Araştırma Makalesi / Research Article

Bilgi Erişimi için Eşli bir Sıralama Algoritması

Engin TAŞ*

Afyon Kocatepe Üniversitesi, İstatistik Bölümü, Afyonkarahisar

Öz
Yapay öğrenmede temel problemlerden biri, ilgilenilen birimler arasındaki tercih ilişkilerinin belirlenmesidir. Bu kapsamda sıralama, verilen bir tercih ilişkisine göre birimleri düzenleme yeteneğine sahip bir fonksiyonu öğrenmek olarak tanımlanabilir. Bu tip problemler genellikle örneklerin çiftler olduğu sınıflandırma problemi olarak ele alınır. Bu çalışmada ise genel sıralamanın bir tahmini için eşli karşılaştırmalara dayanan bir yaklaşım sunulmuştur. Eşli sıralama hatasını minimize eden bu sıralama problemi, bir doğrusal eşitlikler sistemi ile temsil edilmiştir. Bu doğrusal eşitlik sisteminin çözülmesiyle sıralama fonksiyonlarının öğrenilmesi için gradyan düşümü algoritmasının geliştirilmiş bir versiyonu önerilmektedir. Ayrıca, oluşturulan sıralama modelinin genelleştirme performansını kontrol edebilmek için Tikhonov düzeltmesi de bu çalışma kapsamında kullanılmıştır. Geliştirilen hızlı gradyan düşümü algoritması, düzeltme seviyesinden bağımsız olarak çok kısa bir sürede çözüme yakınsamıştır.

Anahtar kelimeler: Sıralama, Cezalı En Küçük Kareler, Gradyan Düşümü, Bilgi Erişimi, Arama Motoru.

A Pairwise Ranking Algorithm for Information Retrieval

Abstract
One of the main problems in machine learning is the determination of preference relations between interested units. In this context, ranking can be defined as learning a function with the ability to organize units according to a given preference relation. This type of problem is often treated as a classification problem where the examples are formed by pairs. In this study, an approach based on pairwise comparisons is presented for an estimation of a general ordering. This ranking problem that minimizes the pairwise ranking error is represented by a system of linear equations. An improved version of the gradient-descent algorithm is proposed to learn the ranking functions for solving this system of linear equations. In addition, Tikhonov regularization is also used to control the generalization performance of the ranking model. The developed rapid gradient descent algorithm converges to the solution in a very short time, regardless of the regularization level.

Keywords: Ranking, Regularized Least Squares, Gradient Descent, Information Retrieval, Search Engine.

1. Giriş

İnternette arama motorları ve çeşitli alanlarda belirli öneri veren sistemlerin son yıllarda çok popüler hale gelmesi, bu alanlarda kullanılan sıralama yordamlarını önemli bir konuma getirmiştir. Bu alanlarda eşi benzeri görülmemiş büyüklükte verinin bulunması, sıralama için kullanılan yapay öğrenme algoritmalarının ölçeklenebilirliğini olmazsa olmaz bir özellik haline gelmiştir. Sıralama problemini modellemeye yönelik popüler bir yaklaşım eşli tercihleri dikkate almaktır. Bu kapsamda, tahmin edilen fayda skorlarına göre bir örnekler kümesi sıralanırken amaç oluşturulan sıralamadaki eşli yanlış sınıflandırma sayısını minimize etmektir.

Caruana vd. [1] RankProp olarak adlandırılan bir sinir ağı sıralama modelini önermiştir. RankProp modeli iki aşamada gerçekleşmektedir: Mevcut hedef değerleri üzerinde bir hata kareler ortalaması (HKO) regresyonunun oluşturulması ve ağ tarafından verilen mevcut sıralamayı yansıtmak için hedef değerlerinin kendi aralarında düzenlenmesidir. Son çıktı, orijinal, ölçülmüş sıra değerinden

*Sorumlu yazar: engintas@aku.edu.tr
Geliş Tarihi: 08.06.2018, Kabul Tarihi: 25.10.2018.

daha iyi çalışan, hak edilen sırayı belirten çok sayıda hedef için veriye bir harita sunar. RankProp modelinin avantajı; çiftlerin yerine bireysel örnekler üzerinde eğitim yapmasıdır. Ancak yakınsama koşulları bilinmemektedir ve olasılıksal bir model vermemektedir.

Herbrich vd. [2] sıralama için öğrenme problemini sıradan regresyon ile çözmeye, yani bir girdi vektörünü sayısal sıraların, sıralı bir kümesine yönlendiren öğrenme üzerinde çalışmışlardır. Sıraları gerçek doğru üzerinde aralıklı olarak modellemişlerdir ve örnek çiftleri ve hedef sıralarına bağımlı olan kayıp fonksiyonunu göz önüne almışlardır. Sıraların sınırlarının konumu, son sıralama fonksiyonu üzerinde önemli bir role sahiptir. Crammer ve Singer [3] de problemi benzer bir şekilde ele almış ve algılayıcılara dayalı bir sıralayıcı (PRank) önermişlerdir. PRank bir anda bir örneği kullanmayı öğrenir ki bu durumda eş-tabanlı öğrenmede m örnek yerine $^{*}O(m^2)$ tane eş kullanıldığı için PRank yöntemi eş-tabanlı öğrenmeye göre daha avantajlıdır. Doğrusal versiyonu çevrimiçi bir algoritma olmasına rağmen PRank yöntemi birçok sıralama algoritmasıyla karşılaştırılmıştır ve Herbrich vd. [2] tarafından tanımlanan karesel çekirdek versiyonunun tüm bu tip algoritmalardan daha üstün olduğu bulunmuştur. Sıralama problemlerinde bir başka yaklaşım ise problemi ikili bir sınıflandırma problemi haline dönüştürmektir [4]. Bu durumda karşılaşılan temel sorun hem örnek sayısının çok artması hem de veri uzayı boyutunun çok büyük olmasıdır. Bu gibi durumlarda veri uzayı boyutunun indirgenmesine yönelik özellik seçiminde sezgisel arama algoritmaları kullanılabilir [5]. Sezgisel arama yaklaşımlarının dışında ikili örüntüleri kullanarak örneklere ilişkin daha etkin özellikler çıkaran yeni yaklaşımlarda bulunmaktadır [6].

