Fırat Üniversitesi Müh. Bil. Dergisi
38(1), 47-56, 2026

Araştırma Makalesi
https://doi.org/10.35234/fumbd.1672351

KarcıFANN Makine Öğrenmesi Yönteminin Matematiksel Modeli

Meral KARAKURT[1*], Hülya SAYGILI[2], Ali KARCI[3]
[1] Bilgisayar Teknolojileri Bölümü, Osmaniye Korkut Ata Üniversitesi Düziçi Meslek Yüksekokulu, Osmaniye, Türkiye
[2] Karayolları 8.Bölge Müdürlüğü, Elazığ, Türkiye
[3] Bilgisayar Mühendisliği Bölümü, İnönü Üniversitesi, Malatya, Türkiye
[*1]meralkarakurt@osmaniye.edu.tr, [2]hulya-saygili@hotmail.com, [3]ali.karci@inonu.edu.tr

(Geliş/Received: 08/04/2025; Kabul/Accepted: 12/01/2026)

Öz: Yapay sinir ağlarının (YSA) eğitiminin başarılı bir şekilde gerçekleşmesi birçok hiperparametreye bağlıdır. Optimizasyon yöntemi, başarımı doğrudan etkileyen hiperparametrelerden biridir. YSA çalışmalarında gradyan (türev) tabanlı optimizasyon yöntemleri popüler olarak kullanılmaktadır. Optimizasyon sürecinde türevlenebilir fonksiyonların kullanılması, her eğitim adımında hata değerlerinin ölçülmesini ve bu hataların düşürülmesini sağlamaktadır. Gradyan tabanlı Stokastik Gradyan İniş (SGD) yönteminde sabit bir öğrenme katsayısı parametresi kullanılmaktadır. Öğrenme katsayısının başarıma etkisinin tam olarak bilinememesi, önemli bir problemdir. Bu problem, modelin tam olarak ifade edilememesine neden olmaktadır. Bu çalışmada, SGD yöntemindeki öğrenme katsayısı yerine Karcı kesir dereceli türevin kullanıldığı KarcıFANN yöntemi ile literatürde yer alan diğer türev tabanlı Gradyan İniş (GD), Stokastik Gradyan İniş (SGD), momentumlu SGD, Uyarlanabilir gradyan algoritması (AdaGrad) ve Ortalama Karekök Yayılımı (RMSProp) optimizasyon yöntemleri matematiksel olarak ifade edilmektedir. KarcıFANN yöntemiyle ağa dışarıdan müdahale minimuma indirilerek diğer yöntemlerde karşılaşılan öğrenememe, ezberleme, türevlerin kaybolması ve türev patlaması gibi önemli problemler daha başarılı bir şekilde çözülmektedir.

Anahtar kelimeler: KarcıFANN, yapay sinir ağları, momentumlu SGD, AdaGrad, RMSProp.

Mathematical Model of KarcıFANN Machine Learning Method

Abstract: The successful training of artificial neural networks (ANNs) depends on many hyperparameters. Among these, the optimization method is one of the hyperparameters that directly affects performance. In ANN studies, gradient (derivative)-based optimization methods are widely used. The use of differentiable functions in the optimization process enables the error values to be measured at each training step and allows these errors to be minimized. In the gradient-based Stochastic Gradient Descent (SGD) method, a fixed learning rate parameter is used. The fact that the exact effect of the learning rate on performance cannot be fully determined constitutes a significant problem. This problem prevents the model from being properly expressed.In this study, the KarcıFANN method, which replaces the learning rate in the SGD method with the Karcı fractional-order derivative, is mathematically formulated together with other derivative-based optimization methods in the literature, including Gradient Descent (GD), Stochastic Gradient Descent (SGD), momentum-based SGD, Adaptive Gradient Algorithm (AdaGrad), and Root Mean Square Propagation (RMSProp). With the KarcıFANN method, external intervention in the training process is minimized, and important problems encountered in other methods—such as failure to learn, overfitting, vanishing gradients, and exploding gradients—are addressed more successfully.

Keywords: KarcıFANN, artificial neural network, momentum-based SGD, AdaGrad, RMSProp.

1. Giriş

Gerçek hayatta problemlerin matematiksel olarak modellenmesi amacıyla Yapay sinir ağları (YSA) kullanılmaktadır. YSA'ların eğitimi belli iterasyonlar boyunca veya hata değeri belirli bir seviyeye düşene kadar devam etmektedir. Optimum bir eğitim sürecinde, hata değerinin her iterasyonda değişmesi (düşmesi) gerekir ve bu değişimleri ifade etmek için de türev kavramı kullanılır. Türev, bağımsız değişkenlerdeki değişime karşılık bağımlı değişkenin cevabıdır [1].

YSA'larda kullanılan önemli bir hiperparametre optimizasyon yöntemidir. Literatürde, özellikle doğrusal olmayan problemlerin çözümünde, türevlenebilir optimizasyon yöntemleri ile ilgili birçok çalışma bulunmaktadır

* Sorumlu yazar: meralkarakurt@osmaniye.edu.tr Yazarların ORCID Numarası: [1]0000-0001-7318-2798, [2] 0000-0003-1926-9918, [3]0000-0002-8489-8617

[2-8]. Türevlenebilir optimizasyon yöntemlerinde, tüm iterasyonlar ya da belli sayıda iterasyon için sabit tutulan öğrenme katsayısının seçiminde bir standardın olmaması, YSA'lar için önemli bir problemdir. Çok büyük öğrenme katsayıları global minimum (ya da maksimum) noktadan uzaklaşmaya, çok küçük öğrenme katsayıları ise eğitim süresinin uzamasına ve yerel minimum (ya da maksimum) noktalara takılmasına neden olmaktadır. Bu problemlerin çözülmesinde kesir dereceli türev yaklaşımı çok önemlidir. Kesir dereceli türev yaklaşımı, sistemleri daha iyi ifade edebileceği ve global modellemede klasik türevden (tamsayılar kullanılarak hesaplanan Newton türevinden) daha başarılı olabileceği fikri ile ortaya atılmış ve geniş bir çalışma alanına sahip olmuştur [7-13].

Bu çalışmada matematiksel olarak ifade edilen KarcıFANN (Karcı Kesir Dereceli Yapay Sinir Ağı - Karcı Fractional Artificial Neural Networks) yönteminde, öğrenme katsayısı yerine Karcı kesir dereceli türev kullanılmakta ve öğrenme katsayısının sebep olduğu problemler daha başarılı bir şekilde çözülmektedir. Çalışmanın devamı dört bölümden oluşmaktadır. 2. bölümde literatürdeki ilişkili çalışmalar açıklanmakta, 3. Bölümde YSA'larla ilgili bilgilendirme yapılmakta, 4. bölümde optimizasyon algoritmalarının matematiksel modellenmeleri gösterilmekte ve 5. bölümde de sonuç yer almaktadır.

