T.C.
MALTEPE ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİLİĞİ ANABİLİM DALI

KISITLANMIŞ BOLTZMANN MAKİNESİ İLE ZAMAN SERİLERİNİN TAHMİNİ

Mehmet PEKMEZCİ

Yüksek Lisans Tezi

Tez Danışmanı

Yrd. Doç. Dr. Ali AKMAN

İSTANBUL 2012

Bu tez çalışması, Maltepe Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 06/02/2012 tarih ve 2012/02 sayılı kararıyla oluşturulan jüri tarafından **Bilgisayar Mühendisliği Tezli Yüksek Lisansı Tezi** olarak kabul edilmiştir.

JÜRİ

Doç Dr. Raif ÖNVURAL
Üye

Yrd. Doç. Dr. Ali AKMAN
Üye
(Danışman)

Yrd. Doç. Dr. Sibel ÇEVİK ERSAN
Üye

ÖZET

Yüksek Lisans Tezi, Kısıtlanmış Boltzmann Makinesi ile Zaman Serilerinin Tahmini, T.C. Maltepe Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı.

Bu çalışmada bir yapay sinir ağı (YSA) çeşidi olan Kısıtlanmış Boltzmann Makinesi ile zaman serileri tahmini yapılmıştır. Karşılaştırma amaçlı olarak da, standart bilinen İleri Yönelimli YSA ile de tahmin yapılmıştır ve karşılaştırma sonuçları bu çalışma içinde grafik olarak verilmiştir.

Bu çalışma için yazılmış olan kodlara "http://java-crbm-backprop-timeseries-sample.googlecode.com/svn/trunk/" adresinden ulaşılabilmektedir. Programı bilgisayara indirip Test sınıfını çalıştırarak karşılaştırma sonuçlarına ulaşılabilmektedir. Bu kodlar yazılırken "jaRBM (Ben Chung,2009)" adlı açık kaynak kodlu yazılım kütüphanesinden yararlanılmıştır.

Bu tez 2012 yılında yapılmıştır ve 73 sayfadan oluşmaktadır.

Anahtar Kelimeler: Boltzmann Makinesi, İleri Yönelimli Yapay Sinir Ağları, Zaman Serileri Tahmini.

ABSTRACT

MASTER THESIS, Time Series Prediction Using Restricted Boltzmann Machine, T.C. Maltepe University, Graduate School of Natural and Applied Sciences, Department of Computer Enginnering.

In this work, a type of artificial neural network called “Restricted Boltzmann Machine” is used to predict time series. Another type of artificial neural network called “Feed Forward Network” is also used to predict the time series to be able to compare prediction results. Prediction results and their comparison are given as both table and graphical representation.

The software codes written within this work can be reached at “http://java-crbm-backprop-timeseries-sample.googlecode.com/svn/trunk/” It is sufficient to download the program and run the “Test” class in the program to obtain the prediction results, graphics and their comparison tables. Some code snippets of the open source java project “jaRBM (Ben Chung,2009)” is also used in the software codes.

This thesis written in 2012 and consists of 73 pages.

Keywords: Boltzmann Machine, Feed Forward Artificial Neural Networks, Time Series Prediction.

TEŞEKKÜR SAYFASI

Yüksek lisans tezimi hazırlamamda bana yardım eden, zaman ayıran, her önerdiğim fikri ciddiye alıp benimle tartışan ve her koşulda destek olan tez danışmanım, hocam Yrd. Doç. Dr. Ali Akman'a, tez konum ile beni ilk tanıştıran hocam Yrd. Doç. Dr. Turgay T. Bilgin'e, her bildiğini tam bilip dışarıya da bunu yansıtabilmenin önemini anlatan, her konuda çok iyi referans kitaplar öneren ve matematiğe daha da ısınmamı sağlayan hocam Prof. Dr. İlhami Yavuz'a, bu konuda çalışma yapmam için beni cesaretlendiren ve yönlendiren Türk Deniz Kuvvetleri Araştırma Merkezi Komutanlığı personeline, çalışma arkadaşlarıma ve aileme sonsuz teşekkürlerimi sunarım.

İçindekiler

KISALTMALAR LİSTESİ

ANN	Artificial Neural Network
AR	Auto Regressive
ARIMA	Auto Regressive, Integrated, Moving Average
ARMA	Auto Regressive, Moving Average
MA	Moving Average
YSA	Yapay Sinir Ağı

ŞEKİLLER LİSTESİ

ÇİZELGELER LİSTESİ

1 GİRİŞ

Çevremizde oluşan olayların bir sonraki adımını tahmin edebilmek çoğu zaman çok önemli hatalardan korunmamızı sağlar. Örneğin bir hisse senedinin gidişatını tahmin edebilirsek yanlış bir yatırım yapmaktan kurtulabiliriz veya birçok uçağın inip kalktığı bir havaalanında birbirlerine yakın iniş/kalkış yapan uçakların yörüngelerini tahmin edebilirsek bu uçakların çarpışma ihtimallerini en aza indirgeyebiliriz.

Her olayın bir sonraki adımı tahmin edilemeyebilir ancak sınırlı sayıda parametreye bağlı olayların birbiri ardına dizilişini tahmin edebilmek, tam kesinlikle olmasa bile, mümkündür. Bizim ulaşmak istediğimiz nokta da tam kesinlikle olmasa bile makul bir yakınsama ile bu olayları tahmin etmektir.

Bir olayı eşit zaman aralıklarıyla gözledikten sonra bir veya birkaç adım sonrasını tahmin etmeye "Zaman Serileri Analizi" veya "Zaman Serileri Tahmini" denmektedir (Hamilton,1994). Zaman serileri analizi yöntemi, fizikte olduğu gibi parametrelerin tamamının bilindiği, örneğin eğik atış, olayları incelemez. Zaman serileri analizi, parametreleri bilinmeyen bir olayı bir matematiksel modele oturtarak parametrelerini belirlemeye çalışır. .

Bu yöntem, "Sonlu Elemanlar Yöntemi" gibi benzetim (simulation) yöntemleriyle de karıştırılmamalıdır. Benzetim yöntemlerinde başlangıçta gözlem verisi yoktur ama olayın genel denklemleri bilinmektedir, bu denklemler kullanılarak sayısal yöntemlerle adımlar ilerletilir ve sonuçları gözlemlenir. İstatistikî benzetim yöntemlerinde ise benzetim işlemine gelmeden çok önce veriler incelenir ve bir istatistiksel dağılım/model çıkarılır, daha sonra benzetim aşamasına geçilir ve bütün sistem bu dağılım/modele göre ilerletilir. Zaman serileri analizinde ise daha önceden yapılmış bir kısım gözlem vardır, gözlem verileri bu yöntemle işlenir ve ortaya bir model/dağılım çıkarılır, daha sonra da yöntemin içinde ileriye yönelik kestirim yapılır.

Zaman serileri analizi için birçok matematiksel araç kullanılmaktadır. Bu araçlardan son zamanlarda üzerinde en çok araştırma yapılanlardan birisi de YSA'lardır. Bu çalışma içerisinde en yeni YSA türlerinden biri olan "Kısıtlanmış Boltzmann Makinesi" (Restricted Boltzmann Machine) (Smolensky, 1986) zaman

serileri analizinde kullanılacaktır. Kısıtlanmış Boltzmann Makinesi stokastik özyinelemeli YSA'dır (Stochastic recursive neural network) (Smolensky, 1986). Diğer YSA'lara göre daha yeni olan Kısıtlanmış Boltzmann Makinesi ile çalışırken en çok Goeffrey Hinton'un yazdığı makalelerden yararlandım.

1.1 PROBLEMİN TANIMI

Zaman serileri (Time Series) , eşit zaman aralıklarında yapılan ölçümlerden elde edilen sayıların oluşturduğu dizilerdir (Hamilton,1994). Örnek olarak çizelge 1-1'de verilen zaman serisi değerleri bir uçağın zaman içindeki yükseklik değerleridir.

Zaman	Yükseklik
1. dakika	100 metre
2. dakika	450 metre
3. dakika	700 metre
4. dakika	900 metre

Çizelge 1-1 Örnek Zaman Serisi

Zaman serilerindeki veriler arasında doğal bir ilişki olması gerektiğini varsayarak, gelecekteki verileri tahmin etmeye **Zaman Serileri Analizi / Tahmini** denmektedir (Hamilton,1994).

Zaman serilerinin analizi/tahmini genel olarak aşağıdaki yöntemlerle yapılabilmektedir:

1. Doğrusal Yöntemler (AR, MA, ARMA, ARIMA,..)
2. Saklı Markov Modelleri (Hidden Markov Models)
3. Tekil Spektrum Analizi (Singular Spectrum Analysis)
4. Zaman-Sıklığı İnceleme Yöntemleri

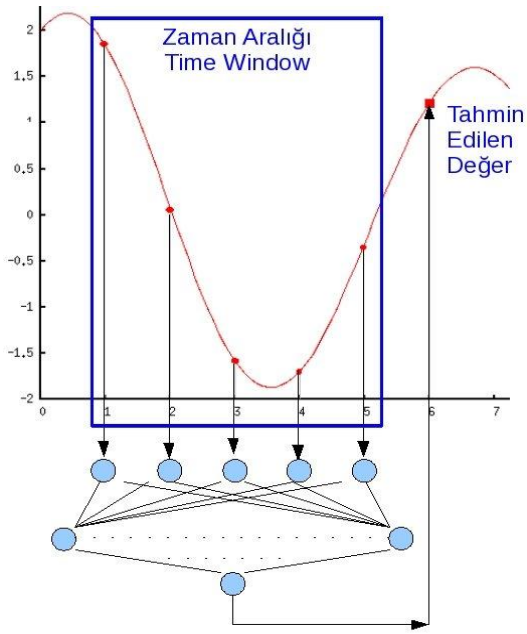
   (Time-Frequency Analysis Techniques)
5. YSA'lar

Bu çalışmada YSA'ların aşağıda belirtilen 2 çeşidi kullanılarak zaman serileri tahmini yapılmaktadır. Bu iki YSA, hata değerleri ve tutarlılık açısından karşılaştırılmaktadır:

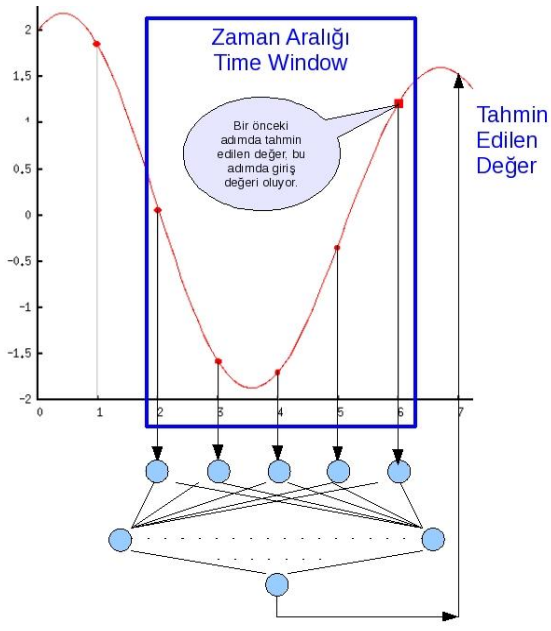
1.**İleri Yönelimli Çok Katmanlı YSA** (Multilayer Feedforward Artificial Neural Network), başka bir deyişle "**Çok Katmanlı Algılayıcı**" (Multi Layer Perceptron, **MLP**)

2.**Kısıtlanmış Boltzmann Makinesi** (Restricted Boltzmann Machine)

YSA'ları zaman serileri verileriyle eğitirken ve tahmin yaparken "**Zaman Aralığı**" (Time Window) kavramı kullanılır. "**Zaman Aralığı**" , sabit sayıda zaman verisi demektir. [32] Bu zaman aralığı 1 kaydırılarak YSA'lar eğitilir ve tahmin yaptırılır. [15] Örneğin, şekil 1-1'deki grafikte ilk önce 1'den 5'e kadar olan sayılar ile eğitilip 6. saniyedeki sayı tahmin edilmekte. Daha sonra şekil 1-2'deki gibi, bu 6. saniye verisi, giriş verisi olarak bir sonraki zaman aralığının son elemanı olarak kullanılmaktadır.



Şekil 1-1 Zaman Aralığı 1. Adım [35]



Şekil 1-2 Zaman Aralığı 2. Adım [35]

Sonuç olarak bu çalışmada aşağıdaki hedeflere ulaşılmaya çalışılmaktadır:

1. Her iki YSA çeşidi için (**İleri Yönelimli Çok Katmanlı YSA** ve **Kısıtlanmış Boltzmann Makinesi**) zaman serileri tahmini yapan algoritmalar kurulup işletilecektir, yapılan tahminler grafiklerle gösterilecektir.
2. Aşağıda belirtilen YSA parametreleri değiştirilerek, her iki YSA için de tahmin sonuçlarına etkisi incelenecektir.
    1. Tahmin yapılmadan hemen önce toplanan verinin büyüklüğü
    2. Saklı hücrelerin sayısı
    3. YSA'larda kullanılan girdinin büyüklüğü.
3. Aynı veri kümesi 4'er kez kullanılıp, her seferinde aynı tahminleri elde edip etmediğimiz araştırılacaktır.

2 KAYNAK ARAŞTIRMASI

Bu çalışma kapsamında sırasıyla zaman serileri analizi, YSA'lar, YSA'lar ile zaman serileri tahmini, Kısıtlanmış Boltzmann Makinesi konuları araştırılmıştır.

2.1 Zaman Serileri

Box-Jenkins 1994 yılında "*Time Series Analysis Forcasting And Control"* adlı kitabında istatistikî yöntemler ile (AR, MA, ARMA, ARIMA gibi) zaman serileri analizi anlatmıştır. Ancak bu kitapta YSA'lar ile ilgili bir çalışma yapılmamıştır.[8]

2.2 YSA'lar

YSA'lar ile ilgili ilk çalışmalar 1943 yılında McCulloch ve Pitts tarafından yapılmıştır. Bu çalışmada YSA'yı oluşturan ilk yapay sinir hücresi tanımlanmış ve bununla dağıtık mimaride hesaplama çalışmaları yapılmıştır (McCulloch & Pitts 1943). Günümüz YSA'ları için bir önemli dönüm noktası da 1969 yılında Minsky ve Papert'in yayınladığı "Perceptron" adlı makaledir. Bu makalede günümüz

YSA'larında kullanılan yapay sinir hücrelerinin yapısı anlatılmıştır (Minsky & Papert, 1969).

Kröse ve Smagt'ın 1996 basımı "An Introduction to Neural Networks" adlı kitabında temel YSA bilgileri verilmekte ve YSA'ların robotik ve görme konularında uygulamaları yapılmaktadır. [29]

Simon Haykin'in 1999 basımı "Neural Networks, A Comprehensive Foundation" adlı kitapta günümüz YSA'larının birçok çeşidi temelleriyle birlikte anlatılmaktadır. Bu tez kapsamında kullanılan çoğu şekil ve YSA tanımı Simon Haykin'in kitabından alıntıdır. Ayrıca gerçeklemesini (implementation) yaptığım Hata Düzelterek Öğrenme (Back Propagation) algoritması da bu kitaptan alıntıdır. [22]

2.3 YSA'lar İle Zaman Serileri Analizi

YSA'lar ile zaman serileri analizinin ilk örneklerinden biri 1993 yılında Tang ve Fishwick tarafından yazılmış olan "Feed-forward Neural Nets as Models for Time Series Forecasting" adlı makaledir. Bu makalede şimdi en çok bilinen "Geriye Yayılım" (Back Propagation) öğrenme algoritması kullanılarak ve istatistikî bir yöntem olan Box-Jenkins Yöntemi kullanılarak kestirim deneyleri yapılmış, sonuçlar tutarlılık açısından ve kestirim yakınsaması açısından karşılaştırılmıştır. Sonuç olarak YSA'nın parametrelerinin (saklı hücre sayısı, öğrenme katsayısı …) düzgün ayarlandığı durumlarda YSA ile yapılan kestirim denemelerinin hep daha iyi çıktığı belirtilmiştir. [17]

1995 yılında Payeur Le-Huy ve Gosselin'in yazdığı "Trajectory Prediction for Moving Objects Using Artificial Neural Networks" adlı makalede bir robotun haraketli bir cismi algılayıp yakalayabilmesi için YSA'ları kullanarak gerçek zamanlı kestirim yaptığında, cisimlerin kübik hareket denklemlerinin bilinerek hesaplandığında elde edilen sonuç kadar iyi sonuç verdiği görülmüştür. [39]

Raul Rojas'ın yazdığı 1996 basımı "Neural Networks, A Systematic

Introduction" adlı kitapta YSA'ların zaman serilerinin analizine uygulanışı anlatılmıştır. [42]

2001 yılında Hamzaçebi ve Kutay'ın yazdığı "Yapay Sinir Ağları ile Türkiye Elektrik Enerjisi Tüketiminin 2010 Yılına Kadar Tahmini" adlı makalesinde Türkiye elektrik enerjisi tüketiminin 10 yıllık kestirimi yapılmıştır. Kestirimler Box-Jenkins ve YSA ile yapılmış ve hangi yöntemin daha iyi sonuç verdiği tartışılmıştır.[31]

2006 yılında Rabunal ve Dorado'nun yazdığı "Artificial Neural Networks In Real Life Applications" adlı kitapta çok katmanlı algılayıcılar (Multi Layer Perceptrons) kullanılarak zaman serileri tahmin yöntemleri tartışılmaktadır. 10 değişik zaman serisi veri tipi için 10 değişik YSA mimarisi önerilmektedir. Önerilen her mimaride YSA "Çok Katmanlı Algılayıcı" tipindedir ancak bu ağ içindeki sinir hücrelerinin birbirlerine bağlanış şekilleri farklıdır. Ayrıca bu bölümde genel olarak hangi mimariye nasıl karar verileceği de anlatılmaktadır.[41]

2.4 Kısıtlanmış Boltzmann Makinesi

1985 yılında Ackley, Hinton ve Sejnowski tarafından yazılan "A Learning Algorithm for Boltzmann Machines" adlı makalede Boltzmann Makinesi için öğrenme algoritmasının genel prensipleri verilmiştir. Sonuç olarak fizikteki simetri prensibine uyan hücreler arası bağlantılar sayesinde iteratif kısıtları gerçekleştirmenin Boltzmann Makinesi ile mümkün olduğu gösterilmiştir.[1]

"Using Boltzmann Machines for Probability Estimation : A general Framework for Neural Network Learning" adlı makalesiyle 1993 yılında Kappen, Boltzmann Makineleri ile bileşik olasılık dağılımlarının tahmin edilebileceğini göstermiştir. En basit model olarak da "Boltzmann Perceptron" modeli ile bir uygulama gerçekleştirmiştir.[28]

Hinton'un kendi örün(web) sitesinde yayınladığı Brookes Üniversitesindeki 2006 yılı ders notlarından "Contrastive Divergence" adlı sunumunda ise Kısıtlanmış

Boltzmann Makinesinde kullandığımız öğrenme algoritmasının matematiksel altyapısı çok açık bir dille anlatılmıştır.[26] 2007 yılında Hinton, Salakhutdinov ve Mnih "Restricted Boltzmann Machines for Collaborative Filtering" adlı makalede Kısıtlanmış Boltzmann Makinesini ve öğrenme algoritması olarak Zıt Iraksamayı (Contrastive Divergence) kullanarak örnek bir filtre geliştirmiştir. Sonuç olarak bu algoritmaları kullanan ve yüz milyonun üzerinde veri ile daha öncekilere göre %6 daha iyi başarım oranıyla çalışan bir sistem yapılmıştır.[43]

2011 yılında Goeffrey Hinton'un kendi örün(web) sitesinde yayınladığı "A Practical Guide to Training Restricted Boltzmann Machines" adlı yazısında Kısıtlanmış Boltzmann Makinesinin eğitilmesini ayrıntılı şekilde açıklamıştır.[24]

3 YAPAY SİNİR AĞLARI

YSA'lar yapay sinir hücrelerinden oluşmaktadır. Ayrıca her YSA çeşidinin kendine özgü öğrenme yöntemi bulunmaktadır. Kısıtlanmış Boltzmann Makinesi de bir YSA'dır ve kendine özgü bir öğrenme yöntemi bulunmaktadır. YSA'ları ve öğrenme algoritmalarını inceleyerek uygulama tarafındaki hataların ve iyileştirmelerin nedenlerini daha net anlayabilmekteyiz.

3.1 Yapay Sinir Hücresi

Yapay Sinir Hücresi gerçek sinir hücresinin yaptığı sinyal işleme işlemini kabaca yapan matematiksel bir fonksiyondur (McCulloch & Pitts 1943). 1969 yılındaki Minsky ve Papert'in "Perceptron" adlı makalesinden sonra yapay sinir hücrelerine "Perceptron" da denmektedir (Minsky-Papert, 1969).

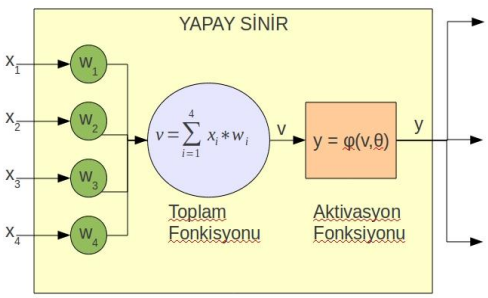
Yapay sinir hücresinin yapısı aşağıdaki gibidir:

1. m adet giriş değeri bulunmaktadır. Bu giriş değerleri herhangi bir reel sayı olabilmektedir. $(x_1 ,x_2 ,....,x_m)$
2. Her giriş için bir ağırlık değeri bulunur ve bu giriş sinyalinin ne kadar önemli olduğunu belirtir. $(w_1 ,w_2 ,....,w_m)$
3. Giriş değerleri ve ağırlık değerlerinin çarpılıp toplandığı bir toplam

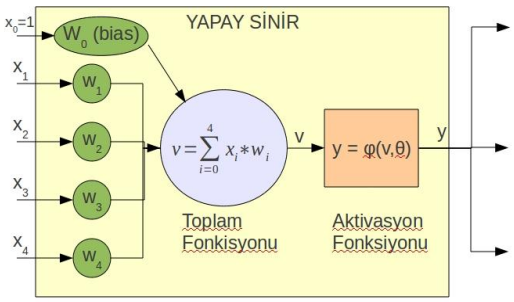
fonksiyonu bulunur.

$$v = \sum_{i=1}^{m} x_i . w_i \qquad 3\text{-}1$$

4. Bulunan bu toplam Aktivasyon (Transfer) Fonksiyonuna girer. Bu fonksiyon, θ aktivasyon katsayısına (treshold) bakarak, çıkış değerini kararlaştırır. Bu aktivasyon katsayısı genelde sabit bir sayıdır. $y = \varphi(v, \theta)$
5. Bir adet çıktısı bulunur. Çıkış değeri genelde 0 ile 1 aralığında olmakla beraber, herhangi bir reel sayı olabilmektedir. Çıkış değeri y'dir.
6. Birden fazla sinir hücresine bağlı olsa bile aynı çıktı değeri hepsine gönderilir.
7. Yapay sinirin hafızası onun ağırlık (weight) değerleridir.
8. Bir yapay sinir hücresini eğitmek, bu sinir hücresinin ağırlık (weight) değerlerinin uygun şekilde değiştirilmesi/düzeltilmesi demektir.
9. Bazen yapay sinirlere elle düzeltme verilmesi gerekebilir (örneğin doğrusal kayma durumlarında). Bu durumda "v" toplam değerinden "Düzeltme Katsayısı" (Bias) çıkarılır ve bu şekilde aktivasyon fonksiyonuna gönderilir.



Şekil 3-1 Yapay Sinir Hücresi[22]



Şekil 3-2 Düzeltme Katsayılı (Bias'lı) Yapay Sinir Hücresi[22]

Toplam Fonksiyonu, girdiler (X) ve ağırlıklarının (W) çarpımlarının toplamıdır, "v" harfi ile gösterilir. Aktivasyon Fonksiyonu, toplam fonksiyonundan alınan toplam değerini belli bir matematiksel fonksiyona tabi tutup, aktivasyon katsayısına da bakarak, çıktının ne olacağına karar verir. Genel olarak 3 tip aktivasyon fonksiyonu kullanılır. Bunlar "Basamak Fonksiyonu", "Doğrusal Kombinasyon" ve "Sigmoid Fonksiyonu" olarak adlandırılır.[22]

3.2 YSA'lar (Artificial Neural Networks)

Yapay sinir hücrelerinin girdilerinin ve çıktılarının birbirlerine bağlanmış halidir. Bugünkü YSA'lara en yakın sayabileceğimiz ilk YSA türü olan ve tek sinir hücresinden oluşan "**Perceptron**" 1969 yılında aynı adlı makalede Minsky ve Papert tarafından bizlere tanıtılmıştır. Bu yapay sinir hücresi aktivasyon fonksiyonu olarak "Basamak Fonksiyonu" kullanmaktadır (0/1) .

YSA'ların çalışma mantığı sırasıyla aşağıdaki gibidir:

1. YSA'lar eğitilir: İçindeki parametreler düzgün şekilde çalışma anında (runtime) belirlenir.
2. YSA aldığı eğitime göre verilen problemi çözer

Standart programlama mantığında ise YSA eğitimi yerine fonksiyonun parametreleri bellidir ve program bu şekilde direk olarak problem çözmeye başlar. YSA'ları eğitme yöntemleri, bir yapay sinir hücresinin hangi prensibe göre eğitileceğini ve bütün YSA'ya nasıl uygulanacağını söylerler.

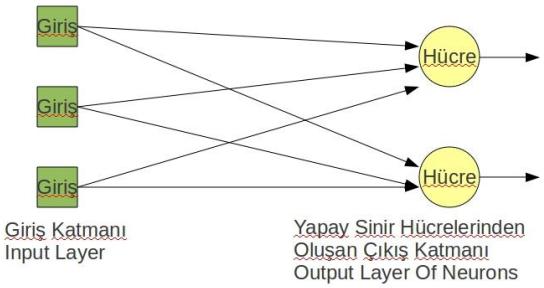
3.3 YSA'ların Bağlanma Çeşitleri

YSA'daki yapay sinir hücrelerinin birbirlerine bağlanma çeşitlerine göre öğrenme algoritmaları, uygulama alanları, öğrenme sığaları (capacity) değişmektedir.

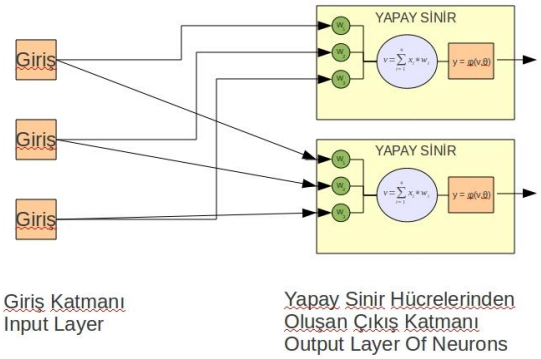
Genel olarak tek katmanlı ve çok katmanlı olarak ayrılan bağ çeşitleri, aynı zamanda kendi içlerinde de ileri yönelimli ve geri yönelimli (özyinelemeli, recursive) olarak ayrılmaktadır [22].

3.3.1 Tek katmanlı YSA'lar

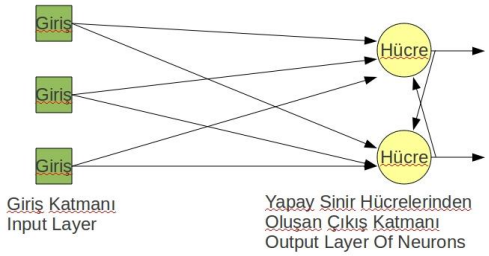
Bu tür bir yapılanmada yapay sinir hücreleri tek katman olarak dizilirler. Her bir yapay sinir hücresi birçok girdiye sahip olabilir. Genelde bütün sinir hücreleri aynı sayıda girdiye sahip olmalarına karşın kuramsal olarak böyle bir zorunluluk yoktur. Eğer yapay sinir hücreleri arasında bir bağ yoksa yani birinin çıktısı diğerinin girdi değerlerinden birini oluşturmuyorsa, buna "İleri Yönelimli Tek Katmanlı YSA" denir (Şekil 3-3 ve Şekil 3-4). Eğer yapay sinir hücreleri arasında bir bağ var ise, yani birinin çıktısı diğerinin girdi değerlerinden birini oluşturuyorsa, bu tür yapay sinir hücrelerinin oluşturduğu ağa "Geri Yönelimli Tek Katmanlı YSA" veya "Özyinelemeli Tek Katmanlı YSA" denmektedir (Şekil 3-5, Şekil 3-6) [3].



