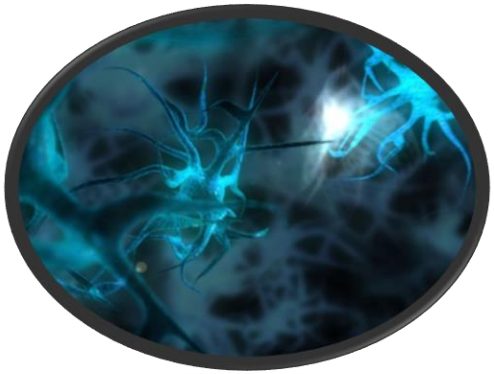


YAPAY SİNİR AĞLARINA GİRİŞ

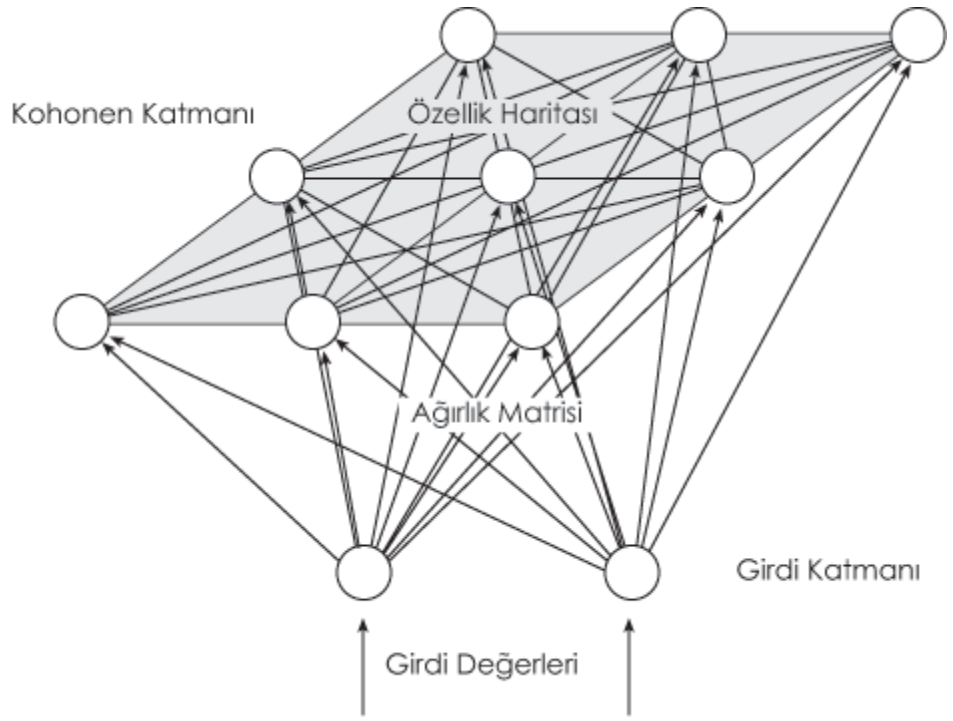
DR. ÖĞR. ÜYESİ BETÜL UZBAŞ

Öz düzenlemeli haritalar (SOM – Self Organizing Maps)

- SOM olarak da adlandırılan Kohonen ağı, beynin Neocortex katmanında yaygın olan duyusal haritalardan esinlenilerek 1972 yılında Kohonen tarafından geliştirilmiştir.

- Öz Düzenlemeli Haritalar (Self Organizing Maps) danışmansız öğrenme kullanan ağlara örnek olarak gösterilebilir.

- Özörgütlemeli harita ağı, tipik olarak iki katmandan oluşur. Girdi katmanı ve iki boyutlu Kohonen çıktı katmanı. Girdi katmanı, Kohonen katmanındaki tüm işlemci elemanlarla bağlantılıdır. Yarışmacı öğrenmeyi kullanan Kohonen ağında kazanan işlemci eleman 1 diğer elemanlar ise 0 değerini alır. Bu stratejiye **Kazanan Hepsini Alır** (Winner-Takes-All) stratejisi denir.



Yapay Sinir Ağları ve Tahmin Modellemesi Üzerine Bir Uygulama, 2006 , İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü.

