Fırat Üniversitesi Müh. Bil. Dergisi
37(2), 897-907, 2025

Araştırma Makalesi
https://doi.org/10.35234/fumbd.1661969

KarcıFANN Yönteminin Yakınsama Kabiliyetinin Analiz Edilmesi

Hülya SAYGILI[1*], Meral KARAKURT[2], Ali KARCI[3]
[1,3] Bilgisayar Mühendisliği Bölümü, İnönü Üniversitesi, Malatya, Türkiye
[2] Bilgisayar Teknolojileri Bölümü, Osmaniye Korkut Ata Üniversitesi Düziçi Meslek Yüksekokulu, Osmaniye, Türkiye
[*1]hulya-saygili@hotmail.com, [2]meralkarakurt@osmaniye.edu.tr,[3]ali.karci@inonu.edu.tr

(Geliş/Received: 20/03/2025; Kabul/Accepted: 29/09/2025)

Öz: YSA'ların optimizasyonunda yaygın kullanıma sahip tekniklerden gradyan iniş tabanlı geriye yayılım algoritmalarında, ağın çıktısı ile beklenen çıktı arasındaki hata hesaplanmakta ve bu hata geriye doğru yayılarak ağırlıklar güncellenmektedir. Ağırlıkların güncellenmesi işlemi, modelin öğrenme süresini ve performansını önemli ölçüde etkilemektedir. Gradyan iniş tabanlı algoritmalarda kullanılan öğrenme katsayısının çok büyük ya da küçük seçilmesi ezberleme, öğrenememe ve ağırlıkların yakınsamaması gibi problemlere neden olmaktadır. YSA'ların eğitimi aşamasında ağırlıkların, iterasyon sayısı arttıkça salınımlar yapmadan kararlı bir eğri çizmesi, başarılı bir öğrenme gerçekleştiğini ya da yerel minimum bir noktaya takıldığını göstermektedir. Bu ayrımı yapabilmek için ağırlık değişim grafikleriyle beraber ağın ürettiği hata değerlerine bakılmalıdır. Bu makalede, kesir dereceli türevle modelleri daha iyi ifade etmek ve YSA'larda karşılaşılan problemleri çözmek amacıyla kullanılan KarcıFANN yönteminin yakınsama, ezberleme durumu ve öğrenme performansına etkisi incelenmiştir. Sabit bir öğrenme katsayısı yerine kesir dereceli türevin kullanıldığı KarcıFANN yöntemi ile ADAM, Momentumlu GD ve SGD yöntemleri karşılaştırılmıştır. Türevlerin öğrenmeye olan etkilerini incelemek amacıyla XOR probleminin çözümü deneysel çalışmalarda ele alınmış ve yöntemlerin ağırlık değişimleri gözlemlenmiştir. MSE değerleri ve ağırlık değişim grafikleri incelendiğinde en başarılı yöntemin Momentumlu GD, ikinci başarılı yöntemin KarcıFANN yöntemi olduğu ve ADAM yönteminin de yerel bir minimum noktaya takıldığı görülmektedir.

Anahtar kelimeler: KarcıFANN, yapay sinir ağları, ADAM, SGD, momentumlu GD.

Analyzing the Convergence Ability of the KarcıFANN Method

Abstract: In gradient descent–based backpropagation algorithms, which are among the most widely used techniques for the optimization of ANNs, the error between the network's output and the expected output is calculated, and this error is propagated backward to update the weights. The process of updating the weights significantly affects both the learning time and the performance of the model. In gradient descent–based algorithms, choosing a learning rate that is too large or too small may lead to problems such as overfitting, failure to learn, or the weights not converging. During the training phase of ANNs, if the weights follow a stable curve without oscillations as the number of iterations increases, this indicates either successful learning or that the model is stuck at a local minimum. To distinguish between these cases, both the weight change graphs and the error values produced by the network should be examined.In this study, the effects of the KarcıFANN method—which uses fractional derivatives to better represent models and to address the problems encountered in ANNs—on convergence, overfitting, and learning performance were investigated. The KarcıFANN method, in which a fractional derivative is used instead of a fixed learning rate, was compared with ADAM, Momentum-based GD, and SGD methods. To examine the effects of derivatives on learning, the XOR problem was addressed in the experimental studies, and the weight changes of the methods were observed. When the MSE values and weight change graphs were analyzed, it was observed that the most successful method was Momentum-based GD, the second most successful was the KarcıFANN method, and the ADAM method got stuck at a local minimum.

Key words: KarcıFANN, artificial neural networks, ADAM, SGD, momentum-based GD.

1. Giriş

Yapay Sinir Ağları (YSA), insan beynindeki biyolojik sinir hücrelerinin (nöronların) çalışma prensiplerinden esinlenerek tasarlanmış, bilgi işleme ve öğrenme yeteneğine sahip matematiksel modellerdir. Bir YSA, veriyi işlemek için kullanılan nöronlar ve bu nöronlar arasındaki bağlantılar (ağırlıklar) gibi temel bileşenlerden oluşur

* Sorumlu yazar: hulya-saygili@hotmail.com. Yazarların ORCID Numarası: [1] 0000-0003-1926-9918, [2] 0000-0001-7318-2798, [3] 0000-0002-8489-8617

[1]. Görüntü işleme, doğal dil işleme, sağlık, finans, otonom sistemler gibi birçok karmaşık problemin çözümünde yaygın olarak kullanılmaktadır.

YSA'larda ağırlıklar, giriş verisinin çıkış verisi üzerine etkisini belirleyen temel unsurlardır. Ağırlık değişimi, modelin performansını, genelleme kabiliyetini ve karar verme süreçlerini doğrudan etkilemektedir. Bu nedenle ağırlıkların optimize edilmesi için optimizasyon yöntemleri kullanılarak geri yayılımlı YSA'lar eğitilmektedir. Geriye yayılımlı YSA' ların çalışması, ileri besleme ve geriye yayılım olarak iki aşamalıdır. İleri besleme aşamasında, ağa giriş verileri sunularak her veri ile o veriye ait ağırlıklar çarpılarak bu çarpımlar toplanmakta ve ağın bir çıktı değeri ile hata değeri üretmesi sağlanmaktadır. Elde edilen hata değeri, çıkış katmanından geriye doğru yayılarak ağırlıklar güncellenmektedir. Bu süreç, belli bir eğitim iterasyonu boyunca ya da hata değeri, istenen minimum değere indirilene kadar tekrarlanmaktadır [2]. Ağırlık güncelleme işlemi, modelin öğrenme sürecini ve yakınsama durumunu doğrudan etkilemektedir. Bir modelin yakınsaması, ağın eğitim verileri üzerinde giderek daha iyi performans göstermesi ve eğitim sonunda optimum çözüme ulaştığı anlamına gelmektedir. Ağırlık değişimleri, bu süreci hızlandırabilmekte, yavaşlatabilmekte ya da engelleyebilmektedir. YSA'ların eğitiminin başarılı olabilmesi için ağırlıkların güncellenmesinde kullanılan parametrelerden biri optimizasyon algoritmalarıdır.