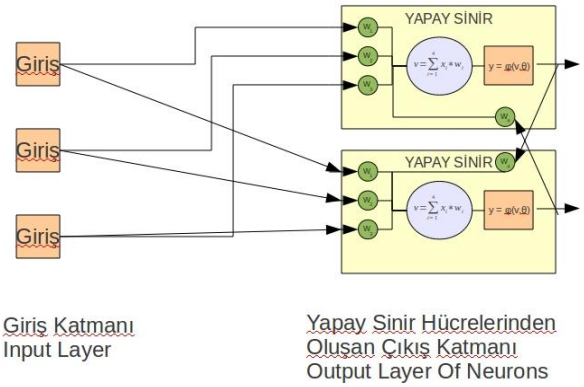
Şekil 3-3 İleri Yönelimli Tek Katmanlı YSA[3]



Şekil 3-4 İleri Yönelimli Tek Katmanlı YSA - Ayrıntılı



Şekil 3-5 Geri Yönelimli (Özyinelemeli) Tek Katmanlı YSA [3]

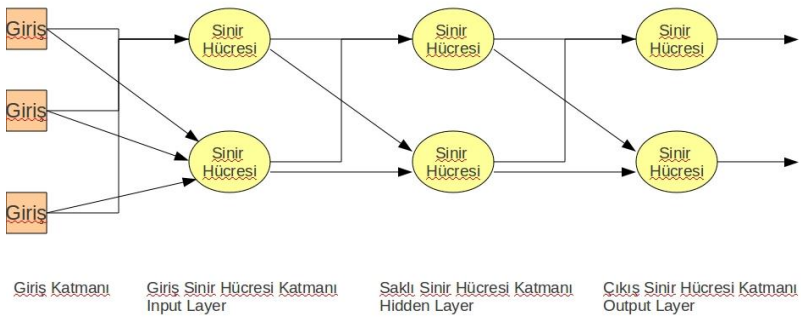


Şekil 3-6 Geri Yönelimli (Özyinelemeli) Tek Katmanlı YSA - Ayrıntılı

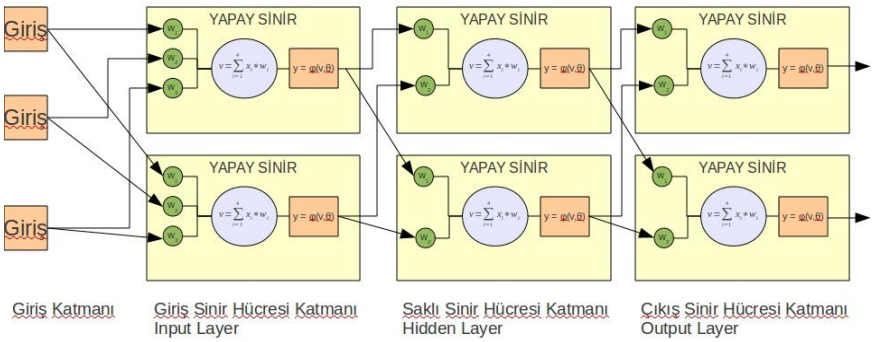
3.3.2 Çok Katmanlı YSA'lar

Bu tür bir yapılanmada yapay sinir hücreleri çok katman halinde dizilirler. Her bir yapay sinir hücresi birçok girdiye sahip olabilir. Genelde bütün sinir hücreleri aynı sayıda girdiye sahip olmalarına karşın kuramsal olarak böyle bir zorunluluk yoktur. Çok katmanlı bir yapı olduğu için her yapay sinir hücresi mutlaka başka bir yapay sinir hücresine bağlıdır. Eğer yapay sinir hücreleri sadece ileri yönde bir bağ oluşturuyorsa, buna "İleri Yönelimli Çok Katmanlı YSA" (Feed-Forward Neural Network) veya "Çok Katmanlı Perceptron" (Multi Level Perceptron) denir (Şekil 3-7 ve Şekil 3-8). Aynı katmanda bulunan sinir hücrelerinin çıktıları birbirlerini etkilemez. [3]

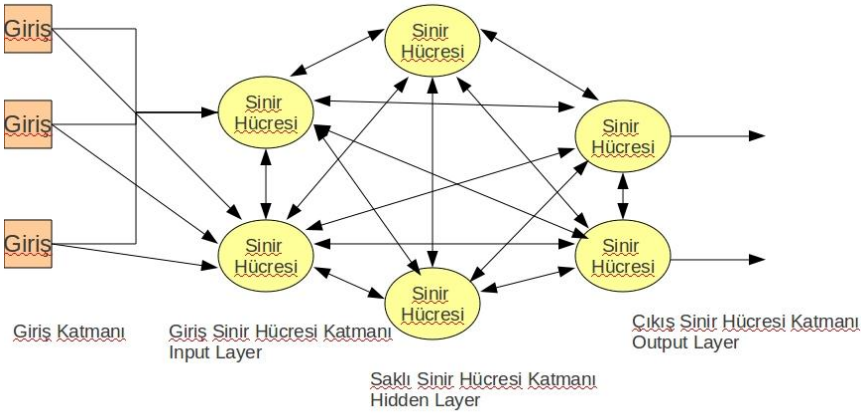
Eğer yapay sinir hücreleri arasındaki bağlar iki yönlü ise ki bu durumda çizge (graph) kuramına göre bir çevrim (cycle) oluşturmaktadırlar, bu tür yapay sinir hücrelerine "Geri Yönelimli Çok Katmanlı YSA" veya "Özyinelemeli Çok Katmanlı YSA" (Recurrent Neural Network) denmektedir (Şekil 3-9, Şekil 3-10). [3] Bu tür YSA'lara örnek olarak Boltzmann Makinesi, Hopfield Ağları, Self Organizing Maps verilebilir. Sinir hücrelerinin bir kısmı giriş için bir kısmı çıkış için kullanılabilir.



Şekil 3-7 Çok Katmanlı İleri Yönelimli YSA[3]



Şekil 3-8 Çok Katmanlı İleri Yönelimli YSA – Ayrıntılı



Şekil 3-9 Özyinelemeli (Geri Yönelimli) YSA [3]

Yukarıda tanımı yapılan bütün YSA çeşitlerinde, bütün sinir hücreleri birbirlerine bağlı (full connected) olmak zorunda değildir. Hücreler arası bağlar istenildiği gibi ayarlanabilmektedir. Örneğin Boltzmann Makinesinde bütün hücreler birbirine bağlıdır, ancak Kısıtlanmış Boltzmann Makinesinde (Restricted Boltzmann Machine) aynı katmanda bulunan sinir hücreleri arası bağa izin verilmez, böylelikle öğrenme algoritmasının daha hızlı çalışması sağlanır. Ayrıca, ağ içinde yapay sinir hücreleri kendilerine de bağ yapabilmektedirler. Ancak karmaşıklıklarından dolayı, çoğunlukla bu yapı kullanılmamaktadır.[3]

3.4 YSA'ları Eğitme Yöntemleri (Training Methods) (Öğrenme Algoritmaları)

Bir YSA örnek olarak örüntü ilişkilendirmeyi (Pattern Association), örüntü tanımayı (Pattern Recognition), fonksiyon yakınsamayı (Function Approximation), süreç kontrolünü (Process Control), gürültülü veriyi süzmeyi (Filtering of Noisy Data) öğrenebilir. Bu işlemleri bir YSA'nın öğrenmesi demek, her bir gözlem verisi için YSA içindeki sinir hücrelerinin birbirleri ile olan bağlarındaki ağrılık değerlerinin (weight) belirlenmesi demektir.[36]

YSA'ları eğitme yöntemleri aşağıdaki gibidir [22] :

1. Yönlendirilmiş (Supervised)
    1. Hata Düzelterek Öğrenme (Error-Correcting Leraning)
    2. Hafıza Tabanlı Öğrenme (Memory Based Learning)
2. Yönlendirilmemiş (Unsupervised)
    1. Hebb Öğrenmesi (Hebbian Learning)
    2. Rekabete Dayalı Öğrenme (Competitive Learning)

       Örnek: Self Organizing Maps (SOM)
    3. Boltzmann Öğrenmesi (Boltzmann Learning)

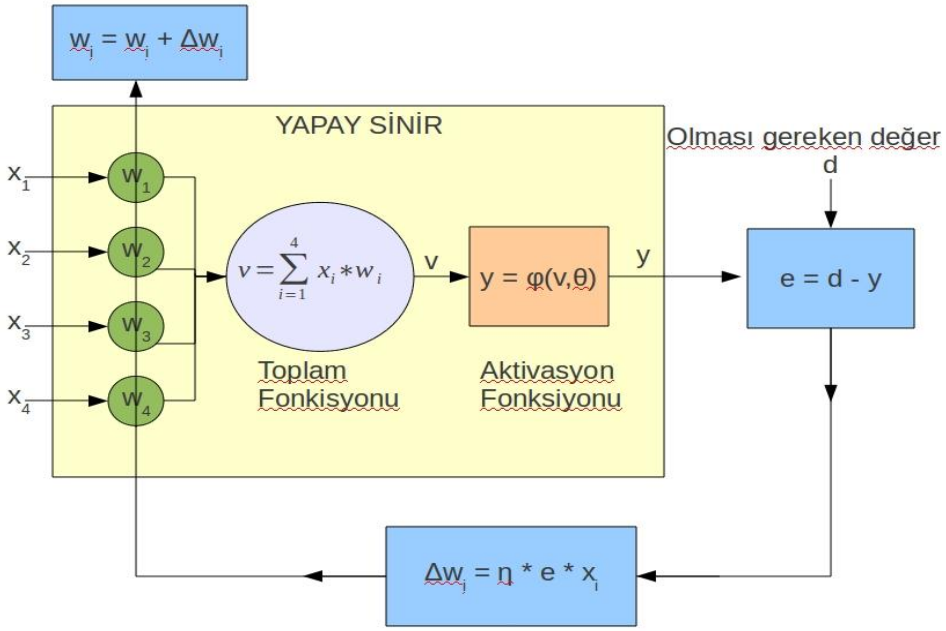
       Örnek: Boltzmann Makinesi, Kısıtlanmış Boltzmann Makinesi (Restricted Boltzmann Machine)
    4. Reinforcement Learning

       Örnek: Neuro Dynamic Programming

Yukarıda sayılan öğrenme yöntemleri arasından sadece Hata Düzelterek Öğrenme ve Boltzmann Öğrenmesi detaylı olarak anlatılacaktır.

3.4.1 Hata Düzelterek Öğrenme (Error-Correcting Learning, Back Propagation)

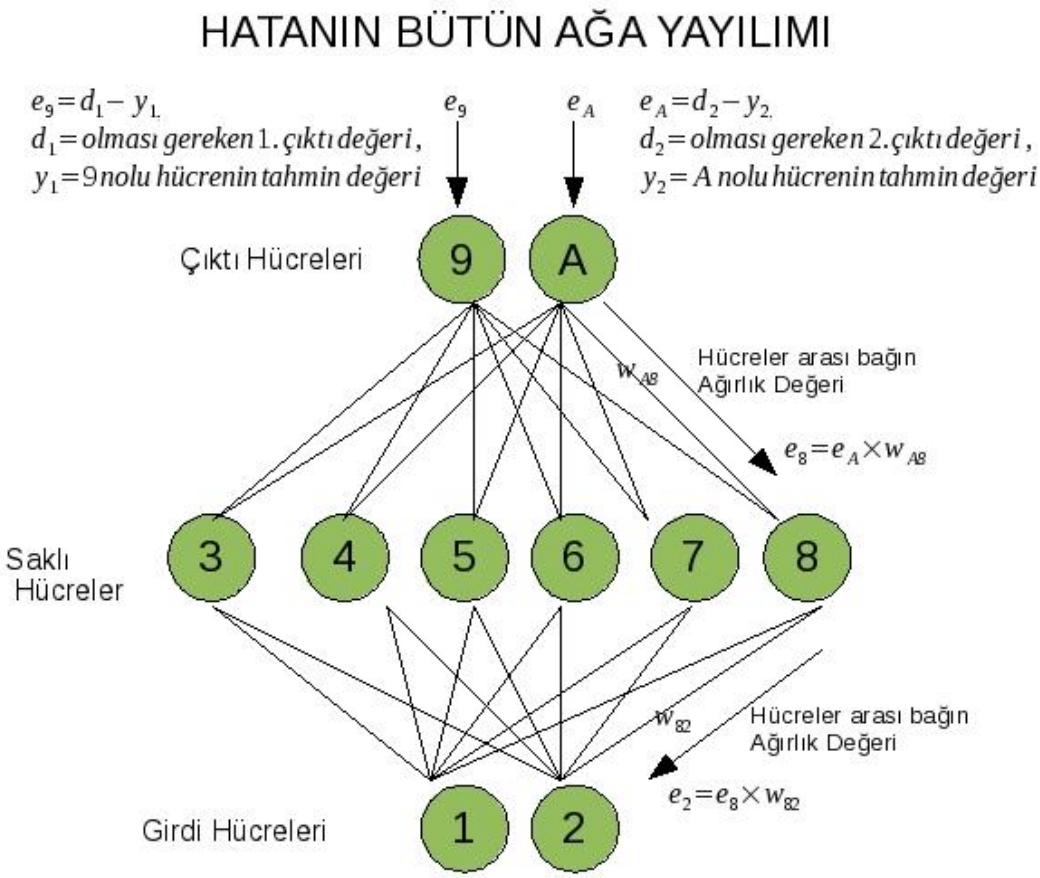
Bu öğrenme biçiminde YSA'ya etiketlenmiş girdi ve çıktılar verilir. Yani her girdiye karşılık, olması gereken çıktıyı YSA bilir. YSA, girdileri ilk önce kendi kendine işler daha sonra elde edilen çıktı ile olması gereken çıktıyı karşılaştırır. Aradaki farkı bularak ağırlık değerlerini (weight) düzeltir. Bu öğrenme yöntemine sıklıkla "Back Propagation" denmektedir.[22] Şekil 3-10'da bir tek hücre için hata düzeltmenin nasıl olacağı gösterilmektedir, Çizelge 3-1'de de bu şekilde kullanılan sembollerin anlamları verilmektedir. Şekil 3-11'de ise tek hücre için hesaplanan hata payının bütün ağa nasıl yayıldığı gösterilmektedir.



Şekil 3-10 Hata Düzelterek Öğrenme (Back Propagation)[22]

Sembol	Anlamı
e	Hata değeri
d	Olması gereken çıktı değeri
y	Hücre içinde hesaplanan çıktı değeri
Δw_i	i indisli ağırlık değeri (weight) için uygulanacak fark miktarı.
η	Öğrenme katsayısı

Çizelge 3-1 Hata Düzelterek Öğrenme Değişkenleri [22]



Şekil 3-11 Hata Değerinin Bütün Ağa Geri Yayılımı [45]

Yöntem şu şekilde işlemektedir:

1. Verilen girdilerle yapay sinir hücresi içinde hesaplanarak elde edilen çıktı ve olması gereken çıktı arasındaki hata hesaplanır.

$$e = d - y \qquad 3\text{-}2$$

2. Bu hatayı düzeltmek için ağırlık (weight) değerlerinde yapılması gereken düzeltme hesaplanır.

$$\Delta w_i = \eta \cdot e \cdot x_i \qquad 3\text{-}3$$

3. Ağırlık değerleri bu düzeltmeyi içerecek şekilde güncellenir.

$$w_i = w_i + \Delta w_i \qquad 3\text{-}4$$

η değeri optimum ağırlık değerine ulaşmak için atılacak adımlarım büyüklüğü olarak düşünülebilir. η değeri eğer çok küçük bir değer seçilirse öğrenme çok yavaş olacaktır. η değeri eğer çok büyük seçilirse Δw çok büyük olacağından hiçbir zaman optimum noktayı bulamayacaktır.

Bir YSA'yı eğitirken öğrenme algoritması şu şekilde uygulanır [16]:

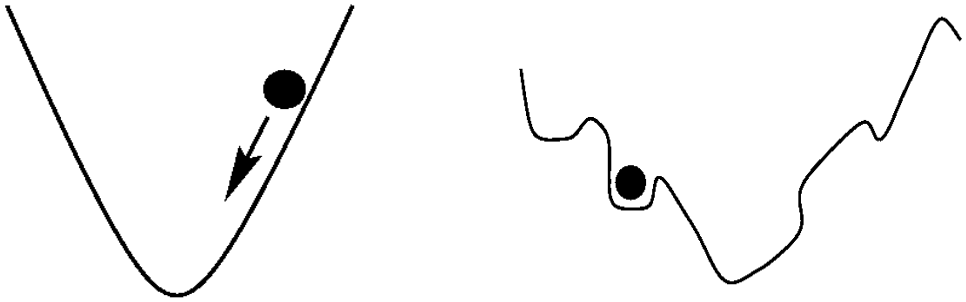
1. Girdiler verilir.
2. Çıktılar elde edilir.
3. Eğer çıktılar ile olması gereken değerler arasındaki fark bir önceki hata değerinden küçükse öğrenme algoritmasına devam edilir.
4. Eğer çıktılar ile olması gereken değerler arasındaki fark bir önceki hata değerinden büyükse öğrenme algoritması durdurulur çünkü hata fonksiyonunda minimuma ulaşıldığı var sayılır.

YSA'larda, çıkış değeri ve olması gereken gerçek değer arasındaki farka "**Hata**" (Error) denir [22]. Eğer YSA'nın çıktısı, olması gereken çıktılara

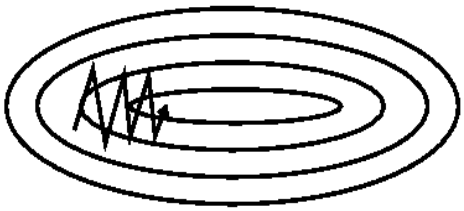
yakınsıyorsa bu durumda hata değerinin bir minimum değeri olması gerekmektedir. Bu durumda hata değerlerinin oluşturduğu eğri aşağıda gösterilen şekilde olması gerekmektedir. Şekil 3-12'deki gibi bir veya birden fazla minimumu olan (genel/yerel minimum) fonksiyonlar YSA için "**Hata Fonksiyonudur"** [22].

Denklem 3-4'te belirtilen ağırlık güncellemesinin sonuca daha çabuk yakınsayabilmesi için, 2. adımdaki güncelleme değerine bir de aşağıdaki gibi momentum ($\alpha * \Delta w_{i-1}$) değeri eklenir. Burada **α** momentum katsayısıdır.[22]

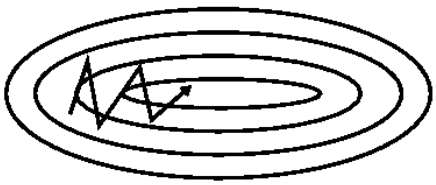
$$\Delta w_i = \eta \cdot e \cdot x_i + \alpha \cdot \Delta w_{i-1} \qquad 3\text{-}5$$



Şekil 3-12 Örnek Hata Fonksiyonları [22]



Şekil 3-13 Momentumsuz Hata Düzeltme[10]



Şekil 3-14 Momentumlu Hata Düzeltme [10]

Momentum katsayısı hata fonksiyonunu yerel minimum değerine takılma ihtimalini azaltır. Ayrıca momentum, gerçek uygulamalarda denklem 3-5'te denklemin solundaki ilk terim ikinci terime göre daha küçük olduğu için, hata düzeltmesinde oluşan osilasyonları da sönümleme özelliğine sahiptir.

3.4.2 Hafıza Tabanlı Öğrenme (Memory Based Learning)

Hafıza tabanlı öğrenme, hata düzeltme öğrenmesindeki gibi bir hücre için öğrenme tanımı yapar, bunu bütün ağa yayılımı, hata düzelterek öğrenmedeki gibidir. Yarıçap Tabanlı Fonksiyon Ağları (Radial-Basis Function Networks) bu mantıkla çalışmaktadır. [22]

Hafıza tabanlı öğrenme aşağıdaki şekilde işletilmektedir [22] :

1. Bütün giriş vektörlerine (bir seferde verilen girdilerin hepsi) karşılık gelen bütün düzgün çıktılar bir hafızaya atılır. Bu hafıza (vektör, sayı) ikililerinden oluşan bir çizelgedir.
2. Yeni gönderilen test verileriyle YSA aşağıdaki gibi eğitilir:
    1. Yeni giriş değerlerini hafızadaki kaç vektörle karşılaştırılacağı belirlenir. ("K-nearest neighbour" yöntemindeki k değerinin belirlenmesi)
    2. Yeni gelen girdi vektörünün daha öncekilerle olan uzaklığı hesaplanır. Bunun için bir uzaklık tanımı seçilir (Öklid, Manhattan, Vektör Çarpımı, Korelasyon Katsayısı...). Uzaklıklar içinde en iyi sonucu veren genelde Korelasyon Katsayısıdır.
    3. Bu uzaklıkları minimize edecek şekilde bir ağırlık fonksiyonu tanımlanır.

3.4.3 Hebb Öğrenmesi

Bu öğrenme yöntemi, Kanadalı bir nöropsikolog olan Donald Hebb tarafından bulunmuş biyolojik bir sinir hücresi olayına dayanır. Hebb'in keşfine göre eğer bir sinir hücresinden diğerine sinyal gittiğinde ikisi birden aktif oluyorsa (ikisi birden sinyal yayınlıyorsa) bu iki sinir hücresi arasındaki bağ güçlenir ve bir dahaki sefere bu hücreler arası sinyaller daha kolay iletilir. Eğer her iki hücre de aktif halde

bulunmuyorsa, aradaki bağ zayıflar ve sinyal bir dahaki seferi daha zor geçer. (Hebb, 1949)

$\mathbf{\Delta w_{kj}}$ j hücresinin çıktısından k hücresine girdi olarak gelen sinyalin ağırlığı, $\boldsymbol{\eta}$ öğrenme katsayısı, $\mathbf{y_k}$ k hücresinden çıkan çıktının değeri, $\mathbf{x_j}$ j hücresinden k hücresine gelen sinyalin değeri olarak tanımlarsak, bu ilke yapay sinir hücresinin eğitiminde şu şekilde tanımlanabilir [22] (sadece k numaralı hücre için):

1. Yeni ağırlık değerlerini belirlemek için uygulanacak olan fark:

$$\Delta w_{kj} = \eta \cdot y_k \cdot x_j \qquad 3\text{-}6$$

2. Yeni ağırlık değeri:

$$w_{kj} = w_{kj} + \Delta w_{kj} \qquad 3\text{-}7$$

Denklem 3-7'nin aşağıdaki gibi yazılması da mümkündür, aşağıda parantez içindeki n, kesikli zaman aralıklarını verir, n+1 de bir şeyin yeni değerini belirtmek için konur.

$$w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj}(n) \qquad 3\text{-}8$$

3.4.4 Rekabete Dayalı Öğrenme (Competitive Learning)

Bu öğrenme yönteminde bütün hücrelerin birbirlerine bağlı olduğu, aynı anda sadece bir hücrenin aktif olabildiği ve bir hücrenin bütün girdilerinin ağırlık değerlerinin toplamının 1 olduğu ön koşulları kabul edilir. Örneğin bir k numaralı hücrenin aktif olma koşulu denklem 3-9 ile ifade edilebilir. Öğrenme Kuralı (Uygulanacak ağırlık değişimi) ise denklem 3-10 ile ifade edilir. Burada yine η öğrenme katsayısını göstermektedir. Eğer hücre bir kere kazanırsa, bir dahaki sefere kazanması daha kolay olacaktır.

$$y_k = \begin{cases} 1 \text{ eğer } 1 \le j \le N, j \ne k \text{ için } \nu_k > \nu_j \text{ ise} \\ 0 \text{ değilse} \end{cases} \quad 3\text{-}9$$

$$\Delta w_{kj} = \begin{cases} \eta \cdot (x_j - w_{kj}) & \text{Eğer k numaralı hücre kazanırsa (aktifse)} \\ 0 & \text{Eğer k numaralı hücre kaybederse (inaktifse)} \end{cases} \quad 3\text{-}9$$

3.4.5 En Büyük Olabilirlik Tahmini (Maximum Likelihood Estimation)

"En Büyük Olabilirlik Tahmini" eldeki verilerden yola çıkarak, bu verilere uygun olasılık dağılım fonksiyonunu tahmin etmeye çalışan bir yöntemdir. Bu yöntem hem Boltzmann Öğrenmesi yönteminde, hem de Zıt Iraksama (Contrastive Divergence) yönteminde kullanıldığı için anlatılmaktadır.

En büyük olabilirlik tahmini yöntemi esas itibariyle "Parametrik Model" kavramının üstüne oturmaktadır. Parametrik model sonlu sayıda parametre alabilen olasılık dağılımlarına verilen genel isimdir (Akdeniz,1998). $P(\theta_1,...\theta_N)$

Bir parametrik model $P(\theta)$ olsun. Eğer θ'nın iki ayrı θ_1 ve θ_2 gibi değeri için hiçbir zaman $P(\theta_1)=P(\theta_2)$ olmuyorsa, $P(\theta)$ tanımlanabilir (identifiable) denir. Bir istatistiksel çıkarım yapabilmek için seçilen modellerin tanımlanabilir (identifiable) olması gerekmektedir. Denklem 3-11 'de parametrik modele bir örnek olarak Poisson dağılımı verilmiştir, denklem 3-12'de ise normal (Gauss) dağılımı verilmiştir.

$$\mathcal{P} = \left\{ p_\lambda(j) = \frac{\lambda^j}{j!} e^{-\lambda},\ j = 0, 1, 2, 3, \dots \;\middle|\; \lambda > 0 \right\}, \quad 3\text{-}10$$

$$\mathcal{P} = \left\{ f_\theta(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(x-\mu)^2} \;\middle|\; \mu \in \mathbb{R}, \sigma > 0 \right\}. \quad 3\text{-}11$$

En Büyük Olabilirlik tahmini yönteminin adımları aşağıdaki gibidir:

1. Örneklenen verinin "Olabilirlik Fonksiyonu" (L) yazılır. Problemin doğasına göre bir parametrik olasılık dağılım fonksiyonu seçilir. (parametrik olasılık fonksiyonuna genel olarak “parametrik model” denir). Örneğin eğer sadece iki ihtimal (yazı-tura gibi) varsa binom dağılımı seçilir. Eldeki verilere göre “Olabilirlik Fonksiyonu” L’nin son hali Birleşik Yoğunluk Fonksiyonu (Joint Density Function) yazılarak elde edilir.
2. Yazılan "Olabilirlik Fonksiyonu" L’nin maksimumu değeri, bu fonksiyonun parametrelerine göre türevi alınıp sıfıra eşitlenerek bulunur.

En büyük olabilirlik tahmini için ilk önce n tane gözlem yapılır ve bu gözlemler $x_1,x_2,...,x_n$ olarak adlandırılır. Daha sonra yapılan deneyin doğasına göre bu verilerin uyabileceği θ parametresine bağlı bir olasılık dağılım fonksiyonu f(.|θ) seçilir. Bu dağılım(yoğunluk) fonksiyonu ve eldeki verilere göre denklem 3-13’teki gibi bileşik yoğunluk fonksiyonu yazılır. Olayların birbirinden bağımsız olduğu var sayıldığı için bileşik yoğunluk fonksiyonu direk olarak bu olaylar oduğundaki x değerleri yerlerine konarak elde edilen olasılık fonksiyonlarının çarpımı şeklinde yazılabilmektedir.

$$f(x_1, x_2, \ldots, x_n \mid \theta) = f(x_1|\theta) \cdot f(x_2|\theta) \cdots f(x_n|\theta). \qquad \textit{3-12}$$

$$\mathcal{L}(\theta \mid x_1, \ldots, x_n) = f(x_1, x_2, \ldots, x_n \mid \theta) = \prod_{i=1}^{n} f(x_i|\theta). \qquad \textit{3-13}$$

Veri kümesindeki $x_1,x_2,..,x_n$ verilerinin sabit olduğu ve θ parametresinin değişken olduğu kabul edilir ve denklem 3-14 yazılır. Sonuçta elde edilen L fonksiyonunun θ parametresine göre türevi alınarak 0’a eşitlenir, Bu eşitliği sağlayan θ parametresi, bu fonksiyonun maksimumunu verir. Bulunan θ parametresini dağılım fonksiyonunda yerine koyarsak, eldeki verilere en çok uyan dağılım fonksiyonunu elde etmiş oluruz. Uygulamada ise daha kolay işlem yapılabilmesi için genelde L fonksiyonunun logaritması alınır.

