Yapay Zeka Sunum Raporu

SAKARYA ÜNİVERSİTESİ
YAPAY ZEKA DERSİ
1.ÖĞRETİM B GRUBU

GRUP ÜYELERİ:

SAMET YAZICI
KÜBRA SALTIK
AHMET KAĞAN YÖRÜK
ALPEREN ÖRSDEMİR
MUSTAFA HAJI IBRAHIM

Parçacık sürü optimizasyonu

Optimizasyon tanımı:

Optimizasyon karşılaşılan problemlere kısıtlamalar dahilinde mümkün olan en iyi sonucu sunabilme olarak ifade edilebilir.

Örnek vererek daha anlaşılır olmasını sağlayalım:

Bir yatırımcı olduğunuzu ve paranızı yatırabileceğiniz 25 farklı seçeneğinizin olduğunu varsayalım. İçlerinden üçünü seçmek istiyorsunuz. Böyle bir durum için toplam 2300 seçeneğiniz olur. Bir yatırımcı olarak en fazla kar elde etmek istediğinizi de hesaba katarsak tüm bu seçenekleri tek tek denemenin kardan çok uzak olduğunu söylemek hiç de zor olmayacaktır. Kaldı ki piyasa durumu farklı hesaplamaları da beraberinde getirebilir. Bu noktada ise optimizasyon devreye girerek mümkün olan en karlı yatırımı yapmanıza yardımcı olur.

Parçacık sürü optimizasyonu özet:

Doğa her zaman bilim için ilham kaynağı olmuştur. Aktarmaya çalıştığımız parçacık sürü optimizasyon konusu da bu duruma örnektir. Dr. Kennedy ve Dr. Eberhart'ın , bazı hayvanların; yiyecek bulmak, korunmak gibi temel ihtiyaçlarını karşılamak için beraber hareket ederek her bireyin ortak fayda sağlamasına katkıda bulunan sürü hareketlerini gözlemleyip bu davranış biçimini bilgisayar bilimine uygulamaları sonucu parçacık sürü optimizasyonu adı verilen algoritmayı oluşturulmuşlardır.

Peki kuş veya balık gibi hayvanların sürü hareketleri bize optimizasyon konusunda ne anlatıyor olabilir?

Optimizasyonun tanımından da hatırlanacağı gibi kısıtlamalar dahilinde mümkün olan en iyi çözüme ulaşabilmeyi hedef olarak gösterebiliriz. Bazı balık ve kuş türlerinde gözlemlenen sürü hareketlerinde sürü içerisindeki hayvanların problem karşısında iletişim kurarak ve de birbirlerinin konumlarını takip ederek çözüme ulaşmaları güzel bir optimizasyon örneğidir. Buradaki problemi bir balık sürüsünün avcıdan kaçması olarak kabul edersek; kısıtlamayı süre ve çözümü sürüde avcıdan en uzak konumdaki balığın konumuna ulaşmak olarak örnekleyebiliriz.

PSO çalışma prensibi:

Şekil 1 den PSO algoritmasını görebilirsiniz. Bu algoritmanın adımları şu şekildedir;

1.Sürünün başlangıç konumu ve parametrelerine rastgele değerler atanır.

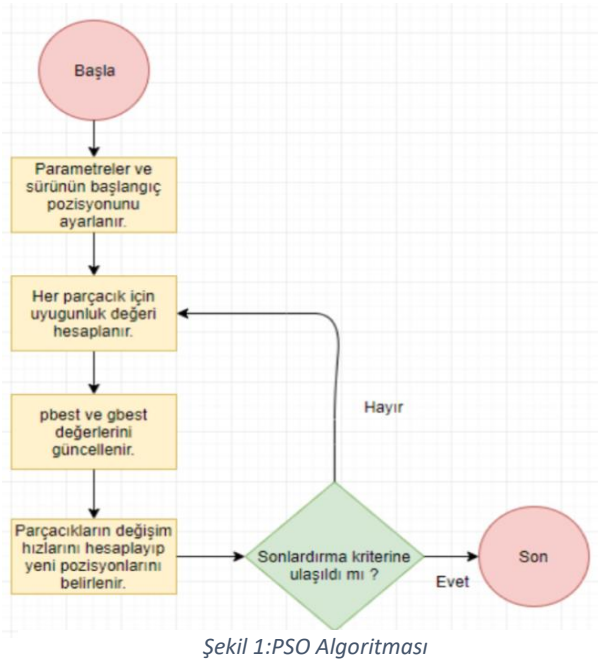
2.Bütün parçacıkların uygunluk fonksiyonuna göre uygunluk değerleri hesaplanır.

3.Her bir parçacığın pbest değeri bulunur.

