İNGİLİZCE BİLGİ ERİŞİMİ VERİ KÜMELERİNDE LATİN DIŞI ALFABELERLE YAZILMIŞ İÇERİĞİN ANALİZİ

Yüksek Lisans Tezi

Ahmet ALKILINÇ

Eskişehir, 2019

İNGİLİZCE BİLGİ ERİŞİMİ VERİ KÜMELERİNDE LATİN DIŞI ALFABELERLE YAZILMIŞ İÇERİĞİN ANALİZİ

Ahmet ALKILINÇ

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı
Danışman : Dr. Öğr. Üyesi. Ahmet ARSLAN

Eskişehir
Eskişehir Teknik Üniversitesi
Lisansüstü Eğitim Enstitüsü
Mayıs, 2019

Bu tez çalışması BAP Komisyonu tarafından kabul edilen 1709F516 no.lu proje kapsamında desteklenmiştir.

JÜRİ VE ENSTİTÜ ONAYI

Ahmet ALKILINÇ'ın **"İngilizce Bilgi Erişimi Veri Kümelerinde Latin Dışı Alfabelerle Yazılmış İçeriğin Analizi"** başlıklı tezi 31/05/2019 tarihinde aşağıdaki jüri tarafından değerlendirilerek Eskişehir Teknik Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliği'nin ilgili maddeleri uyarınca, Bilgisayar Mühendisliği Anabilim dalında Yüksek Lisans tezi olarak kabul edilmiştir.

Jüri Üyeleri	**Unvanı Adı Soyadı**	**İmza**
Üye (Tez Danışmanı)	**: Dr. Öğr. Üyesi Ahmet ARSLAN**	.....
Üye	**: Doç. Dr. Bekir Taner DİNÇER**	.....
Üye	**: Dr. Öğr. Üyesi Alper Kürşat UYSAL**	.....

.....

Enstitü Müdürü

ÖZET

İNGİLİZCE BİLGİ ERİŞİMİ VERİ KÜMELERİNDE LATİN DIŞI ALFABELERLE YAZILMIŞ İÇERİĞİN ANALİZİ

Ahmet ALKILINÇ
Bilgisayar Mühendisliği Anabilim Dalı
Eskişehir Teknik Üniversitesi, Lisansüstü Eğitim Enstitüsü, Mayıs, 2019
Danışman: Dr. Öğr. Üyesi Ahmet ARSLAN

Yüzyıllardır insanlar arşivleme ve bilgi bulmanın öneminin farkında olmuşlardır. Bilgisayarların gelişiyle birlikte, büyük miktarda bilgiyi depolamak mümkün olmuştur ve bu tür koleksiyonlardan yararlı bilgiler bulmak bir gereklilik haline gelmiştir. Bilgi erişimi alanı 1950'lerde bu gereklilikten doğmuştur. Bilgi erişimi kullanıcıların ihtiyaç duydukları bilgi ile ilgili kaynakları büyük koleksiyonlardan bulma işlemidir. Bilgi erişim sistemlerinin başarısı bulunan dokümanların ne kadarının kullanıcının aradığı bilgi ile ilgili olmasıyla doğru orantılıdır. Bilgi erişim sistemlerinin başarımını ölçmek, performansları karşılaştırmak için yıllık olarak Text Retrieval Conference düzenlenmektedir. Bu organizasyon tarafından standart veri setleri oluşturulup yayınlanmaktadır. Bu çalışmada İnternet'ten toplanan ve İngilizce Web sayfalarından oluşan ClueWeb09, ClueWeb12 ve Gov2 veri setleri kullanılmıştır. Her ne kadar bu Web sayfalarındaki kelimelerin çoğu Latin alfabesiyle yazılmış olsa da veri setleri ayrıca Latin dışı alfabelerde (Japon, Kiril, Yunan, Arap, vb.) yazılmış kelimeleri de içermektedir. Ayrıca, bu veri kümeleriyle ilişkilendirilmiş olan sorgu kümeleri, tamamen Latin alfabesinde yazılmış sözcüklerden oluşmaktadır. Bu kapsamda, bu tezin amacı, Latin dışı alfabelerle yazılmış kelimelerin İngilizce veri setleri üzerindeki dağılımı incelemek ve Latin dışı kelimelerin indekse dahil etmenin veya hariç tutmanın bilgi erişim başarımı üzerindeki etkisini araştırmaktır.

