Journal of Computer Science
https://dergipark.org.tr/en/pub/bbd

ISSN,e-ISSN: 2548-1304
Volume:7 , Issue:2 , pp:68-80, 2022
https://doi.org/10.53070/bbd.1194017
Research Paper

Karcı Sinir Ağlarının Uygulaması ve Performans Analizi

Application and Performance Analysis of Karcı Neural Network

Meral Karakurt[1*] ,Elif Aslı Oymak [2] , Hülya Hark [3] ,Mehmet Can Erdoğan[4] ,Ali Karcı [5]

[1*]Bilgisayar Teknolojileri Bölümü, Osmaniye Korkut Ata Üniversitesi Düziçi MYO, Osmaniye, Türkiye

[2]Bilgisayar Mühendisliği Bölümü, İnönü Üniversitesi, Malatya, Türkiye

[3]Bilgisayar Mühendisliği Bölümü, İnönü Üniversitesi, Malatya, Türkiye

[4]Bilgisayar Programcılığı Bölümü, Van Yüzüncü Yıl Üniversitesi, Van, Türkiye

[5]Yazılım Mühendisliği Bölümü, İnönü Üniversitesi, Malatya, Türkiye

(meralkarakurt@osmaniye.edu.tr, elif.oymak@inonu.edu.tr, hhark@kgm.gov.tr,canerdogan@yyu.edu.tr, ali.karci@inonu.edu.tr)

Received:Oct.25,2022 Accepted:Nov.16,2022 Published:Dec.07,2022

***Özetçe*:** Yapay Sinir Ağları (YSA), matematik ve mühendislik problemlerinin çözümünde sıkça kullanılmaktadır. YSA, canlıların beyin sinir hücresinden esinlenerek ortaya atılan ve bir ya da birden fazla nöronun belirli disiplin çerçevesinde bir görevi yerine getirmesini sağlayan matematiksel modeldir. YSA'ların eğitimi aşamasında probleme göre değişebilen gizli katman sayısı, ağırlıklar, öğrenme katsayısı ve daha birçok parametre kullanılmaktadır. Bu çalışmada, Karcı Sinir Ağı (Karcı Neural Network -Karcı NN) olarak adlandırılan ve YSA'nın öğrenmesi aşamasında kullanılan öğrenme katsayısının kullanımı yerine kesir dereceli türev kullanılan yeni bir hesaplama yöntemi kullanılmıştır. Karcı NN yöntemiyle yapılan deneysel çalışmalar sonucunda, özellikle alfa parametresinin 1.4 değeri için hata oranının % 0.019 olarak ölçüldüğü başarılı sonuçlar elde edilmiştir.

Anahtar Kelimeler: Yapay Sinir Ağları, Öğrenme Katsayısı, Karcı NN.

Abstract: Artificial Neural Networks (ANNs) are frequently used in solving mathematical and engineering problems. ANN is a mathematical model inspired by the brain nerve cell of living things and enables one or more neurons to perform a task within the framework of a specific discipline. During the training of ANNs, the number of hidden layers, weights, learning coefficients, and many other parameters that can change according to the problem are used. In this study, a new calculation method called Karcı Neural Network (Karcı NN) and which uses fractional derivatives instead of the learning coefficient used in the learning phase of ANN, was used. As a result of the experimental studies carried out with the Karcı NN method, successful results were obtained, especially for the 1.4 value of the alpha parameter, where the error rate was measured as 0.019%.

Keywords: Artificial Neural Networks, Learning Coefficient, Karcı NN.

1. Giriş

Günümüzde teknolojinin büyük bir hızla gelişmekte olmasından dolayı birçok bilgi elektronik ortama aktarılmakta, birçok probleme bilgisayar destekli çözümler aranmakta, verilerin ve problemlerin büyüklüğünden ve karmaşıklığından dolayı insan beyninin işlem hacmi artık yetersiz kalmaktadır. Bilgisayar destekli sistemlere, insan beyni gibi düşünebilme yeteneğini kazandırma fikri uzun süredir uygulanmaya çalışılmaktadır.

Tarihte bilinen ilk YSA çalışmasını, McCulloch ve Pitts 1943 yılında, yapay sinir hücrelerini elektrik devreleri üzerine tasarlamakla gerçekleştirmişlerdir (McCulloch ve Pitts, 1943). Böylece, her türlü mantıksal ifadenin matematiksel olarak ifade edilebileceğini göstermişlerdir.

1958'de Rosenblatt tarafından perceptron (algılayıcı) adı verilen tek hücreli bir yapay sinir ağı tasarlanmıştır. Perceptron, biyolojik tek bir nöronun matematiksel modellenmesidir. Perceptron, belli girdi değerlerine karşılık

çıktı değeri üreterek sınıflandırma ve tahmin problemlerinin çözümünde kullanılan tek katmanlı bir YSA'dır (Rosenblatt, 1958). Bir perceptron, giriş değerleri, ağırlıklar, giriş değerleri ile ağırlıkların çarpımının toplamı ve aktivasyon fonksiyonundan oluşmaktadır. Tek doğru ile ayrılabilen problemlerin çözümünde kullanılmaktadır. Tek doğru ile ayrılabilen problemlere doğrusal problemler, tek doğru ile ayrılamayan daha karmaşık problemlere de doğrusal olmayan problemler denir. Bazı basit problemlerde (doğrusal problemlerde) tek nöron içeren perceptronlar çözüm üretebilse de gerçek hayattaki çoğu problem (doğrusal olmayan problemler) için girdi katmanı, gizli katman ve çıktı katmanı olmak üzere üç ya da birden fazla girdi katmandan oluşan çok katmanlı algılayıcılara ihtiyaç duyulmaktadır. Günümüzde çok katmanlı algılayıcı kavramı yerine YSA kavramı yaygın olarak kullanılmaktadır.

1980'li yıllarda YSA'ların eğitimi için geri yayılım algoritması kullanılmıştır. Ancak, YSA'lar gerekli donanım ve yazılım eksiklerinden dolayı 2000'li yıllara kadar yaygınlaşamamıştır. 2000'li yıllara gelindiğinde ise bilgisayar teknolojilerindeki gelişmelerin sağladığı imkanlarla Yapay Sinir Ağları teknolojisinde büyük gelişmeler görülmektedir. Günümüzde sınıflandırma ve tahmin problemlerinde yaygın olarak kullanılmaktadır (Akcan, Kartal, 2021).

Motivasyon: Klasik YSA problemlerinin çözümünde, ağın eğitiminde kullanılan öğrenme katsayısı parametresi yerine kesir dereceli türev kullanılan KARCI NN adı verilen yeni bir yöntem önerilmiştir.