Gradyan tabanlı optimizasyon algoritmalarının kullanıldığı geriye yayılımlı YSA'larda ağırlıkların ne kadar güncelleneceğini belirleyen önemli bir parametre de öğrenme katsayısıdır. Bu katsayısının seçiminde bazı belirsizlikler vardır. Öğrenme katsayısının çok küçük seçilmesi, öğrenme sürecini yavaşlatmakta ve dolayısıyla eğitimin süresini uzatmaktadır [3]. Bu durum, modelin istenilen şekilde öğrenememesine, yakınsayamamasına ve optimizasyonun tamamlanamamasına neden olabilmektedir. Öğrenme katsayısı büyük seçildiğinde de ağırlık güncellemeleri büyük adımlarla gerçekleştiğinden model kararsız hale gelmekte ve öğrenme süreci başarısız olabilmektedir. Bu durumlar, modelin öğrenememe ya da ezberlemesine neden olabilmektedir. Öğrenememe, modelin eğitim ve test aşamalarında çok büyük hata değerleri üreterek problemi çözememesidir. Ezberleme ise modelin eğitim aşamasında düşük hata değerleri üretirken test aşamasında çok büyük hata değerleri üretmesidir.

KarcıFANN yöntemi, öğrenme katsayısı yerine her adımdaki hata değerinin ilgili ağırlıklara göre hesaplanan kesir dereceli türevinden faydalanmakta ve ağırlıkların bu hatalara göre güncellenmesini sağlamaktadır. Böylece, sabit bir öğrenme katsayısı yerine ağın ürettiği değerlerin kullanılması ile ağa dışarıdan yapılan müdahaleyi en aza indirmektedir. KarcıFANN yöntemi, ağın daha erken süreçte yakınsamasını sağlayabilmekte ve ezberleme, öğrenememe, kaybolan ya da patlayan gradyan problemleri gibi YSA'da çok karşılaşılan sorunları çözebilmektedir.

Bu çalışmanın amacı, KarcıFANN yönteminin yakınsama kabiliyetini ve ağırlık değişimleri üzerindeki etkilerini detaylı bir şekilde incelemekle beraber SGD, Momentumlu GD ve Adam yöntemleri ile karşılaştırmalı biçimde değerlendirmektir. XOR problemi üzerinde yürütülen deneylerde, yöntemlerin performansları MSE değerleri ve ağırlık değişim grafikleri üzerinden değerlendirilerek yakınsama, ezberleme ve öğrenme performansları ortaya konulmuştur.

Yapılan tüm deneyler Python 3.12.7 sürümü kullanılarak Anaconda dağıtımı üzerinde gerçekleştirilmiştir. Veri işleme ve sayısal hesaplamalarda NumPy (v1.26.4), SciPy (v1.13.1), scikit-learn (v1.5.1) ve pandas (v2.2.2) kütüphanelerinden yararlanılmıştır. Sonuçların görselleştirilmesinde Matplotlib (v3.9.2) kullanılmıştır.

KarcıFANN algoritması ile tasarlanan modelde alfa parametresinin 0.1 ile 3.5 aralığı dışındaki değerler için başarılı bir öğrenme gerçekleşmemiştir. Bu nedenle alfa parametresi 0.1-3.5 aralığında sınırlandırılmıştır.

Çalışmanın devamı 4 bölümden oluşmaktadır. Bölüm 2'de ilgili literatür taraması yapılmıştır. Bölüm 3'te optimizasyon algoritmaları açıklanmıştır. Bölüm 4'te XOR probleminin çözümü ve elde edilen bulgular açıklanmıştır. Son bölümde ise sonuçlar tartışılmıştır.

2. İlişkili Çalışmalar

YSA'ların eğitim sürecinde ağırlıkların kısa sürede yakınsaması ve kararlı davranması beklenir. Ağırlıkların eğitim boyunca salınımlar yapması öğrenmenin başarısız olduğu anlamına gelmektedir. Bu bölümde, ağırlıkların değişimleri ve optimizasyonları amacıyla yapılan bazı çalışmalar sunulmaktadır.

Brutzkus ve Globerson, büyük ölçekli modellerin küçük modellere göre daha iyi genelleme yapmasının nedenlerini açıklamak amacıyla bir çalışma sunmuşlardır. Çalışmada klasik XOR problemi genişletilmiş ve max-pooling içeren üç katmanlı bir evrişimsel sinir ağı (CNN) üzerinden analiz gerçekleştirilmiştir. Ağırlık kümelenmesi ile özellik uzayının keşfi arasındaki etkileşimin modelin genelleme performansı üzerindeki etkisi incelenmiştir [4].

Pinto ve Tavares, yaptıkları çalışmada Parametric ReLU (PReLU) aktivasyon fonksiyonu kullanarak klasik olarak çok katmanlı yapay sinir ağları ile çözülebilen XOR probleminin tek katmanlı bir sinir ağı ile

çözülebileceğini ortaya koymuşlardır. Yapılan deneysel çalışmalarda PReLU, çok katmanlı algılayıcı (MLP) ve Büyüyen Kosinüs Birimi (GCU) aktivasyon fonksiyonları ile karşılaştırılmış ve yüksek başarı sağladıklarını sunmuşlardır [5]. Yang vd., yazılım tabanlı geri yayılım yöntemi ve donanım tabanlı RWC (Random Weight Change – Rastgele Ağırlık değişimi) yöntemini içeren yeni ve hibrit bir öğrenme yöntemi önererek çevrim tabanlı sinir ağlarının geliştirilmesini amaçlamışlardır. Geri yayılımla öğrenilen ağırlık değerlerini sinir ağının donanımına aktarırken yazılım ve donanım arasındaki farktan dolayı önemli miktarda çıktı hatası almışlardır. Önerdikleri yöntemde, basit bir donanımda uyguladıkları RWC algoritmasının tamamlayıcı bir öğrenmesiyle bu hatayı azaltmışlar ve yöntemlerinin kullanışlılığını göstermişlerdir [6]. Starodub vd., YSA'ların öğrenmesini optimize etmek amacıyla gradyanların ağırlık değerlerini değiştirdiği bir yöntem önermiştir. İlgili çalışma, öğrenmenin başarısız olduğu kısımlarda gradyan ağırlıkları değiştirilerek ağ ağırlıklarının eğitim sırasında ne kadar aktif olarak değiştiğini açıklamayı ve dolayısıyla eğitim sürecini anlamayı sağlamaktadır [7]. Li vd, orijinal Hopfield sinir ağları modelini, ağın ağırlıklarını zamanla değişecek şekilde uyarlayarak ağırlık fonksiyonu matrisli Ayrık Hopfield sinir ağları (Discrete Hopfield neural networks with weight function matrix - DHNNWFM) adını verdikleri yöntemi önermişlerdir. Kullandıkları ağırlık fonksiyon matrisini, negatif olmayan ve monoton artan şekilde oluşturmuşlardır. Yaptıkları deneysel çalışmalar sonunda yöntemlerinin istedikleri gibi yakınsadığını göstermişlerdir [8]. Nagel vd., YSA'ların yüksek doğrulukta ve hızlı öğrenmesini sağlamak amacıyla ağırlık ve aktivasyon fonksiyonlarının değerlerinin sıkıştırılarak işlendiği nicelemeye duyarlı eğitim (quantization-aware training - QAT) yöntemini kullanmışlardır. İlgili çalışmada, eğitim aşamasında nicelenmiş ağırlık değerlerinin salınımlarını önlemek için iki yeni QAT algoritmasını önermişlerdir: salınım sönümleme ve yinelemeli ağırlık dondurma. MobileNetV2, MobileNetV3 ve EfficentNet-lite ağ modelleri üzerinde yaptıkları çalışmalarda başarılı sonuçlara ulaşmışlardır [9]. Benzer amaçlarla Nagel vd. ve Banner vd. QAT yöntemini kullanarak çalışmalar yapmışlardır [10-11]. Agrawal ve Tendle derin sinir ağlarının katman bazında ağırlık değişimlerini inceledikleri çalışmada, eğitim süreci boyunca her katmandaki ağırlıkların değişim oranları analiz edilmiş ve derin katmanlarda daha belirgin ağırlık güncellemelerinin gerçekleştiği ortaya koymuşlardır. Elde ettikleri bulgular ile derin sinir ağlarının öğrenme dinamiklerinin katmanlar arası farklılık gösterdiğini ve belirli katmanların öğrenmeye daha fazla katkı sağladığını göstermişlerdir [12]. Wei vd., kesirli diferansiyel denklemlere dayalı çok katmanlı bir yapay sinir ağı modeli önermişlerdir. Çalışmada Caputo türevinden yararlanılmış ve parametre güncellemeleri Adam algoritması ile gerçekleştirilmiştir. Gerçek yaşam verileriyle yapılan tahmin çalışmalarında, önerilen yöntemin klasik lojistik modellere kıyasla daha yüksek doğruluk ve daha fazla parametre serbestliği sağladığı gösterilmiştir [13]. Ancak bu çalışmada optimizasyon süreci, öğrenme katsayısına bağımlı olarak gerçekleşmektedir.

