Yerele Duyarlı Kıyım Tabanlı Ölçeklenebilir İşbirlikçi Filtreleme
Locality Sensitive Hashing Based Scalable Collaborative Filtering

Ahmet Maruf Aytekin, Tevfik Aytekin
Bilgisayar Mühendisliği Bölümü
Bahçeşehir Üniversitesi
İstanbul, Türkiye
tevfik.aytekin@bahcesehir.edu.tr, ahmetmaruf.aytekin@stu.bahcesehir.edu.tr

***Özetçe* —Komşuluk tabanlı işbirlikçi filtreleme yöntemleri kolay uygulanabilir ve etkin yöntemler oldukları için tavsiye sistemlerinde yaygın olarak kullanılmaktadırlar. Bu yöntemlerin önemli bir sorunu büyüyen veri karşısında iyi ölçeklenememeleridir. Bu çalışmada yerele duyarlı kıyım yöntemi komşuluk tabanlı işbirlikçi filtreleme yönteminin ölçeklenebilirlik sorununun çözülmesinde uygulanmıştır. Yerele duyarlı kıyım yönteminin parametrelerinin geliştirilen tavsiye sisteminin ölçeklenebilirliğine ve doğruluğuna etkileri değerlendirilmiştir.**

Anahtar Kelimeler*—*tavsiye sistemleri, işbirlikçi filtreleme, yerele duyarlı kıyım, ölçeklenebilirlik.

***Abstract*—Neighborhood-based collaborative filtering methods are widely used in recommender systems because of their easy-to-implement and effective nature. One important drawback of these methods is that they do not scale well with increasing amounts of data. In this work we applied the locality sensitive hashing technique for solving the scalability problem of neighborhood-based collaborative filtering. We evaluate the effects of the parameters of locality sensitive hashing technique on the scalability and the accuracy of the developed recommender system.**

Keywords*—*recommender systems, collaborative filtering, locality sensitive hashing, scalability.

I. GİRİŞ

Tavsiye sistemleri kullanıcıların geçmiş hareketlerini analiz ederek kullanıcılara ilgilerini çekebilecek öğeleri bulmaları konusunda yardımcı olan araçlardır [1]. Tavsiye sistemleri son yıllarda çok geniş alanlarda kullanılmaya ve farklı problemlere uygulanmaya başlanmıştır. En popüler alanlar film, müzik, kitap, makale tavsiyesi gibi ürün tavsiyeleri olmakla beraber, sosyal medya, finans veya arama motorlarında tavsiye sistemleri kullanılmaktadır [2, p.1-35].

Tavsiye yöntemlerinin bir alt kolu olan komşuluk tabanlı işbirlikçi filtreleme (KTİF) yöntemleri genel olarak kolay anlaşır yöntemlerdir ve tahminleme sonuçları oldukça iyi düzeydedir [1, 3, 4]. Ancak bu algoritmalar bir çok kanaldan üretilen büyük miktarlarda verilerin analizinde veri miktarına bağlı olarak ölçeklenebilirlik konusunda yetersiz kalmaktadırlar. Ölçeklenebilirlik probleminin nedeni bir kullanıcıya tavsiyede bulunmak için tüm komşuların analiz edilmesinin gerekmesi ve bu durumun zaman karmaşıklığını karesel artırmasıdır [3]. Araştırmacılar ölçeklenebilirlik problemini çözmek için sınıflandırma, kümeleme, matris ayrıştırma gibi yaklaşımlar yanında dağıtık algoritmalar ve genel metin arama gibi yöntemler kullanmışlardır [5].

Bu çalışmada yerele duyarlı kıyım (YDK) (locality sensitive hashing) yöntemi KTİF algoritmaları ile birlikte kullanılarak ölçeklenebilir bir KTİF yöntemi önerilmektedir. YDK bazı tavsiye sistemlerinde kullanılmıştır [6] ancak bildiğimiz kadarıyla YDK tekniğinin tavsiye sistemlerinde kullanılmasının sistematik olarak değerlendirilmesi konusunda henüz bir çalışma yapılmamıştır. Bu çalışmada YDK tekniğinin tavsiye sistemlerinde sistematik olarak değerlendirilmesi yanında, önceki çalışmalardan farklı olarak YDK parametrelerinin tavsiye sistemlerine etkilerini de değerlendirdik.

