Açık Geri Bildirim ile Derin Öğrenme Yöntemi Kullanılarak Ürün Öneri Sistemi

Product Recommendation System with Explicit Feedback Using Deep Learning Methods

Enes Kantepe
Bilgisayar Mühendisliği Bölümü
Marmara Üniversitesi
İstanbul, Türkiye
eneskantepe@gmail.com

Zehra Aysun Altikardes
Bilgisayar Teknolojileri Bölümü
Marmara Üniversitesi
İstanbul, Türkiye
aaltikardes@marmara.edu.tr

Hasan Erdal
Elektrik-Elektronik Mühendisliği
Marmara Üniversitesi
İstanbul, Türkiye
herdal@marmara.edu.tr

***Özetçe*— Günümüzde internet alışverişi sırasında kullanıcıları bireysel tercihlerine göre en doğru ürüne yönlendirecek öneri sistemleri geliştirilmeye ve iyileştirilmeye yönelik çalışmalar yapılmaktadır. Bu çalışmada MovieLens veriseti ve derin öğrenme yöntemlerinden biri olan otomatik kodlayıcılar kullanılarak öneri sistemi tasarlanmıştır. Öneri sistemi tasarlanırken, sistemin başarısının artırılması için Gradyan Alçalma, Hızlı Gradyan Alçalma, RmsProp ve Adam optimizasyon algoritmaları Python programlama dilinde TensorFlow platformu kullanılarak denenmiş ve veri miktarındaki artışın optimizasyon algoritması üzerindeki etkisi incelenmiştir. Sonuç olarak en başarılı optimizasyon algoritmasının 1,363 test puan hatası ile Adam algoritması olduğu tespit edilmiştir. Ayrıca eğitim setindeki boşluk oranının azaltılmasıyla test puan hatasının azaldığı görülmüştür.**

Anahtar Kelimeler*— *öneri sistemleri, derin öğrenme, otomatik kodlayıcılar, optimizasyon algoritmaları

***Abstract*— Today, efforts are made to develop and improve recommendation systems that will direct users to the right product according to their individual preferences during internet shopping. In this study, the recommendation system was designed with Autoencoders, which are one of the methods of deep learning and MovieLens dataset. While designing the system, various optimization algorithms, namely Gradient Descent, Gradient Descent with Momentum, RmsProp and Adam (Adaptive Momentum Optimization), were tried by using TensorFlow in the Python programming language. Moreover, the effect of increasing the amount of the data on the optimization algorithm was analyzed. Consequently, it was effectively demonstrated that the most successful one was the Adam algorithm with a test error of 1.363. It was also observed that decreasing the sparsity on the training data leads to a lower test error.**

Keywords—recommendation system, deep learning, autoencoder, optimization algorithms

I. GİRİŞ

Günümüzde kullanıcılar alışveriş yapma, müzik dinleme, film izleme gibi ihtiyaçlarının büyük bir kısmını internet üzerindeki sanal mağazaları kullanarak sağlamaktadır. Kullanıcılar ihtiyaçlarını internet üzerinden sağlarken birkaç problemle karşılaşabilirler. Örneğin bir müşteri film veya ayakkabı gibi bir ihtiyacını karşılamak amacıyla mağazaya girdiğinde mağaza çalışanı tarafından karşılanır ve kendisine ihtiyacı sorulur. Bu sayede mağazadaki ürünlerden kendisine en uygun ürünler önerilir. İnternette ise genellikle kullanıcıları karşılayan bir görevli bulunmamaktadır. Ayrıca alınabilecek ürün sayısı ve çeşitliliği mağazalarda bulunan ürün sayısından çok daha fazladır. Dolayısıyla bu kadar fazla ürünün bulunduğu ürün kataloğundan kullanıcının seçim yapması ve karar vermesi zordur. Bu sorunun çözümü için bazı sanal mağazalarda canlı destek hizmeti sunulmaktadır. Bunların büyük bir kısmı yazılımsal olarak kullanıcıyı karşılamakta ve öneri sunmaktadır. Akıllı yazılımlarla tasarlanmış bu öneri sistemleri kullanıcılardan bilgi toplar. Bunlar, kullanıcılardan internet sitelerine üye olurken alınan bilgiler ya da internet sitesi üzerindeki hareketlerini izleyerek elde edilen bilgiler olabilir. Tıklanan ürünler, hangi haberde kaç saniye kalındığı, bir filme verilen puan ya da satın alınan ürün hakkında yapılan yorum bunlara örnek olarak verilebilir. Bu veriler kullanılarak öneri sistemleri geliştirilmektedir.