3. Optimizasyon Algoritmaları

Optimizasyon, bir problemi çözen en uygun çözümü bulma sürecidir. Gerçek hayattaki çoğu karmaşık problemi çözebilmek amacıyla çeşitli optimizasyon algoritmaları kullanılmaktadır. Gradyan iniş tabanlı bazı optimizasyon algoritmaları bu bölümde açıklanmaktadır.

3.1. Gradyan İniş Algoritması (GD)

GD algoritması, Rumelhart (1986) tarafından önerilmiştir. GD algoritması, hata fonksiyonunun değerini en aza indirmek için fonksiyonun gradyanını (türevini) kullanarak iteratif bir yaklaşım uygulamaktadır [2]. Bu yöntem ile ağ eğitilirken giriş katmanına eğitim verilerinin tümü sunulmaktadır. YSA'larda ağırlık değerlerinin GD algoritması ile güncellenmesi işlemi Denklem 1 ile formüle edilmektedir.

$$w_{t+1} = w_t - \alpha \frac{\partial L}{\partial w_t} \tag{1}$$

α, öğrenme oranı; w_t, ağırlık parametresi; L, hata fonksiyonunu; $\partial L/\partial w_t$, her bir örneğin gradyanıdır.

GD algoritması, büyük veri setlerinde etkili bir şekilde çalışabilmekte fakat bazı durumlarda yerel minimum (ya da maksimum) noktalara takılma, gradyanların kaybolması ve gradyan patlaması gibi problemlere neden olmaktadır. Bu problemleri çözebilmek amacıyla stokastik gradyan iniş, mini-batch gradyan iniş gibi GD tabanlı algoritmalar geliştirilmiştir.

3.2. Stokastik Gradyan İniş Algoritması (SGD)

SGD algoritması, hata fonksiyonunu en aza indirmek amacıyla modelin ağırlıklarını iteratif olarak güncellemektedir. Geleneksel GD algoritmasından farklı olarak, tüm eğitim verileri için gradyan hesaplamak yerine küçük veri grupları ya da her bir veri örneği için gradyan hesaplanarak parametre güncellemesi yapılmaktadır. Bu özelliği, SGD'yi büyük veri setleri için hesaplama açısından daha verimli kılmaktadır. Ancak, SGD algoritmasında gradyan hesaplamalarının her bir eğitim örneği için yapılması, salınımlara neden olabilmektedir. Bu salınımlar, öğrenme sürecinde kararsızlığa yol açarak algoritmanın global minimuma ulaşmasını zorlaştırabilir [14]. SGD algoritmasında ağırlık parametrelerinin güncelleme işlemi, GD algoritmasındaki gibi Denklem 1 ile gösterilen güncelleme denklemiyle hesaplanmaktadır.

3.3. ADAM

Uyarlanabilir Moment Tahmini (Adaptive Moment Estimation, ADAM) makine öğrenme ve derin öğrenme alanında kullanılan optimizasyon yöntemidir. Büyük veri setleri ve yüksek parametre sayısına sahip modeller de etkili olan bu yöntem, Momentum ve RMSProp yöntemlerinin avantajları birleştirdiği için daha hızlı ve kararlı bir yakınsama sağlamaktadır [15]. ADAM yönteminde moment değerleri Denklem 2 ve Denklem 3 ile gösterildiği gibi hesaplanmaktadır. Ağırlıkların güncellenmesi için de Denklem 4 kullanılmaktadır.

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \, , v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{2}$$

$$\widehat{m_t} = \frac{m_t}{1-\beta_1^t} \, , \widehat{v_t} = \frac{v_t}{1-\beta_2^t} \tag{3}$$

$$w_t = w_{t-1} - \frac{\alpha}{\sqrt{\widehat{v_t}+\epsilon}} \widehat{m_t} \tag{4}$$

Yukarıdaki denklemlerde, m_t ilk moment, v_t ikinci momenti, g_t gradyan, $\beta_1$ve β_2 moment tahminleri için çarpanlar, α öğrenme oranı, Є sayısal kararlılığı artırmak için küçük bir değeri ifade etmektedir.

3.4. Momentumlu GD

Momentumlu gradyan iniş algoritması, gradyan iniş algoritmasına momentum terimi eklenerek oluşturulan bir yaklaşımdır. Bu terim, gradyan iniş algoritmasına göre daha hızlı, daha doğru ve daha kararlı öğrenmesini sağlamaktadır [16]. v_t hız (velocity) vektörü, β momentum katsayısı, η öğrenme oranı, $\nabla f(\theta w_t)$ kayıp fonksiyonunun gradyanı ve w_t ağırlık değeri olmak üzere v_t değeri, Denklem 5 ile ve ağırlık güncelleme işlemi Denklem 6 ile gösterildiği gibi hesaplanmaktadır.

$$v_t = \beta v_{t-1} + \nabla f(w_t) \tag{5}$$

$$w_{t+1} = w_t - \eta v_t \tag{6}$$

Momentum teriminin eklenmesi, salınımları önleyerek algoritmanın daha hızlı yakınsamasını ve yerel minimumlara takılmamasını sağlamaktadır. Ancak, her ağırlık değeri için v_t hız vektörünün hesaplanması gerektiğinden hafıza ve hesaplama maliyeti artabilmektedir [17].

3.5. Karcı kesir dereceli YSA (KarcıFANN)

Karcı kesir dereceli YSA yöntemi, SGD yönteminde kullanılan sabit bir öğrenme katsayısı yerine sistemin ürettiği hataların ağırlık değerlerine göre kesir dereceli türevinin kullanıldığı bir yaklaşımdır. SGD yönteminde kullanılan öğrenme katsayısının seçimine dair belirsizlikler, problemlerin modellenmesinde ve karmaşık sistemlerin analizinde başarısızlıklara neden olabilmektedir. Bu durumlar, YSA'larda öğrenememe, ezberleme, yerel minimum noktalara takılma ve çözüme yakınsayamama gibi birçok problemi beraberinde getirmektedir. Bu problemleri çözmek amacıyla Karcı kesir dereceli türev kullanımı daha doğru ve etkin bir modelleme süreci sağlamaktadır. Ayrıca, KarcıFANN yöntemi kullanılarak global modeller üretilebilmektedir.