2. İlişkili Çalışmalar

Karcı, KarcıFANN yönteminin temelini oluşturan kesir dereceli türevin yeni tanımını yaparak literatürde yer alan kesir dereceli türev yöntemlerinin eksiklerini ortaya koymuştur [9-14]. Karcı kesir dereceli türevin kullanıldığı ve KarcıFANN'ın yeni bir optimizasyon algoritması olarak sunulduğu ilk çalışma Karakurt vd. [4] tarafından yapılmıştır. İlgili çalışmada, XOR probleminin çözümünde KarcıFANN yöntemi ile Stokastik Gradyan İniş (SGD) yöntemi karşılaştırılmıştır. Deneysel bulgular, KarcıFANN yönteminin SGD'ye alternatif bir yöntem olabileceğini ortaya koymuştur. KarcıFANN ve SGD yöntemlerinin karşılaştırıldığı diğer bir çalışmada, Karakurt vd. [15] MNIST (Modified National Institute of Standards and Technology) ve Fashion-MNIST veri setleri üzerinde gerçekleştirdikleri sınıflandırma deneylerinde KarcıFANN yönteminin dikkate değer bir performans sergilediğini ve yüksek düzeyde potansiyel taşıdığını göstermişlerdir. Saygılı vd. [16] KarcıFANN yönteminin genelleme performansını ölçmek amacıyla farklı görüntü sayısı ve sınıflardan oluşan Kuzushiji-MNIST, SignMNIST ve Gina_prior2 veri setlerini sınıflandırmışlardır. Elde ettikleri sonuçlar ile KarcıFANN yönteminin ezberleme, öğrenememe, gradyan patlaması gibi problemleri çözdüğünü ve global modellemede kullanılabileceğini göstermişlerdir. Benzer amaçla yapılan başka bir çalışmada, Karakurt [17] üç katmanlı ve dört katmanlı KarcıFANN ve SGD modelleri tasarlayarak MNIST ve Dry Bean veri setlerini sınıflandırmıştır. Sigmoid ve ReLU (Düzeltilmiş Doğrusal Birim) aktivasyon fonksiyonları ile MSE (Ortalama Karesel Hata) ve CE (Çapraz Entropi) hata fonksiyonlarını kullandığı farklı modellerin performanslarını doğruluk, MSE, CE, kesinlik, duyarlılık ve F1 skoru metrikleri ile ölçmüştür. İlgili çalışma, KarcıFANN yönteminin tam anlamıyla optimize edilmemiş olmasına rağmen kayda değer bir genelleme yeteneğini göstermesi bakımından önemlidir. Saygılı vd. [18], KarcıFANN yöntemi ile SGD, ADAM ve Momentumlu Gradyan İniş (GD) yöntemlerini karşılaştırdıkları çalışmada, KarcıFANN yönteminin Momentumlu GD yöntemine yakın sonuçlar ürettiğini ve diğer yöntemlerden başarılı olduğunu göstermişlerdir. Karakurt vd. [7], KarcıFANN yöntemi ile sigmoid, hiperbolik tanjant (tanh) ve hiperbolik tanjant sigmoid (tansig) aktivasyon fonksiyonlarını kullanarak tasarladıkları modellerin performanslarını karşılaştırmışlardır. Sigmoid fonksiyonunu ile tasarlanan modelin en yüksek performansa sahip olduğunu gözlemlemişlerdir. Saygılı vd. [8] KarcıFANN yöntemi ile ortalama karesel hata (MSE), ortalama mutlak hata (MAE) ve ortalama karesel hatanın karekökü (RMSE) hata fonksiyonlarını kullanarak tasarladıkları modellerin başarımlarını karşılaştırmışlardır.

Literatürde çeşitli optimizasyon algoritmalarının kullanıldığı daha birçok çalışma yapılmıştır. Örneğin; Dogo vd. yaptıkları çalışmada evrişimli sinir ağ (ConvNet) mimarilerinde en sık kullanılan optimizasyon algoritmalarından Stokastik Gradyan İniş (SGD), vanilya ile (vSGD), momentum ile (SGDm), momentum ve nesterov ile (SGDm+n), Ortalama Karekök Yayılımı (RMSProp), Uyarlamalı Moment Tahmini (Adam), Uyarlamalı Gradyan (AdaGrad), Uyarlamalı Delta (AdaDelta), Sonsuzluk normuna dayalı Uyarlamalı moment tahmini Uzantısı (Adamax) ve Nesterov hızlandırmalı Uyarlamalı Moment Tahmini (Nadam) kullanılarak yakınsama hızı, doğruluk ve kayıp fonksiyonu açısından performansları karşılaştırmışlardır. Üç farklı görüntü sınıflandırma veri kümesi üzerinde yaptıkları deneysel çalışmalar sonucunda Nadam algoritmasıyla daha iyi performans elde etmişler ve AdaDelta yönteminin de en kötü performansı sergilediğini gözlemlemişlerdir [3]. Benzer bir çalışmada, Tian vd. SGD yönteminin varyantlarını, avantajlarını ve dezavantajlarını ele alarak iris veri kümesinde deneysel çalışmalar yapmışlardır. SGD yöntemini, vanilya gradyan iniş ve toplu gradyan iniş ile karşılaştırarak en düşük yürütme süresine sahip olduğunu gözlemlemişlerdir. Ayrıca SGD varyantlarını evrişimli sinir ağları (CNN) üzerinde uygulayarak AdaMax'ın diğer SGD tabanlı optimize edicilere göre üstün performansa sahip olduğunu yaptıkları deneysel çalışmalar sonucunda elde etmişlerdir [6]. Diğer bir çalışmada ise Zinkevich

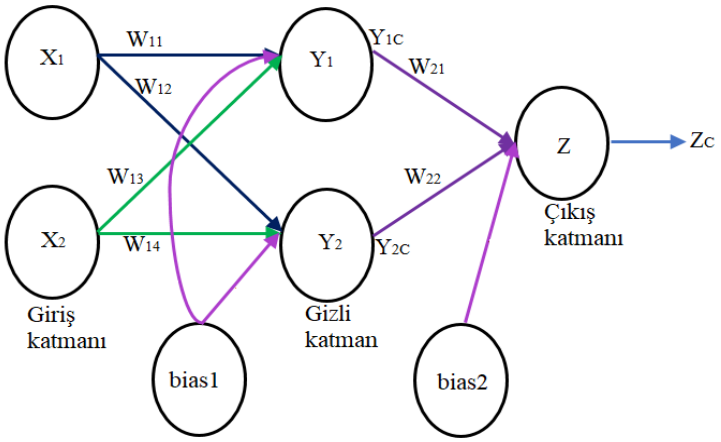
vd., paralel makine öğrenmesinde kullanılmak amacıyla paralel Stokastik Gradyan İniş algoritmasını öneren bir çalışma sunmuşlardır [2].