Örnek olarak $x_1,x_2,\ldots,x_n$ gibi rastgele seçilmiş üstel dağılıma uyduğunu kabul ettiğimiz veriler olsun. Denklem 3-15'te üstel dağılımın denklemi görülmektedir. Eldeki verilere $(x_1,x_2,..,x_n)$ bağlı olarak olabilirlik fonksiyonu da denklem 3-16'daki gibi olmaktadır. Daha kolay işlem yapabilmek için denklem 3-16'nın logaritması alınır ve denklem 3-17'deki θ parametresine göre türevi alınır ve 0'a eşitlenir, denklem 3-18 elde edilir. Sonuç olarak, θ parametresi için en büyük olabilirlik tahmini denklem 3-19'da verilen ifade olacaktır

$$f(x;\theta) = \frac{1}{\theta}e^{-x/\theta},\ 0 < x < \infty,\ \theta \in \Omega = \{\theta;\ 0 < \theta < \infty\}. \qquad \textit{3-14}$$

$$L(\theta) = L(\theta|x_1, x_2, \cdots, x_n) = \left(\frac{1}{\theta}e^{-x_1/\theta}\right)\cdots\left(\frac{1}{\theta}e^{-x_n/\theta}\right) = \frac{1}{\theta^n}e^{\frac{-\sum_{i=1}^{n} x_i}{\theta}} \qquad \textit{3-15}$$

$$\ln L(\theta) = -(n)\ln(\theta) - \frac{1}{\theta}\sum_{i=1}^{n} x_i,\ 0 < \theta < \infty. \qquad \textit{3-16}$$

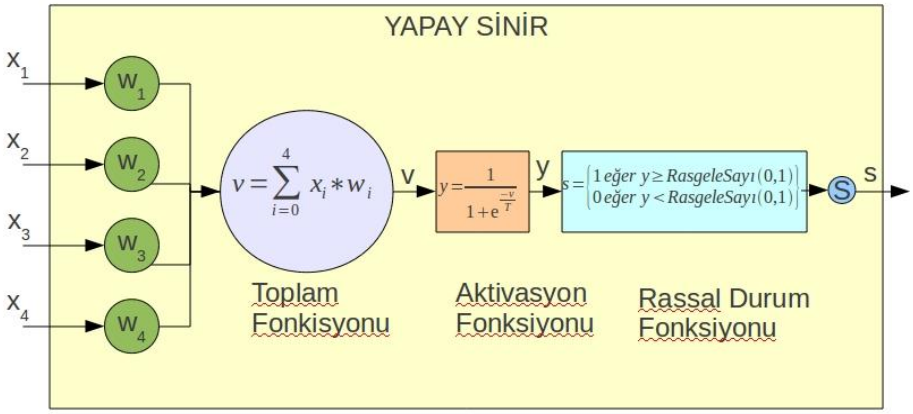
$$\frac{d[\ln L(\theta)]}{d\theta} = \frac{-n}{\theta} + \frac{\sum_{i=1}^{n} x_i}{\theta^2} = 0. \qquad \textit{3-17}$$

$$\theta = \frac{1}{n}\sum_{i=1}^{n} x_i = \bar{x}. \qquad \textit{3-18}$$

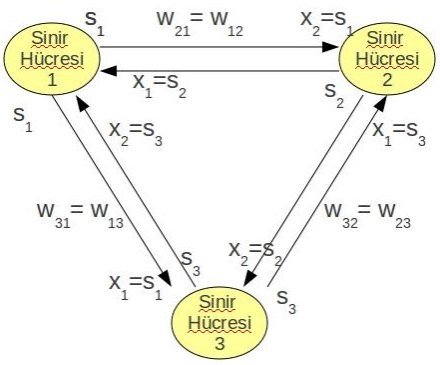
3.4.6 Boltzmann Öğrenmesi (Boltzmann Learning, Simulated Annealing)

Boltzmann Öğrenmesi bir stokastik öğrenme algoritmasıdır. İstatistik mekanikteki Boltzmann kuramından etkilenmiştir. Bu öğrenme biçimi üzerine kurulu olan YSA'ya Boltzmann Makinesi denmektedir. Aktivasyon fonksiyonundaki T harfi, sıcaklığı belirtmektedir. Bu hücre için x girdi değerleri, aynı zamanda diğer hücrelerden gelen s durum değerleridir. Bunun için değişik kitaplardaki formüllerde bazen x bazen de s olarak geçer. Bu yönteme aynı zamanda "Simulated Annealing"

denir. Bazı kitaplar “Boltzmann Öğrenmesi” , bazı kitaplar “Simulated Annealing” başlığı altında bu konuyu vermiştir. T sanal sıcaklığına aynı zamanda “Isıl Gürültü” (Thermal Noise) de denmektedir. Bu teknik “Kombinatoryal Optimizasyon” problemlerimde sıkça kullanılmaktadır.



Şekil 3-15 Boltzmann Makinesi Yapay Sinir Hücresi[13]



Şekil 3-16 Boltzmann Makinesi [13]

Boltzmann Öğrenmesinde hücrelerin giriş/çıkış değerlerinin dışında bir de "**s**" sembolü ile temsil edilen hücrelerin durum değerleri vardır. Her hücre ya "Açık" ya da "Kapalı" durumda olabilir. Hücreler ya ikili (binary) (0/1) , ya da bipolar (-1/+1) mantığı ile aktif olurlar. Ağdaki bütün hücreler birbirlerine bağlıdır. Bütün hücreler arası ağırlıklar bakışımlıdır (simetriktir) (denklem 3-20). Hücreler kendilerine bağ yapamazlar (denklem 3-21). Ağ içerisinde girdi almak ve çıktı vermek için "görünür" (visible) hücreler vardır, dışarıya hiç görünmeyen sadece hesapta kullanılan "görünmez" (hidden) hücreler de vardır. Ağ içerisinde görünmez (hidden) hücreler bulunmak zorunda değildir. S vektörü, bütün hücrelerin durumlarının bulunduğu vektör, W ağırlık değerlerinin bulunduğu matrisi belirtmek üzere enerji fonksiyonu denklem 3-22'deki gibi yazılmaktadır. Aktivasyon fonksiyonu ise **Boltzmann Dağılım** fonksiyonudur ve denklem 3-23'teki gibi ifade edilmektedir.[13]

$$w_{kj} = w_{jk} \qquad 3\text{-}19$$

$$w_{ii} = 0, \forall\ i \qquad 3\text{-}20$$

$$E = -\frac{1}{2} \cdot \sum_{j}^{N} \sum_{k,k \neq j}^{N} w_{kj} \cdot s_k \cdot s_j = -\frac{1}{2} \cdot S \cdot W \cdot S^T \qquad 3\text{-}21$$

$$y = \frac{1}{1 + e^{-\frac{v}{T}}} \qquad 3\text{-}22$$

Denklem 3-23'teki T sembolü "sanal sıcaklık" değerini belirtmektedir. Sistemin T "sanal sıcaklık" değerinin daha önceden belli bir başlangıç ve bitiş değeri vardır. T sanal sıcaklığının bitiş değeri sistemin enerjisinin minimum olduğu yani dengede olduğu zamandaki sıcaklığıdır. Öğrenme süresi minimum sıcaklığa kadar devam eder.

Öğrenme adımları aşağıdaki gibi gerçekleşir[5]:

1. Bir grup örneklem alınır ve bir başlangıç sıcaklığı belirlenir (T)
2. Her örnek sırayla girdi olarak Boltzmann Makinesine verilir
3. Her örnek için aşağıdaki adımlar uygulanır
    1. Sistemin enerjisini daha da düşürecek şekilde işlemler yapılır.

       Örneğin $P_k(s_k \to -s_k) = \frac{1}{1+e^{-\frac{\Delta E_k}{T}}}$, P_k k hücresinin durumunun değişme olasılığını vermektedir. ΔE_k k hücresinin durumunun s_k'dan $-s_k$'ya geçerken sistemdeki enerji değişimini vermektedir.
    2. Bir sonraki adımda T sıcaklığı, minimum değerini alıncaya kadar, azar azar düşürülür.

Bu kural döngüsel olarak T minimum değerini alıncaya kadar uygulanırsa, T minimum değerini aldığında YSA da bir ısıl dengeye (Thermal Equilibrium) ulaşacaktır. Isıl dengeye ulaşması demek, YSA'nın yaklaşım yapılacak fonksiyona en yakın haline gelmesi demektir.[2]

Boltzmann Öğrenmesinin matematiksel altyapısı "En Büyük Olabilirlik Tahmini" yöntemine ve fizikteki "Boltzmann Kuramına" dayanmaktadır. "F" YSA'yı eğitmek için kullanılacak veri kümesi olsun. "X" YSA'daki hücrelerinin durum vektörü (state vector) olsun. Boltzmann öğrenmesini kullanan YSA'nın çıktı değerleri (ve durumları) 0 veya 1'dir. Dolayısıyla bu vektör 0 veya 1'lerden oluşmaktadır. "X_α" YSA'daki görünür hücrelerin durum vektörü (state vector of visible neurons), "X_β" YSA'daki saklı hücrelerin durum vektörü (state vector of hidden neurons) olsun. x, x_α , x_β değerleri X, X_α , X_β vektörleri içinden seçilmiş değerler olsun. x_α F eğitim veri kümesinin bir elemanı olmak üzere $P(X_\alpha=x_\alpha)$ görünür hücrenin durumunun x_α olma ihtimalini temsil etsin. Bu koşullar altında öğrenme

adımları aşağıdaki gibidir[7] :

1. En büyük olabilirlik tahmini yöntemindeki gibi, F eğitim veri kümesindeki her eleman için bileşik olasılık, yani Olabilirlik Fonksiyonu (Likelihood Function) denklem 3-24'teki gibi ifade edilir:

$$\prod_{x_\alpha \in F} P(X_\alpha = x_\alpha) \qquad 3\text{-}23$$

2. Bu fonksiyonun logaritması alınır. Denklem 3-25'teki W, bizim YSA'mızdaki ağırlık değerlerinin matrisidir:

$$L(w) = \log(\prod_{x_\alpha \in F} P(X_\alpha = x_\alpha)) = \sum_{x_\alpha \in F} \log(P(X_\alpha = x_\alpha)) \qquad 3\text{-}24$$

3. **Fizikteki Boltzmann Kuralına göre** bir sistemin x durumunda olma ihtimali denklem 3-26'daki gibi ifade edilir. Bu denklemdeki Z olabilecek tüm x durumları için toplam olasılığı belirtmektedir, sabittir.

$$P(X = x) = \frac{1}{Z} \cdot e^{-\frac{E(x)}{T}} \qquad 3\text{-}25$$

$$Z = \sum_x e^{-\frac{E(x)}{T}} \qquad 3\text{-}26$$

4. Bütün sistem görünür ve saklı hücrelerden oluştuğuna göre denklem 3-26 denklem 3-28'deki gibi açılabilir.

$$P(X = x) = P(X_\alpha = x_\alpha) - \sum P(X_\beta = x_\beta) = P(X_\alpha = x_\alpha) - \frac{1}{Z} \sum_{x_\beta} e^{-\frac{E(x)}{T}} \qquad 3\text{-}27$$

5. En büyük olabilirlik denkleminin logaritmasını alırız ve denklem 3-29 ve sonra denklem 3-30'u elde ederiz. Denklemlerde kullanılan E enerji fonksiyonudur ve denklem 3-31'deki gibi ifade edilmektedir.

$$L(w) = \sum_{x_\alpha \in F} \left(\log\left(\frac{1}{Z} \sum_{x_\beta} e^{-\frac{E(x)}{T}} \right) - \log\big(P(X = x)\big) \right) \qquad 3\text{-}28$$

$$L(w) = \sum_{x_\alpha \in F} \left(\log\left(\frac{1}{Z} \sum_{x_\beta} e^{-\frac{E(x)}{T}} \right) - \log\left(\left(\frac{1}{Z} \sum_{x} e^{-\frac{E(x)}{T}} \right) \right) \right) \qquad 3\text{-}29$$

$$E(x) = -\frac{1}{2} \cdot \sum_{i} \sum_{j, j \neq i} w_{ij} \cdot x_i \cdot x_j \qquad 3\text{-}30$$

6. En büyük olabilirlik tahmini yönteminde olduğu gibi, parametrelere yani W matrisinin elemanlarına göre türev alınır

$$\frac{\partial L(w)}{\partial w_{ji}} = \frac{1}{T} \sum_{x_\alpha \in F} \left(\sum_{x_\beta} P(X_\beta = x_\beta \mid X_\alpha = x_\alpha) \cdot x_j \cdot x_i - \sum_{x} P(X = x) \cdot x_j \cdot x_i \right) \qquad 3\text{-}31$$

$$\frac{\partial L(w)}{\partial w_{ji}} = \frac{1}{T} \left(\sum_{x_\alpha \in F} \sum_{x_\beta} P(X_\beta = x_\beta \mid X_\alpha = x_\alpha) \cdot x_j \cdot x_i - \sum_{x_\alpha \in F} \sum_{x} P(X = x) \cdot x_j \cdot x_i \right) \qquad 3\text{-}32$$

$$\frac{\partial L(w)}{\partial w_{ji}} = \frac{1}{T} \left(\rho_{ji}^{+} - \rho_{ji}^{-} \right) \qquad 3\text{-}33$$

7. Burada ρ_{ji}^{+} YSA dışarıdan veri alıp eğitilirken (yani pozitif fazda) i. ve j. hücre arasındaki ilişkiyi belirtmektedir ve aşağıdaki şekilde ifade edilmektedir:

$$\rho_{ji}^{+} = \left\langle x_j x_i \right\rangle^{+} = \sum_{x_\alpha \in F} \sum_{x_\beta} P(X_\beta = x_\beta \mid X_\alpha = x_\alpha) \cdot x_j \cdot x_i \qquad 3\text{-}34$$

8. Burada ρ_{ji}^{-} YSA dışarıdan veri almazken (yani negatif fazda) i. ve j. hücre arasındaki ilişkiyi belirtmektedir ve aşağıdaki şekilde ifade edilmektedir:

$$\rho_{ji}^{-} = \left\langle x_j x_i \right\rangle^{-} = \sum_{x_\alpha \in F} \sum_{x} P(X = x) \cdot x_j \cdot x_i \qquad 3\text{-}35$$

En büyük olabilirlik yöntemine göre, amacımız L(w) fonksiyonunun en büyük (maksimum) değerini bulmak. Bu amaca sayısal yöntemlerle ulaşmak için "Gradient Ascent" yöntemini kullanabiliriz:

$$\Delta w_{ji} = \varepsilon \cdot \frac{\partial L(w)}{\partial w_{ji}} = \eta \cdot \left(\rho_{ji}^{+} - \rho_{ji}^{-}\right), \;\; \eta = \frac{\varepsilon}{T} \qquad 3\text{-}36$$

3.4.7 Zıt Iraksama Yöntemiyle Öğrenme (Contrastive Divergence)

N adet sayı dizisini gözlemlediğimizi düşünelim. Bu sayı dizilerinin kümesini $X=\{x_1,x_2,x_3\ldots,x_N\}$ olarak ifade edelim. Bu gözlemlerin her birini YSA'ya girdi olarak verip öğrenmesini sağlamalıyız. YSA'nın bu gözlemleri öğrenmesi demek kendi içindeki W ağırlık matrisin değerlerini bu gözlemlere en çok uyacak şekilde belirlemesi demektir. YSA'ya bir matematiksel fonksiyon gözüyle bakarsak ki aslında öyledir, W matrisi bu fonksiyonun parametrelerini oluşturmaktadır. Bu parametreleri bulmak için "En Büyük Olabilirlik" (Maximum Likelihood) yöntemini kullanacağız. Bu sayı dizisini aşağıdaki gibi bir olasılık fonksiyon ile modellemeye çalışalım:

$$p(x;W) = p(x \mid W) = \frac{e^{-E(x;W)}}{Z(W)} \qquad 3\text{-}37$$

öyle ki

- p(x;W) kullanılan YSA'nın matematiksel fonksiyon olarak gösterimi
- W bu olasılık fonksiyonunun parametreleri
- Z(W) normalizasyon sabiti (partition function)

$$Z(W) = \sum_{x_n} e^{-E(x_n;W)} \qquad 3\text{-}38$$

- E(x;W) enerji fonksiyonu

$$E(x;W) = -\frac{1}{2} \cdot \sum_{i} \sum_{j,j \neq i} w_{ij} \cdot x_i \cdot x_j \qquad 3\text{-}39$$

olsun.

En Büyük Olabilirlik (Maximum-Likelihood) yöntemiyle bu fonksiyonun

parametrelerinin (W) bir veri kümesine ($X = \{x_n\}_{n=1}^{N}$) en çok uyan değerlerini bulmak için "Gradient Descent" yöntemi kullanılabilir. Her öğrenme adımında denklem 3-41'deki ağırlık güncelleme değeriyle YSA'daki ağırlıklar güncellenir (W=W+ΔW).

$$\Delta W = \eta \cdot \frac{\partial L(W;X)}{\partial W} \qquad 3\text{-}40$$

X veri setine uygun parametreleri (W) bulmak için yazdığımız olabilirlik (Likelihood) fonksiyonunun logaritması (Log-Likelihood) denklem 3-42'teki gibidir. Göz önünde bulundurmamız gereken bir nokta da Z(W)'nin x_n değerlerine göre sabit olduğudur.

$$L(W;X) = \frac{1}{N}\sum_{n=1}^{N} \log(\, p(x_n;W)) \qquad 3\text{-}41$$

$$L(W;X) = \frac{1}{N}\sum_{n=1}^{N} \log(\frac{e^{-E(x_n;W)}}{Z(W)}) \qquad 3\text{-}42$$

$$L(W;X) = \frac{1}{N}\sum_{n=1}^{N} \left(\log(e^{-E(x_n;W)}) - \log(\, Z(W))\right) \qquad 3\text{-}43$$

$$L(W;X) = \frac{1}{N}\sum_{n=1}^{N} \left(-E(x_n;W) - \log(\, Z(W))\right) \qquad 3\text{-}44$$

$$L(W;X) = -\frac{1}{N}\sum_{n=1}^{N} E(x_n;W) - \log(\, Z(W)) \qquad 3\text{-}45$$

Buradan denklem 3-46'daki fonksiyonun parametrelerinin en büyük değerlerini bulabilmek için aşağıdaki gibi türevleri alınır:

$$\frac{\partial L(W;X)}{\partial W} = -\frac{1}{N}\sum_{n=1}^{N}\frac{\partial E(x_n;W)}{\partial W} - \frac{\partial \log(Z(W))}{\partial W} \qquad 3\text{-}46$$

$$\frac{\partial L(W;X)}{\partial W} = -\frac{1}{N}\sum_{n=1}^{N}\frac{\partial E(x_n;W)}{\partial W} - \frac{1}{Z(W)}\frac{\partial Z(W)}{\partial W} \qquad 3\text{-}47$$

$$\frac{\partial L(W;X)}{\partial W} = -\frac{1}{N}\sum_{n=1}^{N}\frac{\partial E(x_n;W)}{\partial W} - \frac{1}{Z(W)}\frac{\partial}{\partial W}\sum_{x} e^{-E(x;W)} \qquad 3\text{-}48$$

Bu noktada $e^{-E(x;W)}$ ifadesi f(x;W) olarak adlandırılırsa aşağıdaki denklemler elde edilir:

$$\frac{\partial L(W;X)}{\partial W} = -\frac{1}{N}\sum_{n=1}^{N}\frac{\partial E(x_n;W)}{\partial W} - \frac{1}{Z(W)}\frac{\partial}{\partial W}\sum_{x} f(x;W) \qquad 3\text{-}49$$

$$\frac{\partial L(W;X)}{\partial W} = -\frac{1}{N}\sum_{n=1}^{N}\frac{\partial E(x_n;W)}{\partial W} - \frac{1}{Z(W)}\sum_{x}\left(\frac{\partial}{\partial W} f(x;W)\right) \qquad 3\text{-}50$$

$$\frac{\partial L(W;X)}{\partial W} = -\frac{1}{N}\sum_{n=1}^{N}\frac{\partial E(x_n;W)}{\partial W} - \frac{1}{Z(W)}\sum_{x}\left(f(x;W)\frac{\partial}{\partial W}\log(f(x;W))\right) \qquad 3\text{-}51$$

Z(W), x değerlerine için sabit olduğuna göre:

$$\frac{\partial L(W;X)}{\partial W} = -\frac{1}{N}\sum_{n=1}^{N}\frac{\partial E(x_n;W)}{\partial W} - \sum_{x}\left(\frac{f(x;W)}{Z(W)}\frac{\partial}{\partial W}\log(f(x;W))\right) \qquad 3\text{-}52$$

$$\frac{\partial L(W;X)}{\partial W} = -\frac{1}{N}\sum_{n=1}^{N}\frac{\partial E(x_n;W)}{\partial W} - \sum_{x}\left(p(x;W)\frac{\partial}{\partial W}\log(e^{-E(x;W)})\right) \qquad 3\text{-}53$$

$$\frac{\partial L(W;X)}{\partial W} = -\frac{1}{N}\sum_{n=1}^{N}\frac{\partial E(x_n;W)}{\partial W} - \sum_{x}\left(p(x;W)\frac{\partial(-E(x;W))}{\partial W}\right) \qquad 3\text{-}54$$

$$\frac{\partial L(W;X)}{\partial W} = -\frac{1}{N}\sum_{n=1}^{N}\frac{\partial E(x_n;W)}{\partial W} + \sum_{x}\left(p(x;W)\frac{\partial (E(x;W))}{\partial W} \right) \qquad 3\text{-}55$$

Sonuç olarak denklem 3-57 ile ağırlık değerleri güncellenmektedir.

$$\Delta W = \eta \cdot \left(\sum_{x}\left(p(x;W)\frac{\partial (E(x;W))}{\partial W} \right) - \frac{1}{N}\sum_{n=1}^{N}\frac{\partial E(x_n;W)}{\partial W} \right) \qquad 3\text{-}56$$

Yukarıda göründüğü gibi, güncelleme matrisini (ΔW) hesap etmek için, p(x;W) dağılımından birçok örnek veri kümesi üretmek gerekiyor. Ancak bu dağılımı net olarak bilemediğimiz için direk olarak örnek veri kümeleri üretememekteyiz. "Monte Carlo Markov Chain" (MCMC) yöntemini kullanarak birçok veri kümesi üretip istediğimiz veri kümelerine yakınsayabilmekteyiz. Eldeki her bir veri için bu işlemi yapmak çok büyük sayıda (sonsuza yakınsayabilmek için) işlem yapmayı gerektiriyor. 2002 yılında Geoffrey Hinton MCMC yöntemini sadece 1 kez işleterek yeterince yakınsanabildiğini deneysel olarak göstermiştir. Bu yönteme "Zıt Iraksama" (Contrastive Divergence) adı verilmiştir[20].

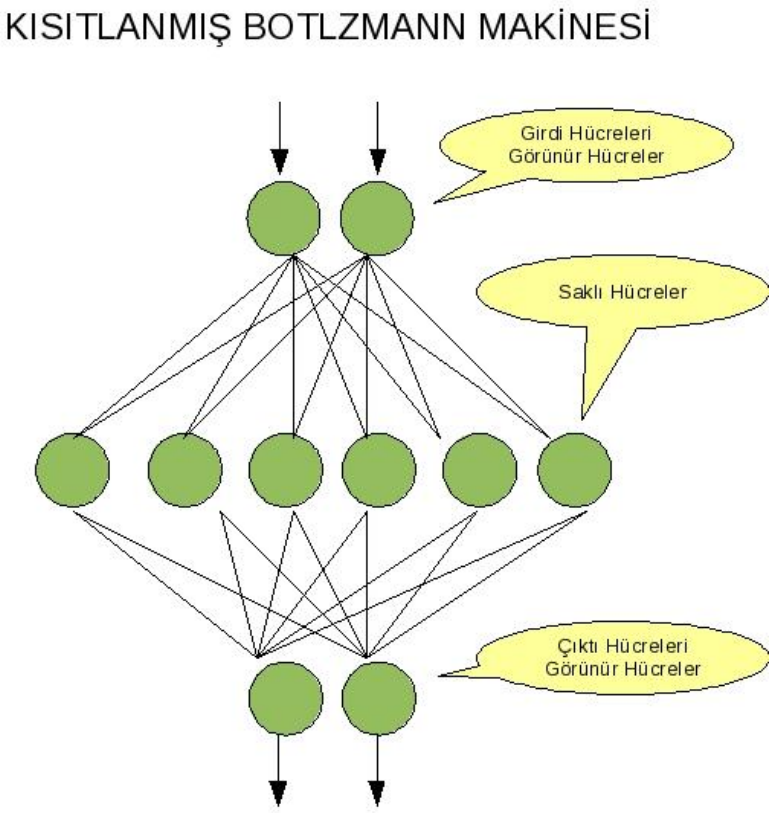
3.5 Boltzmann Makinesi

Boltzmann öğrenmesindeki ön koşulları sağlayan YSA'lara Boltzmann Makinesi denir. [22] Stokastik olması hücre durumunun değerine karar verirken kullanılan fonksiyondan ileri gelmektedir. Özyinelemeli olması da bütün hücrelerin birbirlerine bağlı olmasından ileri gelmektedir. Boltzmann Makinesi gibi stokastik olan diğer YSA'lar ise "Sigmoid İnanç Ağları" (Sigmoid Blief Networks) ve "Helmholtz Makinesi" dir (Hinton-Zemel, 1994).

Boltzmann Makinesi iki fazda çalışır, pozitif faz ve negatif faz. Pozitif fazda YSA kilitlidir (Clamped) ve örnek veri grubunun (eğitileceği verilerin) etkisi altındadır. Başka bir deyişle pozitif fazda YSA'nın girdi hücrelerine değer atanır. Negatif fazda ise dışarıdan herhangi bir değer ataması yapılmaz, YSA belli bir T sıcaklığının altına düşene kadar kilitsiz (serbest) bir biçimde çalıştırılır.

3.6 Kısıtlanmış Boltzmann Makinesi

Bu YSA, Boltzmann Makinesinden türemiştir. Kısıtlanmış Boltzmann Makinesinde, Boltzmann Makinesinde olduğu gibi bütün yapay sinir hücreleri birbirlerine bağlı değildir. Mimarisi şekil 3-17'deki gibidir:



Şekil 3-17 Kısıtlanmış Boltzmann Makinesi [24]

Kısıtlanmış Boltzmann Makinesinin özellikleri (Smolensky, 1986):

1. Öğrenme algoritması olarak Zıt Iraksama yöntemini kullanmaktadır.
2. Görünür hücre, dışarıdan girdi alabilen veya dışarıya çıktı verebilen hücredir.
3. Saklı hücre dışarı ile irtibatı olamayan hücredir.
4. Girdi ve çıktı hücrelerin sayısında bir sınır yoktur, girdi ve çıktı hücrelerinin sayısı birbirlerine eşit olmak zorunda değildir.
5. Görünür hücreler kendi aralarında bağ yapamamaktadır, ancak saklı hücreler ile çift yönlü bağ yapmaktadır.

4 YSA'larla Zaman Serileri Tahmini

Bu çalışma kapsamında YSA'lar ile zaman serileri tahmini yaparken hem ileri yönelimli YSA'lar hem de kısıtlanmış Boltzmann makineleri kullanılmaktadır. Her iki yöntemle de zaman serileri verileri "Zaman Aralığı" kavramı kullanılarak işlenmiştir. Hem YSA'ların eğitimi hem de tahmin yapılması sırasında zaman aralıkları birer kaydırılarak YSA'ya girdi verisi olarak verilmiş, çıktılar da birer kaydırılarak elde edilmiştir [15]. Gerçekleştirilen deneylerde aynı zamanda kaydırmasız, tek adımda, tahmin de yapılarak çıkan sonuçlar gerçeğe yakınsama açısından karşılaştırılmaktadır.

Kısıtlanmış Boltzmann makinesindeki girdi çıktı değerleri sadece ikili (binary) olacağı için, uygulama aşamasında gerçel (real) sayılar bir fonksiyon ile ikili değerlere çevrilmiştir. Testler sırasında gerçel sayıları ikili değerlere çevirmek için 32 basamak kullanılmıştır ancak uygulama içerisinde bu basamak değeri de parametrik olduğu için başka bir deneyde başka bir basamak değeri kullanılabilir.

Zaman aralıklarını birer kaydırarak girdi verme ve çıktı elde etme yöntemi

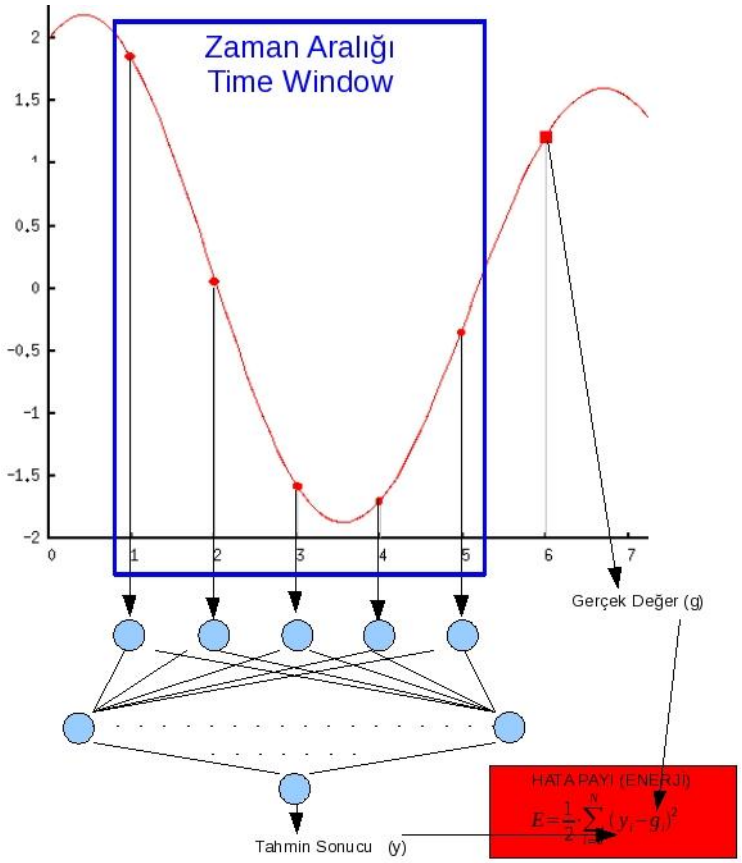
1999 yılında Fablec ve Alliot'nun "Using Neural Networks to Predict Aircraft Trajectories" adlı makalesinde anlatılmıştır [15]. Ancak bu makalede Boltzmann makinesi değil İleri Yönelimli YSA'lar kullanılmıştır. 2009 yılında ise Taylor ve Hinton "Factored Conditional Restricted Boltzmann Machines for Modeling Motion Style" adlı makalesinde Kısıtlanmış Boltzmann Makinesi kullanarak tahmin yapacak uygulamalar için yöntemler önermiştir. Bu çalışma kapsamında ise Kısıtlanmış Boltzmann Makinesi ile zaman serileri tahmini bu iki yöntem birleştirilerek uygulanmıştır.

