GÖRÜNTÜ SINIFLANDIRMA İÇİN DERİN ÖĞRENME İLE BAYESÇİ DERİN ÖĞRENME YÖNTEMLERİNİN KARŞILAŞTIRILMASI

YÜKSEK LİSANS TEZİ

Barış AKPINAR

Danışman
Doç. Dr. Engin TAŞ

İSTATİSTİK ANABİLİM DALI

Eylül 2019

AFYON KOCATEPE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

GÖRÜNTÜ SINIFLANDIRMA İÇİN DERİN ÖĞRENME İLE BAYESÇİ DERİN ÖĞRENME YÖNTEMLERİNİN KARŞILAŞTIRILMASI

Barış AKPINAR

Danışman
Doç. Dr. Engin TAŞ

İSTATİSTİK ANABİLİM DALI

Eylül 2019

TEZ ONAY SAYFASI

Barış AKPINAR tarafından hazırlanan "Görüntü Sınıflandırma için Derin Öğrenme ile Bayesçi Derin Öğrenme Yöntemlerinin Karşılaştırılması" adlı tez çalışması lisansüstü eğitim ve öğretim yönetmeliğinin ilgili maddeleri uyarınca 12/09/2019 tarihinde aşağıdaki jüri tarafından oy birliği ile Afyon Kocatepe Üniversitesi Fen Bilimleri Enstitüsü **İstatistik Anabilim Dalı'nda YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Danışman : Doç. Dr. Engin TAŞ

İmza

Başkan : Dr. Öğr. Üyesi Tarık YILMAZ
Aksaray Üniversitesi,
İktisadi ve İdari Bilimler Fakültesi
.....

Üye : Doç. Dr. Engin TAŞ
Afyon Kocatepe Üniversitesi,
Fen Edebiyat Fakültesi
.....

Üye : Dr. Öğr. Üyesi Ayça Hatice ATLI
Afyon Kocatepe Üniversitesi,
Fen Edebiyat Fakültesi
.....

Afyon Kocatepe Üniversitesi
Fen Bilimleri Enstitüsü Yönetim Kurulu'nun
...../...../..... tarih ve
..... sayılı kararıyla onaylanmıştır.

.....
Prof. Dr. İbrahim EROL
Enstitü Müdürü

BİLİMSEL ETİK BİLDİRİM SAYFASI
Afyon Kocatepe Üniversitesi

Fen Bilimleri Enstitüsü, tez yazım kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içindeki bütün bilgi ve belgeleri akademik kurallar çerçevesinde elde ettiğimi,
- Görsel, işitsel ve yazılı tüm bilgi ve sonuçları bilimsel ahlak kurallarına uygun olarak sunduğumu,
- Başkalarının eserlerinden yararlanılması durumunda ilgili eserlere bilimsel normlara uygun olarak atıfta bulunduğumu,
- Atıfta bulunduğum eserlerin tümünü kaynak olarak gösterdiğimi,
- Kullanılan verilerde herhangi bir tahrifat yapmadığımı,
- Ve bu tezin herhangi bir bölümünü bu üniversite veya başka bir üniversitede başka bir tez çalışması olarak sunmadığımı

beyan ederim.

12/09/2019

Barış AKPINAR

ÖZET
Yüksek Lisans Tezi

GÖRÜNTÜ SINIFLANDIRMA İÇİN DERİN ÖĞRENME İLE BAYESÇİ DERİN ÖĞRENME YÖNTEMLERİNİN KARŞILAŞTIRILMASI

Barış AKPINAR

Afyon Kocatepe Üniversitesi

Fen Bilimleri Enstitüsü

İstatistik Anabilim Dalı

Danışman: Doç. Dr. Engin TAŞ

Temel evrişimli derin öğrenmede ağ mimarisi oluşturulurken iç katman sayısını belirleyen ağın derinliğinin ve ağın eğitimi öncesinde öğrenme oranı, momentum ve L2 düzeltmesi gibi öğrenme parametrelerinin başlangıç değerlerinin belirlenmesi gerekir. Bu da çözülmesi gereken ayrı bir optimizasyon problemidir. Bayesçi derin öğrenme ağ mimarisini ve öğrenme parametrelerinin uygun başlangıç değerlerini bulmak için Bayesçi optimizasyon tekniklerini kullanır. Bu tez çalışmasında, temel evrişimli derin öğrenme ile Bayesçi derin öğrenme popüler bir görüntü sınıflandırma problemi üzerinde karşılaştırılmıştır. Her iki yöntemin birbirine göre performansları, avantajları ve dezavantajları ilgili sınıflandırma problemi üzerinde değerlendirilmiştir. Bayesçi derin öğrenme, test veri kümesi üzerindeki sınıflandırma performansını önemli bir derecede arttırmasına rağmen ağın yapısının ve öğrenme parametrelerinin başlangıç değerlerinin optimizasyonu ek bir zaman maliyeti oluşturmuştur.

2019, ix + 74 sayfa

Anahtar Kelimeler: Derin Öğrenme, Yapay Sinir Ağları, Boltzman Makinesi, Kısıtlı Boltzman Makinesi, Gauss-Bernoulli KBM' leri, Evrişimli Sinir Ağları, Bayesçi Derin Öğrenme

ABSTRACT
M.Sc. Thesis

COMPARISON OF DEEP LEARNING AND BAYESIAN DEEP LEARNING METHODS FOR IMAGE CLASSIFICATION

Barış AKPINAR

Afyon Kocatepe University

Graduate School of Natural and Applied Sciences

Department of Statistic

Supervisor: Assoc. Prof. Engin TAŞ

When constructing network architecture in basic convolutional deep learning, it is necessary to determine the depth of the network which specifies the number of inner layers, and the initial values of learning parameters such as learning rate, momentum and L2 regularization before training of the network. This is also a separate optimization problem that needs to be solved. Bayesian deep learning uses Bayesian optimization techniques to find the network architecture and appropriate initial values of the learning parameters. In this thesis, basic convolutional deep learning and Bayesian deep learning are compared on a popular image classification problem. The performance, advantages and disadvantages of both methods are evaluated on the related classification problem. Although Bayesian deep learning significantly increased the classification performance on the test dataset, Bayesian optimization of the network structure and initial values of the learning parameters introduced an additional time cost.

2019, ix + 74 pages

Keywords: Deep Learning, Artificial Neural Network, Boltzmann Machine, Restricted Boltzmann Machine, Gauss-Bernoulli RBM's, Convolutional Neural Networks, Bayesian Deep Learning

TEŞEKKÜR

Bu araştırmanın konusu, deneysel çalışmaların yönlendirilmesi, sonuçların değerlendirilmesi ve yazımı aşamasında yapmış olduğu büyük katkılarından dolayı tez danışmanım Sayın Doç. Dr. Engin TAŞ'a, her konuda öneri ve eleştirileriyle yardımlarını gördüğüm hocalarıma ve arkadaşlarıma teşekkür ederim. Bu araştırma boyunca maddi ve manevi desteklerinden dolayı aileme teşekkür ederim.

Barış AKPINAR

AFYONKARAHİSAR, 2019

İÇİNDEKİLER DİZİNİ

Sayfa

SİMGELER DİZİNİ

Simgeler

w	Ağırlık
b	Sapma
α	Öğrenme Oranı
θ	Model Parametresi
δ	Hata Terimi
Δ	Ağırlık Güncelleme Terimi
v	Görünür Birim
h	Gizli Birim
E	Enerji Fonksiyonu
σ	Standart Sapma
l	Katman
σ^2	Varyans

ŞEKİLLER DİZİNİ

Sayfa

ÇİZELGELER DİZİNİ

Sayfa

TABLOLAR DİZİNİ

Sayfa

1. GİRİŞ

Yapay zeka, insanda var olan düşüncenin ya da özelliklerin kodlarla ilgili makinaya aktarılması sonucunda makinanın insana özgü bu deneyimleri öğrenmesini, analiz etmesini ve bunları geliştirmesini sağlamakla birlikte yeni girdilere uyum sağlaması sayesinde istenildiği durumda kendi kendine öğrenebilmesi ve bunun sonucunda insan benzeri görevleri gerçekleştirmesini mümkün kılar. Bu teknoloji, büyük verilerin makinalar tarafından işlenmesi ve sonucunda bu verileri tanıyarak belirli görevleri gerçekleştirmek için eğitimin yapılması işlemidir. Yapay zeka artık günümüzde oldukça popüler olan gündelik hayatta cep telefonu, ev aletleri, otonom cihazlar (örneğin sürücüsüz araçlar, pilotsuz uçaklar, kaptansız gemiler gibi) bir çok alanda yaygın olarak kullanılmakta ve daha da geliştirilerek kullanım alanları artmaktadır. Makine öğrenmesi, matematiksel ve istatiksel yöntemler kullanarak veri kümelerinden çıkarımlar yapıp bunlara dair tahminlerle ilgilenen yapay zekanın bir alt kümesidir.

Derin öğrenme ise makine öğreniminin bir alt koludur ve derin öğrenme algoritmalarının temelini Yapay Sinir Ağı (YSA) oluşturmaktadır. Derin öğrenme mimarileri çok katmanlıdır ve birden fazla parametre içerir. Yapay zeka onun alt kümesi makine öğrenmesi ve onun da alt kolu olarak ifade edilen derin öğrenmenin kendilerine özgü terminolojileri vardır.

Şekil 1.1 Yapay zeka, makine öğrenmesi ve derin öğrenme (İnt.Kyn.1).

Görüntü sınıflandırması, görüntünün girdi olarak alınmasının ardından bir sınıfın (araba, kamyon, uçak vb.) ya da görüntü için onu en iyi tanımlayan sınıfların olasılıklarının çıkartılması işlevi anlamına gelmektedir. Bu durum canlılarda özellikle de insanlar için, doğumdan itibaren başlayıp öğrenilen ilk becerilerindendir. İnsan çok uzunca bir süre düşünmeden tek seferde bulunulan alanı, alandaki canlı ya da cansız varlıkları veya cisimleri problemsiz ve hızlı bir şekilde tanımlayabilir. Yani bir görüntü görüldüğünde ya da alana bakıldığında bilinçli bir şekilde fakat farkında dahi olmadan çoğu zaman görüntünün ayırıcı özelliğini ortaya çıkarabilir ve her objeyi etiketleyerek tanımlayabilir. Bilgisayar da aynı şekilde görüntüyü girdi olarak alır ve görüntünün çözünürlüğüne, boyutuna bağlı olarak bir dizi piksel değerleri görür. Bu piksel değerleri 0'dan 255'e kadar o noktadaki piksel yoğunluğunu temsil etmektedir. Bu değerler, görüntü sınıflandırmasında bilgisayar için girdilerdir. Bilgisayarın bu sayı dizisini alıp görüntünün belli bir sınıf, örneğin araba için 0.45, uçak için 0.80, gemi için 0.30 olması olabilirliğini ifade eden olasılıklar bilgisayarın çıktılarıdır ve böylelikle bu sayılar bir anlam ifade eder. Artık buradaki amaç tüm görüntüleri ayırt edebilmek ve görüntülere ait eşsiz özellikleri bulmaktır. İnsan için bilinçaltında da faaliyet gösteren aşamalar aslında budur. Bir uçağın görüntüsüne bakıldığında, görüntünün kanatları veya pervanesi gibi belirlenebilir özellikleri varsa, bu bir uçak olarak sınıflandırılabilir. Aynı şekilde bilgisayar da kenar ve eğriler gibi alt düzeyde özellikleri arayıp daha sonra bir dizi evrişim katmanı yoluyla görüntü sınıflandırmasını gerçekleştirebilmektedir. Bu bir Evrişimli Sinir Ağı (ESA)'nın genel bir yapısıdır.

Bu tez çalışmasında, temel evrişimli derin öğrenme ile Bayesçi derin öğrenme bir görüntü sınıflandırma problemi üzerinde karşılaştırılmıştır. Temel evrişimli derin öğrenmede ağ mimarisindeki iç katmanların sayısını belirleyen ağ derinliği ve öğrenme oranı, momentum, L2 düzeltmesi gibi öğrenme parametrelerinin başlangıç değerlerinin belirlenmesi gerekir. Bu da aslında ayrı bir optimizasyon problemidir. Bu parametrelerin optimal değerlerinin belirlenmesi için son yıllarda önerilen Bayesçi derin öğrenme yöntemleri incelenmiştir ve örnek bir görüntü sınıflandırma problemi üzerinde temel evrişimli derin öğrenme ile Bayesçi derin öğrenme yöntemleri karşılaştırılmıştır.

2. LİTERATÜR BİLGİLERİ

Yapay Sinir Ağlarının (YSA) hesaplama modellerinin temelleri McCulloch and Pitts (1943) tarafından yapılan ‘‘Sinir Sisteminin İçinde Olan Fikirlerin Mantıksal Hesabı’’ başlıklı makaleye dayanmaktadır ve günümüze kadar birçok teorik araştırmanın temelini oluşturmaktadır. Bu çalışmada, genel olarak insan beynindeki merkezi sinir sisteminden esinlenerek basit bir sinir ağı tasarlanmıştır. Öğrenme üzerine çalışmaların yoğunlaştığı 1949'lu yıllarda Hebb (1949), insan öğrenme sürecini modellemeye çalışmıştır ve YSA'da öğrenme için başlangıç noktası sayılabilecek bir kural oluşturmuştur. Hebb kuralına göre aynı anda ateşlenen iki sinir hücresi arasındaki bağlantı ayrı ayrı ateşlenen iki sinir hücresi arasındaki bağlantıya göre daha güçlüdür. Bu kural o dönemde bir sinir ağının öğrenme işini nasıl gerçekleştirdiği konusunda fikir vermekle birlikte bugün hala geçerli olan öğrenme kurallarından birçoğunun temelini oluşturmaktadır.

Rosenblatt (1958)'de Tek Katmanlı Doğrusal Algılayıcı (TKDA) ağ modelini geliştirmiştir. Bu model, bugünkü makine öğrenme algoritmalarının temelini oluşturmaktadır. Sadece doğrusal olarak ayrılabilen sınıflandırma problemlerine çözüm üretebilirken doğrusal olarak ayrılamayan sınıflandırma problemlerine çözüm üretememektedir. Widrow and Hoff (1960) tarafından yeni bir yaklaşım olarak Tek Katmanlı Yinelemeli Doğrusal Sinir Ağı (TKYDSA) geliştirilmiştir. Bu TKDA'ya benzemektedir ancak öğrenme kuralı olarak farklıdır. Bununla birlikte yeni ve güçlü bir öğrenme kuralı olarak Widrow and Hoff (1960) tarafından kendi isimlerinde Widrow Hoff öğrenme kuralı oluşturulmuştur. Bu kuralın en önemli özelliği eğitim boyunca toplam hatayı en aza indirmeyi hedeflemesidir. 1969 yılına gelindiğinde Minsky and Papert (1969), TKDA'nın XOR fonksiyonunu öğrenemeyeceğini matematiksel olarak ispat etmişlerdir. TKDA'nın yetersiz kaldığının gösterilmesi YSA üzerindeki çalışmaları bir süre aksatmıştır. 70'li yıllardan 80'li yıllara kadar YSA'nın gelişimi açısından durgunluk dönemleridir. Bu tarihler arasında bazı bilim adamları kendi kişisel gayretleriyle çalışmalarını devam ettirmeye çabalasalar da önemli bir ilerleme sağlanamamıştır.

80'li yılların başında Hopfield (1982) doğrusal olmayan sınıflandırma problemlerini çözebilecek ağlar üzerinde çalışmaya başlamıştır ve o dönemdeki çözülmesi zor olan

problemlere çözüm getirilebilmiştir. Hopfield'ın çalışması aynı zamanda ilerde Hinton ve arkadaşları tarafından geliştirilecek Boltzmann Makinasının da (BM) temelini oluşturmuştur. Kohonen (1982) tarafından yapılan çalışmalar sonucunda denetimsiz ağların geliştirilmesiyle YSA çalışmalarına ilgi tekrardan artmaya başlamıştır. Rumelhart vd. (1988) Çok Katmanlı Algılayıcı (ÇKA) için geri yayılım olarak adlandırılan bir eğitim algoritmasını geliştirmişlerdir. Böylelikle TKDA'nın çözemediği XOR problemi bu yöntemle çözülebilmiştir ve Hopfield ağları ve BM kısıtlarından da kurtulmayı sağlamışlardır. Bu algoritmanın etkin bir öğrenmeyi mümkün kılması ve iyi sonuçlar verdiğinin gösterilmesiyle birlikte YSA yeniden popülerlik kazanmıştır.

1990'lı yıllarda algılayıcılar çok katmanlı bir yapıya geçmesine rağmen o dönemdeki bilgisayar sistemlerinin yetersizliği ve eğitimdeki zorluklar nedeniyle YSA'da beklenen ilerleme sağlanmamıştır. Hochreiter (1991) tarafından geri yayılım algoritmasının çok fazla eğitime ihtiyaç duyması nedeniyle öğrenme zamanının uzun olması ve derin ağ yapılarındaki geriye yayılımın eksiklerini ortaya koyan araştırmalarıyla birlikte gradyan kaybı problemini keşfetmiştir. Cortes and Vapnik (1995) tarafından ikili sınıflandırmalar için Destek Vektör Makinaları (DVM) geliştirilmiştir. Yapay öğrenme ve özelliklede bilgisayarla görüntü işleme konularında o dönemde alınan sonuçlar gerçek dünya için yapay zekanın kullanılabileceği görüşünü doğurmuştur. Ancak DVM'nin yüzeysel bir mimariye sahip olması dolayısıyla tam anlamıyla kategorize edilmemiş özelliklere ilişkin çözümler üretememiştir.

Boltzman makinelerinde bir çıkış katmanı olmadığı, görünür ve gizli katmanlardaki sinirlerin birbirine bağlanmasından kaynaklı oluşan bir takım kısıtlardan dolayı Hinton and Salakhutdinov (2006) tarafından Kısıtlı Boltzman Makinesi (KBM) adında daha ileri bir algoritma geliştirilmiştir. KBM'de ki sinirler BM'de ki gibi birbirine tamamen bağlı değildir ve görünür ve gizli birimlere ait katmanlarda her katmana ilişkin birimlerin birbirleri arasında herhangi bir bağ bulunmamaktadır. Daha sonra geliştirilecek olan Derin İnanç Ağları (DİA)'nın temelini oluşturmaktadır. Ardından Hinton vd. (2006) yayınladıkları ''Derin İnanç Ağları'' adlı çalışmalarında çok katmanlı derin mimarilerin çalışma prensiplerini ortaya koyarak nasıl başarılı bir şekilde eğitilebileceklerini göstermiştir.

Bayes tekniğinin derin öğrenmede kullanılmasıyla ilgili yapılan çalışmalar 1990'lı yıllara kadar uzanmaktadır. Bu tekniğin YSA ile ilişkisi ilk olarak fonksiyonlar üzerine olasılık dağılımlarını yerleştirerek modellemeler yapılmaya çalışılmasıyla başlamıştır. Ardından oluşturulan bu modeller tahminlerini, rasgele mi yoksa mantıklı bir şekilde yapmalı mı sorunsalı ortaya çıkmıştır. Bu durum, Gauss süreçlerinin modele entegre edilmesiyle birlikte gelişim göstermiştir.

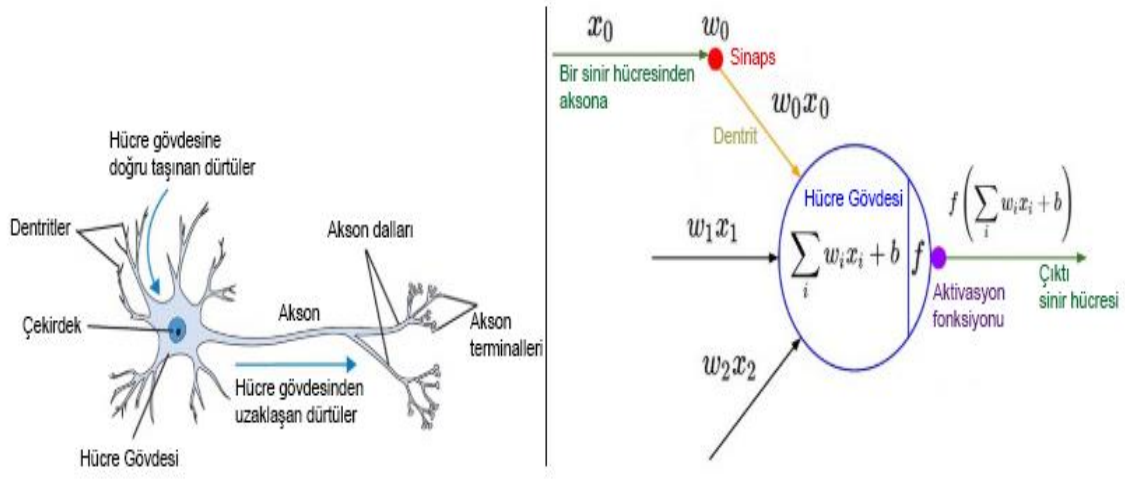
İlk olarak bir sinir ağındaki her ağırlığın üzerine bir olasılık dağılımı yerleştirilmesi fikri Mackay (1992) tarafından önerilmiştir ve Neal (1995) tarafından geliştirilmiştir. Neal (1995), gerçek dünyadaki problemler için sinir ağı modellerinin az sayıda gizli birim içeren ağlarla sınırlandırılması gerektiğine olan genel inancın yanlış olduğunu savunmuştur. Bir ağdaki gizli ünitelerin sayısının sonsuz büyüklükte olduğunu göz önünde bulundurmanın mantıklı olduğunu ve o dönemde Bayesçi modellerle iyi tahminler elde edilebileceğini göstermiştir. Williams (1997)'da, Neal (1995)'dan etkilenerek ağırlıklı doğrusal bir fonksiyon ve ardından doğrusal olmayan bir fonksiyonun özyinelemeli uygulamasıyla bir sinir ağında normal bir dağılımda her bir ağırlığa bir olasılık dağılımı yerleştirilerek sonsuz sayıda ağırlıklı bir Gauss süreci oluşturmuştur. Williams (1997) geniş bir önsel ağırlık sınıfına sahip sinir ağları için, sonsuz sayıda gizli birim içeren ağları kullanarak önsel fonksiyonların bir Gauss gizli birimleri olan ağırlıklara karşılık gelen kovaryans fonksiyonlarını kullanarak tahminin etkin bir şekilde yapılabileceğini göstermiştir. O yıllardaki bu tür çalışmalar sonucunda büyük verilerle çalışmak gibi yeni ihtiyaçlar ortaya çıkmıştır. Ancak bilgisayarların yeteri kadar güçlü olmamasından Bayesçi alana olan ilgi o dönemde azalmıştır.

Krizhevsky and Hinton (2009) tarafından yapılan çalışmada, doğal görüntülerin çok katmanlı modellerde nasıl eğitileceği açıklanmıştır. Çalışmasında milyonlarca renkli görüntüden oluşan bir veri kümesi kullanmıştır. Daha önceki çalışmalarda bu yöntem bazı araştırmacılar tarafından denenmesine rağmen uygulamada başarılı olamamışlardır. Krizhevsky and Hinton (2009) modellerinde KBM'lere ve DİA'lara odaklanmış ve modellerin, görüntü sınıflandırma problemi için daha yararlı olduğunu çalışmada göstermiştir. Günümüzde bilgisayarlardaki gelişmelere paralel bu türden Bayesçi fikirler alandaki yeni gelişmeler ve yeni sonuçlarla gelişimini devam ettirmektedir.

3. MATERYAL ve METOT

3.1 Yapay Sinir Ağları

Yapay sinir ağları, bir insan beynini oluşturan biyolojik sinir ağını taklit edip modelleyerek makinelerin öğrenim yapmasını ve gerçek hayatta insan benzeri görevlerin makineler tarafından yerine getirilebilmesini sağlayan işlemlerin temelidir. Yapay sinir ağları bu özelliğiyle görüntü ve ses tanıma, öğrenme gibi sınıflandırma problemlerine çözüm getirebilmekte ve dolayısıyla günümüzde kendisine çok geniş alanlarda uygulama imkanı bulmaktadır.



Şekil 3.1 Biyolojik bir sinir hücresinin ve matematiksel modelinin gösterimi (İnt.Kyn.2).

