YAPAY ZEKÂ, KAVRAMLAR, YÖNTEMLER VE UYGULAMALAR

Doç. Dr. Ali Ekber Özdemir
Elektrik-Elektronik Mühendisi
Ordu Üniversitesi, Fatsa Deniz Bilimleri Fakültesi
aliekber.ozdemir@emo.org.tr

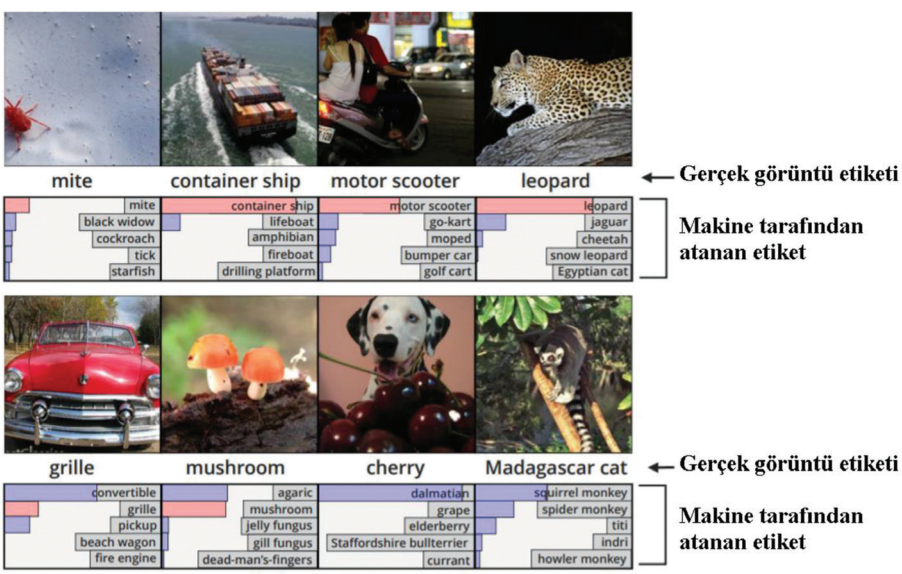
Bu yazıda gelişen teknoloji ile her geçen gün daha sık işitmeye başlanan yapay zekâ kavramı, bu kavram altında yaygın olarak kullanılan yöntemler ve yapay zekanın kullanım alanlarından bahsedilecektir.

Zekâ kavramı Türk Dil Kurumu sözlüğünde; "İnsanın düşünme, akıl yürütme, objektif gerçekleri algılama, yargılama ve sonuç çıkarma yeteneklerinin tamamı, anlak, dirayet, zeyreklik, feraset" olarak tanımlanmaktadır. Ancak yapay zekâ çok genel ve çoğu zaman yanlış anlaşılan bir kavramdır. En genel tanımı *bilinç kazanmış bir bilgisayar yazılımı* olarak ifade edilebilir. Bu kavram 1900'lü yılların ortalarından beri tartışılmakla birlikte ölçüt olarak Turing Testi altın bir standart olarak kabul edilmiştir.

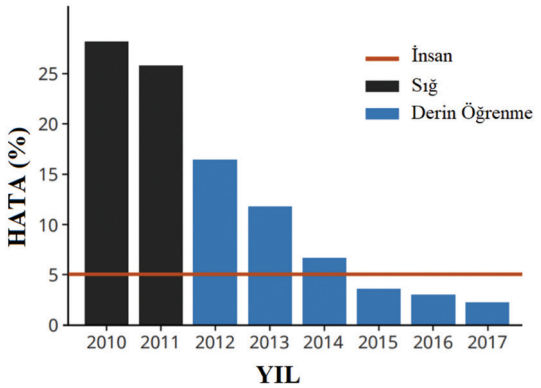
En basit tanımı ile bir yapay zekâ sistemine bir insan operatör tarafından yöneltilen sorulara verilen yanıtların, bu operatör tarafından insan mı yoksa makine tarafından mı verildiğinin anlaşılamaması, o sistemin Turing testini geçme ölçütü olarak kabul edilir. Ancak günümüzdeki yapay zekâ uygulamaları bilinç kazanmış bir bilgisayar sisteminden öte örüntü tanıma, sistem modelleme ve sınıflandırma alanlarında yoğunlaşmıştır.