Anahtar Sözcükler: Bilgi Erişimi, ClueWeb12, Tokenization, Latin Dışı Alfabeler

ABSTRACT

ANALYSIS OF NON-LATIN CONTENT ON THE ENGLISH INFORMATION RETRIEVAL DATASETS

Ahmet ALKILINÇ

Computer Engineering Department

Eskişehir Technical University, Institute of Graduate Programs, May, 2019

Supervisor: Assist. Prof. Dr. Ahmet ARSLAN

For centuries people have been aware of the importance of archiving and finding information. With the advent of computers, it is possible to store large amounts of information and finding useful information from such collections became a necessity. The field of Information Retrieval emerged from this requirement in the 1950s. Information retrieval is the process of finding resources that are relevant to an information the users need from large collections. The success of information retrieval systems is directly proportional to the fact that the documents found are related to the information the user is looking for. The Text Retrieval Conference is organized annually to measure the success of information retrieval systems and to compare their performances. Standard data sets are created and published by this organization. In this study ClueWeb09, ClueWeb12 and Gov2 data sets, which consist of English web pages collected from the Internet, are used. Although the majority of the words in these web pages are written in the Latin alphabet, datasets also include words written in non-Latin alphabets (Japanese, Cyrillic, Greek, Arabic, etc). Moreover, the query sets associated with these datasets consist of words written entirely in Latin alphabet. In this context, the objective of this thesis is to examine the distribution of words written in non-Latin alphabets on English data sets and to investigate the effect of including or excluding non-Latin words in index on information retrieval effectiveness.

Keywords: Information Retrieval, ClueWeb12, Tokenization, non-Latin Scripts

31/05/2019

TEŞEKKÜR

Çalışmalarım sırasında bilgisi ve tecrübesiyle bana yol gösteren, değerli bilgilerini benimle paylaşan, kendisine ne zaman danışsam bana kıymetli zamanını ayırıp sabırla ve büyük bir ilgiyle bana faydalı olabilmek için elinden gelenden fazlasını sunan, gelecekteki mesleki hayatımda da bana verdiği değerli bilgilerden faydalanacağımı düşündüğüm, değerli danışman hocam Dr. Öğr. Üyesi Ahmet ARSLAN'a teşekkürü bir borç biliyor ve şükranlarımı sunuyorum.

Ahmet ALKILINÇ

Mayıs, 2019

31/05/2019

ETİK İLKE VE KURALLARA UYGUNLUK BEYANNAMESİ

Bu tezin bana ait, özgün bir çalışma olduğunu; çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalarında bilimsel etik ilke ve kurallara uygun davrandığımı; bu çalışma kapsamında elde edilemeyen tüm veri ve bilgiler için kaynak gösterdiğimi ve bu kaynaklara kaynakçada yer verdiğimi; bu çalışmanın Eskişehir Teknik Üniversitesi tarafından kullanılan "bilimsel intihal tespit programı" ile tarandığını ve hiçbir şekilde "intihal içermediğini" beyan ederim. Herhangi bir zamanda, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun saptanması durumunda, ortaya çıkacak tüm ahlaki ve hukuki sonuçlara razı olduğumu bildiririm.

.....

Ahmet ALKILINÇ

İÇİNDEKİLER

Sayfa

Sayfa

TABLOLAR DİZİNİ

Sayfa

ŞEKİLLER DİZİNİ

Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

†	Dagger
λ	Lambda
ASCII	American Standard Code for Information Interchange (Bilgi Değişimi İçin Amerikan Standart Kodlama Sistemi)
HTML	Hyper Text Markup Language (Hiper Metin İşaret Dili)
ICU	International Components for Unicode (Unicode için Uluslararası Bileşenler)
IP	Internet Protocol (İnternet Protokolü)
MAP	Mean Average Precision (Ortalama Hassasiyet)
NDCG	Normalized Discounted Cumulative Gain
NTCIR	NII Testbeds and Community for Information Access Research (Bilgi Erişim Araştırması için NII Testleri ve Topluluğu)
URL	Uniform Resource Locator (Standart Kaynak Bulucu)
UTF	Unicode Transformation Format (Unicode Dönüşüm Formatı)
T.C.	Türkiye Cumhuriyeti
TREC	Text REtrieval Conference