Reddi vd., SGD yönteminde birçok varyans azaltma tekniği için bir çerçeve sunmuş ve asenkron bir algoritma önermişlerdir. Önerdikleri yöntemi varyansı azaltılmamış asenkron SGD yöntemleriyle karşılaştırmak amacıyla yapılan deneysel çalışmaların sonucunda SGD'ye kıyasla adım boyutuna daha az duyarlı olduğunu ve artan iş parçacıklarına karşı daha dayanıklı olduğunu gözlemlemişlerdir [19]. Başka bir çalışmada, Chen ve ark. önermiş oldukları Lion optimizasyon yöntemini ImageNet veri setinde farklı görevlerde modelleri eğitmek için Adam ve Adafactor optimizasyon yöntemleri ile karşılaştırmışlardır. Önermiş oldukları yöntemin eğitim grubu boyutu arttıkça performansının arttığını, Adam'a göre daha küçük öğrenme oranıyla çalıştığını ve Adam ile Adafactor yöntemlerine benzer ya da daha iyi performans elde ettiklerini göstermişlerdir [5].

3. Yapay Sinir Ağları (YSA)

YSA, yapay sinir hücreleri (nöronlar) ve bu sinir hücreleri arasındaki bağlantılardan oluşmaktadır. YSA, biyolojik beyne gelen verilerin beyinde çeşitli işlemlerden geçerek düşünme, hatırlama, fikir oluşturma gibi eylemler oluşturmasını matematiksel olarak modellemektir. Günümüzün karmaşık problemlerinin çözümünde kullanılan YSA modelleri için kullanılan belli bir standart model yoktur. Yani, her problemin çözümünü başarıyla gerçekleştirecek model, katman sayısı, nöron sayısı, aktivasyon fonksiyonu ve optimizasyon yöntemleri gibi parametrelerin değiştirilmesiyle elde edilmektedir.

Bir YSA genellikle bir tane giriş katmanı, en az bir tane gizli katman ve bir tane çıkış katmanından oluşmaktadır. Bu bölümde, geri yayılımlı YSA modeli anlatılacak olup kullanılan YSA modeli Şekil 1 ile gösterilmektedir. YSA'larda, giriş katmanı dışındaki katmanlarda kullanılan ve nöronların pasif duruma düşmesini engelleyen önemli bir parametre bias değeridir. Yani bias, nöronların eğitime katılmasını sağlamak amacıyla kullanılır. Aktivasyon fonksiyonu, bir nöronun değer üretip üretmeyeceğini belirleyen diğer bir parametredir. YSA'nın eğitimi için giriş katmanındaki veriler ilgili ağırlıklarla çarpılıp toplanmakta ve aktivasyon fonksiyonundan geçirilerek ikinci katmandaki nöronların çıkış değerlerini üretmektedir. İkinci katmanda da aynı işlemler yapılarak YSA'nın çıkış değeri üretilmektedir. Giriş katmanından çıkışa doğru yapılan bu işlemler dizisine ileri besleme (yayılım) denilmektedir.



Şekil 1. Üç katmanlı YSA modeli

İleri besleme aşamasında, X_1 ve X_2 girişler, Y_1 ve Y_2 gizli katman nöronları, Z çıkış katmanı nöronu, W'ler ağırlık değerleri ve f aktivasyon fonksiyonu olmak üzere gizli katman çıkışları Y_{1C}ve Y_{2C} Denklem 1, modelin çıkış değeri Z_C ise Denklem 2 ile gösterildiği gibi hesaplanmaktadır.

$$Y_1 = X_1.W_{11} + X_2.W_{13} + bias1\,, \quad Y_{1c} = f(Y_1),$$

$$Y_2 = X_1.W_{12} + X_2.W_{14} + bias1\,, \quad Y_{2c} = f(Y_2) \tag{1}$$

$$Z = Y_{1C}.W_{21} + Y_{2C}.W_{22} + bias2\,, \quad Z_c = f(Z) \tag{2}$$

YSA'nın eğitim sürecinde elde edilen çıkış değeri ile ağın üretmesi istenen çıkış değeri arasındaki fark hata değeri olarak ölçülmektedir. Hata fonksiyonunun türevlenebilir olması çok önemlidir. Bu hata değerini düşürmek amacıyla çıkış katmanından giriş katmanına doğru hataların yayılması ve ağırlıkların güncellenmesi işlemlerine geri besleme denilmektedir. Geri besleme sonucunda elde edilen yeni ağırlık değerleri ile tekrar ileri besleme yapılarak bu işlemler minimum olarak kabul edilen bir hata değerine ulaşıncaya dek veya belli bir iterasyon sayısı kadar yinelemeli olarak devam etmektedir.

4. Optimizasyon Yöntemleri

Optimizasyon, bir problemi çözebilen en iyi sonucu bulma işlemidir. YSA'ların başarımını etkileyen önemli parametreleden biridir. Bu bölümde, Şekil 1 ile gösterilen YSA modelinde çeşitli optimizasyon algoritmaları kullanılarak geri yayılım süreçleri açıklanmaktadır.

4.1. Gradyan iniş yöntemi (gradient descent - GD)

GD yöntemi, YSA'ların eğitiminde hata (kayıp) fonksiyonunun değerini en aza indirmek için fonksiyonun ağırlıklara göre gradyanının (türevinin) yinelemeli olarak hesaplanmasıyla tüm ağırlıkların güncellenmesidir. Her iterasyonda veri setinin tümü ağa sunulmaktadır. Gradyan iniş yöntemi, ilk defa 1847'de matematiksel optimizasyon için öneren Augustin-Louis Cauchy'e atfedilmiştir [20]. Gradyan iniş, Rumelhart vd. (1986) tarafından önerilen geri yayılım yönteminde ağırlıkların güncellenmesi amacıyla kullanılmıştır [21]. LeCun (1986) ve Rumelhart vd. (1986) yaptıkları çalışmalarda gradyan iniş yönteminin YSA'ların optimizasyonunda kullanılabileceğini göstermişlerdir [21, 22]. Algoritmanın temel adımları aşağıdaki gibidir:

1. İlk hata değeri hesaplandıktan sonra optimizasyon süreci başlatılır. Şekil 1 ile gösterilen X_1 nöronuna ait W_{11} ağırlığının toplam hataya (H) etkisini hesaplamak için Newton türevi kullanılarak Denklem 3 ile gösterilen zincir kuralı uygulanmaktadır.

$$\frac{\partial_H}{\partial_{W_{11}}} = \frac{\partial_H}{\partial_{Z_C}} . \frac{\partial_{Z_C}}{\partial_Z} . \frac{\partial_Z}{\partial_{Y_{1C}}} . \frac{\partial_{Y_{1C}}}{\partial_{Y_1}} . \frac{\partial_{Y_1}}{\partial_{W_{11}}} \tag{3}$$

   Tüm ağırlık değerleri için Denklem 3'e benzer şekilde gradyan hesaplaması yapılır. Bu gradyanlar, fonksiyonun hangi yönde ve ne kadar hızla değiştiğini göstermektedir.

2. Gradyan hesaplaması yapıldıktan sonra öğrenme katsayısı α olmak üzere, tüm veri setinin ağırlık parametrelerinin (W) herhangi bir t adımında gradyan iniş algoritması ile güncellenmesi işlemi, W_{11} ağırlığı için hesaplanan Denklem 4 ile gösterildiği gibi formüle edilmektedir.

$$w_{11t+1} = w_{11t} - \alpha . \frac{\partial H}{\partial w_{11t}} \tag{4}$$

3. Hata değeri, kabul edilebilir bir küçüklüğe ulaşıncaya ya da belirli bir iterasyon sayısı tamamlanıncaya kadar 1. ve 2. adımlar tekrarlanmaktadır.