Aşağıda yapay zekâ uygulamalarından biri olan nesne tanıma uygulaması gösterilmiştir.

Şekil 1 incelendiğinde, geliştirilen yapay zekâ modülünün kendisine, bir veri tabanından, milyonlarca resim içinden rastgele gösterilen görüntülerdeki nesneleri oldukça başarılı bir şekilde tanımlayabildiği görülmektedir. Bu örnekte kullanılan yapay zekâ modülü 600 bin civarında yapay nörondan oluşan oldukça karmaşık bir sistemdir. Bu alanda yapılan çalışmalarda, tanımlama hatası yıldan yıla azalmış ve günümüzde örneğin medikal uygulamalarda uzman doktorlardan bile daha başarılı bir biçimde medikal görüntülerin tanımlanması yapılabilmiştir. Şekil 2'de bu alanda yapılan uygulamalardaki tanımlama hatasının yıllara göre nasıl değiştiği gösterilmiştir.



Şekil 1. Derin Öğrenimle Nesne Tanıma Uygulaması



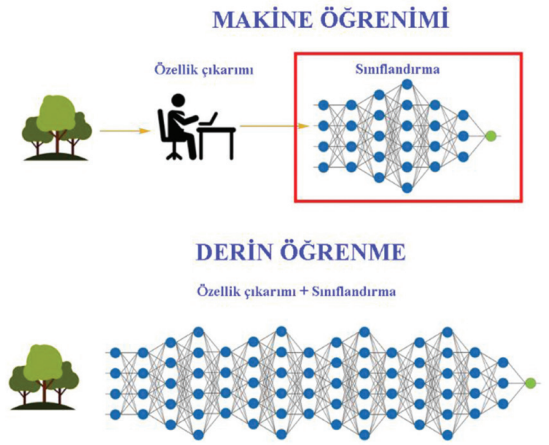
Şekil 2. Yıllar İçerisinde Derin Öğrenme Başarımının Değişimi

Özellikle katlamalı sinir ağlarının (*Convolutional Neural Network*) geliştirilmesi ile birlikte 2015 yılından itibaren elde edilen hata değerleri, insan operatörlerden daha düşük seviyelere çekilebilmiştir. Bu alandaki çığır açan bir diğer gelişme ise son derece karmaşık stratejiler geliştirmeyi gerektiren bilgisayar oyunları alanında olmuştur. Her ne kadar bu alanda en çok bilinen oyun satranç olsa da Uzak Doğu'da oynanan 'Go' isimli oyun satranç ile karşılaştırılamayacak derecede karmaşık oyun stratejileri gerektirmektedir. Bu nedenle geliştirilen bir yapay zekâ modülünün bu oyunda, dünyanın en iyi oyuncusu olarak bilinen kişiyi yenmesi çok büyük bir etki oluşturmuştur.

Yapılan karşılaşmada Google tarafından geliştirilen 'AlphaGo' (Şekil 3) isimli yapay zekâ modülü, üst üste 7 defa bu oyunda dünya şampiyonu olan oyuncuyu yenmeyi başarmıştır. Bu başarı sonucunda derin öğrenmenin daha karmaşık problemlerin çözümünde kullanılabileceği yaygın bir biçimde kabul edilmiştir.

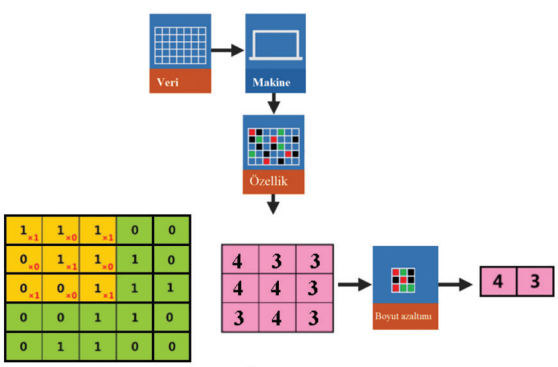
Şekil 3. Derin Öğrenmenin Oyun Alanında Kullanımı

Yapay zekâ kavramının çok genel bir kavram olduğu daha önce belirtilmişti. Çoğunlukla yapay zekâ kavramı, yapay sinir ağları, makine öğrenimi ve derin öğrenme gibi kavramlar ile birlikte kullanılmakta olup bu kavramlar arasında birtakım farklar mevcuttur.