YSA'lar girdi olarak daha önceki verileri alacak ve çıktı olarak sonraki verilerin tahmini değerlerini verecektir. YSA'larda elde edilen sonuç ile olması gereken sonuç arasındaki fark denklem 4-1'deki gibi Öklid uzaklığı ile hesaplanmaktadır. Buna aynı zamanda "Enerji Fonksiyonu" da denmektedir [22].

$$E = \frac{1}{2} \cdot \sum_{i=1}^{N} (y_i - g_i)^2 \qquad \textit{4-1}$$

Bütün uygulama boyunca nihai hedef olarak E enerji/hata fonksiyonunun en küçük değerine ulaşılmaya çalışılmaktadır. Bu fonksiyonda:

- N = YSA'nın çıktı hücre sayısı. Bu sayı aynı zamanda kaç birim zaman sonrası için tahmin yapıldığını da göstermektedir.
- y_i değerleri, çıktı değerleridir. Başka bir deyişle tahmin değerleridir.
- g_i değerleri, gerçekte olan değerlerdir.



Şekil 4-1 Zaman Serisi Tahmini Hata Payı Değeri[35]

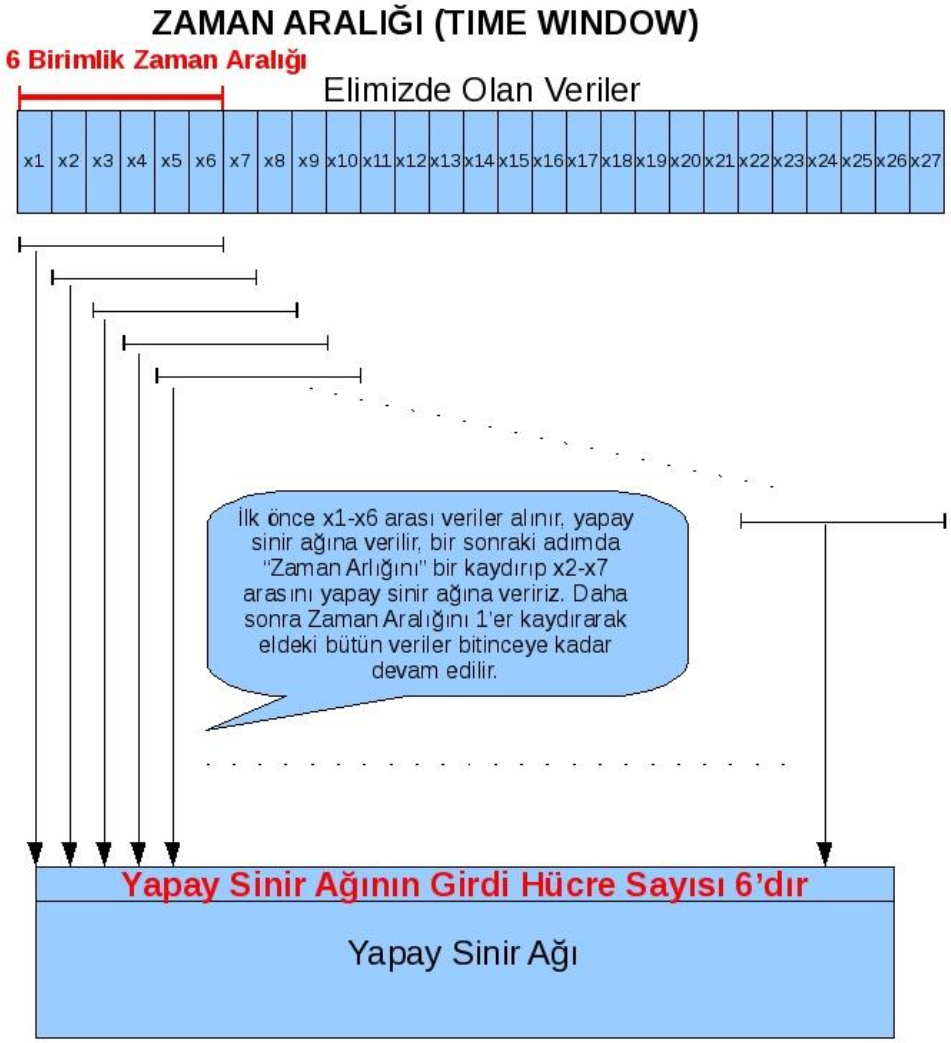
Şekil 4-1'de enerji fonksiyonunun hangi verileri kullandığı gösterilmektedir. Şekil 4-1'deki grafik tahmin edilmeye çalışılan verinin grafiğidir, enerji fonksiyonunun grafiği değildir.

4.1 Zaman Aralığı Tanımı (Time Window)

Zaman aralığı (Time Window) bir kerede YSA'nın alacağı zaman verisi büyüklüğüdür. YSA'lar eğitilirken ve tahmin yaparken, elde olan zaman verilerini girdi olarak alır ve aldığı girdilere göre bir çıktı üretir. Şekil 4-2'de gösterildiği gibi zaman serileri bir YSA tarafından işlenirken sırasıyla:

1. "Zaman Aralığı" kadar (şekilde 6 tane) veri ilk önce "Bilinen Zaman Verileri" dizisinden okunur.
2. Bu veriler YSA'ya girdi olarak verilir, YSA bu verileri işler ve bir çıktı üretir.

3. Zaman aralığı bir adım kaydırılır.
4. İlk 3 adım eldeki veriler bitene kadar ve tahmin sonucu oluşana kadar işletilir.



Şekil 4-2 Zaman Aralığı Tanımı

4.2 Gerçekleştirilen Uygulamanın Algoritması

YSA ile zaman serileri tahmini yapılırken 280 adet veri içeren "Yıllara Göre Güneş Noktaları Dağılımı" verisi kullanılmıştır. Şekil 4-3'te java dilinde ifadesi gösterilen uygulamanın ana algoritması aşağıdaki gibidir:

1. Ana veri kümesindeki veriler daha iyi tahmin sonucu elde edebilmek için

normalize edilir.

2. Beş adet farklı zaman aralığı için (2,4,6,8,10):
    1. Ana dizinden değişik büyüklüklerde 10 adet alt veri kümesi oluşturulur.
    2. Oluşturulan her veri kümesi için:
        1. Aynı veriyle her seferinde aynı sonucu elde edip etmediğimizi görebilmek için, 4'er kez öğrenme ve tahmin algoritması işletilmiştir. Her denemede:
            1. Veri seti içinden, kümenin son elemanlarından zaman aralığı kadar veri tahmin sonuçlarını karşılaştırmak üzere gerçek sonuç olarak saklanır. Geri kalan baştaki elemanlar eğitim için kullanılır.
            2. İleri yönelimli YSA ile öğrenme ve tahmin algoritmaları çalıştırılır (Hem girdi verilerini kaydırarak, hem de kaydırmadan tahmin yapılır, iki sonuç da daha sonra karşılaştırmak üzere kaydedilir.)
            3. Kısıtlanmış Boltzmann makinesi ile öğrenme ve tahmin algoritmaları çalıştırılır. (Hem girdi verilerini kaydırarak, hem de kaydırmadan tahmin yapılır, iki sonuç da daha sonra karşılaştırmak üzere kaydedilir.)
            4. Tahmin sonuçları ile gerçek sonuçlar karşılaştırılır ve hata değerleri hesaplanır.
            5. Sonuçlar bir veri yapısına kaydedilir.
3. Oluşan tüm sonuçlar ve hata değerleri html çizelge, log dosyası ve grafik olarak bir dizine kaydedilir.

```
public static void main(String[] args){
ArrayList<PredictionResult> predictionResults=new ArrayList<PredictionResult>();
for(int timeWindowParameter=0;timeWindowParameter<5;timeWindowParameter++){
        int timeWindowWidth=(timeWindowParameter+1)*2;
        System.out.println("\n\n\n\nTIME WINDOW WIDTH="+timeWindowWidth+"\n\n\n");
        DataProvider.prepareDataSets(timeWindowWidth);
        for(int dataSetNo=0;
            dataSetNo<DataProvider.getDataSets(timeWindowWidth).size();
            dataSetNo++){
                for(int tryTheSameDataNumber=0;tryTheSameDataNumber<4;tryTheSameDataNumber++){
                    BackPropPredictor backPropPredictor=new BackPropPredictor(timeWindowWidth);
                    ConditionalRBMPredictor  conditionalRBMPredictor=
                                                new ConditionalRBMPredictor(timeWindowWidth);
                    PredictionResult predictionResult=new PredictionResult();
                    predictionResult.timeWindowWidth=timeWindowWidth;
                    predictionResult.tryNumber=tryTheSameDataNumber;
                    double[]
sampleData=DataProvider.getDataSets(timeWindowWidth).get(dataSetNo);
                    predictionResult.dataSetLength=sampleData.length;
                    predictionResult.inputData=new double[sampleData.length-timeWindowWidth];
                     System.arraycopy(sampleData, 0, predictionResult.inputData,
                                                        0, sampleData.length-timeWindowWidth);
                     predictionResult.dataToBePredicted=new double[timeWindowWidth];
                  System.arraycopy(predictionResult.inputData,
                                predictionResult.inputData.length-timeWindowWidth,
                                predictionResult.dataToBePredicted, 0, timeWindowWidth);
                  predictionResult.dataSetNumber=dataSetNo;
                  predictionResult.crbmPrediction=conditionalRBMPredictor.predict(
predictionResult.inputData);
                  predictionResult.crbmPrediction.testError=
                        calculateError(predictionResult.crbmPrediction.predictedValues,
                                        predictionResult.dataToBePredicted);
                  predictionResult.backPropPrediction=backPropPredictor.predict(
predictionResult.inputData);
                  predictionResult.backPropPrediction.testError=
                        calculateError(predictionResult.backPropPrediction.predictedValues,
                                        predictionResult.dataToBePredicted);
                  predictionResults.add(predictionResult);
                  System.out.println(predictionResult.toString());
                }
        }
}

StringBuffer outputFileBuffer=new StringBuffer();
for(int i=0;i<predictionResults.size();i++){
        outputFileBuffer.append(predictionResults.get(i).toString());
}
try {
        FileUtils.writeFile("/var/log/PredictionServer/server.log", outputFileBuffer.toString());
} catch (IOException e) {
        e.printStackTrace();
}

// Ozet rapor
MainSummaryReport mainSummaryReport=new MainSummaryReport();
mainSummaryReport.generateAndSaveHTMLTableReport(predictionResults);
mainSummaryReport.generateAndSaveCharts(predictionResults);

//Ayrinti raporu
DetailReports detailReports=new DetailReports();
detailReports.generateAndSaveHTMLTableReport(predictionResults);
detailReports.generateAndSaveCharts(predictionResults);
Matrix.es.shutdown();
}
```

Şekil 4-3 Uygulamanın Ana Algoritmasının Java Dilinde İfadesi

4.2.1 Verilerin Normalize Edilmesi

Bütün tahmin yöntemlerinde göz önünde bulundurmamız gereken önemli bir nokta da verilerin normalizasyonudur. Eğer verileri normalize etmeden direk olarak kullanırsak, oluşturduğumuz model, aynı ölçekte olmayan veriler için (çok büyük değerler ile çok küçük değerler) yanlış sonuç üretebilir. Bunu önlemek için veriler, java dili ile ifadesi Şekil 4-4'teki gibi olan, aşağıdaki algoritmaya uygun olarak normalize edilir:

1. Eldeki sayı dizisi içinde en büyük (EBD) ve en küçük (EKD) değer bulunur.
2. Eldeki her değer için denklem 4-2'deki fonksiyon işletilir ve çıkan sonuç normalize edilmiş değerler dizisine yazılır.

$$NormalizeX = \frac{X - EKD}{EBD - EKD} \times (\ddot{U}STSINIR - ALTSINIR) + ALTSINIR \qquad 4\text{-}2$$

```
public static void prepareNormalizedData(){
        if (normalizedData == null) {
                for (int i = 0; i < allData.length; i++) {
                        if (min > allData[i])
                                min = allData[i];
                        else if (max < allData[i])
                                max = allData[i];
                }
                normalizedData = new double[allData.length];
                for (int i = 0; i < allData.length; i++) {
                        normalizedData[i] = (((allData[i] - min) / (max - min))
                                        * (HIGH - LOW) + LOW);
                }
        }
}
```

Şekil 4-4 Normalizasyon Algoritmasının Java Dilinde İfadesi

4.2.2 Veri Kümeciklerinin Oluşturuması

Zaman serileri tahmini yapan algoritmaları denerken 10 adet farklı

büyüklükte veri kümesi kullanılmıştır. Bu veri kümelerinin büyüklüğü zaman aralığı değeriyle orantılıdır. Veri kümecikleri hazırlanmadan önce veriler normalize edilmiştir. Veri kümeciklerinin Şekil 4-5'te java dilinde ifadesi bulunan oluşturulma algoritması aşağıdaki gibidir:

1. İlk veri kümeciğinin büyüklüğü zaman aralığının 4 katı olmak üzere, veri kümeciklerinin büyüklükleri "zaman aralığı" değerinin artan katı olacak şekilde belirlenir.
2. Ana veri kümesinden kopyalanacak olan veri kümeciğinin başlangıç indeksi olarak 0 ile 20 arasında rastgele bir sayı seçilir.
3. Bu rastgele sayıdan itibaren 1. Maddede belirlediğimiz küme büyüklüğü kadar veri, ana veri kümesinden alınır ve bir veri kümeciği oluşturulmuş olur.
4. 10 adet veri kümeciği elde edene kadar 1,2 ve 3 maddeleri tekrar edilir.

```
public static void prepareDataSets(int timeWindowWidth) {
        prepareNormalizedData();
        for (int i = 0; i < numberOfDataSets; i++) {
                int length = (i + 4) * timeWindowWidth;
                int startIndex = (int) (Math.random() * 20);
                double[] dataSet = new double[length];
                System.arraycopy(normalizedData, startIndex, dataSet, 0, length);
                normalizedDataSets.add(dataSet);
        }
}
```

Şekil 4-5 Veri Kümeciklerini Oluşturma Algoritmasının Java Dilinde İfadesi

4.2.3 İleri Yönelimli YSA ile Öğrenme ve Tahmin

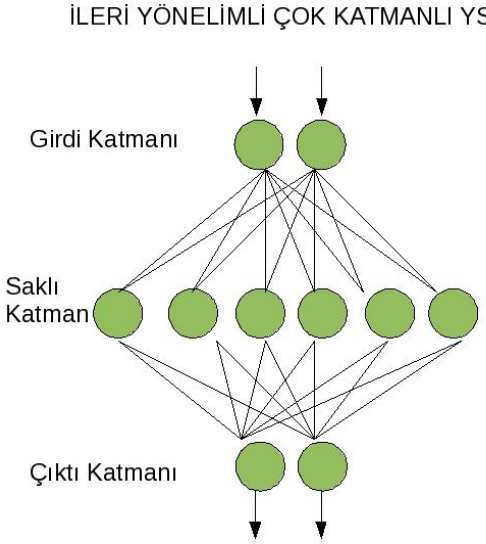
İleri yönelimli çok katmanlı YSA ile zaman serileri tahmini yaparken zaman aralığı (Time Window) değeri baz alınmıştır. Zaman aralığı değeri girdi ve çıktı hücre sayısına eşitlenerek, sadece zaman aralığı değiştiğinde otomatik olarak bütün yapının değişmesi sağlanmıştır.

Zaman serisindeki veriler girdi hücrelerine gerçel sayılar olarak verilmekte ve çıktılar gerçel sayılar olarak alınmaktadır. Arada yapılan tüm işlemler ve hata hesapları direk olarak gerçel sayılar (double) üzerinde yapılmaktadır.

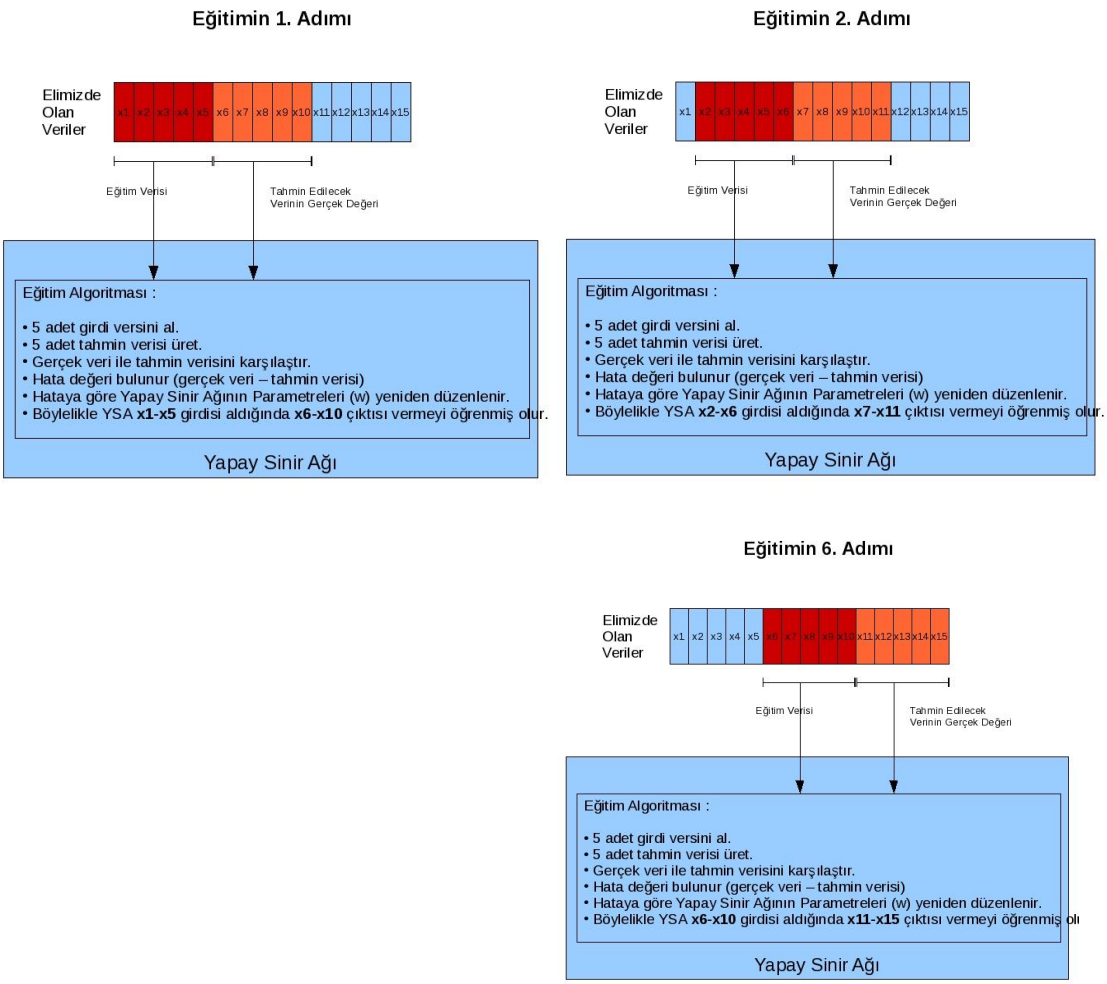
İleri yönelimli YSA'nın çizge yapısı (graph structure) girdi hücreleri, saklı hücreler ve çıktı hücrelerinden oluşmaktadır (şekil 4-6). Girdi hücrelerinin sayısı zaman aralığı büyüklüğü kadardır. Saklı hücreler ise girdi hücrelerinin (dolayısıyla zaman aralığı değerinin) birkaç katı kadardır (Hidden Layer Neurons) .

Saklı hücreler eğer çok az olursa az sayıda kombinasyon YSA tarafından öğrenilmiş olacağından, sonuçta elde edilen tahmin iyi olmayacaktır. Ancak bir noktadan sonra saklı hücre sayısı artsa da tahmin yakınsaması iyi olmamaktadır, bu nedenle deney yoluyla bir ortalama saklı hücre sayısı bulunmalıdır. Öğrenme zamanı, saklı hücre sayısı ve girdi/çıktı hücre sayısının çarpımı ile doğru orantılı olarak artmaktadır. Onun için gereksiz yere öğrenme zamanını arttırmamak için saklı hücre sayısı çok fazla olmamalıdır. Bu uygulama içinde deneysel olarak görülmüştür ki saklı hücre sayısı girdi hücre sayısının 2 katı olduğu durumlarda optimum öğrenme gerçekleşmektedir.

Zaman aralığı büyüklüğü kadar çıktı hücresi vardır. Eğer tek tahmin yapsaydık çıktı hücre sayısı 1 olurdu. Çoklu tahmin yapıldığı için çıktı hücresi de tahmin sayısı kadar olacaktır. Bu gerçeklemede (implementation) zaman aralığı değeri, girdi sayısı ve çıktı sayısı eşit olacak şekilde bir tahmin yapılmaktadır.



Şekil 4-6 İleri Yönelimli Çok Katmanlı YSA Çizge Yapısı [22]



Şekil 4-7 İleri Yönelimli YSA Eğitim Adımları

YSA'yı eğitirken gözlem verileri zaman aralığı büyüklüğü kadar aralıklarla YSA'ya girdi olarak verilir ve sonucunda oluşan tahmin değeri ile gerçek değerler karşılaştırılır. Her adımda çıktılar ile olması gereken gerçek değerler karşılaştırılır ve hata değerine göre ağırlık değerlerinde düzeltme yapılır.

Şekil 4-7'de örnek bir veri kümeciği ve zaman aralığı için eğitim adımları atılırken verilerin nasıl alındığı görülmektedir. 5 büyüklüğünde bir zaman aralığı için 20 büyüklüğünde bir veri kümesi oluşmaktadır. 20 verinin ilk 15'i eğitim için son 5 tanesi (zaman aralığı kadar) tahmin değerleri ile gerçek değerleri karşılaştırabilmek için ayrılmıştır. Şekil 4-7'de 1. 2. ve 6. eğitim adımları gösterilmiştir, aradaki adımlar diğerlerine benzemektedir. İlk 15 veri tamamen kaydırma yöntemiyle okunup bittiği

zaman eğitim de bitmiştir.

Şekil 4-8'de java dilindeki ifadesi görülen ileri yönelimli YSA'da öğrenme algoritması, "EPOCH" denilen adımlardan oluşur. Öğrenilecek verilerin üzerinden birçok defa (EPOCH) geçerek, her seferinde düzeltilen ağırlık değerleriyle, daha çok kombinasyonun denenmesi ve öğrenilmesi sağlanmaktadır.

```
for(int i=0;i<EPOCHS;i++){
        for(int j=0;j<inputData.length-timeWindowWidth-timeWindowWidth+1;j++){
                double[] timeWindow=new double[timeWindowWidth];
                System.arraycopy(inputData, j, timeWindow, 0, timeWindowWidth);
                double[] targetWindow=new double[timeWindowWidth];
                System.arraycopy(inputData, j+timeWindowWidth, targetWindow, 0, timeWindowWidth);
                for(int k=0;k<10;k++){
                        network.setInput(timeWindow);
                        network.propagate();
                        network.backPropagate(targetWindow);
                        returnValue.networkError=network.computeOutputError(targetWindow);
                        if(minimumNetError>returnValue.networkError){
                                minimumNetError=returnValue.networkError;
                                network.savedError=returnValue.networkError;
                                network.saveWeights();
                        }
                }
        }
}
```

Şekil 4-8 İleri Yönelimli YSA Öğrenme Algoritmasının Java Dilinde İfadesi

Döngü içerisinde zaman aralığı 1 kaydırılarak girdi değerlerinin tamamı sırayla YSA'ya verilir. Her adımda 10 kere aynı veri YSA'ya öğrenmek üzere verilir. 10 kere tekrar etmesinin sebebi, her seferinde farklı ağırlık değeri kombinasyonunun, hata değerini azaltmak için denenmesidir. En düşük hata değeri olan için ağırlık değerleri sabitlenir.

Her seferinde "setInput" metoduyla girdi değerleri alınır ve "propagate" metoduyla alınan girdilere karşın bir çıktı üretmesi sağlanır. Daha sonra "backPropagate" metoduyla, tahmin edilmesi gereken hedef değerler ile tahmin değerleri karşılaştırılır, çıkan hata değerleri geriye doğru yayılır ve ağırlık değerleri

bu hata değerlerine göre düzeltilir.

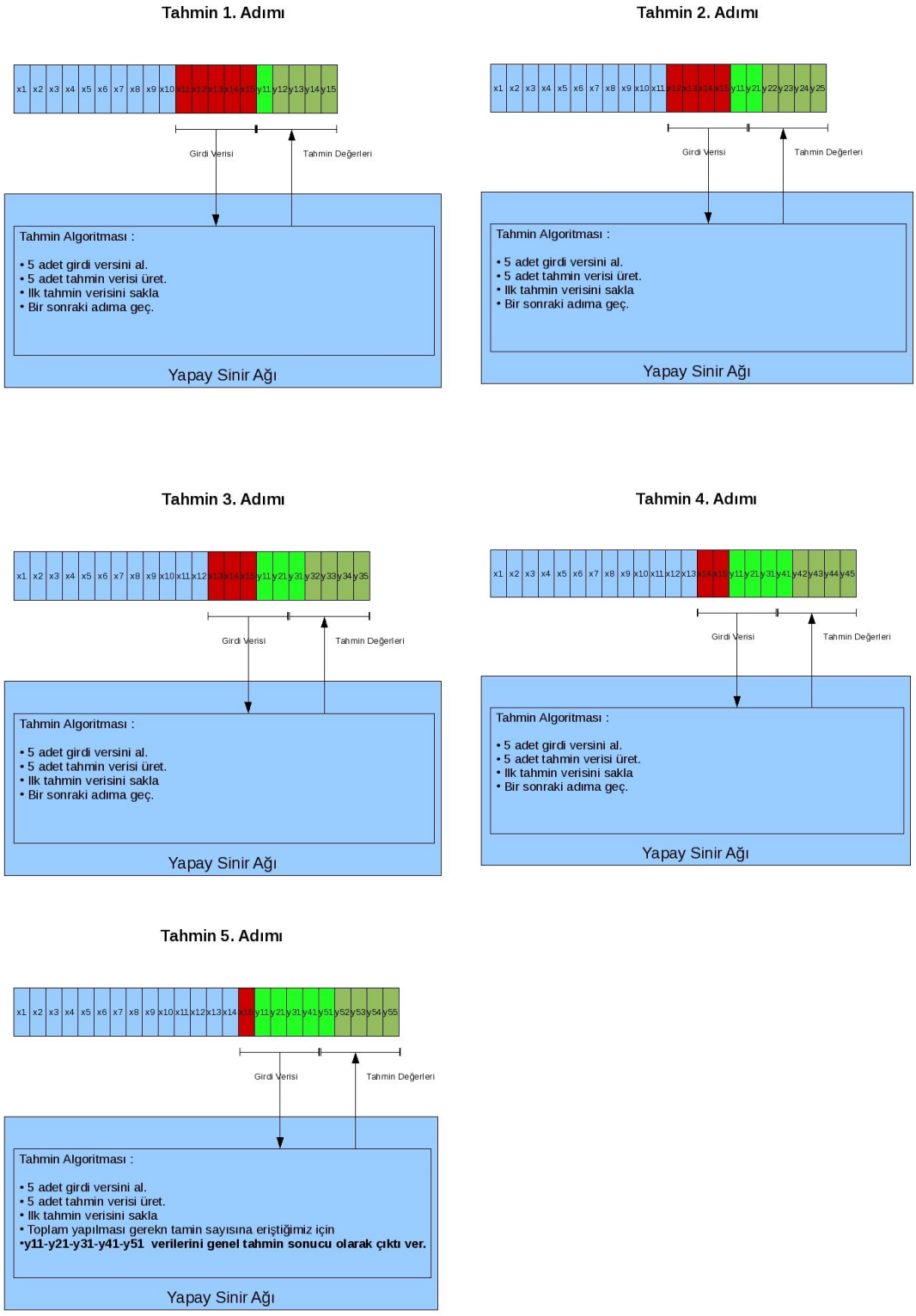
Şekil 4-9'da java dilinde ifadesi görülen "backPropagate" metoduyla, bir önceki katmandaki sinir hücrelerinin hata değerleri sıfırlanır, hata değerleri önceki katmana yayılır, bu katmandaki hücrelerin ağırlık değerleri düzeltilir. Şekil 4-10'da java dilinde ifadesi görülen "adjustWeights" algoritması ile her hücre kendi ağırlık değerlerini düzeltir.

```
public void backPropagate(){/
        if(previousLayer==null)return;
        // 1. distribute errors to previous layer before adjusting weights,
        // 1.1 first reset the errors
        for(int i=1;i<previousLayer.getUnits().size();i++){
                previousLayer.getUnits().get(i).resetError();
        }
        // 1.2 distribute the errors by weights
        for(int i=1;i<units.size();i++){
                units.get(i).distributeErrorToPreviousLayer();
        }
        // 2. adjust weights
        for(int i=0;i<units.size();i++){
                units.get(i).adjustWeights();
        }
}
```

Şekil 4-9 Geriye Yayılım Algoritmasının Java Dilinde İfadesi

```
public void adjustWeights(){
        // this unit is the j.th unit. i is the index of the input.
        // Zj=sum ( wji * xji)
        // output_j = sigmaF(Zj)
        // E= 1/2 * sum (tatget_j - ouput_j)
        // dE/dWji = dE/dZj * dSum/dWji = dE/dZj * Xji
        //          = - (target_j - ouput_j) * dOuput_j/ dZj * Xji
        //          = - (target_j - ouput_j) * dOuput_j/ dZj * Xji
        //          = - error_j * dOuput_j/ dZj * Xji
        //          = - error_j * (1-output_j) * output_j * Xji
        // DeltaWji= - learningRate * dE/dWji
        //          = - learningRate * - error_j * (1-output_j) * output_j * Xji
        //          = learningRate * error_j * (1-output_j) * output_j * Xji
        // Weight_j=  Weight_j + DeltaWji + Momentum * DeltaWjiCalculatedOneCycleBefore

        Iterator<Unit> iterator=input.keySet().iterator();
        while(iterator.hasNext()){
                Unit inputUnit=iterator.next();
                double weightAssociatedWithTheUnit=input.get(inputUnit).doubleValue();
                double weightDelta= Network.DEFAULT_LEARNING_RATE * error * (1-output) * output
                                * inputUnit.getOutput();
                double previousWeightDelta=0;
                if(weightDeltas.get(inputUnit)!=null) previousWeightDelta=weightDeltas.get(inputUnit).doubleValue();

                double currentWeight=weightAssociatedWithTheUnit+ weightDelta+
                                Network.DEFAULT_MOMENTUM* previousWeightDelta;
                input.put(inputUnit,currentWeight);

                weightDeltas.put(inputUnit, weightDelta);
        }
}
```

Şekil 4-10 Hata Düzeltme Alogritmasının Java Dilindeki İfadesi



Şekil 4-11 Tahmin Algoritmasında Girdi Adımları

Eğitim adımları bittikten sonra sıra tahmin adımlarına gelir. Tahmin yapılırken yine her adımda veriler birer kaydırılarak YSA'ya verilir. Yalnız verileri kaydırmadan yapılmış olan, ilk adımdaki tahmin değerleri de sonuçlar arasına kaydedilir.

