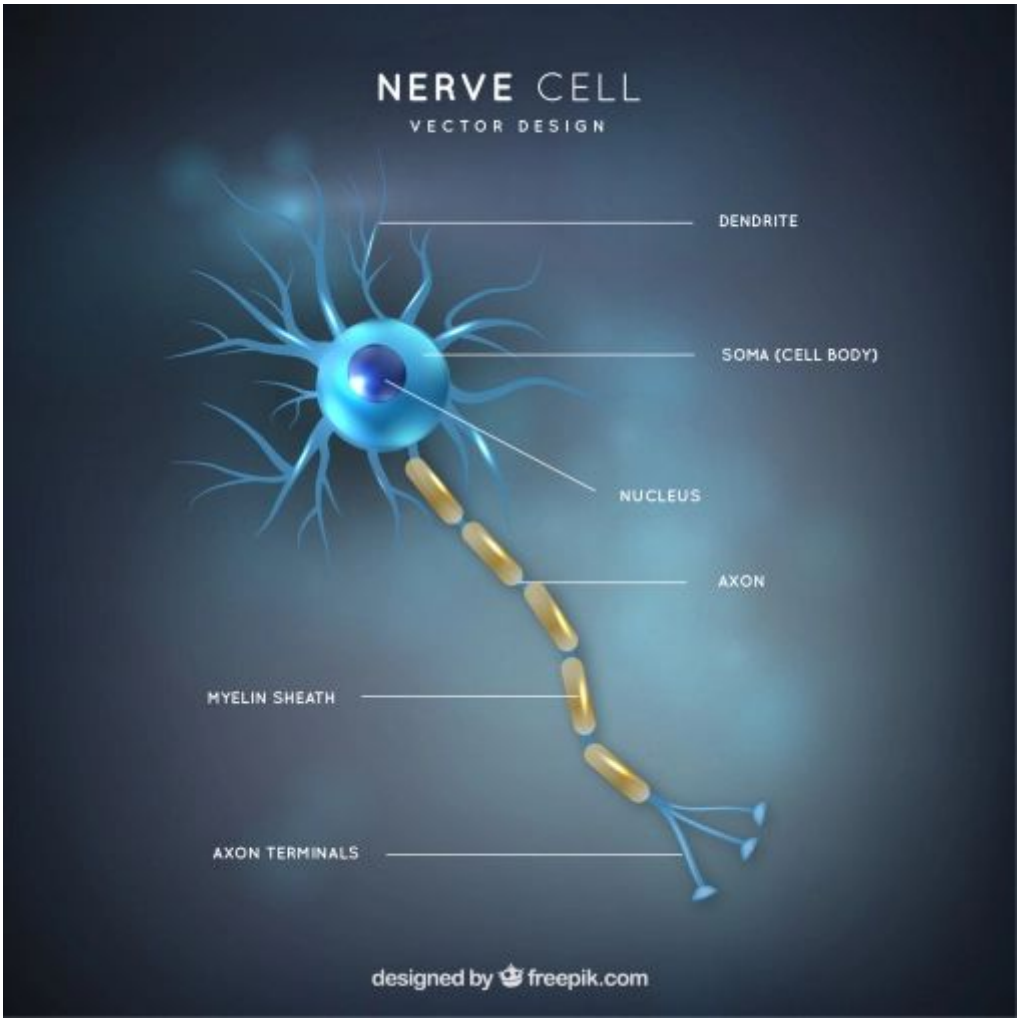
Bölüm 7 : Yapay Sinir Ağları

7.1 Sinir Ağları

İnsan beyni, biyolojik sinir ağı olarak tanımlanır. Nöronlar birbirine bağlı bir web ağı gibi ayrıntılı elektrik sinyalleri üretir. Dendritler giriş sinyallerini alır ve bu girdilere dayanarak bir akson aracılığı ile bir çıkış sinyalini tetikler. İnsan beyninin gerçekte nasıl çalıştığı aslında çok daha ayrıntılı ve karmaşık bir gizemdir.



Bilgisayar bilimcileri uzun süredir insan beyninden esinlenmişlerdir. 1943 yılında, nörolog Warren S. McCulloch, ve mantıkçı Walter Pitts bir yapay sinir ağının ilk kavramsal modelini geliştirmişlerdir. Yapay sinir ağı belirli problemleri çözmek için beyne dayalı bir hesaplama modeli olarak tasarlanmıştır.