Öneri sistemleri kullanıcıların ilgilerini öğrenerek, kullanıcılara özgü kişisel öneri sunan sistemlerdir. Amacı kullanıcıların ilgisini çekebilecekleri ürünleri tahmin etmeye çalışmaktır. Kullanıcılara doğru zamanda doğru ürünü önermek kullanıcı memnuniyetini artırır. Bu sayede şirketler hem satış miktarlarını artırır hem de kalıcı kullanıcı elde ederler [1].

Günümüzde toplanan verinin artması ve bilgisayar donanımlarındaki gelişmeler ile birlikte hesaplama zamanlarının düşmesi, derin öğrenme yöntemlerinin tekrar

978-1-7281-9136-2/20/$31.00 © 2020 IEEE

Authorized licensed use limited to: ULAKBIM UASL - MARMARA UNIVERSITY. Downloaded on December 28,2022 at 08:14:08 UTC from IEEE Xplore. Restrictions apply.

ilgi görmesine neden olmuştur [2]. Çünkü derin öğrenme yöntemleri kullanılan veri miktarı arttıkça diğer makine öğrenmesi yöntemlerine göre daha başarılı sonuçlar vermektedir [3]. Bu sebeple öneri sistemi üzerine çalışma yapan araştırmacılar derin öğrenmeyi öneri sistemleri üzerinde kullanmaya çalışmaya başlamışlardır.

II. BENZER ÇALIŞMALAR

Derin öğrenmenin birçok alanda diğer makine öğrenmesi algoritmalarına göre daha başarılı sonuçlar vermesiyle araştırmacılar aşağıda belirtilen çalışmaları yapmışlardır.

Ruslan Salakhutdinov ürün öneri sistemlerinde Netflix çevrimiçi film servis sağlayıcısının sağladığı veri seti üzerinde yapay sinir ağı (YSA) çeşidi olan kısıtlı boltzman makineleri - restricted boltzman machine (RBM)' leri kullanmış ve Netflix'in kendi öneri sisteminden % 6 daha iyi sonuç verdiğini göstermiştir [4].

Suvash Sedhain Otomatik Kodlayıcıları Netflix ve MovieLens veri seti üzerinde kullanmıştır. Otomatik Kodlayıcıların RBM ve Matrisi çarpanlarına ayırma yöntemlerine göre daha üstün bir performans sağladığını sonuçlar üzerinde göstermiştir [5].

Florian Strub SDA'ları (Stacked Denoising AutoEncoder) MovieLens 1M ve Jester veriseti üzerinde denemiş ve 4 katmanlı bir Otomatik Kodlayıcının 2 katmanlı bir Otomatik Kodlayıcıya göre daha üstün sonuçlar verdiğini göstermiştir [1].

Balazs Hidasi sıralı veri problemlerinde üstün başarı sağlayan tekrarlayan yapay sinir ağları - recurrent neural network (RNN) leri oturum bazlı olarak elde ettiği verileri zamana göre sıralı hale getirmiş ve iki veri seti üzerinde denemiş sonuçlarını paylaşmıştır [6].

Bahriye Akay MovieLens veri seti üzerinde DBN'leri (Deep Belief Network) farklı öğrenme algoritmalarıyla kullanmış ve Adamax algoritmasının ezberleme problemiyle karşı karşıya geldiği sonucuna varmıştır [7].

Yu Zhu bir RNN çeşidi olan LSTM' lere zaman aralıkları ekleyerek üç farklı model önermiş bu yöntemin öneri performansını artırdığının sonuçlarını iki farklı veri seti üzerinde göstermiştir [8].

Daniel Sánchez yüksek lisans tezinde kullanıcı davranışlarının (ürün satin alma, görüntüleme vb.) sıralı bir veriye çevrilebileceğinde bahsetmiştir. MovieLens ve Santander veri setlerini RNN'ler üzerinde kullanarak kullanıcıların bir sonra yapacağı hareketi tahmin etmeye çalışmıştır [9].