KarcıFANN yönteminde kesir dereceli türev tanımından gelen çarpan, her adımdaki hata değerine göre dinamik olarak değişmektedir. Böylece, sistemden gelen verilerle işlem yapılarak sisteme dışarıdan yapılan müdahale azaltılmakta ve modelin öğrenme sürecini daha erken aşamalarda tamamlaması sağlanmaktadır [18-23]. $f(x)$, hata fonksiyonunu; $f'(x)$, hata fonksiyonunun Newton türevini *ve* α, Karcı kesir dereceli türevin derecesini göstermek üzere Karcı kesir dereceli türev Denklem 7 ve ağırlıkların güncelleme işlemi Denklem 8 ile gösterildiği gibi hesaplanmaktadır.

$$D_{\alpha}^{K} f(x) = \left(\frac{f(x)}{x}\right)^{\alpha-1} . f'(x) \tag{7}$$

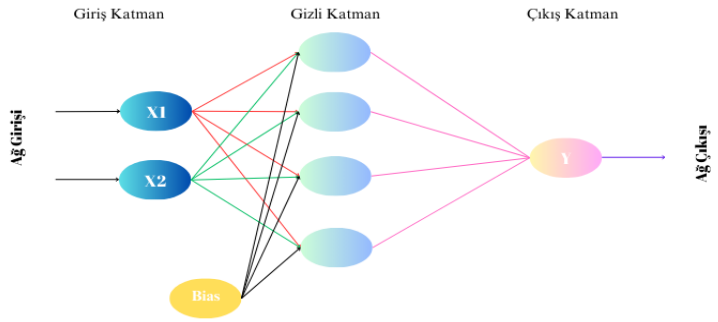
$$w_{t+1} = w_t - D_{\alpha}^{K} \frac{J}{W_t} \tag{8}$$

4. XOR Probleminin Çözümü

4.1.Doğrusal Olmayan Problem: XOR Problemi

XOR problemi, YSA uygulamalarında yaygın olarak kullanılan bir problemdir. XOR fonksiyonu, doğrusal değildir ve fonksiyona girilen iki giriş değeri aynı ise 0, farklı ise 1 değerini döndürmektedir. XOR problemi, doğrusal ayrımı mümkün olmayan bir problem olması nedeniyle tek katmanlı YSA'lar ile çözülemez. Bu problemin çözümünde geri yayılımlı çok katmanlı YSA'lar kullanılmaktadır.

Bu çalışma kapsamında SGD, Momentumlu GD, Adam ve KarcıFANN yöntemlerini kullanarak XOR probleminin çözümü için oluşturulan YSA modeli Şekil 1 ile gösterilmektedir. YSA modeli, giriş katmanında 2 nöron, ara (gizli) katmanda 4 nöron ve çıktı katmanında 1 nöron olmak üzere 3 katmanlı olarak tasarlanmıştır.



Şekil 1. XOR problemi çözümü için kullanılan ağ modeli.

Modelde ağırlık değerleri başlangıçta rastgele atanmış ve tüm optimizasyon yöntemleri için aynı olarak belirlenmiştir. Ağların eğitimi sırasında, eğitim tur sayısı (iterasyon, epoch) sayısı 10.000 olarak ayarlanmıştır. Modelin hata fonksiyonu olarak Ortalama Karesel Hata (Mean Squared Error – MSE) kullanılmıştır. Yi , beklenen değer; Ygi, modelin ürettiği değer ve n, örnek sayısını belirtmek üzere MSE fonksiyonu, Denklem 9 ile gösterildiği gibi hesaplanmaktadır.

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (Y_i - Yg_i)^2 \tag{9}$$

4.2. Uygulama Sonuçları

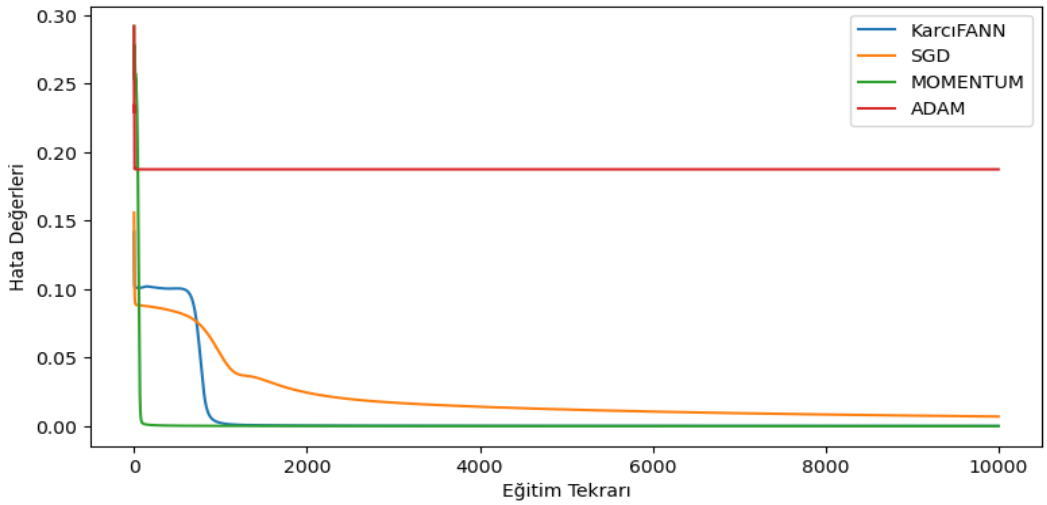
0,1 ile 3,5 aralığında belirlenen alfa ve öğrenme katsayısı değerleri kullanılarak dört optimizasyon yöntemiyle üretilen MSE değerleri Tablo 1 ile gösterilmektedir. Yapılan deneysel çalışmaların sonucunda, Momentumlu GD algoritmasının 0,9 ve 1,0 dışındaki tüm alfa ve öğrenme katsayısı değerleri için en başarılı sonuçları ürettiği ve ADAM algoritmasının da 0,1 ve 0,2 dışındaki tüm alfa ve öğrenme katsayısı değerleri için en yüksek MSE değerleri üreterek başarısız olduğu görülmektedir. Alfa ve öğrenme katsayısının 0,1 ve 0,2 değerleri için KarcıFANN, SGD ve ADAM yöntemleriyle eğitim aşamasında üretilen MSE değerleri test aşamasına göre daha

düşük ve ADAM yönteminin her iki veri setindeki MSE değerleri KarcıFANN ve SGD yöntemlerine göre daha düşük olarak ölçülmüştür. Alfa ve öğrenme katsayısının 0,3 değerinde KarcıFANN ve SGD yöntemlerinde bir önceki değerlere benzer MSE değerleri üretilirken ADAM yönteminde eğitim ve test aşamalarında çok yakın MSE değerleri üretilmiştir. Alfa ve öğrenme katsayısının 0,5-0,8 ve 2,1-3,5 aralığındaki değerler için SGD yöntemi KarcıFANN ve ADAM yöntemlerine göre daha düşük olduğu görülmektedir. ADAM yöntemi, 0,5-3,5 aralığındaki her alfa ve öğrenme katsayısı değeri için eğitim ve test aşamalarında hemen hemen aynı MSE değerlerini üretmektedir. Alfa ve öğrenme katsayısının 0,9 ve 1,0 değerleri için tüm veri setlerinde KarcıFANN yöntemi daha düşük MSE değerleri üretmektedir. KarcıFANN yöntemi, alfa katsayısının 1,0 değerinde SGD yöntemine dönüşmektedir. Alfa ve öğrenme katsayısının 1,1-2,0 aralığındaki değerleri için KarcıFANN yöntemi, Momentumlu GD'ye göre daha yüksek ancak diğer iki yönteme göre daha düşük MSE değerleri üretmektedir.