- Kohonen ağının eğitiminde, herhangi bir *t* zamanında örnek setinden herhangi bir örnek ağa gösterilir. Ağdaki ağırlık değerleri başlangıçta rasgele olarak seçilir. Girdi vektörü ve ağırlık vektörü normalize edilmiş olmalıdır.

- Çıktı elemanlarından kazanan işlemci elemanı bulmak için iki yöntem bulunmaktadır.
  - Bunlardan ilkinde, her elemanın çıktısı girdilerin ağırlıklı toplamı ile bulunur (*yi* = Σ*wixi*). Bu çıktı değerlerinden en yüksek değere sahip olan işlemci eleman yarışmayı kazanır.
  - İkinci yöntemde ise, Öklit mesafesi kullanılarak elde edilen girdi vektörüne en yakın ağırlık vektörüne sahip işlemci eleman kazanan elemandır. $d_i = \sqrt{\sum_j (A_{ij} - x_j)^2}$
  - Her çıktı elemanı için bu mesafeler hesaplanmakta ve en küçük mesafe değerine sahip işlemci eleman kazanan eleman olarak belirlenmektedir.

- Eğitim sırasında hem yarışmayı kazanan işlemci elemanın hem de komşusu olan işlemci elemanların ağırlıkları değiştirilir. Yakın işlemci elemanlar destekleyiciyken (Excitatory), uzak elemanlar engelleyicidir (Inhibitory).

- Kazanan işlemci elemanın belirlenmesinin ardından bu elemanın ve komşularının ağırlıkları değiştirilir. Yani kazanan işlemci elemanın çevresindeki (aynı bölge içindeki) elemanlar kazanan elemanla aynı cevabı vermesi için desteklenir. Eğitim esnasında öğrenme katsayısı sürekli olarak düşürülür ve komşuluk alanı sürekli daraltılır. Yani geniş bölgelerden başlanır, giderek bölgeler daraltılır.

SOM Ağları - Örnek

1. Adım

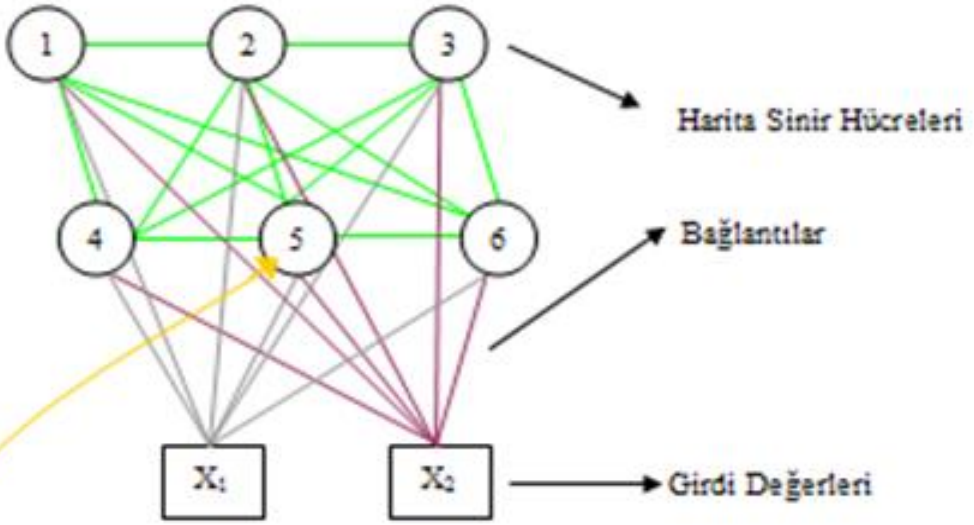
1.1) $X = \begin{bmatrix} 0,52 \\ 0,12 \end{bmatrix}$

$w_1 = \begin{bmatrix} 0,27 \\ 0,81 \end{bmatrix}$, $w_2 = \begin{bmatrix} 0,42 \\ 0,70 \end{bmatrix}$, $w_3 = \begin{bmatrix} 0,43 \\ 0,21 \end{bmatrix}$, $w_4 = \begin{bmatrix} 0,55 \\ 0,72 \end{bmatrix}$, $w_5 = \begin{bmatrix} 0,48 \\ 0,14 \end{bmatrix}$, $w_6 = \begin{bmatrix} 0,69 \\ 0,31 \end{bmatrix}$