1. GİRİŞ

Bilgi ve iletişim teknolojilerindeki ilerlemeler, elektronik cihazların ve sosyal ağların kullanımının artması ile dijital ortamdaki veri miktarı her geçen gün artmaya devam etmektedir. Bazı tahminlere göre, dünya çapında üretilen veri miktarı iki yılda bir ikiye katlanmaktadır. Ayrıca İnternet kullanımının yaygınlaşması da Web ortamına gönderilen farklı dil ve alfabelerde yazılmış dijital içerik miktarını sürekli arttırmaktadır. Oluşan bu devasa boyut ve çeşitlikteki veri miktarıyla birlikte bazı ihtiyaçlar ve buna bağlı olarak da farklı çalışma alanları ortaya çıkmaktadır. Bunlardan bir tanesi çok dilli bilgi alma, yazım denetimi vb. uygulamalar için dokümanlarda geçen dillerin belirlenmesi ihtiyacıdır. Bu ihtiyacı gidermek için Doğal Dil İşleme'nin bir alt çalışma alanı olan Dil tanımlama (Language identification) ortaya çıkmıştır. Dil tanımlama, belgenin içeriğine göre bir belgede bulunan dilleri otomatik olarak tespit etme işlemidir [32]. Dil tanımlama, öncesinde dil tanımlaması gerektiren makine çevirisi, bilgi alma, özetleme ve soru cevaplama sitemleri gibi doğal dil işleme uygulamalarında kullanılan geniş bir araştırma alanıdır.

Oluşan bu devasa veri yığını içerisinden doğan diğer bir ihtiyaç ise bu veriler arasından istenilen doğru bilgiye nasıl ulaşılacağıdır. Bu noktada devreye Bilgi Erişim sistemleri girmektedir. Bilgi Erişimi, ilişkisel veri tabanlarında, belgelerde, metinlerde, multimedya dosyalarında ve Web sitelerinde bilgi aramada kullanılan bir bilimdir [20]. Bilgi Erişimi, metin belgeleri üzerinde kullanıcıların bilgi ihtiyaçlarına cevap vermeyi amaçlamaktadır. Bilgi Erişim uygulamaları, büyük belgelerden bilgi çıkarma, dijital kütüphanelerde arama, bilgi filtreleme, spam filtreleme, görüntülerden nesne çıkarma, otomatik özetleme, belge sınıflandırma ve Web araması gibi konuları içermektedir.

Bilgi Erişimi indeksleme ve arama olmak üzere iki kısımdan oluşur. İndeksleme aşamasında belgeler bir ön işlemden geçirilir. Bu ön işlem sırasında tokenization, küçük harfe çevirme, gövdeleme (stemming) gibi işlemler yapılır. Bu ön işlem adımının ilk adımı olan tokenization az dikkat çekmiş bir çalışma alanıdır. Tokenization, alana özgü bir olaydır. Bu adım için ilk bakışta boşluklara göre ayırmanın yeterli olacağı düşünülse bile, farklı alanlardan gelen düzensiz metinler için daha karmaşık yöntemler gereklidir. Bu duruma, harf ve rakamlardan oluşan marka isimleri (SD500), büyük küçük harf içeren ürün adları (iPhone), tire ile ayrılmış ürün isimleri (wi-fi), Twitter verisi içindeki özel biçimlemeler (#hashtag, @mention), programlama dilleri (C++, C#), noktalama işaretleri, IP adresleri (192.168.1.1), kısaltmalar (T.C.), Web URL ve e-posta adresleri gibi örnekler verilebilir. Ayrıca çalışmalarda kullanılan tokenization yöntemlerinin deneysel sonuçla-

rın tekrarlanabilirliği (reproducibility) üzerinde önemli bir etkisi vardır [30]. Örneğin, bir belgenin hangi bölümlerin tokenlara ayrılacağı, belgede hangi karakterlerin tutulup hangilerinin atılacağı gibi kullanılan tokenization yöntemlerine göre Bilgi Erişim başarımı değişmektedir.