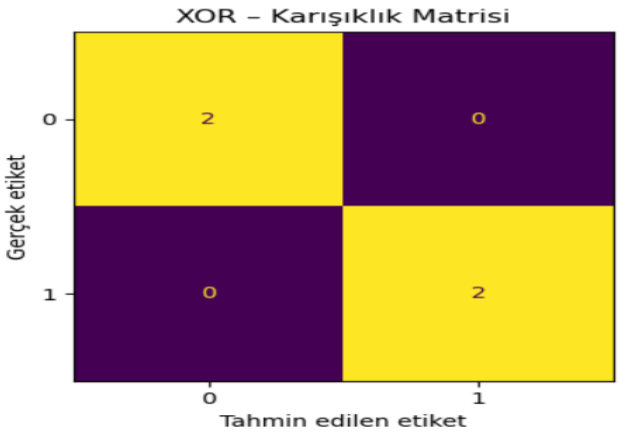
Tablo 1. XOR probleminin çözümünde farklı alfa ve öğrenme oranları için KarcıFANN, SGD, ADAM ve Momentumlu GD yöntemleri kullanılarak elde edilen MSE değerleri.

Alfa/ Öğrenme Oranı	KarcıFANN		SGD		ADAM		Momentumlu GD	
	Eğitim	Test	Eğitim	Test	Eğitim	Test	Eğitim	Test
0,1	0,07812500	0,500000	0,08253006	0,24845703	0,062652000	0,166669377	0,000217	0,00021726
0,2	0,07812500	0,49999998	0,08094572	0,24608712	0,062652000	0,12500013	0,000103	0,000102621
0,3	0,07812499	0,49999993	0,06762222	0,2129957	0,125027000	0,125000085	6,7e-05	6,6765e-05
0,4	0,07812470	0,49999813	0,02841464	0,07935195	0,125016000	0,062512554	4,9e-05	4,9365e-05
0,5	0,22438208	0,10426082	0,01887384	0,05040344	0,187504000	0,187500025	3,9e-05	3,9121e-05
0,6	0,06295320	0,06284979	0,01489248	0,03912254	0,187502000	0,187500018	3,2e-05	3,2386e-05
0,7	0,06282608	0,06283014	0,01261580	0,03284678	0,187502000	0,187500014	2,8e-05	2,7628e-05
0,8	0,06256070	0,06256367	0,01109465	0,02871668	0,187501000	0,187500011	2,4e-05	2,4094e-05
0,9	1,89923899e-06	8,06849866e-06	0,00998274	0,02572742	0,187501000	0,187500007	2,1e-05	2,1372e-05
1,0	4,95884825e-06	1,40047073e-05	0,00912247	0,02343168	0,187501000	0,187500005	1,9e-05	1,9215e-05
1,1	0,00016967	0,00031379	0,00843138	0,02159879	0,187501000	0,187500004	1,7468e-05	1,7e-05
1,2	0,00031259	0,00058045	0,00786209	0,02009758	0,187500000	0,187500003	1,6e-05	1,6029e-05
1,3	7,85829338e-05	0,00038912	0,00738564	0,01884862	0,187500000	0,187500003	1,5e-05	1,4828e-05
1,4	9,31395461e-05	0,00033893	0,00698383	0,01780243	0,187500000	0,187500002	1,4e-05	1,3816e-05
1,5	0,00012893	0,00050246	0,00664565	0,0169295	0,1875	0,187500002	1,3e-05	1,2956e-05
1,6	0,00203343	0,00433123	0,0063659	0,016217	0,1875	0,187500002	1,2e-05	1,222e-05
1,7	0,00073380	0,0020462	0,00614618	0,01567255	0,1875	0,187500001	1,2e-05	1,1591e-05
1,8	0,00145636	0,00398954	0,00600067	0,01534443	0,1875	0,187500001	1,1e-05	1,1054e-05
1,9	0,00304841	0,00690758	0,00598272	0,01541972	0,1875	0,187500001	1,1e-05	1,0594e-05
2,0	0,0003015	0,00258429	0,006415	0,01737656	0,1875	0,187500001	1e-05	1,0189e-05
2,1	0,03057281	0,09036979	0,00624748	0,01619678	0,1875	0,187500001	1e-05	9,774e-06
2,2	0,06158396	0,18229702	0,00559155	0,01450733	0,1875	0,187500001	9e-06	9,215e-06
2,3	0,05739571	0,17876757	0,00524042	0,01360238	0,1875	0,187500001	8e-06	8,372e-06
2,4	0,07754852	0,24460873	0,00498979	0,01295627	0,1875	0,1875	8e-06	7,6e-06
2,5	0,06711669	0,21677706	0,00479145	0,01244448	0,1875	0,1875	7e-06	7,264e-06
2,6	0,07536766	0,23818847	0,00462662	0,01201858	0,1875	0,1875	6e-06	5,778e-06
2,7	0,07536766	0,23818847	0,00448617	0,01165502	0,1875	0,1875	5e-06	5,358e-06
2,8	0,07405632	0,23346845	0,00436522	0,01134117	0,1875	0,1875	5e-06	5,233e-06
2,9	0,08426657	0,24887302	0,00426122	0,0110704	0,1875	0,1875	5e-06	5,343e-06
3	0,07340475	0,24726232	0,00417332	0,01084049	0,1875	0,1875	7e-06	6,872e-06
3,1	0,08165853	0,2497479	0,0041026	0,010654	0,1875	0,1875	7e-06	6,724e-06
3,2	0,08171047	0,24941668	0,0040533	0,01052175	0,1875	0,1875	7e-06	6,523e-06
3,3	0,08655396	0,24944721	0,00403802	0,01047583	0,1875	0,1875	7e-06	6,327e-06
3,4	0,08627174	0,24771767	0,00410504	0,01064029	0,1875	0,1875	6e-06	6,141e-06
3,5	0,08494966	0,25033277	0,00532371	0,01368371	0,1875	0,1875	6e-06	5,964e-06

Alfa ve öğrenme katsayısının Şekil 2 ile 1,4 değeri için KarcıFANN, Momentumlu GD, ADAM ve SGD yöntemlerine ait MSE grafiği, Şekil 3 ile KarcıFANN yöntemine ait XOR karışıklık matrisi gösterilmektedir. Momentumlu GD yönteminde, diğer yöntemlere göre daha hızlı bir öğrenme gerçekleşmiştir. ADAM yönteminde ise öğrenmenin başarısız olduğu görülmektedir. Yaklaşık 1000 iterasyondan sonra KarcıFANN yöntemi, SGD ve ADAM yöntemlerine göre çok daha başarılı performans göstermektedir ve ağırlıkların değişimleri yakınsamıştır.