4.Sürünün geneline bakılarak gbest değeri bulunur.

5.Sürünün konumu ve parametreleri tekrar hesaplanır.

6.İstenilen değer bulunduysa durulur bulunmadı ise 2.adımdan devam edilir.



Şekil 1:PSO Algoritması

Fonksiyon tanımı:

Sürüdeki herhangi bir parçacığın yeni hız değerini hesaplamak için şu fonksiyon kullanılır.

$$v_{i+1} = wv_i + c_1(rand_1)(\rho best - x) + c_2(rand_2)(gbest - x)$$

Bu fonksiyondaki değişkenler şöyle ifade edilebilir.

$\boldsymbol{v}$: Parçacığın hızı.

$\boldsymbol{w}$: Parçacığın şu an ki hızının yeni hıza etkisini belirten değer 0.4 ile 1.4 arasında değer alabilir.

$\boldsymbol{x}$: Parçacık değeri.

$\boldsymbol{rand_1}$ **Ve** $\boldsymbol{rand_2}$**:** Rastgele üretilen değerler.

$\boldsymbol{c_1}$: Parçacığın pbest değerinin yeni hızı ne kadar etkileyeceğini belirten katsayı 1.5 ile 2 arasında değer alabilir.

$\boldsymbol{c_2}$: Parçacığın gbest değerinin yeni hıza ne kadar etki edeceğini belirten katsayı 2 ile 2.5 arasında değer alabilir.

Burada c1 ve c2 katsayıları öğrenme faktörleridir ve hızlanma katsayıları olarak da adlandırılır.

• Bu katsayılar, bir iterasyonda bir parçacığın alabileceği adımın maksimum boyutunu etkiler ve her parçacığı kişisel en iyi veya global en iyi pozisyonlarına doğru çeken, hızlanmayı ifade eder.

• Düşük değerlerin seçilmesi parçacıkların hedef bölgeye doğru çekilmeden önce, bu bölgeden uzak yerlerde dolaşmalarına imkân verir.

• Ancak hedefe ulaşma süresi uzayabilir.

• Diğer yandan, yüksek değerlerin seçilmesi, hedefe ulaşmayı hızlandırırken, beklenmedik hareketlerin oluşmasına ve hedef bölgeye ulaşılmamasına sebep olabilir.

$\boldsymbol{pbest}$: Parçacığın çözüme en çok yaklaştığı durum.

$\boldsymbol{gbest}$: Tüm parçacıklar arasında çözüme en çok yaklaşılan durum.

Örnek uygulama:

$$f(x) = x^2 + 2x - 3$$

Örnek problemimiz verilen denklemin sonucunu 0 yapacak **x** değerini bulmak olsun. Burada denklemi bir işin maliyet fonksiyonu olarak düşünebilirsiniz.

- ***Öncelikle kaç adet parçacık ile çözümü arayacağımızı belirliyoruz.***

Bu sayı arama uzayının genişliğine ve sizin isteğinize bağlı olarak belirlenir. Biz bu örneğimizde 3 belirleyelim.

Parçacıklar rastgele belirlenir. Parçacıkları aşağıda verildiği gibi sırasıyla ***3, 7, 5*** olarak belirledik.

$$P_1 = 3$$
$$P_2 = 7$$
$$P_3 = 5$$

- ***Her parçacığın uygunluk değeri hesaplanır.***

Bu örnekte amacımız problemin denklemini 0'a yaklaştırmak olduğundan uygunluk fonksiyonumuz problemin denkleminin ta kendisidir.

$$f(3) = 12$$
$$f(7) = 60$$
$$f(5) = 32$$

Parçacıkların Uygunluk Değerleri

Görüldüğü üzere belirlediğimiz x değerlerini denklemde yerine koyarak uygunluk değerlerini kolayca elde ettik. Tekrar hatırlatmak gerekirse; amacımız uygunluk değeri 0 olan x değerini bulmaktı.

- ***Pbest ve gbest değerleri hesaplanır.***

Şu an ilk iterasyonda olduğumuzdan parçacıkların kendileri zaten ***pbest***'leridir. ***Gbest*** ise 0'a en yakın olan ***P1*** parçacığıdır.

- ***C1, c2 değerleri ve rastgele olarak rand1, rand2 değerleri belirlenir.***

C1 ve c2 değerleri genellikle 2 belirlenir ben de bu parametreleri 2 olarak belirliyorum.

Rand1 ve ***rand2*** değerini de hesaplama kolaylığı açısından 2 belirliyorum.

Ayrıca daha ilk iterasyonda olduğumuzdan parçacıklar herhangi bir hıza sahip değildir. Bu nedenle ilk değişim hızı ***V0***'ı 0 kabul ediyoruz.