Şekil 4. Derin ve Makine Öğrenimi

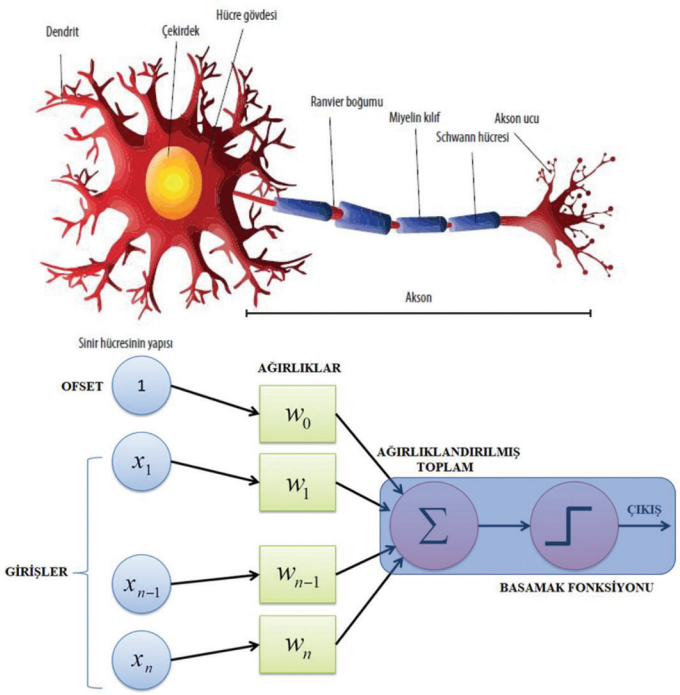
Öncelikle, sınıflandırma, modelleme ya da örüntü tanıma işlemlerini yapan matematiksel modele "Yapay Sinir Ağı (YSA)" denmektedir. Ancak, YSA yapılarının bu işlemleri gerçekleştirebilmeleri için ham veri yerine bu ham veriden türetilmiş özellikleri kullanmaları gerekir. Örneğin, EKG, Electrocardiogram) işaretleri gibi tek boyutlu işaretler üzerinde kalp krizi riskinin değerlendirildiği bir uygulamada ham EKG verilerini kullanmak yerine bu EKG verilerinin Fourier dönüşümünden elde edilen harmoniklerin genlik değerlerinin kullanılması buna örnek verilebilir. Örüntü tanıma yapılacak bir görüntünün ham olarak yapay sinir ağı yapısına verilmesi yerine bu görüntüye bazı filtrelerin uygulanması sonucu elde edilen daha düşük boyutlu verilerin kullanılması, yine özellik çıkarımına başka bir örnektir. İşte özellik çıkarımını kendi başına, insan etkisi ve desteği olmadan yapan sistemler genel olarak "Derin Öğrenme" olarak adlandırılırlar.



Şekil 5. Özellik Çıkartımı

1. Yapay Sinir Ağ Yapısı

Yapay sinir ağları denen yapı ise insan beynindeki sinir hücrelerinin matematiksel modelleri olan yapay sinir nöronlarından oluşan bir ağ yapısıdır.