Şekil 2. Alfa ve öğrenme katsayısının 1,4 değeri için KarcıFANN, momentumlu GD, ADAM ve SGD yöntemlerine ait MSE grafiği.

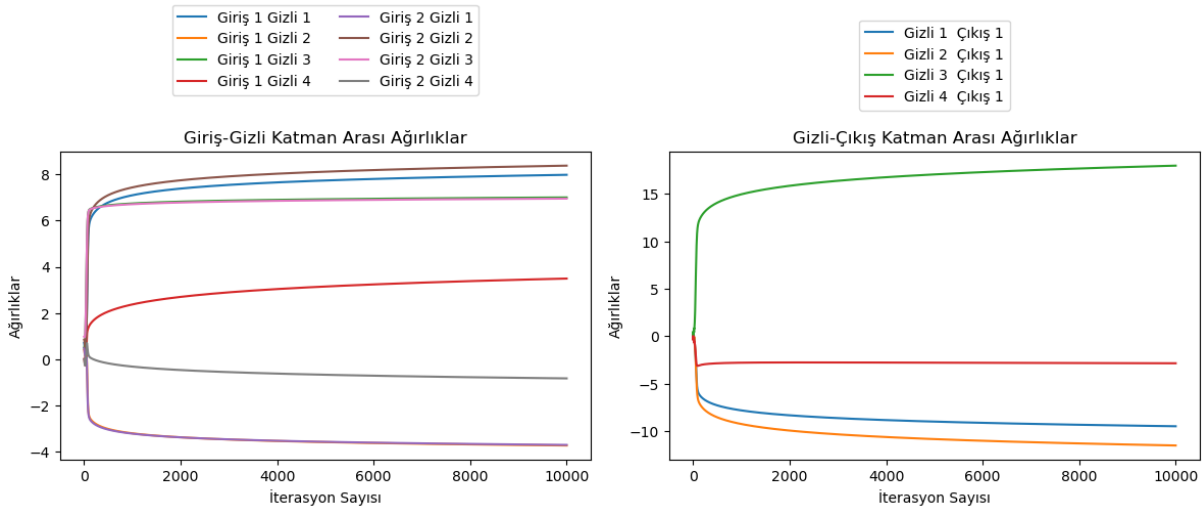


Şekil 3. Alfa 1,4 değeri için KarcıFANN yöntemine ait Karışıklık matrisi.

Bu çalışmada kullanılan optimizasyon yöntemlerine ait ağırlık değişimleri incelenirken ağın yakınsama durumu analiz edilmektedir. Ağırlıklar yakınsadığında ağın öğrenmesinin gerçekleştiği ya da ezberleme durumları; yakınsamadığında ise kullanılan veri setinin modele uygunluğu veya sistem parametrelerinin yetersizliği incelenmektedir.

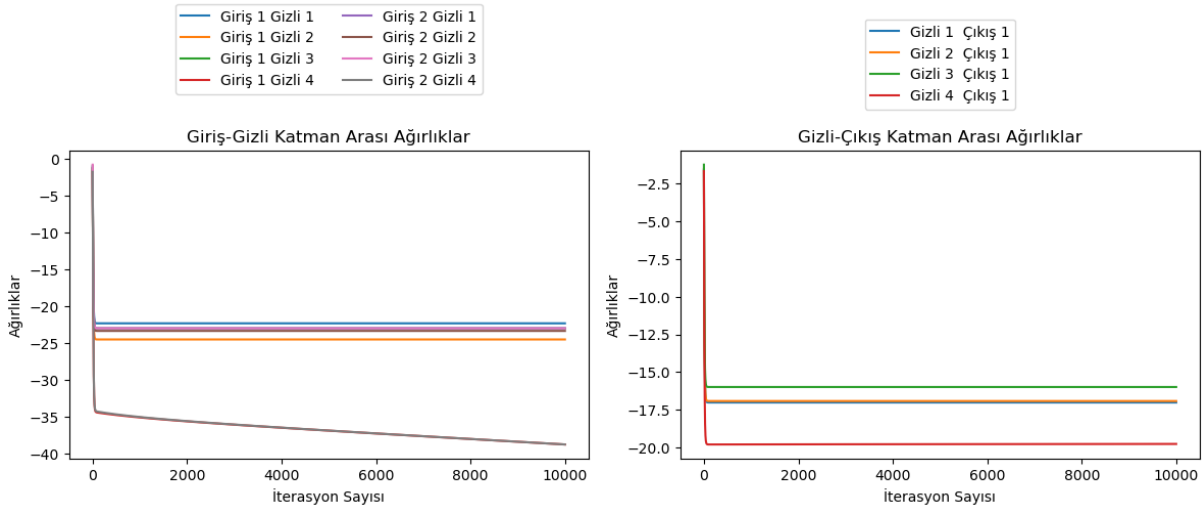
Tablo 1'de Momentum yöntemine ait eğitim ve test verilerinin MSE değerleri incelendiğinde, bu değerlerin birbirine yakın olduğu gözlemlenmiştir. Bu durum, modelin ezberleme yapmadığını ve öğrenmenin başarılı bir şekilde gerçekleştiğini göstermektedir. Momentum yönteminde, giriş katmanı ile gizli katman ve gizli katman ile çıkış katmanı arasındaki ağırlık değişim grafikleri Şekil 4 ile gösterilmektedir. Modele ait ağırlık değişim grafikleri incelendiğinde, iterasyonlar ilerledikçe bazı ağırlık değerlerinin kararlı bir şekilde yakınsadıkları görülmektedir. Fakat bazı ağırlık değerlerinde değişimlerin devam etmesi (yakınsamama durumu) ağın daha fazla eğitime ihtiyaç

duyması, kullanılan veri setinin modele uygun olmaması veya parametrelerin değiştirilmesi gerektiğini göstermektedir.



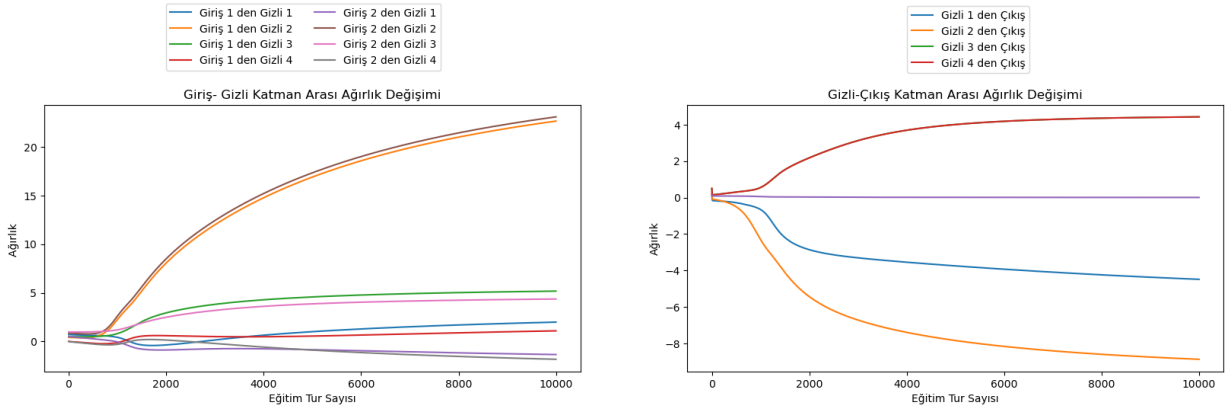
Şekil 4. Öğrenme katsayısının 1,4 değeri için Momentumlu GD yöntemine ait ağırlık değişimleri.