Girdi ve ağırlık değerleri (0,1) aralığından rasgele seçilir.

https://slideplayer.biz.tr/slide/15196876/ [Erişim Tarihi: 25.12.2020]

SOM Ağları - Örnek



$$d_1 = \sqrt{(x_1 - w_{11})^2 + (x_2 - w_{21})^2} = \sqrt{(0,52 - 0,27)^2 + (0,12 - 0,81)^2} = 0,73$$

$d_2 = 0,59$

$d_3 = 0,13$

$d_4 = 0,60$

$d_5 = 0,04$ (Kazanan Sinir Hücresi)

$d_6 = 0,25$

SOM Ağları - Örnek

En küçük uzaklık öklid kriteri kullanılarak en çok etkinleşen (en iyi) sinir hücresi bulunur. Buna göre ağırlıklarda yapılacak değişim miktarları hesaplanır.

1.3) $\alpha = 0,2$ ⟶ Öğrenme oranı

Ağırlıklarda yapılacak değişim miktarları hesaplanır.

$\Delta w_{1s} = \alpha (x_1 - w_{1s}) = 0,2 (0,52 - 0,48) = 0,008$

$\Delta w_{2s} = \alpha (x_2 - w_{2s}) = 0,2 (0,12 - 0,14) = -0,004$

Bu aşamada kazanan sinir hücresi ve komşularının ağırlıkları güncellenir.

1.4) $p = 1$ iken,

$$w_s (p+1) = w_s (p) + \Delta w_s (p) = \begin{bmatrix} 0,48 \\ 0,14 \end{bmatrix} + \begin{bmatrix} 0,008 \\ -0.004 \end{bmatrix} = \begin{bmatrix} 0,488 \\ 0,136 \end{bmatrix}$$

https://slideplayer.biz.tr/slide/15196876/ [Erişim Tarihi: 25.12.2020]

- **Destekleyici Öğrenme:** Bazı durumlarda ağa çıktının ne olduğunu vermek mümkün olmamaktadır, fakat ağın doğru veya yanlış olduğu belirtilebilir.

- Destekleyici Model olarak kullanılan yöntemlerden bir tanesi de LVQ(Lineer Vector Quantization) modelidir.

LVQ

- LVQ Kohonen tarafından 1984 yılında geliştirilmiştir, destekleyici öğrenme stratejisi kullanır.

- Eğitim sırasında ağa sadece öğrenilmesi istenen girdiler verilmekte ve ağın çıktıyı üretmesi istenmektedir. Ağ çıktıyı ürettikten sonra ağa sunulan girdi vektörüne karşılık gelen ağın ürettiği çıktının doğru veya yanlış olup olmadığı söylenmektedir.

- LVQ ağının temel felsefesi ağa sunulan girdi vektörünü problem uzayını temsil eden referans vektörlerinden birisine haritalamaktır.

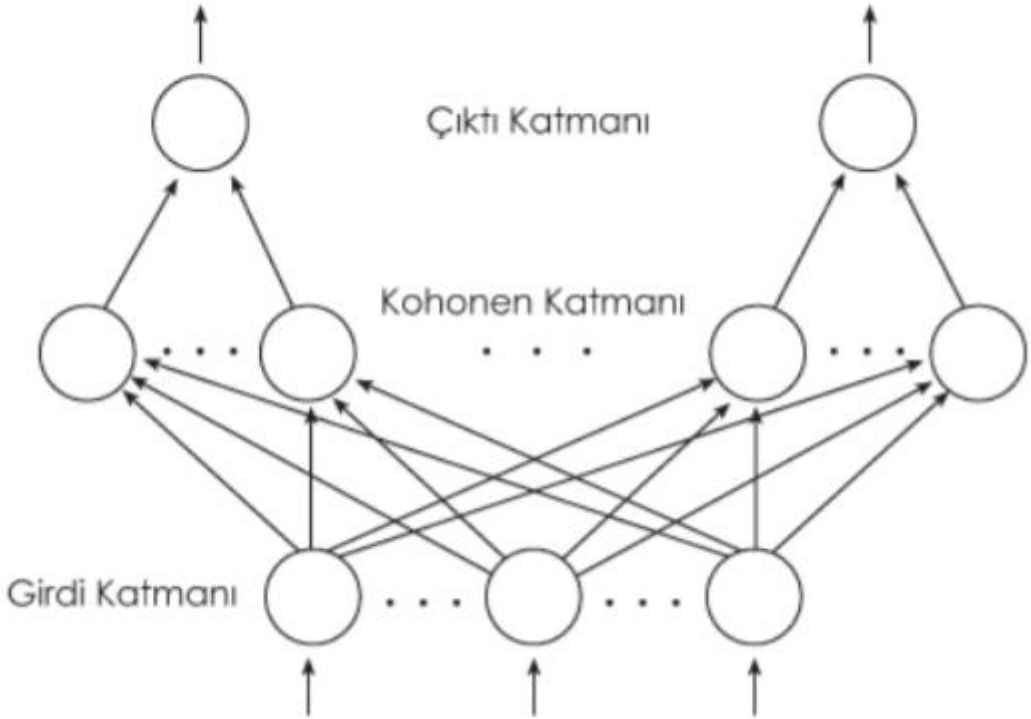
- LVQ ağı 3 katmandan oluşur.