Sinir hücresi beynin temel hesaplama birimidir. İnsan sinir sisteminde yaklaşık 86 milyar sinir hücresi bulunur ve bunlar yaklaşık $10^{14} - 10^{15}$ sinaps ile birbirine bağlanırlar. Şekil 3.1'de biyolojik bir sinir hücresine ve ona karşılık gelen matematiksel modelin çizimi gösterilmektedir. Her bir sinir hücresi dendritlerinden giriş sinyalleri alır ve aksonu boyunca çıkış sinyalleri üretir. Akson, sonunda dallanır ve diğer sinir hücrelerinin dendritlerine sinapslar yoluyla bağlanır. Bir sinir hücresinin hesaplama modelinde, aksonlar boyunca hareket eden sinyaller (x_i), o sinapstaki (w_i) sinaptik kuvvetine dayanarak diğer sinir hücresinin dendritleri ile çarpımsal olarak ($w_i x_i$) etkileşime girer. Buradaki amaç, sinaptik kuvvetlerin (ağırlıklar, w) öğrenilebilir olması ve bir sinir hücresinin diğerinde etki gücünü ve yönünü, uyarıcı (pozitif ağırlık) veya kısıtlayıcı (negatif ağırlık) kontrol etmesidir. Temel modelde dendritler, sinyali toplandıkları hücre

gövdesine taşır. Son toplamın belirli bir eşiğin üstünde olması durumunda hücre uyarılır. Matematiksel modelde, ilgili hücreye gelen sinyaller toplamı daha sonra o hücredeki aktivasyon fonksiyonuna girdi olarak alınır. Kullanılan aktivasyon fonksiyonlarının türü birden fazladır ancak en yaygın olarak sigmoid fonksiyonu kullanılmaktadır. En sonunda aktivasyon fonksiyonu aldığı toplam girdiden bir çıktı üretir ve bu süreç sinirsel hesaplama olarak adlandırılır (İnt.Kyn.2).

Birçok yapay sinir ağı modeli, istatistiksel modellerle benzer olmasına rağmen kullanılan terminoloji arasında farklılıklar vardır. Literatürde bunlar arasında ilişkiyi gösteren çalışmalar bulunmaktadır (Sarle 1994). Çizelge 3.1'de bunlar arasındaki ilişkiyi gösteren terimlerden bazıları gösterilmiştir.

Çizelge 3.1 Yapay sinir ağları ve istatistik terminolojilerinde kullanılan aynı kavramları belirten bazı terimler (Sarle 1994)

Yapay Sinir Ağları Terminolojisi	**İstatistik Terminolojisi**
Yapay Sinir Ağı	Model
Ağırlık	Parametre
Girdi	Bağımsız Değişken
Çıktı	Tahmin Değeri
Hedef	Bağımlı Değişken
Hata	Artık
Hata Çizgisi	Güven Aralığı
Nöron	Temel Fonksiyon
Girdi Tabakası	Bağımsız Değişkenler Kümesi
Gizli Tabaka	Temel Fonksiyonlar Kümesi
Çıktı Tabakası	Tahmin Değerleri Kümesi
Örüntü	Gözlem
Eğitim, Öğrenme ya da Adaptasyon	Kestirim ya da Optimizasyon
Çevrimiçi Öğrenme	Örneklem Adaptasyonu
Çevrimdışı Öğrenme	Grup Adaptasyonu
Fonksiyonel Bağlantı	Dönüşüm
Sınıflama	Diskriminant Analizi
Eşleme, Yaklaşım ya da Denetimli Öğrenme	Regresyon
Denetimsiz Öğrenme ya da Şifreleme	Veri İndirgeme

Yapay sinir ağları öğrenme yöntemine göre gözetimli, gözetimsiz ve destekleyici olarak sınıflandırılırlar.

3.1.1 Algılayıcılar

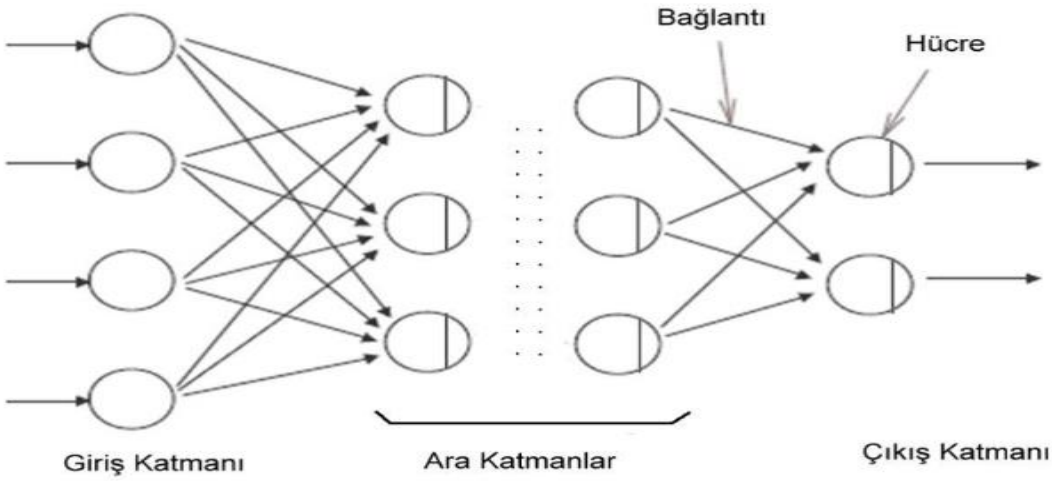
Algılayıcı, girdi ve çıktı katmanlarından oluşan en temel tek katmanlı doğrusal bir sinir ağıdır. Girdi ve çıktılar birbirlerine ağırlıklarla bağlanmaktadır ve girdi verilerinin sınıflandırılmasını sağlayarak gözetimli öğrenmede kullanılır.

Doğrusal fonksiyonlar $y = wx + b$ şeklinde tanımlanır. Burada x değeri bir girdi olduğu için bağımsız değişken, y değeri ise x'e bağlı olduğu için bağımlı değişken olarak tanımlanır. w değeri ağırlık ve b değeri sapma olarak ifade edilerek fonksiyonun parametreleri olarak tanımlanmaktadır. Örneğin burada ki x girdi değerimizi fare resimleri olarak tanımlıyorsak x, fare resmine ait görüntü, y ise bu resmin fareye ne kadar benzediğine dair skoru verir. Bu çıktı skorunu iyileştirmek için b, sapma ve w, ağırlık değerleri kullanılır. Burada amaç en iyi skoru veren parametre değerleri w ve b 'yi hesaplamaktır.

Algılayıcı; giriş değerleri veya bir giriş katmanı, ağırlıklar ve sapma, net toplam ve aktivasyon fonksiyonu olmak üzere 4 bölümden oluşmaktadır. Algılayıcının çalışma şekli aşağıdaki gibidir:

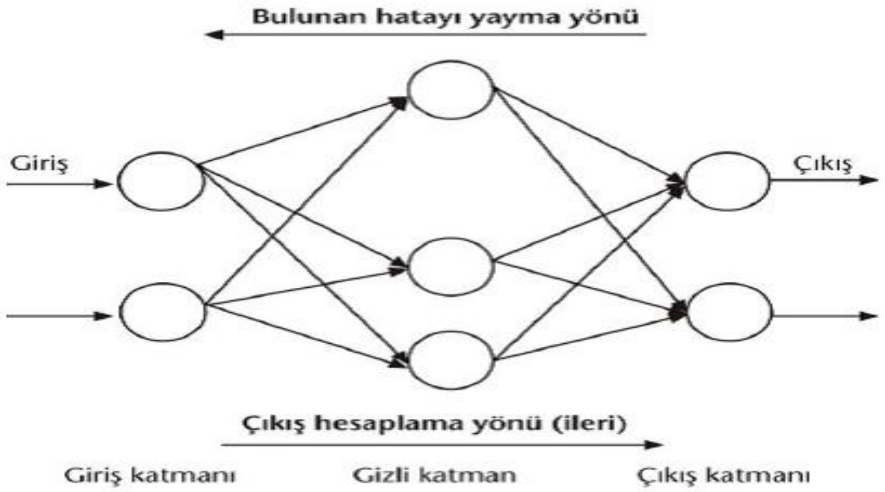
- Tüm x girdileri w ağırlıklarıyla çarpılır.
- Net toplam olarak adlandırılan adım tüm çarpım değerlerinin toplanarak eklenmesi işlemidir.
- Bu toplam daha sonra aktivasyon fonksiyonuna uygulanır.

Tek katmanlı algılayıcı, ağı doğrusal olarak sınıflandırdığından dolayı çıkış fonksiyonunun çıktısı kullanılan hücre modeline göre sınıfları temsilen $(0,1)$ ya da $(-1,1)$ değerlerini almaktadır. Girdilere göre iki sınıfı birbirinden ayıran doğruyu bulmaktadır, dolayısıyla sadece iki sınıfı birbirinden ayırabilir. Doğrusal olmayan modellere çözüm getiremediğinden çok katmanlı algılayıcılar geliştirilmiştir.



Şekil 3.2 Çok katmanlı algılayıcı modeli (Öztemel 2012).

Çok katmanlı algılayıcı da YSA giriş katmanı, ara katman ve çıkış katmanından oluşur. Giriş hücresi sayısında herhangi bir sınır yoktur ancak değişken sayısı neyse giriş birimlerinin sayısı da odur. Herhangi bir önceki katmandaki sinir hücrelerinin çıkışı bir sonraki katmandaki sinir hücrelerine giriş olarak ağırlıklarla bağlanır. Sinir hücrelerinin arasında döngüsel herhangi bir bağlantı yoktur. Burada sinir ağındaki her hücre girişini dikkate alarak sinir hücresinin ağırlıklarıyla çıkışını hesaplayan aktivasyon fonksiyonuna sahiptir. Bu fonksiyonlar farklı katmanlardaki problemlere çözüm getirebilmesi için tasarlanmışlardır. Ardından bu hesaplamalar ve ilgili aktivasyon fonksiyonuyla çıkış belirlenir. Ardından geriye doğru hesaplamayla çıkış ve ara hücreleri ya da katmanlardaki çıkışlardan elde edilen hata fonksiyonunun önceki giriş ve ara hücreleri ya da katmanlarıyla geri yönde yani hata fonksiyonuna girilmesiyle ağırlık güncellenmesi işlemi yapılmaktadır. Bu yapısı gereği geriye doğru hesaplama, doğrusal olmayan bir yöntemdir. Aynı zamanda hücreler ya da katmanlar arasında geriye doğru hesaplamanın yapılabilir olması nedeniyle farklı geriye doğru hesaplamalı YSA yapıları gözlenebilmektedir. Şekil 3.3'te çıkışlarından giriş katmanına geriye doğru hesaplamalı bir YSA yapısı görülmektedir.



Şekil 3.3 Geriye doğru hesaplama (İnt.Kyn.3).

Eğitim, yapay sinir ağlarındaki sinir bağlantılarındaki ağırlık değerlerinin belirlenmesi aşamasıdır. Bu ağırlıklar makine tarafından başlangıçta rastgele atanır. Makine örnekleri çalıştıkça yapay sinir ağı da ağırlıkları günceller. Ağırlıkların güncellenmesi istenilen çıktı elde edilinceye kadar devam eder. Bu sürece öğrenme süreci denir.

Eğitim aşamasında ilk olarak eğitim, doğrulama ve test veri kümeleri rassal olarak oluşturulur. Genellikle eğitim kümesi örneklerin %50−70 kadarını, doğrulama ve test kümesi örneklerin eşit oranda dağılmasıyla %50−30'unu oluşturmaktadır. Makinenin eğitim kümesinde çalışmasıyla ağırlıklar güncellenir. Makinenin ezber yapmaması ve modelin etkili olması için eğitimi doğrulama kümesindeki hata değişimine bakılarak eğitimin ne zaman bitmesi gerektiği belirlenir. Geriye doğru hesaplamayla derin öğrenmede parametrelerin güncellenmesi işlemi yapılmaktadır. Geriye doğru hesaplama algoritması Tablo 1'de sunulmuştur.

Tablo 1 Geri yayılım algoritması (Fausett 1994)

Adım 0 Ağırlıkları rastgele ve küçük değerler atayarak başlat
Adım 1 Tüm giriş ve çıkış eğitim vektörleri için 2-9 arasındaki adımları gerçekleştir.
Adım 2 Her bir eğitim parçası için 3-8 adımlarını uygula
İleriye doğru hesaplama
Adım 3 Her giriş birimi $(x_i, i = 1, \ldots\ldots, n)$ x_i, giriş sinyalini al ve bir üst

Tablo 1 (Devam) Geri yayılım Algoritması (Fausett 1994)

katmandaki gizli birimlere aktar.

Adım 4 Her gizli birim $(z_j, i = 1, \ldots, n)$ ağırlıklandırılmış giriş sinyallerini toplar:

$z\_in_j = v_{0j} + \sum x_i v_{ij}$,

çıkış sinyalini hesaplamak için aktivasyon fonksiyonu uygulanır:

$z_j = f(z\_in_j)$,

ve sinyali çıkış katmanındaki tüm birimlere gönderir.

Adım 5 Her çıkış birimi $(y_k, k = 1, \ldots, m)$ ağırlıklandırılmış giriş sinyallerini toplar:

$y\_in_k = w_{0k} + \sum_{i=0}^{n} z_j w_{jk}$

çıkış sinyalini hesaplamak için aktivasyon fonksiyonu uygulanır:

$y_k = f(y\_in_k)$,

Geriye doğru hesaplama

Adım 6 Her çıkış birimi $(y_k, k = 1, \ldots, m)$, giriş eğitim verisine karşılık gelen hedef değeri alır ve hata terimini hesaplar:

$\delta = (t_k - y_k) f'(y\_in_k)$,

sonra ağırlık düzeltme terimini hesaplar:

$\Delta w_{jk} = \alpha \delta_k z_j$,

daha sonra w_{0k} güncellemesinde kullanılan sapma düzeltme terimini hesaplar:

$\Delta w_{0k} = \alpha \delta_k$ ve δ_k'ı alt katmandaki birimleri gönderir.

Adım 7 Her gizli birim $(z_j, i = 1, \ldots, p)$ üst katmanlardan gelen delta girişlerini toplar:

$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk}$

hata terimini hesaplamak için aktivasyon fonksiyonunun türeviyle çarpılır:

$\delta_j = \delta\_in_j f'(z\_in_j)$,

daha sonra v_{ij} güncellemek için ağırlık düzeltme terimini hesaplar:

Tablo 1 (Devam) Geri yayılım Algoritması (Fausett 1994)

$\Delta v_{ij} = \alpha\, \delta_j\, x_i$,

daha sonra v_{0j} güncellemek için sapma düzeltme terimini hesaplar:

$\Delta v_{0j} = \alpha\, \delta_j$.

Ağırlıkları ve sapmaları güncelleştir.

Adım 8 Her çıktı birimi $(y_k \quad , k = 1 , \ldots . , m)$ sapma değerleri ve $(j = 0 , \ldots . . , p)$ ağırlıkları için:

$w_{jk}(yeni) = w_{jk}\,(eski) + \Delta w_{jk}$,

her gizli birim $(z_j , i = 1 , \ldots . . , p)$ sapma değerleri ve $(i = 0 , \ldots . . , n)$ ağırlıkları güncelle:

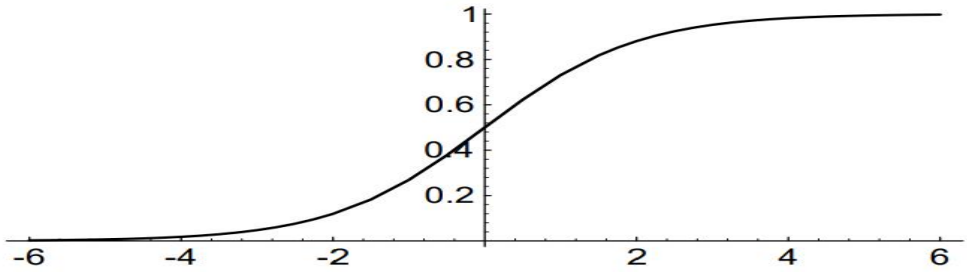
$v_{ij}\,(yeni) = v_{ij}\,(eski) + \Delta v_{ij}$.

Adım 9 Durdurma koşulunu test et.

3.1.2 Aktivasyon Fonksiyonları

Transfer fonksiyonu olarak adlandırılabilen bu aktivasyon fonksiyonları temel olarak yapay bir hücreden çıkan transfer fonksiyonudur ve diğer yapay hücreye sinyal gönderir. Aktivasyon fonksiyonu temel olarak doğrusal aktivasyon fonksiyonu ve doğrusal olmayan aktivasyon fonksiyonu olmak üzere iki türe sahiptir ve genellikle doğrusal olmayan bir fonksiyon kullanılır. Geri yayılım algoritmasında öğrenme işlevi türevdir ve $A = ax$ doğrusal fonksiyonunun x'e göre türevi a'dır. Bu türevin x ile ilişkisinin olmadığını gösterir ve geri yayılım algoritmasında türevinin sabit olması dolayısıyla işlev koordinat düzleminde doğrusal bir çizgi oluşturmaktadır. Bundan kaynaklı işlevlerin çıktısı herhangi bir aralık arasında sınırlandırılmadığından giriş ve çıkış katmanları arasında aynı doğrusal sonuç elde edilir. Bu da birbirine bağlanan hücrelerin işlevsiz olduğu anlamına gelmektedir.

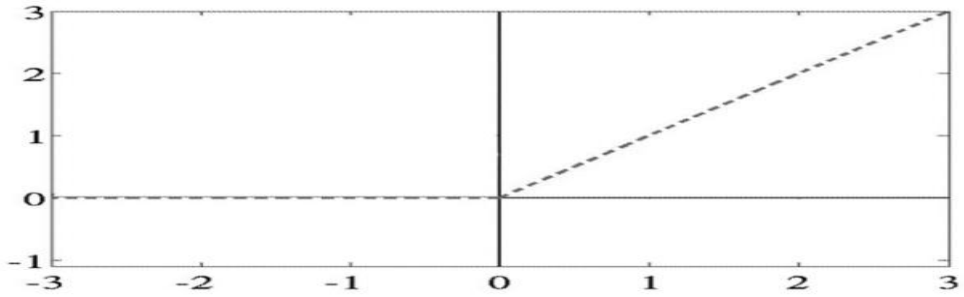
Sınıflandırma problemi için sürekli ve türevlenebilir bir fonksiyon olması nedeniyle Sigmoid fonksiyonu ve gradyan kaybı problemine çözüm getirebilmesinden dolayı Doğrultulmuş Doğrusal Birim (DoDB) fonksiyonu daha fazla tercih edilmektedir.



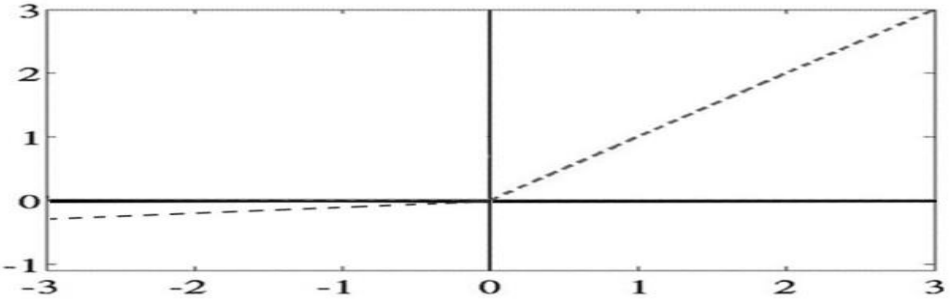
Şekil 3.4 Sigmoid fonksiyonu (LeCun *et al.* 2012)

Aynı zamanda lojistik fonksiyon olarak da adlandırılan sigmoid fonksiyonu doğrusal olmayan bir fonksiyon olduğu için çıktı da doğrusal değildir. Bu da doğrusal fonksiyonun aksine türevinin sabit olmadığını göstermektedir. Şekil 3.4'te x değerleri -2 ile +2 arasında iken y değerleri daha diktir. Bu da x değerlerindeki yapılan küçük değişimlerin y değerlerinde daha büyük değişimlere neden olduğunu göstermektedir. Bu sınıflandırma açısından daha iyi ayırımlar yapılabileceği anlamına gelir. Sigmoid fonksiyonu doğrusal fonksiyonun aksine $(-\infty, +\infty)$ aralığında her zaman $[0,1]$ arasında değerler üretir. Bunun için derin öğrenme ağında çıktı katmanında olasılıksal çıktıyı tahmin için kullanılır ve yorumlamada kolaylık sağlamaktadır.

Derin öğrenme yöntemlerinde kullanımı giderek artan DoDB, $[\mathbf{0}, +\infty)$ aralığında değer almasından dolayı tüm pozitif değerler için doğrusal ve tüm negatif değerler için de sıfır değerini alır. Bu da modelin karmaşık olmamasından dolayı hesaplanması için daha az zamana ihtiyaç duyar. Fakat bu durum geri yayılım algoritmasında da türevinin sıfır olması ve öğrenmenin o bölgede gerçekleşmemesi demektir. Bu zaman açısından iyi gibi gözükse de gerçekte kötü bir durumdur. Örneğin, modeldeki görüntüler de fareler tespit ediliyor ve görüntülerin bir araba ile ilgisi varsa açıkça etkinleştirilmemesi gereken, aracın tekerleğini tanımlayabilen bir sinir hücresi olabilir. Bunun modelde sıfır değer almasıyla modelde hesaplanmaması zaman açısından iyi gibi gözükse de farklı bir görüntüde o tekerlek model için anlamlı olabilmektedir. Çünkü arabanın tekerleğinde bir fare gizlenmiş olabilir. Bunu ortadan kaldırmak için Sızıntı DoDB fonksiyonu geliştirilmiştir.



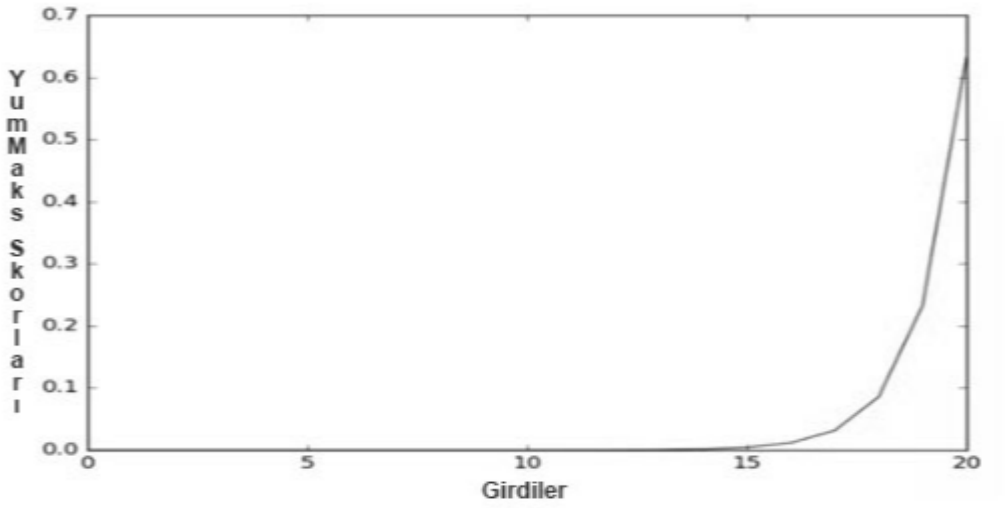
Şekil 3.5 DoDB fonksiyonu (Glorot *et al.* 2011).



Şekil 3.6 Sızıntılı DoDB fonksiyonu (Maas *et al.* 2013).

Sızıntı DoDB fonksiyonu, negatif değerler için sıfır değeri yerine küçük bir eğime sahiptir. Yani $x < 0$ olduğunda $y = 0.01x$ olabilir. Burada ki 0.01 değeri sızıntı değeri olarak ifade edilir. Böylelikle sıfır değeri olmaz ve sıfıra yakın eğimle negatif bölgelerdeki öğrenme sağlanmış olur.

YumMaks fonksiyonu herhangi bir örneğin belirli bir sınıfa ait olması olasılığını hesaplar. Daha sonra hesaplanan olasılıklar, verilen girdiler için hedef sınıfın belirlenmesinde yardımcı olmaktadır. YumMaks kullanmanın en büyük avantajı çıkış olasılıkları aralığıdır. Bu aralık 0 ile 1 arasındadır ve tüm olasılıkların toplamı 1'e eşit olmaktadır. Çoklu sınıflandırma modeli için kullanılan YumMaks fonksiyonu her sınıfın olasılıklarını döndürür ve ilgili örnek en yüksek olasılığa sahip olan sınıfa atanır. Formülü, ilgili örneğin hesaplanan üstel değeri ile tüm örneklerin üstel değerlerinin toplamının oranıdır. Bu oran ilgili örnek için YumMaks fonksiyonunun çıktısıdır (İnt.Kyn.4). Sınıflandırma çıktılarını olasılıksal olarak verdiği ve yorumlamada kolaylık sağladığı için derin öğrenme modellerinin çıkış katmanında yaygın olarak kullanılmaktadır.



Şekil 3.7 YumMaks grafiği (İnt.Kyn.4).

Aktivasyon fonksiyonlarının formül ve türevlerinin genel gösterimi Çizelge 3.2'deki gibidir.