GD yöntemi, büyük veri setlerinde etkili çalışabilmekte fakat bazı durumlarda yerel minimumlara takılma, gradyanların kaybolması, gradyan patlaması, ezberleme ve öğrenememe gibi problemlere neden olmaktadır.

Şekil 1'de verilen YSA için ağırlıkların güncelleme denklemleri için türev bağıntıları aşağıda verilmiştir. Bu ağ için R, beklenen çıkış değeri olmak üzere hata bağıntısı $H = (R - Zc)^2$ olarak verilmiş olsun. Şekil 1'de verilen ağın ağırlıkları ve düğüm fonksiyonları $W_{11} = W(1,1,1), W_{12} = W(1,1,2), W_{13} = W(1,2,1), W_{14} = W(1,2,2), W_{21} = (2,1,1), W_{22} = W(2,2,1), Y_1 = Y(2,1,1), Y_2 = Y(2,2,1), Y_{1C} = Y(2,1,2), Y_{2C} = Y(2,2,2),$ $Z = Y(3,1,1)$ ve $Z_C = Y(3,1,2)$ olsun. Klasik YSA için ağırlık değişimleri (ΔW) Denklem 5, Denklem 6, Denklem 7, Denklem 8, Denklem 9 ve Denklem 10'da sunulan formüller ile hesaplanmaktadır.

$$\Delta W(1,1,1) = \frac{\partial H}{\partial Y(3,1,2)} . \frac{\partial Y(3,1,2)}{\partial Y(3,1,1)} . \frac{\partial Y(3,1,1)}{\partial Y(2,1,2)} . \frac{\partial Y(2,1,2)}{\partial Y(2,1,1)} . \frac{\partial Y(2,1,1)}{\partial W(1,1,1)} \tag{5}$$

$$\Delta W(1,1,2) = \frac{\partial H}{\partial Y(3,1,2)} . \frac{\partial Y(3,1,2)}{\partial Y(3,1,1)} . \frac{\partial Y(3,1,1)}{\partial Y(2,2,2)} . \frac{\partial Y(2,2,2)}{\partial Y(2,2,1)} . \frac{\partial Y(2,2,1)}{\partial W(1,1,2)} \tag{6}$$

$$\Delta W(1,2,1) = \frac{\partial H}{\partial Y(3,1,2)} . \frac{\partial Y(3,1,2)}{\partial Y(3,1,1)} . \frac{\partial Y(3,1,1)}{\partial Y(2,1,2)} . \frac{\partial Y(2,1,2)}{\partial Y(2,1,1)} . \frac{\partial Y(2,1,1)}{\partial W(1,2,1)} \tag{7}$$

$$\Delta W(1,2,2) = \frac{\partial H}{\partial Y(3,1,2)} . \frac{\partial Y(3,1,2)}{\partial Y(3,1,1)} . \frac{\partial Y(3,1,1)}{\partial Y(2,2,2)} . \frac{\partial Y(2,2,2)}{\partial Y(2,2,1)} . \frac{\partial Y(2,2,1)}{\partial W(1,2,2)} \quad (8)$$

$$\Delta W(2,1,1) = \frac{\partial H}{\partial Y(3,1,2)} . \frac{\partial Y(3,1,2)}{\partial Y(3,1,1)} . \frac{\partial Y(3,1,1)}{\partial W(2,1,1)} \quad (9)$$

$$\Delta W(2,2,1) = \frac{\partial H}{\partial Y(3,1,2)} . \frac{\partial Y(3,1,2)}{\partial Y(3,1,1)} . \frac{\partial Y(3,1,1)}{\partial W(2,2,1)} \quad (10)$$

Sigmoid aktivasyon fonksiyonun (Sig) kullanılması durumunda ağırlık değişimleri (ΔW) Denklem 11, Denklem 12, Denklem 13, Denklem 14, Denklem 15 ve Denklem 16'da sunulan formüller ile hesaplanmaktadır.

$$\Delta W(1,1,1) = -2\big(R - Y(3,1,2)\big) Sig\big(Y(3,1,1)\big)\big[1 - Sig\big(Y(3,1,1)\big)\big] W(1,1,1) Sig\big(Y(2,1,1)\big)\big[1 - Sig\big(Y(2,1,1)\big)\big] X_1 \quad (11)$$

$$\Delta W(1,1,2) = -2\big(R - Y(3,1,2)\big) Sig\big(Y(3,1,1)\big)\big[1 - Sig\big(Y(3,1,1)\big)\big] W(1,1,2) Sig\big(Y(2,2,1)\big)\big[1 - Sig\big(Y(2,2,1)\big)\big] X_1 \quad (12)$$

$$\Delta W(1,2,1) = -2\big(R - Y(3,1,2)\big) Sig\big(Y(3,1,1)\big)\big[1 - Sig\big(Y(3,1,1)\big)\big] W(1,2,1) Sig\big(Y(2,1,1)\big)\big[1 - Sig\big(Y(2,1,1)\big)\big] X_2 \quad (13)$$

$$\Delta W(1,2,2) = -2\big(R - Y(3,1,2)\big) Sig\big(Y(3,1,1)\big)\big[1 - Sig\big(Y(3,1,1)\big)\big] W(1,2,2) Sig\big(Y(2,2,1)\big)\big[1 - Sig\big(Y(2,2,1)\big)\big] X_2 \quad (14)$$

$$\Delta W(2,1,1) = -2\big(R - Y(3,1,2)\big) Sig\big(Y(3,1,1)\big)\big[1 - Sig\big(Y(3,1,1)\big)\big] Y(2,1,2) \quad (15)$$

$$\Delta W(2,2,1) = -2\big(R - Y(3,1,2)\big) Sig\big(Y(3,1,1)\big)\big[1 - Sig\big(Y(3,1,1)\big)\big] Y(2,2,2) \quad (16)$$

4.2. Stokastik gradyan iniş algoritması (stochastic gradient descent - SGD)

SGD yöntemi, GD yönteminin kullanımında karşılaşılan problemleri ortadan kaldırmak amacıyla geliştirilmiştir. SGD, eğitim veri kümesindeki tüm verileri bir kerede işlemek yerine küçük gruplar halinde ya da tek tek işleyen bir GD türüdür. Bu özelliği ile SGD, büyük veri setleri için hesaplama açısından da daha verimli olmaktadır [23]. Ağırlık parametrelerinin güncelleme işlemi (4) denklemi ile gösterildiği gibi hesaplanmaktadır.