Çalışmanın genel planı şu şekildedir: Bölüm II'de KTİF yöntemi, Bölüm III'de YDK yöntemi ve Bölüm IV'te önerilen ölçeklenebilir KTİF algoritması tanıtılmaktadır. Deneysel analizler ve sonuçlar sırasıyla Bölüm V ve VI'da sunulmaktadır.

II. İŞBİRLİKÇİ FİLTRELEME

KTİF yöntemlerinde zaman karmaşıklığı, model oluşturma ve bu modeli kullanarak tavsiyede bulunmak için gereken zamanların toplamıdır [7]. Model oluşturma adımında en yakın komşuları bulabilmek için sistemdeki tüm kullanıcıların veya öğelerin birbirlerine benzerliklerinin hesaplanması gerekmektedir. Bu durumda zaman karmaşıklığı kullanıcı veya öğe sayısına bağlı olarak karesel artmaktadır [1]. Bir çok uygulamada bu sorun, benzerlik matrislerinin önceden hesaplanmasıyla çözülmeye çalışılmıştır [7], ancak bu yöntem sistemin gerçek zamanda oluşan veriler üzerinde analiz yapmasını kısıtlamakla beraber veri miktarındaki artışa bağlı olarak ölçeklenebilir değildir.

Bu çalışma Merkezi Kayıt Kuruluşu (MKK) Ar-Ge Merkezi tarafından desteklenmektedir.

978-1-4799-4874-1/14/$31.00

Bu sorunun çözümü için YDK yöntemi [8, ch.3] kullanılarak KTİF algoritmalarında en yakın k komşuyu bulma adımında sistemdeki tüm aday kullanıcı veya öğeleri tek tek karşılaştırmadan benzerlik oranları yüksek olasılıkta olan yakın komşu adayları ile karşılaştırma yapan ölçeklenebilir bir KTİF yöntemi değerlendirilmiştir.

III. YERELE DUYARLI KIYIM

YDK yönteminde kullanıcı veya öğeler birçok defa farklı kıyım fonksiyonlarından geçirilerek her bir kullanıcı veya öğe için kıyım anahtarı oluşturulur. Kıyım anahtarlarından herhangi biri aynı olan kullanıcı veya öğeler aynı sepetlerde toplanırlar. Benzer öğe veya kullanıcıların herhangi bir kıyım anahtarının aynı olma olasılığı benzer olmayan öğelere göre çok daha yüksektir. Herhangi bir kıyım fonksiyonundan geçirme adımında aynı sepete düşen öğeler aday-çift kümesi olarak tanımlanır ve en yakın komşu hesaplamasında bu küme kullanılır. Benzerlik oranı düşük veya benzer olmayan öğelerin aday-çift kümesine girme olasılığı çok düşüktür. Benzerlik oranı düşük olup aynı sepete düşen öğeler yanlış pozitifler olarak adlandırılır. Bu yöntemde benzerlik oranı yüksek olan öğelerin aynı sepete düşme olasılığı çok yüksektir, ancak farklı sepete düşen çok küçük miktarda öğe olabilir. Bu öğeler yanlış negatifler olarak adlandırılır [8, ch.3].

İki öğeyi alıp kıyım fonksiyonlarından geçirip sonuca göre aday-çift olup olmadıklarına karar veren fonksiyon grubu YDK fonksiyon ailesi ($\mathcal{F}$) olarak adlandırılır [8, ch.3] ve şöyle tanımlanır. x ve y R skor veri kümesinden oluşturulan R^n vektör kümesinde yer alan iki farklı skor vektörü, d bir uzaklık ölçüsü ve $d_1 < d_2$ iki farklı uzaklık olmak üzere, $\mathcal{F}$'de yer alan her bir f fonksiyonu için:

1) Eğer $d(x, y) = d_1$ ise, $f(x) = f(y)$ olma olasılığı en az p_1'dir.
2) Eğer $d(x, y) = d_2$ ise, $f(x) = f(y)$ olma olasılığı en fazla p_2'dir.

buradaki YDK yönteminin kullanılır olması için $p_1 > p_2$ olmalıdır.