Çizelge 3.2 Aktivasyon fonksiyonlarının formülleri ve fonksiyonların türevleri

Fonksiyon	**Formül**	**Türev**
Doğrusal Fonksiyon	$f(x) = x$	$f'(x) = 1$
Sigmoid Fonksiyonu	$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1-f(x))$
Dodb Fonksiyonu	$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$
Sızıntı Dodb Fonksiyonu	$f(x) = \begin{cases} \alpha x, & x < 0 \\ x, & x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha, & x < 0 \\ 1, & x \geq 0 \end{cases}$
YumMaks Fonksiyonu	$f(x) = log_e(1+e^x)$	$f'(x) = \frac{1}{1+e^{-x}}$

3.1.3 Gradyan İniş

Bir fonksiyonun nasıl hareket ettiğini belirleyebilmek yani fonksiyonun nerede arttığı, azaldığı gibi durumları inceleyebilmek için eğimlerden yararlanılmaktadır. Bu hareket fonksiyondaki değişkende oluşacak değişimin fonksiyonda yaratacağı durumun, değişkendeki değişime oranıdır. Fonksiyona teğet d doğrusu çizilecek olursa koordinat

düzleminde x'teki değişim $f(x)$'te de değişime neden olacaktır. Bu $f(x)$'te oluşturacağı değişim x'teki değişime oranladığında d teğetinin eğimi elde edilir. Bu da $f(x)$'in x noktasındaki türevidir. Yani fonksiyonun o noktadaki türevi, o noktada ilgili eğriye teğet olan doğrunun eğimidir. Bu durumda fonksiyonun hangi yönde arttığının anlaşılabilmesi açısından eğim kullanılabilir. $y = f(x)$ eğrisinin herhangi bir noktasında dururken en tabana inmeye çabalanırsa o noktada fonksiyonun türevi, x'teki değişim hangi yönde olursa yukarı ya da aşağı hareketi belirler. Türev ile elde edilen eğim hareketi artırmaya yönelik olursa yani değişkeni eğimle artırılırsa üste çıkılabilir. Ayrıca eğim hareketi azaltmaya yönelik olursa yani değişkeni eğimin tersi yönünde artırılırsa tabana ulaşılabilir. Gradyan bir fonksiyonun tüm kısmi türev bilgilerini içerir ve fonksiyonun değişken sayısı ne olursa olsun tüm bilgileri tek vektörde bir araya getirir. Gradyan boyutsal koordinat düzleminde en uzun yüksekliği tercih etmesinden dolayı artımın en çok olduğu yönü temsil eder. Onun için gradyan yönünde hareket edilirse maksimuma ve aynı mantıkla gradyanın tersi yönünde hareket edilirse de minimuma ulaşılır. Gradyan iniş, fonksiyona ilişkin rassal bir nokta koordinatlarıyla belirlenip, koordinatları kısmi türev vektörünün tersi yönünde değiştirerek minimum noktaya ulaşılmaya çalışılması işlemidir. Koordinat düzleminde minimuma ulaşmak için gradyan inişi eğimi minimuma hareket ettirir. Eğim, minimuma yaklaştıkça kademeli olarak azalır. Bu adım, yani yeni değerle eski değer arasındaki fark gitgide azalmaktadır. Buna göre de gradyan iniş için bir durdurma ölçütü belirlenebilir. Bu ölçüte göre adım ilgili değerin altında olduğunda, minimuma ulaşılmış demektir ve işlem sonlandırılmaktadır.

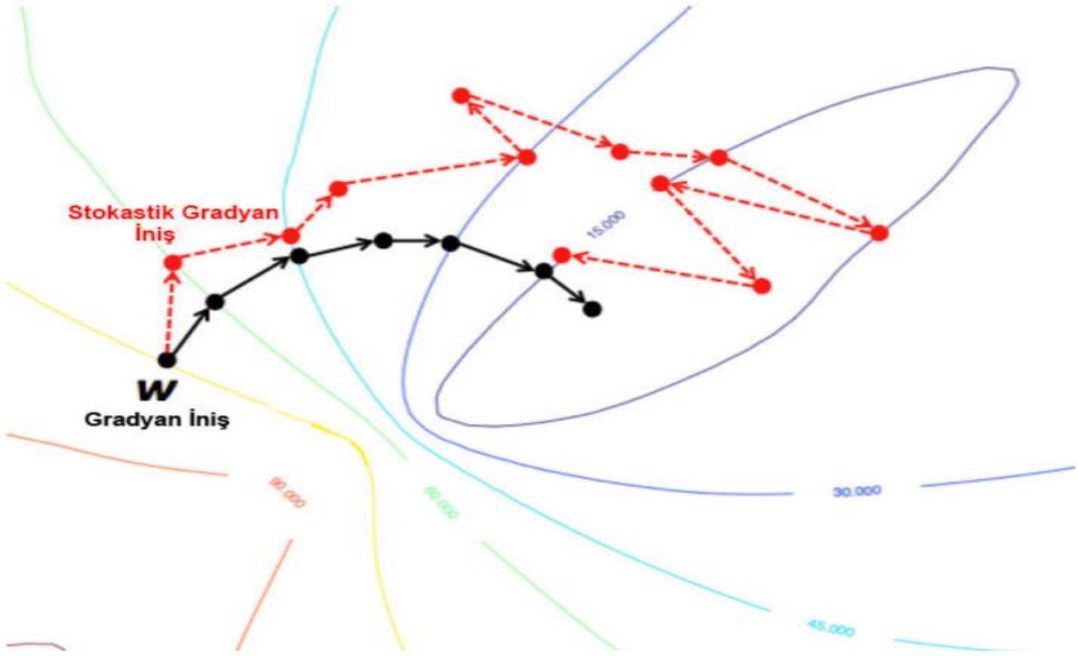
Amaç fonksiyonunun gradyanını hesaplamak için ne kadar veri kullanıldığına bağlı olarak toplu gradyan inişi algoritmaları toplu (batch) gradyan inişi ve stokastik gradyan inişi şeklinde ikiye ayrılır. Toplu gradyan inişi, tüm eğitim veri kümesi için w parametrelerine göre fonksiyonun gradyanını hesaplar. Büyük veri kümeleriyle çalışılması gerektiğinden önceden belirlenmiş belirli bir turda güncelleme işlemi veri kümesinin tamamıyla yapılır. Dolayısıyla makine yüksek hafızaya ihtiyaç duyar ve gradyan iniş yavaştır. Toplu gradyan inişi aşağıdaki gibi verilebilir (Ruder 2016):

$$w = w - \alpha \nabla_w E(w) \tag{3.1}$$

$E(w)$, amaç fonksiyonu, $w \in R^d$, model parametreleri ve α, öğrenme oranıdır.

Stokastik gradyan iniş yönteminde ise eğitim veri kümesinden rassal olarak seçilen her örnek $x^{(i)}$ve etiketi $y^{(i)}$ için bir parametre güncellemesi gerçekleştirilir ve her turda bu seçilen veri kümesi karıştırılır. Amaç fonksiyonunun yoğun bir şekilde dalgalanmasına neden olan yüksek bir varyansla sık sık güncellemeler yapar. Diğer yönteme göre daha hızlı minimuma ulaşır ve makine daha az hafızaya ihtiyaç duyar. Bu yüzden daha sık kullanılır. Formülü aşağıdaki gibidir (Ruder 2016):

$$w = w - \alpha \nabla_w E\left(w; x^{(i)}; y^{(i)}\right) \tag{3.2}$$

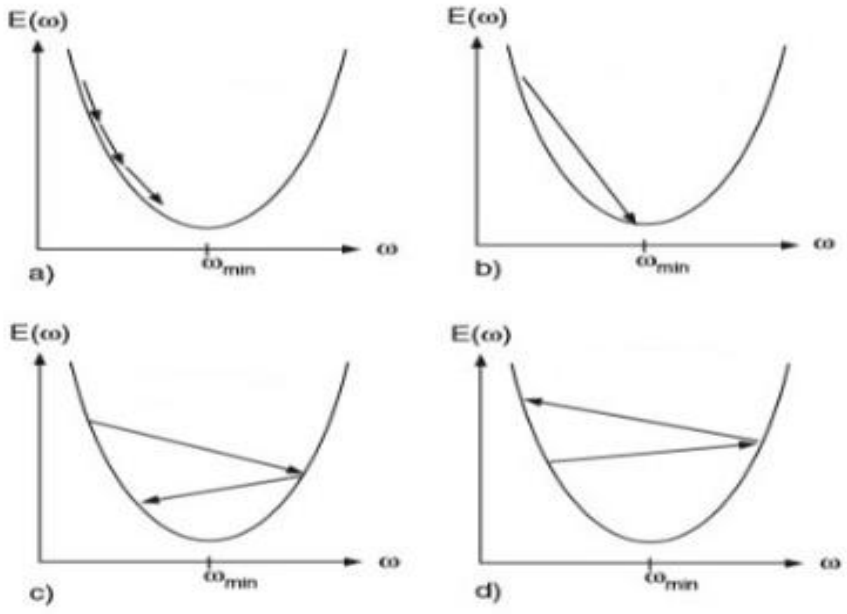


Şekil 3.8 Gradyan iniş ve stokastik gradyan iniş (Int.Kyn.5)

Gradyan iniş yöntemlerinin minimize etmeye çalıştığı hata fonksiyonu en az bir yerel minimuma sahip olduğunda, gradyan inişi yönteminin bu yerel minimuma yakın bir noktaya ulaşması beklenir. $L2$ kaybı, $L1$ ve $L2$ düzenlemeleri de derin öğrenmede gradyan kaybını önlemek için kullanılırlar.

3.1.4 Öğrenme Oranı, Momentum Katsayısı ve Sönümleme

Öğrenme oranı, bir ağ için eski eğitimini yenileri için ne kadar sürede terk ettiğidir. Bir çocuk 15 köpek örneği görürse ve hepsi kahverengi tüylere sahipse, köpeklerin kahverengi tüyü olduğunu düşünecek ve bir köpeği tanımlamaya uğraşırken kahverengi tüyü arayacaktır. Çocuk yeni beyaz bir köpek görür ve ailesi ona bunun bir köpek olduğunu söylerse çocuk burada bir denetimli öğrenme yaparak büyük bir öğrenme oranı ile kahverengi tüyün köpeklerin en önemli özelliği olmadığını anlayacaktır. Küçük bir öğrenme oranı ile bu beyaz köpeğin bir aykırı olduğunu ve köpeklerin hala kahverengi olduğunu düşünecektir. Önemli olan, daha yüksek bir öğrenme oranının ağın zihnini daha hızlı değiştirmesi anlamına gelmesidir. Bu yukarıdaki durum için iyi olsa da aynı zamanda kötü de olabilir. Öğrenme oranı çok yüksekse, beyaz olanlardan daha fazla kahverengi köpek görmüş olsa bile tüm köpeklerin beyaz olduğunu düşünmeye başlayabilir.



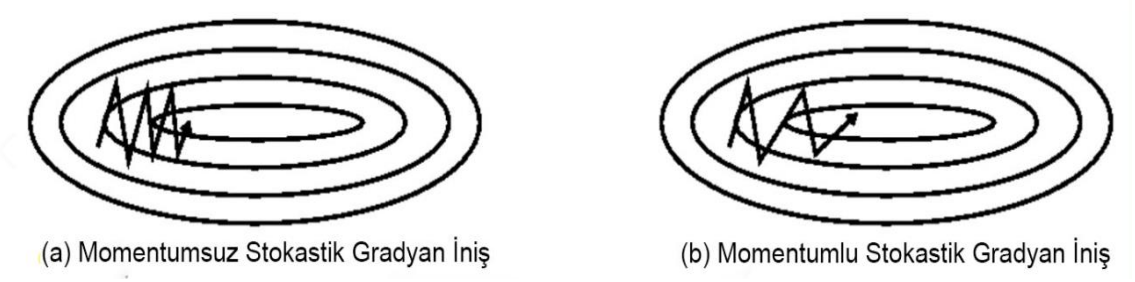
Şekil 3.9 Farklı öğrenme oranlarının yakınsamaya etkisi (LeCun *et al*. 2012).

Şekil 3.9'da öğrenme oranı çok küçük olduğunda (a)'daki gibi yavaş şekilde minimuma yaklaşır. (b)'de ise en iyi öğrenme oranı hızlı bir şekilde minimuma ulaşır. (c) ve (d)'de ise öğrenme hızı büyüktür ve ıraksak davranışlara neden olarak minimumdan uzaklaşır.

Bu durum ilgili örnekte de anlatıldığı gibi istenmeyen bir durumdur. Burada öğrenme oranı sabit bir değer olmasının yanında artan bir değer olarak ve hatta momentum değerine bağlı bir değer olarak da belirlenebilir. Kesin bir değer yoktur ve genellikle 0.01 olarak kullanılmakta belli bir turdan sonra 0.001'e düşürülmektedir. Öğrenme oranı modelimizin minimuma ne kadar hızlı yaklaşabileceğini belirler. Bu nedenle, daha kısa zamanda modeli eğitmeyi sağlar.

$$w_{ij}\,(yeni) = w_{ij}\,(eski) - \alpha \Delta w_{ij} \tag{3.3}$$

Ağırlık güncelleme kuralından $\Delta w_{ij} = -\alpha \frac{\partial E}{\partial w}$ ve α, öğrenme oranıdır (LeCun *et al*. 2012).

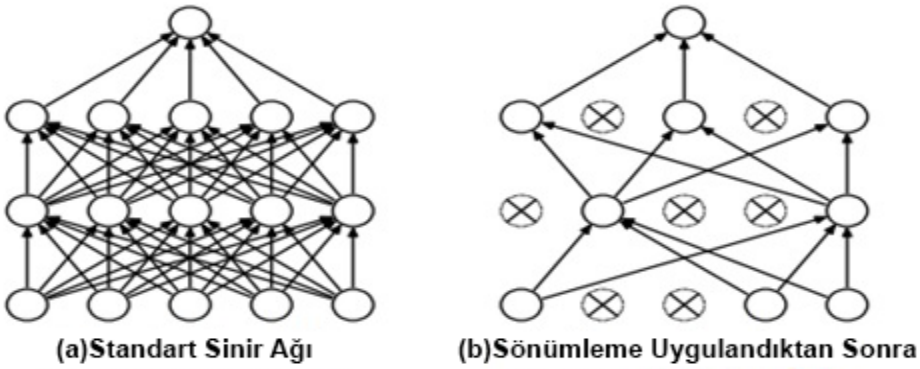


Şekil 3.10 Stokastik gradyan inişinin momentumsuz ve momentumlu gösterimi (İnt.Kyn.6).

Momentum, gereksiz parametre güncellemelerini azaltarak minimuma yakınsamayı daha az salınımla daha hızlı bir şekilde yapar. Momentum katsayısı 0 ile 1 arasında seçilebilir ama genellikle 0,8 ile 0,99 arasında bir değer verilir.

$$\Delta w_{ij}\,(yeni) = \alpha_i\,\delta_i\,y_j + m\Delta w_{ij}\,(eski) \tag{3.4}$$

Burada m, momentum katsayısı ve $\delta_i = \frac{\partial E}{\partial w_i}$'dir (LeCun *et al*. 2012).



Şekil 3.11 Sönümleme sinir ağı modeli (Srivastava *et al.* 2014)

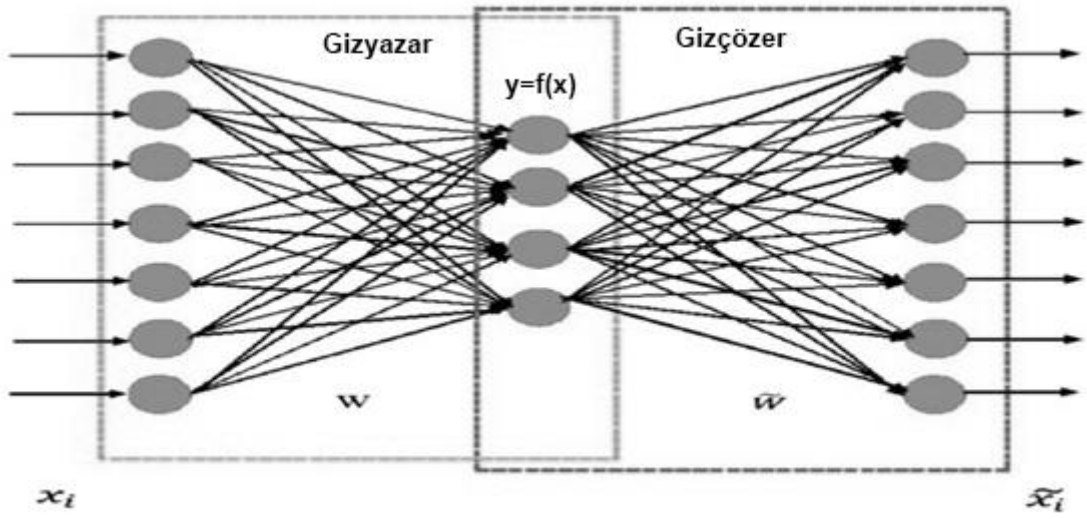
Şekil 3.11 (a)'da ağın orijinali ve (b)'de sönümleme uygulanmış hali gösterilmektedir. Sönümlemenin yani katmanlarda belli bir eşik değerinin altındaki çok küçük ağırlıklara sahip hücrelerin rassal azaltılması öğrenimi arttırmaktadır ve eğitim zamanını kısaltarak performansını yükseltmektedir. Böylece ağın aşırı öğrenme yani ezber yapması önlenmiş olur. Sönümleme çoğunlukla tam katmanlarda kullanılır. Değeri 0 ile 1 arasındadır. Çoğunlukla da 0.5 değeri verilir. Ancak tüm katmanlar için sabit bir değer olması zorunluluğu yoktur (Srivastava *et al.* 2014).

3.2 Otokodlayıcı ve Yığılmış Otokodlayıcı

Özellik çıkarma, orijinal verileri yüksek boyutlu alandan daha düşük boyutlu bir alana dönüştürür. Bu dönüşüm doğrusal veya doğrusal olmayan bir şekilde gerçekleştirilebilir. Özellikle, n örnek ve M özellik ile verilen bir veri kümesi X dikkate alındığında, orijinal özellik seti F_0 olarak adlandırılır ve bir özellik çıkarım fonksiyonu T, yeni özellik seti F_N'i oluşturur. Burada $|F_N| < |F_0|$' dır. Genel olarak, özellik çıkarma yöntemlerinin amaç fonksiyonları, orijinal $F_0(X)$ ve yeni özellik seti $F_N(X)$ arasındaki farkı en aza indirger (Meng *et al.* 2017).

Temel Otokodlayıcı'da 3 katman vardır. Bunlar giriş katmanı, gizli alan ve çıkış katmanıdır. Giriş katmanındaki hücre sayısı ile çıkış katmanındaki hücre sayısı eşittir. Buradaki amaç veriyi özellik çıkarımla yeniden boyutlandırmaktır. Giriş katmanı ile gizli katman arasındaki alan gizyazar olarak adlandırılır. Bu alan çok boyutlu veriyi az boyuta

düşürmeye yarar. Gizli katman ile çıkış katmanı arasındaki alan ise gizçözer olarak adlandırılır. Gizçözer, sıkıştırılmış gizli alanın boyutunu artırarak girişi yeniden yapılandırmaya çalışır. Kısaca çalışma şekli, giriş vektörü rassal ağırlıklarla gizli katmana gizyazar edilir ve gizli katmanda ki değerler yine rassal ağırlıklar gizçözer edilerek çıktı hücreleri oluşturulur. Daha sonra hata hesaplanır ve belirli sayıda tur ile geriye doğru hesaplama yapılarak minimum hata bulunur. Çıktı değerlerinin tahmininde çoğunlukla YumMaks fonksiyonu kullanılır.



Şekil 3.12 Geleneksel otokodlayıcı, x ve $\tilde{x}$ arasındaki yeniden yapılanması (Ahmed *et al.* 2018).

Şekil 3.12'de gösterildiği gibi n örnek ve d özelliği olan bir veri, örnek x göz önüne alındığında y gizyazar çıkışı, x'in indirgenmiş halini temsil eder ve gizçözer, x ve $\tilde{x}$ arasındaki farkı en aza indirerek gizyazardan orijinal veri seti x' i yeniden yapılandırmak üzere ayarlanmıştır. Burada y gizli katmandır. Meng vd. (2017)'e göre denklemleri:

$$y = f(x) = s_f\,(w_x + b_x\,) \tag{3.5}$$

s_f ; doğrusal olmayan bir aktivasyon fonksiyonudur yalnız özdeşlik fonksiyonuysa doğrusal davranır. Gizyazar bir ağırlık matrisi w ve bir sapma vektörü b tarafından parametrelenir.

$$\tilde{x} = g(y) = s_g(\tilde{w}_x y + b_y) \quad (3.6)$$

gizyazar fonksiyonu g, gizli katman y, $\tilde{x}$'in yeniden bir yapılandırmaya geri dönmesi, gizyazarın aktivasyon fonksiyonu olan s_g , doğrusal ya da bir sigmoid'dir, bir sapma vektörü b_y ve $\tilde{w}$ matrisi gizyazarın parametreleridir.

Bir otokodlayıcı eğitimi, verilen veri kümesi x üzerindeki yeniden yapılandırma kaybını en aza indiren parametreler $\theta = (w, b_x, b_y)$ parametrelerini içerir ve amaç fonksiyonu (Meng *et al.* 2017):

$$\theta = \underbrace{min}_{\theta} L(x, \tilde{x}) = \underbrace{min}_{\theta} L(x, g(f(x))) \quad (3.7)$$

şeklinde verilir. Lineer yeniden yapılandırma için, yeniden yapılandırma kaybı (L_1) genellikle hata karelerden hesaplanır (Meng *et al.* 2017):

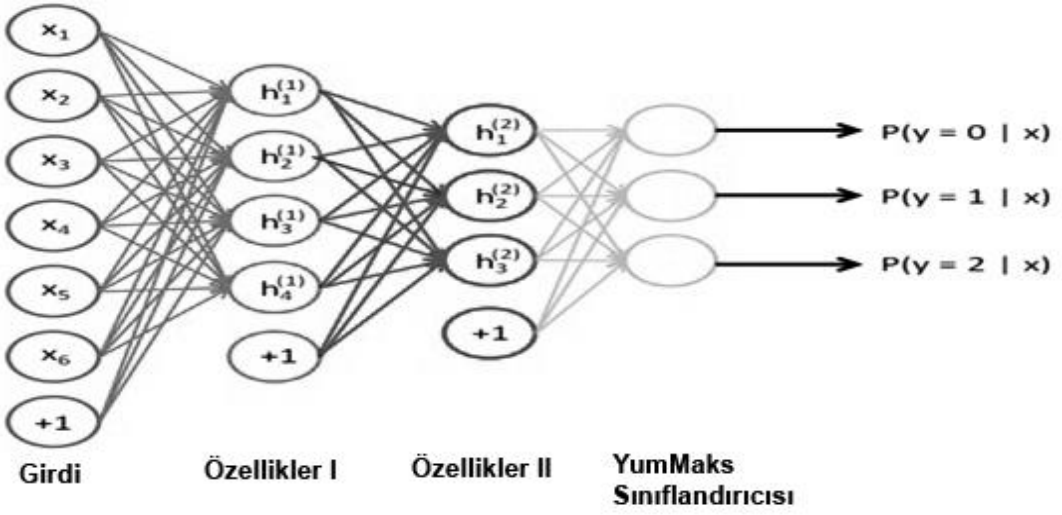
$$L_1(\theta) = \sum_{i=1}^{n} || x_i - \tilde{x}_i ||^2 = \sum_{i=1}^{n} || x_i - g(f(x_i)) ||^2 \quad (3.8)$$

$x_i \in x$, $\tilde{x}_i \in \tilde{x}$'dir.

Lineer olmayan yeniden yapılandırma için, yeniden yapılanma kaybı (L_2) genellikle çapraz entropiden hesaplanır (Meng *et al.* 2017):

$$L_2(\theta) = - \sum_{i=1}^{n} [x_i \log(y_i) + (1 - x_i) \log(1 - y_i)] \quad (3.9)$$

$y_i \in y$'dir.



Şekil 3.13 Yığılmış otokodlayıcı (İnt.Kyn.7)

Ağa birden fazla gizli katman eklendiği durum Yığılmış Otokodlayıcı olarak adlandırılır. Gizli katman yığılmalarının kolay olması, otokodlayıcının sinir ağı modeli tabanlı özellik çıkarma yöntemi olarak önemli bir avantajıdır. Bu sayede ağa gizli katmanlar eklenerek orijinal özelliklere ek farklı seviyelerde yeni özellik çıkarımları yapılabilmektedir. l katmanlı bir yığılmış otokodlayıcı için Meng vd. (2017)'e göre gizyazar süreci:

$$y = f_l (\dots f_i (\dots f_1(x))) \tag{3.10}$$

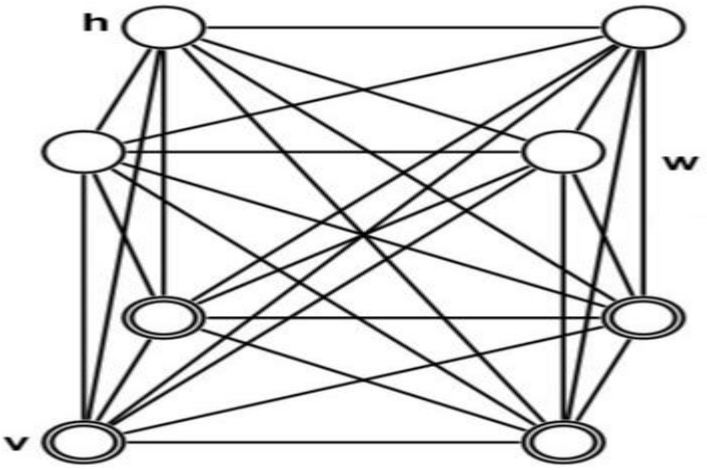
f_i , i.katmanın gizyazar fonksiyonudur.