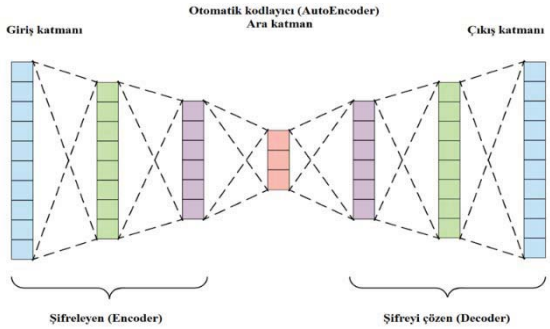
Bu çalışmada yapılan çalışmalardan farklı olarak Movilens 1M veri seti üzerinde otomatik kodlayıcılar kullanılarak en uygun optimizasyon algoritmasının bulunması amaçlanmış ve veri artışının öneri sistemi üzerindeki etkisi incelenmiştir.

III. DERİN ÖĞRENME İLE ÜRÜN ÖNERİ SİSTEMİ

Derin öğrenme temelinde YSA'lara dayanmaktadır. Çok katmanlı tasarlanan bu modeller ile karmaşık ve zor problemlerin (ses tanıma, dil çevirimi, resim tanıma) çözümü kolaylaşmıştır.

A. Otomatik Kodlayıcılar

Bir YSA çeşidi olan Otomatik Kodlayıcılar (AutoEncoder), genellikle verilen giriş değerleri ile çıkışta elde edilen değerlerin biribirine yakın olması beklenen ileri yönlü YSA'lardır. Bir ya da daha fazla gizli katmana sahiptirler. YSA önce giriş verisini şifreleyen (Encoder), sonrada şifrelediği veriyi çözen (Decoder) katmanlardan oluşmaktadır. Şekil 1'de gösterilmiştir.



Şekil 1. Otomatik kodlayıcı (AutoEncoder)

Bu YSA' larda genellikle hata fonksiyonu olarak Ortalama Karesel Hatanın Karakökü -Root Mean Squared Error (RMSE) kullanılır. Eş. 1 ile gösterilmiştir.

$$\text{RMSE} = \frac{1}{m}\sqrt{\sum_{k=0}^{m}(\text{y} - \text{y}')^2} \qquad (1)$$

y= olması gereken değer
y'= tahmin edilen değer
m= toplam örnek sayısı

Çok katmandan oluşan Otomatik Kodlayıcı, eğitilirken geri yayılım algoritması (backpropagation) kullanılır. Otomatik Kodlayıcılar genellikle verilerin boyutlarını azaltmada ve elimizde yeterli miktarda veri olmadığında veri çoğaltmada kullanılır [10].

Öneri sistemleri açısından matrisi çarpanlarına ayırma metodu ile karşılaştırılırsa, matrisi çarpanlarına ayırma metodu doğrusal (lineer) bir algoritma kullandığı için doğrusal gizli özellikleri (latent factor) öğrenebilme yeteneğine sahiptir. Otomatik Kodlayıcılar ise doğrusal olmayan aktivasyon fonksiyonları kullandığı için doğrusal olmayan daha kompleks gizli özellikleri de öğrenebilir [5].

B. Yapılan Çalışma

Yapılan çalışmaya ait Şekil 2'de görülen iş akış şeması aşamaları aşağıda maddeler halinde açıklanmıştır.

Authorized licensed use limited to: ULAKBIM UASL - MARMARA UNIVERSITY. Downloaded on December 28,2022 at 08:14:08 UTC from IEEE Xplore. Restrictions apply.

- GroupLens araştırmacılarının MovieLens internet sitesi kullanıcılarından açık geri bildirimler toplayarak oluşturduğu veriseti, hangi kullanıcının hangi filme kaç puan verdiğini içermektedir.
- Elde edilen veri seti, test ve eğitim amaçlı ikiye bölünmüştür.
- Test ve eğitim seti her kullanıcı için ayrı bir vektör haline getirilmiştir.
- Otomatik Kodlayıcıların eğitimi için hazır hale gelen veriler kullanılarak YSA'nın kaç katmandan oluşacağı, katmanlardaki nöron sayısı ve aktivasyon fonksiyonları belirlenmiştir.
- Oluşturulan modelde dört faklı optimizasyon algoritması test edilmiştir.
- Veri miktarındaki artış ve azalışın seçilen optimizasyon algoritması üzerindeki etkisi incelenmiştir.