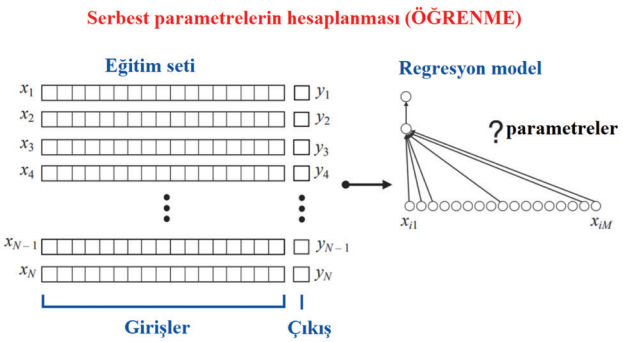
Şekil 6. Biyolojik ve Yapay Sinir Ağı

Bir sinir hücresi dendrit denen kısmı ile diğer sinir hücrelerinin akson uçlarına bağlanır. Dendrit ucundan, diğer sinir hücreleri tarafından gönderilen elektriksel darbeleri alan sinir hücresi bu uyartılara karşı elektriksel bir tepki üretebilir. Eğer bir elektriksel tepki üretilmiş ise bu etki bir elektrik darbesi şeklinde sinir hücresinin akson ucundan diğer sinir hücrelerine aktarılır. Bu duruma sinir hücresinin ateşlenmesi denir. Bir ağ yapısı içerisindeki her bir nöronun ateşlenmesi ya da ateşlenmemesi farklı bir örüntü deseni oluşmasına neden olur ve bu örüntü desenleri yapay sinir ağı çıkışının oluşumunda etkilidir. Yapay bir sinir hücresi bir elektriksel tepki üretip üretmeyeceğine çıkış katında kullanılan bir aktivasyon işlevi ile karar verir. Pek çok farklı aktivasyon işlevi kullanılmakla birlikte yaygın olarak kullanılan aktivasyon işlevleri işaret fonksiyonlarına, birim basamak, tanjant hiperbolik fonksiyonları örnek olarak verilebilir. Tablo 1'de yaygın kullanılan aktivasyon işlevleri verilmiştir.

En basit hali ile bir yapay sinir nöronu, girişleri üzerinden aldığı değerleri ağırlıklandırır ve bu ağırlıklandırılmış değerlerin toplamını bir aktivasyon fonksiyonundan geçirir. Bu değer o nöronun çıkış değeri olarak, ağ yapısı içerisindeki diğer nöronlara giriş olarak iletilir. İşte öğrenme denen süreç, bir ağ yapısı içerisindeki serbest parametrelerin, istenen giriş çıkış ilişkisini sağlayacak şekilde hesaplanmasıdır. Bu örnekte, serbest parametreler yapay sinir ağına ait ağırlık katsayılarıdır.

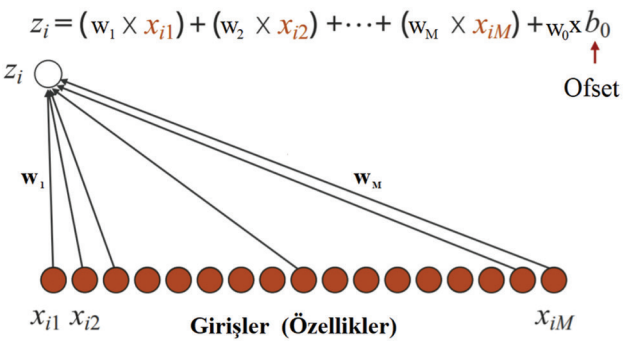
Tablo 1. Aktivasyon İşlevleri

Aktivasyon işlevi	Matematiksel ifadesi	Aralığı
Doğrusal Fonksiyon	$f(x) = x$	$(-\infty, \infty)$
Basamak Fonksiyonu	$f(x) = \begin{cases} 0 \text{ için } x < 0 \\ 1 \text{ için } x \geq 0 \end{cases}$	$\{0, 1\}$
Sigmoid Fonksiyon	$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$	$(0, 1)$
Hiperbolik Tanjant Fonksiyonu	$f(x) = \tanh(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}$	$(-1, 1)$
ReLU	$f(x) = \begin{cases} 0 \text{ için } x < 0 \\ x \text{ için } x \geq 0 \end{cases}$	$[0, \infty)$
Leaky (Sızıntı) ReLU	$f(x) = \begin{cases} 0.01 \text{ için } x < 0 \\ x \text{ için } x \geq 0 \end{cases}$	$(-\infty, \infty)$
Swish Fonksiyonu	$f(x) = 2x\sigma(\beta x) = \begin{cases} \beta = 0 \text{ için } f(x) = x \\ \beta \to \infty \text{ için } f(x) = 2\max(0, x) \end{cases}$	$(-\infty, \infty)$



Şekil 7. Ağ Parametrelerin Hesaplanması

Tek bir yapay sinir ağı nöronundan oluşan yapıya "Regresyon Modeli" denir.



Şekil 8. Regresyon Modeli

Şekil 8'de görüldüğü üzere bu modelin serbest parametreleri w ile gösterilen ağırlık parametreleridir.

