N)" şeklinde ardışık numaralar halinde, isimlendirilerek görüntülenmektedir. Sistemde tanımlı durumdaki gizli katmanların numaralandırmaları, her gizli katman silindiğinde yazılım tarafından tekrar düzenlenmektedir.

Kullanıcılar, aktif sisteme yeni bir katman eklerken ya da tanımlı durumdaki bir katmanın özelliklerini ayarlarken aynı pencereyi kullanmaktadır. Ekleme ya da düzenleme durumuna göre "Yeni Gizli Katman Ekle" ya da "Katman Düzenle" başlıkları altında görüntülenen bu pencere, kullanıcıların katmanlarla alakalı yapay sinir hücresi (nöron) sayısı ya da aktivasyon fonksiyonu gibi temel özellikleri ayarlamasına imkân sağlamaktadır. İlgili pencereden bir ekran görüntüsü, Resim 4.22 altında sunulmaktadır.

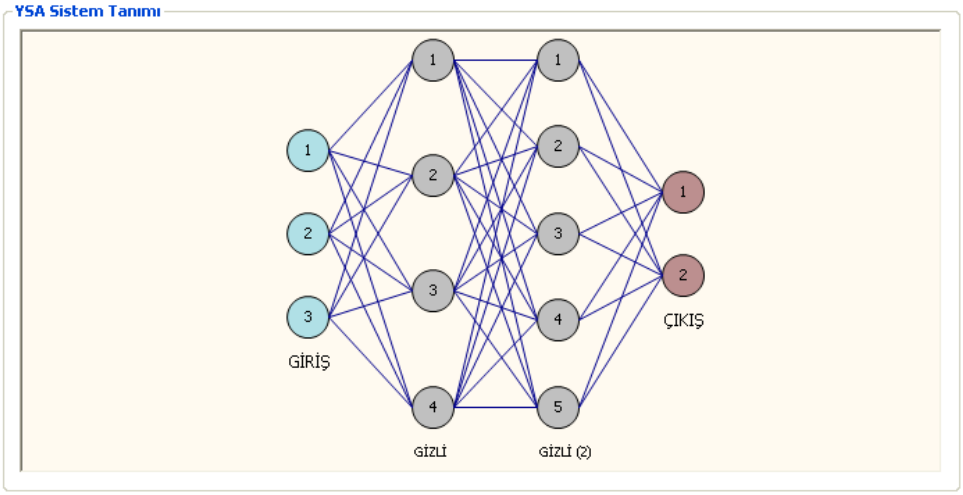


Resim 4.22 Yapay sinir ağları arayüzünde katman ekleme (düzenleme) penceresi.

İlgili pencerede yer alan "Nöron Sayısı" etiketi altındaki veri giriş alanı kullanılarak söz konusu katmanla ilgili yapay sinir hücresi sayısı kolaylıkla belirtilebilmektedir. Bu noktada, belirtilebilecek maksimum yapay sinir hücresi sayısı, her katman türü (giriş, gizli ve çıkış) için varsayılan "500" olarak ayarlanmıştır. Ancak ilerleyen bölümlerde de değinileceği üzere bu değer kullanıcı tarafından düzenlenebilmektedir. Söz konusu pencere altında yer alan diğer bir kontrol ise, ilgili katman için varsayılan aktivasyon fonksiyonunun ayarlanmasını sağlayan, "Aktivasyon Fonksiyonu" başlıklı paneldir. Bu panel yardımıyla atanacak aktivasyon fonksiyonu türleri sırasıyla Lineer (y = x), Sigmoid [y = 1 / (1 + Exp(-x))], Logaritmik [y = Log(1 + |x|)], Sine [y = Sin(x)] ve Tanh [y = Tanh(x)] olarak belirlenmiştir.

Yeni bir yapay sinir ağı sisteminin tasarlanması ya da tasarlanmış durumdaki bir sistemin düzenlenmesi esnasında, ilgili yazılım ortamı tarafından sunulan önemli bir işlev de, ilgili yapay sinir ağı mimarisinin görsel anlamda kullanıcıya görüntülenmesidir. Aktif ağ sisteminde yapılan değişiklikler anında "YSA Sistem Tanımı" başlıklı panel alanına yansımakta ve kullanıcı tasarlamakta olduğu ağ sisteminin görsel modelini böylelikle görüntüleyebilmektedir. Bu işlev sayesinde, yapay sinir ağları ile ilgili gerçekleştirilen çalışma ve uygulamalar, renkli ve etkileşimli bir hal almaktadır. Örnek vermek gerekirse, 3 yapay sinir hücresine sahip giriş katmanı,

sırasıyla 4 ve 5 adet yapay sinir hücresine sahip, 2 adet gizli katmanı ve 2 yapay sinir hücresine sahip çıkış katmanı olan bir yapay sinir ağı sistemi, "YSA Sistem Tanımı" paneli altında, Resim 4.23'de gösterildiği gibi görüntülenmektedir.



Resim 4.23 YSA Sistem Tanımı paneli altında örnek bir yapay sinir ağı sistemi.

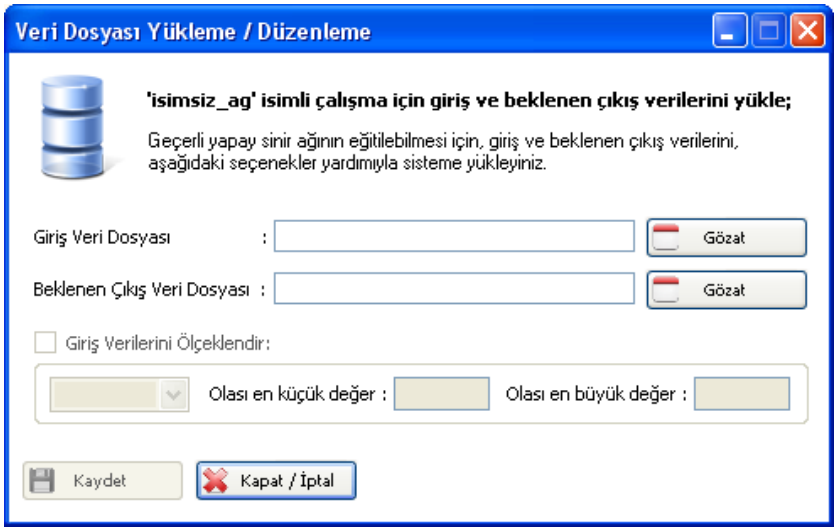
Açıklanan özellik ve işlevleri kullanarak, gerekli yapay sinir ağı sistemi tasarlandıktan sonra, sıra söz konusu ağ sisteminin eğitilmesi ve test edilmesi işlemlerine gelmektedir.

4.5.2 Yapay Sinir Ağlarının Eğitilmesi ve Test Edilmesi

Aktif yapay sinir ağı sisteminin eğitilmesi ve ardından test edilmesi sürecinde, ilgili yazılım arayüzünde bulunan, üç farklı panel söz sahibi olmaktadır. Bu paneller sırasıyla "YSA Sistem Araçları", "YSA İşlem Süreci" ve "YSA İşlem Adımları" başlıkları altında kullanıcılara sunulmaktadır. Bu noktada, eğitim sürecinin başlatılabilmesi için "YSA Sistem Araçları" başlıklı panel kullanılmalıdır.

"YSA Sistem Araçları" paneli altında, tasarlanan yapay sinir ağı sisteminin eğitilmesi, test edilmesi ve eğitilen sistemle alakalı performans ya da çıkış bilgilerinin görüntülenmesini sağlayan çeşitli düğmeler bulunmaktadır. Varsayılan olarak, eğitimi henüz yapılmamış bir ağ sisteminde, ilgili panel altında sadece "Veri" başlıklı düğme

aktif olarak sunulmaktadır. Bu bağlamda, diğer düğmelerin aktif hale gelmesi ve ilgili işlemlerin yapılabilmesi için, "Veri" düğmesi kullanılarak, tasarlanan ağ sisteminin eğitilmesini sağlayacak veri dosyalarının, ilgili yazılım sistemine tanıtılması gerekmektedir. Söz konusu işlem, "Veri" düğmesine tıklandıktan sonra görüntülenen "Veri Dosyası Yükleme / Düzenleme" başlıklı pencere aracılığıyla yapılabilmektedir. İlgili pencere, Resim 4.24'de gösterilmektedir.



Resim 4.24 Veri Dosyası Yükleme / Düzenleme penceresi.