Gizçözer fonksiyonu (Meng *et al.* 2017):

$$\tilde{x} = g_l (\dots g_i (\dots g_1(y)) \tag{3.11}$$

g_i, i.katmanın gizçözer fonksiyonudur.

3.3 Boltzmann Makinesi ve Kısıtlı Boltzmann Makinesi



Şekil 3.14 Boltzmann makinesi (Salakhutdinov *et al.* 2009)

Bir Boltzmann Makinesi (BM), sinir hücresi gibi birimlerin açık ya da kapalı olup olmayacağına dair stokastik kararlar veren simetrik olarak bağlanmış bir ağdır. Boltzmann makineleri, ikili vektörlerden oluşan veri kümelerindeki ilgili özellikleri keşfetmelerini sağlayan basit bir öğrenme algoritmasına sahiptir. Bu ağ, yalnızca yerel olarak mevcut bilgileri kullanmaktadır. Ağırlık değişimi, değişimin kapsamlı mutlak bir ölçümü optimize etmesine rağmen, sadece bağlandığı iki birimin durumuna bağlıdır (Ackley *et al*. 1985). Ağ değişimi mutlak'ta olmasına rağmen stokastik bir yöntem olduğu için aslında yerel'dir. Bütün birimler birbirine bağlıdır ve tekrarlı bir yapıya sahiptir. İkili 1 ya da 0 olmak üzere olası stokastik sinirlerden oluşurlar. Boltzman makinesinin Hinton (2007)'e göre stokastik ifadeleri aşağıdaki gibidir:

$$z_i = b_i + \sum_j s_j w_{ij} \tag{3.12}$$

b_i, giriş katmanından girilen sapma değerleridir. w_{ij}, i ve j sinirleri arasındaki ağırlık, s_j, sembolü sinirlerin durum değerini göstermektedir. Sinir ya açık ya da kapalı durumdadır. Bu gösterim ikili (0/1) değer ile ifade edilir. z, normalizasyon sabiti olarak tanımlanmasının yanında olasılık toplamının 1 olmasını sağlayan ayırma fonksiyonudur. Gizli ve görünür birimlerin aktivasyonları, sinirler birbirine çift yönlü bağlandığından

birbirinden bağımsızdır. j birimi açıksa s_j 1, aksi takdirde 0 olur. Hinton (2007)'e göre i birimi için olasılık:

$$p\,(\,s_i = 1\,) = \frac{1}{1+ e^{-z_i}} \tag{3.13}$$

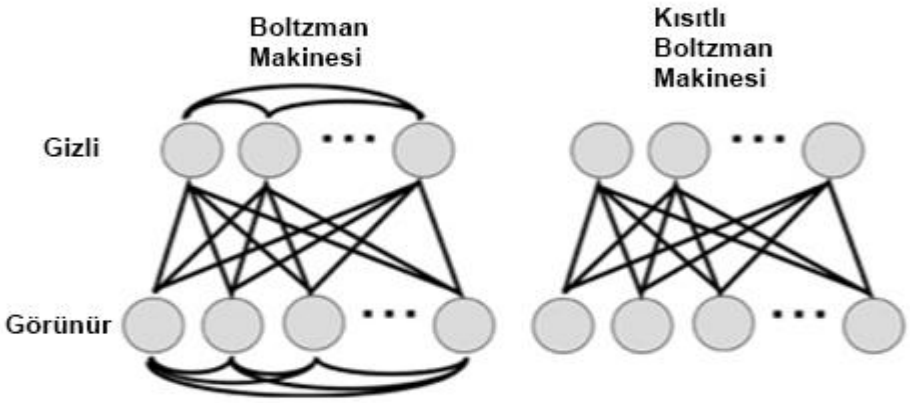
şeklindedir. Giriş değerlerinden bağımsız bir şekilde bütün hücrelerin değerleri sırasıyla güncellenirse, toplam girdiler ağ sonunda bir durum vektörü v'nin olasılığı, tüm olası ikili durum vektörlerinin enerjilerine göre sadece bu durum vektörünün enerjisi ile belirlendiği bir Boltzmann dağılımına yani denge durumuna ulaşacaktır (Hinton 2007):

$$P\,(\,v\,) = \frac{e^{-E(v)}}{\sum_u e^{-E(u)}} \tag{3.14}$$

Hopfield ağlarında olduğu gibi yalnız stokastik olmasından dolayı tam Hopfield gibi değildir, v durum vektörünün Hinton (2007)'e göre enerjisi:

$$E\,(\,v\,) = -\sum_i s_i{}^v b_i - \sum_{i<j} s_i{}^v s_j{}^v w_{ij} \tag{3.15}$$

$s_i{}^v$, i birimine v vektörü tarafından atanan ikili durumdur.



Şekil 3.15 Kısıtlı boltzmann makinesi (O'Connor *et al.* 2013)

Boltzman makinelerinde bir çıkış katmanı olmadığı, görünür ve gizli katmanlardaki sinirlerin birbirine bağlanmasından kaynaklı oluşan bir takım kısıtlardan dolayı Hinton (2010) tarafından Kısıtlı Boltzman Makinesi (KBM) adında daha iyi bir mimari geliştirilmiştir. KBM'de sinirler BM'de olduğu gibi birbirine tamamen bağlı değildir.

Giriş ve çıkış katmanındaki birimlerin sayısında herhangi bir sınırlandırma bulunmamaktadır. Görünür ve gizli birimlere ait katmanlarda her katmana ilişkin birimlerin birbirleri arasında herhangi bir bağ bulunmamaktadır. Görünür birimler gizli birimlerle döngüsel bağlantı yapmaktadır ve her bağlantı bir ağırlık değerine sahiptir. Bu bağlantılar arasındaki ortak atamanın kalitesi enerji fonksiyonuyla ölçülür. Çalışmaya göre eğitim denetimli ya da denetimsiz olarak yapılabilir. Görünür ve gizli birimlerin ortak yapılandırması (v, h) Krizhevsky and Hinton (2009)'a göre aşağıda verilen bir enerjiye sahiptir:

$$E\ (\ v, h\ ) = -\ \sum_{i=1}^{V} \sum_{j=1}^{H} v_i\ h_j\ w_{ij}\ -\ \sum_{i=1}^{V} v_i\ {b_i}^{v}\ -\ \sum_{j=1}^{H} h_j\ {b_j}^{h} \qquad (3.16)$$

v, görünür birimlerin ikili durum vektörüdür, h, gizli birimlerin ikili durum vektörüdür, v_i, i. görünür birimin durumu, h_j, j. gizli birimin durumu, w_{ij}, görünür birim i ile gizli birim j arasındaki ağırlık değeri, ${b_i}^{v}$,i görünür birimindeki sapma değeri, ${b_j}^{h}$, j gizli birimindeki sapma değeridir.

v ve h görünür ve gizli birimlerdeki düğümlerin kesişen vektör değerlerini ifade etmektedir. Bu değişkenler sapma için, yani veri içindeki genel eğilimi saptamaları için modele konulmuştur. Ayrıca $w_i\ h_j\ w_{ij}$ terimi, h üzerinden v'ler arasındaki bağlantıdır ve KBM'de h katmanındaki bağlantıları paylaşmaktadır. Bu h üzerinden bağlantı zorunluluğu KBM'nin özetleme alanını azaltarak genelleme oluşturmasını sağlamaktadır. Gizli tabakaların kendi aralarında ve görünen tabakalarında kendi aralarında doğrudan bağlantıya izin verilmemiştir ve bağlantılara, w üzerinden sadece gizli ve görünen tabakalar arasında izin verilmiştir. Bu yüzden Kısıtlı Boltzmann Makineleri olarak adlandırılmaktadır. Bu ayrıca çalışmada matematiksel olarak bazı kolaylıklar sağlamaktadır.

KBM olasılığı, aşağıdaki şekilde (v, h) konfigürasyonuyla ilişkilidir (Krizhevsky and Hinton 2009):

$$p\ (\ v\ , h\ ) = \frac{e^{-E\ (v,h)}}{\sum_u \sum_g e^{-E(u,g)}} \qquad (3.17)$$

Payda z olarak ifade edilen ayırma fonksiyonu, normalizasyon sabitidir. $h,\ v$ değişkenleri olasılık teorisinden bilinen rasgele değişkenlerdir. Ayrıca, KBM'ler bir olasılık yoğunluk fonksiyonu üzerinden tanımlanırlar. Tüm mümkün değerleri üzerinden integralleri ya da toplamları alınınca sonuç 1 olur. Sezgisel olarak, düşük enerjili yapılandırmalar yüksek olasılık ve yüksek enerjili yapılandırmalar da düşük olasılıklara sahiptir. Paydadaki toplam, olası tüm görünür ve gizli yapılandırmaların üzerinden alınır ve bu nedenle, birim sayısı büyük olduğunda hesaplanması son derece zordur (Krizhevsky and Hinton 2009).

Belirli bir görünür v yapılandırmasının olasılığı:

$$p\,(\,v\,) = \sum_g p(v,g) \qquad (3.18)$$

$$= \frac{\sum_g e^{-E(v,g)}}{\sum_u \sum_g e^{-E(u,g)}}$$

şeklinde verilir. KBM eğitimi çok yüksek bir seviyede, v görünür birimleri istenen bir yapılandırmada sabitlemekten ve daha sonra $p(v)$ büyük olacak şekilde parametrelerin (ağırlık w ve sapma b) ayarlarını bulmaktan oluşur. Beklenti, modelin verilerden anlamlı özellikler çıkarıp genelleştirmek için gizli birimleri kullanmasıdır ve bu nedenle $p(u)$, v ile aynı dağılımdan elde edilen u için de büyük olacaktır (Krizhevsky and Hinton 2009).

Denklem (3.17)'nin gerektirdiği $p(h)$ 'nin denklemi tamamen $p(v)$' nin ki ile aynıdır. Buradan yeni denklem türetilir (Krizhevsky and Hinton 2009):

$$p\,(\,h\,) = \frac{\sum_u e^{-E(u,h)}}{\sum_u \sum_g e^{-E(u,g)}} \qquad (3.19)$$

Bazı temel koşullu olasılıklar (Krizhevsky and Hinton 2009):

$$p\,(\,v \mid h\,) = \frac{p\,(\,v,h\,)}{p\,(\,h\,)} \qquad (3.20)$$

$$= \frac{e^{-E(v,h)}}{\sum_u e^{-E(u,h)}}$$

$$p\,(\,h \mid v\,) = \frac{p\,(\,v,h\,)}{p\,(\,v\,)} \qquad (3.21)$$

$$= \frac{e^{-E(v,h)}}{\sum_g e^{-E(v,g)}}$$

şeklindedir. Ayrıca $p\,(v_k = 1 \mid h)$ için belirli bir görünür birimin olasılığı gizli bir yapılandırmada verilebilir. Bunun gösterimi (Krizhevsky and Hinton 2009):

$$p\,(v_k = 1 \mid h) = \frac{p\,(v_k = 1 \mid h)}{p(h)} \tag{3.22}$$

$$= \frac{\sum_{v_{i\neq k}} p\,(v_k = 1, v_{i\neq k}, h)}{p(h)}$$

$$= \frac{\frac{\sum_{v_{i\neq k}} e^{-E\,(v_k = 1, v_{i\neq k}, h)}}{\sum_u \sum_g e^{-E(u,g)}}}{\frac{\sum_u e^{-E(u,h)}}{\sum_u \sum_g e^{-E(u,g)}}}$$

$$= \frac{\sum_{v_{i\neq k}} e^{-E(v_k = 1, v_{i\neq k}, h)}}{\sum_u e^{-E(u,h)}}$$

$$= \frac{\sum_{v_{i\neq k}} e^{(\sum_{j=1}^{H} h_j w_{kj} + b_k{}^v) + (\sum_{i\neq k}^{V} \sum_{j=1}^{H} v_i h_j w_{ij} + \sum_{i\neq k}^{V} v_i\, b_i{}^v + \sum_{j=1}^{H} h_j\, b_j{}^h)}}{\sum_u e^{-E(u,h)}}$$

$$= \frac{\left[e^{(\sum_{j=1}^{H} h_j w_{kj} + b_k{}^v)}\right] \sum_{v_{i\neq k}} e^{(\sum_{i\neq k}^{V} \sum_{j=1}^{H} v_i h_j w_{ij} + \sum_{i\neq k}^{V} v_i\, b_i{}^v + \sum_{j=1}^{H} h_j\, b_j{}^h)}}{\sum_u e^{-E(u,h)}}$$

$$= \frac{\left[e^{(\sum_{j=1}^{H} h_j w_{kj} + b_k{}^v)}\right] \sum_{v_{i\neq k}} e^{-E(\,v_k = 0, v_{i\neq k}, h)}}{\sum_u e^{-E(u,h)}}$$

$$= \frac{\left[e^{(\sum_{j=1}^{H} h_j w_{kj} + b_k{}^v)}\right] \sum_{v_{i\neq k}} e^{-E(\,v_k = 0, v_{i\neq k}, h)}}{\sum_{u_{i\neq k}} e^{-E(u_k = 1, u_{i\neq k}, h)} + \sum_{u_{i\neq k}} e^{-E(u_k = 0, u_{i\neq k}, h)}}$$

$$= \frac{\left[e^{(\sum_{j=1}^{H} h_j w_{kj} + b_k{}^v)}\right] \sum_{v_{i\neq k}} e^{-E(\,v_k = 0, v_{i\neq k}, h)}}{\left[e^{(\sum_{j=1}^{H} h_j w_{kj} + b_k{}^v)}\right] \sum_{u_{i\neq k}} e^{-E(u_k = 1, u_{i\neq k}, h)} + \sum_{u_{i\neq k}} e^{-E(u_k = 0, u_{i\neq k}, h)}}$$

$$= \frac{1}{1 + e^{-(\sum_{j=1}^{H} h_j w_{kj} + b_k{}^v)}}\,.$$

Aksi halde:

$$p\,(\,hk = 1 \mid v\,) = \frac{1}{1 + e^{-(\sum_{i=1}^{V} v_i w_{ik} + b_k{}^h)}} \tag{3.23}$$

Şekil 3.15'de görünür bir birimin olasılığının, gizli birimler göz önüne alındığında diğer görünür birimlerden bağımsız olduğu görülmektedir. Aynı şekilde, gizli birimler görünür birimler göz önüne alındığında birbirlerinden bağımsızdır. KBM'lerin bu özelliği, tüm gizli birimleri aynı anda ve sonra da tüm görünür birimleri aynı anda örnekleyebildiği için örneklemeyi son derece verimli kılar (Krizhevsky and Hinton 2009).

Denklem 3.23 ile ilgili daha detaylı bilgi EK 1'de verilmiştir.

3.3.1 Kbm' lerin Eğitimi

Buradaki hedef $\{v^c \mid c \in \{1,\ldots,C\}\}$, bir dizi C eğitim durumu göz önüne alındığında, modelin dağılımı altındaki ortalama log olasılığını en üst düzeye çıkarmaktır (Krizhevsky and Hinton 2009):

$$\sum_{c=1}^{C} \log p(v^c) = \sum_{c=1}^{C} \log \frac{\sum_g e^{-E(v^C,g)}}{\sum_u \sum_g e^{-E(u,g)}} \tag{3.24}$$

Bu işlem w_{ij}, ağırlığıyla türevlenebilir olarak gradyan inişi yöntemiyle yapılmaya çalışılır,

$$\begin{aligned} \frac{\partial}{\partial w_{ij}} \sum_{c=1}^{C} \log p(v^c) &= \frac{\partial}{\partial w_{ij}} \sum_{c=1}^{C} \log \frac{\sum_g e^{-E(v^C,g)}}{\sum_u \sum_g e^{-E(u,g)}} \\ &= \frac{\partial}{\partial w_{ij}} \left(\sum_{c=1}^{C} \log \sum_g e^{-E(v^C,g)} - \log \sum_u \sum_g e^{-E(u,g)} \right) \end{aligned} \tag{3.25}$$

İlk önce ilk terime göre işlem yapılır,

$$\begin{aligned} \frac{\partial}{\partial w_{ij}} \sum_{c=1}^{C} \log \sum_g e^{-E(v^C,g)} &= \sum_{c=1}^{C} \frac{\frac{\partial}{\partial w_{ij}} \sum_g e^{-E(v^C,g)}}{\sum_g e^{-E(v^C,g)}} \\ &= - \sum_{c=1}^{C} \frac{\frac{\partial}{\partial w_{ij}} \sum_g e^{-E(v^C,g)} \frac{\partial E(v^C,g)}{\partial w_{ij}}}{\sum_g e^{-E(v^C,g)}} \\ &= - \sum_{c=1}^{C} \frac{\frac{\partial}{\partial w_{ij}} \sum_g e^{-E(v^C,g)} v_i{}^c g_j}{\sum_g e^{-E(v^C,g)}} \end{aligned} \tag{3.26}$$

$\frac{\partial}{\partial w_{ij}} \sum_g e^{-E(v^c,g)} v_i{}^c g_j$ denklemi v için v^c vektörüne bağlandığı göz önüne alındığında $v_i{}^c g_j$'nin beklenen değeridir. $v_i{}^c$ bilinmektedir ve g_j'nin beklenen değeri denklem 3.23'ten hesaplanabilmektedir.

İkinci terime göre işlem yapılırsa:

$$\frac{\partial}{\partial w_{ij}} \sum_{c=1}^{C} \log \sum_u \sum_g e^{-E(u,g)} = \sum_{c=1}^{C} \frac{\frac{\partial}{\partial w_{ij}} \sum_u \sum_g e^{-E(u,g)}}{\sum_u \sum_g e^{-E(u,g)}} \tag{3.27}$$

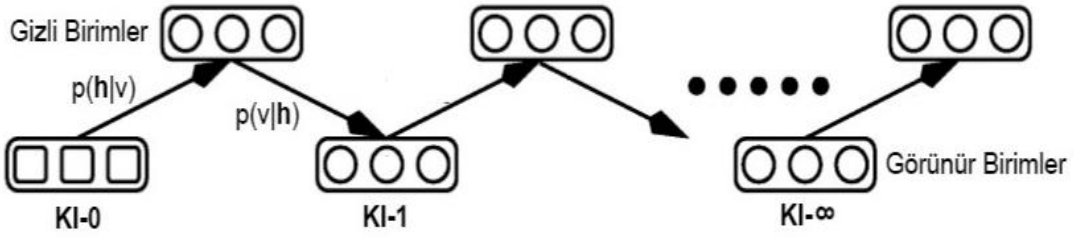
$$= - \sum_{c=1}^{C} \frac{\sum_u \sum_g e^{-E(u,g)} \frac{\partial E(u,g)}{\partial w_{ij}}}{\sum_u \sum_g e^{-E(u,g)}}$$

$$= - \sum_{c=1}^{C} \frac{\sum_u \sum_g e^{-E(u,g)} u_i g_j}{\sum_u \sum_g e^{-E(u,g)}} .$$

$\frac{\sum_u \sum_g e^{-E(u,g)} u_i g_j}{\sum_u \sum_g e^{-E(u,g)}}$ denklemi $u_i g_j$'nin dağılım modeli altındaki beklenen değeridir. v^c vektöründeki görünür birimler tarafından $u_i g_j$ hesaplanabilir. Sonra gizli birimler ondan sonra da görünür birim örnekleri hesaplanır. Bu döngü tekrarlanarak devam eder. Bununla birlikte, bu beklenti Karşıtsal Iraksama (KI) olarak bilinen bir süreçle sınırlı bir sürede yaklaşık olarak tahmin edilebilir (Krizhevsky and Hinton 2009). Tüm bağlantılara ilişkin ağırlığın, görünür ve gizli sapmanın log olasılık gradyanları detaylı bir şekilde EK 2'de verilmiştir.

3.4 Karşıtsal Iraksama

Karşıtsal Iraksama (KI)'da görünür ya da gizli birimler ikili durumdan oluştuğundan tüm görünür ya da gizli birimler kümesi kendi değerleriyle veri vektörünü oluşturur. KI aşamalarında süreç veri alanı olarak adlandırılır ve her veri alanı için enerji fonksiyonu tanımlanarak BM ile bir olasılık dağılımına dönüştürülür.



Şekil 3.16 Karşıtsal ıraksama (Hinton 2002).

Herhangi iki gizli birim ve iki görünür birim bağımsız olduklarından öncelikle tüm gizli birimler ve ardından tüm görünür birimler için bunları birer birer güncelleyerek hesaplamak yerine, bir adımda tüm gizli birimler ve bir adımda tüm görünür birimler güncellenebilir. Böylece, durum alanının tüm birimleri arasında tam bir döngü oluşur. KI eğitimi için $KI-0$ durumuyla başlanır $KI-1$ durumunu bulmak için, bir birim seçer ve o birim için koşullu olasılığı, diğer tüm birimlerin geçerli değerine bağlı olarak hesaplar. Bu ise $p(h|v)$ olasılığına bağlıdır. Daha sonra üniteyi $p(v|h)$ olasılığı 1 olacak şekilde ayarlayıp bunu tekrarlayarak bir sonraki üniteye doğru $KI-1$'den $KI-2$'ye hesaplanarak devam edilir. Böylece mümkün olan tüm gizli ve görünür birimler gezilmiş olur.

Şekil 3.16'da gösterildiği gibi KI öğrenme yordamı, örnekleme zincirini sadece birkaç adım çalıştırarak denklem (3.27)'yi yaklaşık olarak hesaplar. Gizli birimleri $N+1$ kez örnekleyen algoritma $KI-N$ olarak adlandırılır. Uygulamada, yeterli sonuçlar verdiği için çoğunlukla $KI-1$ kullanılmaktadır. $KI-N$ öğrenme sürecinde $E_{model}[v_i h_j]$ tahmini için eğitim görünür birimlerden başlatılır ve daha sonra gizli ve görünür birimler dönüşümlü olarak sırasıyla örneklendirilir. $(N+1)$. örneklemin tahmininde $v_i h_j$'nin gözlem değeri kullanılır (Krizhevsky and Hinton 2009).

Ağırlık w_{ij} için güncelleme kuralı (Krizhevsky and Hinton 2009):

$$\Delta w_{ij} = \alpha_w \left(E_{veri}\left[v_i h_j\right] - E_{model}\left[v_i h_j\right]\right) \tag{3.28}$$

α_w, ağırlık öğrenme oranı hiperparemetresidir, E_{veri}, görünür birimler veriye bağlandığında E_{model} ise görünür birimler veriden ayrıldığında model dağılımı altındaki beklentidir. E_{model} , KI kullanılarak hesaplanır.

Sapma için güncelleme kuralları benzer şekildedir (Krizhevsky and Hinton 2009):

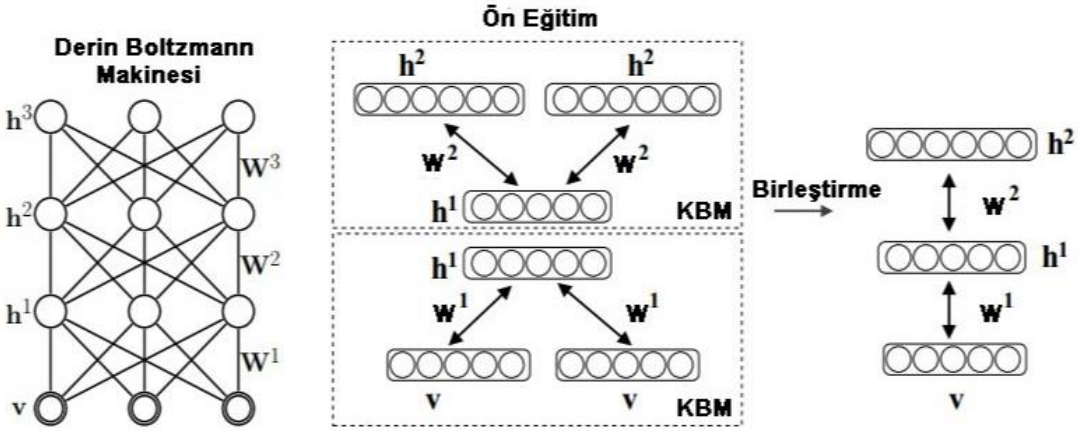
$$\Delta b_i^{\,v} = \alpha_{b^v} \ (E_{veri}[v_i] - E_{model}[v_i]) \tag{3.29}$$

$$\Delta b_i^{\,h} = \alpha_{b^h} \ (E_{veri}[h_j] - E_{model}[h_j]). \tag{3.30}$$

Denklem 3.28, 3.29 ve 3.30 ile ilgili detaylı bilgi Ek 2'de verilmiştir.