Şekil 4-11'de gösterilen tahmin adımlarının hepsinde YSA zaman aralığı kadar çıktıyı tahmin değerleri olarak vermektedir. Kaydırmalı yöntemde her adımda verilen çıktının ilk değeri alınır geri kalanı atılır.

Örneğin zaman aralığı 5 olan bir YSA'da tahmin yaparken, tahminin birinci adımında (y_{11}, y_{12}, y_{13}, y_{14}, y_{15}) çıktıları elde edilir, bu çıktılardan sadece y_{11} değeri alınır ve geri kalanı atılır. Tahminin ikinci adımında (y_{21}, y_{22}, y_{23}, y_{24}, y_{25}) çıktıları elde edilir, y_{21} değeri alınır ve geri kalanı atılır. Aynı şekilde beşinci adıma kadar gidilir ve en sonunda her adımda elde edilen ilk değerler tahmin sonucunu bize verir. (y_{11}, y_{21}, y_{31}, y_{41}, y_{51})

4.2.4 Kısıtlanmış Boltzmann Makinesi ile Öğrenme ve Tahmin

Boltzmann Makinesindeki bütün hücrelerin girdileri ve çıktıları ikili (binary, 0 veya 1) değer almaktadır. Hücrelerin girdi/çıktı değerleri ikili değer alır ama aradaki bağlantıların ağırlık (weight) değerleri gerçel sayılardır. Bütün hücreler girdi değeri olarak ikili değerler aldıkları için zaman serilerindeki gerçel sayılar ikili sayılara çevrilerek Kısıtlanmış Boltzmann Makinesine verilmiştir. YSA'dan elde dilen ikili çıktılar ise gerçel sayılara çevrilmiştir. Şekil 4-12 ve 4-13'te çevrim algoritmasının java dilinde ifadesi görülmektedir. Gerçek verilerle tahmin verilerinin arasındaki farkı hesaplarken de bu çevrim uygulanmıştır. Uygulama içerisinde ikili olarak 32 basamak kullanılmaktadır. Dolayısıyla eğer zaman aralığı değeri 5 olursa, Kısıtlanmış Boltzmann Makinesinin girdi hücre sayısı ve çıktı hücre sayısı 160 olmaktadır.

Kısıtlanmış Boltzmann Makinesinde, Boltzmann Makinesinden farklı olarak tüm hücreler birbirlerine bağlı değildir. Hücreler "Görünür Hücreler" (Visible Layer Neurons) ve "Saklı Hücreler" (Hidden Layer Neurons) olarak iki gruba ayrılmıştır.

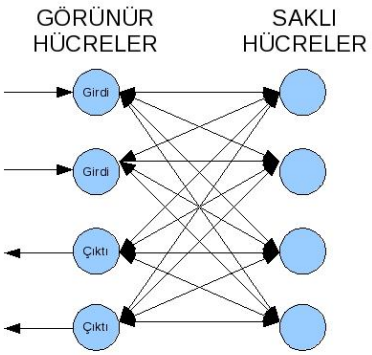
Görünür hücrelerin tamamı saklı hücrelerin tamamına bağlıdır. YSA'nın dışarı ile bağlantısını görünür hücreler sağlamaktadır. Uygulama içerisinde "Koşullu Kısıtlanmış Boltzmann Makinesi" kullanılmaktadır. Bu da, şekil 4-6'da görüldüğü gibi, görünür hücrelerin bir kısmının girdi hücresi, bir kısmının da çıktı hücresi olarak kullanılması demektir: Sonuç olarak oluşan çizge, aslında ileri yönelimli YSA'nın her hücre arasındaki bağının iki yönlü olduğu duruma benzer (şekil 4-14).

```
public final static int DIGIT_SIZE=32;

public final static double DOUBLE_TO_BINARY_PRESENTATION_INTERVAL=1/Math.pow(2, DIGIT_SIZE);

private double[] convertRealVectorToBinaryVector(double[] realVector){
        double[] binaryVector=new double[realVector.length*DIGIT_SIZE];
        for(int i=0;i<realVector.length;i++){
                double[] binaryRepresentaionOfDoubleValue=convertRealToBinary(realVector[i]);
                for(int j=0;j<binaryRepresentaionOfDoubleValue.length;j++){
                        binaryVector[i*DIGIT_SIZE+j]=binaryRepresentaionOfDoubleValue[j];
                }
        }
        return binaryVector;
}


private double[] convertRealToBinary(double value){
        double[] returnValue=new double[DIGIT_SIZE];
        double realValue=value/DOUBLE_TO_BINARY_PRESENTATION_INTERVAL;
        for(int i=returnValue.length-1;i>=0;i--){
                if(realValue-Math.pow(2, i)>0){
                        returnValue[i]=1;
                        realValue-=Math.pow(2, i);
                }
        }

        return returnValue;
}
```

Şekil 4-12 Gerçel Sayılardan İkili Sayılara Çevrim Algoritmasının Java Dilinde İfadesi

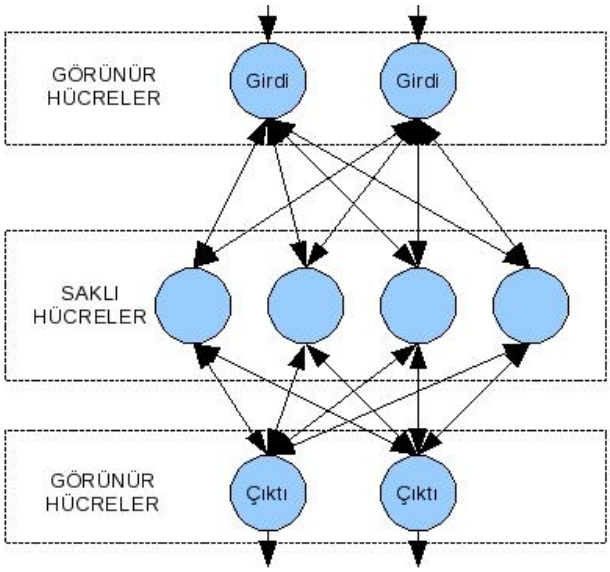
```
private double[] convertPredictedBinaryToReal(int[] binaryMatrix){
        double[] returnValue=new double[timeWindowWidth];

        for(int i=0;i<returnValue.length;i++){
                double realValue=0;
                for(int j=0;j<DIGIT_SIZE;j++){
                        if(binaryMatrix[i*DIGIT_SIZE+j]==1)realValue+=Math.pow(2, j);
                }
                returnValue[i]=realValue*DOUBLE_TO_BINARY_PRESENTATION_INTERVAL;
        }
        return returnValue;
}
```

Şekil 4-13 İkili Sayıların Gerçel Sayılara Çevrim Algoritmasının Java Dilinde İfadesi



Şekil 4-14 Koşullu Kısıtlanmış Boltzmann Makinesi

Öğrenme ve tahmin algoritmalarında ileri yönelimli YSA'larda olduğu gibi (şekil 4-7 ve şekil 4-11) zaman serisi verileri birer kaydırılarak alınır. Öğrenilecek verilerin üzerinden birçok defa (EPOCH) geçerek daha çok kombinasyonun öğrenilmesi sağlanmaktadır. Döngü içerisinde zaman aralığı 1 kaydırılarak girdi değerlerinin tamamı sırayla YSA'ya verilir. Her adımda 10 kere aynı veri YSA'ya öğrenmek üzere verilir. 10 kere tekrar etmesinin sebebi, her seferinde farklı ağırlık değeri kombinasyonu hata değerini azaltmak için denenmektedir. En düşük hata değeri olan için ağırlık değerleri sabitlenir.



Şekil 4-15 İleri Yönelimli YSA Gibi Görünen Koşutlu Kısıtlanmış Boltzmann Makinesi

Şekil 4-17 ve 4-18'de java dilinde ifadesi bulunan öğrenme algoritmasında "visibleInputData" girdi değerleri aslında ikili (binary) olmasına rağmen veri yapısı uyumluluğu için "double[][]" tanımlanmıştır.

Uygulamada kullanılan öğrenme algoritması dört bölümden oluşmaktadır :

1. Verilerin girildiği bölüm (Positive Phase)
2. Boltzmann makinesinin serbestçe çalıştırıldığı bölüm (Negative Phase)
3. Sinir hücreleri arasındaki bağların ağırlık değerlerinin düzeltildiği bölüm (Update Weights)
4. YSA'nın öğrendiği veriler içinde bile bir hata payı bulunmaktadır. Bu hata payının hesaplandığı bölüm (Calculate Error)

```
for(int i=0;i<EPOCHS;i++){

        System.out.println("Epoch:"+i);

        for(int j=0;j<inputData.length-timeWindowWidth-timeWindowWidth+1;j++){

                double[] timeWindow=new double[timeWindowWidth];
                System.arraycopy(inputData, j, timeWindow, 0, timeWindowWidth);
                double[] targetWindow=new double[timeWindowWidth];
                System.arraycopy(inputData, j+timeWindowWidth, targetWindow, 0, timeWindowWidth);
                double minError=Double.MAX_VALUE;
                for(int k=0;k<10;k++){

                        returnValue.networkError=crbmLearner.Learn(
                                                convertRealVectorToBinaryVector(timeWindow),
                                                convertRealVectorToBinaryVector(targetWindow));
                        if(minError>returnValue.networkError){
                                minError=returnValue.networkError;
                                crbm.saveWeights();
                        }
                }

                crbm.restoreWeights();

                returnValue.networkError=minError;
        }
}
```

Şekil 4-166 Kısıtlanmış Boltzmann Makinesi Öğrenim Döngüsü Algoritmasının Java Dilinde İfadesi

Şekil 4-17'deki algoritmada "negative phase" yazısına kadar olan yere kadar Boltzmann makinesi pozitif fazda çalıştırılmaktadır. Yani hiçbir hücrenin değerleri güncellenmemektedir. Sadece negatif faz'daki matris çarpım işlemlerinden sonra hücre değerleri güncellenmektedir. Bu da her öğrenme aşamasında, kuram kısmında belirtildiği gibi, Kısıtlanmış Boltzmann Makinesinin bir pozitif bir de negatif fazda çalıştırıldığını göstermektedir. Şekil 4-18'deki algoritma bölüm 3.4.7'de anlatıldığı gibi, güncelleme değeri pozitif fazda elde edilen değerden negatif fazda elde edilen değerin çıkarılıp veri uzunluğuna (N) bölünmesiyle elde edilir.

```
  public double Learn(double[][] visibleInputData, double[][] visibleOutputData) {
      double[][] outputHiddenActivities = Matrix.multiply(visibleOutputData, crbm.outputWeights);
      double[][] visibleHiddenActivities = Matrix.multiply(visibleInputData, crbm.weights);
      double[][] hiddenActivities = crbm.hiddenLayer.getActivationProbabilities(Matrix.add(outputHiddenActivities,
visibleHiddenActivities));
      double[][] hiddenData = crbm.hiddenLayer.generateData(hiddenActivities);

      // negative phase , where the network is allowed to run freely, i.e. no units have their state determined by external data
      double[][]negPhaseVisibleOutputActivities
=crbm.visibleOutputLayer.getActivationProbabilities(Matrix.multiplyTranspose(hiddenData, crbm.outputWeights));
      double[][] visibleActivities = crbm.visibleLayer.getActivationProbabilities(Matrix.multiplyTranspose(hiddenData,
crbm.weights));
      double[][] negativeOutputHiddenActivities = Matrix.multiply(negPhaseVisibleOutputActivities, crbm.outputWeights);
      double[][] negativeVisibleHiddenActivities = Matrix.multiply(visibleInputData, crbm.weights);
      double[][] negativePhaseHiddenActivities =
crbm.hiddenLayer.getActivationProbabilities(Matrix.add(negativeOutputHiddenActivities, negativeVisibleHiddenActivities));

      // Update weights
      double[][] visibleOutputWeightUpdates = getConnectionWeightUpdates(crbm.outputWeights, visibleOutputData,
hiddenActivities, negPhaseVisibleOutputActivities, negativePhaseHiddenActivities);
      double[][] visibleWeightUpdates = getConnectionWeightUpdates(crbm.weights,visibleInputData, hiddenActivities,
visibleActivities, negativePhaseHiddenActivities);
      updateVisibleWeights(visibleOutputWeightUpdates,crbm.outputWeights,prevVisibleOutputWeightUpdates);
      prevVisibleOutputWeightUpdates = visibleOutputWeightUpdates;
      updateVisibleWeights(visibleWeightUpdates,crbm.weights,prevVisibleWeightUpdates) ;
      prevVisibleWeightUpdates = visibleWeightUpdates;
      crbm.hiddenLayer.updateBiases(hiddenData, negativePhaseHiddenActivities);
      crbm.visibleOutputLayer.updateBiases(visibleOutputData, negPhaseVisibleOutputActivities);

      //Calculate Error
      outputHiddenActivities = Matrix.multiply(visibleOutputData, crbm.outputWeights);
      visibleHiddenActivities = Matrix.multiply(visibleInputData, crbm.weights);
      hiddenActivities = crbm.hiddenLayer.getActivationProbabilities(Matrix.add(outputHiddenActivities,
visibleHiddenActivities));
      visibleActivities =crbm.visibleLayer.getActivationProbabilities(Matrix.multiplyTranspose(hiddenData, crbm.weights));

      double error=0;
      for(int i=0;i<visibleOutputData.length;i++){
          error+=Matrix.getSquaredError(convertPredictedBinaryToReal(visibleOutputData[i],visibleOutputData.length),
convertPredictedBinaryToReal(convertActivationToBinaryVector(visibleActivities[i]),visibleOutputData.length));
      }
      return Math.sqrt(error);
   }
```

Şekil 4-177 Kısıtlanmış Boltzmann Makinesi Öğrenme Algoritmasının Java Dilinde İfadesi

```
public static double[][] getConnectionWeightUpdates(double[][] connectionWeights,
                                    double[][] data, double[][] hiddenActivities,
                                    double[][] generatedData, double[][] negativePhaseHiddenActivities) {

   final int numVisibleUnits = connectionWeights.length;
   final int numHiddenUnits = connectionWeights[0].length;
   double[][] weightUpdates = new double[numVisibleUnits][numHiddenUnits];

   final double[][] positivePhaseProduct = Matrix.transposeMultiply(data, hiddenActivities);
   final double[][] negativePhaseProduct = Matrix.transposeMultiply(generatedData, negativePhaseHiddenActivities);

   for (int v = 0; v < numVisibleUnits; v++) {
      for (int h = 0; h < numHiddenUnits; h++) {
         weightUpdates[v][h] = (positivePhaseProduct[v][h] - negativePhaseProduct[v][h]) / data.length;
      }
   }
   return weightUpdates;
}
```

Şekil 4-188 Kısıtlanmış Boltzmann Makinesi Öğrenme Algoritmasının Java Dilinde İfadesi

5 UYGULAMA SONUÇLARI

Bölüm 4'te anlatılan uygulamada 5 ayrı zaman aralığı değeri için değişik uzunlukta 10 adet veri kümesi oluşturulmuştur. Aynı veri kümesi üzerinde her seferinde aynı sonucu elde edip edemediğimizi görmek için 4'er kez hem kısıtlanmış Boltzmann makinesi ile hem de ileri yönelimli YSA ile öğrenme ve tahmin algoritmaları çalıştırılmıştır.

Uygulama ile yapılmış olan deney Pentium 2.8GHz işlemcili 1GB RAM'li CentOS 5.3 bilgisayar üzerinde çalıştırılmıştır. Program çalışma süresi boyunca işlemciyi %100 oranında kullanmıştır, bellek tüketimi %1 seviyesinde kalmıştır.

Çizelge 5-1'de yapılan deneyin hata sonuçları ve işlem zamanları görülmektedir. Bu çizelgenin sütunlarının açıklaması aşağıdaki gibidir:

1. Zaman Aralığı: Kullanılan zaman aralığı büyüklüğüdür. Bu büyüklüğe bağlı olarak veri kümesi büyüklükleri, tahmin yapacak YSA'ların girdi ve çıktı

büyüklükleri değişmektedir.

2. Veri Kümesi Büyüklüğü: Gözlem verisi sayısıdır. Örneğin veri kümesi büyüklüğü 24 ise ve zaman aralığı 2 ise, YSA'lar 22 veri ile eğitilecek ve son 2 veriyi tahmin etmeye çalışacaklardır. Tahmin sonuçları ile gerçek sonuçlar arasındaki hata payı da "Test Hata Değeri" başlıklarıyla aynı çizelgede verilmektedir.
3. İleri Yönelimli YSA Ortalama Test Hata Değeri: Verilen zaman aralığı ve veri kümesi büyüklüğü için İleri Yönelimli YSA ile 4 adet deney yapılır. Hepsinde çıkan hata değerinin (tahmin ile gerçek sonuç farkının) aritmetik ortalamasıdır.
4. İleri Yönelimli YSA Minimum Test Hata Değeri: Verilen zaman aralığı ve veri kümesi büyüklüğü için İleri Yönelimli YSA ile 4 adet deney yapılır. Bu deneylerde çıkan test hata değerlerinin en küçüğüdür.
5. İleri Yönelimli YSA Kaydırmadan Ortalama Test Hata Değeri: Tahmini kaydırma yöntemiyle değil de , öğrenme bittikten sonra direk olarak yapsaydık ortalama hata değerlerimizin neler olacağını söyler.
6. İleri Yönelimli YSA Kaydırmadan Ortalama Test Hata Değeri: Tahmini kaydırma yöntemiyle değil de , öğrenme bittikten sonra direk olarak yapsaydık minimum hata değerlerimizin neler olacağını söyler.
7. İleri Yönelimli YSA Ortalama İşlem Zamanı : Verilen zaman aralığı ve veri kümesi ile öğrenme ve tahmin için geçen toplam süre verilmektedir.
8. Boltzmann Makinesi Ortalama Test Hata Değeri: Verilen zaman aralığı ve veri kümesi ile yapılan 4 tahminin hata değerlerinin ortalamasıdır.
9. Boltzmann Makinesi Minimum Test Hata Değeri: Verilen zaman aralığı ve veri kümesi ile yapılan 4 tahminin hata değerlerinin en küçüğüdür.
10. Boltzmann Makinesi Kaydırmadan Ortalama Test Hata Değeri: Kaydırmadan, direk, yapılan 4 tahminin (bir veri kümesi için 4 kere üst üste deniyoruz) hata değerleri ortalamasıdır.
11. Boltzmann Makinesi Kaydırmadan Minimum Test Hata Değeri: Kaydırmadan, direk, yapılan 4 tahminin hata değerlerinin en küçüğüdür.
12. Boltzmann Makinesi Ortalama İşlem Zamanı : Verilen zaman aralığı ve veri kümesi ile öğrenme ve tahmin için geçen toplam süre verilmektedir.

Zaman Aralığı	Veri Kümesi Büyüklüğü	İleri Yönelimli YSA Ortalama Test Hata Değeri	İleri Yönelimli YSA Minimum Test Hata Değeri	İleri Yönelimli YSA Kaydırmadan Ortalama Test Hata Değeri	İleri Yönelimli YSA Kaydırmadan Minimum Test Hata Değeri	İleri Yönelimli YSA Ortalama İşlem Zamanı	Boltzmann Makinesi Ortalama Test Hata Değeri	Boltzmann Makinesi Minimum Test Hata Değeri	Boltzmann Makinesi Kaydırmadan Ortalama Test Hata Değeri	Boltzmann Makinesi Kaydırmadan Minimum Test Hata Değeri	Boltzmann Makinesi Ortalama İşlem Zamanı
2	8	0,06496	0,05727	0,06428	0,06947	95,2	0,13091	0,010202	0,062752	0,055923	1320,2
2	10	0,10911	0,08735	0,10691	0,08524	152,5	0,07587	0,048198	0,068937	0,093238	2027,5
2	12	0,04318	0,04227	0,04301	0,04495	204,75	0,00631	5,36E-10	0,007937	4,96E-04	2870
2	14	0,26743	0,19878	0,26257	0,19196	266	0,10817	2,97E-08	0,049774	0,128906	3656,75
2	16	0,16428	0,1566	0,15843	0,15715	321	0,26816	0,139107	0,063766	2,38E-07	4470,75
2	18	0,1557	0,14974	0,15352	0,15619	389,75	0,0827	0,007812	0,03615	3,05E-05	5320,5
2	20	0,07866	0,0745	0,07728	0,07632	432	0,12984	4,76E-07	0,067111	1,91E-06	6197,5
2	22	0,08887	0,04007	0,08714	0,03891	515,75	0,08572	0,007755	0,067049	0,057785	6986,75
2	24	0,29781	0,25932	0,28371	0,24445	561,25	0,25391	0,064461	0,162156	3,78E-10	8068,25
2	26	0,0501	0,02442	0,04925	0,05827	627,25	0,14769	2,13E-06	0,032085	0,023185	8917
4	16	0,30098	0,26124	0,31536	0,39738	418,25	0,32355	5,69E-10	0,082901	0,27623	8360,75
4	20	0,31071	0,16757	0,35912	0,38694	737,5	0,28052	6,85E-10	6,85E-10	6,85E-10	15107,5
4	24	0,11601	0,10947	0,16392	0,19902	1098,8	0,21082	0,161531	0,087285	5,57E-10	21162,8
4	28	0,28091	0,24518	0,37464	0,38788	1436,8	0,28517	6,85E-10	6,10E-05	6,85E-10	27630,3
4	32	0,16828	0,12784	0,27895	0,25683	1734	0,27366	0,234274	0,153965	0,232945	34108,8
4	36	0,31191	0,28644	0,38868	0,38347	2061	0,09484	1,89E-08	0,043777	6,63E-10	40214,5
4	40	0,14149	0,12984	0,18721	0,18158	2419,8	0,21506	0,007813	2,44E-04	9,77E-04	45274,3
4	44	0,29763	0,20739	0,29006	0,28947	2745,8	0,2366	7,36E-10	0,036474	7,36E-10	51648
4	48	0,12386	0,11063	0,29582	0,32183	3043,8	0,23424	0,126753	0,127212	0,265488	57756,8
4	52	0,20891	0,19274	0,27311	0,25845	3422	0,13017	0,016729	0,016458	5,34E-10	64199,5
6	24	0,20243	0,16596	0,37366	0,35993	1167,8	0,1563	6,64E-10	0,062974	0,251895	23932
6	30	0,17897	0,11666	0,32644	0,29137	2174,8	0,09834	4,42E-10	0,098339	4,42E-10	44131,8
6	36	0,2454	0,20666	0,56515	0,65513	3170,5	0,38576	8,83E-10	0,031499	9,77E-04	63319,8
6	42	0,17373	0,07859	0,48488	0,39441	4125,8	0,16619	6,22E-10	0,105281	6,22E-10	83619,3
6	48	0,164	0,11187	0,47548	0,5451	5193,3	0,0422	3,95E-10	0,066081	0,264324	103823
6	54	0,43231	0,3256	0,43665	0,43805	6080,3	0,33427	0,27981	0,334266	0,27981	123307
6	60	0,22425	0,18872	0,39553	0,47125	7122,8	0,14672	6,71E-10	0,03498	6,71E-10	144033
6	66	0,13525	0,10828	0,51119	0,54964	8052,8	3,95E-10	3,95E-10	7,63E-06	3,05E-05	163284
6	72	0,27083	0,22531	0,4682	0,544	9104,5	0,13497	8,52E-10	0,160243	0,249992	183653
6	78	0,5088	0,12297	0,54958	0,6912	10018	0,07344	5,46E-10	0,106696	5,46E-10	204025
8	32	0,42155	0,38189	0,17105	0,16249	2518,3	0,33623	9,00E-10	0,147665	9,00E-10	50116,3
8	40	0,4852	0,42109	0,42573	0,3265	4740,8	0,34619	0,081507	0,099371	0,081507	96000
8	48	0,56516	0,37864	0,48338	0,35262	6921	0,3507	0,145997	0,108536	9,98E-10	140453
8	56	0,8126	0,77481	0,61044	0,57133	9042,8	0,2767	8,54E-10	0,290864	0,563041	181783
8	64	0,90897	0,65729	0,86439	0,96514	11340	0,09725	7,90E-10	7,90E-10	7,90E-10	217444
8	72	0,73628	0,68898	0,566	0,52235	13436	0,45855	0,204429	0,325908	9,00E-10	261565
8	80	0,79537	0,29225	0,9312	0,40396	15797	0,03624	7,62E-10	7,62E-10	7,62E-10	316843
8	88	0,76342	0,50958	0,84559	1,18787	18170	0,03601	7,71E-10	0,011959	7,71E-10	352674
8	96	0,48752	0,47493	0,46814	0,47193	20076	0,07003	8,37E-10	0,07003	8,37E-10	402514
8	104	0,41153	0,36713	0,43947	0,69256	22246	0,10322	7,36E-10	0,035181	0,007813	430257
10	40	0,40015	0,30096	0,19677	0,22471	4590	0,11133	9,85E-10	9,85E-10	9,85E-10	88686
10	50	0,44803	0,29637	0,18964	0,18459	8613	0,00874	1,03E-09	1,03E-09	1,03E-09	166452
10	60	0,39857	0,36345	0,2318	0,22405	13103	0,06801	8,52E-10	8,52E-10	8,52E-10	247611
10	70	0,37194	0,27079	0,24878	0,29449	16728	5,63E-10	5,63E-10	5,63E-10	5,63E-10	327631
10	80	1,25429	1,11854	1,43706	1,43333	21254	0,08468	7,79E-10	0,175759	0,125004	407144
10	90	0,59591	0,42473	0,62309	0,97346	24665	0,1328	8,52E-10	0,131005	8,52E-10	534361
10	100	0,48044	0,40785	0,39691	0,46027	30343	0,16957	0,131896	0,158086	0,133693	688848
10	110	0,60283	0,58127	0,78828	0,78101	33306	0,07973	8,05E-10	0,032505	0,129042	784103
10	120	0,79893	0,67133	0,75065	0,78214	39506	0,0538	8,73E-10	0,053799	0,215197	886343
10	130	3,38686	0,80905	0,88906	0,22103	170841	0,5285	9,98E-10	0,528503	9,98E-10	3925374

Çizelge 5-1 Deney Hata Değerleri , Çalışma Zamanları

Deneyler başlamadan önce bütün veriler 0 ile 1 aralığına normalize edilmektedir. Dolayısıyla elde edilen tahmin değerleri de 0 ile 1 aralığındadır. Çizelge 5-1'de verilen hata değerleri, 0 ile 1 arasında kalan, tahmin değerleri ile gerçek değerler arasındaki Öklid uzaklığıdır. Ortalama değerlerin yanında minimum değerlerin de verilmesinin sebebi, Kısıtlanmış Boltzmann makinesinin, yakınsadığı zaman, gerçek verilere çok daha yakın tahmin değerleri verdiğini göstermektir.