Pencerede yer alan kontroller kullanılarak, daha önce .XLS ya da .CSV dosya formatlarında düzenlenmiş olan veri dosyaları, yazılım sistemine kolaylıkla yüklenebilmektedir. Tasarlanan ağ sisteminin eğitilmesi için gerekli olan, örnek giriş verilerinin olduğu dosya, "Giriş Veri Dosyası" etiketinin olduğu satırda sunulmuş olan "Gözat" düğmesi kullanılarak sisteme yüklenmektedir. Diğer taraftan, yine eğitim işlemi için gereken, beklenen çıkış verilerinin olduğu dosya, "Beklenen Çıkış Veri Dosyası" etiketinin bulunduğu satırda sunulan "Gözat" düğmesi ile yine sisteme yüklenmektedir. İlgili iki veri dosyası sisteme yüklenirken, yazılım sistemi dosyalar içerisinde sunulan verilerin, tasarlanan ağ sisteminin giriş ve çıkış katmanlarına olan uyumunu kontrol etmekte, herhangi bir hatayla karşılaşıldığında ise durum kullanıcıya mesaj kutusu eşliğinde bildirilmektedir. Bu noktada, sisteme yüklemeden önce kullanılan .XLS ya da .CSV dosyalarının içeriğinin, birtakım kurallara bağlı kalınarak

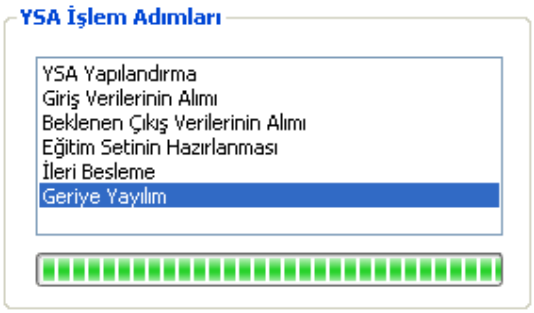
hazırlanması da oldukça önemlidir. Buna göre veriler hazırlanırken, Microsoft Excel yazılımı ortamında, her yeni veri kümesi yeni bir satırda, ilgili giriş ya da çıkış katmanındaki nöron sayısına göre sunulan her veri de, yine ilgili satırlarda, birbirlerinin arasında "virgül" karakteri olacak şekilde ayarlanmalıdır. Aksi takdirde, belirtildiği üzere yazılım kullanıcıyı ilgili dosya içeriklerinde hata olduğu konusunda uyarmaktadır.

Veri dosyalarının, tasarlanan sisteme olan uyumluluğu onaylandığında, söz konusu pencerede yer alan, "Giriş Verilerini Ölçeklendir" kontrolü de aktif hale gelmektedir. Kullanıcı dilerse, sisteme tanıtılan verilerin, bu kontrol altında sunulan parametreleri kullanarak, probleme uygun bir biçimde ölçeklendirilmesini sağlayabilmektedir.

Veri Dosyası Yükleme / Düzenleme penceresinde, sisteme tanıtılacak verilerle ilgili düzenlemeler yapıldıktan sonra, "Kaydet" düğmesi kullanılarak değişiklikler onaylanabilmektedir. Söz konusu pencere kapatıldığında, "YSA Sistem Araçları" paneli altında sunulan "Eğit" düğmesi de aktif hale gelmektedir. Kullanıcı bu düğmeye tıklayarak tasarlanmış olan ağ sistemini istediği zaman eğitmeye başlayabilmektedir. Ancak eğitme işlemine başlamadan önce, "YSA Özellikleri" paneli altında sunulmuş olan, "Öğrenme – Geriye Yayılım Algoritması" panelindeki parametrelerin gözden geçirilmesi gerekebilmektedir. Burada yer alan veri giriş alanları kullanılarak, eğitim esnasında kullanılabilen bazı değerler, sisteme tanıtılabilmektedir. Söz konusu veri giriş alanları kullanılarak, geri yayılım algoritması kapsamında kullanılacak olan "Öğrenme Oranı", "Momentum", "İterasyon" ve "Toplam Karesel Hata Limiti" gibi değerler kullanıcı tarafından belirtilebilmektedir. Bu noktada, eğitim sürecinin durma kriteri olarak, toplam iterasyon sayısının ya da toplam karesel hata limitinin dikkate alınması sağlanabilmektedir. Son olarak, yine ilgili panelde, "Toplam Karesel Hata" etiketli alan altında, eğitim sonrasında elde edilen hata değeri görüntülenmektedir. Varsayılan olarak, tasarlanan sistem henüz eğitilmemişse, bu etiket altında "Ağ henüz eğitilmedi" ibaresi yer almaktadır.

Eğitim için gerekli olan parametreler belirtildikten sonra, "Eğit" düğmesine tıklanarak eğitim süreci başlatılmaktadır. Eğitimin başlaması ile birlikte, daha önce pasif yapıda olan, "YSA İşlem Süreci" ve "YSA İşlem Adımları" panelleri otomatikman aktif hale

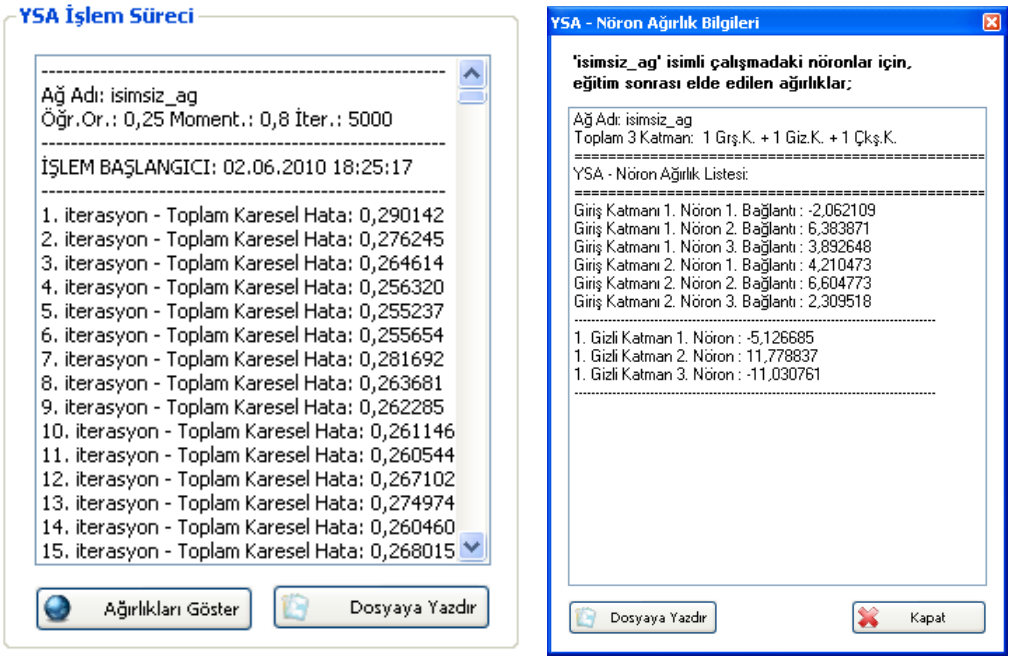
gelmektedir. “YSA İşlem Adımları” paneli altında sunulan kontroller, ağ sisteminin eğitimi esnasında aktif halde olan işlem adımını, ilgili başlığı seçili hale getirerek kullanıcıya bildirmektedir. Söz konusu işlem adımları “YSA Yapılandırma”, “Giriş Verilerinin Alımı”, “Beklenen Çıkış Verilerinin Alımı”, “Eğitim Setinin Hazırlanması”, “İleri Besleme” ve “Geriye Yayılım” şeklinde başlıklar halinde sunulmaktadır. Bir eğitim süreci esnasında, son iki başlık haricindeki diğer başlıklar birer kez aktif hale gelmekte, son iki başlık ise eğitim-öğrenme sürecinin uzunluğuna bağlı olarak, değişmeli bir şekilde aktif hale gelmektedir. İşlem adımları içerisinde gerçekleşen bu değişimler esnasında, yine “YSA İşlem Adımları” panelinin alt kısmında yer alan ve gerçekleşen işlemin durumu hakkında kullanıcıya bilgi veren “işlem durum çubuğu” (Progress Bar) kontrolü de otomatikman dolmaktadır. “YSA İşlem Adımları” panelinden bir ekran görüntüsü, Resim 4.25’de sunulmaktadır.



Resim 4.25 YSA İşlem Adımları panelinden bir ekran görüntüsü.