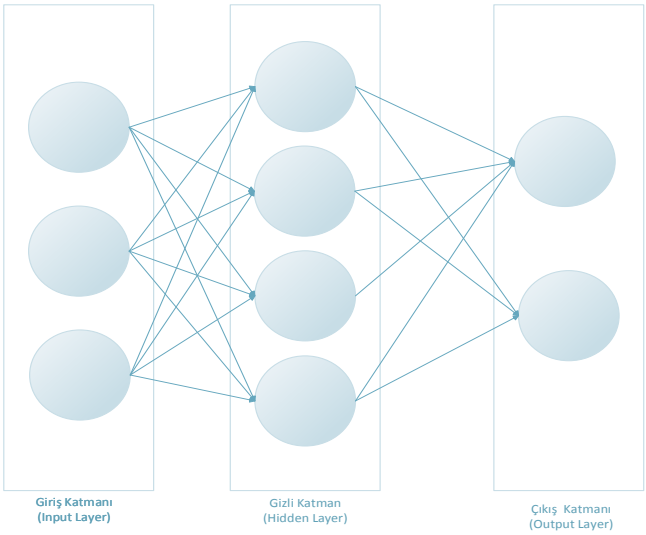
2. Yapay Sinir Ağları

Yapay Sinir Ağları, Yapay Zekâ kavramı altında incelenen, bu konuya ilgi duyan araştırmacıların ilgisini çekmeyi başarmıştır. Yapay Sinir Ağları (Artifical Neural Network - ANN) veya basit kullanımı ile Sinir Ağları (Hardesty, 2017), hayvan beynindeki biyolojik sinir ağlarından ilham alan bilgi işlem sistemleridir (Yang, 2014). Biyolojik bir beyinde nöron denilen sinir hücrelerine karşılık Yapay Sinir Ağlarında yapay sinir hücreleri bulunmaktadır. Yapay sinir hücreleri, mühendislikte işlem elamanları olarak da adlandırılmaktadır (Karcı,2015).

Genel anlamda YSA, bir problemi çözebilmek için beyin sinir hücresini basit bir şekilde taklit eden çalışma yapısına sahip bir yapı olarak tanımlanabilir. Beynin çalışma şekline uygun olarak girdi olarak gelen sinyallerin (verinin) belli bir düzende birbirine bağlı yapay nöronlardan geçerek bilgiye ulaşmayı sağlamaktadır(Ataseven, 2007).

Yapay sinir ağları öğrenme fonksiyonunu gerçekleştiren, adaptif olan elemanların paralel bağlanmasıyla ortaya çıkan ağlardır. Bu ağlar, birbirine ağırlıklı bağlantılar aracılığıyla bağlı nöronlardan oluşmaktadır, yani nöronlar arasındaki her bağlantının bir ağırlık değeri vardır. Öğrenme, bağlantı ağırlıklarının güncellenmesi ile gerçekleşmektedir (Öztemel, 2003; Anonim, 2022). Ağırlıklar, referans değer ile YSA çıktısı arasındaki hata miktarına göre geri yayılım yolu ile eğitilerek değiştirilmektedir (Öztemel, 2003).

Şekil 1 ve Şekil 2'de Yapay Sinir Ağlarının en yalın hali ve matematiksel modeli gösterilmektedir. Matematiksel modelde görüldüğü gibi temel bir YSA, girdiler, gizli katmanda bulunan nöronlar, ağırlıklar, toplama fonksiyonu, aktivasyon fonksiyonu ve çıktılardan oluşmaktadır.



Şekil 1. Temel YSA Örneği (Pişkin, 2022)

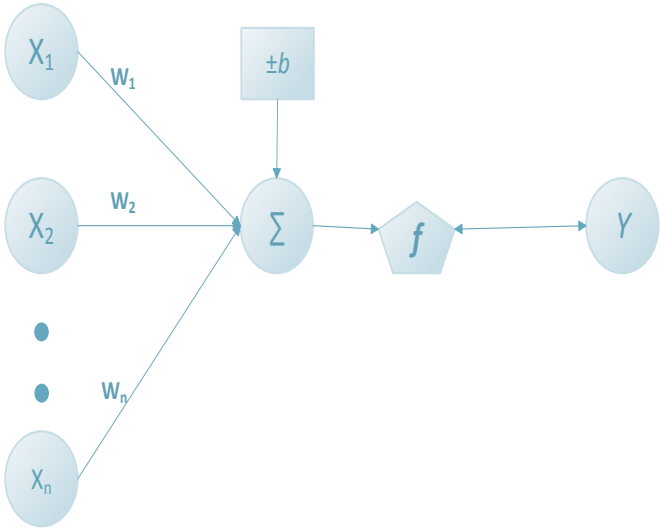
Girdi katmanı, dışarıdan girdileri alan nöronları içerir. Bu katmanda sinir ağına verilecek değerler sayısallaştırılır. Bu girdi değerleri, rastgele üretilen ve daha sonra çözüme yaklaştıracak değerler verilen ağırlıklar ile çarpılarak sonraki katmana, yani gizli katmana iletilmektedir. Burada başka işlem yapılmadığından bazı

araştırmalarda girdi katmanı bir katman olarak değerlendirilmemektedir. Çıktı katmanı ise yapay sinir ağından istenen çıkış bilgisinin veya etiket olarak hesaplan çıktıları dışarı ileten nöronların bulunduğu katmandır. Gizli katman olarak nitelenen kısım probleme göre birden fazla katmana sahip olabilmektedir. Gizli katman bir den fazla nörondan oluşabilmekte ve bu nöronların hepsi diğer katmanlar ile iletişim halindedir. Birçok YSA'da, gizli katmandaki bir nöron sadece bir önceki katmanın tüm nöronlarından sinyal (sayısal değer) almaktadır. Her bir nöron, kendisine gelen nöronun değeri ile o nörona ait ağırlık değerlerini çarpıp belli bir aktivasyon fonksiyonundan geçirdikten sonra kendinden sonraki katmana iletmektedir. Girdi katmanından çıktı katmanına doğru yapılan bu işlemler dizisine ileri besleme denmektedir (Anderson, McNeill,1992; Ergür, 2007). Her katmanın sadece kendinden sonraki katmana bağlantısı bulunmaktadır.

YSA'ları, yapısına veya kullandığı öğrenme algoritmalarına göre sınıflandırmak mümkündür. YSA'da girdi katmanından çıktı katmanına doğru yapılan ve yukarıda tanımlanan işlemler dizisine ileri besleme (feed forward); öğrenmeyi arttırmak veya bulunan çıktı değerini beklenen değere yaklaştırmak için yapılacak değer güncellemelerini elde etmek için çıktı katmanından girdi katmanına doğru yapılan işlemlere ise Şekil 3'te temel algoritması gösterilen geri besleme (feedback, recurrent) işlemi denirİleri beslemeli YSA'da katmanlı yapı olark düzenlenir ve bir katmandaki nöronlar sadece kendinden sonraki katmana üzerinde ağırlıklar üzerinden kendinden sonraki katmana giriş olarak iletilerek gizli katmanda işlenip, çıkış katmanına yönlendirilir. Çıkış katmanında aktivasyon fonksiyonundan geçirilerek ağın çıkışı elde edilir (Ataseven, 2007).

Bir YSA'nın belli girdi değerlerine karşılık belli çıktı değerleri üretmesine denetimli öğrenme, belli girdi değerlerini ortak özelliklere göre (çıktı değerleri belli değildir) kümelendirmesine denetimsiz öğrenme denir.

Bir YSA'yı oluşturan her bir nöron, kendisine gelen sinyalleri bir sonraki katmandaki yapay nöronlara iletmektedir. Biyolojik beyindeki sinyalin yapay sinir ağındaki karşılığı gerçek bir sayıdır. Her bir nöronun çıktısı, Şekil 2'de gösterildiği gibi girdilerin ağırlıklarla çarpımının toplanması ve bu toplamın doğrusal olmayan bir aktivasyon fonksiyondan geçirilmesiyle elde edilmekte ve bu çıktı sonraki nöronlara iletilerek ağın ileri beslemesi yapılmaktadır.