İşlem zamanı değerleri ise bir YSA için öğrenme ve tahmin için geçen toplam zamanın milisaniye cinsinden ifadesidir. Bir YSA'da verileri kaydırarak veya kaydırmadan tahmin yapmak, öğrenme zamanını değiştirmediği için her iki tahmin tipi için de Çizelge 5-1'de aynı işlem zamanı verilmiştir.

Çizelge 5-1'e bakarak aşağıdaki sonuçlara ulaşılabilmektedir:

1. Kısıtlanmış Boltzmann Makinesi, İleri Yönelimli YSA'ya göre çok daha yavaş çalışmaktadır.
2. 50 deneyin 39'unda Kısıtlanmış Boltzmann Makinesi, İleri Yönelimli YSA'ya göre ortalama hata değeri olarak daha düşük sonuç vermiştir. Küçük zaman aralıkları için (2, 4) ortalama hata değerleri olarak İleri Yönelimli YSA daha iyi sonuç verse de, zaman aralığı büyüdükçe Kısıtlanmış Boltzmann Makinesi daha iyi sonuç vermektedir.
3. 50 deneyin 47'sinde Kısıtlanmış Boltzmann Makinesi daha düşük minimum hata değeri vermektedir. Bir deney için aynı verilerle 4 alt deney (tahmin) yapılmaktadır, 4 tahmin değeri içinde en düşük olanı minimum hata değerini vermektedir. Dolayısıyla eğer stabil hale getirilebilirse (her denemede minimum hata değerini verebilirse), Kısıtlanmış Boltzmann Makinesi %94 oranında daha iyi sonuç verecektir.
4. 50 deneyin 34'ünde Kısıtlanmış Boltzmann Makinesi 10^{-6} değerine kadar sonuca yaklaşmıştır (minimum hata değeri 10^{-6} 'nın altındadır).
5. İleri yönelimli YSA için 50 deneyin 25'inde verileri kaydırarak yapılan tahminlerin ortalama hata değerleri verileri kaydırmadan yapılan

tahminlerden daha iyi sonuç vermiş. Bu demek oluyor ki ortalama hata değerleri bakımından bu iki tip tahmin için bir farklılık olmamaktadır.

6. İleri yönelimli YSA için 50 deneyin 36'sında verileri kaydırmadan yapılan tahminlerin minimum hata değerleri verileri kaydırarak yapılan tahminlerden daha iyi sonuç vermiş. Bu demek oluyor ki minimum hata değerleri bakımından bu iki tip tahmin için verileri kaydırmadan tahmin yapmak daha iyi sonuç vermektedir.
7. Kısıtlanmış Boltzmann makinesi için 50 deneyin 37'sinde verileri kaydırmadan yapılan tahminlerin ortalama hata değerleri verileri kaydırarak yapılan tahminlerden daha iyi sonuç vermiş. Bu demek oluyor ki ortalama hata değerleri bakımından verileri kaydırmadan yapılan tahminler daha iyi sonuç vermektedir.
8. Kısıtlanmış Boltzmann makinesi için 50 deneyin 40'ında verileri kaydırarak yapılan tahminlerin minimum hata değerleri verileri kaydırmadan yapılan tahminlerden daha iyi sonuç vermiş. Bu demek oluyor ki minimum hata değerleri bakımından verileri kaydırarak yapılan tahminler daha iyi sonuç vermektedir.

Elde edilen sonuçlara bakarak söyleyebiliriz ki, ortalama hata değerleri açısından en iyi sonucu veren Kısıtlanmış Boltzmann Makinesinde verileri kaydırmadan tahmin yapmaktır, minimum hata değerleri açısından ise en iyi sonucu veren Kısıtlanmış Boltzmann Makinesinde verileri kaydırarak tahmin yapmaktır.

Öte yandan EK-A Tahmin Sonuçları çizelgesindeki verilere bakarak İleri Yönelimli YSA hiç bir zaman tam isabetli tahmin sonucu vermezken, Kısıtlanmış Boltzmann Makinesi 67 adet tam isabetli sonuç vermiştir, isabetli sonuçlar kırmızı ile bu çizelgede işaretlenmiştir. Bu yönüyle de Kısıtlanmış Boltzmann Makinesi , ileri yönelimli YSA'ya göre daha iyi tahmin sonuçları üretmektedir.

Hem İleri Yönelimli YSA'larda hem de Kısıtlanmış Boltzmann Makinesinde aynı veri kümesi aynı zaman aralığı ile 4'er kez tahmin yapıp her seferinde farklı sonuç elde edebiliyoruz. Bunun nedeni :

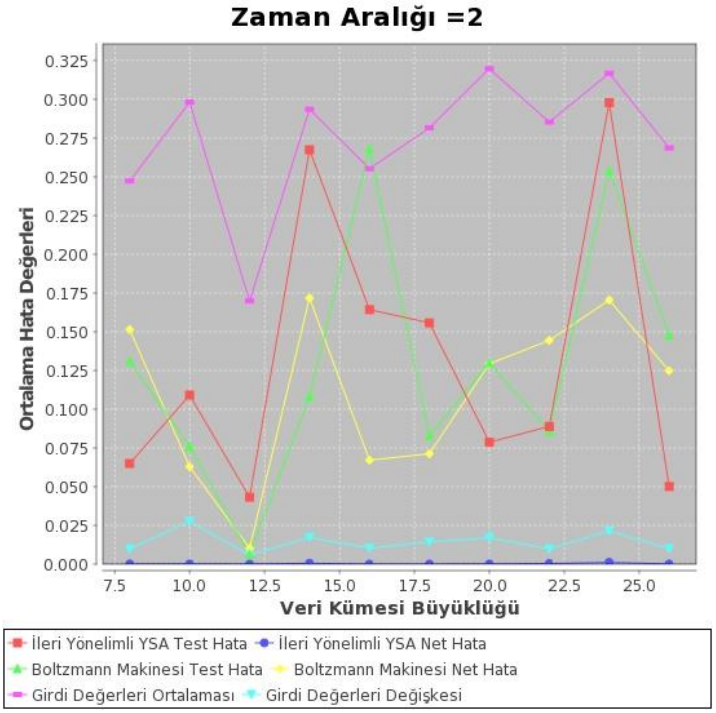
1.İleri Yönelimli YSA'da ağırlık değerlerini ilklendirirken rasgele (Math.random()) değer atanmaktadır. Bu da öğrenme algoritmaları ve veriler aynı olsa bile , başlangıç noktaları farklı olduğu için farklı sonuçlara yol açmaktadır.

2.Kısıtlanmış Boltzmann Makinesinde hem ağırlık değerlerini ilklendirirken , hem de öğrenme algoritmasının içinde rastgelelik olduğu için, her seferinde farklı sonuç vermektedir. Ancak deneylerde dikkat çeken nokta İleri Yönelimli YSA'ya göre gerçek sonuca daha yakın sonuçlar elde edilmektedir.

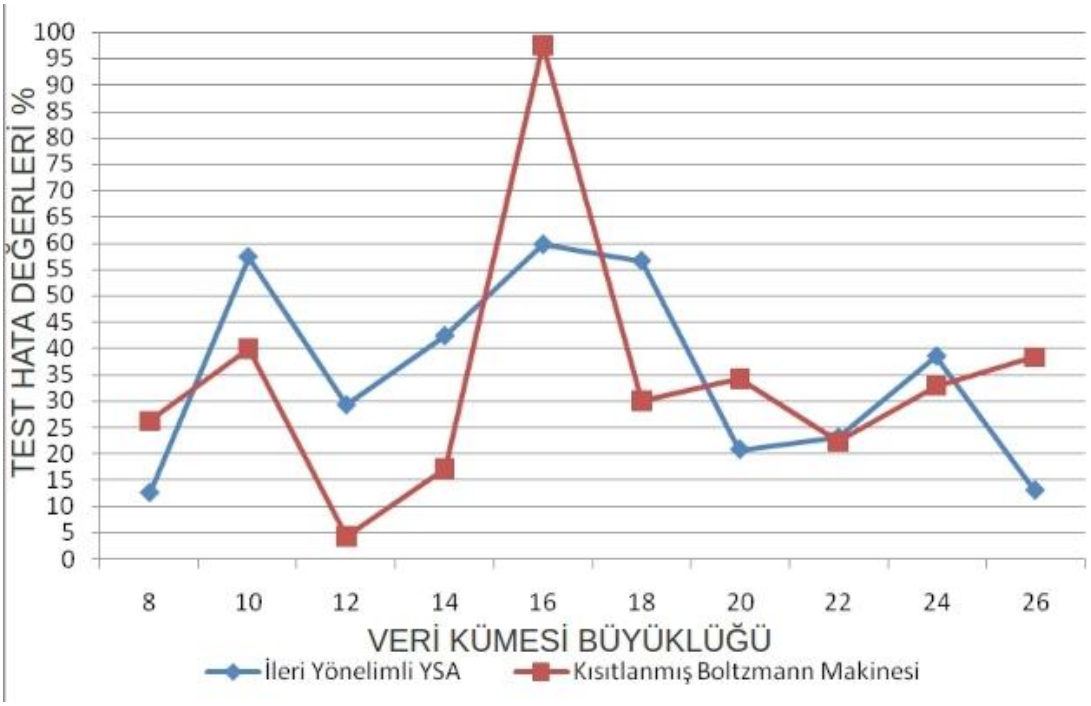
5.1 Zaman Aralığı 2 İçin Tahmin Sonuçlarının İncelemesi

Zaman aralığı değeri 2 için 10 adet farklı uzunlukta örnek veri kümesi ile deneyler yapılmıştır, bu deneylerin ortalama hata değerleri grafikleri Şekil 5-1'de, ortalama hata değerleri çizelgesi de Çizelge 5-1'de görülmektedir.

Şekil 5-1'deki grafiğe bakarak söyleyebiliriz ki, zaman aralığı 2 için, veri kümesinin büyüklüğü, yani tahmin yapılmadan önce öğrenilen gözlem verilerinin çokluğunun tahmin sonuçlarına etkisi azdır. Ortalama olarak standart bilinen İleri Yönelimli YSA ve Kısıtlanmış Boltzmann Makinesi yakın sonuçlar vermektedir.



Şekil 5-1 Zaman Aralığı 2 İçin Ortalama Hata Değerleri



5-2 Zaman Aralığı 2 İçin Ortalama Test Hata Yüzde Değerleri

Şekil 5-1'de görülen "Net Hata" grafikleri ise, YSA'ların ağ hatası (network error) değerlerini belirtmektedir ve bu hata değerleri YSA ile tahmin yapmadan önce bilinebilmektedir. Dolayısıyla eğer "Net Hata" değerleri "Test Hata" değerleri ile uyumlu ise, YSA tarafından verilen tahmin değerlerine "Net Hata" değerine bakarak güvenebiliriz. Şekil 5-1'deki grafikte görüldüğü üzere Kısıtlanmış Boltzmann Makinesinde "Net Hata" değeri "Test Hata" değeriyle uyumludur, yani Öklid uzaklıkları küçüktür.

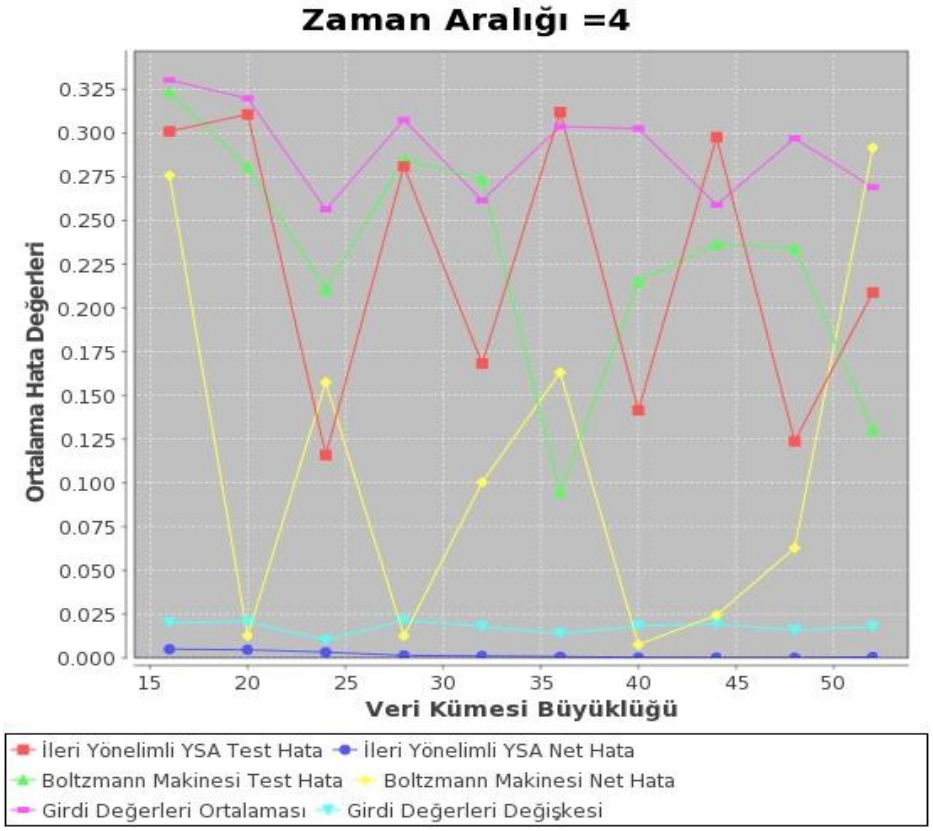
Şekil 5-1'de görülen hata değerleri, EK-A Tahmin Sonuçları çizelgesinde görülen tahmin değerleri ve gerçek değerler arasındaki Öklid uzaklıklarının aritmetik ortalamasıdır. Örneğin şekil 5-1'de kırmızı renkle belirtilmiş "İleri Yönelimli YSA Test Hata" grafiğindeki ilk veri olan değer (0,6 civarındaki değer), veri kümesi büyüklüğü 8'e denk gelen değer, aşağıdaki gibi elde edilmektedir:

1. EK-A Tahmin Sonuçları çizelgesinde zaman aralığı 2 ve veri büyüklüğü 8 için İleri Yönelimli YSA ile 4 adet tahmin yapılmıştır. Bu tahminlerin her birinin gerçek verilerle olan Öklid uzaklığı bulunur.
2. Bulunan bu Öklid uzaklıklarının aritmetik ortalaması alınır, bu da "Ortalama Test Değerini" verir.

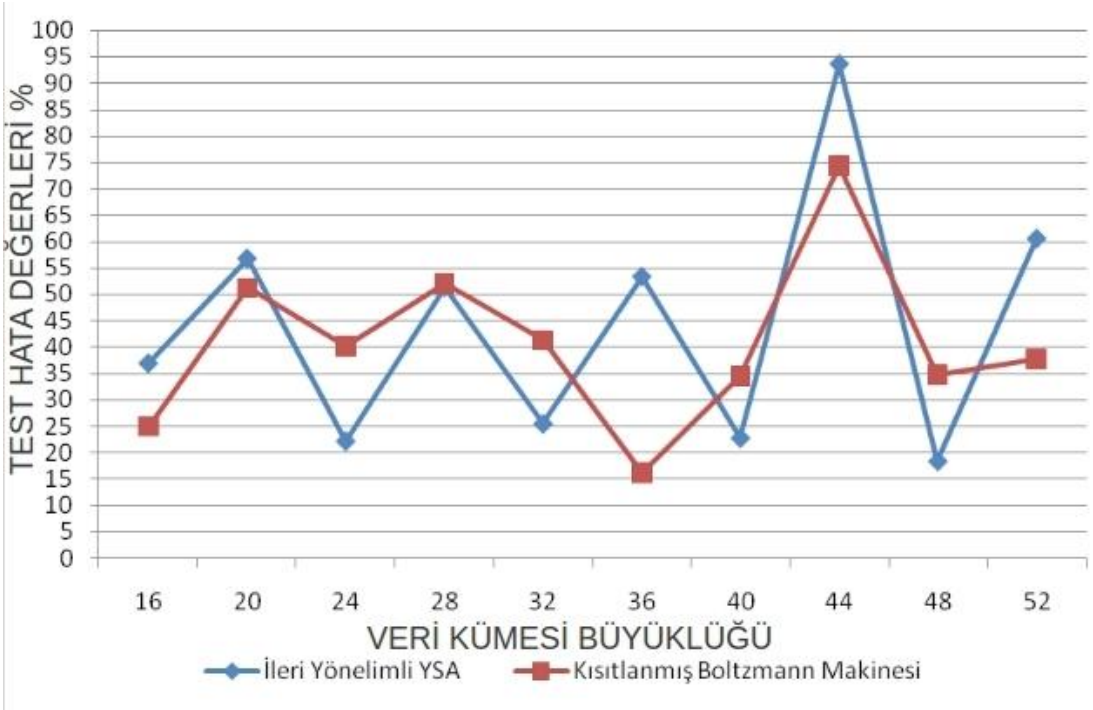
5.2 Zaman Aralığı 4 İçin Tahmin Sonuçlarının İncelemesi

Zaman aralığı değeri 4 için 10 adet farklı uzunlukta örnek veri kümesi ile deneyler yapılmıştır, bu deneylerin ortalama hata değerlerinin grafikleri şekil 5-3 ve 5-4'te, ortalama hata değerleri çizelgesi de çizelge 5-1'de görülmektedir.

Şekil 5-3 ve 5-4'teki grafiğe bakarak söyleyebiliriz ki, zaman aralığı 4 için, veri kümesinin büyüklüğü, yani tahmin yapılmadan önce öğrenilen gözlem verilerinin çokluğunun tahmin sonuçlarına etkisi yoktur. Ortalama olarak standart bilinen İleri Yönelimli YSA ve Kısıtlanmış Boltzmann Makinesi yakın sonuçlar vermektedir. Şekil 5-3'deki grafikte görüldüğü üzere Kısıtlanmış Boltzmann Makinesinde "Net Hata" değeri "Test Hata" değeriyle uyumludur.



Şekil 5-3 Zaman Aralığı 4 İçin Ortalama Hata Değerleri

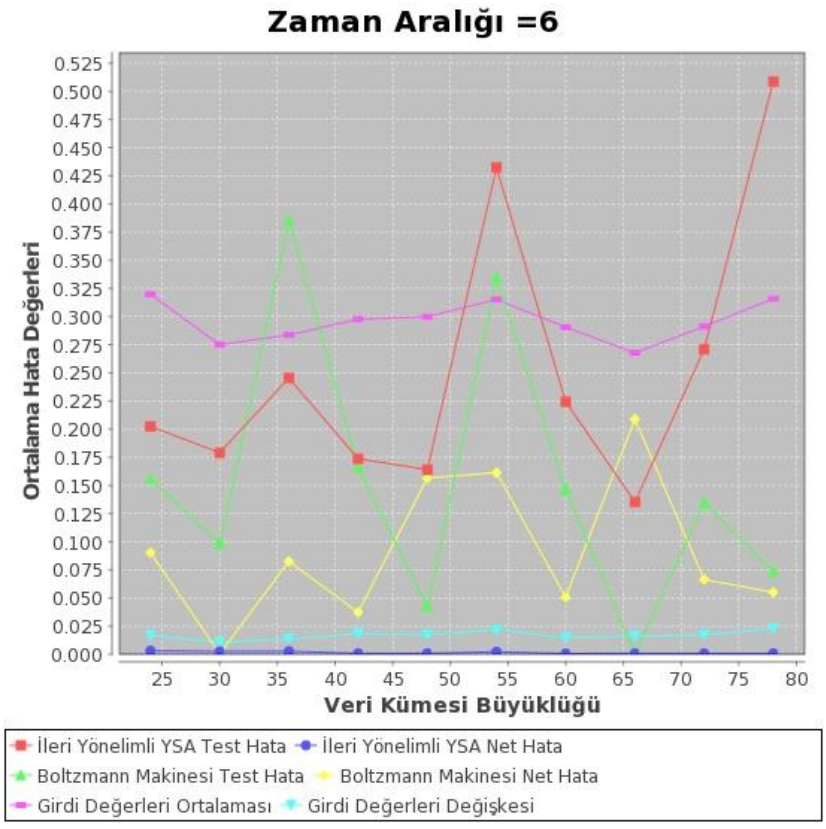


5-4 Zaman Aralığı 4 İçin Ortalama Test Hata Yüzde Değerleri

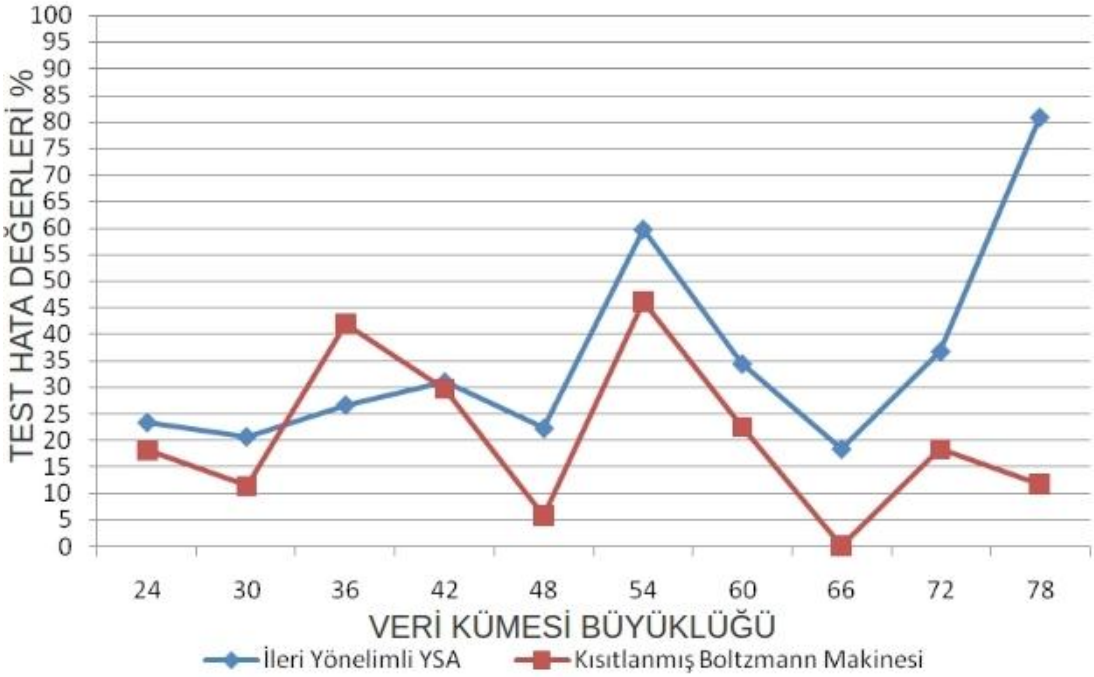
Zaman aralığı 4 için yapılan tahminlerdeki hata değerleri ile zaman aralığı 2 için yapılan tahminlerin hata değerleri iki YSA çeşidi için de çok farklı gözükmemektedir. Hata değerleri standart bir eğri vermese de genel olarak 0,1-0.35 arasında kaldıklarından bir iyileşme veya gerileme olmadığı söylenebilmektedir.

5.3 Zaman Aralığı 6 İçin Tahmin Sonuçlarının İncelemesi

Zaman aralığı 6'da Kısıtlanmış Boltzmann Makinesi , İleri Yönelimli YSA'ya göre daha az hata değerleri üretmektedir. Zaman aralığı 4 ile karşılaştırırsak, Kısıtlanmış Boltzmann Makinesi daha düşük hata değerleri üretmektedir, fakat İleri Yönelimli YSA'nın hata değerleri zaman aralığı 6'da daha yüksektir.



Şekil 5-5 Zaman Aralığı 6 İçin Ortalama Hata Değerleri

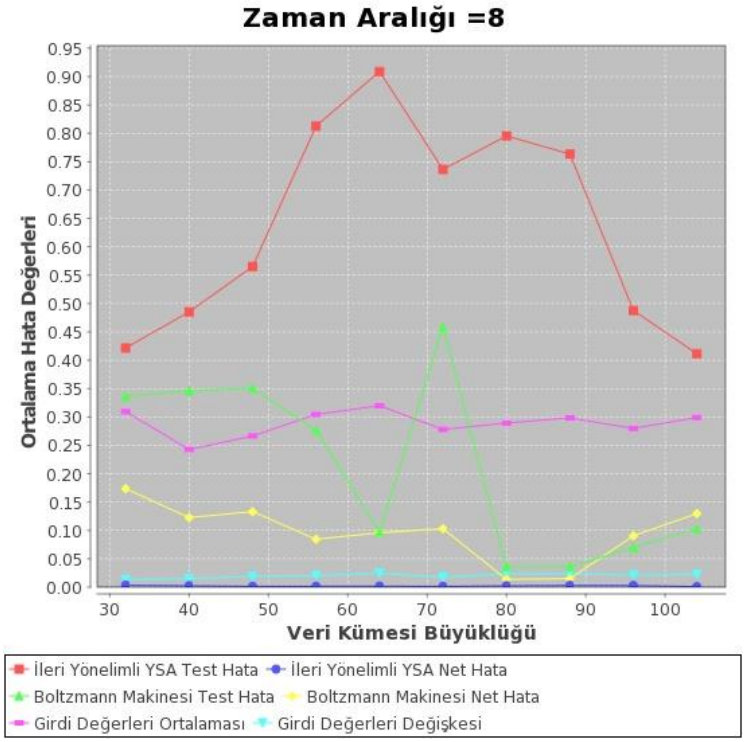


5-6 Zaman Aralığı 6 İçin Ortalama Test Hata Yüzde Değerleri

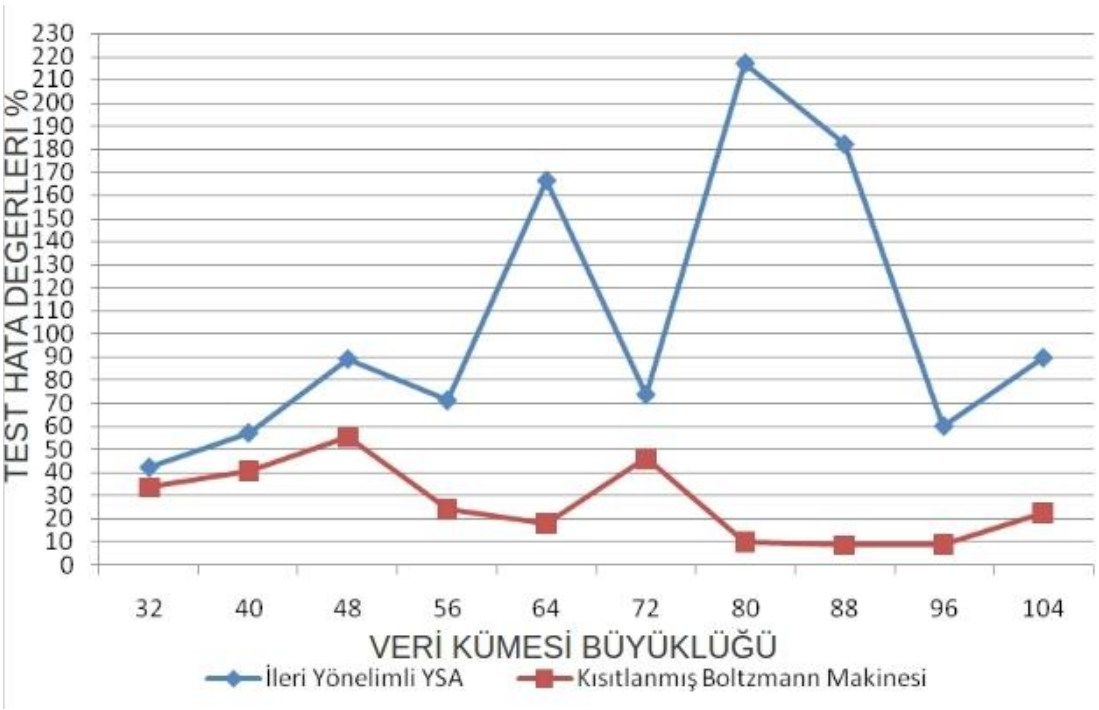
Dikkat çeken bir başka nokta da, Kısıtlanmış Boltzmann Makinesi için "Net Hata Değerleri" ve "Test Hata Değerleri" birbirlerine paralel şekilde artmaktadır veya azalmaktadır. Bu da demek oluyor ki Kısıtlanmış Boltzmann Makinesi ile tahmin yaparken, tahminin güvenirliğini "Net Hata Değerine" bakarak sorgulayabiliriz. Büyük bir "Net Hata Değeri" için, büyük ihtimalle Kısıtlanmış Boltzmann Makinesi kötü bir sonuç üretecektir.