Bilgi Erişim alanında yapılan çalışmalarda tokenization'ın önemi pek fark edilememiştir. Çoğu çalışmada ne tür tokenizer kullanıldığı bile belirtilmemiştir [1]. Bunun sebebi bu bileşenin ya da adımın göz ardı edilip önemsenmediğinin belirtisidir. Bu durum Jimmy Lin ve çalışma arkadaşları tarafından fark edilip, bir çalışmalarında: "Tek bir bileşenin (ör. Tokenizer) sistemin genel performansı üzerindeki etkisi hakkında çok az şey biliyoruz." demiştir [7]. Bu tezin amacı tam olarak anlaşılamayan bu belirsizliği ortadan kaldırmaktır. Bu kapsamda, bu çalışmada özellikle Web dokümanlarından oluşan veri kümelerinin içeriğinde geçen Latin dışı alfabelerle yazılmış içeriklerin tokenization aşamasında nasıl ele alınacağı üzerine çalışılmıştır. Özellikle, bu tez aşağıdaki araştırma sorularını ele almaktadır:

- Bilgi erişiminde kullanılan İngilizce veri setleri tamamen Latin alfabesi ile yazılmış içeriklerden mi oluşmaktadır?
- İçerikte bulunan Latin dışı alfabeler nelerdir?
- Veri seti indekslenirken Latin dışı alfabelerle yazılmış içerikler ters indekste tutulmalı mı yoksa tutulmamalı mı?
- Latin dışı alfabelerle yazılmış bu içeriklerin Bilgi Erişim başarımı üzerinde etkisi var mıdır?

1.1. Tezin Organizasyonu

Bu tezin içeriği şu şekilde devam etmektedir: İkinci kısımda Bilgi Erişim ve tokenization alanlarında yapılan ilgili çalışmalardan bahsedilmektedir. Üçüncü kısımda Bilgi Erişim sistemi ile ilgili genel bilgiler anlatılmaktadır. Dördüncü kısımda karakter kodlamasından bahsedilmektedir. Beşinci kısımda deneysel değerlendirme için kullandığımız veri setleri, araçlar ve yöntemleri içeren deneysel metodoloji anlatılmıştır. Altıncı kısımda elde edilen deneysel sonuçlar gösterilmiştir. Yedince ve son bölümde elde ettiğimiz bulguların genel değerlendirmesi yapılmıştır.

2. İLGİLİ ÇALIŞMALAR

Tokenization, metin işleme barındıran (Bilgi Erişimi, metin sınıflandırma, doğal dil işleme vb.) konularda kritik bir öneme sahiptir. Tokenization, bir metnin "belirli bir anlamı olan herhangi bir karakter dizgisi (token)" yani anlamlı kısımlara bölünmesinin prosedürüdür. Başka bir deyişle bir metni parçalama biçimidir ve hem dokümanlar hem de sorgular üzerinde çalıştırılan ilk adımdır. Uysal ve Günal (2014) iki farklı tokenization yöntemini metin sınıflandırma bağlamında değerlendirmişlerdir [48]. Kullandıkları tokenization türlerden birisi sadece harflerden oluşan (ör. biyomedikal) karakter dizisini dikkate alındığı alfabetik tokenization yöntemidir. Diğeri ise harf ve rakamların (ör. ISO14001) dikkate alındığı alfa numerik tokenization yöntemidir. Bu çalışmada alfabetik tokenization yöntemi kullanıldığı durumda, "1st, 2nd ve 3rd" gibi sayısal terimleri "st, nd, rd" şeklinde tokenlara ayırarak terimlerin anlamsal güçlerinin yitirilmesine sebep olduğunu ama alfa nümerik tokenization kullanıldığı durumda böyle bir problemin olmadığını gözlemlemişlerdir. Biyomedikal Bilgi Erişimi alanında tokenization'ın etkisi Jiang ve Zhai (2007) tarafından incelenmiştir [25]. Bu alandaki veri enzim adları, Latince kimyasal ve organik bileşenlerden oluşmaktadır (ör. 1,25-dihydroxyvitamin ve dead/h). Kuşkusuz bu tür veri üzerinde tokenization daha da önem kazanmaktadır. Yaptıkları çalışmada biyomedikal metinlerin normal metinler gibi sadece boşluklara göre tokenlara ayırmanın, sorgu-belge eşleşmesinde olumsuz etkilerinin olduğunu, bu yüzden biyomedikal metinler için uygun tokenization yöntemleri kullanmanın daha iyi olduğunu gözlemlemişlerdir.