3.5 Derin İnanç Ağları

Bir KBM, eğim kaybı sorununa sahiptir. Bununla birlikte, birkaç yığınlı KBM ve bir sınıflandırıcıdan oluşan kombinasyonu sayesinde, sorunu çözebilecek bir sinir ağı oluşturulmuştur ve bu ağ Derin İnanç Ağı (DİA) olarak adlandırılır. Derin yapay sinir ağlarının veri ihtiyacının fazla olması ve buna bağlı olarak eğitimin uzun zaman alması, katman sayısının da fazla olmasından kaynaklı kayıp fonksiyonlarının gradyanları sıfıra yakınsar. Bundan dolayı eğim kaybı problemi ortaya çıkar. Sinir ağlarının gradyanları, geriye doğru hesaplamayla bulunmaktadır. Geriye doğru hesaplama ağın türevlerini, katmanı son katmandan ilk katmana hareket ettirerek bulur. Zincir kuralı ile her bir katmanın türevleri, ilk katmanların türevlerini hesaplamak için son katmandan ilk katmana çarpılır. Bununla birlikte, n tane gizli katmanda sigmoid işlevi gibi bir aktivasyon kullandığında, n tane küçük türevler ile birlikte çarpılır. Bu nedenle, ilk katmanlara doğru ilerlerken gradyan katlanarak azalır. Küçük bir gradyan, ilk katmanların ağırlıklarının ve sapmalarının her eğitimde etkili bir şekilde güncellenemeyeceği anlamına gelir. Bu ilk katmanlar, girdi verilerinin ana unsurlarını tanımada önemli olduğundan, tüm ağın genel olarak yeterli derecede eğitilmemesine yol açabilir.



Şekil 3.17 Derin boltzmann makinesi (Elaraby *et al.* 2016)

Derin inanç ağları, KBM mimarisini, l katmanındaki ağırlıkların, alt katmanlardaki tüm ağırlıkları sabit tutarak ve $l-1$ katmanındaki gizli birimlerin faaliyetlerini veri alarak eğitildiği çoklu gizli katmanlara genişletir. Böylece DİA eğitim algoritması katmanlarını sırayla eğitir. Katman l, katman $l-1$'den sonra eğitilir. İlk gizli katmanın boyutu ikinci gizli katmanın boyutuyla aynı yapılır ve ikincisinin ağırlıkları ilk ağırlıklarından başlatılırsa ilk gizli katmanın ağırlıklarını sabit tutarken ikinci gizli katmanın eğitilmesi, verilerin logaritmik olasılığını arttırır (Krizhevsky and Hinton 2009).

DİA eğitim adımları aşağıdaki gibidir:

- İlk KBM girişi mümkün olduğu kadar doğru bir şekilde yeniden yapılandırmak için eğitilmiştir.
- Daha sonra ilk KBM'nin gizli katmanı ikinci KBM'nin görünür katmanı olarak kabul edilir.
- İlk KBM çıkışıyla ikinci KBM eğitilir.
- Bu süreç ağdaki her katman eğitilene kadar tekrarlanır.

Jensen'in eşitsizliği ile yukarıdan aşağıya bir üretici model olarak DİA (Krizhevsky and Hinton 2009):

$$log\ p(v\ |W_1, W_2\ ) \geq \sum_{h_1} q(\ h_1 \mid v\ ) \log \frac{p\ (v, h_1 \mid W_1, W_2\ )}{q\ (h_1 \mid v\ )}$$

$q(\ h_1 \mid v\ )$'nin herhangi bir dağılımı için özellikle $q(\ h_1 \mid v\ )= p\ (h_1 \mid v\ , W_1)$ olduğunda,

$$log\ p(v\ |W_1, W_2\ ) \geq \sum_{h_1} p(h_1 \mid v\ , W_1\ ) \log \frac{p\ (v, h_1 \mid W_1, W_2\ )}{p\ (h_1 \mid v\ , W_1\ )} \tag{3.31}$$

$$= \sum_{h_1} p(h_1 \mid v\ , W_1\ ) \log \frac{p\ (v, h_1 \mid W_1\ )\ p\ (h_1 \mid W_2\ )}{p\ (h_1 \mid v\ , W_1\ )}$$

$$= \sum_{h_1} p(\ h_1 \mid v\ , W_1\ ) \log\ p(h_1 \mid W_2) + \sum_{h_1} p(\ h_1 \mid v\ , W_1\ ) log \frac{p(\ v \mid h_1\ , W_1)}{p(h_1 \mid v, W_1)}$$

$$= \sum_{h_1} p(\ h_1 \mid v\ , W_1\ ) \log p(h_1 \mid W_2) + \sum_{h_1} p(\ h_1 \mid v\ , W_1\ ) log \frac{p(\ v \mid W_1)}{p(h_1 \mid W_1)}$$

$$= \sum_{h_1} p(\ h_1 \mid v\ , W_1\ ) \log p(h_1 \mid W_2) - \sum_{h_1} p(\ h_1 \mid v\ , W_1\ ) log\ p(h_1 \mid W_1) +$$

$$\log p(v \mid W_1).$$

W_2, W_1 ile başlatılırsa, ilk iki terim iptal olur ve dolaysıyla sınır sıkılaşır. W_1'i sabit tutarak DİA'nın ikinci katmanını eğitmek denklem (3.32)'yi maksimize etmeye denktir. Bu da denklem (3.31)'deki ilk terimdir.

$$\sum_{h_1} p(\ h_1 \mid v\ , W_1\ ) \log p(h_1 \mid W_2) \tag{3.32}$$

Diğer terimler sabit kaldığından, ikinci gizli katmanı eğittiğimizde verinin log olasılığı üzerindeki sınırı arttırırız. Başlangıçta sınır sıkı olduğu için, biz aynı zamanda verinin log olasılığını da arttırmış oluruz. W_2 ve W_1'i öğrendikten sonra W_3'ün başlangıç değerini atadığımızda sınırın sıkı olmadığını da dikkate alarak bu çıkarsamayı daha fazla katmana genişletebiliriz. Bu nedenle, verinin o anki log olasılığını azaltırken sınırı artırabiliriz. (Krizhevsky and Hinton 2009).

3.6 Gauss-Bernoulli Kbm'leri

Bayesçi bir yaklaşımda üst parametrelere karşılık gelen hücrelerin olduğu ağaç benzeri bir yapı olan KBM, birinci düzey parametrelerin önsellerini tanımlayan ikinci düzey parametreleri de içerir. Burada önsellik KBM'de açıklanan ilk katmandır. Önsel olarak gerçek görünür birimler kullanılır ve $KI-N$'deki aşamalarla eğitimini yapar. Bu da başarıyı artırmaktadır.

Görünür birimlerin yalnızca ikili değerleri alabileceği bir KBM, piksel yoğunlukları gibi gerçek değerli verileri modellemek için çok elverişsizdir. Gerçek değerli verileri modellemek için, modelin enerji fonksiyonu kullanılır (Krizhevsky and Hinton 2009):

$$E\ (\ v,h\ ) = \sum_{i=1}^{V} \frac{(v_i - b_i^{\,v})^2}{2\ {\sigma_i}^2} - \sum_{j=1}^{H} {b_j}^{h} h_j - \sum_{i=1}^{V}\sum_{j=1}^{H} \frac{v_i}{\sigma_i}\ h_j\ w_{ij}. \tag{3.33}$$

Enerji fonksiyonu denklem (3.33) göz önüne alındığında, Krizhevsky ve Hinton (2009)'a göre $p\ (v \mid h)$ dağılımını şu şekilde türetilebilir:

$$p\ (v \mid h) = \frac{e^{-E(v,h)}}{\int_u e^{-E(u,h)}du} \tag{3.34}$$

$$= \frac{e^{-\sum_{i=1}^{V}\frac{(v_i - b_i^{\,v})^2}{2\ {\sigma_i}^2} - \sum_{j=1}^{H} {b_j}^{h} h_j - \sum_{i=1}^{V}\sum_{j=1}^{H}\frac{v_i}{\sigma_i} h_j\, w_{ij}}}{\int_u e^{-\sum_{i=1}^{V}\frac{(u_i - b_i^{\,v})^2}{2\ {\sigma_i}^2} - \sum_{j=1}^{H} {b_j}^{h} h_j - \sum_{i=1}^{V}\sum_{j=1}^{H}\frac{u_i}{\sigma_i} h_j\, w_{ij}}du}$$

$$= \frac{e^{-\sum_{i=1}^{V}\frac{(v_i - b_i^{\,v})^2}{2\ {\sigma_i}^2} - \sum_{j=1}^{H} {b_j}^{h} h_j - \sum_{i=1}^{V}\sum_{j=1}^{H}\frac{v_i}{\sigma_i} h_j\, w_{ij}}}{\prod_{i=1}^{V}[e^{\frac{1}{2}\cdot\left(\sum_{j=1}^{H} h_j\, w_{ij}\right)^2 + \sum_{j=1}^{H} {b_j}^{h} h_j + \frac{1}{\sigma_i} {b_i}^{v} \sum_{j=1}^{H} h_j\, w_{ij}}.\sigma_i\sqrt{2\pi}]}$$

$$= \frac{\prod_{i=1}^{V} e^{-\frac{(v_i - b_i^{\,v})^2}{2\ {\sigma_i}^2} + \sum_{j=1}^{H} {b_j}^{h} h_j + \frac{v_i}{\sigma_i}\sum_{j=1}^{H} h_j\, w_{ij}}}{\prod_{i=1}^{V}[e^{\frac{1}{2}\cdot\left(\sum_{j=1}^{H} h_j\, w_{ij}\right)^2 + \sum_{j=1}^{H} {b_j}^{h} h_j + \frac{1}{\sigma_i} {b_i}^{v} \sum_{j=1}^{H} h_j\, w_{ij}}.\sigma_i\sqrt{2\pi}]}$$

$$= \prod_{i=1}^{V} \frac{1}{\sigma_i\sqrt{2\pi}} \cdot e^{-\frac{(v_i - b_i^{\,v})^2}{2\ {\sigma_i}^2} - \frac{1}{2}\left(\sum_{j=1}^{H} {b_j}^{h} h_j\right)^2 + \frac{1}{\sigma_i}(v_i - b_i^{\,v})\sum_{j=1}^{H} h_j\, w_{ij}}$$

$$= \prod_{i=1}^{V} \frac{1}{\sigma_i\sqrt{2\pi}} \cdot e^{-\frac{1}{2\ {\sigma_i}^2}\left((v_i - b_i^{\,v})^2 + {\sigma_i}^2\left(\sum_{j=1}^{H} h_j\, w_{ij}\right)^2 - 2\sigma_i(v_i - b_i^{\,v})\left(\sum_{j=1}^{H} h_j\, w_{ij}\right)\right)}$$

$$= \prod_{i=1}^{V} \frac{1}{\sigma_i \sqrt{2\pi}} \cdot e^{-\frac{1}{2{\sigma_i}^2}\left(v_i - {b_i}^v - \sigma_i \sum_{j=1}^{H} h_j\, w_{ij}\right)^2},$$

şeklinde elde edilen dağılım $\sum$ köşegen kovaryans matrisi ve μ_i ortalamasıyla V-boyutlu normal dağılımdır, Burada

$$\sum = \begin{bmatrix} {\sigma_1}^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & {\sigma_v}^2 \end{bmatrix}$$

$$\mu_i = {b_i}^v + \sigma_i \sum_{j=1}^{H} h_j\, w_{ij} \tag{3.35}$$

şeklinde tanımlanır (Krizhevsky and Hinton 2009).

$p\,(\,h_k = 1 \mid v\,)$'in hesaplanması aşağıda verilmiştir (Krizhevsky and Hinton 2009):

$$p\,(\,h_k = 1 \mid v\,) = \frac{\sum_{h_{j\neq k}} p(v, h_k = 1, h_{j\neq k},)}{p(v)} \tag{3.36}$$

$$= \frac{\sum_{h_{j\neq k}} e^{-E(v, h_k = 1, h_{j\neq k})}}{\sum_g e^{-E(v,g)}}$$

$$= \frac{\sum_{h_{j\neq k}} e^{\left(\sum_{i=1}^{V} \frac{v_i}{\sigma_i} w_{ij} + {b_k}^h\right) + \left(\sum_{i=1}^{V} \sum_{h_{j\neq k}}^{H} \frac{v_i}{\sigma_i} h_j w_{ij} + \sum_{i=1}^{V} \frac{(v_i - {b_i}^v)^2}{2{\sigma_i}^2} + \sum_{h_{j\neq k}}^{H} h_j {b_j}^h\right)}}{\sum_g e^{-E(v,g)}}$$

$$= \frac{e^{\left(\sum_{i=1}^{V} \frac{v_i}{\sigma_i} w_{ik} + {b_k}^h\right)} \sum_{h_{j\neq k}} e^{\left(\sum_{i=1}^{V} \frac{v_i}{\sigma_i} w_{ij} + {b_k}^h\right) + \left(\sum_{i=1}^{V} \sum_{h_{j\neq k}}^{H} \frac{v_i}{\sigma_i} h_j w_{ij} + \sum_{i=1}^{V} \frac{(v_i - {b_i}^v)^2}{2{\sigma_i}^2} + \sum_{h_{j\neq k}}^{H} h_j {b_j}^h\right)}}{\sum_g e^{-E(v,g)}}$$

$$= \frac{e^{\left(\sum_{i=1}^{V} \frac{v_i}{\sigma_i} w_{ik} + {b_k}^h\right)} \sum_{h_{j\neq k}} e^{-E(v, h_k = 0, h_{j\neq k})}}{\sum_{g_{j\neq k}} e^{-E(v, g_k = 0, g)} + \sum_{g_{j\neq k}} e^{-E(v, g_k = 1, g)}}$$

$$= \frac{e^{\left(\sum_{i=1}^{V} \frac{v_i}{\sigma_i} w_{ik} + {b_k}^h\right)} \sum_{h_{j\neq k}} e^{-E(v, h_k = 0, h_{j\neq k})}}{\sum_{g_{j\neq k}} e^{-E(v, g_k = 0, g)} + e^{\left(\sum_{i=1}^{V} \frac{v_i}{\sigma_i} w_{ij} + {b_k}^h\right)} \sum_{g_{j\neq k}} e^{-E(v, g_k = 0, g)}}$$

$$= \frac{e^{\left(\sum_{i=1}^{V} \frac{v_i}{\sigma_i} w_{ik} + {b_k}^h\right)}}{1 + e^{\left(\sum_{i=1}^{V} \frac{v_i}{\sigma_i} w_{ij} + {b_k}^h\right)}}$$

$$= \frac{1}{1 + e^{-\left(\sum_{i=1}^{V} \frac{v_i}{\sigma_i} w_{ij} + {b_k}^h\right)}}$$

Denklem (3.34) ve (3.36) ile ilgili daha detaylı bilgi Ek 3 ve Ek 4'te verilmiştir.

3.6.1 Gauss-Bernoulli Kbm'lerinin Eğitimi

Gauss-Bernoulli KBM için eğitim süreci sıradan bir KBM ile aynıdır. Bu durumda, (3.25)'te gösterilen denklemin türevi alınmaktadır (Krizhevsky and Hinton 2009):

$$\frac{\partial}{\partial w_{ij}} \sum_{c=1}^{C} \log \sum_{g/} e^{-E(v^c,g)} = -\sum_{c=1}^{C} \frac{\sum_g e^{-E(v^c,g)} \frac{\partial E(v^c,g)}{\partial w_{ij}}}{\sum_g e^{-E(v^c,g)}} \tag{3.37}$$

$$= -\frac{1}{\sigma_i} \sum_{c=1}^{C} \frac{\sum_g e^{-E(v^c,g)} v_i{}^c h_j{}^c}{\sum_g e^{-E(v^c,g)}} .$$

Ve benzer şekilde:

$$\frac{\partial}{\partial w_{ij}} \sum_{c=1}^{C} \log \sum_u \sum_g e^{-E(u,g)} = -\frac{1}{\sigma_i} \sum_{c=1}^{C} \frac{\sum_u \sum_g e^{-E(u,g)} u_i g_j}{\sum_u \sum_g e^{-E(u,g)}} \tag{3.38}$$

tahmin KI kullanarak yapılmaktadır.

3.6.2 Görünür Varyansların Eğitimi

Bir Gauss-Bernoulli KBM'nin enerji fonksiyonu denklem (3.33)'den veri vektörlerinin log olasılığı maksimize edilmeye çalışılmaktadır (Krizhevsky and Hinton 2009):

$$\log p(v) = \log \frac{\sum_h e^{-E(v,h)}}{\int_u \sum_g e^{-E(u,g)} du.} \tag{3.39}$$

$$= \log \sum_h e^{-E(v,h)} - \log \int_u \sum_g e^{-E(u,g)} du.$$

İlk terim, modelin v vektörüne atadığı serbest enerjinin negatifidir ve aşağıdaki gibi genişletilir (Krizhevsky and Hinton 2009):

$$
\begin{aligned}
-F(v) &= \log \sum_{h} e^{-E(v,h)} \\
&= \log \sum_{h} e^{\sum_{i=1}^{V} \frac{(v_i - {b_i}^v)^2}{2{\sigma_i}^2} + \sum_{j=1}^{H} {b_j}^h h_j + \sum_{i=1}^{V} \sum_{j=1}^{H} \frac{v_i}{\sigma_i} h_j w_{ij}} \\
&= \log(e^{-\sum_{i=1}^{V} \frac{(v_i - {b_i}^v)^2}{2{\sigma_i}^2}} \sum_{h} e^{\sum_{j=1}^{H} {b_j}^h h_j + \sum_{i=1}^{V} \sum_{j=1}^{H} \frac{v_i}{\sigma_i} h_j w_{ij}}) \\
&= -\sum_{i=1}^{V} \frac{(vi - bi^v)^2}{2{\sigma_i}^2} + \log \sum_{h} e^{\sum_{j=1}^{H} {b_j}^h h_j + \sum_{i=1}^{V} \sum_{j=1}^{H} \frac{v_i}{\sigma_i} h_j w_{ij}} \\
&= -\sum_{i=1}^{V} \frac{(v_i - {b_i}^v)^2}{2{\sigma_i}^2} + \log \sum_{h} e^{\sum_{j=1}^{H} h_j ({b_j}^h + \sum_{j=1}^{V} \frac{v_i}{\sigma_i} w_{ij})} \\
&= -\sum_{i=1}^{V} \frac{(v_i - {b_i}^v)^2}{2{\sigma_i}^2} + \log \sum_{h} \prod_{j=1}^{H} e^{h_j ({b_j}^h + \sum_{j=1}^{V} \frac{v_i}{\sigma_i} w_{ij})} \\
&= -\sum_{i=1}^{V} \frac{(v_i - {b_i}^v)^2}{2{\sigma_i}^2} + \log \prod_{j=1}^{H} (1 + e^{{b_j}^h + \sum_{j=1}^{V} \frac{v_i}{\sigma_i} w_{ij}}) \\
&= -\sum_{i=1}^{V} \frac{(v_i - {b_i}^v)^2}{2{\sigma_i}^2} + \sum_{j=1}^{H} \log\left(1 + e^{{b_j}^h + \sum_{j=1}^{V} \frac{v_i}{\sigma_i} w_{ij}}\right).
\end{aligned}
\tag{3.40}
$$

Sondan ikinci adım, her bir h_j'nin gerçekte 0 veya 1 olmasıyla doğrulanır. σ_i'ye göre $-F(v)$'in türevi (Krizhevsky and Hinton 2009):

$$
\begin{aligned}
\frac{\partial(-F(v))}{\partial \sigma_i} &= \frac{(v_i - {b_i}^v)^2}{{\sigma_i}^3} + \frac{\delta}{\delta \sigma_i} \sum_{j=1}^{H} \log(1 + e^{{b_j}^h + \sum_{i=1}^{V} \frac{v_i}{\sigma_i} w_{ij}}) \\
&= \frac{(v_i - {b_i}^v)^2}{{\sigma_i}^3} - \sum_{j=1}^{H} \frac{e^{{b_j}^h + \sum_{j=1}^{V} \frac{v_i}{\sigma_i} wij}}{1 + e^{{b_j}^h + \sum_{i=1}^{V} \frac{v_i}{\sigma_i} wij}} \cdot \frac{w_{ij} v_i}{{\sigma_i}^2} \\
&= \frac{(v_i - {b_i}^v)^2}{{\sigma_i}^3} - \sum_{j=1}^{H} \frac{1}{1 + e^{-{b_j}^h - \sum_{i=1}^{V} \frac{vi}{\sigma_i} wij}} \cdot \frac{w_{ij} v_i}{{\sigma_i}^2} \\
&= \frac{(v_i - {b_i}^v)^2}{{\sigma_i}^3} - \sum_{j=1}^{H} a_j \cdot \frac{w_{ij} v_i}{{\sigma_i}^2} \cdot
\end{aligned}
\tag{3.41}
$$

a_j, v görünür vektörü verildiğinde j. gizli birimin gerçel değerli deterministik aktivasyon olasılığıdır.

Aynı şekilde, $\log \int_u \sum_g e^{-E(u,g)} du$'un türevi (Krizhevsky and Hinton 2009):

$$\frac{\partial}{\partial \sigma_i} \log \int_u \sum_g e^{-E(u,g)} du = \frac{\int_u \sum_g e^{-E(u,g)} (\frac{(v_i - {b_i}^v)^2}{{\sigma_i}^3} - \sum_{j=1}^{H} gj \cdot \frac{w_{ij} u_i}{{\sigma_i}^2}) du}{\int_u \sum_g e^{-E(u,g)} du} \quad (3.42)$$

Model dağılımı altındaki beklenen değer (Krizhevsky and Hinton 2009):

$$\frac{(v_i - {b_i}^v)^2}{{\sigma_i}^3} - \sum_{j=1}^{H} hj \cdot \frac{w_{ij} u_i}{{\sigma_i}^2} \quad (3.43)$$

Bu nedenle, σ_i için güncelleme kuralı (Krizhevsky and Hinton 2009):

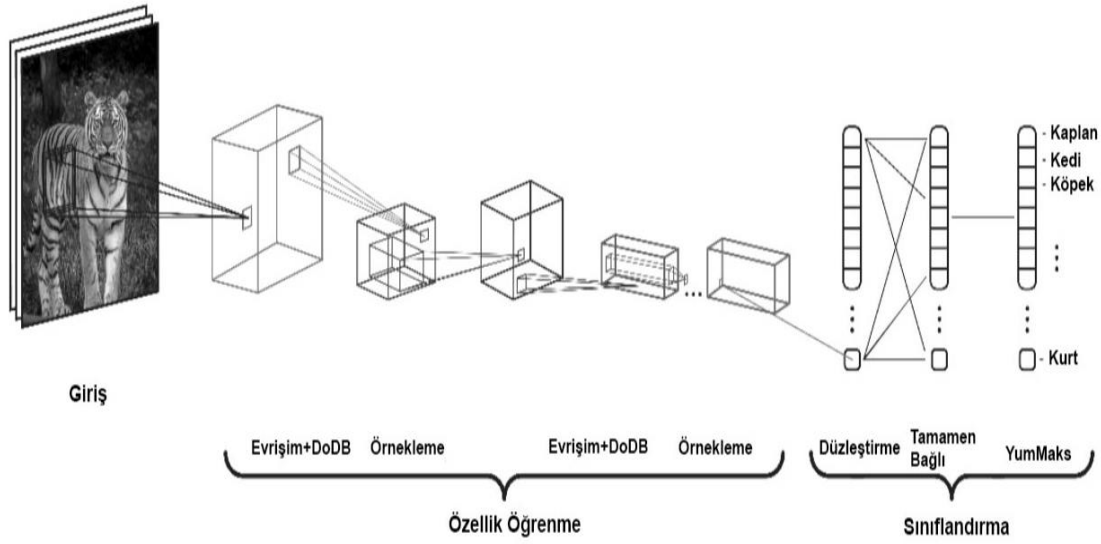
$$\Delta\sigma_i = \alpha_\sigma . \left(E_{veri} \left[\frac{\left(v_i - {b_i}^v\right)^2}{{\sigma_i}^3} - \sum_{j=1}^{H} h_j \cdot \frac{w_{ij} v_i}{{\sigma_i}^2} \right] - E_{model} \left[\frac{\left(v_i - {b_i}^v\right)^2}{{\sigma_i}^3} - \sum_{j=1}^{H} h_j \cdot \frac{w_{ij} v_i}{{\sigma_i}^2} \right] \right) \quad (3.44)$$

α_σ, öğrenme oranı hiperparametresidir. Ağırlıklar için modelin beklentisi altındaki tahminin de $KI - 1$ kullanılmaktadır. Ağırlık, görünür sapma, gizli sapma, denklem (3.41) ve (3.44) ile ilgili daha detaylı bilgi Ek 5'te verilmiştir.