Nihayet sıra değişim fonksiyonumuzu kullanarak parçacıkların değişimini hesaplamaya geldi 😃

- ***Parçacıkların değişim hızları hesaplanır.***

$P_1 \rightarrow 0 + 2 * 2 * (3 - 3) + 2 * 2 * (3 - 3) = 0$

$P_2 \rightarrow 0 + 2 * 2 * (7 - 7) + 2 * 2 * (3 - 7) = -16$

$P_3 \rightarrow 0 + 2 * 2 * (5 - 5) + 2 * 2 * (3 - 5) = -8$

Parçacıkların Değişim Hızının Hesaplanması

Böylece formülümüzü kullanarak parçacıkların değişimi hesaplanmış oldu.

- ***Parçacıkların yeni değerleri belirlenir.***

Bu aşamada parçacıklar, değişim değerleri ile toplanarak yeni parçacıklar belirlenir.

$$P_1 \rightarrow 3 + 0 = 3$$
$$P_2 \rightarrow 7 - 16 = -9$$
$$P_3 \rightarrow 5 - 8 = -3$$

Yeni Parçacıkların Belirlenmesi

Şimdiki görevimiz bu parçacıklar ile çözüme ne kadar yaklaştığımızı değerlendirmek.

- ***Yeni parçacıkların uygunluk değerleri bulunur.***

$$f(3) = 12$$
$$f(-9) = 34$$
$$f(-3) = 0$$

Yeni Parçacıkların Uygunluk Değerinin Hesaplanması

Tebrikler. Daha ilk iterasyonda bırakın problemin çözümüne yaklaşmayı çözdünüz bile

Denklemimizi 0 yapan değer ***-3*** olarak bulunmuş oldu.

Eğer çözüme ulaşamamış olsaydık; yeni parçacıklar da göz ününe alınarak yeniden ***pbest*** değeri ve bu zamana kadar gelmiş tüm ***pbest'***lerin en iyisi olan ***gbest*** değeri belirlenecekti. Ek olarak **rand1** ve ***rand2*** değerleri tekrar belirlenip parçacıkların değişimi hesaplanacak ve yeni parçacıklar bulunacaktı. Ve bir kez daha uygunluk değerini bulup çözüme ne kadar yaklaştığımızı değerlendirecektik.

PSO Algoritmalarının kötü yönleri

PSO Algoritmaları tek çözümü olan problemleri çözmede çok iyidirler fakat birden fazla çözüme veya çözüm uzayına sahip problemler için aynı şey söylenemez. Bunun sebebi de şu şekilde izah edilebilir; parçacıkların yeni değerleri hesaplanırken kullanılan uygunluk fonksiyonundan ötürü parçacıklar gbest değerine yaklaşmaya çalışmaktadır. Bu demektir ki diğer parçacıklar gbest değerini belirleyen parçacığın yakın olduğu çözüme doğru ilerlerler bu sebeple diğer çözümleri kaçırırlar. Bu yaklaşımı c2 parametresi ile belirleyebiliriz.

PSO Algoritmalarının iyi yönleri

PSO Algoritmalarında parçacıkların yeni değerlerini hesaplamada kullanılan uygunluk fonksiyonu radyan hesap ya da diferansiyel denklem gibi karmaşık işlemler içermediğinden kolay uygulanabilirdir bu da bu algoritmayı hızla yapan etmenlerden biridir. Algoritmanın hızlı olma sebeplerinden biri de belirlenmesi gereken az sayıda parametre olmasıdır. Her iterasyonda pbest ve gbest değerleri güncellenir bunun haricinde güncellenmesi gereken ya da değiştirilmesi gereken başka bir parametre yoktur.

PSO Algoritmalarının kullanıldığı yerler

- Fonksiyon optimizasyonu
- Çizelgeleme
- Yapay Sinir Ağlarının Eğitimi
- Bulanık Mantık Sistemleri
- Görüntü İşleme
- Anten Dizayn
- Radyo İstasyonu konumlandırılması.

Gibi non-lineer denklem içeren birçok alanda kullanılan bir algoritmadır.

Kaynakça:

- https://medium.com/deep-learning-turkiye/par%C3%A7ac%C4%B1k-s%C3%BCr%C3%BC-optimizasyonu-pso-1402d4fe924a
- https://www.muhendisbeyinler.net/parcacik-suru-optimizasyonu/
- https://medium.com/@hamzaerguder/par%C3%A7ac%C4%B1k-s%C3%BCr%C3%BC-optimizasyonu-24e01beec438
- http://acikerisimarsiv.selcuk.edu.tr:8080/xmlui/bitstream/handle/123456789/7732/237263.pdf?sequence=1&isAllowed=y