Harrington [7], PRank modelinin ortalamasını alarak yaklaşık olarak Bayes noktasını bulan Prank modelinin basit ama etkin bir genişlemesini öne sürmüştür. Dekel vd. [8], A' dan B ye bir bağlantının A'nın sırasının B'den büyük olduğu anlamına geldiği, döndürülmüş grafikleri kullanan sıralama için oldukça geniş çerçeveli bir çalışma sunmuşlardır.

Freund vd. [9], RankBoost adı verilen bir sıralama algoritmasını önermişlerdir. Bu yöntem, çiftler üzerinde eğitilen ve sıradan regresyon probleminin çözümü yerine öğrenme problemini doğrudan ele alan bir yöntemdir. Bu yöntemin gradyan düşümü yöntemlere benzerliğinden esinlenen bir başka çalışmada ise RankNet algoritmasını derin öğrenme ile birleştiren bir yaklaşım kişiselleştirilmiş arama için kullanılmıştır [10].

Bu çalışmada, büyük ölçekli veri kümeleri için eşli sıralama hatasını en küçük kareler yaklaşımıyla optimize etmeye dayanan bir gradyan düşümü algoritması önerilmiştir. Önerilen bu yaklaşımın esas özelliği, çözümünün doğrusal bir denklemler sistemi ile ifade edilebilmesidir. Bunun sonucu olarak, oldukça etkin olan ve basit matris cebirine dayanan bir gradyan düşümü algoritması geliştirilmiştir. Bu yaklaşım cezalı en küçük kareler metodunun farklı bir düzenlenmesi olarak da düşünülebilir [11].

2. Materyal ve Metot

$\boldsymbol{Z} = \big((\boldsymbol{x_1}, \boldsymbol{y_1}), (\boldsymbol{x_2}, \boldsymbol{y_2}), \dots, (\boldsymbol{x_m}, \boldsymbol{y_m})\big) \in \boldsymbol{Z^m}$ serisi, $\boldsymbol{Z} = \boldsymbol{R^n} \times \boldsymbol{R}$ şeklinde bir örneklem uzayı üzerinde tanımlanmış bir olasılık dağılımına göre çekilsin. $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{y}) \in \boldsymbol{Z}$ şeklinde bir örnek, gerçel-değerli özelliklerden oluşan $\boldsymbol{n}$-boyutlu bir sütun vektörü ve karşılık gelen gerçel-değerli bir fayda skorundan oluşur. $\boldsymbol{X} \in \boldsymbol{R^{n \times m}}$, sütunları eğitim örneklerinin özellik temsillerini içeren $\boldsymbol{n} \times \boldsymbol{m}$ boyutlu girdi matrisi ve $\boldsymbol{y} \in \boldsymbol{R^m}$ eğitim kümesindeki fayda skorlarını içeren bir sütun vektörüdür.

Amaç, veriden bir sıralama fonksiyonu olan $\boldsymbol{f}: \boldsymbol{R^n} \rightarrow \boldsymbol{R}$ 'yi öğrenmektir. Doğrusal durumda bu tip bir sıralama fonksiyonu $\boldsymbol{f(x)} = \boldsymbol{x^T w}$ şeklinde ifade edilebilir, burada $\boldsymbol{w} \in \boldsymbol{R^n}$ optimize edilecek olan parametre vektörüdür. Başka bir ifadeyle, $\boldsymbol{x_j}$ örneğine göre daha tercih edilen bir $\boldsymbol{x_i}$ örneği verildiğinde yani $\boldsymbol{y_i} > \boldsymbol{y_j}$ ise, sıralama fonksiyonundan istenilen $\boldsymbol{f(x_i)} > \boldsymbol{f(x_j)}$ olmasıdır. Sıralamanın değerlendirilmesi açısından buradaki amaç basit doğrusal regresyondan oldukça farklıdır, çünkü ilgilenilen esas nokta sıralama fonksiyonunun aldığı değerlerden ziyade yeni bir örnekler kümesi sıralama fonksiyonundan elde edilen skorlar kullanılarak sıralandığında, bu sıralamanın gerçek sıralamayla ne kadar iyi örtüştüğüdür. Bu kriter, esas amacın, web sayfalarının veya ürünlerin uygun bir sırasının belirlenmesi olduğu internet arama motorları veya son kullanıcıya tavsiyede bulunan sistemler için makul bir kriterdir. Bu kriteri modellemenin bir yolu bir sıralama isteği ile tetiklenen eşli tercihleri dikkate almaktır [12].