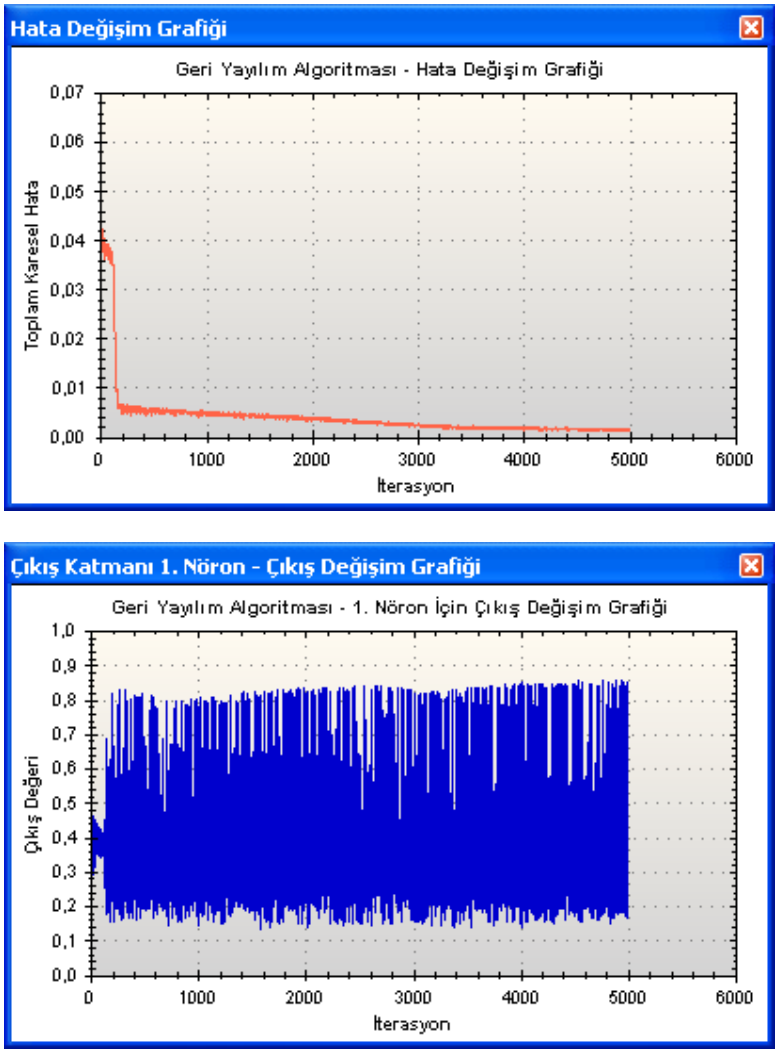
Aktif yapay sinir ağı sisteminin eğitilmesi sonucunda, eğitim süreci ile alakalı bazı sayısal bilgiler, “YSA İşlem Süreci” paneli altındaki kontroller aracılığıyla kullanıcıya sunulmaktadır. İlgili sayısal bilgiler, geri yayılım algoritması ile ilgili seçilen parametre değerlerini ve her yeni iterasyon için, toplam karesel hata değerlerini kapsamaktadır. Yine bu panelde yer alan “Ağırlıkları Göster” düğmesi kullanılarak, her katmandaki yapay sinir hücreleri için, eğitim sonunda elde edilen ağırlık değerlerini gösteren, “YSA – Nöron Ağırlık Bilgileri” penceresi görüntülenebilmektedir. Gerek “YSA İşlem Süreci” panelinde gösterilen bilgileri, gerekse “YSA – Nöron Ağırlık Bilgileri” penceresinde görüntülenen bilgileri, “Dosyaya Yazdır” düğmelerini kullanarak, .TXT

dosya formatında kaydetmek de mümkün olmaktadır. "YSA İşlem Süreci" paneli ve "YSA – Nöron Ağırlık Bilgileri" penceresi, Resim 4.26 altında gösterilmektedir.



Resim 4.26 YSA İşlem Süreci paneli ve YSA – Nöron Ağırlık Bilgileri penceresi.

Eğitim işleminin sona ermesi ile birlikte, yazılım sistemi tarafından otomatik olarak yerine getirilen bir işlev de, ilgili eğitim süreci kapsamında elde edilen hata değerindeki değişim ve çıkış katmanında yer alan her yapay sinir hücresi için elde edilen, çıkış değerindeki değişim grafiklerinin görüntülenmesidir. Söz konusu grafikler, eğitim süreci kapsamında meydana gelen değişimleri, görsel anlamda daha etkili ve anlaşılır bir biçimde kullanıcıya sunmaktadır. Grafikler vektörel bir yapıda sunulduğundan dolayı, kullanıcının yakınlaştırma / uzaklaştırma ve büyütme / küçültme gibi işlevleri kullanması yoluyla, daha detaylı bir şekilde incelenebilmektedir. Yine sunulan grafiklerin yazıcı çıktıları alınabilmekte, hatta farklı resim dosya formatları halinde kaydedilmeleri sağlanabilmektedir. İlgili işlevlere, grafikler üzerinde sağ tuş tıklandıktan sonra görüntülenen menüler üzerinden ulaşılabilmektedir. Örnek olması açısından, çıkış katmanında tek yapay sinir hücresine sahip bir ağ için, kısa bir eğitim süreci sonucu elde edilen hata ve çıkış değişim grafikleri Resim 4.27'de sunulmuştur.

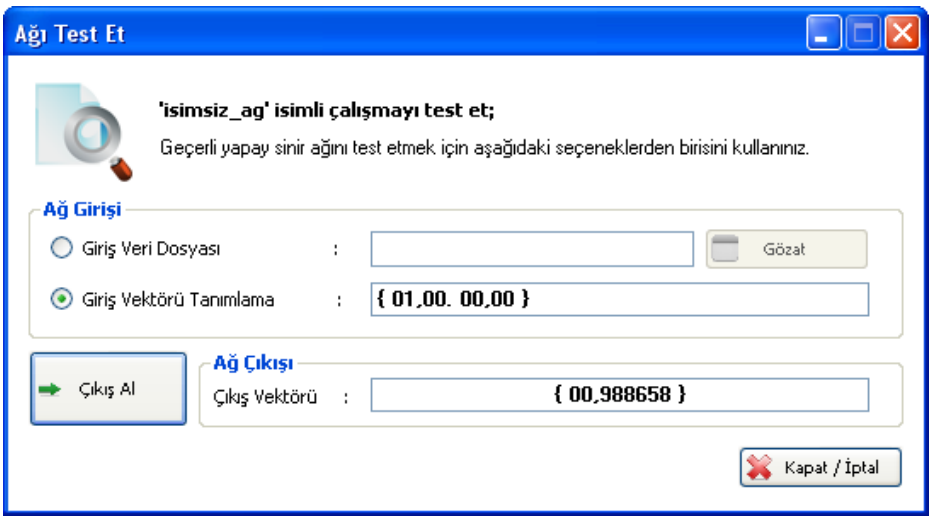


Resim 4.27 Örnek hata ve çıkış değişim grafikleri.

Kullanıcı, elde edilen hata değişim ve çıkış değişim grafiklerine daha sonra "YSA Sistem Araçları" paneli altında sunulan, "Grafik" düğmesi aracılığıyla ulaşabilmektedir. Eğitim sürecinin bitmesiyle birlikte aktif hale gelen, "YSA Sistem Araçları" panelindeki "Performans" ve "Test Et" düğmeleriyle, aktif durumdaki ağ sisteminin görsel performans bilgilerinin görüntülenmesi ve gerek harici veri dosyaları yardımıyla, gerekse yazılım içi sunulan kontroller yardımıyla, farklı giriş verilerine karşı ağ çıkış(lar)ının test edilmesi mümkündür. "Test Et" düğmesine tıklayarak görüntülenen, "Ağı Test Et" başlıklı pencere üzerinde, iki farklı seçenek üzerinden test işleminin gerçekleştirilmesi mümkün olmaktadır. İlgili pencerede sunulan "Giriş Veri Dosyası"

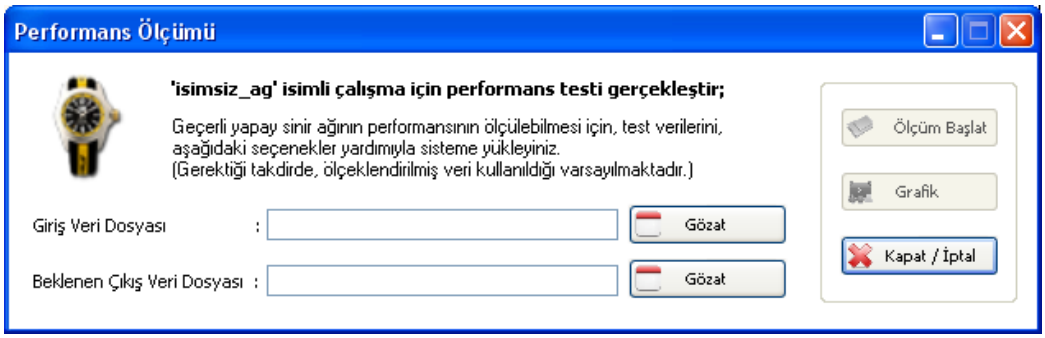
seçeneği yardımıyla, .XLS ya da .CSV dosya türündeki veriler, aktif ağ sistemini test etmek amacıyla sisteme tanıtılabilmektedir. Söz konusu dosya içeriklerinin hazırlanmasında, yine giriş ve beklenen çıkış veri dosyalarının hazırlanmasında uyulan kuralların geçerli olduğu, bu noktada ifade edilmelidir. Yine bu noktada, sisteme tanıtılan test dosyasının içeriğinin, aktif durumdaki ağ sisteminin yapısına göre kontrol edilmesi, herhangi bir sorun halinde de kullanıcının uyarılması, yazılım sistemi tarafından sunulan işlevlerden birisidir. Değinilmesi gereken diğer bir önemli husus ise, test amaçlı kullanılan dosyada kaç satır veri bulunursa bulunsun, sistemin sadece ilk satırı dikkate alarak test işlemini gerçekleştirmesidir.

Test işleminde uygulanabilen diğer bir yaklaşım da yine ilgili pencere üzerinde sunulan, "vektör tanımlama" işlevidir. Pencere üzerinde sunulan, "Giriş Vektörü Tanımlama" seçeneği yardımıyla, sistem tarafından, giriş katmanının yapay sinir hücresi yapısına göre hazırlanmış olan bir metin maskesi kullanılarak, test amaçlı giriş verisi sisteme kolaylıkla gönderebilmektedir. Açıklanan her iki test yaklaşımında, yine ilgili pencere üzerinde sunulan "Çıkış Al" düğmesi kullanılarak, aktif ağ sisteminden çıkış verisi ya da verileri hızlı bir şekilde alınabilmektedir. İlgili çıkış(lar), "Ağ Çıkışı" paneli altında sunulmuş olan, "Çıkış Vektörü" etiketli alanda, otomatikman görüntülenmektedir. Resim 4.28'de, söz konusu test penceresinden bir ekran görüntüsü gösterilmiştir.