Şekil 2. İş akış şeması

IV. MATERYAL VE YÖNTEM

A. *Veri Seti*

Yapılan çalışmada kullanılan veriler https://grouplens.org sitesinden alınan MovieLens 1M veri setidir [11]. Veri seti dört adet dosyadan oluşmaktadır. README, ratings.dat, movies.dat, users.dat. Çalışmada sadece ratings.dat dosyasındaki veriler kullanılmıştır.

- ratings.dat: Bir kullanıcının hangi filme ne zaman kaç puan verdiği bilgisinden oluşmaktadır. Tablo 1'de gösterilmiştir. Son sütünda yer alan zaman bilgisi ise kullanıcının puanı verdiği tarihin saniye cinsinden değeridir. Hangi tarihe denk geldiği www.epochconverter.com sitesinden çevrilerek bulunabilir.

Tablo I. Puan veri seti

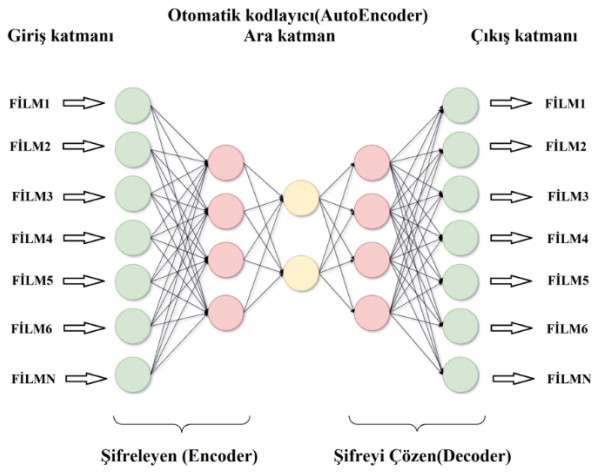
Kullanıcı id	Film id	Puan
1	1193	5
2	434	2
2	3108	3
5	47	3
6	914	5
9	1	5

Dosyada toplam 3.952 farklı film ve 6.040 farklı kullanıcı bulunmaktadır. Kullanıcıların filmlere verdiği puanlar 1 ile 5 arasında değişmektedir. Örneğin 1 id'li kullanıcı 1193 id'li filme 5 puan vermiştir. Veri seti bu şekilde toplam 1.000.209 adet puandan oluşmaktadır.

B. *Yaklaşım*

Verilerin ilk olarak test ve eğitim setine bölünmesi gerekmektedir. Toplam 6.040 adet kullanıcı bulunmaktadır ve bu kullanıcılara ait 1 ile 5 arasındaki 1.000.209 adet puan vardır. Veri seti ikiye bölünürken her kullanıcı hem test setinde hem de eğitim setinde olacak şekilde bölünmüştür. Her kullanıcının verdiği toplam puan sayısının %90'ı eğitim setine %10' u ise test setine dahil edilmiştir. Bu sayede 897.450 adet puan eğitim setini, 102.759 adet puan ise test setini oluşturmuştur.

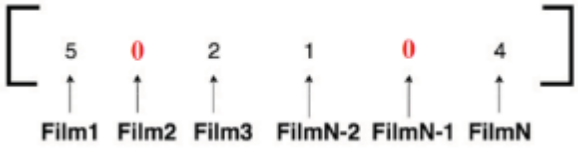
Çalışmada kullanılan veriler Otomatik Kodlayıcıların eğitimi için uygun hale getirilmelidir. Şekil 3'de Otomatik Kodlayıcıların öneri sistemindeki kullanım yapısı gösterilmiştir.



Şekil 3. Öneri sisteminin otomatik kodlayıcı ile modellenmesi

Otomatik Kodlayıcı yöntemini öneri sisteminde kullanabilmek için, giriş katmanındaki her bir nöron bir filmi temsil etmelidir. Dolayısıyla toplam giriş değeri film sayısı kadar olmalıdır. Veri seti hazırlanırken her bir kullanıcı için bir eğitim verisi olacaktır. Film sayısı 3.952 olduğundan bir eğitim verisi 3.952 rakamdan oluşan bir vektör olacaktır. Şekil 4'de gösterildiği gibi vektördeki her değer sırası ile bir filme verilen puanı temsil etmektedir. Her filmin her vektördeki yeri aynı olmalıdır. Kullanıcıların puan vermediği filmler için giriş değeri 0 olarak atanacaktır. Öneri sisteminin amacı bu değerleri tahmin etmektir. Veriler test ve eğitim setine bölünürken eğitim ve test seti eşit sayıdadır. Çünkü her kullanıcı hem test hem de eğitim setine dahil edilmiştir. Farklılık ise toplamdaki 0 değerleridir.