Şekil 2. Temel YSA Hücresi

Çıkış değeri Y, (1) denkleminde gösterildiği gibi W ağırlık değerleri ile X giriş değerlerinin çarpımı şeklinde hesaplanmaktadır.

$$Y = f(WX + b) \tag{1}$$

n, giriş sayısı olarak ifade edilirse W ve X değerleri aşağıdaki gibi yazılabilmektedir.

$$W = W_1, W_2, W_3, \ldots, W_n$$

$$X = X_1, X_2, X_3, \ldots, X_n$$

Bu bilgiler (1) denkleminde yerine yazıldığında, (2) ve (3) denklemlerinde gösterildiği gibi Y'nin genel formülü hesaplanmaktadır.

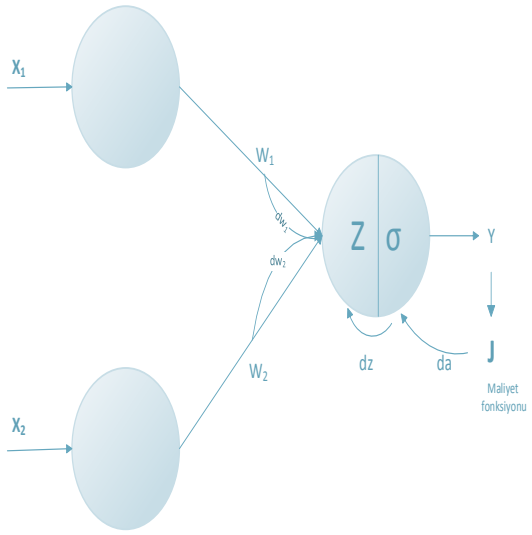
$$A = \sum_{i=1}^{n} W_i X_i + b \quad ve \; Y = f(A) \tag{2}$$

$$Y = f(\sum_{i=1}^{n} W_i X_i + b) \tag{3}$$

Burada kullanılan f aktivasyon fonksiyonunu ifade etmektedir. Aktivasyon fonksiyonu, bir nöronun çıkış değeri üretip üretmeyeceğine (aktif olup olmayacağına) karar vermek için kullanılan bir fonksiyondur (Karakurt ve İşeri, 2022). Örneğin; Step, Linear, Sigmoid ve Hiperbolik Tanjant, ReLu ve Swish ve daha birçok aktivasyon fonksiyonu literatürde yer almaktadır. Bu çalışmada ise, Sigmoid fonksiyonu kullanılmıştır. Sigmoid fonksiyonu, 0 ve 1 aralığında çıkış değeri üretmekte olup türevlenebilir bir fonksiyon olduğundan öğrenme işlemlerinde kullanılmaktadır (Anonim,2022). Matematiksel ifadesi, (4) denklemi ile gösterilmektedir.

$$Sigmoid(x) = \frac{1}{1+e^{-x}} \tag{4}$$

İleri besleme işlemi sonucunda elde edilen çıktının gerçek çıktı miktarından uzaklaşma miktarını ölçmek için bir hata değeri hesaplanır. Yapay sinir ağlarında diğer bir kavram olan geri yayılım, hatayı azaltabilmek için ağırlıklar üzerinde güncelleme yapılabilmesine olanak sunmaktadır. Gradient Descent algoritması türev alarak, hatanın lokal ya da global minimumunu bulabilmeyi sağlamaktadır (Anonim, 2022a). Ağın hata değeri, Şekil 3'te gösterildiği gibi geriye doğru gönderilir ve her girdi değerinin ağırlığı bu hatadan pay alacak şekilde güncellenir. Güncellenmiş ağırlıklar kullanılarak ileri besleme işlemleriyle tekrar çıktı değeri üretilir. Bu işlem, belli bir hata toleransına ya da belli bir iterasyona (eğitim tur sayısına) ulaşıncaya kadar devam eder.



Şekil 3. İki Girişli Bir Ağda Geri Yayılım

J hata fonksiyonu olmak üzere W_1 güncellenmek istenirse;

$$\frac{\partial J}{W_1} = \frac{\partial J}{\partial Y_c} \cdot \frac{\partial Y_c}{\partial Y} \cdot \frac{\partial Y}{W_1} \tag{5}$$

Denklem 5'te ifade edildiği gibi hesaplanması gerekmektedir ve ağırlık güncellemesi

$$W_1 = W_1 - \alpha \Delta W_1 \quad (6)$$

Denklem 6'da belirtilen formül ile yapılmaktadır. Bu güncelleme işleminde α katsayısı öğrenme katsayısı olup rastgele belirlenen bir parametredir. Bu parametrenin rastgele belirlenmesi öğrenme sürecine olumlu yönden mi olumsuz yönden mi etkisi olduğu kesin olarak bilinmemektedir. Günümüzde bu parametrenin rastgele belirlenmesi problem olmaktadır. Bu çalışmada tanımı verilen ağda kullanılmakta olan yeni üretilen parametre rastgele belirlenmemektedir ve tamamen sistemin o adımdaki hatanın kesir dereceli türev tanımından gelmektedir. Bu durum adaptif olduğunda daha olumlu sonuçları olmaktadır.

Yapay sinir ağları mühendislikte birçok probleme başarılı çözümler sunabilmektedir. Birçok nörondan meydana gelen YSA, herhangi bir nöronun işlevsiz olması veya veri girdisinin eksikliğine rağmen kendi kendine öğrenebilmektedir. Eksik bilgi ile çalışabildiği gibi hata toleransına sahip olabilmektedir.

Bunlara rağmen kendi içinde dezavantajlara da sahip olabilmektedir. Çok sayıda nöron ve ağırlık barındırdığından ve çok sayıda ağırlık sayıları ile işlem yapmasından dolayı donanıma bağlı olarak çözüm üretmekte ciddi zaman alabilmektedir. Her problem için uygun ağ yapısının belirlenmesinde (gizli katman sayısı, her bir gizli katmandaki nöron sayısı, öğrenme katsayısı ve ağırlıklar) herhangi bir kural olmadığından dolayı en uygun çözümü bulmak zaman alabilmektedir. YSA'lar sadece sayısal değerler ile çalıştığından, çözülecek problemin YSA'ya sunumunda kesin kurallar olmadığından, problemin ağa sunulması ağın performansını doğrudan etkilemektedir. Ağın ne zaman duracağı hakkında belirgin bir yöntem bulunmamaktadır. Hatanın belli bir eşik değerinin altına düşmesi veya belirlenen adım sayısı(iterasyon) kadar işlem yapılsa bile kesin kurallar bulunmamaktadır (Örneğin: 10 bin adım olarak belirlenen bir YSA'da, bininci adımda çözüme çok yaklaşılmış olabilir. Bu durumda geri kalan 9 bin adım, ağı yormaktan başka bir şey yapmamaktadır.) (Mijwel, 2018).

Ağdaki ağırlıkların optimum aralıkları, kullanılacak gizli katman sayısı ve her bir katmandaki nöron sayısı, her katmanda kullanılabilecek yanlılık (bias) değerinin kullanılıp kullanılmayacağı veya kullanılacaksa hangi aralıklarda olması gerektiği, aynı şekilde öğrenme katsayısının kullanılmasının veya hangi aralıkta en iyi öğrenmeyi sağlayacağı gibi problemler YSA'da çözüm veya iyileştirme bekleyen konuların başında gelmektedir.