D.Roy ve arkadaşları (2018) WT10G, GOV2 ve ClueWeb gibi Web doküman koleksiyonları üzerinde yaptıkları çalışmada, dokümanlarda bulunan HTML ve meta etiket verilerinin indekse dahil edilmesi veya edilmemesi durumlarında standart Bilgi Erişim modellerinin başarım sonuçlarına etkisini incelemişlerdir [44]. Çalışmalarında, her koleksiyon için *"Full indeks"* ve *"Clean indeks"* olmak üzere iki farklı indeks oluşturmuşlardır. *"Full indeks"*, koleksiyondaki tüm dokümanların orijinal halleri (tüm HTML ve meta etiketleri de dahil) indekse eklenerek oluşturulmuştur. *"Clean indeks"* ise, dokümanlardaki HTML ve meta etiket verilerini kaldıran bir ön işleme metodu uygulanarak oluşturulmuştur. Oluşturulan bu iki farklı indeks, farklı Bilgi Erişim modelleri (Language model-Jelinek-Mercer smoothing, Language model-Dirichlet smoothing, BM25, Divergence from Randomness) üzerindeki başarım sonuçlarına göre kıyaslanmıştır. Elde ettikleri sonuçlarda, çoğu Bilgi Erişim modelinde dokümanların indekslenmeden önce HTML ve meta etiketlerden temizlenmesinin daha iyi olduğunu ancak bazı modellerde bu işle-

min başarıma olumsuz etki yaptığını gözlemlemişlerdir. Gürültü olarak görülen HTML ve diğer meta etiketler, bir sorgu-belge eşleşmesine katılamayabilirler, ancak belgenin uzunluğunu artırarak dolaylı olarak belge sıralamasını değiştirebilmektedirler.

Terrier'in[1] varsayılan İngilizce tokenizer'ı $[a-zA-Z\_0-9]$ kuralına göre çalışır. Terrier'in Bilgi Erişimi aracını kullanan akademik çalışmalarda bu varsayılan tokenizer'ın kullanıldığını (aksi yazarlar tarafından belirtilmemişse) söyleyebiliriz. Örneğin İlker Kocabaş ve arkadaşları (2013) yaptıkları otomatik terim ağırlıklandırma yönteminin deneylerinde Terrier'in varsayılan tokenizer'ını kullanmışlardır [26]. Yine M. Catena ve arkadaşları (2014) farklı modern tam sayı sıkıştırma algoritmalarını deneyerek modern bir Bilgi Erişimi sistemine entegre ettikleri çalışmada Terrier'in varsayılan tokenizer'ını kullanmışlardır [13].

Yaptığımız araştırmalar kadarıyla, Web Bilgi Erişim derlemleri üzerinde $[a-zA-Z\_0-9]$ dışı içeriğin olup olmadığını kontrol eden bir çalışma bulunmamaktadır. Dahası, bu içerik $[a-zA-Z\_0-9]$ gibi bir tokenization yönteminde yok edilmektedir. Web sayfalarından oluşan veri setlerinin içeriğinde her şey olabilmektedir. Web sayfaları, makale özetleri gibi kontrollü/düzenli bir alan değil, aksine kaotik bir yapıya sahiptir. Bir sayfa birden fazla dilde içerik bulundurabilir ya da sadece rakamlardan oluşmuş sayfalardan oluşabilmektedir. Bu çalışmada Web dokümanlarından oluşan İngilizce veri setlerinde bulunan Latin dışı alfabelerle yazılmış içeriğin nasıl ele alınacağı üzerine çalışılmıştır ve bunun Bilgi Erişimi başarımı üzerindeki etkisi incelenmiştir. Görüldüğü gibi tokenization gözden kaçan bir bileşen olduğu için bu bileşenle ilgili çok fazla bir çalışma bulunmamaktadır.