4.3. Uyarlanabilir gradyan algoritması (adaptive gradient – AdaGrad)

AdaGrad yöntemi, Duchi vd. tarafından 2011 yılında önerilmiş GD tabanlı bir yöntemdir. AdaGrad yöntemi, sabit öğrenme katsayısı problemini ortadan kaldırmak, daha kararlı bir yakınsama süreci elde etmek ve ezberlemeyi azaltmak amacıyla geliştirilmiştir. Her adımda farklı bir öğrenme oranı kullanılmaktadır [24, 25]. Her ağırlık parametresinin yeni (güncel) değeri, o parametrenin önceki gradyanlarının (g) karelerinin kümülatif (biriktirilmiş) toplamına bağlıdır. Herhangi bir zamanda (t), w_{11} ağırlığı için G kümülatif gradyanlar Denklem 17 ile hesaplanmaktadır.

$$G_t = G_{t-1} + (g_t)^2, \quad G_t = G_{t-1} + \left(\frac{\partial H}{\partial w_{11t}}\right)^2 \quad (17)$$

α başlangıç öğrenme oranı ve ϵ sıfıra bölmeyi önlemek (sayısal kararlılığı artırmak) için kullanılan küçük bir değer olmak üzere Şekil 1'deki w_{11} parametresinin bir t adımında güncelleme işlemi Denklem 18 ile gösterilmektedir.

$$w_{11t+1} = w_{11t} - \frac{\alpha}{\sqrt{G_{t+\epsilon}}} . \frac{\partial H}{\partial w_{11t}} \quad (18)$$

Gradyan değerleri küçük olduğunda yakınsamayı hızlandırmak için öğrenme katsayısı büyütülmekte ve gradyan değerleri büyüdüğünde hata fonksiyonunun minimum değerinin aşılmaması için de öğrenme katsayısı küçültülmektedir. Bu durum dinamik ve değişken problemlerin çözümünde AdaGrad yönteminin daha başarılı olmasını sağlamaktadır. AdaGrad yöntemi ile doğal dil işleme ve görüntü işlemede seyrek veri setlerinde daha iyi sonuçlar elde edilmektedir [24].

4.4. Momentumlu SGD

Stokastik gradyan inişi (SGD) yönteminde karşılaşılan çok küçük öğrenme oranının çok zaman alması ve çok büyük öğrenme oranının da salınımlara neden olması ve hedef noktadan uzaklaştırma problemlerini gidermek amacıyla ortaya atılmış bir optimizasyon yöntemidir. Momentumlu SGD yöntemi, SGD yönteminden daha hızlı bir yakınsama sağlamakla beraber SGD yönteminde bir problem olan yerel minimumlara takılma problemini başarılı bir şekilde çözmektedir [25, 26]. Momentum, geçmiş adımlardaki eğimlere göre sonraki adımın yönünün belirlenmesidir. Geçmiş eğimlerin yeni gradyan hesaplamada hangi oranda etkileyeceğini belirlemek amacıyla momentum sabiti (β) kullanılmaktadır. β değerleri 0 ile 1 arasında seçilir ve 0'a yaklaştıkça momentumsuz SGD'ye benzer. β'nin 0 değeri momentumsuz SGD yöntemi gibi çalışması demektir. Momentum hesaplamada kullanılacak diğer bir terim olan V terimi, gradyandaki değişimi belirtmek için kullanılan hız terimidir ve başlangıç değeri 0 olarak ayarlanmaktadır. Herhangi bir t zamanda, Şekil 1'deki w_{11} ağırlığının güncellenmesinde kullanılan ve değiştirilmiş adım yönü terimi olarak da adlandırılan V_t değeri Denklem 19 ile hesaplanmaktadır [27].

$$V_t = \beta V_{t-1} + (1-\beta)\frac{\partial H}{\partial w_{11t}} \tag{19}$$

Şekil 1'deki w_{11} ağırlığının güncellenme işlemi Denklem 20 ile gösterildiği gibi hesaplanmaktadır.

$$w_{11t+1} = w_{11t} - \alpha V_t \tag{20}$$

4.5. Ortalama karekök yayılımı (root mean square propagation - RMSProp)

Stokastik gradyan iniş (SGD) yöntemiyle karşılaşılan sorunları çözmek için Geoffrey Hinton tarafından tasarlanmıştır. Uyarlanabilir öğrenme oranının kullanıldığı RMSProp yöntemi, AdaGrad yönteminin bir uzantısıdır. Karesel olarak hesaplanan momentumlu gradyanların hareketli ortalaması kullanılmaktadır. Gradyanlar küçüldükçe öğrenme oranı arttırılmaktadır. Öğrenme oranının uyarlanabilir ve dinamik olarak ayarlanması, RMSProp yönteminin değişken ve karmaşık problemlerin çözümünde daha iyi performans göstermesini sağlamaktadır [28, 29]. V_t, gradyanın karelerinin hareketli ortalamasını; β katsayısı ise genellikle 0.9 gibi bir değeri göstermek üzere V_t değerinin RMSProp yönteminde hesaplanması Denklem 21 ile gösterilmektedir.

$$V_t = \beta V_{t-1} + (1-\beta)\left(\frac{\partial H}{\partial w_{11t}}\right)^2 \tag{21}$$

Şekil 1'deki X_1 nöronuna ait w_{11} ağırlığının güncellenme işlemi Denklem 22 ile gösterilmektedir.

$$w_{11t+1} = w_{11t} - \frac{\alpha}{\sqrt{V_{t+\epsilon}}}.\frac{\partial H}{\partial w_{11t}} \tag{22}$$

Burada, α, başlangıç öğrenme oranı ve ϵ, sayısal kararlılığı artırmak için kullanılan küçük bir değerdir.

4.6. Karcı kesir dereceli YSA (Karcı fractional artificial neural networks – KarcıFANN) yöntemi

KarcıFANN yönteminin anlaşılabilmesi için türev ve Karcı kesir dereceli türev tanımlarının yapılması gerekmektedir.

Türev konusu, matematik temelli birçok disiplinde uzun bir geçmişe sahiptir. Sistemlerdeki değişimlerin miktarını ölçerek bu değişimleri matematiksel olarak ifade etmek amacıyla türev kavramı kullanılmaktadır. Bu kavram ilk kez Isaac Newton tarafından tanımlanmıştır [9]. Newton türevinde tam sayı olarak kullanılan türev derecesine reel sayı verilmesi durumunu incelemek üzere Caputo [30], Ross [31], Riemann–Liouville [32, 33], ve Grünwald–Letnikov [9, 10] türevleri gibi türev tanımları yapılmıştır. Bu tanımların, türevin çarpım ve zincir

kuralları gibi temel özelliklerini tam olarak karşılamaması, özellikle optimizasyon yöntemlerinde doğrudan kullanılmasını sınırlamaktadır [10, 34, 35]. Bu eksiklikler, Karcı [9] tarafından sunulan Karcı kesir dereceli türev ile giderilmektedir. Bu sayede, türevin genel özellikleri korunmakta, YSA'ların öğrenme algoritmalarında etkin bir şekilde kullanılabilmekte ve daha geniş kapsamlı problemlere uygulanabilmektedir.