Bu durumda $\mathcal{F}$ (d_1, d_2, p_1, p_2)-*duyarlı*'dır denir.

Bu çalışmada kosinüs benzerliği için bir $\mathcal{F}$ kıyım fonksiyonu ailesi tanımlanmıştır [9]. $\vec{r}$ n-boyutlu normal dağılım kümesinden seçilen bir vektör olmak üzere rasgele hiper-düzlem tabanlı kıyım fonksiyonu $h_{\vec{r}}$ şu şekilde tanımlanır:

$$h_{\vec{r}}(\vec{u}) = \begin{cases} 1 & \text{if } \vec{r} \cdot \vec{u} \geq 0 \\ 0 & \text{if } \vec{r} \cdot \vec{u} < 0 \end{cases}$$

Bu durumda vektörler $\vec{u}$ ve $\vec{v}$ için,

$$Pr[h_{\vec{r}}(\vec{u}) = h_{\vec{r}}(\vec{v})] = 1 - \frac{\theta(\vec{u}, \vec{v})}{\pi},$$

burada $\vec{u}$, kullanıcı u'nun skor vektörü, $\vec{v}$, kullanıcı v'nin skor vektörü olmak üzere $Pr[h_{\vec{r}}(\vec{u}) = h_{\vec{r}}(\vec{v})]$ kullanıcı u ve v'nin bir aday-çift olma (aynı sepete düşme) olasılığıdır.

Yukarıda bahsedilen rasgele hiper-düzlem tabanlı kıyım fonksiyonunu kullanarak kosinüs benzerlik hesaplaması için

	Alan	Süre	
		Model Oluşturma	Sorgulama
kullanıcı-tabanlı	$O(\|U\|^2)$	$O(\|U\|^2 p)$	$O(\|I\|k)$
öğe-tabanlı	$O(\|I\|^2)$	$O(\|I\|^2 q)$	$O(\|I\|k)$
KT-YDK	$O(L\|U\|)$	$O(L\|U\|\kappa t_g)$	$O(L\|C\|\kappa t_g)$
ÖT-YDK	$O(L\|I\|)$	$O(L\|I\|\kappa t_g)$	$O(L\|C\|\kappa t_g)$

Tablo I: Kullanıcı-tabanlı ve öğe-tabanlı işbirlikçi filtreleme yöntemleri ile ölçeklenebilir işbirlikçi filtreleme yöntemi alan ve zaman karmaşıklığı.

$\mathcal{F}$ kıyım fonsiyonu ailesi elde edilir. Bu şekilde, $\theta = \cos^{-1} \frac{\vec{u} \cdot \vec{v}}{\|\vec{u}\| \|\vec{v}\|}$ olmak üzere u ve v'nin aday çift olma olasılığı

$$Pr[h(u) = h(v)] = 1 - \frac{\theta}{\pi} \tag{1}$$

olan bir YDK şeması elde edilir.