2. Yapay Sinir Ağlarının Gerçek Bir Probleme Uygulanması

Konunun anlaşılabilmesi için aşağıda, yağış olup olmayacağının tahminini yapan bir hava durumu modeli verilmiştir.

Görüldüğü üzere model girişleri bulut, nem, sıcaklık ve basınç parametreleri olup bu parametrelerin farklı değerleri için yağmur yağması veya yağmaması durumları 1 ve 0 değerleri ile etiketlendirilmiştir. Tek bir nöron için bu değerler aynı ağırlık değerleri ile çarpılarak y = x olarak ifade edilen doğrusal aktivasyon işlevinden geçirilmiştir.

Sonuç olarak, bu ağırlık değerleri yukarıda gösterildiği gibi bir matris tersi alma işlemi ile bulunabilir. Kullanışlı bir model olsa bile regresyon modelinin belli kısıtları vardır. En önemli kısıt ise regresyon modelinin doğrusal karar yüzeyleri oluşturmasıdır. Karar yüzeyi kavramı hava durumu tahmini için şu şekilde özetlenebilir; Yapılan her bir gözlem ya da toplanan her bir veri çözüm uzayında 1 veya 0 verisi ile etiketlenir. Karar yüzeyi bu verileri birbirinden ayıran sınır çizgileridir. Başarılı bir modelleme için kullanılan ağ yapısı, 1 ve 0 verilerini doğru şekilde birbirinden ayırmayı başarmalıdır. Ancak, bazı durumlarda çıkış verileri doğrusal karar yüzeyleri

Tablo 2. Hava Durumu Tahmini

x_i= i.gün için özellik vektörü

Özellikler / Sonuç

Bulut	Nem	Sıcaklık(F)	Basınç	Yağmur
0.5	80%	75	1.2	1
0.2	95%	83	1.3	0

y_i = 1, Evet
y_i = 0, Hayır

Hava durumu — Yağmur ?

$z_1 = (w_1 x 0.5) + (w_2 x 0.8) + (w_3 x 75) + (w_4 x 1.2) + w_0 x b_0 \quad y_1 = 1$
$z_1 = (w_1 x 0.2) + (w_2 x 0.95) + (w_3 x 83) + (w_4 x 1.3) + w_0 x b_0 \quad y_2 = 0$

x_i= i.gün için özellik vektörü

Özellikler / Sonuç

Bulut	Nem	Sıcaklık(F)	Basınç	Yağmur
0.5	80%	75	1.2	1
0.2	95%	83	1.3	0

y_i = 1, Evet
y_i = 0, Hayır

Hava durumu — Yağmur ?

$z_1 = (w_1 x 0.5) + (w_2 x 0.8) + (w_3 x 75) + (w_4 x 1.2) + w_0 x b_0 \quad y_1 = 1$
$z_1 = (w_1 x 0.2) + (w_2 x 0.95) + (w_3 x 83) + (w_4 x 1.3) + w_0 x b_0 \quad y_2 = 0$

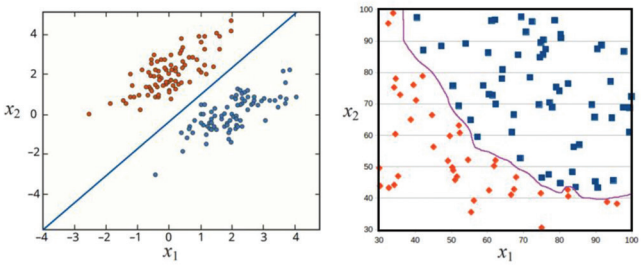
$$\begin{bmatrix} 0.5 & 0.8 & 75 & 1.2 & 1 \\ 0.2 & 0.95 & 83 & 1.3 & 1 \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

ile ayrılamayacak derecede karmaşık olabilir. İşte bu türdeki verilerin regresyon modeli ile ayrılması mümkün değildir.

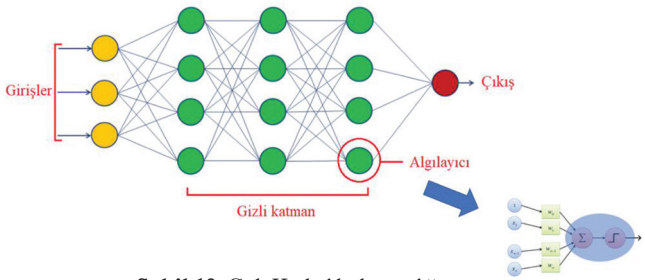
Bu durumda, doğrusal olmayan karar yüzeyleri oluşturabilen çok katlı nöron ağlarının kullanılması gereklidir. İşte bu yapılara "Çok Katmanlı Algılayıcı" ağları denmektedir.