SGD yönteminde kullanılan sabit öğrenme katsayısı problemini ortadan kaldırmak, daha kararlı bir yakınsama süreci elde etmek, salınım, öğrenememe ve ezberleme gibi problemleri ortadan kaldırmak amacıyla geliştirilmiştir. KarcıFANN yönteminde, her adım için hesaplanan hata değerinin kesir dereceli türevi alınarak ağırlık güncelleme işlemi yapılmaktadır. Böylece, öğrenme katsayısı gibi sabit olan bir değer yerine ağın ürettiği veriler sonraki adımı belirlemektedir. $f(x)$, hata fonksiyonunu; $f'(x)$, hata fonksiyonunun Newton türevini *ve* α, Karcı kesir dereceli türevin derecesini göstermek üzere KarcıFANN yönteminde kullanılan Karcı kesir dereceli türev Denklem 23 ile ifade edilmektedir [9-13]. Böylece, SGD yönteminde kullanılan Newton türevine, Karcı kesir dereceli türev tanımından gelen $\left(\frac{f(x)}{x}\right)^{\alpha-1}$ çarpanı getirilmektedir. Bu çarpan, KarcıFANN yönteminde öğrenme katsayısı yerine kullanılmakta ve ağa dışarıdan müdahaleyi minimuma indirmektedir.

$$f^{a}(x) = \left(\frac{f(x)}{x}\right)^{\alpha-1} . f'(x) \tag{23}$$

D_{α}^{K} ifadesi, Karcı kesir dereceli türevi temsil etmek üzere Şekil 1 ile gösterilen X_1 nöronuna ait W_{11} ağırlığının toplam hataya (H) etkisini hesaplamak için kesir dereceli türev Denklem 24 ile hesaplanmaktadır.

$$D_{\alpha}^{K}\frac{H}{W_{11}} = \left(\frac{H}{\cancel{Z_C}}\right)^{\alpha-1}.\frac{\partial_H}{\partial_{Z_C}}.\left(\frac{\cancel{Z_C}}{Z}\right)^{\alpha-1}.\frac{\partial_{Z_C}}{\partial_Z}.\left(\frac{Z}{\cancel{Y_{1C}}}\right)^{\alpha-1}.\frac{\partial_Z}{\partial_{Y_{1C}}}.\left(\frac{\cancel{Y_{1C}}}{\cancel{Y_1}}\right)^{\alpha-1}.\frac{\partial_{Y_{1C}}}{\partial_{Y_1}}.\left(\frac{\cancel{Y_1}}{W_{11}}\right)^{\alpha-1}.\frac{\partial_{Y_1}}{\partial_{W_{11}}}$$

$$D_{\alpha}^{K}\frac{H}{W_{11}} = \left(\frac{H}{W_{11}}\right)^{\alpha-1}.\frac{\partial_H}{\partial_{Z_C}}.\frac{\partial_{Z_C}}{\partial_Z}.\frac{\partial_Z}{\partial_{Y_{1C}}}.\frac{\partial_{Y_{1C}}}{\partial_{Y_1}}.\frac{\partial_{Y_1}}{\partial_{W_{11}}} \tag{24}$$

KarcıFANN yönteminde SGD'deki gibi tüm ağırlık değerleri için Denklem 12'ye benzer şekilde gradyan hesaplaması yapılmaktadır. Daha sonra kesir derecesi α olmak üzere, w_{11} ağırlığının herhangi bir t adımında KarcıFANN yöntemi ile güncellenmesi işlemi Denklem 25 ile yapılmaktadır. Aynı işlem veri setindeki tüm ağırlık parametreleri için yapılmaktadır.

$$w_{11t+1} = w_{11t} - D_{\alpha}^{K}.\frac{H}{W_{11t}} \tag{25}$$

Burada dikkat edilmesi gereken önemli bir nokta, KarcıFANN yönteminde kesir dereceli türevin derecesi 1'e yaklaştıkça SGD yöntemine benzer özellikler açığa çıkmasıdır. Yani, Newton türevinin özelliklerini göstermesidir.

Klasik YSA için kullanılan türev Newton türevidir. KarcıFANN ağı için ise, Karcı tarafından tanımı yapılan kesir dereceli türev kullanılmıştır. Şekil 1'de verilen ağ için KarcıFANN ağırlık değişimleri (ΔW) Denklem 26, Denklem 27, Denklem 28, Denklem 29, Denklem 30 ve Denklem 31'de sunulan formüller ile hesaplanmakta ve kırmızı ile gösterilen terim öğrenme katsayısı yerine kullanılmaktadır. Bu sayede Klasik YSA'larda kullanılan öğrenme katsayısına ihtiyaç duyulmamaktadır.

$$D_{\alpha}^{K}W(1,1,1) = \left(\frac{H}{W(1,1,1)}\right)^{\alpha-1}\frac{\partial H}{\partial Y(3,1,2)}.\frac{\partial Y(3,1,2)}{\partial Y(3,1,1)}.\frac{\partial Y(3,1,1)}{\partial Y(2,1,2)}.\frac{\partial Y(2,1,2)}{\partial Y(2,1,1)}.\frac{\partial Y(2,1,1)}{\partial W(1,1,1)} \tag{26}$$

$$D_{\alpha}^{K}W(1,1,2) = \left(\frac{H}{W(1,1,2)}\right)^{\alpha-1}\frac{\partial H}{\partial Y(3,1,2)}.\frac{\partial Y(3,1,2)}{\partial Y(3,1,1)}.\frac{\partial Y(3,1,1)}{\partial Y(2,2,2)}.\frac{\partial Y(2,2,2)}{\partial Y(2,2,1)}.\frac{\partial Y(2,2,1)}{\partial W(1,1,2)} \tag{27}$$

$$D_{\alpha}^{K}W(1,2,1) = \left(\frac{H}{W(1,2,1)}\right)^{\alpha-1}\frac{\partial H}{\partial Y(3,1,2)}.\frac{\partial Y(3,1,2)}{\partial Y(3,1,1)}.\frac{\partial Y(3,1,1)}{\partial Y(2,1,2)}.\frac{\partial Y(2,1,2)}{\partial Y(2,1,1)}.\frac{\partial Y(2,1,1)}{\partial W(1,2,1)} \tag{28}$$

$$D_{\alpha}^{K}W(1,2,2) = \left(\frac{H}{W(1,2,2)}\right)^{\alpha-1}\frac{\partial H}{\partial Y(3,1,2)}.\frac{\partial Y(3,1,2)}{\partial Y(3,1,1)}.\frac{\partial Y(3,1,1)}{\partial Y(2,2,2)}.\frac{\partial Y(2,2,2)}{\partial Y(2,2,1)}.\frac{\partial Y(2,2,1)}{\partial W(1,2,2)} \tag{29}$$

$$D_{\alpha}^{K}W(2,1,1) = \left(\frac{H}{W(2,1,1)}\right)^{\alpha-1}\frac{\partial H}{\partial Y(3,1,2)}.\frac{\partial Y(3,1,2)}{\partial Y(3,1,1)}.\frac{\partial Y(3,1,1)}{\partial W(2,1,1)} \tag{30}$$

$$D_{\alpha}^{K}W(2,2,1)=\left(\frac{H}{W(2,2,1)}\right)^{\alpha-1}\frac{\partial H}{\partial Y(3,1,2)}.\frac{\partial Y(3,1,2)}{\partial Y(3,1,1)}.\frac{\partial Y(3,1,1)}{\partial W(2,2,1)} \tag{31}$$