Authorized licensed use limited to: ULAKBIM UASL - MARMARA UNIVERSITY. Downloaded on December 28,2022 at 08:14:08 UTC from IEEE Xplore. Restrictions apply.



Şekil 4. Otomatik kodlayıcı giriş vektörü

Şekil 4'de gösterilen vektördeki N değeri veri setindeki toplam film sayısıdır.

Çalışmada oluşturulan modeller ve denenen optimizasyon algoritmaları Python programlama dilinde TensorFlow kullanılarak yazılmıştır [12]. TensorFlow kütüphanesi araştırmacılara makine öğrenmesi uygulamalarını kurabilecekleri ortam sağlayan açık kaynak kod platformudur [13]. Şekil 4'de gösterildiği gibi vektör haline getirilen veri TensorFlow'da kullanabilecek şekilde hazır hale getirilmiştir.

Giriş değeri 3.952 uzunluğunda bir vektörden oluşmaktadır. Ara katmanlar ve bu katmanlardaki aktivasyon fonksiyonları deneysel olarak belirlenmiştir. Örneğin ara katmanlardaki nöron sayısının artması başarı oranını artırmamış ve eğitim süresinin artmasına neden olmuştur. Aktivasyon fonksiyonu olarak relu, tanh ve leakly relu fonksiyonları denenmiş ve bu fonksiyonların başarısının çok daha düşük olduğu gözlemlenmiştir. Ara katmanlarda aktivasyon fonksiyonu olarak sigmoid fonksiyonu kullanılmıştır. Deneysel olarak belirlenen Otomatik Kodlayıcının katmanları sırası ile 80, 40, 80 nörondan oluşmaktadır. Çıkış katman sayısı ise film sayısı olan 3.952 uzunluğundadır. Bu işlemlerin yapılabilmesi için ağırlıkların ve bias sabitinin oluşturulması gerekmektedir.

Öneri sisteminin eğitilme algoritması aşağıdaki gibidir.

1) Eğitim verisi okunur.
2) Okunan veri her kullanıcı için vektör haline getirilir. Verilmeyen puanlar 0 olarak yazılır.
3) Ağırlıklar ve bias sabitleri rastgele değerler ile oluşturulur.
4) Eğitim verisi Otomatik Kodlayıcıyı ileri yönlü olarak eğitir.
5) Giriş katmanında 0 olan verilerin modelin öğrenmesine yanlış bir etkisinin olmaması için çıktı değerindeki bu veriler 0 yapılır. Ve hata fonksiyonu hesaplanır.
6) Geri yayılım algoritması ve seçilen optimizasyon algoritması ile ağırlıklar güncellenir.
7) Bu işlem seçilen devir sayısı kadar 4. madde ki işleme gidilerek tekrar yapılır.

Yukarıda belirtilen algoritma ile MovieLens veri seti için uygun olan optimizasyon algoritması bulunmaya çalışılmıştır. Uygun olan optimizasyon algoritmasının bulunması YSA'nın eğitim zamanının kısalmasını ve daha başarılı bir öneri sistemi tasarlanmasını sağlayacaktır.

Gradyan Alçalma – Gradient Descent (GA), Hızlı Gradyan Alçalma - Gradient Descent With Momentum (HGA), RMSProp ve Adam (Adaptive Momentum Optimization) algoritmaları veri seti üzerinde denenmiştir.

Modellerin eğitimi sırasında TensorFlow platformunun sağladığı hazır fonksiyonlar kullanılmıştır. Bu sayede geri yayılım algoritmasının yazımı, gradyanların hesaplanması ve optimizasyon algoritmalarının çalışması gibi işlemler TensorFlow'a bırakılmıştır.

C. Gradyan Alçalma Algoritması

Makine öğrenmesinde en yaygın kullanılan optimizasyon algoritması GA algoritmasıdır [14]. GA'da maliyet fonksiyonunun minimum olması için ağırlıklar ve sabitler güncellenir.