Bir sıralama fonksiyonunun performansı aşağıdaki gibi tanımlanan eşli sıralama hatası ile ölçülebilir.

$$\frac{1}{N}\sum_{y_i<y_j} H\left(f(x_j)-f(x_i)\right), \tag{1}$$

burada H

$$H(a)=\begin{cases}1, & a>0\\ 1/2, & a=0\\ 0, & a<0\end{cases}$$

şeklinde tanımlanan bir adım fonksiyonudur. N, $y_i<y_j$ koşulunu sağlayan eşlerin sayısıdır. Eş. 1 basit olarak gerçek sıralama ile f tarafından üretilen sıralama arasında farklı sıralanmış çiftleri yani yanlış sıralanmış çiftleri sayar. Tahminleri rassal olarak gerçekleştiren, veya tüm örnekler için aynı tahmini veren bayağı tahmin ediciler için, oluşan hata 0.5 olur. Eş. 1'de verilen hatayı doğrudan minimize etmek hesapsal olarak takip edilebilir değildir, eşli bir kritere göre sıralamayı öğrenen başarılı yaklaşımlar tipik olarak bu fonksiyonların konveks türlerini minimize eder. Bu çalışmada bu problemi matematiksel olarak izlenebilir hale getirmek için başlangıç olarak Airola vd. [13]'deki çalışmaya benzer bir yol izlenmiştir. Bunun için Eş. 1 hatasının bir tahmini olan gözlemsel risk, eğitim kümesi üzerinde tanımlanan normalleştirilmiş eşli en küçük kareler kayıp fonksiyonudur ve aşağıdaki gibi tanımlanır.

$$\frac{1}{2|T|}\sum_{i,j\in T}(y_i-y_j-x_i^Tw+x_j^Tw)^2. \tag{2}$$

Varsayalım ki

$$L_h=I_h-\frac{1}{h}1_h(1_h)^T$$

$h\times h$ merkezselleştirme matrisi, I, $h\times h$ birim matris ve 1, $h\times 1$ birlerden oluşan bir sütun vektörüdür. L, simetrik bir idempotent matristir ve bir vektörle çarpıldığında vektörün tüm elemanlarından ilgili vektörün ortalamasını çıkartır. Hatta, aşağıdaki eşitlik gösterilebilir.

$$\frac{1}{2h}\sum_{i,j=1}^{h}(c_i-c_j)=c^TL_hc,$$

L matrisi

$$L=I-pp^T$$

olacak şekilde ayrıştırılabilir, burada $p=\left(1/\sqrt{|T|}\right)1$ olmak üzere $m\times 1$ boyutlu bir vektördür. L, simetrik ve idempotent bir matris olduğu için Eş. 2'deki gözlemsel risk matris notasyonunda

$$\left(X^Tw-y\right)^TL\left(X^Tw-y\right)=\left(LX^Tw-Ly\right)^T(LX^Tw-Ly),$$

şeklinde yeniden yazılabilir. Dolayısıyla veri matrisini ve etiket vektörünü merkezleştirerek, gözlemsel risk terimi, standart bir en küçük kareler formuna dönüştürülmüş olur ve denk olarak aşağıdaki gibi verilebilir.

$$\left\|LX^Tw-Ly\right\|^2. \tag{3}$$

Eş. 3'ün minimize edilmesiyle

$$\boldsymbol{LX^T w = Ly} \tag{4}$$

denklemi elde edilir, bu denklem en küçük kareler bağlamında en iyi uyan minimumdur. Burada ele alınan yaklaşım ile standart en küçük kareler arasındaki ilişki teorik olarak şu anlama gelir; Veri matrisi ve etiket vektörü öncelikle merkezileştirilir ve iyi bilinen bir en küçük kareler eğitim algoritması kullanılarak sürece devam edilebilir. Ama, bu veri matrisinin seyreklik yapısını bozar ve sonuç olarak uygulanan metot büyük seyrek verili problemlere elverişli bir şekilde ölçeklenemez.

Doğrusal denklem sistemlerinin cebirsel olarak çözülmesi literatürde üzerinde yoğun olarak çalışılmış konulardan biridir. Bu tür problemlerin genellikle hasta-koşullu olduğu iyi bilinen bir gerçektir. Gözlemsel risk minimizasyonunda elde edilen çözüm altta yatan gerçek konsepti modellemek yerine verideki gürültüyü modeller. Bu durumda aşırı-uyum problem ile karşılaşırız ve bunun üstesinden gelebilmek için modele bir düzeltme uygulanması gerekir.

Bu zamana kadar bu konudaki en iyi yaklaşım tipik olarak $\boldsymbol{\lambda\|w\|^2}$ şeklinde bir düzeltme teriminin minimizasyon problemine eklendiği Tikhonov düzeltmesidir. $\boldsymbol{\lambda}$, veri uyumu ile model karmaşıklığı arasındaki dengeyi kontrol eden pozitif bir sabittir ve düzeltme parametresi olarak bilinir. Eş. 4'e Tikhonov düzeltmesi uygulanmasıyla

$$\boldsymbol{LX^T w + \lambda w = Ly} \tag{5}$$

eşitliği elde edilir. Düzeltmenin uygulanması

$$\boldsymbol{w = \left(XLX^T + \lambda I\right)^{-1} XLy} \tag{6}$$

olarak verilen tek bir minimumun bulunmasını garanti eder. Bu çözüme ulaşırken doğrudan matris tersinin alınması etkin bir yol olmamaktadır. Bunun yerine, veri matrisinin tekil değer ayrışımı gibi matris ayrışımlarına dayanan algoritmalar önerilmektedir. Yoğun doğrusal cebire dayanan bu tür eğitim algoritmaları $\boldsymbol{m}$ ya da $\boldsymbol{n}$'den biri küçük olduğu sürece oldukça pratik algoritmalardır. Birinci duruma örnek vermek gerekirse, yaşam bilimlerinde binlerce gen ölçüldüğü için verinin boyutu çok yüksektir ama insanların üzerinde bu oldukça pahalı testleri gerçekleştirmenin maliyeti çok yüksek olduğundan örneklem genişliği oldukça küçüktür. Buna karşın, ikinci duruma örnek olarak arama motoru gibi enformasyon teknolojilerinin kullanıldığı alanlarda yapılan araştırmalarda ele alınan veri kümeleri on binlerden yüz binlere kadar olan örneklerden oluşmakla beraber genellikle 10 ile 100 arasında üst düzey değişken içerir. Her iki durumda da yoğun doğrusal cebire dayanan eğitim algoritmaları oldukça etkindir. Ama, hem büyük örneklem genişliğine sahip hem de yüksek boyutlu ama seyrek veri kümelerinde bu algoritmalar elverişli değildir çünkü $\boldsymbol{m}$ veya $\boldsymbol{n}$'e göre kuadratik hafıza gereksinimi ve kübik hesaplama zamanı gerektirir.