ADAM yönteminde, özellikle öğrenme katsayısının 0,5-3,5 aralığındaki değerleri için eğitim ve test aşamalarındaki MSE değerlerinin oldukça yakın ve büyük değerler olduğu Tablo 1'de görülmektedir. Modelin Şekil 5 ile gösterilen değişim grafikleri incelendiğinde ağırlık değerlerinin istikrarlı bir şekilde yakınsadığı görülmektedir. Tablo ve grafikler birlikte incelendiğinde de modelin ezberleme yapmadığı ancak yerel bir minimum noktaya takıldığı gözlemlenmektedir.



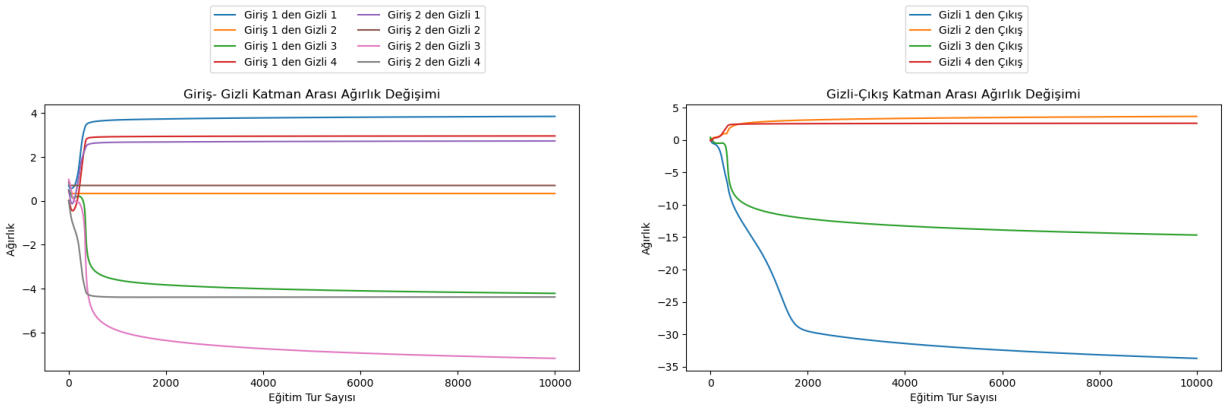
Şekil 5. Öğrenme katsayısının 1,4 değeri için ADAM yöntemine ait ağırlık değişimleri.

SGD yöntemine ait eğitim ve test verilerinin Tablo 1 ile gösterilen MSE değerleri incelendiğinde, bazı öğrenme katsayısı değerleri için eğitim hatalarının test hatalarına göre çok düşük olduğu görülmektedir. Bu durum, modelin ezberleme yaptığına işaret etmektedir. Modelin Şekil 6 ile gösterilen ağırlık değişim grafiği incelendiğinde ağırlık değerlerinden bazılarının yakınsayamadığı görülmektedir. Burada modelin tam olarak öğrenemediği ve kararsız bir öğrenme sürecinde olduğu söylenebilir. Bu problemin çözümü için ağın daha fazla eğitilmesi, modele uygun bir veri setinin seçilmesi veya parametrelerin değiştirilmesi gibi yöntemler uygulanabilir.



Şekil 6. Öğrenme katsayısının 1,4 değeri için SGD yöntemine ait ağırlık değişimleri.

KarcıFANN yöntemine ait eğitim ve test verilerinin Tablo 1 ile verilen MSE değerlerine bakıldığında bazı alfa değerlerinde öğrenmenin gerçekleşmediği görülmektedir. KarcıFANN yöntemi, 0,8-2,1 aralığındaki alfa değerlerinde oldukça başarılı bir öğrenme gerçekleştirmektedir. Modelin Şekil 7 ile gösterilen ağırlık değişim grafiğinde, çoğu ağırlık değerlerinin kararlı bir şekilde yakınsadıkları görülmektedir. Yaklaşık 500 iterasyonda çoğu ağırlık değerinin yakınsaması ve tüm eğitim sürecinde kararlı bir şekilde devam etmesi öğrenmenin başarılı bir şekilde gerçekleştiğini göstermektedir.



Şekil 7. Alfanın 1,4 değeri için KarcıFANN yöntemine ait ağırlık değişimleri.

5. Sonuçlar

YSA'ların eğitimi aşamasında hataların düşmesi ve ağırlıkların eğitim boyunca kararlı bir şekilde yakınsaması istenir. YSA'ların başarılı bir şekilde eğitilip eğitilmediğini öğrenmek amacıyla hata değerleriyle beraber ağırlık değişimlerinin incelenmesi önemlidir. Bu sayede, ağın öğrenme durumu ve bir yerel minimum noktaya takılıp takılmadığı anlaşılmaktadır.

Bu makalede, XOR probleminin çözümünde YSA'larda yaygın olarak kullanılan Momentumlu GD, ADAM ve SGD yöntemleri ile KarcıFANN yönteminin öğrenme, ezberleme ve ağırlık değerleri üzerine etkileri incelenmiştir. Yapılan deneysel çalışmalar sonucunda elde edilen MSE tablosu incelendiğinde, 0,9 ve 1,0 dışındaki tüm alfa ve öğrenme oranı katsayıları için Momentumlu GD yönteminin diğer yöntemlerden daha düşük MSE değerleri ürettiği görülmektedir. Mometumlu GD yönteminin eğitim ve test aşamalarında üretilen MSE değerlerinin yakın olması, modelin ezberlemediğini göstermektedir. KarcıFANN yöntemi, alfa ve öğrenme katsayısının 0,9-2,0 aralığındaki değerleri için SGD yöntemine ve 0,3 değeri ile 0,6-3,5 aralığındaki değerler için de ADAM yöntemine göre daha düşük MSE değerleri üretmektedir. Ayrıca, KarcıFANN yönteminde alfa ve öğrenme katsayısının 0,9 ve 1,0 değerleri için üretilen MSE değerleri, tüm veri setlerinde diğer üç yönteme göre daha düşük olarak ölçülmektedir. Ezberleme bakımından incelendiğinde KarcıFANN yönteminde, alfa

katsayısının 0,8-2,1 aralığı dışındaki değerler için üretilen eğitim hatalarının test hatalarına göre düşük olması modelin ezberlemesi anlamına gelebilir. SGD yönteminde de 0,1-0,3 aralığındaki öğrenme katsayıları için modelin ezberlediği söylenebilir. ADAM yöntemi, 0,1 ve 0,2 dışındaki tüm öğrenme katsayısı değerleri için en başarısız sonuçları üretmektedir. ADAM yönteminin eğitim ve test aşamalarında üretilen MSE değerlerinin yakın olması modelin ezberleme yapmadığını ancak, çok büyük hatalar olduğundan öğrenme başarımının düşük olduğunu göstermektedir.