Şekil 1 ile gösterilen ağ için Sigmoid aktivasyon fonksiyonun (Sig) kullanılması durumunda ağırlık değişimleri (ΔW) Denklem 32, Denklem 33, Denklem 34, Denklem 35, Denklem 36 ve Denklem 37'de sunulan formüller ile hesaplanmaktadır.

$$\Delta W(1,1,1)=\left(\frac{H}{W(1,1,1)}\right)^{\alpha-1}\{-2(R-Y(3,1,2))Sig(Y(3,1,1))[1-Sig(Y(3,1,1))]W(1,1,1)Sig(Y(2,1,1))\,[1-Sig(Y(2,1,1))]X_1\} \tag{32}$$

$$\Delta W(1,1,2)=\left(\frac{H}{W(1,1,2)}\right)^{\alpha-1}\{-2(R-Y(3,1,2))Sig(Y(3,1,1))[1-Sig(Y(3,1,1))]W(1,1,2)Sig(Y(2,2,1))[1-Sig(Y(2,2,1))]X_1\} \tag{33}$$

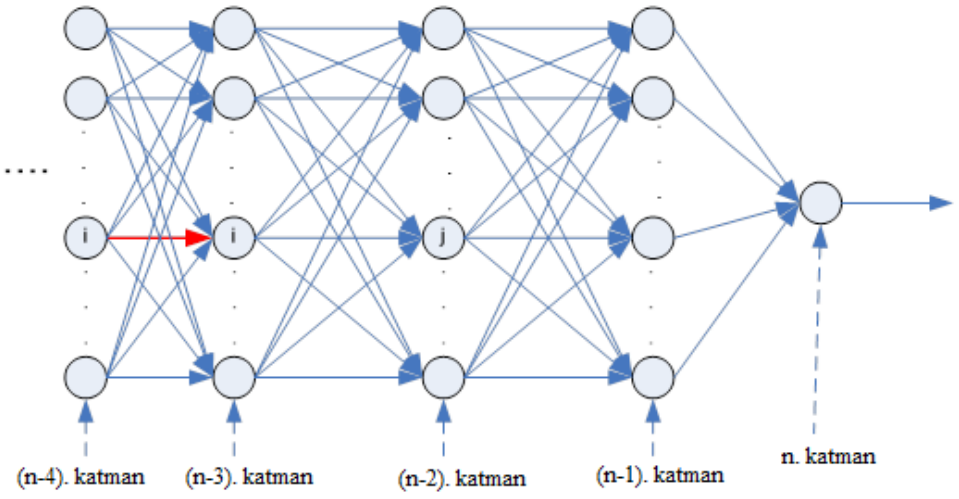
$$\Delta W(1,2,1)=\left(\frac{H}{W(1,2,1)}\right)^{\alpha-1}\{-2(R-Y(3,1,2))Sig(Y(3,1,1))[1-Sig(Y(3,1,1))]W(1,2,1)Sig(Y(2,1,1))[1-Sig(Y(2,1,1))]X_2\} \tag{34}$$

$$\Delta W(1,2,2)=\left(\frac{H}{W(1,2,2)}\right)^{\alpha-1}\{-2(R-Y(3,1,2))Sig(Y(3,1,1))[1-Sig(Y(3,1,1))]W(1,2,2)Sig(Y(2,2,1))[1-Sig(Y(2,2,1))]X_2\} \tag{35}$$

$$\Delta W(2,1,1)=\left(\frac{H}{W(2,1,1)}\right)^{\alpha-1}\{-2(R-Y(3,1,2))Sig(Y(3,1,1))[1-Sig(Y(3,1,1))]Y(2,1,2)\} \tag{36}$$

$$\Delta W(2,2,1)=\left(\frac{H}{W(2,2,1)}\right)^{\alpha-1}\{-2(R-Y(3,1,2))Sig(Y(3,1,1))[1-Sig(Y(3,1,1))]Y(2,2,2)\} \tag{37}$$

Şekil 2'de verilen ağda ağırlıklar $W(k,i,j)$ şeklinde tanımlanmış olup k ağın seviyesini, i okun çıkış yaptığı düğümü ve j ise okun vardığı düğümü göstermektedir. Düğüm fonksiyonları ise $Y(k,i,1)$ şeklinde olup düğüm fonksiyonun aktivasyon fonksiyonundan geçirilmiş hali $Y(k,i,2)=f(Y(k,i,1))$ şeklindedir ve $f()$ aktivasyon fonksiyonunu göstermektedir. Her seviyedeki düğüm sayısı k_i ve katman sayısı n olarak ifade edilmektedir.



Şekil 2. KarcıFANN ağının son dört seviyesi.

k_n=1 için kırmızı okun ağırlık değişimi Denklem 38 ile gösterildiği gibi hesaplanmaktadır.

$$D_{\alpha}^{K}W(n-4,i,j)=\left(\frac{H}{W(n-4,i,j)}\right)^{\alpha-1}\frac{\partial H}{\partial Y(n,1,2)}.\frac{\partial Y(n,1,2)}{\partial Y(n,1,1)}.$$

$$\left\{\sum_{m=1}^{k_{n-1}} \frac{\partial Y(n,1,1)}{\partial Y(n-1,m,2)} \cdot \frac{\partial Y(n-1,m,2)}{\partial Y(n-1,m,1)} \left(\begin{array}{c} \sum_{r=1}^{k_{n-2}} \frac{\partial Y(n-1,m,1)}{\partial Y(n-2,r,2)} \cdot \frac{\partial Y(n-2,r,2)}{\partial Y(n-2,r,1)} \cdot \\ \left(\sum_{p=1}^{k_{n-3}} \frac{\partial Y(n-2,r,1)}{\partial Y(n-3,p,2)} \cdot \frac{\partial Y(n-3,p,2)}{\partial Y(n-3,p,1)}\right) \end{array} \right)\right\} \cdot \frac{\partial Y(n-3,r,1)}{\partial W(n-4,i,j)} \quad (38)$$

Bu durum geriye kalan bütün seviye ve ağırlıklar için uygulanabilir.

6.Sonuçlar

Bu çalışmada, KarcıFANN, GD, SGD, AdaGrad, RMSProp ve momentumlu SGD yöntemlerinin matematiksel gösterimleri sunulmaktadır.

Öğrenme katsayısı olarak tamsayıların kullanıldığı ve klasik Newton türevini içeren modellerde salınım, öğrenememe, ezberleme, gradyanların kaybolması ve gradyan patlaması gibi problemlerle karşılaşılmaktadır. Bu problemleri çözmek amacıyla önerilen KarcıFANN yönteminde, öğrenme katsayısının yerine kullanılan kesir dereceli türevin modele dışarıdan müdaheleyi en aza indirerek modelin kendi verileriyle işlem yapması sağlanmaktadır.

KarcıFANN yönteminin, özellikle doğrusal olmayan problemlerin çözümünde başarılı sonuçlar üreteceği düşünülmektedir. Ayrıca, KarcıFANN yönteminde türevin derecesi 1'e yaklaştıkça klasik türeve benzer özellikler açığa çıkmaktadır.