[1] `http://terrier.org/`

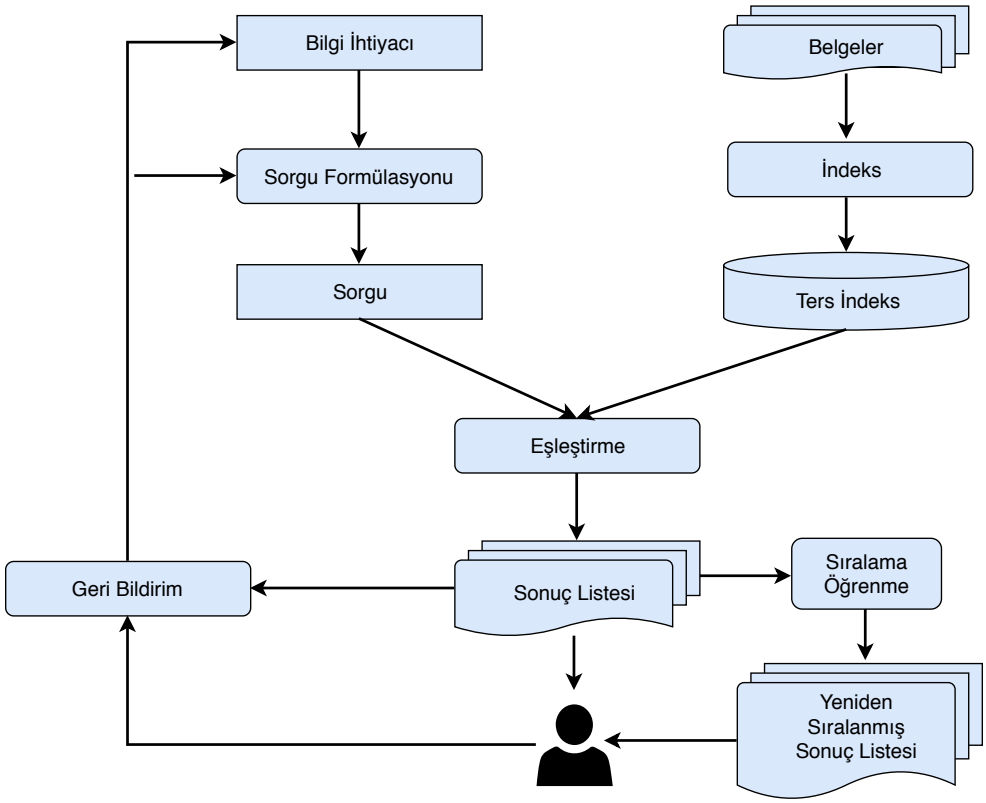
3. BİLGİ ERİŞİMİ

Bu bölümde, geleneksel bir Bilgi Erişim sisteminin yöntemleri, modelleri ve değerlendirmesi anlatılmaktadır. Daha spesifik olarak, bölüm 2.1'de tipik bir Bilgi Erişim sisteminin genel yapısı ve aşamalarından bahsedilmektedir. İndeksleme aşaması ve bu aşamada yapılan ön işlem süreçleri; tokenization, küçük harfe dönüştürme (lowercase conversion), durak kelimeleri ayıklama (stopword removal) ve gövdeleme (stemming) gibi ön işlemler bölüm 2.2'de anlatılmaktadır. Bölüm 2.3'te sorgu formülasyonundan bahsedilmektedir. Boole modeli, vektör uzayı modeli ve çeşitli olasılık yaklaşımları dahil olmak üzere bölüm 2.4'de Bilgi Erişim modelleri tanıtılmaktadır. Bölüm 2.5'te yeniden sıralama anlatılmıştır. Bilgi Erişimi'nin değerlendirilmesi ve değerlendirme ölçütleri bölüm 2.6'da ele alınmıştır.

3.1. Giriş

Bilgi Erişimi, bilgisayar tarihinde köklü bir araştırma alanına sahiptir. "Bilgi Erişim" terimi, Mooers tarafından 1950'lerin başlarında üretilmiştir. Bilgi Erişimi, genellikle geleneksel veri alımına paralel bir eşdeğer olarak görülmektedir, ancak bunun yerine doğal dil sorgularını kullanarak yarı yapılandırılmış veya yapılandırılmamış bilgi kümelerinden bilgi almakla ilgilenmektedir. Bilgi Erişimi, büyük koleksiyonlardan bir bilgi ihtiyacını karşılayan yapısal olmayan bir materyali (genellikle dokümanları) bulma işlemidir [35]. Burada asıl amaç kullanıcının ihtiyaç duyduğu bilgiye ulaşmasını sağlamaktır. Kullanıcının bilgi ihtiyacının karşılayan belgelere $ilgili$ $(relevant)$ belge denir. İdeal bir Bilgi Erişim sisteminin yalnızca ilgili belgeleri getirmesi beklenmektedir. Ancak, bunun gibi mükemmel Bilgi Erişim sistemleri mevcut değildir, çünkü arama ifadeleri mutlaka eksiktir ve alaka düzeyi kullanıcının öznel görüşüne bağlıdır. İki kullanıcı aynı sorguyu bir Bilgi Erişim sistemine yönlendirebilir ve alınan belgeler üzerinde farklı kararlar verebilir.