3. Karcı Sinir Ağları

Karcı Sinir Ağları, yapay sinir ağlarında güncelleme yaparken sistemden gelen veriyi kullanarak hesaplama yapmakta olup kesir dereceli türevden faydalanarak geri yayılım işleminde dolayısıyla ağırlıkların güncellenmesinde bir **α** öğrenme katsayısına ihtiyaç duymadan hesaplama yapmaktadır. Kesir Dereceli Türev, global modelleme için kullanılmakta olup aşağıdaki gibi ifade edilmektedir. (Karci,2013; Karci, 2015a; Karci, 2015c; Karci, 2019)

$$f^{\alpha}(x) = \left(\frac{f(x)}{x}\right)^{\alpha-1} . f^{1}(x) \quad (7)$$

R, beklenen sonuç değeri ; Z_c, bulunan değer olmak üzere hata hesaplanırken (8) denklemi ile ifade edilen formül kullanılmaktadır.

$$Hata = \frac{1}{n} \sum_{i=1}^{n} (R - Z_c)^2 \quad (8)$$

$W_2(1)$ ağırlığı Newton türevine göre güncellenecek olursa aşağıdaki bağıntı ile ağırlık değişimi bulunur.

$$\frac{\partial Hata}{\partial W_2(1)} = \frac{\partial Hata}{\partial Z_c} . \frac{\partial Z_c}{\partial Z} . \frac{\partial Z}{\partial W_2(1)} = -\frac{1}{n} \sum_{i=1}^{n} (R - Z_c) . Z_c . (1 - Z_c) . Y_{2c} \quad (9)$$

Kesir dereceli türev ile geri besleme için ağırlığın değişim miktarını bulmak amacıyla klasik Newton türevinin başına $\left(\frac{Hata}{W_2(1)}\right)^{\alpha-1}$ çarpanı gelmektedir. Bu çarpan türev tanımından gelmektedir ve öğrenme katsayısı yerine kullanılacaktır. Bu parametre her iterasyonda Hata ve ağırlık değerine göre değişmekte olup dışarıdan müdahaleye gerek kalmamaktadır.

$$D_{\alpha}^{K} \frac{Hata}{W_2(1)} = \left(\frac{Hata}{Z_c}\right)^{\alpha-1} . \frac{\partial Hata}{\partial Z_c} . \left(\frac{Z_c}{Z}\right)^{\alpha-1} . \frac{\partial Z_c}{\partial Z} . \left(\frac{Z}{W_2(1)}\right)^{\alpha-1} . \frac{\partial Z}{\partial W_2(1)}$$

$$= \frac{Hata^{\alpha-1}}{Z_c^{\alpha-1}} . \frac{\partial Hata}{\partial Z_c} . \frac{Z_c^{\alpha-1}}{Z^{\alpha-1}} . \frac{\partial Z_c}{\partial Z} . \frac{Z^{\alpha-1}}{W_2(1)^{\alpha-1}} . \frac{\partial Z}{\partial W_2(1)}$$

$$= Hata^{\alpha-1} . \frac{\partial Hata}{\partial Z_c} . \frac{\partial Z_c}{\partial Z} . \frac{1}{W_2(1)^{\alpha-1}} . \frac{\partial Z}{\partial W_2(1)} \quad (10)$$

elde edilmektedir.

$$W_2(1)' = W_2(1) - D_\alpha^K \frac{Hata}{W_2(1)} \quad (11)$$

bulunmaktadır. D_α^K ifadesi Karcı kesir dereceli türevi temsil etmektedir ve burada kullanılan α parametresi türev derecesi olup öğrenme katsayısı kadar olumsuz etkilere sahip olmadığı yapılan uygulama ile gösterilmektedir.

$W_1(0,2)$ ağırlığı Newton türevine göre güncellenecek olursa Denklem 12'deki bağıntı ile ağırlık değişimi hesaplanmaktadır.

$$\frac{\partial Hata}{\partial W_1(0,2)} = \frac{\partial Hata}{\partial Z_c} . \frac{\partial Z_c}{\partial Z} . \frac{\partial Z}{\partial Y_{3c}} . \frac{\partial Y_{3c}}{\partial Y_3} . \frac{\partial Y_3}{\partial W_1(0,2)}$$

$$= -\frac{1}{n} \sum_{i=1}^{n} (R - Z_c) . Z_c . (1 - Z_c) . W_2(2) . Y_{3c} . (1 - Y_{3c}) . X_3 \quad (12)$$

Aynı ağırlığın Karcı kesir dereceli türev ile geri beslemesi Denklem 13'te belirtilen formül ile hesaplanmaktadır.

$$D_\alpha^K \frac{Hata}{W_1(0,2)} = \left(\frac{Hata}{Z_c}\right)^{\alpha-1} . \frac{\partial Hata}{\partial Z_c} . \left(\frac{Z_c}{Z}\right)^{\alpha-1} . \frac{\partial Z_c}{\partial Z} . \left(\frac{Z}{Y_{3c}}\right)^{\alpha-1} . \frac{\partial Z}{\partial Y_{3c}} . \left(\frac{\partial Y_{3c}}{\partial Y_3}\right)^{\alpha-1} . \frac{\partial Y_{3c}}{\partial Y_3} . \left(\frac{Y_3}{W_1(0,2)}\right)^{\alpha-1} . \frac{\partial Y_3}{\partial W_1(0,2)}$$

$$= \frac{Hata^{\alpha-1}}{Z_c^{\alpha-1}} . \frac{\partial Hata}{\partial Z_c} . \frac{Z_c^{\alpha-1}}{Z^{\alpha-1}} . \frac{\partial Z_c}{\partial Z} . \frac{Z^{\alpha-1}}{Y_{3c}^{\alpha-1}} . \frac{\partial Z}{\partial Y_{3c}} . \frac{Y_{3c}^{\alpha-1}}{Y_3^{\alpha-1}} . \frac{\partial Y_{3c}}{\partial Y_3} . \frac{Y_3^{\alpha-1}}{W_1(0,2)^{\alpha-1}} . \frac{\partial Y_3}{\partial W_1(0,2)}$$

$$= Hata^{\alpha-1} . \frac{\partial Hata}{\partial Z_c} . \frac{\partial Z_c}{\partial Z} . \frac{\partial Z}{\partial Y_{3c}} . \frac{\partial Y_{3c}}{\partial Y_3} . \frac{1}{W_1(0,2)^{\alpha-1}} . \frac{\partial Y_3}{\partial W_1(0,2)} \quad (13)$$

$$W_1(0,2)' = W_1(0,2) - D_\alpha^K \frac{Hata}{W_1(0,2)} \quad (14)$$

Ağırlık güncellenmesi Denklem 12'de ifade edildiği gibi hesaplanmaktadır.