- Girdi Katmanı
- Kohonen Katmanı
- Çıktı Katmanı

- Girdi katmanındaki her eleman Kohonen katmanına bağlıdır. Girdi katmanından Kohonen katmanına bağlantıların ağırlıkları bir referans vektörünü oluşturur. Öğrenme sırasında sadece bu referans vektörün değerleri (ağırlık) değiştrilir.

- Kohonen katmanındaki elemanlaın her biri çıktı katmanındaki sadece tek bir elemana bağlıdır.Kohonen katmanı ile Çıktı katmanındaki ağırlık sabit olup değeri 1'dir. Bu ağırlıklatın değerleri eğitim sırasında değiştirlimez.

Tez Kohonen Referans



Yapay Sinir Ağları ve Tahmin Modellemesi Üzerine Bir Uygulama, 2006 , İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü.

- LVQ ağlarında öğrenme girdi vektörü ile referans vektörleri arasındaki öklid mesafesine dayanır. Kohonen katmanındaki her eleman bir referans vöktörünü göstermekte olup birbirleri ile yarışırlar. Öklid mesafesi en kısa olan yarışı kazanır. Çıktı değerlerinin belirlenmesinde «kazana herşey alır» (winner takes all) stratejisi uygulanmaktadır. Kazanan elemanın çıktısı 1, diğerleri 0 olur.

- Yarşmayı kazanan çıktı elemanı ilgili girdinin sınıfını gösterir. Eğer girdi doğru sınıflandırılmışsa ilgili referans vektörü girdi vektörüne yaklaştırılır. Değilse uzaklaştırılır. Yaklaştırma ve uzaklaştırma öğrenme katsayısıyla gerçekleştirilir.

LVQ Çalışma Prosedürü

- 1. Örneklerin belirlenmesi
- 2. Ağın topolojisinin belirlenmesi
- 3. Ağın öğrenme parametresi belirlenmesi
- 4. Ağırlıkların başlangıç değerlerinin atanması
- 5. Öğrenme setinden bir örneğin ağa gösterilmesi
- 6. Kazanan proses elemanın bulunması
- 7. Ağırlıklrın değiştirlmesi
- 8. Bütün örnekler doğru sınıflandırılıncaya kadar yukarıdaki adımların (5-7) tekrar edilmesi

LVQ öğrenme kuralı

- Kohonen kuralı da denilmektedir.
- Girdi vektörü ile ağırlık vektörü (referans vektörler) arasındaki öklid mesafesi hesaplanmasına dayanır. Hangi proses elemanın referans vektörü girdi vektörüne en yakınsa o kazanır.

$$d_i = \sqrt{\sum_j (A_{ij} - x_j)^2}$$

- Öğrenme sırasında sadece girdi katmanını bu proses elemanına bağlayan ağırlık değerleri değiştirilir.

- Kazanan proses doğru sınıfa aitse ağırlıklar girdi vektörüne daha yaklaştırılır.

$$A_y\_A_e + \lambda(x - A_e)$$

- Kazanan proses yanlış sınıftaysa ağırlık vektörü girdi vektöründen uzaklaştırılır.

$$A_y\_A_e - \lambda(x - A_e)$$

- LVQ modelinin bazı dezavantajları bulunmaktadır. Bunlardan ilki öğrenme katsayısının zamanında 0 değerini almaması durumunda ağın doğru ağırlık değerlerinden uzaklaşmasıdır. Ayrıca bazı problemlerde sürekli aynı referans vektörü yarışmayı kazanmakta olduğundan ağın esnekliği ortadan kalmaktadır.