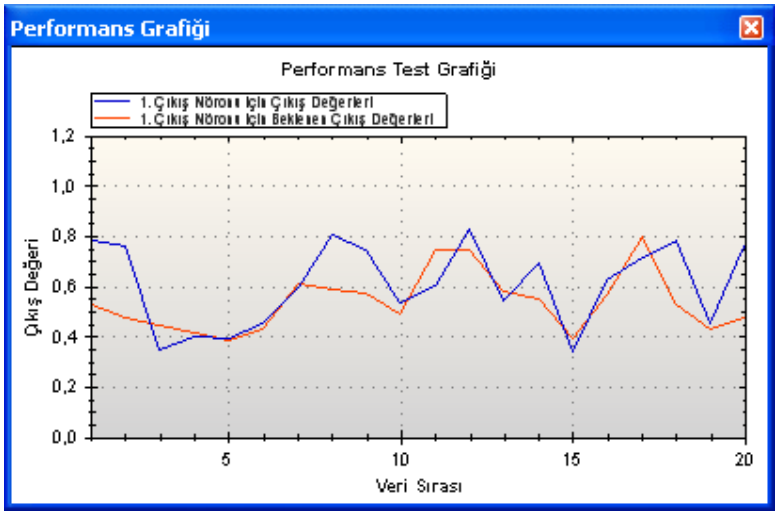
Resim 4.28 Yazılım bünyesinde sunulmuş test penceresinden bir ekran görüntüsü.

"Performans" düğmesi yardımıyla açılan "Performans Ölçümü" penceresi, eğitimi gerçekleştirilmiş olan ağ sistemine, görsel anlamda performans ölçümlerinin uygulanmasını sağlamaktadır. Bu noktada, pencere üzerinde sunulan kontroller kullanılarak, daha önce (eğitim esnasında) ağ sistemine verilmemiş olan çeşitli giriş verilerinin ve bu verilere bağlı olarak elde edilmesi beklenen çıkış verilerinin tanıtımı gerçekleştirilmekte ve bu veriler kapsamında elde edilen çıkışlarla, beklenen çıkışların sergilendiği bir grafik elde edilmektedir. Böylece, ilgili ağ sisteminin, "yeni" verilere karşı olan performansı, görsel anlamda kolaylıkla yorumlanabilmektedir. "Performans Ölçümü" penceresinden bir ekran görüntüsü, Resim 4.29'da sunulmaktadır.



Resim 4.29 Performans Ölçümü penceresi.

Söz konusu pencere üzerinde sunulan kontroller, tıpkı "Veri Dosyası Yükleme / Düzenleme" penceresinde olduğu gibi, performans ölçümünde kullanılacak giriş ve beklenen çıkış veri dosyalarının, ayrı şekillerde sisteme tanıtılmasını sağlamaktadır. Bu bağlamda, kullanılacak veri dosyaları yine .XLS veya .CSV dosya türlerinde sisteme yüklenmekte ve dosya içerikleri yine sistem tarafından otomatik olarak kontrol edilmektedir. Performans ölçümü, gerekli dosyalar sisteme tanıtıldıktan sonra aktif hale gelen, "Ölçüm Başlat" düğmesiyle yerine getirilmektedir. Söz konusu ölçüm sonrası, eğitilmiş durumdaki ağ sisteminde, çıkış katmanındaki her yapay sinir hücresi için elde edilen çıkış değerleri ve her değer için gerçekte elde edilmesi beklenen çıkış değerleri, farklı renkler kullanılarak, aynı grafik ortamı üzerinde görüntülenmektedir. Elde edilen son grafik, yine "Performans Ölçümü" penceresinde sunulmuş olan "Grafik" düğmesi yardımıyla, tekrar ekrana getirilebilmektedir. Çıkış katmanında tek yapay sinir hücresi olan bir ağ için, örnek bir performans ölçüm grafiği, Resim 4.30 altında sunulmaktadır.



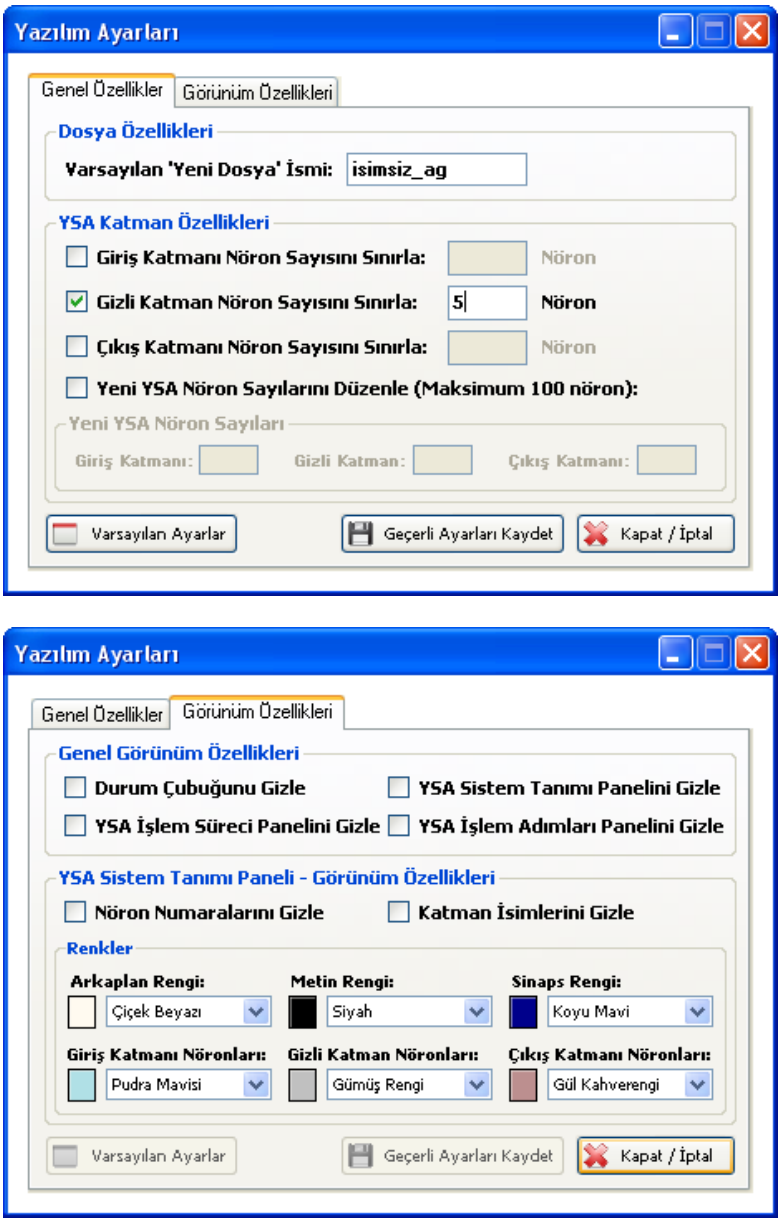
Resim 4.30 Örnek bir performans ölçüm grafiği.

Açıklanan kullanım özellikleri ve işlevleri, tipik bir yapay sinir ağı çalışması ya da uygulamasının, yapay sinir ağları uygulama ve eğitim yazılımı arayüzünde gerçekleştirilmesine dayalı olarak ifade edilmiştir. Bulanık mantık uygulama ve eğitim yazılımında olduğu gibi, yapay sinir ağları yazılımında da, bahsedilen özellik ve işlevlerin dışında, değinilmesi gereken, birtakım ek özellik ve işlevler de bulunmaktadır.