Gradyan düşümü algoritması seyrek doğrusal denklem sistemlerini çözmek için güçlü iteratif bir metottur. Bu metot seyrek denklemler için de oldukça elverişlidir çünkü bir vektörün seyrek matrislerle tekrarlı çarpımına dayanır. Ayrıca, uygulaması oldukça basit ve hafıza gereksinimi az olan bir metottur. Ama, en önemli dezavantajı diğer daha karmaşık algoritmalara göre daha yavaş bir yakınsama oranına sahip olmasıdır. Gradyan düşümü, $\boldsymbol{H}$'nin simetrik ve pozitif tanımlı olduğu $\boldsymbol{Hw = b}$ türünde denklemleri çözmek için bir metottur.
Eş. 6'da verilen çözüme ulaşmamızı sağlayan Eş. 5

$$\boldsymbol{(XX^T - Xpp^T X^T + \lambda I)w = X(y - pp^T y)} \tag{7}$$

şeklinde yazılabilir. Bu durumda $\boldsymbol{H = XX^T - Xpp^T X^T + \lambda I}$ ve $\boldsymbol{b = X(y - pp^T y)}$ alındığında Eş. 7 de gradyan düşümü ile çözülebilir.

Taş ve Memmedli [14], gradyan düşümü algoritmasının hızlı bir versiyonunu (HGD) geliştirmiştir (Algoritma 1). HGD'nin temeli, kuadratik hata fonksiyonunun Hessian matrisinin en büyük ve en küçük özdeğerinden hesaplanan optimala yakın bir öğrenme oranı ($\boldsymbol{\eta}$) ve momentum katsayısına ($\boldsymbol{\mu}$) dayanır. Optimale yakın öğrenme oranı ve momentum katsayısı aşağıdaki gibi hesaplanır.

$$\boldsymbol{\eta = \frac{1}{\sqrt{\kappa_1 \kappa_n}}}, \qquad \boldsymbol{\mu = \frac{\left(\sqrt{\frac{\kappa_1}{\kappa_n}} - 1\right)^2}{\left(\sqrt{\frac{\kappa_1}{\kappa_n}} + 1\right)^2}},$$

burada $\boldsymbol{\kappa_1}$ ve $\boldsymbol{\kappa_n}$ sırasıyla Hessian matrisinin en büyük ve en küçük özdeğerlerini ifade etmektedir. HGD'nin en büyük avantajı, normalde elle ayarlanması gereken öğrenme oranı ve momentum katsayısının optimale en yakın değerlerini ilgili formüllerle doğrudan elde edilmesidir. Bu sayede parametre optimizasyonu için gereken hesaplama maliyeti ortadan kalkar. Kullanıcı bu parametreleri optimizasyon başlangıcında kolayca hesaplar.

Algoritma 1: HGD algoritması

Veri: Hata fonksiyonu F (Eş. 1), girdi matrisi $\boldsymbol{X}$, çıktılar $\boldsymbol{y}$, başlangıç noktası $\boldsymbol{w_0}$, yakınsama toleransı ϵ ve düzeltme parametresi λ

Çıktı: Çözüm vektörü $\boldsymbol{w}^*$

κ_1 ve κ_n'yi **Algoritma 2**'yi kullanarak yaklaşık olarak tahmin et.

Öğrenme oranını ve momentum katsayısını hesapla

$$\eta = \frac{1}{\sqrt{\hat{\kappa}_1 \hat{\kappa}_n}}, \qquad \mu = \frac{\left(\sqrt{\frac{\hat{\kappa}_1}{\hat{\kappa}_n}} - 1\right)^2}{\left(\sqrt{\frac{\hat{\kappa}_1}{\hat{\kappa}_n}} + 1\right)^2}$$

Tekrarla

$\nabla F(\boldsymbol{w}_t) = \boldsymbol{A w}_t - \boldsymbol{b}$;

$\boldsymbol{w}_t = -(1-\mu)\eta \nabla F(\boldsymbol{w}_t) + \mu(\boldsymbol{w}_t - \boldsymbol{w}_{t-1})$;

$\epsilon < 1e-5$ olana kadar.

HGD algoritmasında öğrenme oranını ve momentum katsayısını belirleyebilmek için Hessian matrisinin en büyük ve en küçük özdeğerini bulmamız gerekir. Yukarıda ele aldığımız eşli sıralama problemi için bunun hesaplama maliyeti $O(n^3)$'tür. Dolayısıyla, ilgili özdeğerleri kesin olarak hesaplamak yerine Power iterasyonunu kullanarak bu değerleri yaklaşık olarak tahmin edebiliriz. Bunun sonucu olarak oluşan algoritma aşağıda verilmiştir (Algoritma 2). Yapılan deneylerde en büyük ve en küçük özdeğerin yaklaşık tahminleriyle hesaplanan öğrenme oranı ve momentum katsayısının bu parametrelerin gerçek değerlerine çok yakın olduğu görülmüştür [14].