IV. ÖLÇEKLENEBİLİR İŞBİRLİKÇİ FİLTRELEME

KTİF yöntemlerinde iki kullanıcı veya öğenin benzerliği skor vektörlerinin benzerliği olarak hesaplanır [2, p1-35]. İki vektörün benzerlik hesaplamasında kosinüs tabanlı benzerlik hesaplaması kullanılmıştır. KTİF algoritmalarında en maliyetli adım en yakın k komşuyu hesaplama adımı olan benzerlik hesaplama [7] adımıdır. Bu işlem en yakın komşu arama problemi olarak değerlendirilip YDK yöntemi uygulanarak tüm komşular analiz edilmeden en yakın k komşu hesaplanır. İlk olarak algoritma, $\mathcal{G}$ kıyım fonksiyon ailesinden rasgele seçtiği κ adet g kıyım fonksiyonu birleştirerek $g(p)$ fonksiyonunu elde eder. Bu fonksiyon $g(p) = [h_1(p), ..., h_\kappa(p)]$ olarak gösterilir. Algoritma bu işlemi L adet kıyım tablosu için yaparak herbir kıyım tablosu için rasgele seçilmiş farklı bir kıyım fonksiyonu $g(p)$ oluşturmuş olur. Kıyım fonksiyonları oluşturulduktan sonra model oluşturulur. Model oluşturma adımında L kıyım tablolarından her bir t tablosu için şu işlemler yapılır. U kullanıcı kümesinde yer alan her bir kullanıcı u'nun skor vektörü, t tablosunun kıyım fonksiyonu $g(t)$'den geçirilerek u kullanıcısının kıyım anahtarı key_{ug} oluşturulur. Hesaplanan anahtar key_{ug} kullanılarak kullanıcı t tablosuna eklenir. Bu işlem tüm kullanıcılar için tamamlandığında benzer kullanıcıların (potansiyel yakın komşular olarak) en az bir t tablosunda aynı kıyım anahtarına sahip olup aynı sepette yer almaları beklenmektedir. Bu işlem için gerekli olan maksimum bellek miktarı $O(L|U|)$ olmakla beraber model oluşturma adımı zaman karmaşıklığı Tablo I'de görüldüğü gibi kullanıcı ve öğe sayısına bağlı olarak doğrusal büyüyen bir karmaşıklığa indirgenmiş olmaktadır. Burada t_g, bir kıyım fonksiyonunu değerlendirme için gereken süreyi, KT-YDK, yerele duyarlı kıyım ile kullanıcı-tabanlı ölçeklenebilir işbirlikçi filtreleme yöntemini ve ÖT-YDK, yerele duyarlı kıyım ile öğe-tabanlı ölçeklenebilir işbirlikçi filtreleme yöntemini göstermektedir. Ayrıca öğe kümesi I, bir kullanıcı için maksimum skor sayısı p, bir öğe için maksimum skor sayısı q, ve skor tahmin hesabında kullanılan maksimum en yakın komşu sayısı k ile gösterilmektedir.

Bu sistemde bir v kullanıcısının yakın komşu adaylarını sorgulama şöyle yapılır. L kıyım tablolarından her bir t tablosu için kullanıcı v'nin skor vektörü kullanılarak kıyım anahtarı key_{vg} üretilir. Aynı anahtara sahip t tablosunda yer alan tüm kullanıcılar aday-çiftler olarak değerlendirilir. Tüm tablolardan

Veri Kümesi	Kullanıcı Sayısı	Öğe Sayısı	Toplam Skor	Seyreklik
MovieLens	1779	3569	300000	0.0472
Yahoo! Music	15400	1000	311704	0.0202

Tablo II: Veri kümeleri.

gelen aday-çiftler birleştirilerek aday-çift kümesi (C) oluşturulur. C kümesi oluşturulduktan sonra KTİF algoritmaları kullanılarak v kullanıcısı için tahminleme yapılır.

Bu adımda KTİF algoritmaları en yakın k komşu kullanıcı veya öğeyi hesaplayabilmek için sistemdeki tüm kullanıcı veya öğeleri analiz etmeden sadece C kümesini analiz eder. Bu işlemin hesaplama karmaşıklığı $O(L|C|)$ olur. Böylece doğrusal büyüyen hesaplama karmaşıklığına sahip, büyük veri miktarları karşısında ölçeklenebilir bir algoritma elde edilmiş olur. KTİF yöntemleri ve YDK yöntemi ile elde edilen ölçeklenebilir algoritmanın bellek kullanımı ve hesaplama karmaşıklığı Tablo I'de gösterilmektedir.

V. DENEYSEL ANALİZLER

KTİF yöntemi ile bu çalışmada önerilen ölçeklenebilir işbirlikçi filtreleme (ÖİF) yöntemi etkinlik ve doğruluk yönlerinden karşılaştırılıp, önerilen yöntemin etkin kullanımı için farklı konfigürasyon parametrelerinin etkileri gerçek veri kümeleri (Tablo II) üzerinde değerlendirilmiştir. Deneylerde çapraz geçerleme tekniği olarak rastgele örnekleme yöntemi kullanılmıştır. Veri kümelerinin %5'i rastgele seçilerek test kümesi oluşturulmuş kalan %95'i öğrenme kümesi olarak kullanılmıştır. Deneyler üç defa tekrar edilip sonuçların ortalaması alınmıştır.