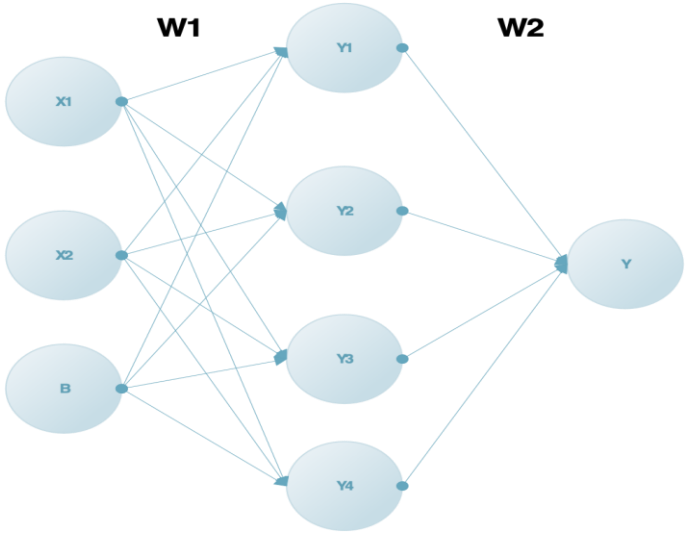
4. XOR Problemi

Çalışmanın bu kısmında geri yayılımlı bir yapay sinir ağı kullanılarak XOR problemi çözülmüştür. XOR problemi, doğrusal olmayan bir problemdir ve denetimli öğrenme yapan çok katmanlı yapay sinir ağları kullanılarak çözülebilmektedir. XOR kapısı aynı giriş değerleri için düşük (0) bir çıktı değeri ve farklı giriş değerleri için yüksek (1) bir çıktı değeri üretmektedir. XOR için doğruluk tablosu, Tablo 1'de gösterildiği gibi X1 ve X2 giriş değerlerine karşılık Y çıktı değerlerinin üretilmesi ile elde edilmektedir.

Tablo 1. Xor Doğruluk Tablosu

X1	X2	Y
0	0	0
0	1	1
1	0	1
1	1	0

XOR kapısı için oluşturulan YSA modeli, Şekil 5'te gösterildiği gibi X1, X2 girdi nöronları, Y1,Y2, Y3 ile Y4 ara katman nöronlarını ve Y çıktı nöronu olmak üzere 3 katmanlı bir yapay sinir ağı mimarisi tasarlanmıştır.



Şekil 4. YSA Modeli

Modellenen YSA'ya ait rastgele oluşturulan ağırlık değerleri Tablo 2'de gösterilmektedir.

Tablo 2. Modellenen Ağa ait Ağırlık Değerleri Tablosu

W1			W2
0.7079939	0.49332882	0.37309297	0.23193633
0.42064022	0.85487726	0.16101624	0.29147887
0.50529493	0.98083863	0.59775229	0.5756014
0.02408863	0.00595876	0.41439012	0.13920469

X1 ve X2 giriş değerleri kullanılarak modellenen ağların eğitimi aşamasında eğitim tur sayısı (iterasyon - epoch) 20000 olarak ayarlanmıştır.

4.1. Bulgular ve Tartışma:

Klasik NN yönteminde kullanılan Öğrenme Katsayısı (Learning Rate – LR) değerleri ve Karcı NN yönteminde kullanılan alfa değerleri için ağların ürettiği çıktı değerleri ve hata miktarları Tablo 3 ile gösterilmektedir. LR=0.4 ve 0.6 değerleri için Klasik NN yönteminde beklenen çıktı değerlerine daha yakın sonuçlar üretilirken alfa=0.4 ve

0.6 değerlerinin kullanıldığı Karcı NN yönteminde, beklenen çıktı değerleri üretilememiştir. Hata değerlerine bakıldığında da ~%2 ve ~%1 hata değerleriyle Klasik NN yönteminin ~%25 hata üreten Karcı NN yönteminden daha iyi çalıştığı görülmektedir.

LR=0.8 değeri için Klasik NN yönteminde hata değeri ~%1 'e düşerken alfa=0.8 değeri için Karcı NN yönteminde hata değeri ~%6'ya düşmektedir.

LR=1.2 değeri için Klasik NN yönteminde hata değeri ~%1 iken alfa=1.2 değeri için Karcı NN yönteminde hata değeri ~%0.03 olarak ölçülmüştür ve Karcı NN yöntemiyle elde edilen Y çıktıları, beklenen değerlere daha iyi yakınsamaktadır. alfa=1.4 ve 1.6 değerlerinin kullanıldığı Karcı NN yönteminin çıktıları ile hata değerlerinin, LR=1.4 ve 1.6 değerlerinin kullanıldığı Klasik NN yönteminin çıktıları ile hata değerlerinden daha iyi olduğu görülmektedir.

LR=1.8 ve alfa=1.8 için her iki yöntemin hata değerleri birbirine yakın olarak ölçülmüştür.

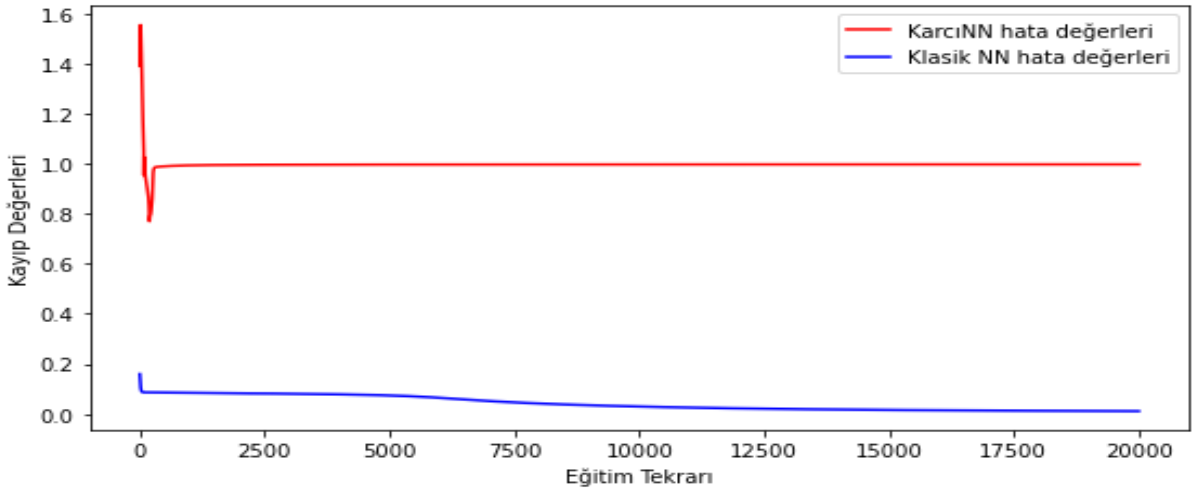
Karcı NN yönteminde alfa=1, alfa=2 değerleri için çıktı değerleri üretilememektedir. alfa=2.2 ve üzeri değerler için de istenmeyen hata ve sonuç değerleri üretilmektedir. Klasik NN yönteminde de LR=4 ila LR= 5.8 aralığındaki değerler ile LR=12 ila LR=60 aralığındaki değerler için çıktı üretilememektedir. LR=6 ila LR=11.8 aralığındaki değerler için çıktı değerlerini ürettiği görülmüştür.

Karcı NN yöntemi, Tablo 3'te görüldüğü gibi alfa= 0.8 ila alfa=1.8 aralığındaki değerler için beklenen çıktı değerlerine Klasik NN yönteminden daha iyi yakınsamaktadır. Aynı aralıkta, Karcı NN yönteminin hata değerlerinin Klasik NN yönteminin hata değerlerinden daha düşük olduğu görülmektedir.