Dikkat edilirse, verinin seyreklik yapısını koruyan etkin bir hesaplama gerçekleştirmek için Eş. 7'de L yerine $I - pp^T$ kullanılmıştır. Sol taraftaki matris çarpımlarının hiçbiri doğrudan gerçekleştirilmez, bunun yerine alternatif olarak sunulan bir vektörle çarpımı kullanılır. Gradyan düşümünün her bir iterasyonunda, o iterasyondaki çözüm vektörü w_t kullanılarak Eş. 7'nin sol tarafının hesaplanması gerekir. Bu hesaplama $X\left(X^T w_t - p\left(p^T (X^T w_t)\right)\right) + \lambda w_t$ şeklinde hesaplanabilir, burada parantezler çarpımların hangi sırada gerçekleşeceğini belirtmek için kullanılmıştır. Sadece seyrek matris-vektör çarpımlarına ihtiyaç duyulur. X, ms girdiye ve p' de m girdiye sahip olduğundan tüm işlemler serisi $O(ms)$ zamanda hesaplanabilir. Gradyan düşümü ile modelin eğitiminin karmaşıklığı $O(tms)$'dir, burada t gerekli olan iterasyon sayısıdır.

Gradyan düşümü hakkında temel bir sonuç t'nin, H'nin farklı özdeğerlerinin sayısı ile orantılı olarak sınırlı olmasıdır. XLX^T'nin maksimum rankı gerekli olan iterasyon sayısı için bir üst sınır verir. Bir matris en fazla rankı kadar farklı özdeğere sahiptir ve düzeltmede en fazla bir yeni özdeğer ekler. Bu durumda gradyan düşümü örneklem genişliği ya da verinin boyutundan biri az olduğunda hızlı bir şekilde yakınsayabilir. Diğer yandan, çok büyük seyrek veri matrisleri için bu garanti edilemez çünkü hem m hem de n çok büyük olabilir.

XLX^T matrisinin koşul sayısı $\frac{\gamma_{maks}}{\gamma_{min}}$ olsun, burada γ_{maks} ve γ_{min} sırasıyla XLX^T'nin en büyük ve en küçük özdeğerleridir. Bu matrisin özdeğer ayrışımı incelenerek, $XLX^T + \lambda I$ matrisinin, Eş. 7'nin

sol tarafında verilen matrise denk olduğunu belirtmek kaydıyla, koşul sayısının $\kappa = \frac{\gamma_{maks}+\lambda}{\gamma_{min}+\lambda}$ olduğu görülür. κ, düzeltme seviyesi arttırıldığında yukarıdan bire yaklaşır, bu sınır yakınsama hızının düzeltme parametresi λ ile ters orantılı olduğunu gösterir. Yapılan deneylerde de bu durum gözlemlenmiştir, λ'nın yeterince büyük değerlerinde gradyan düşümü algoritması çözüme çok hızlı bir şekilde ulaşmıştır. Diğer yandan, λ'nın çok küçük değerlerinde gerekli olan iterasyon sayısı çok daha büyük olmaktadır (Şekil 2).

Algoritma 2: κ_1 ve κ_2'nin tahmini için Power iterasyonu

Veri: Girdi matrisi $\boldsymbol{X}$, düzeltme parametresi λ ve yakınsama toleransı ϵ

Çıktı: Hessian'ın en büyük ve en küçük özdeğer tahmini $\hat{\kappa}_1$ ve $\hat{\kappa}_n$

$k \leftarrow 0$;

n boyutlu $\boldsymbol{v}$ ve $\boldsymbol{\psi}$ vektörlerini rassal olarak oluştur;

$\tilde{\boldsymbol{v}} = \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|}$, $\tilde{\boldsymbol{\psi}} = \frac{\boldsymbol{\psi}}{\|\boldsymbol{\psi}\|}$;

Tekrarla

$\boldsymbol{v} \leftarrow \boldsymbol{X}\boldsymbol{X}'\tilde{\boldsymbol{v}} + \lambda\tilde{\boldsymbol{v}}$;

$\hat{\kappa}_1 \leftarrow \|\boldsymbol{v}\|$;

$\tilde{\boldsymbol{v}} \leftarrow \frac{\boldsymbol{v}}{\hat{\kappa}_1}$;

$\boldsymbol{\omega} \leftarrow \boldsymbol{X}\boldsymbol{X}'\tilde{\boldsymbol{\omega}} + \lambda\tilde{\boldsymbol{\omega}} - \hat{\kappa}_1\tilde{\boldsymbol{\omega}}$;

$\tilde{\boldsymbol{\omega}} = \frac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|}$;

$\epsilon \lhd 1e-5$ olana kadar.

$\hat{\kappa}_n \leftarrow \frac{\boldsymbol{\omega}'\boldsymbol{X}\boldsymbol{X}'\boldsymbol{\omega}+\boldsymbol{\omega}'\lambda\boldsymbol{\omega}}{\boldsymbol{\omega}'\boldsymbol{\omega}}$.