İsbirlikçi filtreleme yöntemlerinde tahminleme yapılırken kullanılan önemli bir parametre en yakın komşu sayısı k'dır. Bu çalışmada yapılan deneylerde k değeri 20 olarak kullanılmıştır [4].

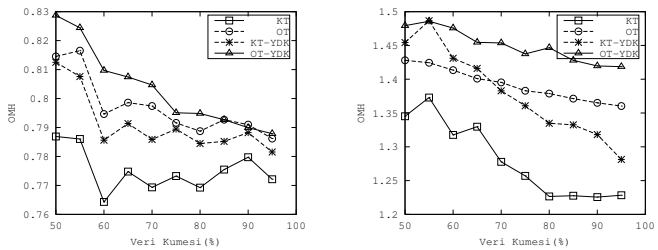
ÖİF yönteminin konfigüre edilmesi gereken iki temel parametresi kıyım fonksiyon sayısı κ ve kıyım tablo sayısı L'dir. Her bir veri seti için optimum değerleri belirlemek amacıyla bu parametreler 1'den 25'e kadar değişik değerlerle denenerek 625 farklı model oluşturulmuştur. KTİF yöntemiyle karşılaştırma yapılırken bu optimum değerler kullanılmıştır.

Tahmin doğruluğunu değerlendirmede en yaygın kullanılan yöntemler Ortalama Mutlak Hata (OMH) ve Karekök Ortalama Karesel Hata (KOKH) yöntemleridir [10]. Bu yöntemlerden OMH yöntemi, hata oranını daha kolay ifade etme yeteneginden ötürü tercih edilmiştir. OMH hesaplamasında Formül 2 kullanılmıştır [1].

$$OMH(f) = \frac{1}{\mid R_{test} \mid} \sum_{r_{ui} \in R_{test}} \mid f(u,i) - r_{ui} \mid, \qquad (2)$$

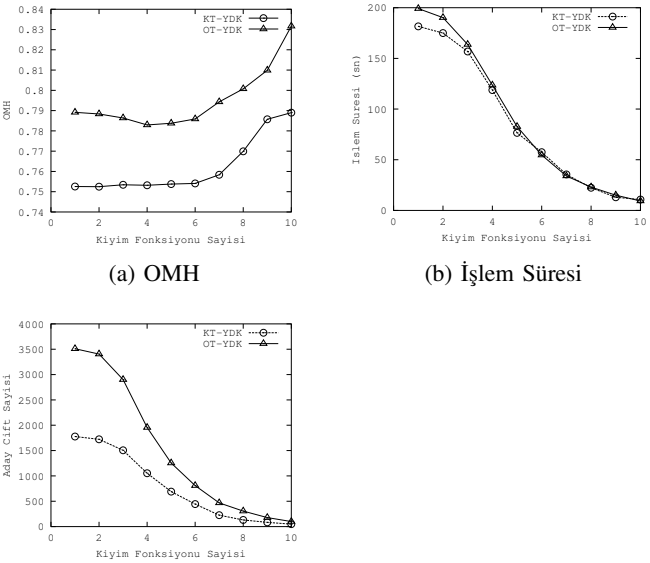
burada $f(u,i)$, kullanıcı u'nun öğe i için tahmin edilen skoru ve r_{ui}, kullanıcı u'nun öğe i için gerçek skorudur.

Denemelerden ilki KTİF yöntemleri ve önerilen ÖİF yönteminin seyreklik oranı karşısındaki davranışlarını analiz etmek için yapılmıştır. Öğrenme kümeleri, R veri kümesinden "basit rastgele örnekleme" yöntemiyle seçilip seyreklik oranı değişen veri kümeleri ile deneyler yapılmıştır. Şekil 1'de



(a) MovieLens (b) Yahoo! Music

Şekil 1: OMH'nın sistemdeki veri miktarına göre değişimi.