Bu durumda, değeri hesaplanacak olan parametre sayısı, ağdaki nöron ya da algılayıcı sayısına bağlı olarak artacaktır. Ağın başarımının tespiti için de farklı kriterler kullanılmaktadır,

Bu kriterler, ağa verilen her bir veri için ağın ürettiği hataların karelerinin ya da mutlak değerlerinin kullanıldığı işlev çıkışlarıdır. Hataların karelerinin ya da mutlak değerlerinin kullanılmasının sebebi iyi eğitilmiş bir ağ için hata değerinin 0'a yakınsamasının sağlanmasıdır. Çünkü hatanın değerinin negatif ya da pozitif büyük bir değer olması istenmez. Aşağıda, çok katlı algılayıcı ağları için bir örnek verilmiştir.



Şekil 9. Karmaşık ve Doğrusal Karar Yüzeyleri



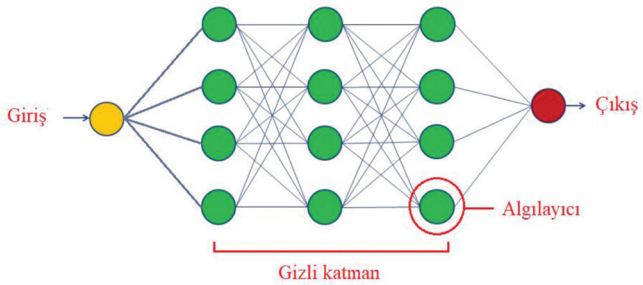
Şekil 10. Çok Katlı Algılayıcı Ağı

Tablo 3. Eğitim Sürecinde Kullanılan Başarım Ölçütleri

Ortalama karesel hata (Mean squared error)	$\mathrm{MSE} = \frac{1}{n}\sum_{t=1}^{n} e_t^2$
Karekök ortalama karesel hata (Root mean squared error)	$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2}$
Ortalama mutlak hata (Mean absolute error)	$\mathrm{MAE} = \frac{1}{n}\sum_{t=1}^{n} \lvert e_t \rvert$
Yüzdelik ortalama mutlak hata (Mean absolute percentage error)	$\mathrm{MAPE} = \frac{100\%}{n}\sum_{t=1}^{n} \left\lvert \frac{e_t}{y_t} \right\rvert$

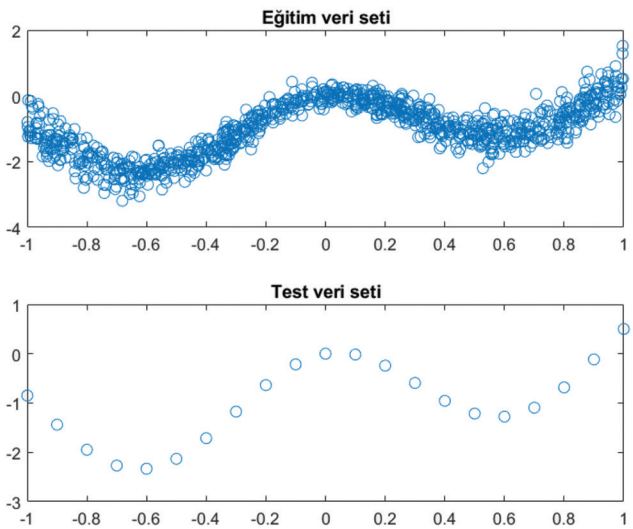
```
% Eğitim için veri setinin oluşturulması
ornek_sayisi = 1000;
a = -1;
b =  1;
ex = a+(b-a).*rand(ornek_sayisi,1);
ey = -0.5*ex.^3+1*cos(ex*5)+exp(ex)-2+0.2*(abs(ex)+1).*randn(ornek_sayisi,1);
% Test için veri setinin oluşturulması
tx = (a:0.1:b)';
ty = -0.5*tx.^3+1*cos(tx*5)+exp(tx)-2;
% Ağın oluşturulması ve eğitimi
noron_sayisi = [4 4 4];
net_ilk = feedforwardnet(noron_sayisi);
net_son = train(net_ilk,ex',ey');
%Eğitim ve test verileri için ağ çıkışlarının gösterilmesi
subplot(211);plot(ex,ey,'o');hold on; title('Eğitim verileri ve ağ çıkışı')
subplot(211);plot(ex,net_son(ex'),'r*');
subplot(212);plot(tx,ty,'o');hold on; title('Test verileri ve ağ çıkışı')
subplot(212);plot(tx,net_son(tx'),'r*');
figure;
subplot(211);plot(ex,ey,'o');title('Eğitim veri seti')
subplot(212);plot(tx,ty,'o');title('Test veri seti')

mse_egitim = perform(net_son, ey', net_son(ex'))
mse_test = perform(net_son, ty', net_son(tx'))
```