Günümüzdeki hesaplamalarda sinir ağlarının en yaygın uygulaması, genellikle “örüntü tanıma” olarak adlandırılan, “insan için kolay, makine için zor” görevleri gerçekleştirmektir.

Uygulamaları, optik karakter tanıma (basılı veya el yazısı taramaları dijital metin haline getirme) ve yüz tanıma olarak örneklendirebiliriz.

Bir sinir ağı “bağlantılı” bir hesaplama sistemidir. Sinir ağında, bilgi bir düğüm ağı (düğümler, nöronlar) boyunca paralel olarak topluca işlenir.

Ağın bireysel unsurları olan nöronlar basit yapılardır. Bir girişi okur, işler ve bir çıktı üretir. Bununla birlikte, bir çok nöron ağı son derece zengin ve zeki davranışlar gösterebilir.

Sinir ağının en önemli unsurlarından biri öğrenme kabiliyetidir. Bir sinir ağı sadece karmaşık bir sistem değil, karmaşık bir uyarlanabilir sistemdir, yani içinde akan bilgiye dayalı olarak iç yapısını değiştirebilir. Tipik olarak, bu ağırlıkların ayarlanmasıyla sağlanır.

Öğrenme için birkaç yöntem vardır.

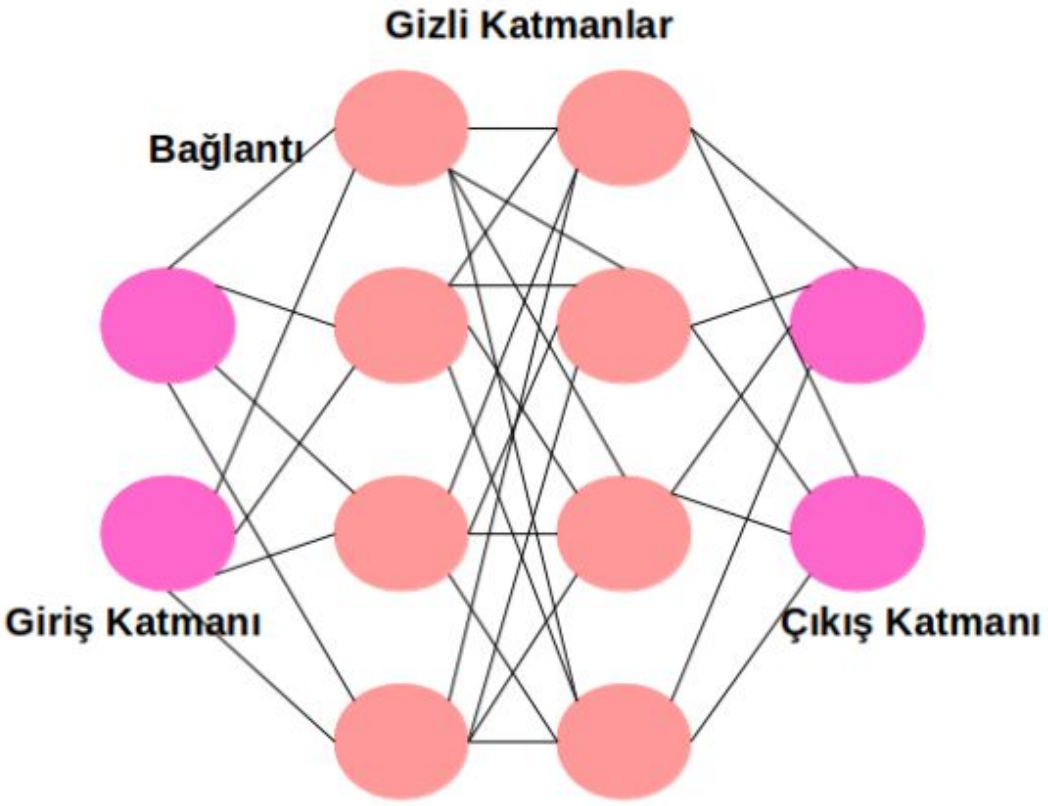
1. Denetimli Öğrenme: Örneğin, yüz tanıma örneğini ele alalım. Ağa bir sürü yüz gösterilir ve zaten her yüz ile ilişkili kişiyi biliriz. Ağ tahminlerini yapar, ardından ağa doğru cevapları sunarız. Ağ, tahminlerini, bilinen “doğru” cevaplarla karşılaştırabilir ve hatalarına göre düzenlemeler yapabilir.
2. Denetimsiz Öğrenme: Bilinen cevaplarla örnek veri seti olmadığında gereklidir. Örneğin bir veri kümesindeki gizli bir kalıp aramayı hayal edin. Bunun bir uygulaması kümeleme, yani bilinmeyen bir desene göre bir grup öğenin gruplara bölünmesidir.
3. Takviyeli Öğrenme: Gözlem üzerine kurulan bir stratejidir. Sinir ağı bir sonuçla (sağa veya sola dönmek gibi) karar verir ve çevresini gözlemler. Gözlem negatif ise, ağ bir dahaki sefere farklı bir karar vermek için ağırlıklarını ayarlayabilir.