Geliştirilen sıralama algoritmasının performansını değerlendirmek için bir metin sınıflandırması problemi ele alınmıştır. Veri kümesi Reuters RCV1 [15] koleksiyonu kullanılarak oluşturulmuştur. Bu veri kümesinin tercih edilmesini sebebi büyük ölçekli (örnek sayısının fazla olması) ve büyük boyutlu (değişken sayısının çok olması) olmasıdır. Aynı zamanda, bu koleksiyon farklı sınıflandırma metotlarının karşılaştırılmasında oldukça sık kullanılan popüler bir veri kümesidir. Veri kümesi 103 kategoriye ait 804,414 doküman örneğinden oluşmakta ve bir sözlük kullanılarak türetilen 47236 özellikten oluşmaktadır. Oluşan veri matrisi dikkate alındığında matrisin %0.16'sı sıfırdan farklı değerlerden oluşmaktadır, dolayısıyla seyrek bir matristir (0'dan farklı elemanların az olduğu). Amaç bu veri kümesinde CCAT kategorisinde bulunan dokümanları bu kategoride bulunmayan dokümanlara göre daha üst sıralarda yer alacak şekilde sıralamaktır. Bunun için, dokümanlardan terim sıklığı (ts) ve ters doküman sıklığı (tds) değerlerini içeren özellikler çıkartılmıştır. Terim sıklığı-ters doküman sıklığı (ts-tds), bir dokümandaki bir terimin önemini, o dokümanda ve dokümanların belirli bir koleksiyonunda ne sıklıkta göründüğüne dayanan sayısal bir ölçüttür. Bu ölçümün altında yatan fikir: Bir terim bir dokümanda sık sık görülüyorsa, bu önemlidir ve bu terime yüksek puan verilmesi gerekir. Ancak, bu terim çok fazla dokümanda da görünüyorsa, muhtemelen bu terim bu doküman için benzersiz bir tanımlayıcı değildir ve bu nedenle bu terime daha düşük bir puan atamak gerekir. Bu ölçümün matematiksel formülü

$$ts\text{-}tds(t,d,D) = ts(t,d) \times tds(t,D)$$

şeklinde verilir. Burada t terimleri, d her bir dokümanı ve D dokümanların bir koleksiyonunu gösterir. Tds

$$tds(t, D) = \frac{|D|}{1 + |\{d \in D : t \in d\}|}$$

formülü ile hesaplanır. Veri kümesi, eğitim kümesinde 781,265 doküman ve test kümesinde 23,149 doküman olacak şekilde iki kümeye ayrılmıştır.

3. Bulgular ve Tartışma

Öncelikle geliştirilen HGD algoritmasının yakınsama performansını test etmek için bir simülasyon çalışması yapılmıştır. Bu simülasyon çalışmasında

$$F(x) = \frac{1}{2}x^T Hx - bTx + c \tag{8}$$

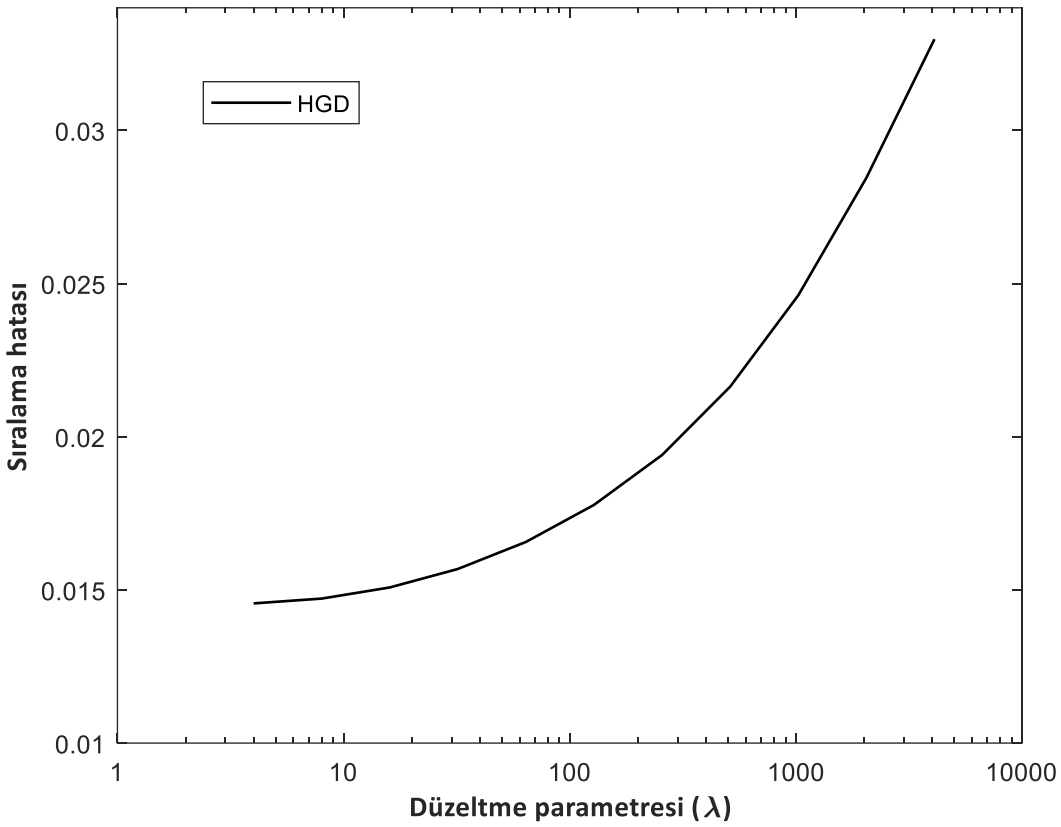
şeklinde kuadratik bir hata fonksiyonu rassal olarak oluşturulmuştur ve basit olması için, bu hata fonksiyonunun durağan noktasının orijinde olduğu ve o noktada sıfır değerini aldığı varsayılmıştır. Bu durumda Eş. 8'deki b ve c terimleri yok olur. Dolayısıyla, Eş. 8'deki hata fonksiyonu sadece Hessian matrisi (H) ile belirlenebilir, yani belirli bir yapay test problemi oluşturmak için sadece bir H matrisi oluşturmak yeterlidir. Test problemleri, problemin boyutu ve Hessian matrisinin koşul sayısı parametrik olarak belirlenecek şekilde oluşturulmuştur. Daha sonra HGD ve MGD algoritmaları bu yapay test problemleri üzerinde çalıştırılmıştır.

Tablo 1. HGD ve MGD algoritmalarının farklı problem boyutu ve koşul sayısına göre yakınsama süreleri (sn.)