4.5.3 Yazılım Bünyesindeki Diğer Özellik ve İşlevler

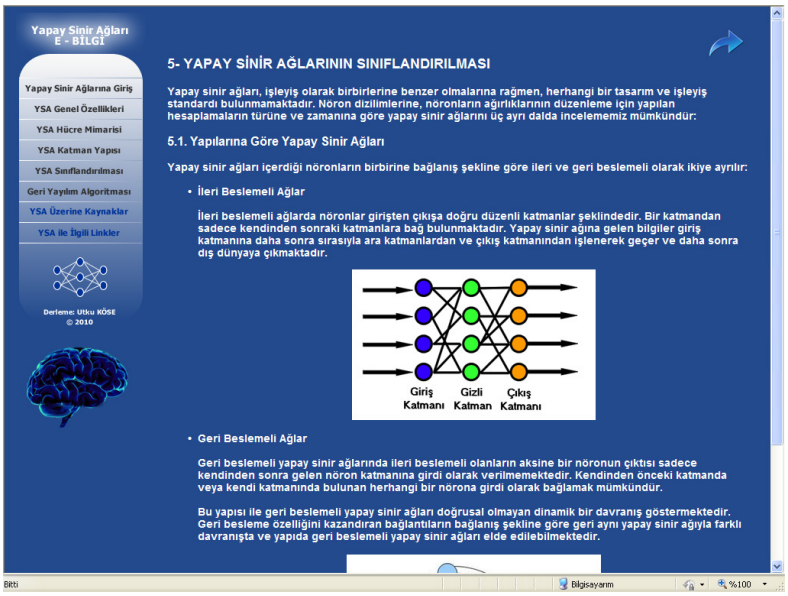
Varsayılan kullanım özelliklerine ek olarak, yazılımın bazı özellik ve işlevleri, yine kullanıcı tarafından ilgili kontroller yardımıyla ayarlanabilmektedir. Yapay sinir ağları uygulama ve eğitim yazılımının arayüzünde sunulmuş olan, "Eğitim Yazılımı Araçları" panelindeki "Yazılım Ayarları" düğmesi yardımıyla, söz konusu düzenlemelerin yapılabileceği bir pencere ekrana getirilebilmektedir. "Yazılım Ayarları" başlıklı bu pencere altında sunulmuş olan kontroller yardımıyla, yazılım bünyesinde kullanılan varsayılan dosya ismi, farklı katman türlerinde kullanılabilecek maksimum yapay sinir hücresi sayıları, yeni bir çalışma açıldığında aktif hale gelen ağ mimarisinde yer alacak olan yapay sinir hücrelerinin sayıları ve görsel anlamdaki diğer ayarlamalar (panellerin gizlenmesi, görsel ağ modelinde kullanılan renkler…vb.) kolaylıkla gerçekleştirilebilmektedir. İlgili ayarlamalar, pencere üzerinde "Genel Özellikler" ve

"Görünüm Özellikleri" başlıklı sekmeler altında sunulmuştur. Bu noktada önemli olan nokta, bu pencerede yapılan değişikliklerin, aktif haldeki ağ sistemini de etkilemesidir. Bu sebeple kullanıcı yazılım tarafından, değişikliklerin kaydedilmesi esnasında, çeşitli mesaj pencereleri aracılığıyla uyarılmaktadır. "Yazılım Ayarları" penceresindeki ilgili sekmeler, Resim 4.31'de gösterilmektedir.



Resim 4.31 Yazılım Ayarları penceresi ve ilgili sekmeler.

Söz konusu "Eğitim Yazılımı Araçları" panelinde sunulmuş olan, "YSA E-Bilgi" düğmesi ile birlikte, yapay sinir ağları tekniği ile alakalı temel prensiplerin ve geriye yayılım algoritmasının kısa bir şekilde anlatıldığı, ayrıca konuyla ilgili ek kaynakların verildiği, web sayfası formatında hazırlanmış olan, "YSA E-Bilgi" – HTML sayfası ekrana getirilebilmektedir. Bu sayfa, kullanıcıların yazılımı kullanırken, ilgili teknikle alakalı birtakım bilgilere kolayca ulaşmasını, öğrenmek istedikleri noktalarda, yine ilgili sayfadan bir başvuru kaynağı niteliğinde yararlanmalarını amaç edinmektedir. YSA E-Bilgi sayfasından örnek bir ekran görüntüsü, Resim 4.32'de sunulmaktadır.



Resim 4.32 YSA E-Bilgi – HTML sayfasından bir ekran görüntüsü.

Çalışma şekilleri açıklanan yapay sinir ağları uygulama ve eğitim yazılımının etkinliği ve doğruluğu konusunda daha fazla fikir sahibi olmak için, ilgili yazılım ile gerçekleştirilen çeşitli uygulamaların sonuçlarını değerlendirmek mümkündür. Bu bağlamda, ilgili yazılım ortamında iki farklı uygulama gerçekleştirilmiştir.

4.5.4 Yapay Sinir Ağları Uygulama ve Eğitim Yazılımı ile Örnek Çalışmalar

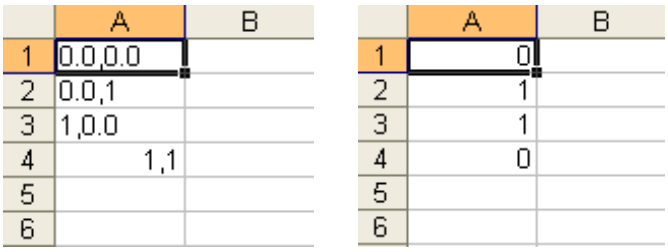
Söz konusu yazılım ortamındaki ilk uygulama, XOR mantık kapısı kurallarını yerine getiren bir yapay sinir ağı sisteminin oluşturulmasına dayalı olarak gerçekleştirilmiştir.

Bilindiği üzere, XOR mantık kapısında iki girişe karşılık bir çıkış elde edilmekte, bu kapsamda iki girişin aynı olması durumunda çıkış 0, aksi takdirde ise 1 değerini almaktadır. XOR mantık kapısının doğruluk tablosu, Çizelge 4.5'de gösterilmektedir.

Çizelge 4.5 XOR mantık kapısının doğruluk tablosu.

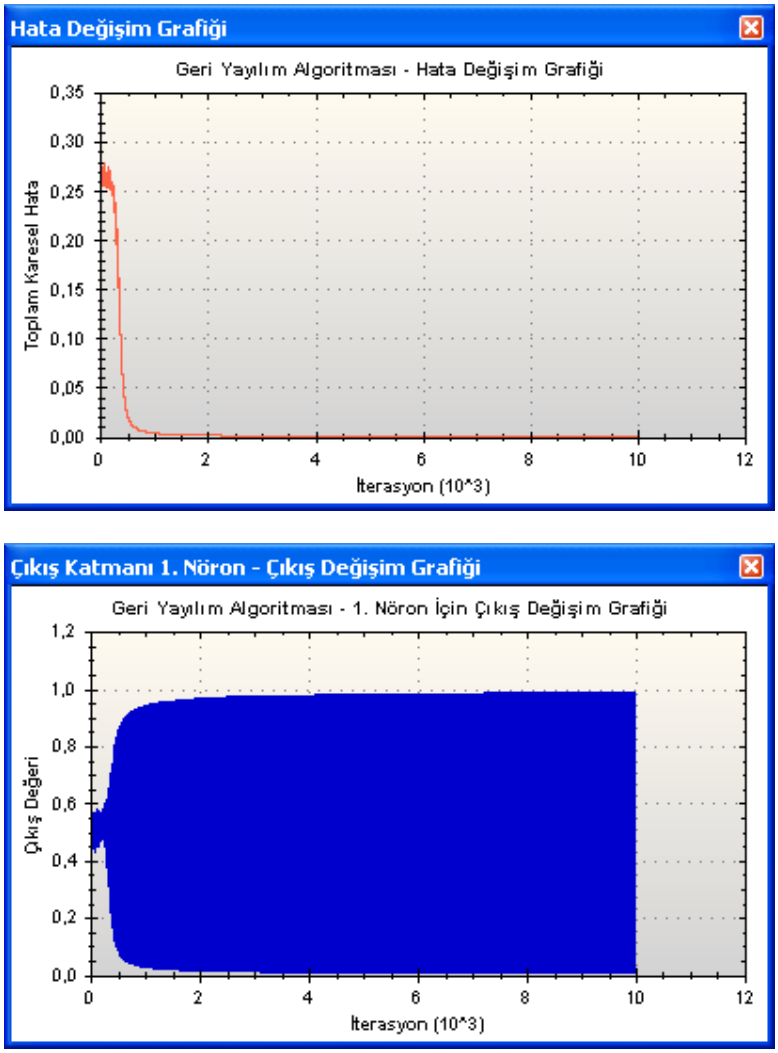
Giriş 1	**Giriş 2**	**Çıkış**
0	0	0
0	1	1
1	0	1
1	1	0

Problem bu kapsamda ele alındığında, tasarlanacak olan ağ mimarisinin, 2 yapay sinir hücresine sahip bir giriş katmanı ve 1 yapay sinir hücresine sahip çıkış katmanından oluşacağı değerlendirmesi yapılmıştır. Ek olarak, tasarlanan ağ mimarisinde, 3 yapay sinir hücresinden oluşan bir gizli katmana da yer verilmiştir. Bu noktada, gizli katman ve çıkış katmanlarının aktivasyon fonksiyonları "Sigmoid" olarak belirlenmiştir. Yapay sinir ağı mimarisinin oluşturulmasından sonra, sistemin XOR mantık kapısını öğrenebilmesi için gereken veri dosyalarının içeriği hazırlanmıştır. Buna göre, tasarlanan ağ sisteminin alacağı ikili verileri içeren bir "giriş veri dosyası" ve bu girişlere karşılık üretilmesi gereken çıkış verilerini gösteren bir adet de "beklenen çıkış veri dosyası" hazırlanmıştır. Resim 4.33, ilgili dosyalarının içeriğinden birer ekran görüntüsü sunmaktadır.



Resim 4.33 XOR uygulaması için "giriş" ve "beklenen çıkış" veri dosyaları.

İlgili veri dosyalarının tanıtımından sonra, geri yayılım algoritması kapsamında kullanılacak olan parametreler belirlenmiştir. Buna göre öğrenme oranı 0.25, momentum değeri 0.8 olarak belirlenmiş, eğitim için de 10000 iterasyon sınırı belirtilmiştir. Gerçekleştirilen eğitim sonrası elde edilen toplam karesel hata değeri 0.000174 olmuştur. İlgili eğitim süreciyle alakalı elde edilen hata değişim grafiği ve çıkış katmanındaki yapay sinir hücresine ait çıkış değişim grafiği, Resim 4.34'de gösterilmektedir.