Bir sinir ağının öğrenmesi ve yapısını zaman içinde ayarlayabilme kabiliyeti onu yapay zeka alanında faydalı kılmaktadır. Günümüz yazılımlarında sinir ağlarının bazı standart kullanımları aşağıda verilmiştir.

- Desen Tanınması
- Zaman Serileri Tahmini
- Sinyal İşleme
- Kontrol
- Yumuşak Algılayıcılar
- Anormallik Tespiti

Yapay sinir ağı (ANN, artificial neural network) olarak adlandırılan bir sinir ağının en basit tanımı, ilk nöro-bilgisayardan biri olan Dr. Robert Hecht-Nielsen tarafından yaratılmıştır.

Sinir ağları genellikle katmanlar halinde organize edilir. Katmanlar, bir ‘aktivasyon fonksiyonu’ içeren birbirine bağlı birçok ‘düğümlerden’ oluşur. Kalıplar ağa gerçek hesaplamanın ağırlıklı ‘bağlantılar’ sistemi üzerinden yapıldığı bir veya daha fazla ‘gizli katman’ ile iletişim kuran ‘girdi katmanı’ aracılığıyla sunulur. Gizli katmanlar daha sonra bir ‘çıktı katmanı’ ile bağlantı kurar.

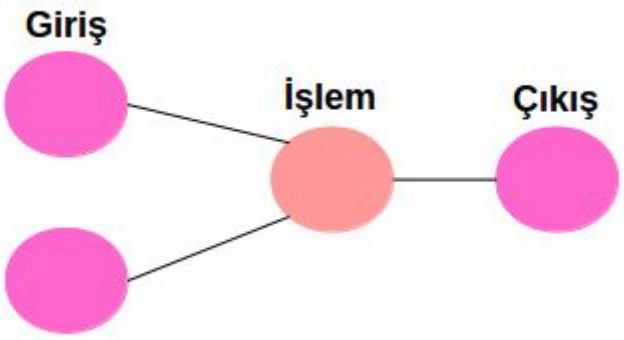


Çoğu ANN'ler, bağlantıların ağırlıklarını, sunulduğu girdi düzenlerine göre değiştiren bir takım 'öğrenme kuralı' içerir. Bir bakıma, ANN'ler biyolojik benzerleri kadar örnekle öğrenirler.

Biyolojik Sinir Sistemi	Yapay Sinir Sistemi
Nöron	Hesaplama Yapan Düğüm
Dendrit	Toplama Fonksiyonu
Hücre Gövdesi	Transfer Fonksiyonu
Akson	Çıkış Düğümü
Sinapslar	Ağırlıklar

7.2 Algılayıcı (Perceptron)

1957'de Cornell Havacılık Laboratuvarı'ndaki Frank Rosenblatt tarafından icat edilen bir algılayıcı, mümkün olan en basit sinir ağıdır: tek bir nöronun hesaplama modelidir. Bir algılayıcı, bir veya daha fazla girdi, bir işlem ve tek bir çıktıdan oluşur.



Bir “algılayıcı”, “ileri besleme” modelini izler; bu da girişler nörona gönderilir, işlenir ve çıktı haline getirilir demektir.