3.7 Evrişimli Sinir Ağları

Evrişimli Sinir Ağları (ESA), görüntü sınıflandırma, nesne bulma, doğal dil işleme gibi birçok alanda problemlerin çözümü için kullanılan YSA tabanlı bir yöntemdir ve yaygın olarak görüntü sınıflandırmada kullanılmaktadır. Görüntü sınıflandırma, görüntünün girdi olarak alınmasının ardından ilgili görüntünün hangi görüntü sınıfına ait olduğu (kaplan, kedi, köpek vb. gibi) ya da ilgili görüntü için onu en iyi tanımlayan sınıfların olasılıklarının çıkartılması işlevidir. Bu durum canlılarda özellikle de insanlar için, doğumdan itibaren başlayıp öğrenilen ilk becerileridir. İnsan çok uzunca bir süre düşünmeden tek seferde bulunulan alanı, alandaki canlı ya da cansız varlıkları veya cisimleri problemsiz ve hızlı bir şekilde tanımlar. Bir görüntü gördüğünde ya da alana

baktığında bilinçli bir şekilde farkında dahi olmadan çoğu zaman görüntünün ayırıcı özelliğini ortaya çıkarır ve her objeyi tanımlar.

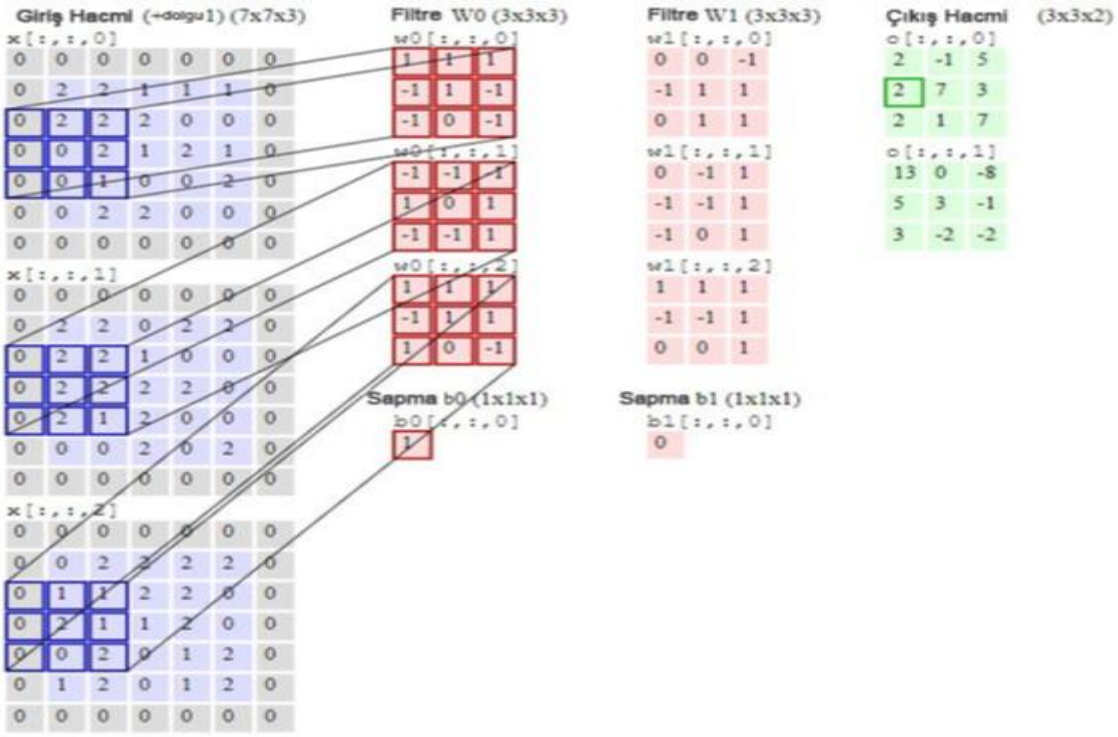


Şekil 3.18 Evrişimli sinir ağı (Sewak *et al.* 2018)

Bilgisayarda görüntüyü girdi olarak alır ve görüntünün çözünürlüğüne, boyutuna bağlı olarak bir dizi piksel değerleri görür. Ardından görüntü boyutuna ve çözünürlüğüne bağlı olarak bilgisayar bir takım sayı dizisi oluşturur. Görüntüye ilişkin boyutlarıyla üç ana renk Kırmızı Yeşil Mavi (KYM), örneğin $32 \times 32 \times 3$ gibi ifade edilebilir. Burada ki 3 sayısı 3 ana rengi temsil eden KYM değeridir. Örneğin, 480×480 boyutunda bir renkli resmin temsili dizisi $480 \times 480 \times 3$ piksel değerinde oluşacaktır. Piksel değerleri ise 0'dan 255'e kadar o noktadaki piksel yoğunluğunu temsil etmektedir. Bu değerler, görüntü sınıflandırmasında bilgisayar için girdilerdir. Bilgisayarın bu sayı dizisini alıp görüntünün belli bir sınıf, örneğin kaplan için 0.45, kedi için 0.80, köpek için 0.30 olması olabilirliğini ifade eden olasılıklar bilgisayarın çıktılarıdır ve böylelikle bu sayılar bir anlam ifade etmektedir. Bilgisayardan yapılması istenilen şey görüntülere ait özellikleri bulup tüm görüntüleri ayırt etmesidir. İnsan için bilinçaltında da faaliyet gösteren aşamalar budur. Bir kaplanın görüntüsüne bakıldığında, kaplan görüntüsü için onu ayırt edici pençeleri veya kürkü gibi belirlenebilir özellikleri varsa, bu bir kaplan olarak sınıflandırılabilir. Aynı şekilde bilgisayar da kenar ve eğriler gibi alt düzeyde özellikleri arayıp daha sonra bir dizi evrişim katmanı yoluyla görüntü sınıflandırmasını gerçekleştirmektedir. Bu ESA'nın genel bir yapısıdır.

3.7.1 Evrişim Katmanı

ESA'daki ilk katman her zaman evrişim katmanıdır. Evrişim katmanı farklı boyutlarda düzenlenebilir. Çalışma renkli görüntülerde yapılıyorsa üç ana renk kırmızı, yeşil ve mavi (KYM) renkli üç kanaldan meydana gelmektedir. Bu durumda evrişim üç kanal için yapılır ve bu da üç boyutta düzenlenmiş sinir hücrelerinden oluşmaktadır. Bunlar genişlik, yükseklik ve derinliktir. Burada derinlik aktivasyon hacminin üçüncü boyutudur. Herhangi bir görüntü veri kümesi için giriş görüntüleri bir giriş hacmini ifade ederse aktivasyon hacmi örneğin $32 \times 32 \times 3$ boyutundadır ve bu değerler sırasıyla genişlik, yükseklik ve derinliği ifade etmektedir. Derinlik üç ana renkten oluşmaktadır. Bu katmanda ilk önce giriş görüntüsü için bir filtre uygulanır. Bu filtre örneğin $5 \times 5 \times 3$ 'lük bir filtreyse, bu giriş görüntüsünün sol üst köşesinden başlayarak ilk önce adım adım sağa doğru görüntünün sağ sonuna kadar ve daha sonra bir alt kademeden aynı işlemi tekrar ederek tüm görüntüyü tarar. Burada derinlik boyutu giriş görüntüsüyle aynıdır. Giriş görüntüsü ve filtre sürekli geri yayılımla güncellenen ağırlıklar matrisidir. Örneğin $5 \times 5 \times 3$ 'lük filtre uygulanırsa burada filtre $5 \times 5 \times 3 = 75$ tane ağırlığa ve 1 tane sapmaya sahip olacaktır. Her bir filtreye aynı zamanda bir tane skaler bir sapma eklenir.



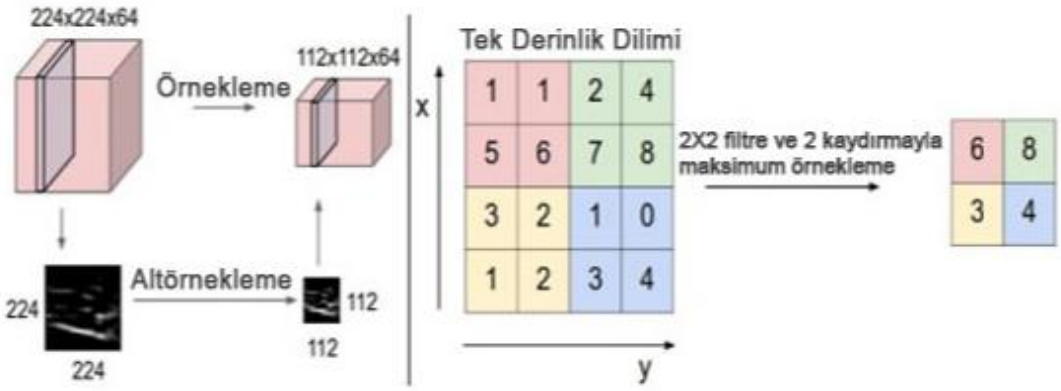
Şekil 3.19 Evrişim katmanı (İnt.Kyn.8)

G_1 genişliğinde, Y_1 yüksekliğinde ve D_1 derinliğine sahip bir giriş görüntüsü $G_1 \times Y_1 \times D_1$ olsun. Filtrelerin sayısı K, filtrenin mekansal boyutu (giriş görüntüsüyle aynıdır) F, kaydırma S, kenar dolgu miktarı P olduğunda çıkış görüntüsünün büyüklüğü $G_2 \times Y_2 \times D_2$ olur. Burada, $G_2 = \frac{G_1 - F + 2P}{S} + 1$, $Y_2 = \frac{Y_1 - F + 2P}{S} + 1$ (Genişlik ve yükseklik simetrik olarak eşit hesaplanır.) ve $D_2 = K$'dır (İnt.Kyn.8).

Parametre paylaşımı ile her bir filtre $F \times F \times D_1$ adet ağırlık üretir. Toplamda $(F \times F \times D_1) \times K$ adet ağırlık ve K adet sapma oluşur. Giriş hacmi üzerinde S kaydırmalı ve $d.$ sapma kadar ofsetli $d.$ filtre ile gerçekleştirilen geçerli bir evrişimin sonucu olarak çıkış hacminde $G_2 \times Y_2$ büyüklüğünde $d.$ derinlik dilimi oluşur. En fazla kullanılan parametreler $F = 3, S = 1$ ve $P = 1$'dir (İnt.Kyn.8).

3.7.2 Örnekleme Katmanı

Bu katman, evrişimli ağdaki art arda gelen evrişim katmanları arasına eklenen bir katmandır. Bu katmanın işlevi, ağdaki parametreleri ve görüntünün kayma boyutunu azaltarak hesaplamaların daha kısa sürede daha kolay yapılmasını sağlamasıdır. Maksimum, minimum, ortalama örnekleme olmak üzere birçok örnekleme türü vardır ve evrişimli ağlarda en çok maksimum örnekleme kullanılmaktadır.



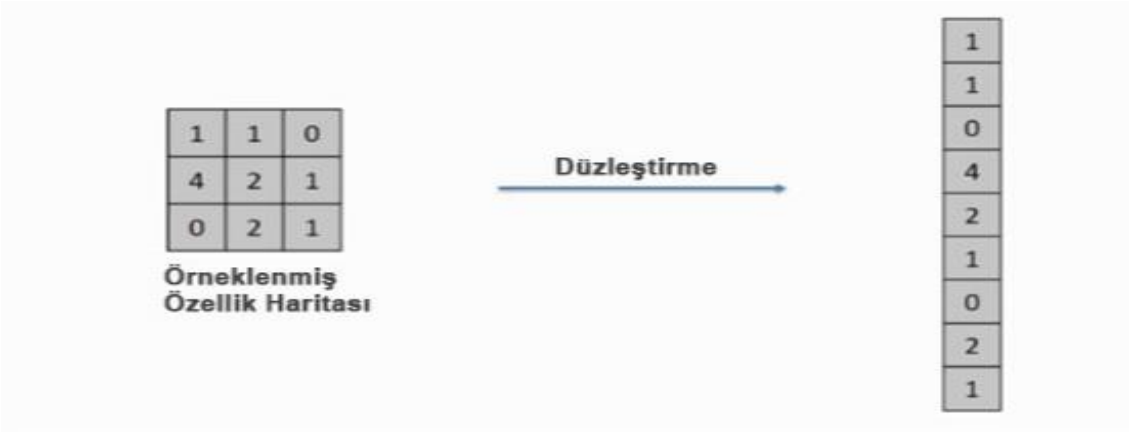
Şekil 3.20 Örnekleme katmanı (İnt.Kyn.8)

Örnekleme katmanına giren görüntü G_1 genişliğinde, Y_1 yüksekliğinde ve D_1 derinliğinde $G_1 \times Y_1 \times D_1$ görüntü büyüklüğüne sahip olsun. Filtrenin mekansal boyutu (giriş

görüntüsüyle aynıdır) F, kaydırma S olduğunda örnekleme katmanının çıkış görüntüsü $G_2 \times Y_2 \times D_2$ olur. Burada, $G_2 = \frac{G_1 - F}{S} + 1$, $Y_2 = \frac{Y_1 - F}{S} + 1$ (Genişlik ve yükseklik simetrik olarak eşit hesaplanır.) ve $D_2 = D_1$'dir. Uygulamalarda en fazla kullanılan iki parametre çifti; $F = 3, S = 2$ ve $F = 2, S = 2$'dir (İnt.Kyn.8).

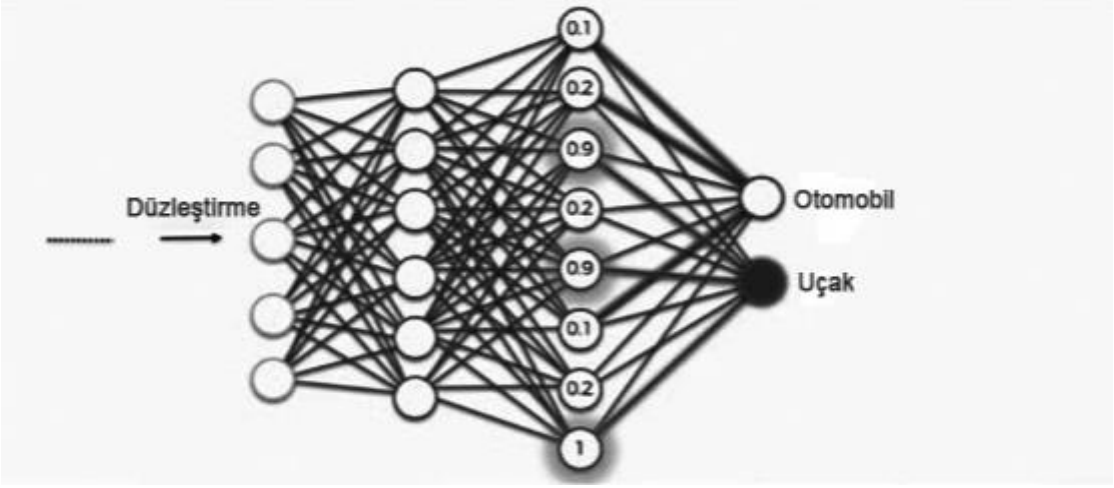
3.7.3 Düzleştirme Katmanı

Bu katman, son katman olan Tamamen Bağlı (TB) katmana verileri hazırlamak amacıyla kullanılır. Verilerin tek boyutlu bir diziden alınması ihtiyacından dolayı evrişim ve örnekleme katmanından gelen matrisleri tek boyutlu diziye dönüştürür.



Şekil 3.21 Düzleştirme katmanı (İnt.Kyn.9)

3.7.4 Tamamen Bağlı Katman



Şekil 3.22 Tamamen bağlı katman (İnt.Kyn.10)

Bu katmanda çıktı üretilmeden önce sınıflara ilişkin herhangi bir görüntü, hücreleri üzerindeki ağırlıklarıyla o görüntüyü en iyi tanımlayan görüntünün özelliği ya da özellikleri yüksek değerli ağırlıklara sahip olacaktır. Örneğin otomobil ve uçaktan oluşan ikili bir görüntü sınıflandırma problemi üzerinde çalışılsın. TB katmanına kadar tüm süreç işledikten sonra TB katmanında hücreler uçağa özgü belirli bir özelliği tespit edecektir. Bunun uçağın kanatları olduğu varsayılırsa bu kanatlara ilişkin ağırlıklar yüksek değerlere sahip olacaktır. Ardından her iki sınıf için bu özellik kontrol edilip kanatların uçakla ilgisinin olup olmadığına karar verilir. Kanat, uçağın hücrelerinde yüksek değere sahip olduğunda bu kanadın uçakla ilişkili olduğu anlamına gelmektedir. Bu bilgi her iki sınıf için hesaplandığından kanat, otomobil için düşük ağırlık değerlerine sahip olacaktır ve TB, bu özelliğin otomobile özgü bir özellik olmadığına karar verecektir. Aynı işlemler otomobil içinde binlerce kez tekrar edecek ve otomobile özgü özellikler öğrenilecektir.

Ağın en sonunda çıkışları her sınıfın olasılık değerlerine dönüştürmek için aktivasyon fonksiyonlarından çoğunlukla YumMaks fonksiyonu kullanılmaktadır. Bu katman temelde ön katmanlarda hesaplanan bir girdi hacmi alır. Çıktı hacmi ise örneklerin atanacağı toplam sınıf sayısına (N) eşit olur. Örneğin, 10 basamaklı bir sınıflandırma probleminde 10 tane basamak olduğu için $N = 10$ olur. Bu N boyutlu vektördeki her değer belirli bir sınıfın olasılığını temsil eder. Örneğin, bir görüntü sınıflandırma problemi için görüntü sınıflarına ilişkin sonuç vektörü $[0.15\ \ 0.15\ \ 0.1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0.60\ \ 0]$ şeklinde bir çıktıya sahip olursa, bu sonuç vektörü çıktı sınıflarına ilişkin ilgili görüntünün 1. ve 2. sınıfa $\%\,15$ olasılıkla, 3.sınıfa $\%10$ olasılıkla ve 8.sınıfa $\%60$ olasılıkla ilgili görüntünün o sınıflara benzediğini göstermektedir.

4. BULGULAR

Temel derin öğrenme ile Bayesçi derin öğrenme yöntemlerinin karşılaştırılması için cifar-10 veri kümesi üzerinde bir uygulama yapılmıştır. Burada kullanılan cifar-10 veri kümesi, her sınıf başına 6000 resim içeren 10 sınıftan, toplamda 60000, 32x32 boyutlu renkli görüntülerden oluşmaktadır. Bunların 50000'i eğitim ve 10000'i test görüntüsüdür. Veri kümesi, her biri 10000 görüntü içeren bir test ve beş eğitim grubuna ayrılmıştır. Test grubu her sınıftan tam olarak rassal seçilen 1000 görüntü içermektedir. Eğitim grupları, kalan görüntüleri rassal sırayla içermekte olup bazı eğitim grupları bir sınıftan diğerine göre daha fazla görüntü içerebilmektedir. Genel olarak eğitim grupları her sınıftan tam 5000 görüntü içermektedir. Sınıflar birbirinden tamamen ayrıktır. Otomobiller ve kamyonlar arasında çakışma yoktur. Örneğin; "Otomobil", sedanlar, suv'lar ve bu türe benzer araçları içerir. "Kamyon" sadece büyük kamyonları içerir, pikapları içermez (İnt.Kyn.11).

4.1 Temel Evrişimli Derin Öğrenme Uygulaması

Uygulamada cifar-10 veri kümesiyle ilk olarak YSA tabanlı görüntü sınıflandırma problemi için temel evrişimli derin öğrenme tekniği kullanılmıştır. Veri kümesinden 10 sınıfa ayrılmış ilgili yükseklik, genişlik ve derinlikte $32 \times 32 \times 3$ boyutunda 50000 adet eğitim görüntüsünün yüklenmesiyle başlamıştır. Ardından evrişimli derin öğrenmede yapılacak ilk işlem eğitim veri kümesiyle giriş katmanını oluşturmaktır. Bu yüzden $32 \times 32 \times 3$ boyutunda cifar-10 görüntüleri için görüntü giriş katmanı oluşturulmuştur. Ardından evrişim katmanı parametreleri olan Evrişim Filtresi, DoDB, Maksimum Örnekleme için katmanlar ilgili boyut, dolgu, kaydırma gibi kendi özellikleriyle oluşturulmuştur. Ardından evrişimli derin öğrenmenin görüntü sınıflarıyla ilgili çıktıların hesap ve yorumlanması açısından çıkış katmanı Tamamen Bağlı, DoDB, YumMaks, Sınıflandırma Çıktısı gibi katmanlar ilgili sayıda oluşturularak eğitim için evrişimli ağ mimarisi hazır hale getirilmiştir. Ağ mimarisinin son hali Çizelge 4.3'te verilmiştir. Uygulama görüntü sınıflandırma probleminden oluştuğundan Grafik İşlem Birimi (GİB) kullanılmıştır. Maksimum tur sayısı, momentum, öğrenme oranı, sönümleme, L2 düzeltmesi gibi öğrenme parametrelerinin başlangıç değerleri verilmiştir. Ardından

eğitim yapılıp modelin başarısını yani ne kadar doğru bir şekilde görüntüleri sınıflandırabildiğini görebilmek için test veri kümesi modele sunularak ağın başarısı hesaplanmıştır.

Çizelge 4.1 Evrişim katmanındaki iç katmanların gösterimi

İç Katmanlar	Katmanlar	Boyut	Dolgu (piksel)	Kaydırma
1	1-Evrişim Filtresi	$5 \times 5 \times 3$	2	1×1
	2-DoDB			
	3-Maksimum Örnekleme	3×3		2×2
2	4-Evrişim Filtresi	$5 \times 5 \times 3$	2	1×1
	5-DoDB			
	6-Maksimum Örnekleme	3×3		2×2
3	7-Evrişim Filtresi	$5 \times 5 \times 3$	2	1×1
	8-DoDB			
	9-Maksimum Örnekleme	3×3		2×2

İlk evrişim katmanı için $5 \times 5 \times 3$'lük filtre oluşturulmuştur. Görüntünün sınırlarının bilinilebilirliliğinin daha iyi sağlanabilmesi için 2 piksellik simetrik dolgu oluşturulmuştur. Bu aynı zamanda görüntü sınır bilgilerinin ağda çok erken kaybının önlenmesi açısından önemlidir. Ardından DoDB katmanı oluşturulmuştur. Daha sonrada 2 piksellik kaydırmayla 3×3'lük boyutta maksimum örnekleme katmanı oluşturulmuştur. Evrişim katmanının tamamlanması için 3 iç katman ağa eklenmiştir.

Çizelge 4.2 Çıkış katmanındaki iç katmanların gösterimi

Sıra	Katmanlar	Sayısı
1	Tamamen Bağlı	64
2	DoDB	
3	Tamemen Bağlı	10
4	YumMaks	
5	Sınıflandırma Çıktısı	

Çıkış katmanında öncelikle 64 hücreden oluşan bir TB katmanı sonra DoDB katmanı ve son olarak da 10 sınıflı çıktı birimine sahip bir TB katmanı eklenmiştir. Bu noktada ağ, giriş görüntüsünün hangi sınıfa ait olup olmadığını ölçmek için kullanılabilecek 10 sinyal üretmektedir. Ardından YumMaks katmanı ve sınıflandırma katmanı oluşturulmuştur. Son katmanlar, görüntü sınıfları üzerindeki kategorik olasılık dağılımını hesaplamak için

TB katmanının çıktısını kullanmaktadır.

Çizelge 4.3 Evrişimli derin ağ mimarisinin son hali

Sıra	Katmanlar	Boyut	Dolgu (piksel)	Kaydırma
1	Giriş Görüntüsü	$32 \times 32 \times 3$		
2	Evrişim Filtresi	$5 \times 5 \times 3$	2	1×1
3	DoDB			
4	Maksimum Örnekleme	3×3		2×2
5	Evrişim Filtresi	$5 \times 5 \times 3$	2	1×1
6	DoDB			
7	Maksimum Örnekleme	3×3		2×2
8	Evrişim Filtresi	$5 \times 5 \times 3$	2	1×1
9	DoDB			
10	Maksimum Örnekleme	3×3		2×2
11	Tamamen Bağlı			
12	DoDB			
13	Tamamen Bağlı			
14	Yummaks			
15	Sınıflandırma Çıktısı			

Daha sonra ağın öğrenme parametrelerinin başlangıç değerleri öğrenme oranı 0.002, momentum 0.9, L2 düzeltmesi 0.004, sönümleme 0.1 olarak verilmiştir. Maksimum tur sayısı 20 alınarak eğitime başlanmıştır.