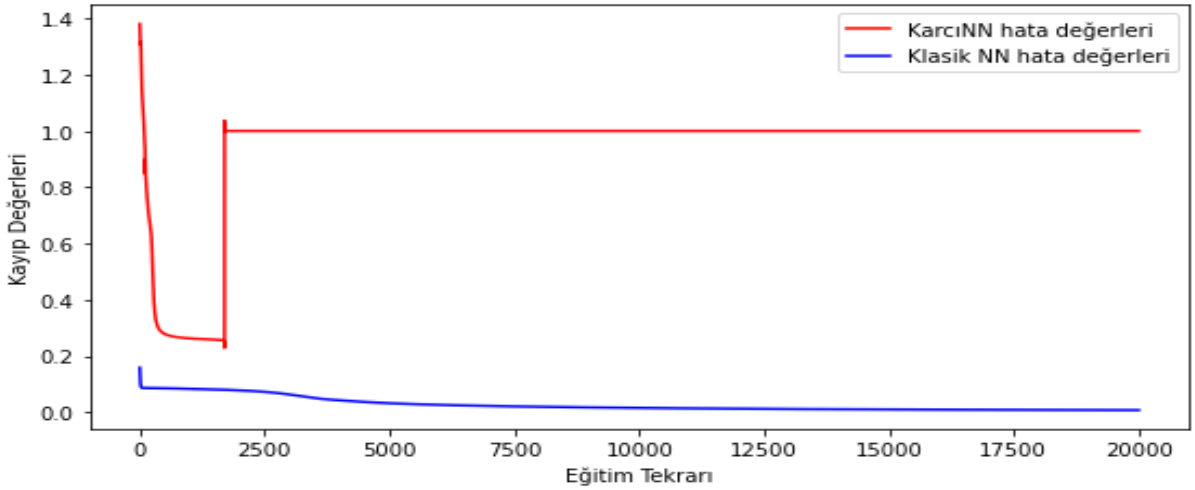
Tablo 3**.** Klasik NN ve KarcıNN Ağlarının Farklı Öğrenme Katsayıları ve Alfa Değerleri için Ürettiği Çıktılar ve Hata Değerleri

Öğrenme Katsayısı ve Alfa Değerleri	X1	X2	Y_Klasik	Y_KarcıNN	Hata_Klasik	Hata_KarcıNN
LR=0.4 Alfa=0.4	0	0	0.145	0.001	0.02565125	0.24986965
	0	1	0.851	1.0		
	1	0	0.834	0.999		
	1	1	0.178	1.0		
LR=0.6 Alfa=0.6	0	0	0.12	0.0	0.01781402	0.25000052
	0	1	0.875	0.0		
	1	0	0.863	0.999		
	1	1	0.15	0.001		
LR=0.8 Alfa=0.8	0	0	0.108	0.006	0.01434217	0.06267135
	0	1	0.887	0.993		
	1	0	0.878	0.997		
	1	1	0.135	0.501		
LR=1.2 Alfa=1.2	0	0	0.093	0.004	0.01078577	0.00037029
	0	1	0.902	0.98		

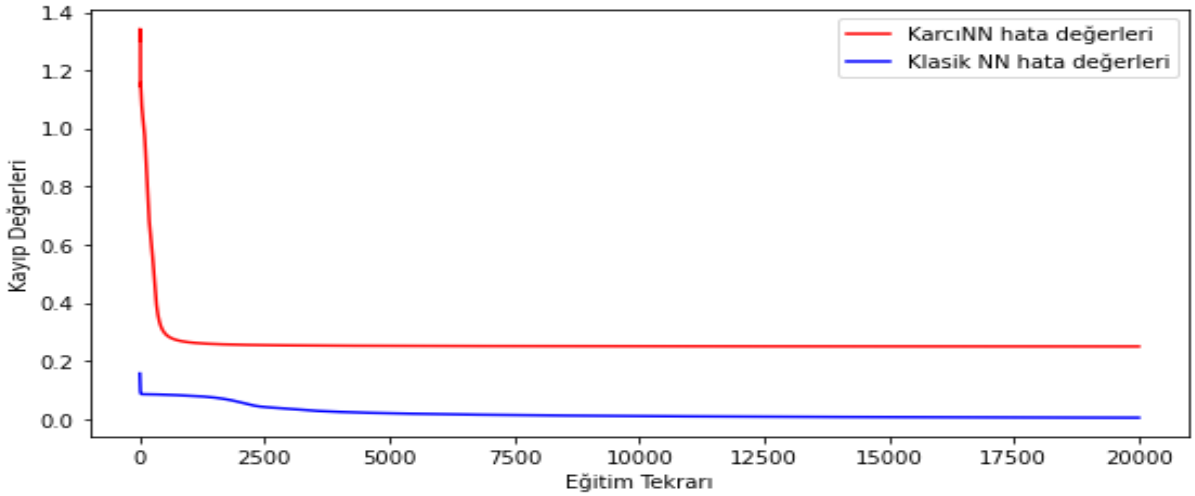
	1	0	0.894	0.981		
	1	1	0.117	0.026		
LR=1.4 Alfa=1.4	0	0	0.088	0.004	0.00973141	0.00019628
	0	1	0.907	0.974		
	1	0	0.9	0.993		
	1	1	0.112	0.004		
LR=1.6 Alfa=1.6	0	0	0.085	0.053	0.00896149	0.00174422
	0	1	0.91	0.952		
	1	0	0.905	0.959		
	1	1	0.107	0.012		
LR=1.8 Alfa=1.8	0	0	0.082	0.114	0.00847416	0.00988874
	0	1	0.912	0.916		
	1	0	0.908	0.889		
	1	1	0.104	0.084		
LR=2.2 Alfa=2.2	0	0	0.085	0.522	0.00875406	0.2441756
	0	1	0.9	0.474		
	1	0	0.919	0.404		
	1	1	0.106	0.268		



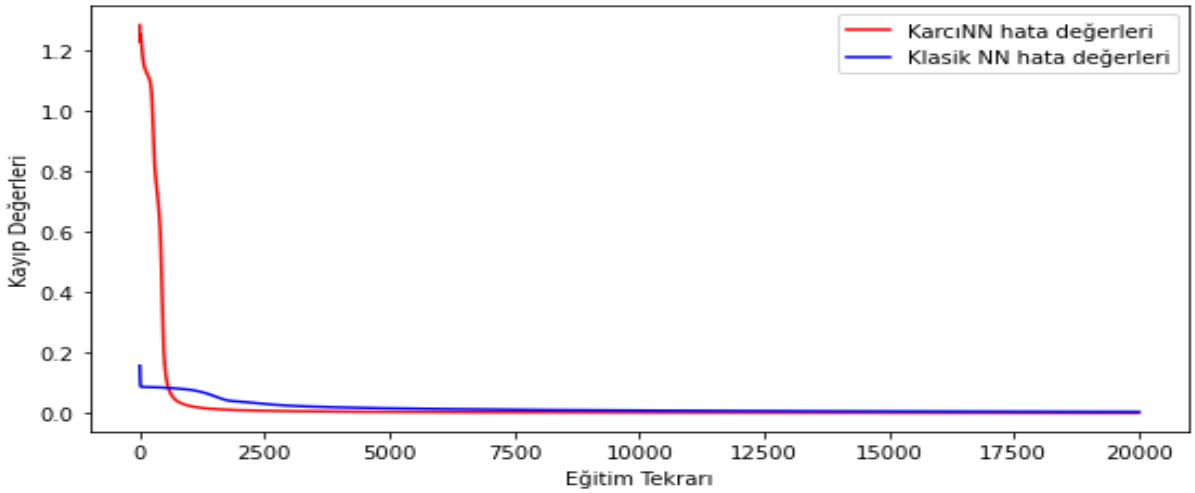
Şekil 5. Alfa ve Öğrenme Katsayısı 0.4 için Hata Grafiği