Bu adımların her birine daha ayrıntılı bakalım:

Girdileri Almak
Girdi 1: g1 = 5
Girdi 2: g2 = 6

Girdilerin Ağırlıkları
Nörona gönderilen her girdi önce ağırlıklandırılmalıdır, yani bir miktar çarpılmalıdır. Bir algılayıcı oluştururken, tipik olarak rastgele ağırlık arayarak başlayabiliriz.

Ağırlık 1: -1
Ağırlık 2: 0.8

Her girdiyi alıp ağırlığı ile çarpalım:

Girdi 1 * Ağırlık 1: 5 * -1 = -5

Girdi 2 * Ağırlık 2: 6 * 0.8 = 4.8

Girdilerin Toplamı
Daha sonra ağırlıklandırılmış girdileri toplayalım:

Toplam: -5 + 4.8 = -0.2

Çıktı Üretmek
Bir algılayıcının çıktısı, bu toplamı bir aktivasyon fonksiyonundan geçirerek üretilir. Basit bir dijital çıkış durumunda, aktivasyon fonksiyonu algılayıcıya tetiklenip tetiklenmediğini söyler.

Aktivasyon fonksiyonunu toplamın işareti yapalım. Yani, toplam pozitif bir sayı ise çıkış 1, negatif ise, çıktı -1 olsun:

Çıktı = işaret (toplam): işaret (-0.2) ⇒ -1

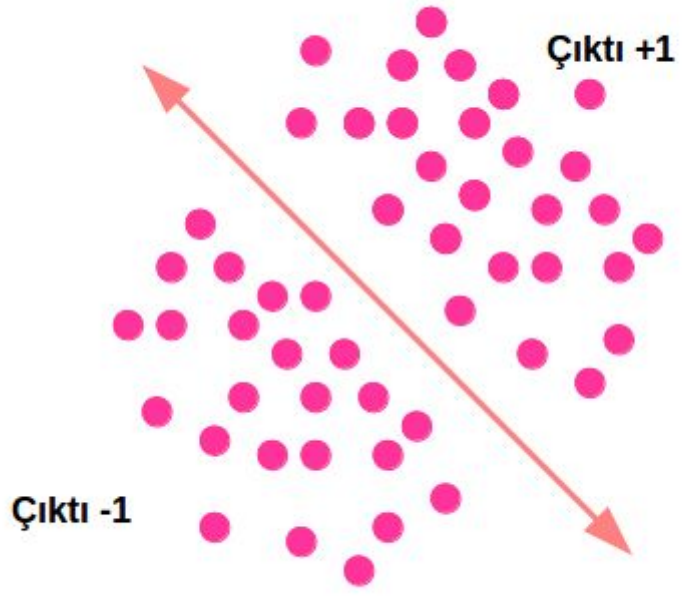
Algılayıcı Algoritması
Her girdi için, bu girdiyi ağırlığıyla çarpın

Tüm ağırlıklı girdileri toplayın
Algılayıcının çıktısı, bu toplamın, bir aktivasyon fonksiyonundan geçirilmesi ile hesaplayın

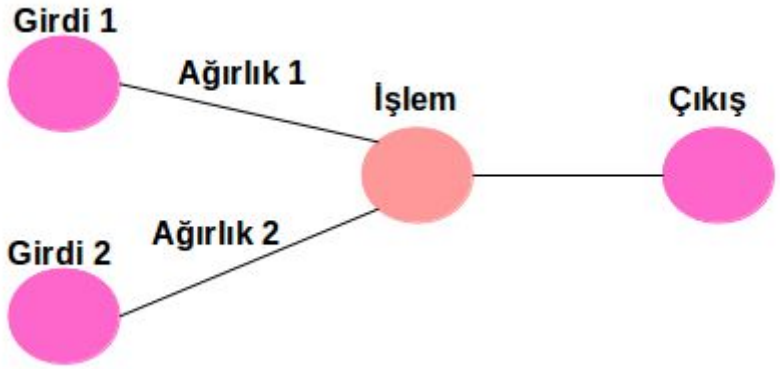
```
import numpy as np

girdiler = np.array([5,6])

agirliklar = np.array([-1,0.8])

agirlikli_toplam = girdiler * agirliklar

toplam = sum(agirlikli_toplam)

aktivasyon_fonksiyonu = (lambda x: 1 if x&amp;gt;0 else -1)

cikti = aktivasyon_fonksiyonu(toplam)

print cikti
```

ALGILAYICI KULLANARAK BASİT BİR ÖRÜNTÜ TANIMA

İki boyutlu uzayda bir çizgi düşünün. O alandaki noktalar, çizginin bir tarafında ya da diğer tarafında olarak sınıflandırılabiliriz.