Bir Bilgi Erişim sistemi birbiriyle etkileşime giren birkaç işlemden oluşur: (i) indeksleme, (ii) sorgu formülasyonu, (iii) eşleştirme ve (iv) yeniden sıralama. Şekil 3.1'de bu süreç gösterilmektedir.



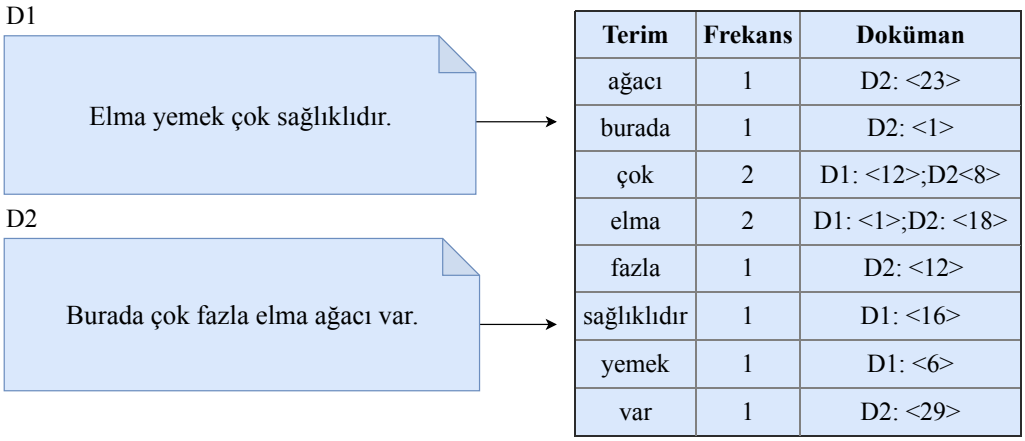
Şekil 3.1. *Bilgi Erişim süreci*

3.2. İndeksleme

Bir dosya sisteminde, belgeler arasında doğrusal arama yapmaktan kaçınmak için verilerin özel olarak tasarlanmış veri yapılarında saklanması gerekir. Belgelerin doğrusal taranması çok yavaş bir işlem olduğundan, eşleştirme işlemini hızlandırmak için belgelerin ters bir dizine kaydedildiği çevrimdışı bir işlem gerekmektedir. Ters indeks, koleksiyon üzerinde hızlı arama yapılmasını sağlayan en çok kullanılan veri yapısıdır. Ters indeks koleksiyondaki tüm benzersiz kelimeleri içeren terimler sözlüğünü tutmaktadır. Ters bir indeksin temel yapısı Şekil 3.2'de gösterilmiştir. Her terim için, hangi belgelerde bulunduğunu ve belgedeki pozisyonlarını kaydeden bir kayıt listesi vardır.

3.2.1 Ön işleme

İndeksleme işlemi öncesinde belgeler, Bilgi Erişim başarımı üzerinde önemli bir etkisi olan bazı ön işleme adımlarından geçmektedir. Ön işleme, genellikle belgedeki gürültüyü gidermek için yapılır, böylece metin terimler listesine dönüştürülür.



Terim	Frekans	Doküman
ağacı	1	D2: <23>
burada	1	D2: <1>
çok	2	D1: <12>;D2<8>
elma	2	D1: <1>;D2: <18>
fazla	1	D2: <12>
sağlıklıdır	1	D1: <16>
yemek	1	D1: <6>
var	1	D2: <29>

Sözlük ve kayıt listesi

Şekil 3.2. *Ters indeks örneği*

Metin ön işleme genel olarak şu adımlardan oluşur: tokenization, küçük harfe dönüştürme (lowercase conversion), durak kelimeleri ayıklama (stopword removal) ve gövdeleme (stemming).

3.2.1.1 Tokenization

Tokenization, bir metnin içerik akışını sözcüklere, terimlere, sembollere veya $token$ olarak adlandırılan diğer anlamlı ögelere bölme işlemidir. Bilgi Erişim sürecinde önemli bir adımdır. Tokenization, metinsel bilgilerin birbirinden ayrı kelimelere bölünmesine yardımcı olur.

Bazı sistemlerde, cümleleri sadece boşluklara göre bölmek yeterli olurken, bazı sistemlerde daha sofistike bir tokenizera ihtiyaç duyabilir (ör., e-posta adreslerinin tanınması ve bunların bir token olarak tutulması gibi). Ancak, tokenization işlemi, kelime sınırlarında boşluk kullanmayan diller için daha sıkıntılı olabilir (ör., Çince, Japonca, Tayca). Bu dillerde, tokenization'ın görevi için *kelime segmantasyonu (word segmentation)* kullanılmalıdır.