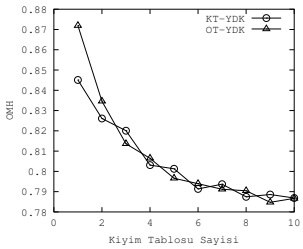
(a) OMH (b) İşlem Süresi

(c) Aday Çift Sayısı

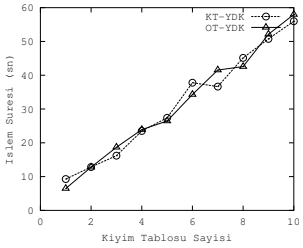
Şekil 2: OMH, işlem süresi ve aday çift sayısı'nın kıyım fonksiyon sayısı (κ)' na göre değişimi. Kıyım tablo sayısı $L = 10$ alınmıştır (MovieLens).

görüldüğü üzere tüm yöntemlerde OMH değerleri öğrenme veri miktarı artışından olumlu yönde etkilenmiştir. Bu çalışma açısından önemli olan ise veri miktarı arttıkça ÖİF yönteminin OMH değerlerinin KTİF yöntemlerine yaklaşmasıdır. Geliştirilen yöntem büyük verilerin işlenmesinde kullanılmak için tasarlandığından bu olumlu bir sonuçtur.

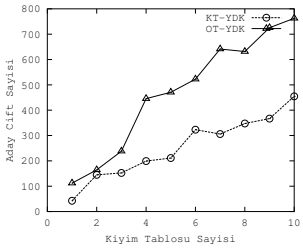
Diğer bir deney ÖİF yönteminin parametreleri olan kıyım fonksiyon sayısı κ ve kıyım tablo sayısı L'nın OMH üzerindeki etkilerini görmek için yapılmıştır. Buradaki bulgular dikkat çekicidir. Şekil 2a'da görüldüğü üzere düşük κ değerleri için OMH sabit kalırken belirli bir noktadan sonra OMH'de hızlı bir artış olmaktadır. Bunun nedeni κ değeri arttıkça bir süre sonra sistemdeki sepet sayısı kullanıcı (veya öğe) sayısına yaklaşması ve dolayısıyla aday-çift kümesi C'nin büyüklüğünün sıfıra doğru düşmesidir (Şekil 2c). Bu durumun tersi olarak L'deki artışın OMH'yi olumlu yönde etkilediği (Şekil 3a) görülmüştür. Bunun nedeni ise L'nin artışına bağlı olarak C'nin büyümesi (Şekil 3c) ve tüm doğru pozitiflerin yakalanmasıdır.



(a) OMH



(b) İşlem Süresi



(c) Aday Çift Sayısı

Şekil 3: OMH, işlem süresi ve aday çift sayısı'nın kıyım tablo sayısı (L)' na göre değişimi. Kıyım fonksiyon sayısı $\kappa = 6$ alınmıştır (MovieLens)

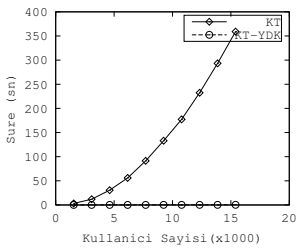
ÖİF yönteminde κ parametresinin artmasının OMH üzerindeki etkisi olumsuz olmasına karşın model oluşturma süresi üzerindeki etkisi çok olumludur (Şekil 2b). Bunun nedeni κ büyüdükçe C kümesinin giderek küçülmesi (Şekil 2c) ve dolayısıyla C içindeki yakın komşuları bulma süresinin azalmasıdır. L sayısındaki artışın ise OMH üzerinde olumlu etkisi (Şekil 3a) olmasına rağmen model oluşturma süresi üzerinde olumsuz etkisi vardır (Şekil 3b). Bunun nedeni ise L'deki artışın C kümesinin büyümesine yol açmasıdır (Şekil 3c). κ ve L parametrelerinin uygulamanın ihtiyaçlarına göre optimum OMH ve işlem sürelerini sağlayacak şekilde konfigüre edilmesi gerektiği gözlemlenmiştir.