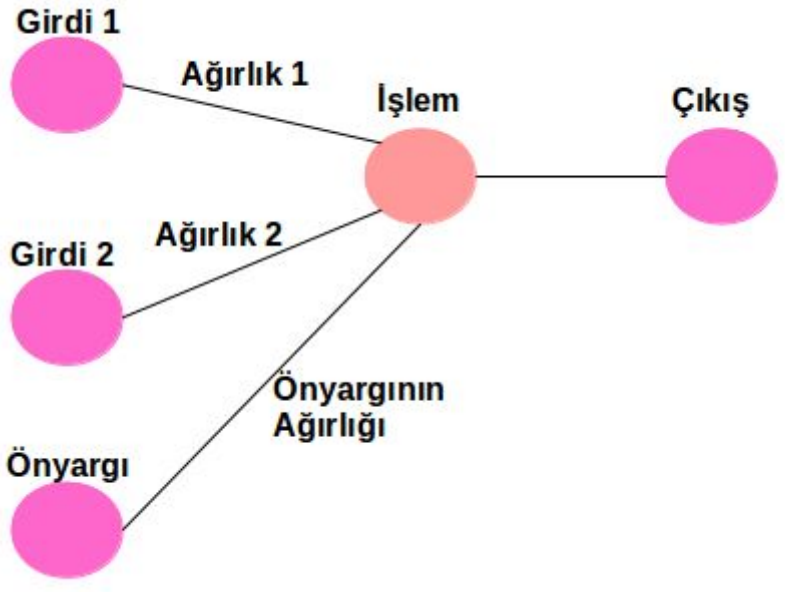
Algılayıcımızın x ve y koordinatı olmak üzere, girdi 1 ve girdi 2 isimli iki girdisi olsun. İşaret aktivasyon fonksiyonunu kullanalım, çıktı ya -1 ya da 1 olsun, yani, girdi verileri çıktı işaretine göre sınıflandırılsın.



İki girdinin (girdi 1 ve girdi 2), her girdi için bir ağırlığı (ağırlık 1 ve ağırlık 2) ve çıktıyı üreten bir işlem nöronumuz olsun.

Ancak burada çok önemli bir problemimiz var. (0,0) noktasını düşünelim. Bu noktayı algılayıcıya girdi olarak verirsek ne olur: girdi 1 = 0 ve girdi 2 = 0 ? Ağırlıklı girdilerin toplamı ne olacaktır? Ağırlıklar ne olursa olsun, toplam daima 0 olacaktır. Ancak bu doğru olamaz, sonuçta (0,0) noktası kesinlikle iki boyutlu uzayımızda çizgimizin ya üstünde ya da altında olmak zorundadır.

Bu ikilemden kaçınmak için, algılayıcımız, tipik olarak bir önyargı girişi olarak ifade edilen üçüncü bir girdi gerektirecektir. Bir önyargı girişi her zaman 1 değerine sahiptir ve ayrıca ağırlıklandırılmıştır. Önyargı girdisini ekleyelim, algılayıcımız şöyle olacaktır:



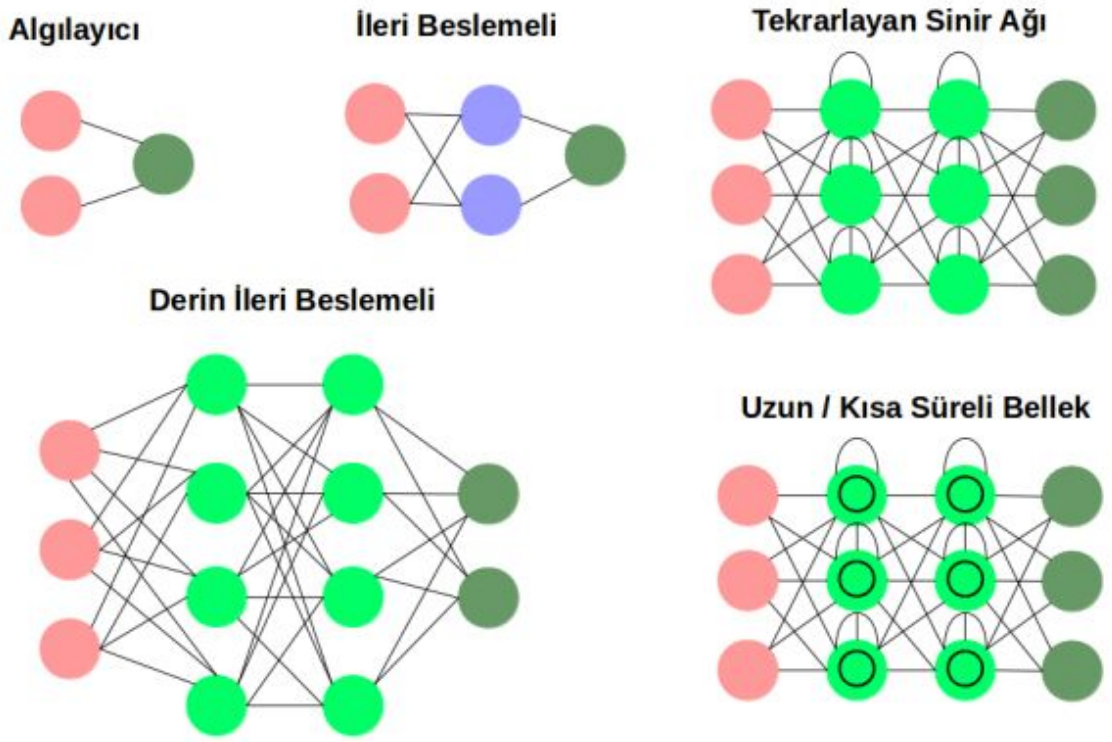
(0,0) noktasına dönelim. İşte girdilerimiz:

0 * girdi 1 için ağırlık = 0
0 * girdi 2 için ağırlık = 0

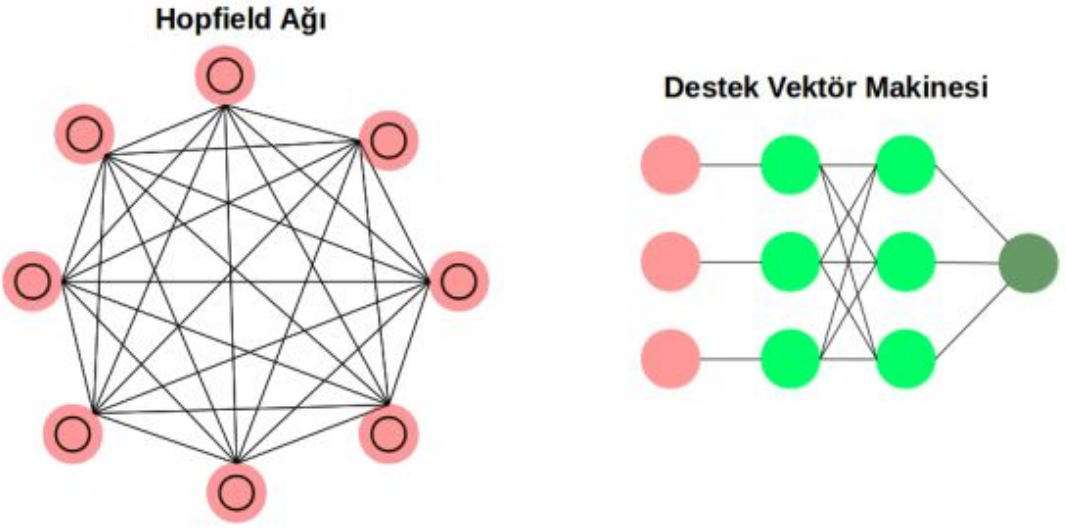
1 * önyargı için ağırlık = önyargı için ağırlık

Çıktı, yukarıdaki üç değerin toplamıdır, 0 + 0 + önyargı ağırlığı. Dolayısıyla, önyargı, tek başına, (0,0) ‘un çizgi ile ilişkili olduğu yönündeki soruyu yanıtlar. Önyargı ağırlığı pozitifse, (0,0) noktası çizginin üstündedir; Negatif ise altındadır. Algılayıcının (0,0) noktasının çizgiye göre konumunu anlamasına “önyargılar” denir.