Çizelge 4.4 Temel evrişimli derin ağın eğitim süreci

Tur	Geçen Süre (Saniye)	Kayıp Fonksiyonun Değeri	Doğruluk Oranı	Öğrenme Oranı
1	6.91	2.3026	%8.59	0.0020
1	11.73	2.3017	%13.28	0.0020
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
13	423.97	0.3698	%85.94	0.0002
13	428.41	0.3300	%89.84	0.0002
.	.	.	.	.
.	.	.	.	.
20	700.46	0.3621	%85.94	$2e^{-0.5}$

20 turun sonunda yaklaşık 700 saniyede eğitim veri kümesi için %85.94'lük doğru sınıflandırma oranı elde edilmiştir. Eğitim süresince kayıp fonksiyonun ve öğrenme oranının değerleri çizelge 4.4'te verilmiştir. Burada daha hızlı ve tutarlı bir optimizasyon için öğrenme oranı adımsal olarak düşürülerek mutlak ya da yerel minimuma ulaşılmaya çalışılmaktadır.

Modelin başarısını yani ne kadar doğru bir şekilde görüntüleri sınıflandırabildiğini görebilmek için test veri kümesi modele sunularak test kümesindeki görüntülerin sınıfları belirlenmiştir. Temel evrişimli derin ağ test görüntülerini %75.88 oranında doğru olarak sınıflandırmıştır.

4.2 Bayesçi Derin Öğrenme

Temel evrişimli derin öğrenmede ağın mimarisi oluşturulurken ağ derinliğinin, eğitim sürecinde ise başlangıç öğrenme oranı, momentum katsayısı ve L2 düzeltmesi gibi parametrelerin belirlenmesi gerekir. Bu nedenle, Bayesçi optimizasyon temel derin öğrenme üzerinde uygulanmış ve oluşturulan ağ için en uygun parametre değerleri belirlenmiştir. Daha sonra bu parametre değerleri kullanılarak ağın eğitimi gerçekleştirilmiştir. Eğitim ve test görüntüleri sınıflarıyla birlikte cifar-10 veri kümesinden yüklenmiştir. Ayrıca ağın doğrulamasını etkinleştirmek için, eğitim görüntülerinden rassal olarak **5000** görüntü seçilmiş ve bu görüntülerle doğrulama veri kümesi oluşturulmuştur. Bayesçi optimizasyonda sabit sayıda tur için eğitim yapılmıştır ve son turlarda öğrenme oranı adımsal olarak değiştirilmiştir. Bu, parametre güncellemelerinin gürültüsünü azaltır ve ağ parametrelerinin kayıp fonksiyonunu en aza indirgenmesini sağlar.

Veri artırımı kullanılarak eğitim görüntüleri dikey eksen boyunca rassal çevrilmiştir ve bunlara rassal olarak yatay ve dikey dört piksel boyutunda çevirme işlemi yapılmıştır. Veri artırma işlemi ağın eğitim görüntülerinin tam detaylarını ezberlemesini ve aşırı öğrenmeyi önlemesi için kullanılan bir yöntemdir.

4.2.1 Bayesçi Optimizasyon için Değişkenlerin Seçimi

Bayesçi optimizasyon kullanarak optimize edilecek değişkenlerin ve aralıklarının belirlenmesi gerekmektedir. Aşağıdaki değişkenler optimize edilmiştir:

- Ağ bölüm derinliği ağın derinliğini kontrol eder. Ağ, her biri özdeş evrişim katmanları olan üç bölüme sahiptir. Dolaysıyla toplam evrişim katman sayısı $3 \times bölümderinliği$'dir.
- İlk öğrenme oranı, en iyi öğrenme oranı, verilere ve eğitilen ağa bağlı olarak değişebilir.
- Daha düzgün parametre güncellemeleri ve stokastik gradyan inişine özgü gürültünün azalması için momentum kullanılmıştır.
- L2 düzeltmesi, aşırı öğrenmeyi önlemek için kullanılır. Aynı zamanda veri artışı ve küçük veri kümeleri normalizasyonu da ağ düzeltmesine yardımcı olur.

4.2.2 Bayesçi Optimizasyonun Amaç Fonksiyonu

Eğitim ve doğrulama veri kümeleri girdi olarak kullanılarak Bayesçi optimizasyon için amaç fonksiyonu oluşturulmuştur. Amaç fonksiyonu, evrişimli bir sinir ağını eğitir ve doğrulama veri kümesindeki sınıflandırma hatasını döndürür. Bayes optimizasyonu, en iyi modeli seçmek için doğrulama veri kümesindeki hata oranını kullandığından, doğrulama veri kümesindeki son ağda aşırı öğrenme olması olasıdır. Bunu önlemek amacıyla son seçilen model daha sonra genelleme hatasını tahmin etmek için bağımsız test veri kümesinde test edilir. Amaç fonksiyonu aşağıdaki adımları yerine getirir:

- Optimizasyon için sınıflara ilişkin değişken değerleri girdi olarak alınır. Amaç fonksiyonu her sütun adı değişken adına eşit olan bir tablodaki optimizasyon değişkenlerinin geçerli değerleriyle amaç fonksiyonunda tanımlanır.
- Ağ mimarisi ve optimizasyona ilişkin eğitim seçenekleri tanımlanır.
- Ağ eğitilir ve doğrulanır.
- Eğitimli ağ, doğrulama hatası ve optimizasyona ilişkin eğitim seçenekleriyle kaydedilir.
- Doğrulama hatası ve kaydedilen ağ ilgili dosya adıyla geri çağrılır.

Bayesçi optimizasyon sonucunda en iyi parametre değerleri ve tahminleri sırasıyla çizelge 4.5 ve 4.6'da gösterilmiştir.

Çizelge 4.5 Bayesçi optimizasyon ile derin sinir ağı eğitimi parametre sonuçları

Optimizasyon için maksimum süre: 7200 saniye
Toplam fonksiyon değerlendirnesi: 6
Toplam geçen süre: 8845.45 saniye
Toplam amaç fonksiyonu değerlendirme süresi: 8822.52

En iyi gözlenen uygun nokta için:

Ağ Derinliği	Öğrenme Oranı	Momentum	L2 Düzeltmesi
2	0.002693	0.936	$2.8e^{-07}$

Gözlenen amaç fonksiyonu değeri: 0.159
Hesaplanan amaç fonksiyonu değeri: 0.168
Fonksiyon değerlendirme süresi: 1657.81 saniye

Modele göre en iyi tahmini uygun nokta için:

Ağ Derinliği	Öğrenme Oranı	Momentum	L2 Düzeltmesi
2	0.002693	0.936	$2.8e^{-07}$

Tahmin edilen amaç fonksiyonu değeri: 0.168
Tahmin edilen fonksiyon değerlendirme süresi: 1447.62

Doğrulama veri kümesindeki sınıflandırma hatasını en aza indirmek için Bayes optimizasyonunda toplam optimizasyon süresi saniyelerle belirtilmiştir.

Çizelge 4.6 Bayesçi optimizasyon ile derin sinir ağı eğitimi

Yineleme	Değerlendirme Sonuçları	Amaç (Minimum)	Geçen Süre	Şimdiye Kadar En İyi (Gözlem)	Şimdiye Kadar En İyi (Tahmin)	Ağ Derinliği	Başlangıç Öğrenme Oranı	Momentum	L2 Düzeltmesi
1	En iyi	0.193	1483	0.193	0.193	3	0.00116	0.844	$1.7e^{-09}$
2	En iyi	0.159	1657	0.159	0.161	2	0.00269	0.936	$2.8e^{-07}$
3	Onay	0.253	1096	0.159	0.159	1	0.03563	0.858	0.00023
4	Onay	0.171	1309	0.159	0.163	2	0.00786	0.838	0.00227
5	Onay	0.172	1362	0.159	0.168	2	0.00146	0.947	$1.03e^{-10}$
6	Onay	0.170	1912	0.159	0.168	2	0.04443	0.801	0.0012

Her ağ eğitimi tamamladıktan sonra Bayes optimizasyon sonuçları çizelge 4.6'da gösterilmiştir. Amaç fonksiyonu için parametre sayısı ve her yineleme için hesaplama maliyeti hemen hemen aynıdır. Çizelge 4.6'da görüldüğü üzere 6 yinelemede optimizasyon tamamlanmıştır.

Bayesçi optimizasyon sonucunda oluşan ağ mimarisi önceden belirlenen ilgili katman boyutlarıyla aşağıdaki gibidir:

- İlk evrişim katmanlarının giriş ve çıkış boyutları 32×32'dir ve ardından 2×2'lik kaydırmayla maksimum örnekleme katmanı bunu 16×16'ya düşürür.
- İlgili maksimum örneklemeye ilişkin evrişim katmanlarının da giriş ve çıkış boyutları 16×16'dır ve ardından 2×2'lik kaydırmayla bir maksimum örnekleme katmanı bunu 8×8'e düşürür.
- İlgili maksimum örneklemeye ilişkin evrişim katmanlarının da giriş ve çıkış boyutları 8×8'dir.
- TB katmanı ve en sonunda YumMaks ve sınıflandırma çıktısı eklenmiştir.

Bayesçi optimizasyon kullanılarak eğitilen evrişimli derin ağın doğrulama veri kümesi için hata oranı 0.1592, test veri kümesi için hata oranı ise 0.1639 olarak gerçekleşmiştir. Burada ağ $\%83.61$'lik bir başarı göstermektedir.

Veri kümesinden rassal olarak seçilen ve yalnızca o seçilen görüntüye ilişkin tahmin edilen sınıflar ve bu sınıfların olasılıkları ile birlikte örnek test görüntüleri Şekil 4.1'de gösterilmiştir.

Şekil 4.1: Bazı örnek görüntüler için tahmin edilen sınıf olasılıkları

Model dikkate alınarak test veri kümesiyle sınıflandırma matrisi oluşturulmuştur. Sütun ve satır özetleri kullanılarak her sınıf için doğruluk görüntülenmiştir. Sınıflandırma matrisi, sınıflandırma modelinin performansını değerlendirebilmek için tahmin değerleriyle gerçek değerleri karşılaştırıp matris şeklinde görselleştirilerek sınıflar hakkında yorum yapılabilmesini sağlamaktadır ve hem doğru hem de yanlış tahminlerin sayısını göstermesi bakımından yüzdelik olarak hesaplanıp ona göre yorumlanmıştır.

Sınıflandırma Matrisi

	Uçak	Otomobil	Kuş	Kedi	Geyik	Köpek	Kurbağa	At	Gemi	Kamyon
Uçak	**892**	11	39	7	16	1	3	3	19	9
Otomobil	9	**953**	1	0	1	0	3	1	4	28
Kuş	53	2	**782**	19	49	21	48	16	4	6
Kedi	28	7	66	**610**	55	113	75	21	10	15
Geyik	11	1	51	17	**832**	8	50	26	2	2
Köpek	14	2	45	89	36	**746**	28	35	1	4
Kurbağa	9	0	28	26	11	5	**912**	3	3	3
At	16	2	21	17	41	27	7	**865**	1	3
Gemi	69	15	7	1	3	1	7	2	**883**	12
Kamyon	31	58	5	3	0	0	1	5	11	**886**

Sınıfların Doğru Sınıflandırma Oranları

Uçak	Otomobil	Kuş	Kedi	Geyik	Köpek	Kurbağa	At	Gemi	Kamyon
89.2	95.3	78.2	61	83.2	74.6	91.2	86.5	88.3	88.6

Şekil 4.2: Sınıflandırma matrisi

Sınıflandırma matrisinde sınıflar için en yüksek tahmin değerinden en düşüğüne yorumlamaları aşağıdaki gibidir:

- Otomobil için görüntü gerçekte otomobilken %95.3'lük doğrulukla model görüntüyü otomobil olarak tahmin etmiştir.
- Kurbağa için görüntü gerçekte kurbağayken %91.2'lik doğrulukla model görüntüyü kurbağa olarak tahmin etmiştir.
- Uçak için görüntü gerçekte uçakken %89.2'lik doğrulukla model görüntüyü uçak olarak tahmin etmiştir.
- Kamyon için görüntü gerçekte kamyonken %88.6'lık doğrulukla model görüntüyü kamyon olarak tahmin etmiştir.
- Gemi için görüntü gerçekte gemiyken %88.3'lük doğrulukla model görüntüyü gemi olarak tahmin etmiştir.
- At için görüntü gerçekte atken %86.5'lik doğrulukla model görüntüyü at olarak tahmin etmiştir.
- Geyik için görüntü gerçekte geyikken %83.2'lik doğrulukla model görüntüyü geyik olarak tahmin etmiştir.

- Kuş için görüntü gerçekte kuşken %78.2'lik doğrulukla model görüntüyü kuş olarak tahmin etmiştir.
- Köpek için görüntü gerçekte köpekken %74.6'lık doğrulukla model görüntüyü köpek olarak tahmin etmiştir.
- Kedi için görüntü gerçekte kediyken %61'lik doğrulukla model görüntüyü kedi olarak tahmin etmiştir.

5. TARTIŞMA ve SONUÇ

Temel evrişimli derin öğrenmede ağın mimarisi oluşturulurken hangi katmanların kullanılacağı, katmanlardaki filtrelerin sayısı ve boyutu, kaydırma miktarı ve dolgu miktarı gibi parametrelerin belirlenmesi gerekir. Ayrıca, eğitim sürecinde öğrenme oranı, momentum ve L2 düzeltmesi gibi öğrenme parametrelerinin de başlangıç değerlerinin eğitimden önce belirlenmesi gerekir. Bu parametrelerin seçimi ağın performansını ve eğitim süresini etkiler. Bayesçi derin öğrenme bu problemlerin üstesinden gelmek için ağın performans fonksiyonunu modelleyerek ağın performansını bu parametrelere göre optimize etmeye çalışır. Bayesçi optimizasyon optimum parametre değerlerini bulmak için eğitim veri kümesi içerisinden seçilen ayrı bir doğrulama veri kümesi de kullanır. Bu optimizasyon süreci zaman açısından maliyetlidir. Daha sonra, elde edilen optimal parametre değerleri kullanılarak ağ eğitimi gerçekleştirilir.

Bu tür yöntemlere ilişkin çalışmalarda Merkezi İşlem Birimi (MİB) ya da GİB kullanılabilmektedir. Uygulama, görüntü sınıflandırmaya ilişkin bir eğitimden oluşmaktadır. Bilgisayarlar görüntülere ilişkin oluşturulan yüksek paralel yapılı karmaşık algoritmaları MİB'den daha etkin GİB'ler ile çalıştırabiliyor olmasından dolayı uygulamada grafik işlemcisi kullanılmıştır. Ayrıca bu tür derin öğrenme yöntemlerinin MİB ya da GİB'lerin kendi aralarında paralel eğitim yapmasına da izin vermesinden dolayı çalışma süresini etkilemektedir. Aynı veri kümesi üzerinde aynı ve tek bir GİB, nvidia geforce 940mx ile temel evrişimli ve Bayesçi derin öğrenme yöntemi kullanılarak uygulama üzerinde çalışmalar yapılıp sonuçlandırılmıştır. Uygulamaların, eğitim sürecinden sonra ilgili yöntemlere ilişkin model başarıları hesaplanmıştır. Temel evrişimli derin öğrenme test veri kümesindeki örneklerin 75.88'ini doğru olarak sınıflandırırken, Bayesçi derin öğrenmede bu oran 83.61 olarak gerçekleşmiştir. Bayesçi derin öğrenme yöntemi başarı da 7.73'lük bir artış sağlamıştır. Bunu Bayesçi derin öğrenme yöntemi sağlarken optimizasyon ve eğitim süresi diğer yönteme göre yaklaşık 2 saat 30 dakika daha fazladır.

Burada Bayesçi derin öğrenmenin başarı oranını artırdığı gözlense de ağın eğitimi oldukça uzun sürmektedir. Bunu azaltmak için Krizhevsky and Hinton (2009) bu tez

çalışmasında kullanılan yöntem ve buna paralel yöntemleri cifar-10 veri kümesiyle deneyerek bilgisayarda daha az işlemle daha iyi sonuçlar alınıp alınamayacağını test etmiştir. Öncelikle görüntüleri beyazlaştırarak görüntü derinliği bakımından görüntüyü tek boyuta indirgeyip daha az işlemle test veri kümesinde sınıflandırma performansını, beyazlatılmamış görüntülerle yapılan çalışmayla karşılaştırmıştır. Daha sonra beyazlatılmış görüntüler üzerinde KBM eğitimi yapıp bunu beyazlatılmamış görüntülerle yapılan KBM ile karşılaştırmıştır. Ardından 1000 ve 10000 hücreli gizli katmana sahip bir geri yayılım eğitimi yapıp bundan elde ettiği ağırlıkları kullanarak beyazlatılmamış görüntülerde 1 ve ardından 2 gizli katmanlı KBM eğitimi yaparak performanslarını karşılaştırmıştır. Krizhevsky and Hinton (2009)'a göre en yüksek performansı beyazlatılmamış veriler üzerinde eğitilmiş bir KBM tarafından öğrenilen 10000 birimlik özellikle başlatılan gizli bir katmanı olan geri yayılım ağı göstermiştir. Buna göre bilgisayarın daha az işlemle çalışmasını yapabilmesi açısından denenen yöntemler model performansını da düşürmektedir. Onun için bu tez çalışmasında da kullanılan ham görüntülerle çalışmak en iyi sonucu vermektedir.

Aynı zamanda Bayesçi derin öğrenme yöntemindeki optimizasyon sürecinde ağın iç katmanlarının oluşturulma aşamasında ilk evrişim katmanı ve ona bağlı oluşturulan örnekleme ve evrişim katmanlarına ilişkin filtre, dolgu ve kaydırma miktarları ve sınıflandırma aşamasında TB katmanındaki hücrelerin sayısı da en uygun değerleriyle belirlenebilir. Böylece daha yüksek başarı oranları elde edilebilecektir. Bu durum optimizasyon sürecinde zaman açısından artı bir maliyete sebep olacaktır. Bu olumsuz olarak değerlendirilebilir ancak GİB'lerin gelişimi bu durumu da olumlu yönde etkileyecektir.

6. KAYNAKLAR

Ahmed, H. O. A., Wong, M. D. and Nandi, A. K. (2018). Intelligent Condition Monitoring Method for Bearing Faults from Highly Compressed Measurements Using Sparse Over-Complete Features. *Mechanical Systems and Signal Processing*, **99:** 459-477.

Ackley, D. H. Hinton, G. E., and Sejnowski, T. J. (1985). A Learning Algorithm for Boltzmann Machines. *Cognitive Science*, **9(1):** 147-169.

Cortes, C. and Vapnik, V. (1995). Support-Vector Networks. *Machine Learning*, **20(3):** 273-297.

Elaraby, N. M., Elmogy, M. and Barakat, S. (2016). Deep Learning: Effective Tool for Big Data Analytics. *International Journal of Computer Science Engineering (IJCSE)*, **5(05).**

Fausett, L. (1994) Fundamentals of Neural Networks Architectures Algorithms and Applications. , Englewood Cliffs, NJ : Prentice-Hall, 294-296

Glorot, X., Bordes, A. and Bengio, Y. (2011). Deep Sparse Rectifier Neural Networks. In *Proceedings of The Fourteenth International Conference on Artificial Intelligence and Statistics* (pp. 315-323).

Hebb, D. O. (2005). *The Organization of Behavior: A Neuropsychological Theory*. Psychology Press

Hopfield, J. J. (1982). Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proceedings of The National Academy of Sciences*, ***79*(8):** 2554-2558.

Hochreiter, S. (1991). Untersuchungen zu Dynamischen Neuronalen Netzen. *Diploma, Technische Universität München*, **91(1).**

Hinton, G. E. (1986) Learning Distributed Representations of Concepts. In Proceedings of The Eighth Annual Conference of The Cognitive Science Society. Vol. 1, P. 12.

Hinton, G. E. (2002). Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, **14(8):** 1771-1800.

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the Dimensionality of Data with Neural Networks. *Science*, **313(5786):** 504-507.

Hinton, G. E., Osindero, S. and Teh, Y. W. (2006). A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, **18(7):** 1527-1554.

Hinton, G. E (2007) Boltzmann Machines. March 25, (pp. 1-2)

Hinton, G. (2010) A Practical Guide to Training Restricted Boltzmann Machines, Department of Computer Science, University of Toronto

Kendall, A. and Gal, Y. (2017). What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?. In *Advances in Neural Information Processing Systems* (pp. 5574-5584).

Kohonen, T. (1982). Self-Organized Formation of Topologically Correct Feature Maps. *Biological Cybernetics*, **43(1):** 59-60.

Krizhevsky, A. and Hinton, G. (2009). *Learning Multiple Layers of Features From Tiny Images.* Vol. 1, No. 4, p. 7. Technical Report, University of Toronto.

LeCun, Y. A., Bottou, L., Orr, G. B. and Müller, K. R. (2012). Efficient Backprop. In *Neural Networks: Tricks of The Trade*. Springer, Berlin, Heidelberg. 9-48

Maas, A. L., Hannun, A. Y. and Ng, A. Y. (2013). Rectifier Nonlinearities Improve Neural Network Acoustic Models. In *Proc. Icml* (Vol. 30, No. 1, p. 3).

MacKay, David JC. (1992). A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation*, **4(3):** 448-472.

Meng, Q., Catchpoole, D., Skillicom, D. and Kennedy, P. J. (2017). Relational Autoencoder for Feature Extraction. In *2017 International Joint Conference on Neural Networks (IJCNN)* (pp. 364-371). IEEE.

McCulloch, Warren S. and Walter Pitts. (1943) A Logical Calculus of the Ideas Immanent in Nervous Activity. *The Bulletin of Mathematical Biophysics* **5(4):** 115-116.

Minsky, M. and Papert, S. (1969). An İntroduction to Computational Geometry. *Cambridge Tiass., HIT.*

Neal, R. M. (2012). *Bayesian Learning for Neural Networks*. Vol. 118. Springer Science & Business Media.

O'Connor, P., Neil, D., Liu, S. C., Delbruck, T. and Pfeiffer, M. (2013). Real-Time Classification and Sensor Fusion with a Spiking Deep Belief Network. *Frontiers in Neuroscience*, **7:** 178.

Öztemel, E. (2012) Yapay Sinir Ağları [Artificial Neural Networks]. *İstanbul: Papatya Yayıncılık.*

Rosenblatt, F. (1958). The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review*, ***65*(6):** 386.

Ruder, S. (2016). An Overview of Gradient Descent Optimization Algorithms. *Arxiv Preprint Arxiv:1609.04747.*

Rumelhart, D. E., Hinton, G. E. and Williams, R. J. (1988). Learning Representations by Back-Propagating Errors. *Cognitive Modeling*, ***5*(3):** 1.

Salakhutdinov, R. and Hinton, G. (2009). Deep Boltzmann Machines. In *Artificial Intelligence and Statistics* (pp. 448-455).

Sarle, W. S. (1994) Neural Networks and Statistical Models. Sas Institute Inc., NC, USA, (pp. 1539)

Sewak, M., Karim, M. R. and Pujari, P. (2018). *Practical Convolutional Neural Networks: Implement Advanced Deep Learning Models Using Python*. Packt Publishing Ltd.

Schluter, J. (2014). *Restricted Boltzmann Machine Derivations.* Technical Report TR-2014-13, Österreichisches Forschungsinstitut für Artificial Intelligence (OFAI), Vienna, Austria.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *The Journal Of Machine Learning Research,* **15(1):** 1929-1958.

Widrow, B. and Hoff, M. E. (1960). Adaptive Switching Circuits (No. TR-1553-1). Stanford Univ Ca Stanford Electronics Labs.

Williams, C. K. (1997). Computing with Infinite Networks. In *Advances in Neural Information Processing Systems.* pp. 295-301.

İnternet Kaynakları

1-https://www.kdnuggets.com/2017/07/rapidminer-ai-machine-learning-deep-learning.html, 05.01.2019

2-http://cs231n.github.io/neural-networks-1/ , 11.01.2019

3-https://ekblc.files.wordpress.com/2013/09/ysa.pdf , 01.03.2019

4-http://dataaspirant.com/2017/03/07/difference-between-softmax-function-and-sigmoid-function/ **,** 27.02.2019

5-https://wikidocs.net/3413 , 05.03.2019

6-https://www.willamette.edu/~gorr/classes/cs449/momrate.html , 10.03.2019

7-http://ufldl.stanford.edu/wiki/index.php/Stacked_Autoencoders , 15.03.2019

8-http://cs231n.github.io/convolutional-networks/#architectures , 27.03.2019

9-https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-3-flattening , 01.04.2019

10-https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-4-full-connection , 05.04.2019

11-https://www.cs.toronto.edu/~kriz/cifar.html , 10.04.2019

ÖZGEÇMİŞ

Adı Soyadı	: Barış AKPINAR
Yabancı Dili	: İngilizce

EKLER

EK 1. Bernoulli Kbm (Schluter 2014)

Enerji Fonksiyonu:

$$E\ (\ v,h,\theta\ ) = -\sum_i v_i\ a_i - \sum_{i,j} v_i\ w_{ij}\ h_j - \sum_j h_j\ b_j\ .$$

Görünür birimlerin koşullu dağılımı:

$$p(v_k = 1|h;\ \theta) = \sigma\left(a_k + \sum_j w_{kj}\ h_j\right)$$

Burada $\sigma(x) = (1 + e^{(-x)})^{-1}$, bir lojistik sigmoid fonksiyonudur,
u, görünür birimlerin g ise gizli birimlerin durumlarıdır,
θ, model parametreleri kümesidir (ağırlıklar ve sapmalarla),
a_i, i. görünür birimdeki sapma ve b_j ise j. gizli birimdeki sapmadır.