Kaynaklar

[1] Newton, I. Philosophiae Naturalis Principia Mathematica, 1687.

[2] Zinkevich, M.A., Weimer, M., Smola, A., & Li, L. Parallelized Stochastic Gradient Descent. Neural Information Processing Systems, 2010.

[3] Dogo, E.M., Afolabi, O.J., Nwulu, N.I., Twala, B., Aigbavboa, C.O. A Comparative Analysis of Gradient Descent-Based Optimization Algorithms on Convolutional Neural Networks. 2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS), 92-99.

[4] Karakurt, M., Oymak, E.A., Hark, H., Erdoğan, M.C. ve Karcı, A. Karcı Sinir Ağlarının Uygulaması ve Performans Analizi, Journal of Computer Science, Vol:7, 68-80, 2022.

[5] Chen, X., Liang, C., Huang, D., Real, E., Wang, K., Liu, Y., Pham, H., Dong, X., Luong, T., Hsieh, C., Lu, Y., & Le, Q. Symbolic Discovery of Optimization Algorithms, 2023, ArXiv, abs/2302.06675.

[6] Tian Y, Zhang Y, Zhang H. Recent Advances in Stochastic Gradient Descent in Deep Learning. Mathematics, 2023; 11(3):682. https://doi.org/10.3390/math11030682.

[7] Karakurt, M., Saygılı, H. ve Karcı, A. KarcıFANN Yönteminde Aktivasyon Fonksiyonlarının Karşılaştırılması, 8th International Artificial Intelligence and Data Processing Symposium (IDAP2024), IEEE, 2024, doi: 10.1109/IDAP64064.2024.10711149 . https://ieeexplore.ieee.org/document/10711149.

[8] Saygılı, H., Karakurt, M. ve Karcı, A. KarcıFANN Yönteminde Kayıp Fonksiyonlarının Karşılaştırılması, 8th International Artificial Intelligence and Data Processing Symposium (IDAP2024), IEEE, 2024, doi: 10.1109/IDAP64064.2024.10710967. https://ieeexplore.ieee.org/document/10710967.

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

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

[11] 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.

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

[13] Karcı, A. Properties of Karcı's Fractional Order Derivative, Universal Journal of Engineering Science, Vol:7, pp:32-38, 2019.

[14] Karci, A. Fractional Order Integration: A New Perspective based on Karcı's Fractional Order Derivative. Computer Science, 2021, 6(2), 102-105.

[15] Karakurt, M., Saygılı, H. & Karcı, A. (2025). 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), 248-263. doi: 10.55730/1300-0632.4125.

[16] Saygılı, H., Karakurt, M. & Karcı, A. (2025). Karcı kesir dereceli yapay sinir ağı (KarcıFANN): öğrenme oranı, aşırı uyum ve yetersiz uyum sorunlarını çözme. Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi.

[17] Karakurt, M. (2025). YSA'larda Kesir Dereceli Türev Kullanımının Öğrenmeye Etkisi. Computer Science, 10(2), 153-179. https://doi.org/10.53070/bbd.1808790.

[18] Saygılı, H., Karakurt, M., & Karcı, A. (2025). KarcıFANN Yönteminin Yakınsama Kabiliyetinin Analiz Edilmesi. Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 37(2), 897-907. https://doi.org/10.35234/fumbd.1661969

[19] Reddi, S.J., Hefny, A.S., Sra, S., Póczos, B., & Smola, A., On Variance Reduction in Stochastic Gradient Descent and its Asynchronous Variants. 2015, ArXiv, abs/1506.06840.
[20] Lemaréchal, C. Cauchy and the Gradient Method . Doc Math Extra: 251–254, 2012, Archived from the original (PDF) on 2018-12-29.
[21] Rumelhart, D., Hinton, G. & Williams, R. Learning representations by back-propagating errors. Nature 323, 533–536, 1986, https://doi.org/10.1038/323533a0.
[22] Le Cun, Y. Learning Process in an Asymmetric Threshold Network. In: Bienenstock, E., Soulié, F.F., Weisbuch, G. (eds) Disordered Systems and Biological Organization. NATO ASI Series, vol 20. Springer, Berlin, Heidelberg, 1986, https://doi.org/10.1007/978-3-642-82657-3_24
[23] Bottou, L. Stochastic gradient learning in neural networks. Proceedings of Neuro-Nımes, 1991, 91(8), 12.
[24] Duchi, J.C., Hazan, E., Singer, Y., Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. J. Mach. Learn. Res., 2011, 12, 2121-2159.
[25] Seyyarer, E., Ayata, F., Uçkan, T., Karci, A., Derin Öğrenmede Kullanılan Optimizasyon Algoritmalarının Uygulanması Ve Kıyaslanması. Computer Science, 2020, 5(2), 90-98.
[26] Fu, J., Wang, B., Zhang, H., Zhang, Z., Chen, W., & Zheng, N. When and why momentum accelerates sgd: An empirical study, 2023, arXiv preprint arXiv:2306.09000.
[27] Lee, T., Gasswint, G. and Henning E., Momentum. Cornell University Computational Optimization Open Textbook. SYSEN5800. Cornell University, 2021, https://optimization.cbe.cornell.edu/index.php?title=Momentum.
[28] Yazan, E., Talu, M. F., Comparison of the stochastic gradient descent based optimization techniques, International Artificial Intelligence and Data Processing Symposium (IDAP), Malatya, Turkey, 2017, pp. 1-5, doi: 10.1109/IDAP.2017.8090299.
[29] Rakshitha Kiran P., Naveen N. C. Op-RMSprop (Optimized-Root Mean Square Propagation) Classification for Prediction of Polycystic Ovary Syndrome (PCOS) using Hybrid Machine Learning Technique, International Journal of Advanced Computer Science and Applications (IJACSA), 2022, 13(6). http://dx.doi.org/10.14569/IJACSA.2022.0130671
[30] Caputo, M., Linear models of dissipation whose Q is almost frequency independent—II. Geophysical journal international, 1967, 13(5), 529-539.
[31] Ross, B., The development of fractional calculus 1695–1900. Historia mathematica, 1977, 4(1), 75-89.
[32] Haq, A., Partial-approximate controllability of semi-linear systems involving two Riemann-Liouville fractional derivatives. Chaos, Solitons & Fractals, 2022, 157, 111923.
[33] Rahman, G. U., Ahmad, D., Gómez-Aguilar, J. F., Agarwal, R. P., & Ali, A., Study of Caputo fractional derivative and Riemann–Liouville integral with different orders and its application in multi-term differential equations. Mathematical Methods in the Applied Sciences, 2025, 48(2), 1464-1502.
[34] Tarasov, V. E., On chain rule for fractional derivatives. Communications in Nonlinear Science and Numerical Simulation, Elsevier. 2016, 30(1-3), 1-4. https://doi.org/10.1016/j.cnsns.2015.06.007
[35] Huseynov, I. T., Ahmadova, A., & Mahmudov, N. I., Fractional Leibniz integral rules for Riemann-Liouville and Caputo fractional derivatives and their applications. 2020, arXiv preprint arXiv:2012.11360.