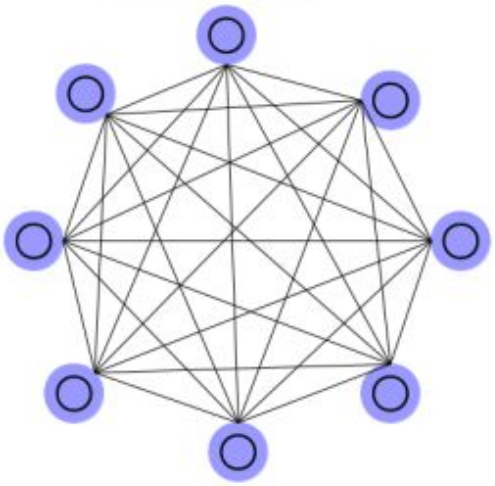
7.3 Sinir Ağ Çeşitleri



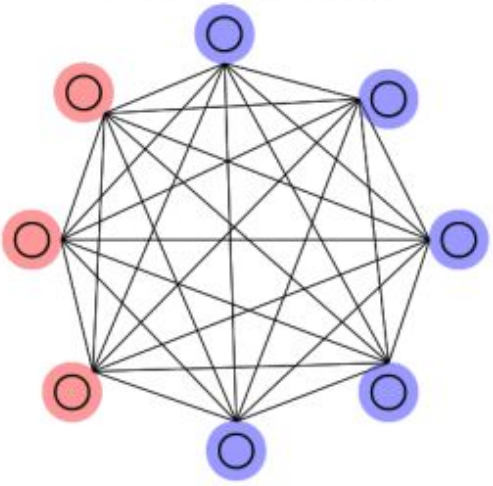
,



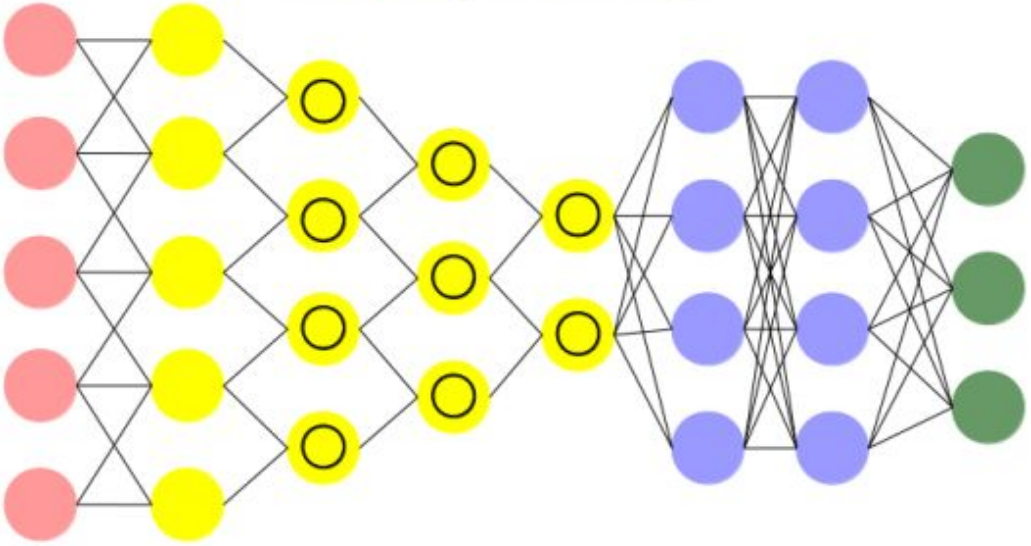
Markov Zinciri



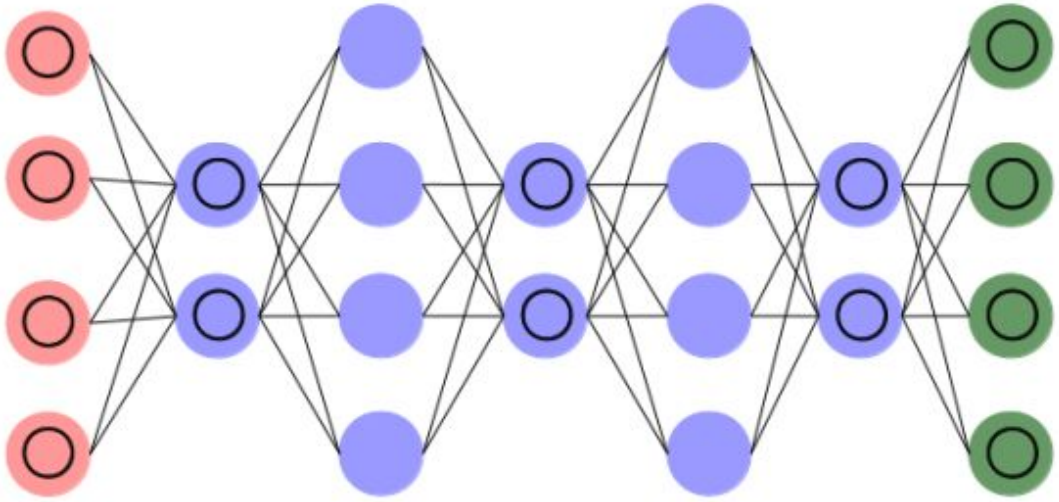
Boltzmann Makinesi



Konvolüsyonel Sinir Ağı



Derin İnanç Ağı



- Arka Uç Giriş Hücresi
- Giriş Hücresi
- Gürültülü Giriş Hücresi
- Gizli Hücre
- Olasılıksız Gizli Hücre
- Sivri Gizli Hücre
- Çıkış Hücresi
- Eşleştirilmiş Giriş-Çıkış Hücresi
- Tekrarlayan Hücre
- Bellek Hücresi
- Farklı Bellek Hücresi
- Çekirdek
- Konvolüsyon veya Gölet

1. Algılayıcı (Perceptron (P))
2. İleri Beslemeli (Feed Forward (FF))
3. Tekrarlayan Sinir Ağı (Recurrent Neural Network (RNN))
4. Derin İleri Beslemeli (Deep Feed Forward (DFF))
5. Uzun / Kısa Süreli Bellek (Long / Short Term Memory (LSTM))
6. Yarıçapsal Temelli Ağ (Radial Basis Network (RBF))
7. Kapılı Tekrarlayan Hücre (Gated Recurrent Unit (GRU))
8. Otomatik Kodlayıcı (Auto Encoder (AE))
9. Varyasyonel Otomatik Kodlayıcı (Variational AE (VAE))