Maliyet fonksiyonunda belirtilen W ve b değerlerinin gradyanları hesaplanır.

α : Yapay Sinir Ağlarında öğrenme oranı

W : Yapay Sinir Ağlarında oluşturulan ağırlık

b : Yapay Sinir Ağlarında oluşturulan bias sabiti

$$dW = \frac{dJ}{dW} \ , \ db = \frac{dJ}{db} \qquad (2)$$

Hesaplanan gradyan değerlerine göre W ve b değerlerinin güncellenir. Eş. 3 ve Eş. 4 ile gösterilmiştir.

$$W = W - \alpha dW \qquad (3)$$

$$b = b - \alpha db \qquad (4)$$

Değerler güncellenirken rastgele bir öğrenme oranı belirlenir. Formülde gösterilen α öğrenme oranıdır.

Tasarlanan öneri sistemi GA algoritması ile üç farklı şekilde test edilmiş ve sonuçları Tablo 2'de gösterilmiştir.

Tablo II. Gradyan alçalma algoritması deneyler

	Devir sayısı	**Öğrenme oranı**	**Eğitim hatası**	**Test hatası**
Birinci deney	25	0.05	3,585	3,782
İkinci deney	25	3	1,096	2,135
Üçüncü deney	600	0,05	1,347	2,442

Authorized licensed use limited to: ULAKBIM UASL - MARMARA UNIVERSITY. Downloaded on December 28,2022 at 08:14:08 UTC from IEEE Xplore. Restrictions apply.

D. Hızlı Gradyan Alçalma Algoritması

Bu algoritmada bir önceki iterasyondan hesaplanan gradyanlar hesaba katılır.

$$v_{dw} = \beta v_{dw} + (1-\beta)dw \quad (5)$$

$$v_{db} = \beta v_{db} + (1-\beta)db \quad (6)$$

$$W = W - \alpha v_{dw} \quad (7)$$

$$b = b - \alpha v_{db} \quad (8)$$

Matematiksel olarak yukarıda formülleri verilmiştir [3]. 0 ile 1 arasında belirlenen β değeri hesaplanan gradyanların bir önce hesaplanan gradyanlara ne kadar yakın olacağını belirler.

Tasarlanan öneri sistemi HGA algoritması ile üç farklı şekilde test edilmiş ve sonuçları Tablo 3'de gösterilmiştir.

Tablo III. Hızlı gradyan alçalma algoritması deneyler

	Devir sayısı	**Öğrenme oranı**	**Eğitim hatası**	**Test hatası**
Birinci deney	25	0.05	2,623	3,338
İkinci deney	25	2,7	1,105	2,074
Üçüncü deney	150	0,05	1,094	2,100

E. RmsProp Algoritması

RmsProp algoritmasında bir önceki iterasyondan hesaplanan gradyanların kareleri hesaba katılır [15]. Aşağıdaki formülde görülen β değeri ile bir önceki iterasyondaki değere ne kadar yakın olacağı belirlenir.

$$s_{dw} = \beta s_{dw} + (1-\beta)dw^2 \quad (9)$$

$$s_{db} = \beta s_{db} + (1-\beta)db^2 \quad (10)$$

W ve b değeri ise aşağıdaki formüllerde gösterildiği gibi hesaplanır.

$$W = W - \alpha \frac{dw}{\sqrt{Sdw}} \quad (11)$$

$$b = b - \alpha \frac{db}{\sqrt{Sdb}} \quad (12)$$

Tasarlanan öneri sistemi RmsProp algoritması ile üç farklı şekilde test edilmiş ve sonuçları aşağıda verilmiştir.