- Diğer dezavantaj ise sınıflandırmayı yaparken iki sınıfın tam ortasında veya sınırlara çok yakın bulunan vektörlerin hangi sınıfa gireceklerinin belirlenememesidir. Bu sorunları ortadan kaldırmak için LVQ2, Ceza Mekanizmalı LVQ ve LVQ-X modelleri geliştirilmiştir.

LVQ2

- Kohonen tarafından geliştirilmiştir. LVQ2, standart LVQ modelinin uygulanmasından sonra elde edilen çözümün iyileştirilmesi için kullanılır. Sınır değerlerdeki vektörlerin yanlış sınıflandırılmasını önlemeye çalışır.
- LVQ'nun aksine, aynı anda 2 referans vektörünün ağırlıkları değiştirilir. Bu vektörlere A1 ve A2 denirse, bunların ağırlıklarının değiştirilmesi için iki koşulun sağlanması gerekmektedir.

- 1. A1 girdi vektörüne en yakın ağırlık vektörü, A2 ise ondan sonraki en yakın ağırlık vektörüdür. A1 (en yakın olan) yanlış, A2 doğru sınıftadır.
- 2. Girdi vektörü, bu iki vektörün arasında merkezi olarak belirlenmiş bir aralık içerisinde kalmaktadır.

- Bu durumda iki vektörün de ağırlıkları değiştirilir.

Ceza Mekanizmalı LVQ

- Standart LVQ ağlarında bazı ağırlık vektörleri sürekli kazanmakta ve bu da ağın esnekliğini bozmakta, diğer ağırlık vektörleri referans vektörü olma niteliklerini kaybetmektedir. Belirlenen her ağırlık vektörü girdinin bir yönünü göstermesi gerekirken sadece bir tanesi tüm sorumluluğu üstlenmektedir.

- Bu problemden kurtulmak için 1988 yılında DeSieno tarafından Ceza Mekanizmalı LVQ modeli geliştirilmiştir. Bu modelde sürekli kazanan ağırlık vektörü cezalandırılmakta ve peş peşe kazanması engellenmektedir. Sürekli kazanan vektöre kaç defa kazandığı ile ilgili olarak uzaklığına bir değer eklenir, yani uzaklaştırılır. Bu sayede diğer vektörlere de kazanma şansı tanınmaktadır.

LVQ-X Modeli

- LVQ2 modelinde olduğu gibi aynı anda iki ağırlık vektörünün değerlerini değiştirmekte, bu sayede hem öğrenme hızı hem de ağın genelleme yeteneği arttırılmaktadır.

- LVQ-X kuralı Öztemel tarafından geliştirilmiştir. Her iterasyonda mutlak kazanan ve yerel kazanan olmak üzere iki işlemci eleman seçilir.

- Mutlak kazanan girdi vektörüne en yakın ağırlık vektörü, yerel kazanan ise, doğru sınıf içerisinde girdi vektörüne en yakın ağırlık vektörüdür.

- Eğer mutlak kazanan işlemci eleman doğru sınıf içerisinde değilse onun ağırlık vektörü o girdi vektöründen uzaklaştırılır. Aynı zamanda doğru sınıf içerisindeki en yakın olan işlemci elemanın (yerel kazanan) ağırlık vektörü de girdi vektörüne yaklaştırılır. Bu doğru işlemci elemana daha sonraki iterasyonlarda kazanma şansı vermektedir.

- Eğer mutlak ve yerel kazanan aynı işlemci eleman ise o zaman tek bir ağırlık vektörü değiştirilmekte ve bu ağırlık vektörü girdi vektörüne yaklaştırılmaktadır.

- **Öztemel, E.**, 2003. *Yapay Sinir Ağları*, Papatya Yayıncılık, İstanbul.

- Yapay Sinir Ağları ve Tahmin Modellemesi Üzerine Bir Uygulama, 2006 , İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü.

- https://slideplayer.biz.tr/slide/15196876/ [Erişim Tarihi: 25.12.2020]