Problem boyutu	Algoritma	Koşul sayısı (κ_1/κ_n)			
		10	10^2	10^3	10^4
$n = 10$	**HGD**	**0.01**	**0.04**	**0.03**	**0.06**
	MGD	0.22	1.11	1.93	5.51
Problem boyutu	Algoritma	Koşul sayısı (κ_1/κ_n)			
		10	10^2	10^3	10^4
$n = 10^2$	**HGD**	**0.06**	**0.05**	**0.06**	**0.14**
	MGD	0.39	1.15	3.55	10.67
Problem boyutu	Algoritma	Koşul sayısı (κ_1/κ_n)			
		10	10^2	10^3	10^4
$n = 10^3$	**HGD**	**9.47**	**7.03**	**12.66**	**23.97**
	MGD	67.68	204.15	623.49	1819.30

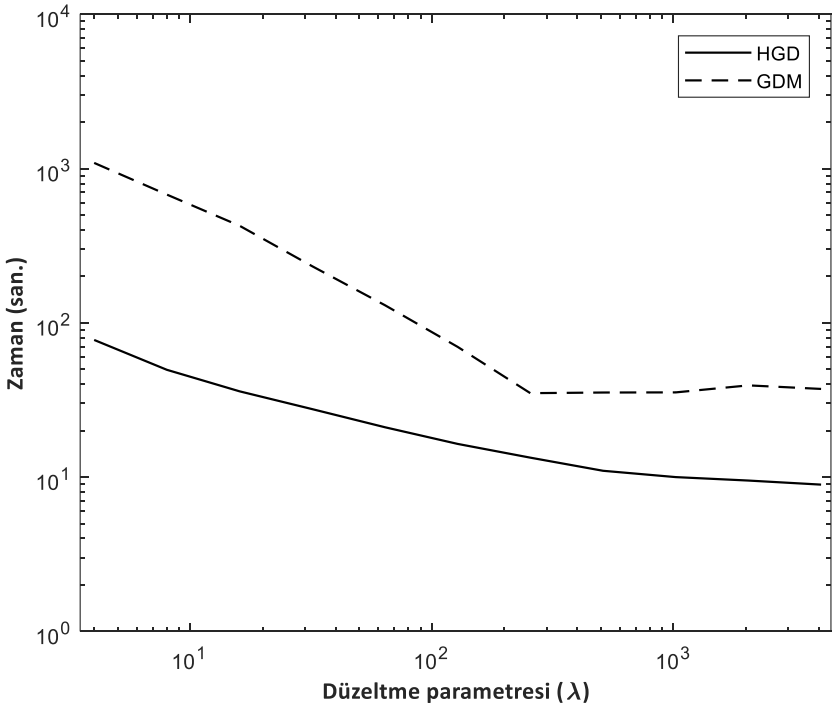
İlk olarak HGD'nin sıralama problemindeki performansı sıralama hatası göz önüne alınarak incelenmiş ve daha sonra yakınsama performansı klasik gradyan düşümü algoritması ile karşılaştırmalı olarak ele alınmıştır. Bu algoritmaların geliştirilmesinde Linux işletim sistemi üzerinde çalışan özgür bir paket program olan Octave 4.4.1 kullanılmıştır. Deneyler 64 bit Linux işletim sistemli 8GB yüklü belleği olan Intel® XEON® 3.40 GHz işlemcili masaüstü bir bilgisayar üzerinde gerçekleştirilmiştir.

Her bir deneyde farklı problem boyutu ve koşul sayısı kullanılmış ve her durum için 10 tekrar gerçekleştirilmiştir. Algoritmaların yakınsama sürelerinin ortalamaları Tablo 1'de verilmiştir. Tablo 1'deki koyu değerler 0.01 anlamlılık düzeyinde istatistiksel olarak anlamlı farklılıkları göstermektedir. Önerilen HGD algoritması her durumda klasik MGD'den daha iyi yakınsama performansı göstermiştir. Özellikle problem boyutu büyüdüğünde HGD, MGD'ye göre çok hızlı bir şekilde çözüme yakınsamıştır.



Şekil 1. HGD'nin farklı düzeltme seviyelerine göre sıralama performansı

HGD'nin farklı düzeltme düzeylerine (λ) göre test kümesindeki sıralama performansı Şekil 1'de gösterilmiştir. λ'nın değeri arttıkça model karmaşıklığı azalmakta, dolayısıyla sıralama problemini çözmek için daha basit bir model kullanılmakta ve bu da test kümesinde elde edilen sıralama performansını düşürmektedir yani bir başka ifadeyle test kümesi üzerindeki sıralama hatası artar. Bu durum aynı zamanda klasik MGD için de geçerlidir, her iki algoritma da belirli bir λ değeri için aynı sıralama performansına sahiptir. Diğer yandan, belirli düzeltme seviyeleri için algoritmaların ilgili sıralama performanslarına karşılık gelen yakınsama zamanları Şekil 2'de verilmiştir. λ'ya göre yakınsama zamanları incelendiğinde HGD, klasik MGD'den çok daha iyi bir performans göstermiştir. Özellikle çok düşük ve orta seviyelerdeki düzeltmelerde, yani λ yeterince küçük olduğunda (daha karmaşık modellerde), HGD çözüme çok hızlı bir şekilde yakınsarken MGD'nin yakınsaması HGD'ye göre çok daha yavaş gerçekleşmiştir. Düzeltme seviyesi çok büyük olduğunda ise HGD ile MGD arasında anlamlı bir farklılık gözlemlenmemiştir. Bir noktanın altını tekrar çizmek gerekir, yüksek düzeltme seviyelerinde modelin öğrenme kapasitesi o kadar azalır ki sıralama modeli çok basit bir model haline gelir ve bu da test kümesi üzerindeki sırlama hatasında ciddi bir artışa neden olur. Önemli olan düşük ve orta seviyeli uygun bir düzeltme seviyesinde hızlı bir yakınsama gerçekleştirmektir. Bu durum dikkate alındığında HGD'nin klasik MGD'den daha etkin bir algoritma olduğu açıktır. Ayrıca, bu sonuçlar simülasyon çalışmasını doğrular niteliktedir.