Resim 4.34 XOR uygulamasında elde edilen hata ve çıkış değişim grafikleri.

Gerçekleştirilen eğitim süreci sonunda, tasarlanan sistemin ilgili problemi öğrenip öğrenmediğini test etmek amacıyla ilgili test penceresi kullanılmıştır. Hızlı sonuç almak adına, "Giriş Vektörü Tanımlama" kısmında belirtilen giriş vektörlerine karşılık, çıkış vektörü olarak elde edilen değerler, Çizelge 4.6'da gösterilmektedir.

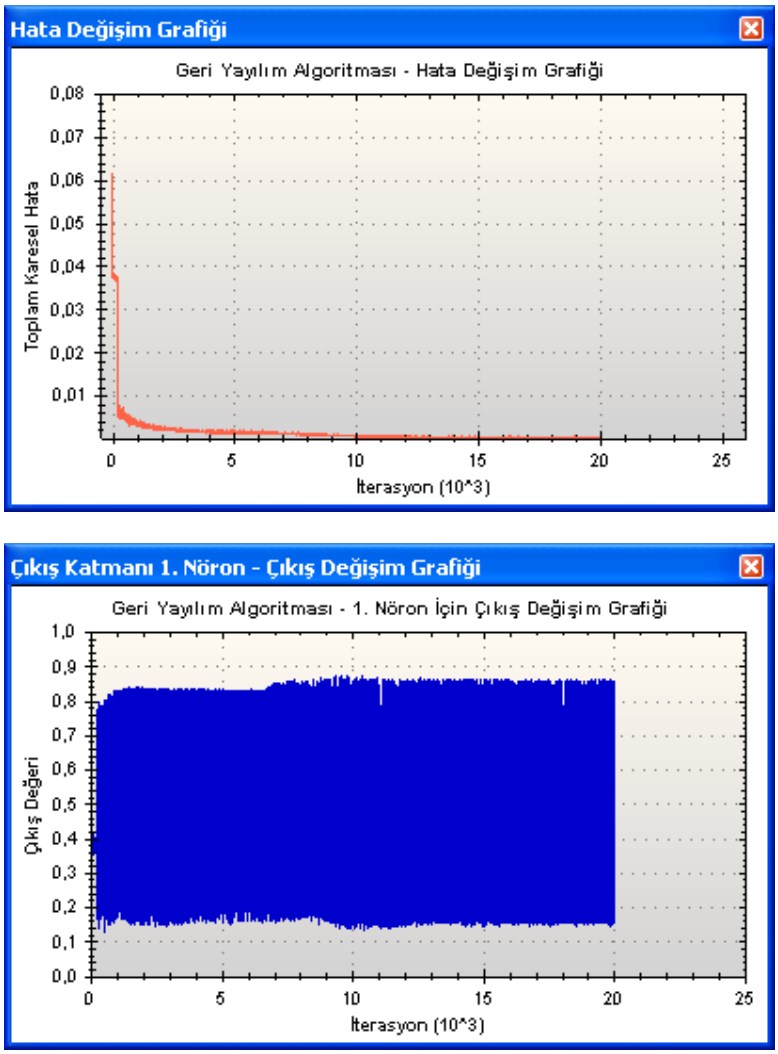
Çizelge 4.6 XOR uygulaması kapsamında elde edilen test sonuçları.

Giriş Vektörü	**Çıkış Vektörü**
{ 00,00. 00,00 }	{ 00,004057 }
{ 00,00. 01,00 }	{ 00,986095 }
{ 01,00. 00,00 }	{ 00,985617 }
{ 01,00. 01,00 }	{ 00,017239 }

Elde edilen sonuçlar da göstermektedir ki, elde edilen hata değeri dâhilinde, tasarlanmış olan yapay sinir ağı sistemi, söz konusu XOR mantık kapısı problemini, başarıyla öğrenmiştir.

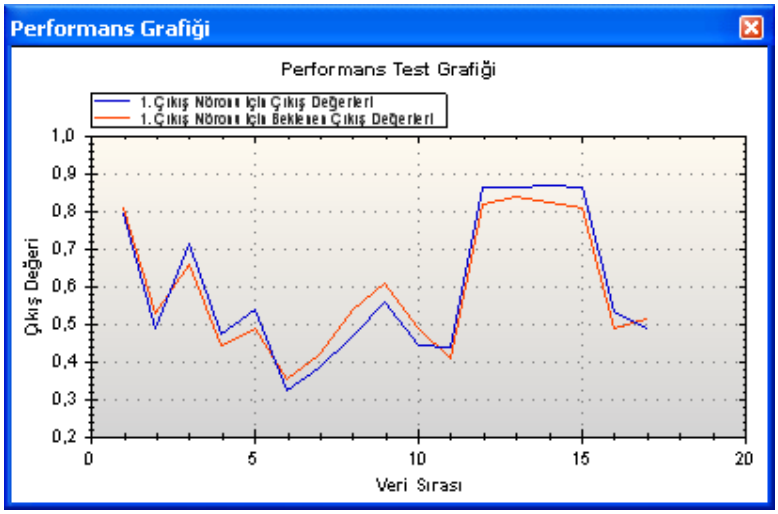
Yazılım ortamında gerçekleştirilen ikinci uygulamada, dört farklı parametre (su buharı basıncı, bağıl nem, rüzgâr şiddeti, hava basıncı) üzerinden, "sıcaklık" değeri tahmininde bulunan bir yapay sinir ağı sistemi oluşturulmuştur. Anlaşılacağı üzere, problem kapsamında oluşturulan yapay sinir ağı mimarisi, 4 yapay sinir hücresine sahip bir giriş katmanı ve 1 yapay sinir hücresine sahip çıkış katmanından oluşan bir sistem olarak ortaya çıkmıştır. Bunun dışında, söz konusu ağ mimarisinde, 5 adet yapay sinir hücrelerinden oluşan, üç gizli katmana da yer verilmiştir. XOR probleminde olduğu gibi, bu uygulama kapsamında oluşturulan ağ mimarisinde de, çıkış katmanı ve gizli katmanlar için "Sigmoid" aktivasyon fonksiyonu kullanılmıştır. İlgili mimari oluşturulduktan sonra, sistemin ele alınan problemi öğrenebilmesi için, toplam 33 adet veriden oluşan "giriş veri dosyası" ve bu giriş verilerine karşılık elde edilmesi gereken çıkışların belirtildiği, "beklenen çıkış veri dosyası" hazırlanmıştır. Dosyalar içerisinde yer alan veriler ölçeklendirilmemiş formatta olduğundan dolayı, "Veri Dosyası Yükleme / Düzenleme" penceresinde gerekli ölçeklendirme işlemleri de uygulanmıştır. Buna göre, ilgili girişler kapsamındaki olası en küçük ve olası en büyük değerler, "su

buharı basıncı" için 3 ve 13, "bağıl nem" için 50 ve 100, "rüzgâr şiddeti" için 1 ve 6, "hava basıncı" için ise 1000 ve 1025 değerleri olarak sisteme tanıtılmıştır. İlgili veri dosyalarının tanıtımı ve ölçeklendirmelerden sonra, geri yayılım algoritması kapsamında kullanılacak olan parametreler, sistem arayüzünde belirtilmiştir. Buna göre öğrenme oranı 0.75, momentum değeri 0.85 olarak belirlenmiş, eğitim durma kriteri için de 20000 iterasyon sınırı belirtilmiştir. Eğitim süreci sonrasında elde edilen toplam karesel hata değeri 0.000163 olmuştur. Söz konusu eğitim süreci kapsamında elde edilmiş olan hata değişim grafiği ve çıkış katmanındaki yapay sinir hücresine ait çıkış değişim grafiği, Resim 4.35'de gösterilmektedir.



Resim 4.35 Uygulama sonucu elde edilen hata ve çıkış değişim grafikleri.

Eğitimi gerçekleştirilen ağ sisteminin performansını gözlemlemek amacıyla, "Performans Ölçümü" penceresi üzerinden bir ölçüm işlemi gerçekleştirilmiştir. Bu bağlamda, söz konusu ağ sistemin, eğitim esnasında görmemiş olduğu 17 adet yeni veri, düzenlenen giriş ve beklenen çıkış veri dosyaları aracılığıyla yazılıma tanıtılmış ve işlem sonucunda performans ölçüm grafiği elde edilmiştir. Elde edilen grafik, Resim 4.36'da gösterilmektedir.



Resim 4.36 Uygulama kapsamında elde edilen performans ölçüm grafiği.

Söz konusu uygulamalar sonucunda elde edilen veriler göstermektedir ki, ele alınmış olan problemler, geliştirilen yapay sinir ağları uygulama ve eğitim yazılımı tarafından, doğru ve tutarlı bir biçimde değerlendirilmektedir. Bu açıdan incelendiğinde, yapay sinir ağları uygulama ve eğitim yazılımının, kullanıcılara etkin ve doğru ölçüde bir uygulama ortamı sağladığı değerlendirmesini yapmak mümkündür.