F. Adam Algoritması

Adam algoritması 2015 yılında bulunmuştur [16]. Birçok problemin çözümünde çok başarılıdır [14]. Adam algoritması, HGA algoritmasının ve RmsProp algoritmasının birleşimi olarak düşünülebilir. Gradyanlar hesaplanırken aşağıdaki denklemlerde görüldüğü gibi, hem geçmiş gradyanların kareleri hem de gradyanların kendisi hesaba katılır [3]. Denklemlerde görülen β_1 değeri geçmiş gradyanın ne kadar hesaba katılacağını, β_2 değeri ise geçmiş gradyanın karesinin ne kadar hesaba katılacağını belirtir.

$$v_{dw} = \beta_1 v_{dw} + (1-\beta_1)dw \quad (13)$$

$$v_{db} = \beta_1 v_{db} + (1-\beta_1)db \quad (14)$$

$$s_{dw} = \beta_2 s_{dw} + (1-\beta_2)dw^2 \quad (15)$$

$$s_{db} = \beta_2 s_{db} + (1-\beta_2)db^2 \quad (16)$$

$$v_{dw} = \frac{v_{dw}}{1-\beta_1^t} \quad (17)$$

$$v_{db} = \frac{v_{db}}{1-\beta_1^t} \quad (18)$$

$$s_{dw} = \frac{s_{dw}}{1-\beta_2^t} \quad (19)$$

$$s_{db} = \frac{S_{db}}{1-\beta_2^t} \quad (20)$$

$$W = W - \alpha \frac{v_{dw}}{\sqrt{Sdw}+\gamma} \quad (21)$$

$$b = b - \alpha \frac{v_{db}}{\sqrt{Sdb}+\gamma} \quad (22)$$

Tasarlanan öneri sistemi Adam algoritması yapılan 25 devir ve 0,05 öğrenme oranı ile test edilmiş ve 0,995 eğitim hatası ve 1,363 test hatasına ulaşmıştır. Öğrenme oranının ya da devir sayısının değişmesi test puan hatasında fazla bir değişiklik oluşturmamıştır.

G. Veri artışının optimizasyon algoritması üzerindeki etkisi

Diğer algoritmalara göre daha başarılı görülen Adam optimizasyon algoritması kullanılarak veri artışının öneri sistemi üzerindeki performansı incelenmiştir. Veri bölünürken %10'luk test verisi sabit tutulmuş önce her kullanıcının sırası ile %50, %60, %70, %80 ve %90 oranında oylarının bulunduğu veri ile eğitilmiştir. Verilerin boşluk oranı ise Eş. 23'de gösterildiği gibi hesaplanmıştır.

$$\text{Boşluk Oranı} = 1 - \frac{\text{Sıfır olmayan puanlar}}{\text{Kullanıcı sayısı} * \text{Film sayısı}} \quad (23)$$

Boşluk oranı 0,979'dan 0,962'ye düşerken, test verisi üzerindeki hatanın 2,713 puandan 1,382 puana düştüğü görülmüştür. Tablo 4' de gösterilmiştir.

Tablo IV. Veri miktarındaki artışın etkisi

Eğitim veri setindeki oy sayısı	**Boşluk oranı**	**Eğitim hatası**	**Test hatası**
498.623	0,979	1,016	2,713
597.742	0,974	0,995	2,488
697.435	0,970	0,985	2,253
797.758	0,966	1,026	1,785
897.450	0,962	1,023	1,382

Authorized licensed use limited to: ULAKBIM UASL - MARMARA UNIVERSITY. Downloaded on December 28,2022 at 08:14:08 UTC from IEEE Xplore. Restrictions apply.

V. SONUÇLAR

Bu çalışmada Movilens 1M veri seti üzerinde öneri sistemi modeli oluşturulurken literatürdeki benzer çalışmalardan farklı olarak otomatik kodlayıcılar kullanılmış ve en uygun optimizasyon algoritmasının 25 devir 0.05 öğrenme oranı ve 1,363 test hatası ile Adam algoritması olduğu görülmüştür. En uygun optimizasyon algoritmasının bulunmasının önemi tasarlanan modelin eğitim süresinin azalmasını ve daha başarılı bir model tasarlanmasını sağlayacak olmasıdır.

Veri miktarındaki artış ve azalışın YSA üzerindeki etkisi incelendiğinde, daha fazla veri ile eğitilen YSA'ların kullanıcılar ve filmler arasındaki ilişkiyi daha iyi öğrenmeye başladığı sonucuna ulaşılmıştır. Boşluk oranı 0,979'dan 0,962'ye azalttığında 1-5 arasındaki puandan oluşan veri setinde hatanın 2,713 puandan 1,382 puana düşmesi veri toplamanın önemini göstermektedir.