Şekil 2. HGD ve GDM'nin farklı düzeltme seviyelerine göre yakınsama zamanları

4. Sonuç ve Öneriler

Sıralamayı öğrenmek yapay öğrenmede önemli bir role sahip olmuştur çünkü sıralama belirli bir alana ilgili aramaları ve sorguları elektronik ortamdaki dokümanlarla ilişkilendirir. Eşli bir sıralama hatasını minimize etmeye çalışan birçok sıralama metodu bulunmaktadır. Ama eşli bir sıralama hatasını doğrudan minimize etmek hesapsal olarak izlenebilir değildir. Dolayısıyla bu çalışmada bu problemin üstesinden gelebilmek için, sonucunda doğrusal bir denklem sistemine ulaşan eşli bir en küçük kareler yaklaşımı kullanılmıştır. Bu doğrusal sistemi çözmek kuadratik bir hata fonksiyonunu minimize etmeye denktir. Kuadratik hata fonksiyonunu minimize etmeye yönelik optimale yakın öğrenme oranı ve momentum faktörüne sahip hızlı bir gradyan düşümü algoritması önerilmiştir. Optimale yakın öğrenme parametreleri Hessian matrisinin en büyük ve en küçük özdeğerleri kullanılarak belirlenmiştir. Power iterasyonu bu özdeğerleri etkin ve hızlı bir şekilde tahmin etmek için uyarlanmıştır. Oluşan hızlı gradyan düşümü (HGD) algoritması klasik momentumlu gradyan düşümü (GDM) algoritmasıyla yakınsama sürelerine göre karşılaştırıldığında çok daha iyi performans göstermiştir. HGD'nin bu performansı düzeltme parametresinin seçimine ve problemin zorluğuna bağlı kalmaksızın hemen hemen aynı düzeyde gerçekleşmiştir.

Değişken sayısının çok olduğu ve örneklem genişliğinin de çok büyük olduğu veri kümeleri derlenen alanlar hızla artmakta ve ölçeklenebilirlik bir öğrenme algoritmasının sahip olması gereken temel bir özellik ve aynı zamanda farklı öğrenme algoritmalarını karşılaştırmada ve geliştirmede önemli bir kriter haline gelmektedir. Bu çalışma daha çok deterministik bir kuadratik hata fonksiyonu üzerine kurulmuş olmasına rağmen HGD stokastik duruma da uyarlanabilir. Bu alanda gelecek için açık bir problem olarak görülebilir.

Kaynaklar

[1] Caruana R., Baluja S., Mitchell, T. 1996. Using the future to "sort out" the present: Rankprop and multitask learning for medical risk evaluation, in *Advances in Neural Information Processing Systems*, 959-965.

[2] Herbrich R., Graepel, T., Obermayer, K. 2000. Large margin rank boundaries for ordinal regression, in *Advances in Large Margin Classifiers*, MIT Press, 115-132.

[3] Crammer, K., Yoram S. 2002. Pranking with ranking, in *Advances in neural information processing systems*, 641-647.

[4] Menon A.K., Williamson, R.C. 2016. Bipartite ranking: a risk-theoretic perspective, Journal of Machine Learning Research, 17 (195): 1-102.

[5] Haltaş A., Alkan A., Karabulut M. 2015. Performance analysis of heuristic search algorithms in text classification, Journal of the Faculty of Engineering and Architecture of Gazi University, 30 (3): 417-427.

[6] Kaya Y., Ertugrul, O.F. 2016. A novel feature extraction approach for text-based language identification: Binary patterns, Journal of The Faculty of Engineering and Architecture of Gazi University, 31 (4): 1085-1094.

[7] Harrington, E. 2003. Online ranking/collaborative filtering using the Perceptron algorithm, International Conference on Machine Learning, pp250-257.

[8] Dekel, O., Singer, Y., Manning, C.D. 2004. Loglinear models for label-ranking, in *Advances in neural information processing systems*, 497-504.

[9] Freund, Y., Iyer, R., Schapire, R., Singer, Y. 2003. An efficient boosting algorithm for combining preferences, Journal of Machine Learning Research, 4, 933-969.

[10] Song Y., Wang, H., He, X. 2014. Adapting deep ranknet for personalized search, Proceedings of the 7th ACM international conference on Web search and data mining, pp83-92, ACM.

[11] Zong, W., Huang, G.B. 2014. Learning to rank with extreme learning machine, Neural processing letters, 39 (2): 155-166.

[12] Busa-Fekete, R., Hüllermeier, E. 2014. A survey of preference-based online learning with bandit algorithms, International Conference on Algorithmic Learning Theory, pp18-39, Springer, Cham.

[13] Airola, A., Pahikkala, T., Salakoski, T. 2010. Large scale training methods for linear RankRLS, Proceedings of the ECML/PKDD-Workshop on Preference Learning, E. Hüllermeier and J. Fürnkranz, Eds.

[14] Taş, E., Memmedli, M. 2017. Near optimal step size and momentum in gradient descent for quadratic functions, Turkish Journal of Mathematics, 41 (1): 110-121.

[15] Lewis, D.D., Yang, Y., Rose, T.G., Li, F. 2004. Rcv1: A new benchmark collection for text categorization research, Journal of machine learning research, 5 (Apr): 361-397.