5. SONUÇ VE TARTIŞMA

Bu tez çalışması kapsamında geliştirilmiş olan uygulama yazılımı, bulanık mantık ve yapay sinir ağları tekniklerinin eğitiminde ve yine bu tekniklerle ilgili araştırma ve uygulama çalışmalarında kullanılabilen, yeni ve farklı bir uygulama platformunu ortaya koymaktadır. Geliştirilen yazılım, sunduğu etkileşimli ve görsel özellikler sayesinde, bulanık mantık ve yapay sinir ağları gibi teknik yönden ağır konularla ilgili çalışma ve uygulamalar gerçekleştiren kullanıcılara, kolay ve etkili bir çalışma ortamı vaat etmektedir. Yine ilgili yazılım, benzer nitelikteki yazılımlarla kıyaslandığında, eğitimsel kaygıları daha üst düzeyde tutulmuş, ancak bunu yaparken de teknik ve uygulamaya yönelik yaklaşımlardan taviz verilmemiş bir uygulama ve eğitim aracını gözler önüne sermektedir. Yazılım bünyesinde sunulan özellik ve işlevler, farklı bulanık kontrol sistemleri ve yapay sinir ağları sistemleri üzerinde çalışırken, söz konusu tekniklerle alakalı temel prensiplerin ve uygulama yaklaşımlarının da öğrenilmesini, ayrıca kazanılmış durumdaki bilgi ve becerilerin de, bilgisayar ortamında denenerek pekiştirilmesini sağlamaktadır. Yazılımın tamamının Türkçe tasarlanmış ve geliştirilmiş olması, teknik düzeydeki İngilizce yazılımları kullanmakta güçlük çeken kullanıcıların da, geliştirilen yazılım ortamında daha kolay çalışmasına imkân sağlamaktadır. Daha önce de ifade edildiği üzere söz konusu yazılım, bu açıdan bakıldığı takdirde, Türkçe bir uygulama platformu sunan ender çalışmalardan birisi olarak da ilgili literatürdeki yerini almaktadır.

Geliştirilen yazılımın en önemli karakteristiği, piyasada bulunan benzer yazılımların karmaşık düzeydeki, önemli işlev ve özelliklerini, kullanıcıya daha basit şekillerde sunuyor olması, ayrıca bu özelliklerin yanında, kullanım şekillerini hızlandıracak ve ön bilgi sahibi olmaksızın, uygulamaların daha kolay yapılmasını sağlayacak bazı özellik ve işlevleri de beraberinde getiriyor olmasıdır. Söz konusu bu özellik ve işlevler, piyasada bulunan ya da araştırma çalışmaları kapsamında geliştirilmiş olan benzer uygulamalar tarafından "tam anlamıyla" sunulmamaktadır. Diğer yandan, elbette ki geliştirilen yazılımın sunduğu özellik ve işlevler, piyasada bulunan, büyük çaptaki ve teknik anlamda daha ileri düzeydeki uygulama yazılımlarına göre basit bir yapıda sunulmaktadır. Ancak bu noktada geliştirilen yazılım, çalışmanın amaçları

doğrultusunda ihtiyaç duyulan ve bu bağlamda yeterli olduğu düşünülen, teknik açıdan olduğu kadar, eğitimsel açıdan da katkı sağlayacağı değerlendirilen, öncelikli özellik ve işlevleri kullanıcılara sunmaktadır. Sunulan bu özellik ve işlevler, söz konusu yazılımın, basit düzeydeki çalışma ve uygulamalarda olduğu kadar, ileri düzeydeki çalışma ve uygulamalarda, etkili bir şekilde kullanılmasına da imkân vermektedir. Bu kapsamda daha detaylı değerlendirildiği zaman, geliştirilen yazılım, ilgili tekniklerle alakalı çalışma ve uygulamalar gerçekleştiren araştırmacılar ve bilim adamları tarafından olduğu kadar, bu konularla alakalı olan herkes tarafından kullanılabilmektedir. Daha da önemlisi yazılım, lisans ve lisansüstü düzeydeki elektrik, elektronik ve bilgisayar mühendisliği gibi mühendislik bilimleri ile bilgisayar bilimleri kapsamındaki diğer bölümlerde, bulanık mantık kontrol sistemleri, yapay sinir ağları ve yapay zekâ teknikleri ile bağlantılı derslerde, eğitimsel anlamda yardımcı bir araç olarak da yine kullanılabilmektedir.

Yazılım üzerinde gerçekleştirilen örnek çalışmalar, gerek bulanık mantığa dayalı uygulamalarda, gerekse yapay sinir ağlarına dayalı uygulamalarda, geliştirilen yazılım ile birlikte, oldukça tutarlı ve doğru sonuçların elde edildiğini göstermiştir. Yazılım bu bağlamda değerlendirildiğinde, uygulamalar sonucunda elde edilen değerlerin ve sonuçların, piyasada bulunan gelişmiş yazılımlarla elde edilen değer ve sonuçlarla aynı olduğunu gözlemlenmektedir. Böylelikle, geliştirilen yazılımın teknik ve matematiksel anlamda doğru ve tutarlı bir yapıda tasarlanmış ve programlanmış olduğu ifade edilebilmektedir. Bu noktada değinilmesi gereken diğer önemli bir nokta da, geliştirilen yazılım bünyesindeki bulanık mantık ve yapay sinir ağları alt yapısının, geniş bir uygulama alanını kapsar nitelikte olmasıdır. Bulanık mantık altyapısı ile birlikte, matematiksel fonksiyonu “A.x + B.u” şeklinde ifade edilebilen bütün sistemler, bulanık kontrol yaklaşımı kapsamında, yazılım bünyesinde uygulanabilmektedir. Diğer yandan, yaygın bir kullanım şekline sahip olan, çok katmanlı, ileri beslemeli yapay sinir ağı sistemleri de, ilgili mimari ve geri yayılım algoritması ile çözülebilen her türlü problem için, geliştirilmiş olan yapay sinir ağları altyapısı ile birlikte, yazılım bünyesinde uygulanabilmektedir.

Söz konusu yazılımın, daha farklı özellik ve işlevlerin eklenmesi suretiyle geliştirilmesi, tasarımsal ve yapısal anlamda yine mümkündür. Buna göre yazılım bünyesinde sunulan bulanık mantık altyapısı ile yapay sinir ağları altyapısı, uygun yaklaşımlar ve düzenlemeler sonucunda "uyarlamalı sinirsel-bulanık sistem" (ANFIS) uygulamaları için uygun bir platform haline getirilebilecek bir düzende sunulmaktadır. Bunun dışında, yazılımla birlikte gelen bulanık mantık model desteğinin, Mamdani ve Takagi-Sugeno türleri dışında diğerleri için de genişletilmesi yine imkân dâhilinde bahsedilebilmektedir. Diğer yandan, yapay sinir ağlarının da farklı mimarilere uyarlanabilir hale getirilmesi de yine geliştirilmiş olan altyapı düzeni sayesinde mümkün olarak gözükmektedir. Bahsedilen bütün bu geliştirmelerin dışında, çalışmanın amaçları doğrultusunda oluşturulan yazılım sistemine, ilerleyen sürümlerle birlikte, farklı yapay zekâ tekniklerine ait uygulama ve eğitim ortamlarının eklenmesi, yazılım kapsamının ilgili alanda genişletilmesi ve böylelikle daha büyük bir uygulama ortamının sunulması açısından önemli bir adım olarak değerlendirilecektir.

6. KAYNAKLAR

Basheer, I.A. and Hajmeer, M., 2000, "Artificial neural networks: fundamentals, computing, design and application", Journal of Microbiological Methods, December, Vol.43, pp. 3-31

Brown, M. and Harris, C., 1995, "Neurofuzzy Adaptive Modeling and Control", Prentice-Hall Int., 1. edition, Cambridge, UK.

Castillo, O., Luis, T.A., Cardenas, S., 2006, "Fuzzy logic tracking control for unicycle mobile robots", Engineering Letters, August, Vol.13, pp. 73-77

Çetin, M., Uğur, A., Bayzan, Ş., 2006, "İleri Beslemeli Yapay Sinir Ağlarında Backpropagation (Geriye Yayılım) Algoritmasının Sezgisel Yaklaşımı", Akademik Bilişim 2006 (AB 2006), Pamukkale Üniversitesi, Denizli, 9-11 Şubat.

Deperlioğlu, Ö., 2001, "Bir Anahtarlama Kipli Konvertörün Sinirsel-Bulanık Denetimi", Doktora Tezi, Gazi Üniversitesi, Fen Bilimleri Enstitüsü, Ankara.

Drew, P.J. and Monson, J.R.T., 2000, "Artificial neural networks", Surgery, January, Vol.127, pp. 3-11

ECMA 2006, C# Language Specification / ECMA-334 Standard, Çevrimiçi: http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-334.pdf (Erişim Tarihi: 21.12.2009).

Elmas, Ç., 2003a, "Bulanık Mantık Denetleyiciler", Seçkin Yayıncılık, 1. Baskı, Ankara, Türkiye.

Elmas, Ç., 2003b, "Yapay Sinir Ağları (Kuram, Mimari, Eğitim, Uygulama)", Seçkin Yayıncılık, 1. Baskı, Ankara, Türkiye.

Fuller, R., 1999, "On Fuzzy Reasoning Schemes", Chapter of The State of The Art of Information Systems in 2007, TUCS (Turku Centre for Computer Science) General Publications, Turku, 85 p.