5.4 Zaman Aralığı 8 İçin Tahmin Sonuçlarının İncelemesi

Zaman aralığı 8'de Kısıtlanmış Boltzmann Makinesi , İleri Yönelimli YSA'ya göre çok daha az hata değerleri üretmektedir. "Zaman Aralığı" değeri büyüdükçe Kısıtlanmış Boltzmann Makinesi daha da iyi sonuç vermekte, İleri Yönelimli YSA ise daha kötü sonuç vermektedir.



Şekil 5-7 Zaman Aralığı 8 İçin Ortalama Hata Değerleri

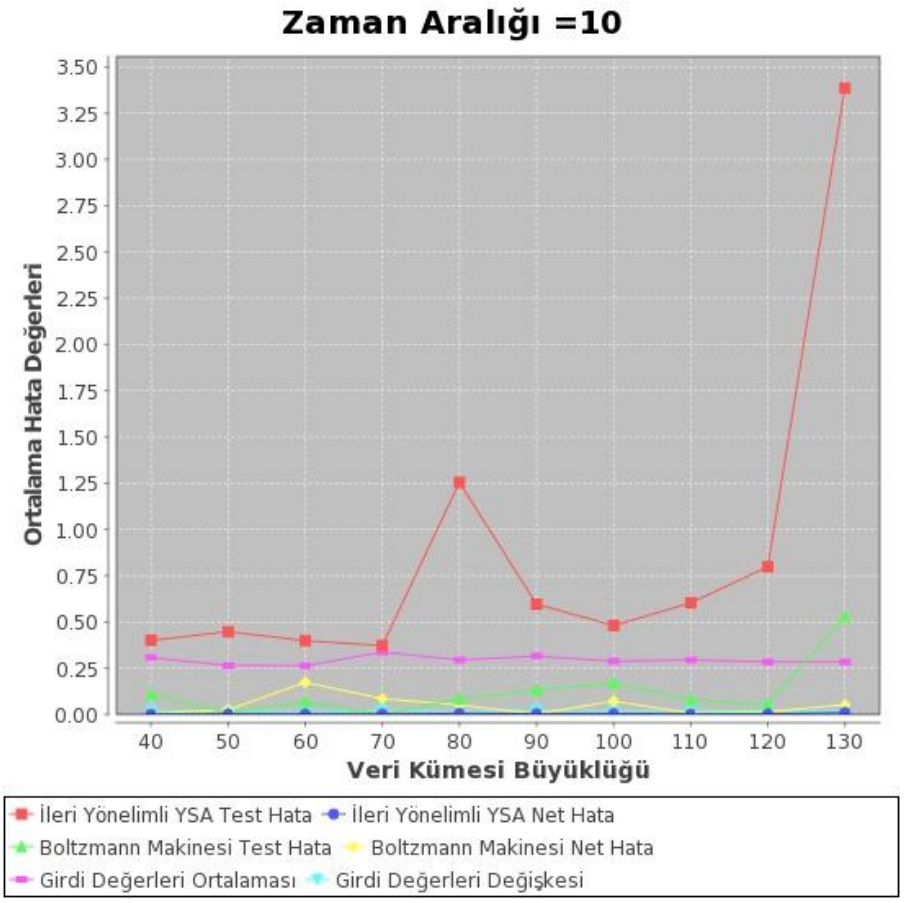


5-8 Zaman Aralığı 8 İçin Ortalama Test Hata Yüzde Değerleri

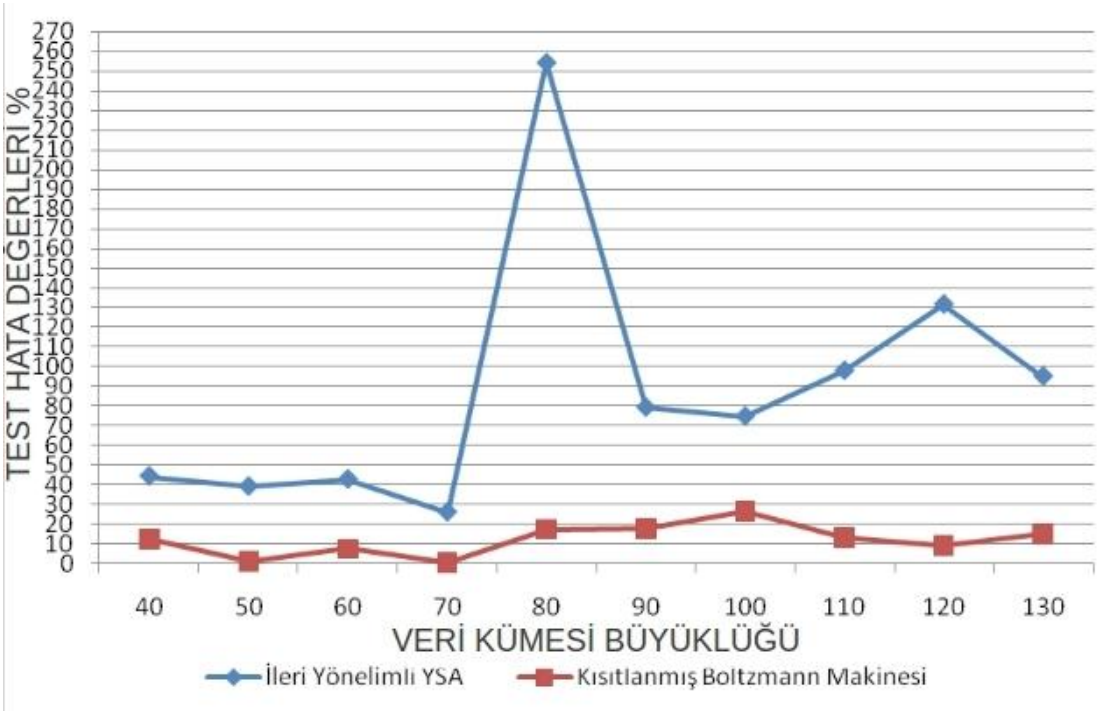
5.5 Zaman Aralığı 10 İçin Tahmin Sonuçlarının İncelemesi

Zaman aralığı 10 için Kısıtlanmış Boltzmann Makinesi daha da iyi sonuçlar üretirken, İleri Yönelimli YSA zaman aralığı 8'e göre daha da kötü sonuçlar üretmektedir. Sonuç olarak Kısıtlanmış Boltzmann Makinesinin zaman aralığı büyüdükçe daha iyi sonuçlar vermekte, İleri Yönelimli YSA daha kötü sonuç vermektedir. Veri kümesi büyüklüğü ise tahminlere hiçbir etki yapmamaktadır.

Eğer bu grafiklerde görülen ortalama hata değerlerinden ziyade, tam isabet etme oranıyla ilgileniyorsak EK-A Tahmin Sonuçları çizelgesindeki kırmızı ile işaretlenmiş tahmin sonuçlarına bakmalıyız.



Şekil 5-9 Zaman Aralığı 10 İçin Ortalama Hata Değerleri



5-10 Zaman Aralığı 10 İçin Ortalama Test Hata Yüzde Değerleri

6 SONUÇ VE ARAŞTIRMALAR

Deneylerden elde edilen sonuçlara göre İleri Yönelimli YSA ile yapılan tahminlerde sonuca en çok 10^{-2} (Test Error) mertebesine kadar yaklaşabilmektedir. Ancak Kısıtlanmış Boltzmann Makinesi 10^{-10} mertebesine kadar yaklaşabilmektedir.

Eğer 10^{-4} 'den küçük hata payını ihmal edilebilir bir hata payı olarak düşünürsek, EK-A Tahmin Sonuçları çizelgesine göre, İleri Yönelimli YSA hiç isabetli tahmin sonucuna ulaşamazken, Kısıtlanmış Boltzmann Makinesi toplamda 67 isabetli sonuca ulaşabilmektedir.

Yapılan deneylerden elde edilen sonuca göre gözlem veri kümesi büyüklüğü tahmin sonuçlarını etkilememektedir. Ancak burada şunu da göz önünde bulundurmalıyız ki, bütün deneylerde YSA'ların eğitilmesi için gerekli olan minimum veri , zaman aralığı değerinin 3 katı, mutlaka YSA'lara sağlanmıştır. Dolayısıyla veri kümesinin bundan sonra artması YSA için öğrenme açısından bir şey değiştirmeyebilir.

Zaman aralığı değeri arttıkça İleri Yönelimli YSA daha kötü tahminler yapmaya başlarken, Kısıtlanmış Boltzmann Makinesi daha iyi tahmin üretebilmektedir. Dolayısıyla uzun bir zaman aralığı tahmini yapılacaksa Kısıtlanmış Boltzmann Makinesi tercih edilmelidir.

Çizelge 5-1'de görüldüğü gibi Kısıtlanmış Boltzmann Makinesinde tahmin süresi , İleri Yönelimli YSA'nın tahmin süresinin 20 katına ulaşabiliyor. Eğer çok hızlı ama daha az keskinlikte bir tahmin isteniyorsa İleri Yönelimli YSA tercih edilmelidir.

Kısıtlanmış Boltzmann Makinesi ile yapılan tahminlerin bir kısmı tam isabetli olurken , bir kısmı olamamaktadır. Dolayısıyla bundan sonraki çalışmalarda aşağıdaki konular araştırılabilir :

1. Yapılan tahminlerin isabetli olma durumları ile Saklı Hücre Sayısı (Number Of Hidden Neurons) arasındaki ilişki.
2. "Test Hata Değerleri" ile "Net Hata Değerleri" arasındaki ilişki.
3. Kısıtlanmış Boltzmann Makinesinin java dilinde yazılmış kodlarını daha hızlı çalıştırmanın yolları (Paralell Processing gibi)

7 KAYNAKLAR

[1] D. H. Ackley, G. E. Hinton, and J. Sejnowski, "A Learning Algorithm for Boltzmann Machines *," *Machine Learning*, vol. 169, pp. 147-169, 1985.

[2] D. Andrzejewski, "Training Binary Restricted Boltzmann Machines with Contrastive Divergence," in *ICML*, 2009, pp. 1-3.

[3] M. Anthony, *Discrete Mathematics of Neural Networks: Selected Topics (Monographs on Discrete Mathematics and Applications)*. Society for Industrial Mathematics, 1987.

[4] M. Anurag and A. Chaturvedi, "Gradient Descent Feed Forward Neural Networks for Forecasting the Trajectories," *International Journal of Computer Applications*, vol. 17, no. 2, pp. 33-35, Mar. 2011.

[5] A. U. Asuncion, Q. Liu, A. T. Ihler, and P. Smyth, "Learning with Blocks : Composite Likelihood and Contrastive Divergence," in *Conference on Uncertainty at University of California Irvine*, 2010, vol. 9.

[6] N. Baccour, H. Kaaniche, M. Chtourou, and M. B. Jemaa, "Recurrent neural network based time series prediction : Particular design problems," in *SSD*, 2007.

[7] Y. Bengio and O. Delalleau, "Justifying and generalizing contrastive divergence.," *Neural computation*, vol. 21, no. 6, pp. 1601-21, Jun. 2009.

[8] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel, *Time Series Analysis Forcasting And Control*. WILEY, 1994.

[9] M. Carreira-Perpinan, "On contrastive divergence learning," *Frontiers in Computational Neuroscience*, 2005.

[10] A. J. Champandard, "Neural Networks Tutorials," 2007. [Online]. Available: http://neuralnetworks.ai-depot.com/Tutorials.html. [Accessed: 2011].

[11] K. Cho, A. Ilin, A. L. I. Lin, and A. Fi, “Enhanced Gradient and Adaptive Learning Rate for Training Restricted Boltzmann Machines,” in *ICML*, 2011.

[12] S. F. Crone, “Forecasting with Artificial Neural Networks,” *Management Science*, 2005. [Online]. Available: http://www.neural-forecasting.com/Downloads/EVIC05_tutorial/EVIC%2705 Slides - Forecasting with Neural Networks Tutorial SFCrone.pdf. [Accessed: 2011].

[13] M. A. Cueto, J. Morton, and B. Sturmfels, “Geometry of the restricted Boltzmann machine,” p. 18, Aug. 2009.

[14] J. A. C. Danilo P. Mandic, *Recurrent neural networks for prediction*. WILEY, 2001.

[15] Y. L. Fablec and J. M. Alliot, “Using Neural Networks to predict aircraft trajectories,” *CENA FR*, 1999. [Online]. Available: http://pom.tls.cena.fr/papers/articles/icai99.pdf. [Accessed: 2011].

[16] L. Fausett, *Fundamentals of Neural Networks: Architectures, Algorithms And Applications*. Prentice Hall, 1993.

[17] P. A. Fishwick and Z. Tang, “Feed-Forward Neural Networks as Models for Time Series Forecasting,” *ORSA Journal on Computing*, 1993.

[18] R. J. Frank, N. Davey, and S. P. Hunt, “Time Series Prediction and Neural Networks,” *Journal of Intelligent & Robotic Systems*, 2001.

[19] J. A. Freeman and D. M. Skapura, *Neural Networks Algorithms, Applications, and Programming Techniques*. Addison-Wesley Pub, 1991.

[20] A. Gagrani, “Image Modeling using Hierarchical Conditional Random Field,” INDIAN INSTITUTE OF THECNOLOGY MADRAS, 2007.

[21] I. A. Gheyas and L. S. Smith, “A Neural Network Approach to Time Series Forecasting,” in *WCE*, 2009, vol. II, no. 1.

[22] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1998.

[23] S. Haykin, *KALMAN FILTERING AND NEURAL NETWORKS*. WILEY, 2001.

[24] G. Hinton, "A Practical Guide to Training Restricted Boltzmann Machines," *Toronto University*, 2010. [Online]. Available: http://www.cs.toronto.edu/~hinton/absps/guideTR.pdf. [Accessed: 2011].

[25] G. E. Hinton, "Training Products of Experts by Minimizing Contrastive Divergence," *Neural Computation*, 2002.

[26] G. Hinton and O. Woodford, "Contrastive Divergence," *Brookes University*, 2006. [Online]. Available: http://cms.brookes.ac.uk/research/visiongroup/talks/woodford/cont_div.ppt. [Accessed: 2011].

[27] J.-hwa Ju, D.-hui Lee, J.-ho Lee, and J.-myung Lee, "Trajectory Estimation of a Moving Object using Kohonen Networks," *ICCAS THAILAND*, 2004.

[28] H. J. Kappen, "Using Boltzmann Machines for probability estimation : A general framework for neural network learning," *CiteSeerX*, 1993.

[29] C. Knerr, "TIME SERIES PREDICTION USING NEURAL NETWORKS," Texas Tech University, 2004.

[30] B. Krose and P. V. D. Smagt,"An Introduction to Neural Networks" , University of Amsterdam , 1996.

[31] F. Kutay ve C. Hamzaçebi, "Yapay Sinir Ağları ile Türkiye Elektrik Enerjisi Tüketiminin 2010 Yılına Kadar Tahmini", *Gazi Uni. Mim. Muh. Fak. Der.*, 2001.

[32] C. Man-chung, W. Chi-cheong, and L. A. M. Chi-chung, "Financial Time Series Forecasting by Neural Network Using Conjugate Gradient Learning

Algorithm and Multiple Linear Regression Weight Initialization," *Economics and Finance*, 2000.

[33] V. Mnih and G. E. Hinton, "Conditional Restricted Boltzmann Machines for Structured Output Prediction," in *Uncertainty In Artificial Intelligence*, 2008.

[34] A. Neural, N. Fall, D. Touretzky, and K. Laskowski, "Neural Networks for Time Series Prediction," *Neural Networks*, 2006. [Online]. Available: http://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15782-f06/slides/timeseries.pdf. [Accessed: 2011].

[35] M. Obitko, "Neural Network Training," 2011. [Online]. Available: http://www.obitko.com/tutorials/neural-network-prediction/neural-network-training.html. [Accessed: 2011].

[36] G. Orr and K.-R. Müller, *Neural networks: tricks of the trade*. Springer, 1998.

[37] D. C. Parkes, "Designing a neural network for forecasting financial time series," 2008. [Online]. Available: http://www.eecs.harvard.edu/~parkes/cs286r/spring08/cours.pdf. [Accessed: 2011].

[38] J. F. B. Paul A. Jensen, "22 . 1 Time Series Models," in *Operations Research Models and Methods*, 2002.

[39] P. Payeur , H. Le-Huy and C. M. Gosselin, "Trajectory prediction for moving objects using artificial neural networks," *IEEE Transactions on Industrial Electronics*, vol. 42, no. 2, pp. 147-158, Apr. 1995.

[40] J. C. Principe, A. Rathie, and J.-ming Kuo, "Prediction of Chaotic Time Series with Neural Networks," *Citeseer*, 1993.

[41] J. R. Rabunal and J. Dorado, *Artificial Neural Networks In Real Life Applications*. IDEA GROUP, 2006.

[42] R. Rojas, *Neural Networks A Systematic Introduction*. Springer, 1996.

[43] R. Salakhutdinov, A. Mnih, and G. Hinton, “Restricted Boltzmann machines for collaborative filtering,” *Proceedings of the 24th international conference on Machine learning - ICML ’07*, pp. 791-798, 2007.

[44] T. J. Sejnowski and G. Hinton, “Learning and Relearning In Boltzman Machines,” in *Graphical models: foundations of neural computation*, MIT Press, 1986.

[45] S. Srihari, “Error Backpropagation,” *Machine Learning*, 2010. [Online]. Available: http://www.cedar.buffalo.edu/~srihari/CSE574/Chap5/Chap5.3-BackProp.pdf. [Accessed: 2011].

[46] G. W. Taylor and G. E. Hinton, “Factored Conditional Restricted Boltzmann Machines for Modeling Motion Style,” in *ICML*, 2009.

[47] B. Walsh, “Markov Chain Monte Carlo and Gibbs Sampling,” *Notes*, 2004. [Online]. Available: http://www.mit.edu/~wingated/introductions/mcmc-gibbs-intro.pdf. [Accessed: 2011].

[48] O. Woodford, “Notes on Contrastive Divergence,” *Oxford University*, 2006. [Online]. Available: http://www.robots.ox.ac.uk/~ojw/files/NotesOnCD.pdf.

[49] M. D. Zeiler, G. W. Taylor, N. F. Troje, and G. E. Hinton, “Modeling pigeon behaviour using a Conditional Restricted Boltzmann Machine,” in *ESANN, 2009*.

[50] G. Zhang, B. E. Patuwo, and M. Y. Hu, “Forecasting with artificial neural networks : The state of the art,” *International Journal of Forecasting*, vol. 14, pp. 35-62, 1998.

8 ÖZGEÇMİŞ

1978 yılında Ankara'da doğdum. Lise eğitimimi Çorlu Mehmet Akif Ersoy Anadolu Lisesinde tamamladım. 1997 – 2003 yılları arasında T.C. Galatasaray Üniversitesi Bilgisayar Mühendisliği bölümünde öğrenim gördüm. 2003 yılında mezun olduktan sonra 32bit Ltd. Şti.'de yazılım mühendisi ve sistem mühendisi olarak çalışmaya başladım. 2008 yılından beri Havelsan'da bilgisayar mühendisi olarak çalışmaktayım.

EK-A TAHMİN SONUÇLARI

Zaman Aralığı	Veri Kümesi Büyüklüğü	Gerçek Veri	İleri Yönelimli YSA 1. Tahmin	İleri Yönelimli YSA 2. Tahmin	İleri Yönelimli YSA 3. Tahmin	İleri Yönelimli YSA 4. Tahmin	Boltzmann Makinesi 1. Tahmin	Boltzmann Makinesi 2. Tahmin	Boltzmann Makinesi 3. Tahmin	Boltzmann Makinesi 4. Tahmin
2	8	0,3650045	0,296549	0,299509	0,3048489	0,3101025	0,4861574	0,3597092	0,3649948	0,2154841
		0,3523776	0,366607	0,335886	0,351639	0,3360667	0,3689794	0,361098	0,1023756	0,361174
	10	0,1210717	0,204561	0,202252	0,2076462	0,2114846	0,1843699	0,18195	0,2460641	0,1594852
		0,1462451	0,171937	0,197039	0,2701689	0,1937634	0,1360515	0,120942	0,1540885	0,1171347
	12	0,1	0,137544	0,137341	0,1364578	0,1371689	0,0961548	0,1	0,1	0,099999
		0,1084448	0,132548	0,129891	0,1307857	0,128569	0,0996338	0,1084448	0,1240698	0,1084447
	14	0,4402835	0,257539	0,271115	0,262625	0,2649045	0,354346	0,4402835	0,2681608	0,2684956
		0,4508193	0,372595	0,212988	0,2230911	0,2195321	0,4461429	0,4508193	0,449957	0,4814212
	16	0,2291646	0,296556	0,293454	0,2965524	0,2871069	0,4713214	0,0510656	0,7212298	0,3601477
		0,151312	0,303919	0,302976	0,3062482	0,2967943	0,1356241	0,2291339	0,2177193	0,1981527
	18	0,2291646	0,293516	0,278689	0,2845835	0,2869681	0,2213523	0,0497231	0,299595	0,1642232
		0,151312	0,299234	0,29263	0,2929496	0,3001219	0,1513119	0,1337416	0,1842157	0,151312
	20	0,3010656	0,304028	0,310715	0,3206436	0,3114595	0,2681886	0,3010661	0,3041198	0,3319229
		0,2291646	0,303607	0,311406	0,3070944	0,3054237	0,4106848	0,2291646	0,4752583	0,3124164
	22	0,2897255	0,31181	0,310295	0,3084597	0,3084157	0,2909448	0,2916634	0,2737326	0,2968817
		0,2531316	0,28657	0,316405	0,3108607	0,4403914	0,2687566	0,4709053	0,3535325	0,2501426
	24	0,5668744	0,351214	0,332138	0,3617654	0,3351295	0,757245	0,5668659	0,1897095	0,1952657
		0,5248115	0,380805	0,298334	0,2661819	0,3755404	0,5872886	0,5892723	0,5638818	0,5209742
	26	0,2897255	0,303011	0,304526	0,3075173	0,3056341	0,8543907	0,293966	0,2897246	0,305317
		0,2531316	0,309168	0,272559	0,3106698	0,3090906	0,3056562	0,259941	0,2531297	0,2521549
4	16	0,4406856	0,286953	0,309151	0,2991658	0,2991039	0,3861736	0,464753	0,4406856	0,492998
		0,5668744	0,305115	0,340044	0,3315165	0,3976321	0,101907	0,4408383	0,5668744	0,4569501
		0,5248115	0,377836	0,389423	0,3964214	0,4097492	0,0174569	0,5668725	0,5248115	0,6908078
		0,4070675	0,440505	0,449262	0,4458734	0,4865609	0,8752636	0,5248725	0,4070675	0,5981395
	20	0,4070675	0,420045	0,494347	0,4396974	0,4421432	0,4070675	0,531228	0,3783688	0,5756617
		0,2682517	0,282658	0,436858	0,2896751	0,4398817	0,2682517	0,2434884	0,2196888	0,5556495
		0,1841259	0,390706	0,439502	0,2838034	0,3761144	0,1841259	0,43062	0,1493845	0,4246369
		0,1673168	0,49031	0,402039	0,296236	0,3088853	0,1673168	0,4021005	0,4333615	0,4243787
	24	0,1429476	0,218436	0,226815	0,2240354	0,226557	0,1559833	0,1336275	0,1512473	0,1054542
		0,2367246	0,213481	0,27608	0,2128494	0,2168588	0,1444114	0,1898507	0,1977144	0,1428557
		0,300181	0,230222	0,275433	0,2392593	0,2433298	0,146693	0,142711	0,143687	0,2276886
		0,3271238	0,257912	0,274318	0,277199	0,2810771	0,2992054	0,3001924	0,3304513	0,0179468
	28	0,4070675	0,501859	0,472272	0,4929605	0,4637635	0,4070675	0,7441215	0,4226774	0,8290793
		0,2682517	0,450288	0,430637	0,4610286	0,4111787	0,2682517	0,5195045	0,26874	0,3783975

		0,1841259	0,362433	0,353935	0,3812107	0,3421691	0,1841259	0,4246497	0,168745	0,4226813
		0,1673168	0,280189	0,278068	0,2963242	0,2745297	0,1673168	0,3075776	0,0423134	0,0556421
	32	0,1673168	0,213742	0,207485	0,2311394	0,2306244	0,1208924	0,1247681	0,2343525	0,2459458
		0,2429979	0,202983	0,283984	0,2236356	0,2258917	0,1678051	0,120591	0,1200717	0,2460488
		0,3944405	0,27165	0,315353	0,278976	0,26363	0,182451	0,1834267	0,1364915	0,1726854
		0,4406856	0,345565	0,35825	0,3274822	0,2994249	0,3944711	0,4539344	0,204842	0,4569176
	36	0,228682	0,312269	0,322208	0,3270628	0,3172159	0,2424774	0,2190323	0,1974241	0,228682
		0,129436	0,176221	0,180646	0,1754533	0,3290069	0,1630003	0,230514	0,1372523	0,129436
		0,1833216	0,178814	0,159545	0,1657137	0,3302425	0,3843709	0,167704	0,181307	0,1833216
		0,4891022	0,219188	0,174843	0,1850739	0,3345683	0,4888884	0,3951214	0,4736013	0,4891022
	40	0,2367246	0,212725	0,215336	0,2193561	0,2201072	0,1547039	0,1741976	0,142948	0,2367246
		0,300181	0,228723	0,23332	0,2342822	0,2335319	0,1742056	0,4867206	0,1742056	0,300181
		0,3271238	0,241168	0,254102	0,2444299	0,2480862	0,0492044	0,0116674	0,3470598	0,3193113
		0,3646024	0,267568	0,283379	0,2679752	0,268855	0,3661595	0,3593524	0,3193151	0,3646024
	44	0,1673168	0,463721	0,346623	0,5080929	0,4339984	0,1673168	0,2416048	0,0422715	0,1514635
		0,1210717	0,223309	0,215482	0,2535966	0,2390559	0,1210717	0,7319001	0,2283414	0,1829346
		0,1462451	0,187528	0,190195	0,1888938	0,1926924	0,1462451	0,0603348	0,0183717	0,1191186
		0,1925706	0,167505	0,18888	0,1603327	0,1675273	0,1925706	0,3963061	0,2081956	0,1463053
	48	0,4432593	0,360862	0,401214	0,3866585	0,3729037	0,4433204	0,3963814	0,1932574	0,3807593
		0,3797225	0,306037	0,262161	0,289753	0,2728916	0,1931972	0,2622904	0,3876566	0,2509373
		0,2463758	0,295764	0,240814	0,269076	0,2571087	0,1838757	0,238548	0,4655988	0,1760633
		0,228682	0,278085	0,227948	0,2493016	0,2421531	0,4786744	0,2328253	0,2328326	0,2286896
	52	0,151312	0,303758	0,358404	0,2772963	0,2889026	0,1353285	0,2138122	0,151312	0,1981948
		0,140374	0,2556	0,222816	0,2476878	0,2411693	0,1354645	0,1561878	0,140374	0,0270718
		0,1429476	0,23273	0,183884	0,2417261	0,2362058	0,1434956	0,3787951	0,1434359	0,1390624
		0,2367246	0,210278	0,17192	0,2380109	0,2366915	0,2367184	0,4871935	0,2054746	0,2366971
6	24	0,3523776	0,251208	0,255664	0,2595329	0,268351	0,2683738	0,0629929	0,3523776	0,3523776
		0,4402835	0,311746	0,320431	0,2667307	0,3332499	0,4773776	0,3133298	0,4402835	0,4402835
		0,4508193	0,342731	0,3542	0,3059569	0,3610366	0,4401617	0,3775697	0,4508193	0,4508193
		0,3006635	0,307305	0,318465	0,3218691	0,3055061	0,3258193	0,4420209	0,3006635	0,3006635
		0,3010656	0,28635	0,29307	0,3371055	0,278884	0,299565	0,2663632	0,3010656	0,3010656
		0,2291646	0,265149	0,266784	0,2971553	0,2503675	0,4729406	0,3036195	0,2291646	0,2291646
	30	0,300181	0,276545	0,292438	0,2804568	0,2861717	0,300181	0,050181	0,300181	0,2992196
		0,3271238	0,285905	0,30635	0,2813352	0,3046057	0,3271238	0,3187925	0,3271238	0,3543802
		0,3646024	0,309892	0,304839	0,271283	0,3222358	0,3646024	0,3646024	0,3646024	0,3349362
		0,4613552	0,34598	0,291446	0,2654858	0,358439	0,4613552	0,4613394	0,4613552	0,3413027
		0,3574445	0,33908	0,262102	0,252042	0,3404288	0,3574445	0,3574445	0,3574445	0,3695275
		0,2897255	0,28109	0,236959	0,2482577	0,2745425	0,2897255	0,2897255	0,2897255	0,3555216
	36	0,1879059	0,181987	0,214555	0,1656782	0,2776821	0,2814286	0,1879059	0,3903464	0,3498418
		0,1479341	0,183259	0,274217	0,1884227	0,2661897	0,2533729	0,1479341	0,1440352	0,4477462
		0,2590027	0,244172	0,330925	0,2350875	0,3349307	0,2001687	0,2590027	0,4291952	0,4406621
		0,3936363	0,331844	0,404869	0,3132725	0,3609874	0,2707369	0,3936363	0,4643379	0,3867078