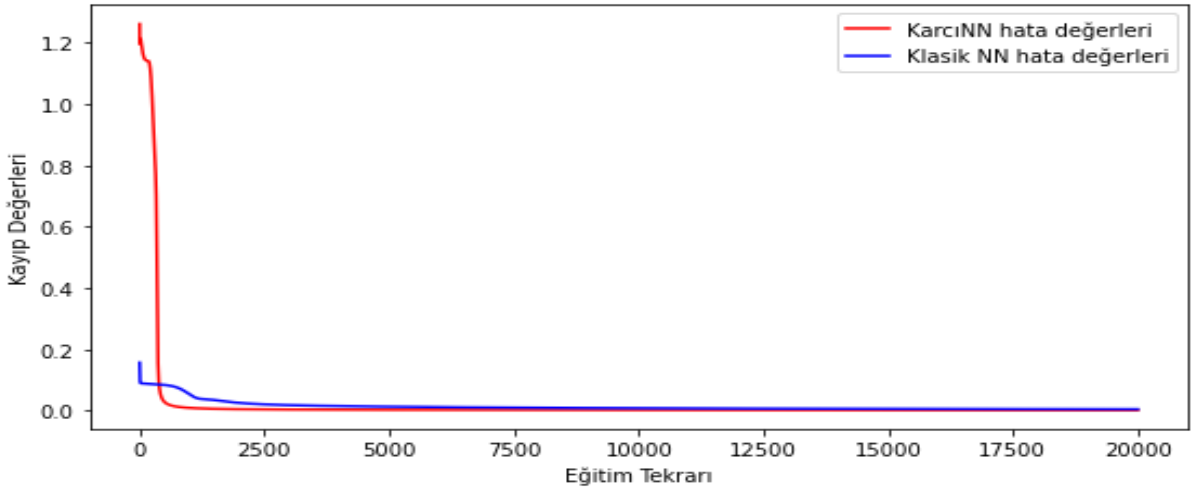
Şekil 6. Alfa ve Öğrenme Katsayısı 0.6 için Hata Grafiği



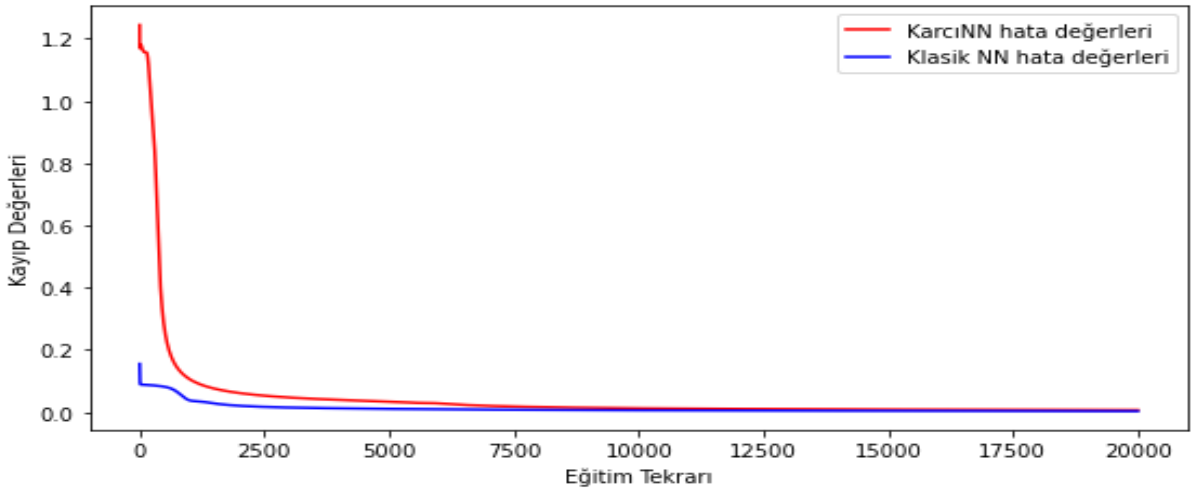
Şekil 7. Alfa ve Öğrenme Katsayısı 0.8 için Hata Grafiği



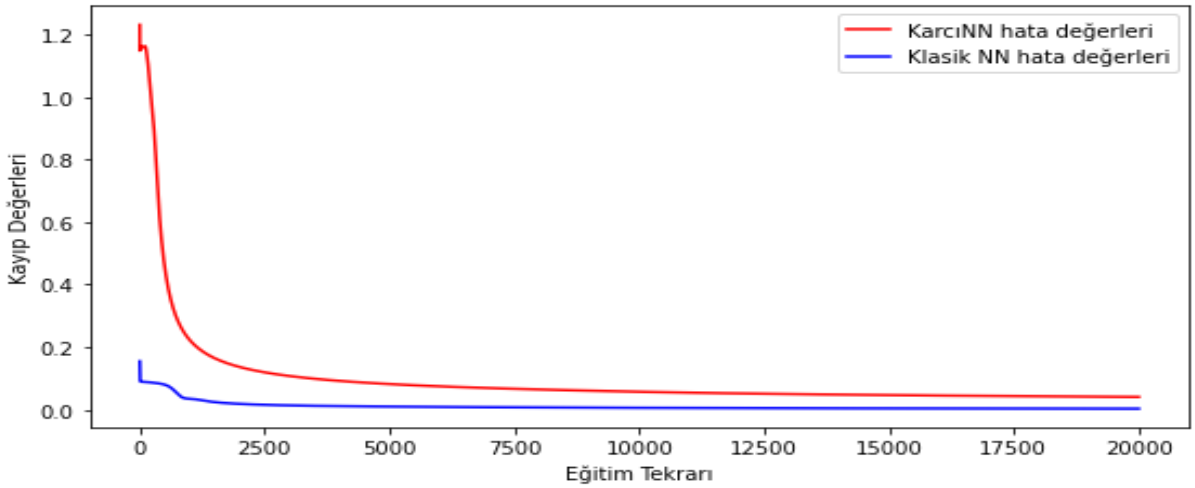
Şekil 8. Alfa ve Öğrenme Katsayısı 1.2 için Hata Grafiği



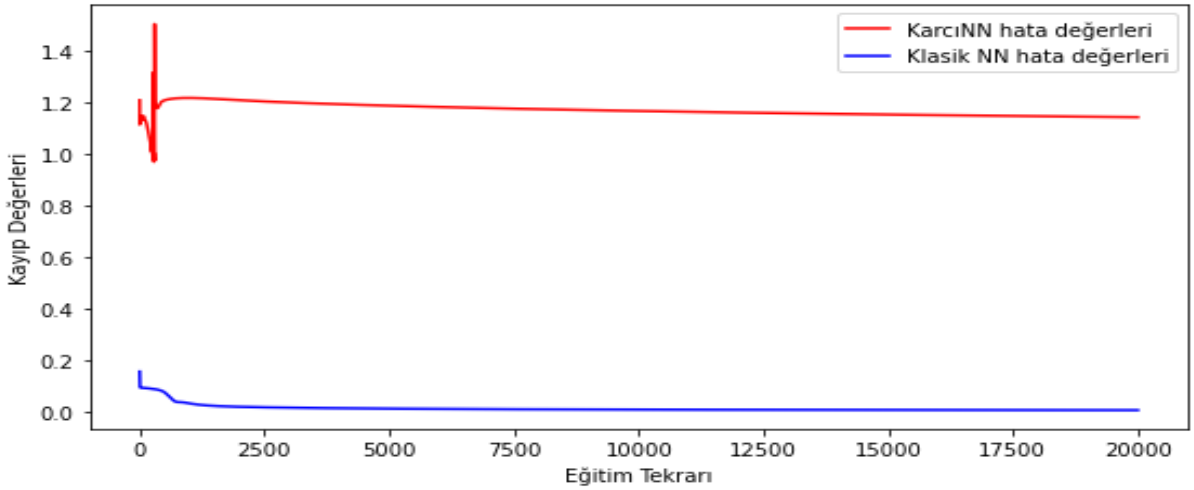
Şekil 9. Alfa ve Öğrenme Katsayısı 1.4 için Hata Grafiği