Tüm yöntemlerin grafikleri incelendiğinde, Momentumlu GD ve ADAM yöntemlerinin ağırlık değişimlerinin diğer yöntemlere göre daha iyi yakınsadığı görülmektedir. SGD yönteminde, ağırlık değerlerinin değişimlerinin devam ettiği ve yakınsama sağlanamadığı gözlemlenmiştir. Bu durum, SGD yönteminin kararsız olduğuna işaret etmektedir. KarcıFANN yönteminde de az da olsa ağırlık değişimlerinin devam etmesi, tam olarak yakınsama sağlanamadığını göstermektedir. Bu problemi çözmek için modelin daha fazla eğitilmesi, veri setinin değiştirilmesi ve model parametreleri değiştirilerek daha uygun bir model elde edilmesi gibi yöntemler uygulanabilir.

MSE değerleri ve ağırlık değişim grafikleri birlikte incelendiğinde en başarılı yöntemin Momentumlu GD, ikinci başarılı yöntemin KarcıFANN yöntemi olduğu ve ADAM yönteminin de yerel bir minimum noktaya takıldığı görülmektedir.

KarcıFANN yöntemi, görüntü sınıflandırma gibi problemler üzerinde farklı aktivasyon ve farklı kayıp fonksiyonları kullanılarak test edilmiş ve başarılı sonuçlar elde edilmiştir [24-26].

Tam optimize edilmemiş KarcıFANN yönteminde yapılacak iyileştirmelerle yakınsama kabiliyetinin artacağı ve daha başarılı modeller elde edileceği düşünülmektedir.

Bu çalışmada KarcıFANN yöntemi XOR problemi üzerinde test edilerek ağırlık değişimleri incelenmiştir. Daha önceki çalışmalarda yöntem görüntü işleme veri kümelerinde uygulanmış ve sınıflandırma performansı değerlendirilmiştir. Gelecek çalışmalarda, KarcıFANN yönteminde kullanılan alfa parametresi, farklı değer aralıklarında seçilerek genellenebilirlik, daha yüksek doğruluk ve güvenilirlik elde etmek amacıyla farklı ağ modelleri ile metin sınıflandırma, görüntü ve ses işleme ile ilgili veri setlerini sınıflandırmada kullanılacaktır.

Kaynaklar

[1] Abraham A. Meta-Learning Evolutionary Artificial Neural Networks, Neurocomputing Journal, 2004, Vol. 56c, Elsevier Science, Netherlands, (1–38).

[2] Rumelhart DE, Durbin R, Golden RM, Chauvin Y. Backpropagation: the basic theory, 1995.

[3] Smith LN. A disciplined approach to neural network hyper-parameters: Part 1 – Learning rate, batch size, momentum, and weight decay. arXiv preprint arXiv:1803.09820, 2018.

[4] Brutzkus A, Globerson A. Why do larger models generalize better? A theoretical perspective via the XOR problem. Proceedings of the 36th International Conference on Machine Learning (ICML), 2019, 97, 822–830.

[5] Pinto RC, Tavares AR. (2024, September 17). PReLU: Yet Another Single-Layer Solution to the XOR Problem (arXiv:2409.10821v1) [Preprint].

[6] Yang C, Kim H, Adhikari SP, Chua LO. A Circuit-Based Neural Network with Hybrid Learning of Backpropagation and Random Weight Change Algorithms. Sensors. 2017, 17(1):16.

[7] Starodub A, Eliseeva N, Georgiev M. Gradient-based algorithm for tracking the activity of neural network weights changing. EPJ Web of Conferences, 248, 2021, 01012.

[8] Li Jun & Diao, Yongfeng & Li, Mingdong & Yin, Xing., Stability Analysis of Discrete Hopfield Neural Networks with the Nonnegative Definite Monotone Increasing Weight Function Matrix. Discrete Dynamics in Nature and Society. 2009.

[9] Nagel M, Fournarakis M, Bondarenko Y, Blankevoort T. Overcoming Oscillations in Quantization-Aware Training. International Conference on Machine Learning (ICML),2022,

[10] Nagel M, van Baalen M, Blankevoort T, and Welling, M. Data-free quantization through weight equalization and bias correction. In International Conference on Computer Vision (ICCV).

[11] Banner R, Nahshan Y, Soudry D. Post training 4-bit quantization of convolutional networks for rapiddeployment. In Advances in Neural Information Processing Systems (NeuRIPS), arXiv:1810.05723v3 [cs.CV] 29 May 2019.

[12] Agrawal AM, Tendle A. (2020). Investigating learning in deep neural networks using layer-wise weight change [Conference paper]. Proceedings of the International Joint Conference on Neural Networks (IJCNN), 1-8., 2020, IEEE.

[13] Wei JL, Wu GC, Liu BQ. et al. An optimal neural network design for fractional deep learning of logistic growth. Neural Comput & Applic 35, 10837–10846 (2023).

[14] Bottou L. Stochastic gradient learning in neural networks. Proceedings of Neuro-Nımes, 1991, 91(8), 12.

[15] Kingma DP, Ba J, Adam: A Method for Stochastic Optimization. Proceedings of the 3rd International Conference on Learning Representations (ICLR), Banff, 14-16 April 2014.

[16] Fu J, Wang B, Zhang H, Zhang , Chen W, Zheng N, When and why momentum accelerates sgd: An empirical study, 2023, arXiv preprint arXiv:2306.09000.

[17] Seyyarer E, Ayata F, Uçkan T, Karcı A. Applıcatıons and comparıson of optımızatıon algorıthms used ın deep learnıng. Anatolian Science , 2020, vol.5, no.2, 90-98.
[18] Karcı A. A New Approach for Fractional Order Derivative and Its Applications, Universal Journal of Engineering Sciences, 2013, Vol:1, pp: 110-117.
[19] Karcı A. Properties of Fractional Order Derivatives for Groups of Relations/Functions, Universal Journal of Engineering Sciences, 2015a , vol:3, pp:39-45.
[20] Karcı A. The Properties of New Approach of Fractional Order Derivative, Journal of the Faculty of Engineering and Architecture of Gazi University, 2015b ,Vol.30, pp:487-501.
[21] Karci A. Chain rule for fractional order derivative, Science Innovation, 2015c, Vol:3, pp:63-67.
[22] Karcı A. Properties of Karcı's Fractional Order Derivative, Universal Journal of Engineering Science, 2019, Vol:7, pp:32-38.
[23] Karci A. Fractional Order Integration: A New Perspective based on Karcı's Fractional Order Derivative. Computer Science, 2019, 6(2), 102-105.
[24] Karakurt M, Saygılı H, Karcı A. RMSE Yönteminde Aktivasyon Fonksiyonlarının Karşılaştırılması, 8th International Artificial Intelligence and Data Processing Symposium (IDAP2024), IEEE.
[25] Karakurt M, Saygılı H, Karcı A. Karcı fractional artificial neural networks (KarcıFANN): a new artificial neural networks model without learning rate and its problems. Turkish Journal of Electrical Engineering and Computer Sciences, 33(3), 2025; 248-263.
[26] Saygılı H, Karakurt M, Karcı A. Comparison of Loss Functions in the KarcıFANN Method. In 2024 8th International Artificial Intelligence and Data Processing Symposium (IDAP), 2024 ;(pp. 1-5). IEEE.