		0,5462853	0,389293	0,426688	0,3603551	0,3696262	0,2589165	0,5462853	0,31249	0,272935
		0,5240072	0,41114	0,41617	0,3831439	0,3442639	0,1738488	0,5240072	0,7792873	0,1420268
	42	0,151312	0,184923	0,195941	0,1942954	0,2025384	0,151312	0,1825391	0,1356259	0,2448525
		0,140374	0,200461	0,180485	0,2918874	0,2908833	0,140374	0,1354755	0,1559837	0,1404
		0,1429476	0,210128	0,190293	0,3084124	0,3130557	0,1429476	0,1512473	0,1429446	0,1977338
		0,2367246	0,249566	0,233145	0,3320785	0,3421983	0,2367246	0,1429708	0,2367208	0,142959
		0,300181	0,303809	0,284695	0,3499676	0,3610981	0,300181	0,1742046	0,0492044	0,1107441
		0,3271238	0,343065	0,318098	0,3532074	0,3532512	0,3271238	0,3304578	0,2606259	0,2724329
	48	0,4613552	0,393975	0,378443	0,3131801	0,3742011	0,3647245	0,4613552	0,4613552	0,4613552
		0,3574445	0,44769	0,309256	0,2679081	0,4541106	0,4495141	0,3574445	0,3574445	0,3574445
		0,2897255	0,378594	0,255371	0,2254998	0,3836212	0,3574747	0,2897255	0,2897255	0,2897255
		0,2531316	0,296212	0,212231	0,1973731	0,2989532	0,2897255	0,2531316	0,2531316	0,2531316
		0,1879059	0,2318	0,180919	0,199254	0,2331093	0,2530706	0,1879059	0,1879059	0,1879059
		0,1479341	0,187855	0,168365	0,2141038	0,1933517	0,1254054	0,1479341	0,1479341	0,1479341
	54	0,4566905	0,379667	0,341656	0,3844329	0,3637846	0,4606006	0,2106006	0,7106006	0,6950366
		0,3864783	0,167141	0,108219	0,1127237	0,2189917	0,3866147	0,3866223	0,3785654	0,3941917
		0,2619785	0,152535	0,129764	0,1315079	0,1952191	0,3874693	0,3874693	0,3864926	0,3864926
		0,1958681	0,175422	0,210562	0,2016606	0,2131046	0,4458033	0,3833032	0,3835476	0,4495457
		0,1429476	0,258215	0,39233	0,354171	0,2951709	0,1350131	0,1350131	0,1330589	0,213507
		0,2013371	0,405603	0,578469	0,5239508	0,4048217	0,2012989	0,1973946	0,1973946	0,1273265
	60	0,2897255	0,174581	0,176271	0,1801389	0,183349	0,2897255	0,2897255	0,3517379	0,3558883
		0,2531316	0,243154	0,235635	0,1794162	0,2535013	0,2531316	0,2531316	0,2911603	0,2931441
		0,1879059	0,27312	0,260312	0,2088825	0,2768937	0,1879059	0,1879059	0,3126713	0,4379158
		0,1479341	0,316535	0,303124	0,2493512	0,3135779	0,1479341	0,1479341	0,1284042	0,1909042
		0,2590027	0,362646	0,354184	0,2867416	0,3438246	0,2590027	0,2590027	0,0014497	0,3764502
		0,3936363	0,38642	0,365036	0,3119315	0,3581829	0,3936363	0,3936363	0,3781596	0,3762056
	66	0,4613552	0,398428	0,398694	0,392144	0,3795557	0,4613552	0,4613552	0,4613552	0,4613552
		0,3574445	0,410305	0,394063	0,2284997	0,3046587	0,3574445	0,3574445	0,3574445	0,3574445
		0,2897255	0,349269	0,335767	0,2001544	0,2481397	0,2897255	0,2897255	0,2897255	0,2897255
		0,2531316	0,270693	0,267179	0,1895141	0,2012318	0,2531316	0,2531316	0,2531316	0,2531316
		0,1879059	0,219415	0,220847	0,1888089	0,170249	0,1879059	0,1879059	0,1879059	0,1879059
		0,1479341	0,20261	0,203197	0,2038749	0,16603	0,1479341	0,1479341	0,1479341	0,1479341
	72	0,3797225	0,277469	0,180473	0,1895218	0,183574	0,3797225	0,3797225	0,3797225	0,4431988
		0,2463758	0,218213	0,145781	0,1172082	0,1708918	0,2463605	0,2463758	0,2463758	0,1452712
		0,228682	0,209285	0,159975	0,1367024	0,1921252	0,2267222	0,228682	0,228682	0,2305064
		0,129436	0,23054	0,212302	0,183026	0,2484747	0,1294398	0,129436	0,129436	0,1894984
		0,1833216	0,285787	0,31491	0,290378	0,3396445	0,1823451	0,1833216	0,1833216	0,1372399
		0,4891022	0,353383	0,464466	0,4206925	0,4480214	0,2391195	0,4891022	0,4891022	0,2360275
	78	0,2893234	0,274569	0,284876	0,2841799	0,2797211	0,2893234	0,2893234	0,4304388	0,4148727
		0,2812808	0,695596	0,678938	0,6486201	0,2280386	0,2812808	0,2812808	0,2893835	0,2893379
		0,2997788	0,670405	0,594268	0,5911282	0,2207114	0,2997788	0,2997788	0,2820742	0,2977035
		0,2775008	0,552112	0,476725	0,5063817	0,2236381	0,2775008	0,2775008	0,3097848	0,3048447

		0,2182266	0,444909	0,418781	0,413466	0,2102969	0,2182266	0,2182266	0,1800887	0,2797257
		0,1424651	0,350333	0,391858	0,3594602	0,1970427	0,1424651	0,1424651	0,1479141	0,1419767
8	32	0,2897255	0,27319	0,313232	0,2448348	0,2967676	0,2897255	0,2892373	0,2981222	0,4973144
		0,2531316	0,295421	0,291141	0,2794291	0,2900194	0,2531316	0,2599713	0,385185	0,4635239
		0,1879059	0,310164	0,298129	0,316972	0,3125534	0,1879059	1,62E+12	0,350658	0,2887073
		0,1479341	0,345583	0,319347	0,3585433	0,3126262	0,1479341	0,1440278	0,3915265	0,3920084
		0,2590027	0,373492	0,355694	0,386197	0,3416076	0,2590027	0,0011902	0,319536	0,4448181
		0,3936363	0,338796	0,345588	0,3564993	0,3351322	0,3936363	0,3936654	0,445714	0,4662946
		0,5462853	0,286007	0,326247	0,3201528	0,3018357	0,5462853	0,5463463	0,2726642	0,4055099
		0,5240072	0,269993	0,318502	0,2618381	0,3029626	0,5240072	0,5240072	0,25794	0,2910382
	40	0,5332563	0,419886	0,435837	0,6073542	0,3726817	0,5332552	0,5490329	0,6116329	0,611754
		0,4070675	0,258838	0,318452	0,3081715	0,2490012	0,4090206	0,6589905	0,7840563	0,6583249
		0,2976877	0,258145	0,301374	0,2198703	0,2956811	0,2976877	0,0320637	0,2976887	0,4226887
		0,2472605	0,30271	0,359916	0,2688263	0,359117	0,1720652	0,4224541	0,2361671	0,4851908
		0,1462451	0,368085	0,375521	0,3294872	0,4442729	0,1775563	0,4226887	0,1775582	0,1461862
		0,1210717	0,368554	0,36785	0,3767032	0,4652936	0,1190876	0,1249088	0,247991	0,187322
		0,1673168	0,387431	0,343301	0,397365	0,4749463	0,1673168	0,2454432	0,2499431	0,1210717
		0,2429979	0,366123	0,296203	0,3706916	0,3892417	0,2439745	0,1678103	0,2459315	0,2298182
	48	0,2682517	0,233764	0,218364	0,3180637	0,2470174	0,3931335	0,2975485	0,268496	0,3601781
		0,1841259	0,324014	0,283546	0,3697494	0,3345922	0,059246	0,39502	0,1841221	0,0553922
		0,1673168	0,33117	0,379596	0,4376129	0,428703	0,1672557	0,4993022	0,1824535	0,1672564
		0,1210717	0,338329	0,465587	0,4745983	0,4739956	0,1053088	0,1048974	0,0585717	0,244942
		0,1462451	0,265024	0,479284	0,4685492	0,4701013	0,0213662	0,669725	0,1853689	0,0311326
		0,1925706	0,193824	0,462679	0,4010462	0,4277814	0,1925716	0,3925173	0,2089281	0,1924495
		0,2682517	0,187579	0,388645	0,2869679	0,3127807	0,2682517	0,3942302	0,1442302	0,0194705
		0,3523776	0,18521	0,279847	0,1948976	0,2279598	0,3525187	0,3170843	0,3523626	0,3132998
	56	0,4432593	0,266216	0,29633	0,2485867	0,2947719	0,3807574	0,8208821	0,4432593	0,4432593
		0,3797225	0,30449	0,338886	0,3053419	0,2943412	0,3797225	0,4422223	0,3797225	0,3797225
		0,2463758	0,390964	0,434117	0,4576144	0,392316	0,2151105	0,3875045	0,2463758	0,2463758
		0,228682	0,457382	0,471608	0,539233	0,4627912	0,2308726	0,2306285	0,228682	0,228682
		0,129436	0,400984	0,444575	0,4998724	0,4242906	0,1294398	0,1583705	0,129436	0,129436
		0,1833216	0,300929	0,344959	0,3986451	0,3490822	0,182284	0,1265056	0,1833216	0,1833216
		0,4891022	0,227473	0,238045	0,2566058	0,2186683	0,2388754	0,2489935	0,4891022	0,4891022
		0,749442	0,171433	0,164704	0,1537258	0,1751192	0,2499303	0,4895828	0,749442	0,749442
	64	0,2972856	0,156486	0,123815	0,1899661	0,1998716	0,2972857	0,2972856	0,2970119	0,2972856
		0,2724339	0,160059	0,204457	0,206252	0,3561871	0,2997815	0,2724339	0,3036877	0,2724339
		0,1895949	0,342779	0,42768	0,3499754	0,4536511	0,127339	0,1895949	0,439839	0,1895949
		0,1673168	0,549978	0,630109	0,5192891	0,5229692	0,1361891	0,1673168	0,1272777	0,1673168
		0,1269428	0,638502	0,693819	0,5966309	0,4843019	0,128377	0,1269428	0,1259357	0,1269428
		0,1172112	0,624358	0,673163	0,5990635	0,3814941	0,0394448	0,1172112	0,2343987	0,1172112
		0,1286317	0,54345	0,537194	0,5511612	0,2655536	0,128647	0,1286317	0,1364595	0,1286317
		0,1609631	0,389224	0,368424	0,441747	0,2159011	0,1609612	0,1609631	0,157055	0,1609631

	72	0,2897255	0,309269	0,32958	0,3143259	0,3185411	0,289726	0,3517373	0,3517379	0,3517373
		0,2531316	0,397691	0,407753	0,5096353	0,4498255	0,2520973	0,2599103	0,291248	0,2599103
		0,1879059	0,443922	0,466066	0,5388937	0,4786689	0,0629059	0,3126713	0,3751713	0,3126713
		0,1479341	0,415475	0,46246	0,4770334	0,4299174	0,1439515	0,1284038	0,1284042	0,1284038
		0,2590027	0,347452	0,384263	0,3759056	0,3323946	0,2590027	0,2511903	0,2668153	0,2511903
		0,3936363	0,258196	0,287569	0,2334203	0,2274322	0,268631	0,393785	0,3937889	0,393785
		0,5462853	0,173519	0,197242	0,1646787	0,1436439	0,0460411	0,0150962	0,5150962	0,0150962
		0,5240072	0,135209	0,128538	0,149795	0,1102713	0,5234577	0,5244953	0,5270595	0,5244953
	80	0,1424651	0,175013	0,157153	0,1575892	0,1444533	0,1424651	0,1424651	0,1424651	0,1424832
		0,1341007	0,191833	0,163824	0,5644449	0,6826741	0,1341007	0,1341007	0,1341007	0,1497254
		0,1105358	0,212033	0,194776	0,6904464	0,7709068	0,1105358	0,1105358	0,1105358	0,1105361
		0,1	0,246921	0,24778	0,7294084	0,7439488	0,1	0,1	0,1	0,0960785
		0,1058711	0,259322	0,292847	0,6705899	0,6504311	0,1058711	0,1058711	0,1058711	0,1068477
		0,1210717	0,252432	0,273795	0,5739415	0,5521172	0,1210717	0,1210717	0,1210717	0,1210412
		0,151312	0,227414	0,220826	0,4374578	0,4589095	0,151312	0,151312	0,151312	0,2450627
		0,1584699	0,20883	0,189549	0,3635411	0,366255	0,1584699	0,1584699	0,1584699	0,0490796
	88	0,1341007	0,16173	0,157027	0,1661138	0,1637227	0,1341007	0,1341007	0,1496647	0,1498479
		0,1105358	0,664995	0,288279	0,4816274	0,3148549	0,1105358	0,1105358	0,0949108	0,07953
		0,1	0,702773	0,371393	0,5144662	0,35166	0,1	0,1	0,0997554	0,0951019
		0,1058711	0,638557	0,393327	0,5028095	0,360229	0,1058711	0,1058711	0,1078242	0,1058406
		0,1210717	0,546968	0,344597	0,4742531	0,3734529	0,1210717	0,1210717	0,1224764	0,1205844
		0,151312	0,457897	0,267821	0,4484862	0,3252963	0,151312	0,151312	0,2138196	0,1200605
		0,1584699	0,37183	0,196473	0,4324753	0,2816193	0,1584699	0,1584699	0,1276929	0,1663892
		0,248869	0,333966	0,169593	0,3935763	0,276157	0,248869	0,248869	0,186369	0,2449628
	96	0,1075601	0,230394	0,237111	0,2565215	0,2297986	0,1075601	0,1090859	0,1075601	0,1075601
		0,1357897	0,252441	0,272785	0,2606025	0,2411948	0,1357897	0,1357897	0,1357897	0,1357897
		0,16981	0,219035	0,245001	0,2249072	0,1932419	0,16981	0,1717784	0,16981	0,16981
		0,252649	0,181637	0,209704	0,1867129	0,1634968	0,252649	0,0104005	0,252649	0,252649
		0,3086257	0,16169	0,191045	0,156182	0,1564559	0,3086257	0,2930007	0,3086257	0,3086257
		0,3700714	0,150035	0,169383	0,1304181	0,1532683	0,3700714	0,3075673	0,3700714	0,3700714
		0,3818136	0,142155	0,147779	0,1208091	0,1424164	0,3818136	0,2568136	0,3818136	0,3818136
		0,3982206	0,137414	0,126858	0,1351027	0,1351053	0,3982206	0,3972745	0,3982206	0,3982206
	104	0,2775008	0,125849	0,124366	0,1303065	0,1398193	0,277501	0,2775008	0,3089949	0,2775008
		0,2182266	0,26525	0,1216	0,1317485	0,1374647	0,4682266	0,2182266	0,2465793	0,2182266
		0,1424651	0,338703	0,159639	0,1675454	0,1748655	0,142526	0,1424651	0,267465	0,1424651
		0,1341007	0,359623	0,233812	0,2264147	0,2490906	0,1498479	0,1341007	0,1342229	0,1341007
		0,1105358	0,358143	0,30174	0,2771687	0,3079478	0,2355358	0,1105358	0,11078	0,1105358
		0,1	0,318841	0,311632	0,2978426	0,2993272	0,100061	0,1	0,0990846	0,1
		0,1058711	0,258891	0,253137	0,2514742	0,2351641	0,1058711	0,1058711	0,1214619	0,1058711
		0,1210717	0,242171	0,208349	0,2126172	0,1962	0,1205834	0,1210717	0,1205834	0,1210717
10	40	0,1841259	0,413713	0,40902	0,3495761	0,3889592	0,1841259	0,1841717	0,1841259	0,1841259
		0,1673168	0,355483	0,343985	0,2582032	0,2981715	0,1673168	0,4326037	0,1673168	0,1673168

		0,1210717	0,275979	0,263024	0,214289	0,2201206	0,1210717	0,4333919	0,1210717	0,1210717
		0,1462451	0,218537	0,208911	0,2002431	0,2110424	0,1462451	0,2263921	0,1462451	0,1462451
		0,1925706	0,193175	0,183599	0,212029	0,2115453	0,1925706	0,0912827	0,1925706	0,1925706
		0,2682517	0,178527	0,175374	0,2252975	0,227017	0,2682517	0,1736499	0,2682517	0,2682517
		0,3523776	0,189675	0,192146	0,2688506	0,2456144	0,3523776	0,3153899	0,3523776	0,3523776
		0,4402835	0,231693	0,234381	0,3054021	0,2789342	0,4402835	0,4596929	0,4402835	0,4402835
		0,4508193	0,275372	0,283577	0,3296253	0,3120744	0,4508193	0,4507661	0,4508193	0,4508193
		0,3006635	0,315121	0,329424	0,3361174	0,3270304	0,3006635	0,3554445	0,3006635	0,3006635
	50	0,2429979	0,172071	0,124033	0,1669139	0,1138109	0,2429979	0,2429979	0,2429979	0,2429979
		0,3944405	0,277145	0,119403	0,2484959	0,1945092	0,3944405	0,3944405	0,3944405	0,3944405
		0,4406856	0,402746	0,210571	0,346005	0,3302954	0,4406856	0,4406856	0,4406856	0,4406856
		0,5668744	0,525638	0,393213	0,477702	0,5014405	0,5668744	0,5668744	0,5668744	0,5668744
		0,5248115	0,586894	0,562211	0,5699219	0,6338026	0,5248115	0,5248115	0,5248115	0,5248115
		0,4070675	0,556248	0,641105	0,577349	0,6468967	0,4069454	0,4070675	0,4070675	0,4070675
		0,2682517	0,443393	0,599598	0,4924587	0,5146443	0,2681297	0,2682517	0,2682517	0,2682517
		0,1841259	0,278774	0,424857	0,3359793	0,3296725	0,1841259	0,1841259	0,1841259	0,1841259
		0,1673168	0,178943	0,236839	0,2112262	0,1795225	0,1360515	0,1673168	0,1673168	0,1673168
		0,1210717	0,126256	0,116006	0,1283513	0,1027084	0,1054505	0,1210717	0,1210717	0,1210717
	60	0,3523776	0,187592	0,196622	0,1879981	0,1890078	0,3523776	0,3523776	0,3542089	0,3523776
		0,4402835	0,325436	0,331525	0,3386502	0,3589257	0,4402835	0,4402835	0,440253	0,4402835
		0,4508193	0,390095	0,404219	0,4294316	0,4513758	0,4508193	0,4508193	0,4510028	0,4508193
		0,3006635	0,440483	0,407404	0,4467207	0,4650046	0,3006635	0,3006635	0,3162984	0,3006635
		0,3010656	0,42117	0,426856	0,4584057	0,4654286	0,3010656	0,3010656	0,3015196	0,3010656
		0,2291646	0,418581	0,390362	0,4108097	0,4148808	0,2291646	0,2291646	0,49894	0,2291646
		0,151312	0,353661	0,307056	0,3178015	0,3208731	0,151312	0,151312	0,1825317	0,151312
		0,140374	0,280293	0,24956	0,2391401	0,237254	0,140374	0,140374	0,1393841	0,140374
		0,1429476	0,221576	0,205425	0,1980133	0,1844411	0,1429476	0,1429476	0,1435246	0,1429476
		0,2367246	0,202544	0,213131	0,1868005	0,1916427	0,2367246	0,2367246	0,2367209	0,2367246
	70	0,1429476	0,1948	0,22334	0,2161786	0,2182301	0,1429476	0,1429476	0,1429476	0,1429476
		0,2013371	0,22526	0,225923	0,2312433	0,2289735	0,2013371	0,2013371	0,2013371	0,2013371
		0,4487283	0,327172	0,271396	0,2954001	0,2872375	0,4487283	0,4487283	0,4487283	0,4487283
		0,6552629	0,471675	0,339753	0,385507	0,3934337	0,6552629	0,6552629	0,6552629	0,6552629
		0,6505982	0,536295	0,442831	0,4816894	0,499389	0,6505982	0,6505982	0,6505982	0,6505982
		0,596793	0,510507	0,53569	0,5675454	0,5657995	0,596793	0,596793	0,596793	0,596793
		0,4781643	0,470401	0,580285	0,5489267	0,5302642	0,4781643	0,4781643	0,4781643	0,4781643
		0,3801247	0,397316	0,525649	0,4589693	0,4577818	0,3801247	0,3801247	0,3801247	0,3801247
		0,3523776	0,341509	0,41232	0,3652027	0,3727945	0,3523776	0,3523776	0,3523776	0,3523776
		0,2972856	0,32181	0,342467	0,3139246	0,3167814	0,2972856	0,2972856	0,2972856	0,2972856
	80	0,2182266	0,238492	0,241614	0,2212456	0,1921174	0,2182266	0,2023575	0,2794801	0,2182266
		0,1424651	0,757106	0,672367	0,7213966	0,7679834	0,1424651	0,142465	0,1419767	0,1424651
		0,1341007	0,766166	0,658439	0,7329582	0,7789509	0,1342229	0,1342228	0,1497869	0,1341007
		0,1105358	0,650484	0,553896	0,6869241	0,7056216	0,1105358	0,1105359	0,2123444	0,1105358

		0,1	0,476997	0,445421	0,5737522	0,5786346	0,1009766	0,0960785	0,1107799	0,1
		0,1058711	0,379275	0,40269	0,5008541	0,4055325	0,105871	0,0355281	0,0980241	0,1058711
		0,1210717	0,34477	0,379987	0,4455834	0,3270794	0,1210717	0,121064	0,1196002	0,1210717
		0,151312	0,350377	0,396786	0,3933887	0,3448332	0,026312	0,1513119	0,1200906	0,151312
		0,1584699	0,435388	0,434248	0,4368871	0,4443949	0,1584699	0,1584775	0,1669537	0,1584699
		0,248869	0,579799	0,544721	0,5366795	0,6164855	0,248869	0,2459317	0,1824556	0,248869
	90	0,3801247	0,367837	0,332955	0,3522664	0,368985	0,3801247	0,4699534	0,3800636	0,4777504
		0,3523776	0,504159	0,191136	0,2047016	0,3216504	0,3523776	0,4469905	0,3523776	0,4380795
		0,2972856	0,639149	0,307134	0,367258	0,5509546	0,2972856	0,3051134	0,2972856	0,2894656
		0,2724339	0,625016	0,418086	0,4402367	0,6465611	0,2724339	0,2919528	0,3017307	0,2665468
		0,1895949	0,561935	0,427582	0,4360223	0,5899738	0,1895949	0,2836442	0,1896254	0,3305383
		0,1673168	0,480802	0,349855	0,368965	0,4337684	0,1673168	0,1434837	0,1358226	0,1273388
		0,1269428	0,398371	0,266022	0,291692	0,2902693	0,1269428	0,1283762	0,1269123	0,2279856
		0,1172112	0,335079	0,218995	0,2388201	0,1668487	0,1172112	0,2269176	0,0859612	0,1679576
		0,1286317	0,314477	0,130781	0,1743289	0,1034058	0,1286317	0,1596529	0,1911317	0,024071
		0,1609631	0,27683	0,05517	0,1115609	0,0697721	0,1609631	0,1646201	0,1570492	0,1568338
	100	0,1058711	0,199873	0,186501	0,1936797	0,1975256	0,1058406	0,1058406	0,1058368	0,0980279
		0,1210717	0,470323	0,443437	0,3669982	0,3747155	0,1205844	0,1210717	0,1205844	0,1205834
		0,151312	0,454529	0,451976	0,35561	0,3774962	0,1200605	0,182562	0,1200605	0,1200605
		0,1584699	0,366703	0,398337	0,292076	0,3265703	0,1663892	0,158592	0,1585767	0,1664655
		0,248869	0,287597	0,329033	0,2454474	0,2654439	0,2449551	0,248869	0,248869	0,2449556
		0,2926209	0,22263	0,256348	0,2208102	0,2287884	0,1668923	0,4168923	0,0418923	0,1668913
		0,2729165	0,19857	0,222194	0,2183907	0,2317778	0,3046562	0,3041684	0,3124692	0,3046562
		0,2265909	0,208419	0,210644	0,2371293	0,2544769	0,2285438	0,2265909	0,1660438	0,1670204
		0,2005328	0,235853	0,222899	0,2687209	0,2673035	0,2005176	0,2005176	0,2005214	0,2317753
		0,1656278	0,318297	0,297695	0,3258269	0,3059296	0,1656278	0,1656278	0,1656278	0,1651395
	110	0,2812808	0,178329	0,165265	0,1750635	0,1600146	0,2893077	0,2812808	0,28125	0,2812808
		0,2997788	0,457264	0,430043	0,5254179	0,4164707	0,2820785	0,2997788	0,2997788	0,2997788
		0,2775008	0,534401	0,510304	0,575081	0,5106095	0,2989854	0,2775008	0,2775008	0,2775008
		0,2182266	0,531419	0,520526	0,5475271	0,5233093	0,404699	0,2182266	0,2182285	0,2182266
		0,1424651	0,442265	0,448531	0,4432066	0,4485376	0,147838	0,1424651	0,142465	0,1424651
		0,1341007	0,320387	0,339978	0,325009	0,3365275	0,1498499	0,1341007	0,1342228	0,1341007
		0,1105358	0,223917	0,24533	0,2230621	0,2356065	0,1419117	0,1105358	0,2355358	0,1105358
		0,1	0,167373	0,178641	0,1637939	0,1645569	0,11078	0,1	0,115625	0,1
		0,1058711	0,171166	0,171561	0,1705401	0,1581654	0,107805	0,1058711	0,1058711	0,1058711
		0,1210717	0,210955	0,20256	0,221591	0,1912974	0,1058711	0,1210717	0,1210688	0,1210717
	120	0,1105358	0,223512	0,238995	0,2183955	0,2152824	0,1654691	0,1105358	0,1105358	0,1105358
		0,1	0,595327	0,671724	0,4815973	0,4990848	0,1095745	0,1	0,1	0,1
		0,1058711	0,593494	0,645046	0,4681956	0,4982578	0,1	0,1058711	0,1058711	0,1058711
		0,1210717	0,49601	0,545424	0,418569	0,4518861	0,1054439	0,1210717	0,1210717	0,1210717
		0,151312	0,378205	0,407119	0,3539204	0,3695159	0,1210715	0,151312	0,151312	0,151312
		0,1584699	0,283616	0,289456	0,2955697	0,3010074	0,1825624	0,1584699	0,1584699	0,1584699

		0,248869	0,218342	0,221106	0,2508731	0,2504697	0,1623757	0,248869	0,248869	0,248869
		0,2926209	0,18456	0,189107	0,2017986	0,2171673	0,1833059	0,2926209	0,2926209	0,2926209
		0,2729165	0,199192	0,211933	0,2107073	0,2232314	0,2848756	0,2729165	0,2729165	0,2729165
		0,2265909	0,27369	0,302056	0,2675859	0,275782	0,3744182	0,2265909	0,2265909	0,2265909
	130	0,1075601	0,220016	0,206364	0,2239141	0,2341586	0,1075601	0,1079873	0,1075601	0,1079892
		0,1357897	0,645969	0,60971	0,6437575	0,6647723	0,1357897	0,1357897	0,1357897	0,1982897
		0,16981	0,637533	0,620178	0,6403874	0,6771141	0,16981	0,1715247	0,16981	0,1717785
		0,252649	0,578282	0,568167	0,5788474	0,592427	0,252649	0,0104615	0,252649	0,1285645
		0,3086257	0,46047	0,462945	0,4634972	0,4330084	0,3086257	0,2930007	0,3086257	0,2929931
		0,3700714	0,306136	0,320373	0,3177824	0,2928947	0,3700714	0,3075673	0,3700714	0,3073213
		0,3818136	0,212629	0,22334	0,2236812	0,2125819	0,3818136	0,2568146	0,3818136	0,3818147
		0,3982206	0,179655	0,177289	0,1829432	0,1971133	0,3982206	0,3972288	0,3982206	0,3972898
		0,3010656	0,220754	0,193871	0,2085149	0,2358329	0,3010656	0,3035075	0,3010656	0,3972575
		0,215653	0,327391	0,293834	0,3181832	0,3408205	0,215653	0,4031606	0,215653	0,1530996