Gizli birimlerin koşullu dağılımı:

$$p(h_k = 1|v;\ \theta) \overset{1}{\Rightarrow} \frac{p(v, h_k = 1; \theta)}{p(v;\theta)}$$

$$= \frac{\sum_{\{g|g_k=1\}} p(v, g; \theta)}{\sum_g p(v, g; \theta)}$$

$$\overset{2}{\Rightarrow} \frac{\sum_{\{g|g_k=1\}} e^{\ (-E(v,g;\theta)}}{\sum_g e^{(-E(v,g;\theta)}}$$

$$\overset{3}{\Rightarrow} \frac{\sum_{\{g|g_k=1\}} e^{(-E(v,g;\theta)}}{\sum_{\{g|g_k=1\}} e^{(-E(v,g;\theta)} + \sum_{\{g|g_k=0\}} e^{(-E(v,g;\theta)}}$$

$$= \frac{1}{1 + \frac{\sum_{\{g|g_k=0\}} e^{(-E(v,g;\theta)}}{\sum_{\{g|g_k=1\}} e^{(-E(v,g;\theta)}}}$$

$$\overset{4}{\Rightarrow}\frac{1}{1+\frac{\sum_{\{g|g_k=0\}} e^{(\sum_{i,j} v_i\, w_{ij}\, h_j+\sum_i v_i\, a_i+\sum_j g_j\, b_j)}}{\sum_{\{g|g_k=1\}} e^{(\sum_{i,j} v_i\, w_{ij}\, h_j+\sum_i v_i\, a_i+\sum_j g_j\, b_j)}}}$$

$$\overset{5}{\Rightarrow}\frac{1}{1+\frac{\sum_{\{g|g_k=0\}} e^{(\sum_i \sum_{j\neq k} v_i\, w_{ij}\, h_j+\sum_i v_i\, a_i+\sum_{j\neq k} g_j\, b_j)}}{\sum_{\{g|g_k=1\}} e^{(\sum_i \sum_{j\neq k} v_i\, w_{ij}\, h_j+\sum_i v_i\, a_i+\sum_{j\neq k} g_j\, b_j)}}\cdot\frac{e^{(\sum_i v_i\, w_{ik}.0+0b_k)}}{e^{(\sum_i v_i\, w_{ik}.1+1b_k)}}}$$

$$\overset{6}{\Rightarrow}\frac{1}{1+1\cdot\frac{1}{e^{(\sum_i v_i\, w_{ik}.1+1b_k)}}}$$

$$=\frac{1}{1+e^{-(b_k+\sum_i v_i\, w_{ik})}}$$

$$=\sigma\left(b_k+\sum_i v_i\, w_{ik}\right)$$

Sayı ile işaretlenmiş adımların açıklaması:

1- Koşullu olasılık tanımından geliyor.
2- Ortak olasılık yoğunluk fonksiyonun tanımından geliyor.
3- Aktif ve aktif olmayan gizli birim g_k ile toplam ikiye ayrıldı.
4- Enerji fonksiyonundan geliyor.
5- $g_j = g_k$ bilinen değerlerinin yerine kullanılan j'nin toplamlarından $j = k$ ilgili terimleri çıkarıldı.
6- g_k 'ya bağlı terimler iki uzun üstel fonksiyondan çıkarıldığından $\{g|g_k = 0\}$ ve $\{g|g_k = 1\}$ olduğunda bölümleri 1'e eşittir.

EK 2. Log Olasılık Gradyanı (Schluter 2014)

Model parametresi θ için bir bağlantı ağırlığı w_{ij}, görünür sapma a_i, gizli sapma b_j'dir. Bağlantı ağırlığı:

$$Gw_{ij}(v,\theta) = \sum_g p(g|v;\theta)\cdot\frac{\partial}{\partial w_{ij}}(-E(v,g,\theta))$$

$$\overset{1}{\Rightarrow} \sum_g p(g|v;\theta)\cdot v_i\, g_j$$

$$\overset{2}{\Rightarrow} v_i \cdot \sum_g p(g|v;\theta)\, g_j$$

$$\overset{3}{\Rightarrow} v_i \cdot \sum_{g_j}\sum_{g\neg_j} p(g_j|v;\theta)\, p(g\neg_j|v;\theta)\, g_j$$

$$\overset{4}{\Rightarrow} v_i \cdot \sum_{gj} p(g_j|v;\theta) g_j \sum_{g\neg_j} p(g\neg_j|v;\theta)$$

$$\overset{5}{\Rightarrow} v_i \cdot \sum_{g_j} p(g_j|v;\theta) g_j$$

$$\overset{6}{\Rightarrow} v_i \cdot \left(p(h_j=1|v;\theta)\cdot\mathbf{1} + p(h_j=0|v;\theta)\cdot\mathbf{0}\right)$$

$$= v_i \cdot p(h_j=1|v;\theta)$$

$$\overset{7}{\Rightarrow} v_i \cdot \sigma\left(b_j + \sum_k v_k\, w_{kj}\right)$$

3. ve önceki işlemlerde aşağıdaki denklemle işlem yapılmıştır:

$$\sum_g p(g|v;\,\theta) g_j \;=\; p(h_j=1|v;\theta)$$

Görünür sapmanın gradyanı:

$$Ga_i(v,\theta) = \sum_g p(g|v;\theta)\cdot\frac{\partial}{\partial a_i}(-E(v,g,\theta))$$

$$\overset{1}{\Rightarrow} \sum_g p(g|v;\theta)\cdot v_i$$

$$\overset{2}{\Rightarrow} v_i \cdot \sum_g p(g|v;\theta)$$

$$\overset{5}{\Rightarrow} v_i$$

Gizli sapmanın gradyanı:

$$Gb_j(v,\theta) = \sum_g p(g|v;\theta) \cdot \frac{\partial}{\partial b_j}(-E(v,g,\theta))$$

$$\overset{1}{\Rightarrow} \sum_g p(g|v;\theta) \cdot g_j$$

$$\overset{8}{\Rightarrow} \sigma\left(b_j + \sum_k v_k\, w_{kj}\right)$$

Sayı ile işaretlenmiş adımların açıklamaları:

1- Enerji fonksiyonu yerine konulur ve türevi alınır.
2- Her g için v_i faktörü aynıdır. Böylece toplam çekilebilir.
3- Gizli birimler için $g\neg_j$ ve g_j'nin ikili durumları için toplam ikiye ayrıldı.
4- Her $g\neg_j$ için $p\left(g_j\middle|v;\theta\right)g_j$ faktörü aynıdır. Böylece toplam çekilebilir.
5- Olasılık dağılımında toplam 1 olarak değerlendirilir ve düşürülebilir.
6- g_j ya da h_j'nin toplamdaki iki olası değeri yazılıyor.
7- Gizli birimler için koşullu aktivasyon olasılığı değişimi yapılıyor.
8- Döngüyü ifade ediyor.

$G_{w_{ij}}(v,\theta)$, $G_{a_i}(v,\theta)$, $G_{b_j}(v,\theta)$'ler için aşağıdaki sonuçlar elde edilir:

$$\frac{\partial}{\partial w_{ij}}\log p(v;\,\theta) = v_i \cdot \sigma(b_j + \sum_k v_k\, w_{kj}) - \langle u_i \cdot \sigma(b_j + \sum_k u_k w_{kj})\rangle_{p(u;\theta)}$$

$$\frac{\partial}{\partial a_i}\log p(v;\,\theta) = v_i - \langle u_i\rangle_{p(u;\theta)}$$

$$\frac{\partial}{\partial b_j}\log p(v;\ \theta) = \ \sigma(b_j + \sum_k v_k\, w_{kj}) - \langle\sigma(b_j + \sum_k u_k w_{kj})\rangle_{p(u;\theta)}$$

Bu aşamadan sonra $p(v;\ \theta)$ için ikinci denklemlerle güncellemeler yapılır. İlgili gradyanlar:

$$\overline{hj}(v;\theta) \coloneqq p(h_j = 1|v;\theta)$$

$$\frac{\partial}{\partial w_{ij}}\log p(v;\ \theta) = v_i.\overline{h_j}(v;\theta) - \langle u_i.\overline{h_j}(u;\theta)\rangle_{p(u;\theta)}$$

$$\frac{\partial}{\partial a_i}\log p(v;\ \theta) = v_i - \langle u_i\rangle_{p(u;\theta)}$$

$$\frac{\partial}{\partial b_j}\log p(v;\ \theta) = \ \overline{h_j}(v;\theta) - \langle\overline{h_j}(u;\theta)\rangle_{p(u;\theta)}$$

Burada $\overline{h_j}(v;\theta)$ ve $\overline{h_j}(u;\theta)$ ikili örnekler değil gizli birimlerin gerçek değerli aktivasyon olasılıklarıdır.

EK 3. Gauss-Bernoulli Kbm Krizhevsky and Hinton (2009)'a göre (Schluter 2014)

Enerji Fonksiyonu:

$$E\,(\,v,h,\theta\,) = \sum_i \frac{(v_i-a_i)^2}{2\,{\sigma_i}^2} - \sum_{i,j} \frac{v_i}{\sigma_i}\, w_{ij}\, h_j - \sum_j h_j\, b_j.$$

Görünür birimlerin koşullu dağılımı:

$$p(v|h;\,\theta) = \frac{e^{-E(v,h,\theta)}}{\int_u e^{-E(u,h,\theta)}du}$$

$$= \frac{e^{\left(-\sum_i \frac{(v_i-a_i)^2}{2\sigma_i^2} + \sum_{i,j} \frac{vi}{\sigma_i} w_{ij}\, h_j + \sum_j h_j\, b_j\right)}}{\int_u e^{\left(-\sum_i \frac{(u_i-a_i)^2}{2\sigma_i^2} + \sum_{i,j} \frac{u_i}{\sigma_i} w_{ij}\, h_j + \sum_j h_j\, b_j\right)}du}$$

$$= \frac{e^{\left(-\sum_i \frac{(v_i-a_i)^2}{2\sigma_i^2} + \sum_{i,j} \frac{vi}{\sigma_i} w_{ij}\, h_j\right)} e^{(\sum_j h_j\, b_j)}}{\int_u e^{\left(-\sum_i \frac{(u_i-a_i)^2}{2\sigma_i^2} + \sum_{i,j} \frac{u_i}{\sigma_i} w_{ij}\, h_j\right)} e^{(\sum_j h_j\, b_j)}du}$$

$$\overset{1}{\Rightarrow} \frac{e^{\left(-\sum_i \frac{(v_i-a_i)^2}{2\sigma_i^2} + \sum_i \frac{v_i}{\sigma_i} \sum_j w_{ij}\, h_j\right)}}{\int_u e^{\left(-\sum_i \frac{(u_i-a_i)^2}{2\sigma_i^2} + \sum_i \frac{u_i}{\sigma_i} \sum_j w_{ij}\, h_j\right)}du}$$

$$\overset{2}{\Rightarrow} \frac{e^{\left(-\sum_i \frac{(v_i-a_i)^2}{2\sigma_i^2} + \sum_i \frac{v_i}{\sigma_i} B_i\right)}}{\int_u e^{\left(-\sum_i \frac{(u_i-a_i)^2}{2\sigma_i^2} + \sum_i \frac{u_i}{\sigma_i} B_i\right)}du} \quad , \quad B_i \coloneqq \sum_j w_{ij} h_j$$

$$= \frac{\prod_i e^{\left(-\sum_i \frac{(v_i-a_i)^2}{2\sigma_i^2} + \sum_i \frac{v_i}{\sigma_i} B_i\right)}}{\int_u \prod_i e^{\left(-\sum_i \frac{(u_i-a_i)^2}{2\sigma_i^2} + \sum_i \frac{u_i}{\sigma_i} B_i\right)}du}$$

$$\overset{3}{\Rightarrow} \frac{\cdots}{\int_u \prod_i e^{\left(-\sum_i \frac{(u_i-a_i)^2}{2\sigma_i^2} + \sum_i \frac{u_i}{\sigma_i} B_i\right)}du}$$

$$= \frac{\cdots}{\int_u \prod_i e^{\left(-u_i^2 \frac{1}{2\sigma_i^2} + u_i\left(\frac{a_i}{\sigma_i^2} + \frac{Bi}{\sigma_i}\right) - \frac{a_i^2}{2\sigma_i^2}\right)}du_i}$$

$$\overset{4}{\Rightarrow}\frac{\cdots}{\prod_i \sqrt{\pi 2\sigma_i^2}\, e^{\left(\frac{2\sigma_i^2}{4}\left(\frac{a_i}{\sigma_i^2}+\frac{Bi}{\sigma_i}\right)^2-\frac{a_i^2}{2\sigma_i^2}\right)}}$$

$$=\frac{\cdots}{\prod_i \sigma_i\sqrt{2\pi}\, e^{\left(\frac{1}{2}\left(\frac{a_i}{\sigma_i}+B_i\right)^2-\frac{a_i^2}{2\sigma_i^2}\right)}}$$

$$=\frac{\cdots}{\prod_i \sigma_i\sqrt{2\pi}\, e^{\left(\frac{1}{2}\left(\frac{a_i^2}{\sigma_i^2}+2\frac{a_i}{\sigma_i}B_i+B_i^2\right)-\frac{a_i^2}{2\sigma_i^2}\right)}}$$

$$=\frac{\prod_i e^{\left(-\frac{(v_i-a_i)^2}{2\sigma_i^2}+\frac{vi}{\sigma_i}B_i\right)}}{\prod_i \sigma_i\sqrt{2\pi}\, e^{\left(\frac{a_i}{\sigma_i}B_i+\frac{1}{2}B_i^2\right)}}$$

$$=\prod_i \frac{1}{\sigma_i\sqrt{2\pi}} e^{\left(-\frac{(v_i-a_i)^2}{2\sigma_i^2}+\frac{v_i}{\sigma_i}B_i-\frac{a_i}{\sigma_i}Bi+\frac{1}{2}B_i^2\right)}$$

$$=\prod_i \frac{1}{\sigma_i\sqrt{2\pi}} e^{\left(-\frac{1}{2\sigma_i^2}((v_i-a_i)^2-2\sigma_i(v_i-a_i)B_i+\sigma_i^2B_i^2)\right)}$$

$$=\prod_i \frac{1}{\sigma_i\sqrt{2\pi}} e^{\left(-\frac{1}{2\sigma_i^2}((v_i-a_i)-\sigma_iB_i)^2\right)}$$

$$\overset{5}{\Rightarrow}\prod_i \mathcal{N}(v_i; a_i+\sigma_i\sum_j w_{ij}h_j, \sigma_i^2)$$

Sayı ile işaretlenmiş adımların açıklaması:

1- Paydadaki $e^{(\sum_j h_j b_j)}$ terimi u ile ilgili sabittir. Böylece integralden bir faktör olarak dışarı çekilebilir.

2- Sadece gösterimi kolaylaştırmak için $B_i \coloneqq \sum_j w_{ij}h_j$ kullanıldı

3- $\int_u \prod_i f(u_i)du = \int_{u_1}\int_{u_2}\cdots\int_{u_n} f(u_1)f(u_2)\cdots f(u_n)du_1du_2\cdots du_n$,yeniden düzenlenebilir: $\int_{u_1} f(u_1)du_1 \int_{u_2} f(u_2)du_2 \cdots \int_{u_n} f(u_n)du_n = \prod_i \int_{u_i} f(u_i)du_i$. Çünkü diğer bütünleşmiş değişkenler ile ilgili olarak integrallerin herhangi biri sabit bir çarpandır.

4- $\int_x e^{(-x^2a+xb+c)dx} = \sqrt{\frac{\pi}{a}}\, e^{(\frac{b^2}{4a}+c)}$, Gauss integral fonksiyonun çözümündeki

dönüşüm.

5- Kolaylık sağlaması için normal dağılım: $\mathcal{N}(\mathrm{x};\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{(-\frac{(x-\mu)^2}{2\sigma^2})}$. Denklem aynı zamanda Gauss dağılımının olasılık yoğunluk fonksiyonu olarak da adlandırılmaktadır.

EK 4. Gizli Birimlerin Koşullu Dağılımı (Schluter 2014):

$$p(h_k = 1|v) = \frac{1}{1 + \frac{\sum_{\{g|g_k=0\}} e^{-E(v,g,\theta)}}{\sum_{\{g|g_k=1\}} e^{-E(v,g,\theta)}}}$$

$$\overset{1}{\Rightarrow} \frac{1}{1 + \frac{\sum_{\{g|g_k=0\}} e^{\left(\sum_{i,j}\frac{v_i}{\sigma_i}w_{ij}g_j - \sum_i\frac{(v_i-a_i)^2}{\sigma_i^2} + \sum_j g_j b_j\right)}}{\sum_{\{g|g_k=1\}} e^{\left(\sum_{i,j}\frac{v_i}{\sigma_i}w_{ij}g_j - \sum_i\frac{(v_i-a_i)^2}{\sigma_i^2} + \sum_j g_j b_j\right)}}}$$

$$\overset{2}{\Rightarrow} \frac{1}{1 + \frac{e^{\left(\sum_i\frac{v_i}{\sigma_i}w_{ik}.0 + 0.b_k\right)}}{e^{\left(\sum_i\frac{v_i}{\sigma_i}w_{ik}.1 + 1.b_k\right)}}}$$

$$= \frac{1}{1 + \frac{1}{e^{\left(\sum_i\frac{v_i}{\sigma_i}w_{ik} + b_k\right)}}}$$

$$= \frac{1}{1 + e^{\left(-\left(b_k + \sum_i\frac{v_i}{\sigma_i}w_{ik}\right)\right)}}$$

$$= \sigma\left(b_k + \sum_i \frac{v_i}{\sigma_i} w_{ik}\right)$$

Sayı ile işaretlenmiş adımların açıklaması:

1- Enerji fonksiyonu yerine koyuluyor.
2- Bilinen değerleri $g_j = g_k$ ile değiştiriliyor ve g_k'ya bağlı olmayan tüm terimler iptal ediliyor.

EK 5. Log Olasılık Gradyanı: (Schluter 2014)

Model parametresi θ için bir bağlantı ağırlığı w_{ij}, görünür sapma a_i, gizli sapma b_j ve görünür standart sapma σ_i, $G_\theta(v,\theta)$'da hesaplanıyor. Bağlantı ağırlığı:

$$G_{w_{ij}}(v,\theta) = \sum_g p(g|v;\theta)\cdot\frac{\partial}{\partial w_{ij}}(-E(v,g,\theta))$$
$$\overset{1}{\Rightarrow} \sum_g p(g|v;\theta)\cdot\frac{v_i}{\sigma_i}gj$$
$$\overset{2}{\Rightarrow} \frac{v_i}{\sigma_i}\cdot\sum_g p(g|v;\theta)gj$$
$$\overset{3}{\Rightarrow} \frac{v_i}{\sigma_i}\cdot p(h_j=1|v;\theta)$$
$$\overset{4}{\Rightarrow} \frac{v_i}{\sigma_i}\cdot\overline{h_j}(v;\theta)$$

Görünür sapmanın gradyanı:

$$G_{a_i}(v,\theta) = \sum_g p(g|v;\theta)\cdot\frac{\partial}{\partial \sigma_i}(-E(v,g,\theta))$$
$$\overset{1}{\Rightarrow} \sum_g p(g|v;\theta)\cdot\frac{v_i-a_i}{\sigma_i^2}$$
$$\overset{5}{\Rightarrow} \frac{v_i-a_i}{\sigma_i^2}\cdot\sum_g p(g|v;\theta)$$
$$\overset{6}{\Rightarrow} \frac{v_i-a_i}{\sigma_i^2}$$

Gizli sapma için gradyan:

$$G_{b_j}(v,\theta) = \sum_g p(g|v;\theta)\cdot\frac{\partial}{\partial b_j}(-E(v,g,\theta))$$
$$\overset{1}{\Rightarrow} \sum_g p(g|v;\theta)\cdot g_j$$

$$\overset{3,4}{\Rightarrow} \overline{h_j}(v;\theta)$$

Görünür standart sapma için gradyan hesabı:

$$G_{\sigma_i}(v,\theta) = \sum_g p(g|v;\theta)\cdot\frac{\partial}{\partial\sigma_i}(-E(v,g,\theta))$$

$$\overset{1}{\Rightarrow} \sum_g p(g|v;\theta)\cdot\left(\frac{(v_i-a_i)^2}{\sigma_i^3}-\frac{v_i}{\sigma_i^2}\sum_j w_{ij}g_j\right)$$

$$=\sum_g p(g|v;\theta)\frac{(v_i-a_i)^2}{\sigma_i^3}-\sum_g p(g|v;\theta)\frac{v_i}{\sigma_i^2}\sum_j w_{ij}g_j$$

$$\overset{7}{\Rightarrow}\frac{(v_i-a_i)^2}{\sigma_i^3}\cdot\sum_g p(g|v;\theta)-\frac{v_i}{\sigma_i^2}\sum_j w_{ij}\cdot\sum_g p(g|v;\theta)g_j$$

$$\overset{6,3}{\Rightarrow}\frac{(v_i-a_i)^2}{\sigma_i^3}-\frac{v_i}{\sigma_i^2}\sum_j w_{ij}\cdot p(h_j=1|v;\ \theta)$$

$$\overset{4}{\Rightarrow}\frac{(v_i-a_i)^2}{\sigma_i^3}-\frac{v_i}{\sigma_i^2}\sum_j w_{ij}\cdot\overline{h_j}(v;\theta)$$

Sayı ile işaretlenmiş adımların açıklaması:

1- Enerji fonksiyonu ve farklılaştırılması.

2- Her g için $\frac{v_i}{\sigma_i}$ faktörü aynıdır. Böylece toplam çekilebilir.

3- KBM'deki log olasılık gradyanıyla aynı işlemdir.

4- KBM'deki h kısaltması

5- Her g için $\frac{v_i-a_i}{\sigma_i^2}$ faktörü aynıdır. Böylece toplam çekilebilir.

6- Olasılık dağılımındaki toplam 1 olarak değerlendirilir ve düşürülebilir.

7- g'den bağımsız olan tüm faktörleri g üzerindeki toplamlardan çıkarılır.

$G_{w_{ij}}(v,\theta)$, $G_{a_i}(v,\theta)$, $G_{b_j}(v,\theta)$, $G_{\sigma_i}(v,\theta)$, 7'de adım için aşağıdaki sonuçlar elde edilir:

$$\frac{\partial}{\partial w_{ij}} \log p\ (v;\ \theta) = \frac{v_i}{\sigma_i} \cdot \overline{h_j}(v;\theta) - \langle \frac{u_i}{\sigma_i} \cdot \overline{h_j}(v;\theta) \rangle_{p(u;\theta)}$$

$$\frac{\partial}{\partial ai} \log p\ (v;\ \theta) = \frac{v_i - a_i}{\sigma_i^2} - \langle \frac{u_i - a_i}{\sigma_i^2} \rangle_{p(u;\theta)}$$

$$= \frac{v_i}{\sigma_i^2} - \frac{a_i}{\sigma_i^2} - \langle \frac{u_i}{\sigma_i^2} - \frac{a_i}{\sigma_i^2} \rangle_{p(u;\theta)}$$

$$= \frac{v_i}{\sigma_i^2} - \frac{a_i}{\sigma_i^2} + \frac{a_i}{\sigma_i^2} - \langle \frac{u_i}{\sigma_i^2} \rangle_{p(u;\theta)}$$

$$= \frac{v_i}{\sigma_i^2} - \langle \frac{u_i}{\sigma_i^2} \rangle_{p(u;\theta)}$$

$$\frac{\partial}{\partial b_j} \log p\ (v;\ \theta) = \overline{h_j}(v;\theta) - \langle \overline{h_j}(v;\theta) \rangle_{p(u;\theta)}$$

$$\frac{\partial}{\partial \sigma_i} \log p\ (v;\ \theta) = \frac{(v_i - a_i)^2}{\sigma_i^3} - \frac{v_i}{\sigma_i^2} \sum_j w_{ij} \cdot \overline{h_j}(v;\theta) -$$

$$\langle \frac{(v_i - a_i)^2}{\sigma_i^3} - \frac{v_i}{\sigma_i^2} \sum_j w_{ij} \cdot \overline{h_j}(v;\theta) \rangle_{p(u;\theta)}$$