Tablo I'de görüldüğü gibi KTİF yöntemlerinde model oluşturma zaman karmaşıklığı kullanıcı veya öğe sayısına bağlı olarak karesel artmasına karşın ÖİF yönteminde hesaplama karmaşıklığı doğrusal artan bir karmaşıklığa indirgenmiştir. Yahoo! Music veri kümesi için, kullanıcı ve öğe sayılarındaki artışa bağlı olarak, model oluşturma süreleri Şekil 4'de gösterilmiştir. MovieLens veri kümesi için de benzer sonuçlar alınmıştır. Şekil 4'de görüldüğü üzere KTİF yönteminde model oluşturma süresi hızla artarken, ÖİF yönteminde neredeyse sabit kalmıştır. Bu da yöntemin çok büyük veriler için ölçeklenebilir olduğunu göstermektedir.

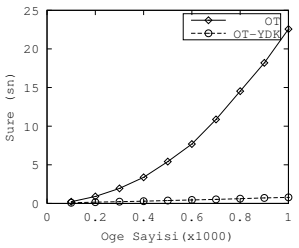
Önerilen ÖİF yönteminde model oluşturma adımı bir defa yapılıp, sisteme yeni bir kullanıcı veya öğe eklenmesi durumunda bu öğe veya kullanıcı için kıyım anahtarı oluşturulur ve kıyım tabloları güncellenir. Bu şekilde yeniden model oluşturmaya gerek kalmadan bu kullanıcı veya öğe sisteme girmiş olur ve hesaplamalarda kullanılabilir.

VI. VARGILAR

Bu çalışmada sistematik bir şekilde YDK tekniğinin KTİF algoritmalarının ölçeklenebilirlik probleminin çözümünde



(a) Kullanıcı-tabanlı



(b) Öğe-tabanlı

Şekil 4: Benzerlik matrisi (model) oluşturma işlem süreleri (Yahoo! Music).

uygulanması değerlendirilmiştir. YDK tekniğini KTİF yöntemleri ile birleştirerek ölçeklenebilir, zaman karmaşıklığı doğrusal olarak artan bir yöntem (ÖİF) elde edildiği gösterilmiştir.

ÖİF yönteminde YDK parametrelerinin sonuçlar üzerinde çok etkili olduğu gözlenmiştir. YDK modelinin, sistemden beklentilere bağlı olarak, OMH ve işlem süresi arasında dengeli sonuçları verebilecek şekilde konfigüre edilmesi gerektiği gözlemlenmiştir.

YDK yönteminin tavsiye sistemlerindeki çeşitlilik ve yenilik gibi önemli metrikler üzerindeki etkisinin değerlendirilmesi ve YDK yönteminin kümeleme yöntemleri ile karşılaştırılması önümüzdeki planlarımız arasında yer almaktadır.

KAYNAKÇA

[1] C. Desrosiers and G. Karypis, "A comprehensive survey of neighborhood-based recommendation methods," 2011.

[2] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds., *Recommender Systems Handbook*. Springer, 2011.

[3] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. Riedl, "Evaluating collaborative filtering recommender systems," 2004.

[4] J. L. Herlocker, J. A. Konstan, and J. Riedl, "An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms," *Inf. Retr.*, vol. 5, no. 4, pp. 287–310, 2002.

[5] H. Yu, C. Hsieh, S. Si, and I. S. Dhillon, "Scalable coordinate descent approaches to parallel matrix factorization for recommender systems," 2012.

[6] A. Das, M. Datar, A. Garg, and S. Rajaram, "Google news personalization: scalable online collaborative filtering," 2007.

[7] M. Deshpande and G. Karypis, "Item-based top-*N* recommendation algorithms," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 143–177, 2004.

[8] A. Rajaraman and J. D. Ullman, *Mining of Massive Datasets*. New York, NY, USA: Cambridge University Press, 2011.

[9] M. Charikar, "Similarity estimation techniques from rounding algorithms," J. H. Reif, Ed. ACM, 2002, pp. 380–388.

[10] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," 1999.