Hejlsberg, A., 2008, The A-Z of Programming Languages: C#, Computerworld, Çevrimiçi: http://www.computerworld.com.au/article/261958/a-z_programming_languages_c_/?pp=7 (Erişim Tarihi: 11.01.2010).

Jang, J.-S.R., Sun, C.-T., Mizutani, E., 1997, Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence, Prentice-Hall, US edition, New Jersey, USA.

Jantzen, J., 1998, "Design of Fuzzy Controllers", Technical Report (No: 98-E 864), Technical University of Denmark, Department of Automation, Lyngby, Denmark, Çevrimiçi: http://faculty.petra.ac.id/resmana/private/fuzzy/design.pdf (Erişim Tarihi: 26.12.2009).

Kıyak, E. ve Kahvecioğlu, A., 2003, "Bulanık mantık ve uçuş kontrol problemine uygulanması", Havacılık ve Uzay Teknolojileri Dergisi, Temmuz, Cilt1, syf. 63-72

Kosko, B., 1992, "Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence", Prentice-Hall, Har/Dsk edition, New Jersey, USA.

Kosko, B., 1997, "Fuzzy Engineering", Prentice-Hall, 1. edition, New Jersey, USA.

Lee, C.C., 1990, "Fuzzy logic in control systems: Fuzzy logic controller-Part I / Part II", IEEE Transactions on Systems, Man, and Cybernetics, March/April, Vol.20, pp. 404-435

Lin, C.-T. and George Lee, C.S., 1996, "Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems", Prentice-Hall, Har/Dsk edition, New Jersey, USA.

Luger, G.F., 2002, "Artificial Intelligence: Structures and Strategies for Complex Problem Solving", Addison-Wesley, 4. edition, Massachusetts, USA.

McCulloch, W.S. and Pitts, W., 1990, "A logical calculus of the ideas immanent in nervous activity", Bulletin of Mathematical Biology, January, Vol.52, pp. 99-115 [Reprinted from the Bulletin of Mathematical Biophysics, Vol.5, pp. 115-133 (1943)]

Nabiyev, V.V., 2005, "Yapay Zeka – Problemler, Yöntemler, Algoritma", Seçkin Yayıncılık, 2. Baskı, Ankara, Türkiye.

Naugler, D., 2007, "C# 2.0 for C++ and Java programmer: Conference workshop", Journal of Computing Sciences in Collages, May, Vol. 22, pp. 1-1

Nilsson, N.J., 1998, "Artificial Intelligence: A New Synthesis", Morgan Kaufmann Publishers, 1. edition, San Francisco, USA.

Öz, C., Köker, R., Çakar, S., 2002, "Yapay Sinir Ağları ile Karakter Tabanlı Plaka Tanıma", Elektrik, Elektronik ve Bilgisayar Mühendisliği Sempozyumu 2002 (ELECO'2002), Uludağ Üniversitesi, Bursa, 18-22 Aralık, 322-326.

Öztemel, E., 2006, "Yapay Sinir Ağları", Papatya Yayıncılık, 2. Baskı, İstanbul, Türkiye.

Passino, K.M. and Yurkovich, S., 1997, "Fuzzy Control", Addison-Wesley, 1. edition, Massachusetts, USA.

Pillay, A., 2007, "Object Oriented Programming using Java", Notes for the Computer Science Module: Object Oriented Programming – COMP200, University of KwaZulu-Natal, Durban, South Africa, Çevrimiçi: http://math.hws.edu/eck/cs124/downloads/OOP2_from_Univ_KwaZulu-Natal.pdf (Erişim Tarihi: 03.01.2010).

Russell, S.J. and Norvig, P., 1995, "Artificial Intelligence: A Modern Approach", Prentice-Hall, 1. edition, New Jersey, USA.

Stanek, W.R., 2002, "XML Pocket Consultant", Microsoft Press, 1. edition, Washington, USA.

Svenk, G., 2003, "Object-Oriented Programming: Using C++ for Engineering and Technology", Delmar – Thomson Learning, 2. edition, Toronto, Canada.

Uğur, A. ve Kınacı, A.C., 2005, "İnternet Üzerinde Yapay Zeka ve Yapay Sinir Ağları", COMPOTEK 2005 Bilişim Seminerleri Programı, İzmir, 16-20 Kasım.

Uğur, A. ve Kınacı, A.C., 2006, "Yapay Zeka Teknikleri ve Yapay Sinir Ağları Kullanılarak Web Sayfalarının Sınıflandırılması", INET-TR 2006, XI. Türkiye'de İnternet Konferansı, TOBB Ekonomik ve Teknoloji Üniversitesi, Ankara, 21-23 Aralık.

Watts, M., Woodford, B.J., Kasabov, N., 1999, "FuzzyCOPE: A Software Environment for Building Intelligent Systems – The Past, The Present and The Future", ICONIP/ANZIIS/ANNES'99 Workshop, Dunedin/Queenstown, New Zealand, 22-23 November, 188-192.

Whitby, B., 2003, "Artificial Intelligence: A Beginner's Guide", Oneworld Publications, 1. edition, Oxford, UK.

Wierman, M.J., 2008, “Fuzzy Sets, Fuzzy Logic, and Control”, Textbook, 1. edition, Nebraska, USA, Çevrimiçi: http://duck.creighton.edu/download/FuzzyMath.pdf (Erişim Tarihi: 21.12.2009).

Wirfs-Brock, R., Wilkerson, B., Wiener, L., 1990, “Designing Object-Oriented Software”, Prentice-Hall, 1. edition, New Jersey, USA.

Wong, W., 2002, Why Microsoft’s C# isn’t?, CNET News, Çevrimiçi: http://news.cnet.com/2008-1082-817522.html (Erişim Tarihi: 13.01.2010).

Yan, J., Ryan, M., Power, J., 1994, “Using Fuzzy Logic: Towards Intelligent Systems”, Prentice-Hall Int., 1. edition, Cambridge, UK.

Yegnanarayana, B., 2006, “Artificial Neural Networks”, Prentice-Hall of India Pvt. Ltd., 2. edition, New Delhi, India.

Zhao, R. and Govind, R., 1991, “Defuzzification of fuzzy intervals”, Fuzzy Sets and Systems, September, Vol.43, pp. 45-55

6.1 İnternet Kaynakları **Erişim Tarihi**

1. http://www.fuzzytech.com/ 21.01.2010
2. http://www.havana7.com/dotfuzzy/ 22.01.2010
3. http://www.mathworks.com/products/fuzzylogic/ 23.01.2010
4. http://ssdi.di.fct.unl.pt/scl/docs/texts/fuzzy logic toolbox.pdf 23.01.2010
5. http://fuzzy.cs.uni-magdeburg.de/archive/pub/nefclass/manual.pdf 24.01.2010
6. http://jfuzzylogic.sourceforge.net/html/ 24.01.2010
7. http://ffll.sourceforge.net/ 25.01.2010
8. http://www.aforgenet.com/framework/ 27.01.2010
9. http://www.mathworks.com/access/helpdesk/help/pdf_doc/nnet/nnet.pdf 28.01.2010
10. http://www.mathworks.com/products/neuralnet/description3.html 28.01.2010
11. http://en.wikipedia.org/wiki/NeuroSolutions 29.01.2010
12. http://www.statsoft.com/textbook/neural-networks/?button=2 02.02.2010
13. http://media.wolfram.com/documents/NeuralNetworksDocumentation.pdf 03.02.2010
14. http://www.ire.pw.edu.pl/~rsulej/NetMaker/ 05.02.2010
15. http://en.wikipedia.org/wiki/JOONE 06.02.2010
16. http://www.heatonresearch.com/encog 08.02.2010
17. http://leenissen.dk/fann/ 08.02.2010
18. http://tr.wikipedia.org/wiki/C_Sharp_(programlama_dili) 11.02.2010
19. http://en.wikipedia.org/wiki/C_Sharp_(programming_language) 14.02.2010
20. http://zedgraph.org/wiki/index.php 17.02.2010
21. http://tr.wikipedia.org/wiki/Genişletilebilir_işaretleme_dili 23.02.2010
22. http://tr.wikipedia.org/wiki/Nesne_Yönelimli_Programlama 02.03.2010
23. http://en.wikipedia.org/wiki/Object-oriented_programming 02.03.2010

ÖZGEÇMİŞ

Adı Soyadı : Utku KÖSE

Doğum Yeri : Afyon

Doğum Tarihi : 26.03.1985

Medeni Hali : Bekâr

Yabancı Dili : İngilizce

Eğitim Durumu (Kurum ve Yıl)

Lise : Afyon Gazi Anadolu Meslek Lisesi (1999–2003)

Lisans : Gazi Üniversitesi, Endüstriyel Sanatlar Eğitim Fakültesi, Bilgisayar Eğitimi (2004–2008)

Yüksek Lisans: Afyon Kocatepe Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Anabilim Dalı (2008–……)

Çalıştığı Kurum/Kurumlar ve Yıl

Afyon Kocatepe Üniversitesi, Bilgi İşlem Daire Başkanlığı (2009–2009)

Afyon Kocatepe Üniversitesi, Uzaktan Eğitim Meslek Yüksekokulu (2009–……)