Şekil 10. Alfa ve Öğrenme Katsayısı 1.6 için Hata Grafiği



Şekil 11. Alfa ve Öğrenme katsayısı 1.8 için Hata Grafiği



Şekil 12. Alfa ve Öğrenme Katsayısı 2.2 için Hata Grafiği

5. Sonuç

Yapılan çalışmada öğrenme katsayısı kullanılmadan geri beslemeli yapay sinir ağlarına bir çözüm olarak kesir dereceli türevin kullanılması öne sürülmüştür. Önerilen yöntemin doğruluğunu ispatlamak için XOR problemi ele alınarak klasik yapay sinir ağları yöntemi ve Karcı NN ile karşılaştırılması yapılmıştır. Sistemleri eşit şartlarda başlatmak için her iki ağın ağırlık değerlerini aynı nümerik değerler alınmıştır ve klasik yapay sinir ağındaki öğrenme katsayısı ile Karcı NN'deki türevin kesir derecesi de aynı değerler alınarak böylece her iki sistem eşit şartlarda çalıştırılmıştır. Genel olarak ilgili tablo ve grafikler incelendiğinde klasik yöntem olan yapay sinir ağlarından Karcı NN'in 0.8 ile 1.8 aralığındaki türev derecesinde hem daha düşük hata değeri hem de ağın kısa sürede eğitilmesi bakımından daha iyi performans değerleri elde edildiği gözlemlenmiştir. Özellikle alfa parametresinin 1.4 değeri için hata oranının % 0.019 olarak ölçüldüğü başarılı sonuçlar elde edilmiştir.

Klasik yapay sinir ağlarında kullanılan öğrenme katsayısının küçük bir değer verilmesi öğrenme sürecini yavaşlatmakta, büyük değer verilmesi ise yerel çözümlere takılmasına neden olabilmektedir. Burada ağa dışarıdan müdahale edilmesini gerektirmektedir. Önerdiğimiz yöntem ise ağa dışarıdan müdahaleyi değil ağın kendi içinde var olan değeri kullanarak hem öğrenme sürecini hem de öğrenme hızını arttırdığı performans değerlendirmelerinin yapılan deneysel çalışmalardan anlaşılmaktadır.

Ayrıca yapılan deneysel çalışmalardan elde edilen başarıma göre Karcı NN'in yapay sinir ağı çalışması yapan araştırmacılar tarafından klasik yapay sinir ağı yerine kullanmayı tercih edecekleri bir yöntem olacağına inanıyoruz.

Kaynaklar

Anderson, D., & McNeill, G. (1992). Artificial neural networks technology. Kaman Sciences Corporation, 258(6), 1-83.

Anonim, 2022. https://www.elektrikport.com/teknik-kutuphane/aktivasyon-fonksiyonu-nedir/23511#ad-image-0, Erişim 10.08.2022.

Anonim, 2022a. https://kadirguzel.medium.com/geri-yay%C4%B1l%C4%B1ml%C4%B1-%C3%A7ok-katmanl%C4%B1-yapay-sinir-a%C4%9Flar%C4%B1-2-6a47b4f3a6c, Erişim 10.08.2022.

Ataseven, B. (2007). Satış öngörü modellemesi tekniği olarak yapay sinir ağlarının kullanımı: ”Petkim’de Uygulanması”, Yüksek Lisans Tezi, Celal Bayar Üniversitesi, Manisa.

Ergür, H.S. (2007). Aşındırıcı Su jetinin Teorik Analizi ve Yapay Sinir Ağı Yöntemiyle Modellenmesi. Doktora Tezi, Eskişehir Osmangazi Üniversitesi, Eskişehir.

Hardesty, Larry (14 April 2017). "Explained: Neural networks". MIT News Office. Retrieved 2 June 2022. https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414. Erişim: 08.08.2022.

Karakurt, M. & İşeri, İ. (2022). Patoloji Görüntülerinin Derin Öğrenme Yöntemleri İle Sınıflandırılması. Avrupa Bilim ve Teknoloji Dergisi, (33), 192-206.

KARCI, A. (2015). KESİR DERECELİ TÜREVİN YENİ YAKLAŞIMININ ÖZELLİKLERİ. *Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, *30*(3).

Karcı,A., “A New Approach for Fractional Order Derivative and Its Applications”, Universal Journal of Engineering Sciences, Vol:1, pp: 110-117, 2013.

Karcı, A., “Properties of Fractional Order Derivatives for Groups of Relations/Functions”, Universal Journal of Engineering Sciences, vol:3, pp:39-45, 2015a.

Karcı,A., “The Properties of New Approach of Fractional Order Derivative”, Journal of the Faculty of Engineering and Architecture of Gazi University, Vol.30, pp:487-501, 2015b.

Karci, A.,”Chain rule for fractional order derivative”, Science Innovation, Vol:3, pp:63-67, 2015c.

Karcı, A.,” Properties of Karcı’s Fractional Order Derivative”, Universal Journal of Engineering Science, Vol:7, pp:32-38, 2019.

McCulloch, W. S. and Pitts, W. 1943. A Logical Calculus of the İdeas İmmanent in Nervous Activity. The Bulletin of Mathematical Biophysics, 5:4, 115-133.

Mijwel, M. M. (2018). Artificial neural networks advantages and disadvantages. https://www.linkedin.com/pulse/artificial-neural-networks-advantages-disadvantages-maad-m-mijwel/. Erişim 09.08.2022.

Öztemel, E. (2006).Yapay Sinir Ağları. 2. Baskı. İstanbul: Papatya Yayıncılık.

Pişkin, M. https://mesutpiskin.com/blog/yapay-sinir-agi-derin-ogrenme.html, Erişim Tarihi: 05.08.2022 .

Rosenblatt, F. 1958. The Perceptron: A Probabilistic Model for Information Storage And Organization in the Brain. Psychological review, 65:6, 386.

Sağıroğlu, Ş.; Beşdok E. & Erler, M. (2003). Mühendislikte Yapay Zeka Uygulamaları I: Yapay Sinir Ağları. Kayseri: Ufuk Kitap Kıratsiye-Yayıncılık.

Şen, Z. (2004). *Yapay sinir ağları*. Su Vakfı.

Yang, Z.R.; Yang, Z. (2014). Comprehensive Biomedical Physics. Karolinska Institute, Stockholm, Sweden: Elsevier. p. 1. ISBN 978-0-444-53633-4.5054955669.