Bu örnekte, üstel değişen bir fonksiyona gürültü eklenmiş ve yapay sinir ağının gürültüsüz fonksiyona yakınsaması amaçlanmıştır. Örneğin her katmanında 4 adet nöron bulunan 3 katlı bir ağ yapısı kullanılmıştır. Kullanılan ağ yapısı Şekil 11' de gösterilmiştir.



Şekil 11. Örnek Uygulama İçin Seçilen Ağ Yapısı

Ağın eğitimi için gürültülü veriler kullanılmış ve test etmek için gürültüsüz veriler ağa sorulmuştur. Eğitim ve test verileri, Şekil 12' de gösterildiği gibidir.

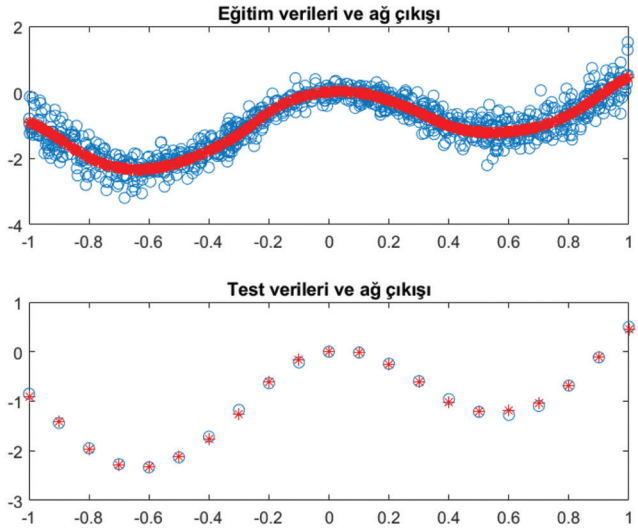


Şekil 12. Eğitim ve Test Verileri

Eğitim tamamlandıktan sonra ağın eğitim ve test verileri için ürettiği çıkışlar Şekil 13'te gösterilmiştir.

Serbest parametrelerin hesaplanması için en yaygın kullanılan yöntem Gradyan Azalması yöntemidir.

Gradyan Azalması Algoritması, amaç fonksiyonu olarak belirlenen hata kare fonksiyonunun türev operatörü kullanılarak minimize edilmesidir. Bilindiği üzere, karesel bir ifadenin alabileceği en küçük değer sıfırdır. Yukarıdaki şekilde 2 adet serbest parametre-



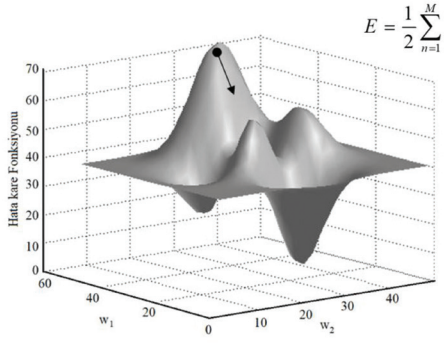
Şekil 13. Eğitim ve Test Sonuçları

nin çözüm uzayında aldıkları olası değerler için hata kare fonksiyonunun aldığı değerler gösterilmiştir. Bu örnekte de görüldüğü gibi türevin 0 değerini aldığı yerel minimum değerleri olmakta ve bu algoritmanın en büyük dezavantajı bu yerel minimum değerlerine takılma olasılığıdır. Her bir adımda amaç fonksiyonunun türevi alınır ve türevin gösterdiği yönün aksi istikamette adım adım ilerlenir, her adımda serbest parametreler bu yön doğrultusunda güncellenir. Bu nedenle Gradyan Azalma Algoritması yinelemeli bir algoritmadır.