10. Gürültü Giderici Otomatik Kodlayıcı (Denoising AE (DAE))
11. Seyrek Otomatik Kodlayıcı (Sparse AE (SAE))
12. Markov Zinciri (Markov Chain (MC))
13. Hopfield Ağı (Hopfield Network (HN))
14. Boltzmann Makinesi (Boltzmann Machine (BM))
15. Kısıtlı Boltzmann Makinesi (Restricted BM (RBM))
16. Derin İnanç Ağı (Deep Belief Network (DBN))
17. Konvolüsyonel Sinir Ağı (Convolutional Neural Network (CNN))
18. Dekonvolüsyonel Ağ (Deconvolutional Network (DN))
19. Derin Konvolüsyonlu Ters Grafik Ağı (Deep Convolutional Inverse Graphics Network (DCIGN))
20. Üretici Çekişmeli Ağ (Generative Adversarial Network (GAN))
21. Sıvı Hal Makinesi (Liquid State Machine (LSM))
22. Aşırı Öğrenen Makine (Extreme Learning Machine (ELM))
23. Yankılanan Durum Ağı (Echo State Network (ESN))
24. Derin Kalıntı Ağı (Deep Residual Network (DRN))
25. Kohonen Ağı (Kohonen Network (KN))
26. Destek Vektör Makinesi (Support Vector Machine (SVM))
27. Turing Sinir Makinesi (Neural Turing Machine (NTM))