Öneri sistemlerinin başarısını artırmak için tek bir model kullanmak yerine hibrid yaklaşımlar tercih edilebilir. Bu sayede tasarlanan öneri sisteminin bazı dezavantajları giderilebilir. Tasarlanan öneri sisteminde, sisteme yeni giren filmler oluşturulan vektörlerde yer alamayacağı için, öneri sistemine dahil olamazlar. Bu durumda soğuk başlangıç durumu oluşacaktır. Bu durum filme puan verilene kadar devam eder. Bu sorun içerik bazlı öneri metodu ile giderilebilir. Bu çalışmadaki öneri sistemi ile içerik bazlı öneri metodu birleştirilip hibrid bir öneri sistemi geliştirilebilir.

Veri setindeki ratings.dat dosyasında bulunan zaman bilgisi tasarlanan öneri sisteminde kullanılmamıştır. Kullanıcının hangi filme ne zaman puan verdiğini belirten bu bilgi ileriki çalışmalarda kullanılabilir. Bu veriler zamana göre sıralı hale getirilip kullanıcının bir sonra izlemek isteyeceği film, derin öğrenme yöntemi olan RNN'ler kullanılarak tahmin edilebilir.

KAYNAKÇA

[1] Strub, F., Mary, J., Collaborative filtering with stacked denoising autoencoders and sparse inputs, In NIPS workshop on machine learning for eCommerce, Montreal-Kanada, Kasım 2015.

[2] Aygün, R.C., Derin Öğrenme Yöntemleri ile Bilgisayar Ağlarinda Güvenliğe Yönelik Anormallik Tespiti, Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü, Yıldız Teknik Üniversitesi, İstanbul, 2017.

[3] Cousera. https://www.coursera.org/learn/neural-networks-deep-learning/. Erisim tarihi Mayıs 18, 2019

[4] Salakhutdinov, R., Mnih, A., Hinton, G., Restricted Boltzmann machines for collaborative filtering, In Proceedings of the 24th international conference on Machine learning, Oregon-Amerika Birleşik Devletleri, 791-798, Haziran 2007

[5] Sedhain, S., Menon, A. K., Sanner, S., Xie, L., Autorec: Autoencoders meet collaborative filtering, In Proceedings of the 24th International Conference on World Wide Web, Florence-İtalya, 111-112, Mayıs 2015

[6] Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D., Session-based recommendations with recurrent neural networks, San Juan – Porto Riko, 2015

[7] Akay, B., Kaynar O., Demirkoparan F., Derin Öğrenme Tabanlı Önerici Sistemler, Uluslararası Bilgisayar Bilimleri ve Mühendisliği Konferansı, Antalya-Türkiye, 645-648, 2017

[8] Zhu, Y., Li, H., Liao, Y., Wang, B., Guan, Z., Liu, H., Cai, D., What to Do Next: Modeling User Behaviors by Time-LSTM, In IJCAI, Melbourne- Australia, 3602-3608, Ağustos 2017,

[9] Santolaya, D.S., Using Recurrent Neural Networks to Predict Customer Behavior From Interaction Data, Yüksek Lisans Tez, Amsterdam Üniversitesi, Amsterdam, 7 Temmuz 2017.

[10] An MIT Press Book. https://www.deeplearningbook.org. Erisim tarihi Haziran 18, 2019

[11] Harper, F. M., Konstan, J. A., The movielens datasets: History and context. Acm transactions on interactive intelligent systems (tiis), 5(4), 19, 2016

[12] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Kudlur, M., Tensorflow: A system for large-scale machine learning, In 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16), Georgia - Amerika Birleşik Devletleri, 265-283, 2016

[13] TensorFlow. https://www.tensorflow.org/. Erisim tarihi Haziran 18, 2019

[14] Gündüz, H., Derin Öğrenme Yöntemleri ile Zaman Serisi Tahmini, Doktora Tezi, Fen Bilimleri Enstitüsü, İstanbul Teknik Üniversitesi, İstanbul, 2019.

[15] Department of Computer Science, University of Toronto. http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf. Erisim tarihi Haziran 18, 2019

[16] Kingma, D. P., Ba, J., Adam: A method for stochastic optimization, San Diego- Amerika Birleşik Devletleri, 2015

Authorized licensed use limited to: ULAKBIM UASL - MARMARA UNIVERSITY. Downloaded on December 28,2022 at 08:14:08 UTC from IEEE Xplore. Restrictions apply.