Görüldüğü gibi hangi parametre güncellenecek ise amaç fonksiyonunun o parametre için türevi alınır, elde edilen değer 0 ile 1 arasında bir katsayı ile çarpılır. Bu katsayıya "Öğrenme Katsayısı" denir ve bu çarpım değeri ilgili parametreden çıkarılarak o parametre güncellenmiş olur.

Gradyan Azalma Algoritması

$$E = \frac{1}{2}\sum_{n=1}^{M}\left(d(n) - y(n)\right)^2$$



$$\frac{dE}{dw} = \frac{1}{2}\frac{d}{dw}(\overline{d(n)} - \overline{y(n)})^2 \Rightarrow \frac{dE}{dw} = \frac{1}{2}\frac{d}{dw}(d(n) - \overline{w}.\overline{O(n)})^2$$

$$\frac{dE}{dw} = -(\frac{1}{2}).(2).(\overline{d(n)} - \overline{w}.\overline{O(n)}).\overline{O(n)} = \overline{\Delta E}$$

$$\overline{w_{(i+1)}} = \overline{w_i} - \eta\,\overline{\Delta E} \qquad \overline{w_{(i+1)}} = \overline{w_i} + \eta.\overline{e}.\overline{O(n)}$$

Yapay zekâ kavramı içerisinde kullanılan pek çok matematiksel model vardır. Çok katmanlı algılayıcı ağları bu modellerden yalnızca biridir. Radyal tabanlı fonksiyon ağları, ANFIS, dalgacık ağları vb. gibi diğer yöntemler yapılarına uygun farklı öğrenme algoritmaları kullanabilirler. Örneğin, dik en küçük kareler yöntemi (OLS), Levenberg – Marquardt (LM), Gustafson Kessel (GK) vb. gibi algoritmaların yanı sıra bu algoritmalardan bazılarının birlikte kullanıldığı hibrit öğrenme algoritmaları da mevcuttur.

Kaynaklar

1. X. Wang et al., "UD-MIL: Uncertainty-Driven Deep Multiple Instance Learning for OCT Image Classification," in IEEE Journal of Biomedical and Health Informatics, vol. 24, no. 12, pp. 3431-3442, Dec. 2020, doi: 10.1109/JBHI.2020.2983730.

2. B. Xue and N. Tong, "Real-World ISAR Object Recognition and Relation Discovery Using Deep Relation Graph Learning," in IEEE Access, vol. 7, pp. 43906-43914, 2019, doi: 10.1109/ACCESS.2019.2896293.

3. T. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng and Y. Ma, "PCANet: A Simple Deep Learning Baseline for Image Classification?," in IEEE Transactions on Image Processing, vol. 24, no. 12, pp. 5017-5032, Dec. 2015, doi: 10.1109/TIP.2015.2475625.

4. L. Ren, J. Lu, J. Feng and J. Zhou, "Uniform and Variational Deep Learning for RGB-D Object Recognition and Person Re-Identification," in IEEE Transactions on Image Processing, vol. 28, no. 10, pp. 4970-4983, Oct. 2019, doi: 10.1109/TIP.2019.2915655.

5. G. Dong and V. Taslimitehrani, "Pattern-Aided Regression Modeling and Prediction Model Analysis," in IEEE Transactions on Knowledge and Data Engineering, vol. 27, no. 9, pp. 2452-2465, 1 Sept. 2015, doi: 10.1109/TKDE.2015.2411609.

6. J. -. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system," in IEEE Transactions on Systems, Man, and Cybernetics, vol. 23, no. 3, pp. 665-685, May-June 1993, doi: 10.1109/21.256541.

7. A. E. Özdemir and İ. Eminoğlu, "A two-pass hybrid training algorithm for RBF networks," 2013 8th International Conference on Electrical and Electronics Engineering (ELECO), 2013, pp. 617-620, doi: 10.1109/ELECO.2013.6713920.