3.2.1.2 Küçük harfe çevirme (Lowercase Conversion)

Küçük harfe çevirme, tüm alfabetik belirteçleri küçük harfe dönüştürme işlemidir. Her zaman küçük harfleri kullanmak en iyisidir, çünkü kullanıcılar genellikle doğru yazım kurallarından bağımsız olarak küçük harf kullanmaktadırlar. Örneğin, bir çok kullanıcı "Ferrari" arabasına arayacakları zaman "ferrari" şeklinde yazarak bir arama gerçekleşti-

receklerdir. Ayrıca, bir cümlenin başlangıcındaki "Otomobil" örneklerinin bir "otomobil" sorgusuyla eşleşmesini de sağlamaktadır.

3.2.1.3 Durak kelimelerin kaldırılması (Stop words removal)

Bir belgede binlerce kelime vardır ve kullanıcı bakış açısına göre her kelime eşit derecede öneme sahip değildir. Kelimelerin bazıları, sadece gramer gereksinimlerinden dolayı kullanılmaktadır. Genel olarak, bu kelimeler sıklıkla belgede görülür ve belgenin hiçbir bilgisine katkıda bulunmaz (ör., edatlar, zamirler, bağlaçlar). Yüksek sıklıkta olma nedeniyle, Bilgi Erişim'deki varlıkları belgelerin içeriğini anlamada engel teşkil edebilmektedir. Bu son derece sık kullanılan kelimeler (ör., "for", "the", "of") *durak kelimeler (stop words)* olarak adlandırılmaktadır. Bu kelimeler genellikle indeks işlemi öncesinde yapılan ön işlem sürecinde içerikten çıkarılıp indekse dahil edilmemektedir. Bu işlemin indeks ve arama sürecini hızlandırması gibi avantajları olmasına rağmen, bazı dezavantajları da olabilmektedir. Örneğin, "Let it be", "To be or not to be" gibi durak kelime içeren sorguların aranmasında problem teşkil etmektedir.

3.2.1.4 Gövdeleme (Stemming)

Gövdeleme, kelimelerin morfolojik köklerine indirgenmesidir. Gövdeleme'nin arkasındaki hipotez, aynı kök veya sözcük köküne sahip kelimelerin genellikle metin içindeki aynı veya nispeten yakın kavramları tanımlamasıdır. Bu, Bilgi Erişimi için metin işlemede yaygın olarak kullanılan bir işlemdir. Çoğul ve geçmiş zaman ekleri, bir sorgu terimi ile ilgili bir belge terimi arasında mükemmel bir eşleşmeyi önleyen söz dizimsel varyasyon örnekleridir. Gövdeleme algoritmaları, morfolojik türevleri tanımlayarak benzer terimleri ortak bir kök formuna indirger. Örneğin, "$likes, liked, likely, liking$" hepsi "$like$" kelimesine indirgenir. Bilgi Erişim de kullanılmak üzere pek çok gövdeleme algoritması önerilmiştir, en yaygın kullanılanı Porter stemmer [38] 'dır. KStem [28], Porter stemmer için daha az agresif bir alternatif olan İngilizce için yaygın olarak kullanılan bir diğer gövdeleme yöntemidir.

3.3. Sorgu Formülasyonu (Query Formulation)

Sorgu formülasyonu, başarılı bir Bilgi Erişim'inin önemli bir parçasıdır. Bilgi ihtiyacı ile Bilgi Erişim sistemine gönderilen gerçek sorgu arasında bir farklılık vardır. Mesela bilgi ihtiyacı: "Ev dekorasyonu için hangi ürünler var?" sorgusu "ev dekorasyon ürünler" şeklinde sorgulanabilir. Bazı arama sistemleri, kullanıcılarına geri bildirim sağlayarak sorgu formülasyonlarında rehberlik eder; örneğin, ticari arama motorlarının "*bunu mu demek istediniz?*" özellikleri gibi.

3.4. Bilgi Erişim Modelleri

3.4.1 Boolean modeli (Boolean Model)

Boolean modeli, en eski Bilgi Erişim modellerinden biridir. Bu model, Boole cebri ve küme teorisine dayanmaktadır. Boolean modeli, sorgu terimlerini birleştirmek için George Boole'nin matematiksel mantık operatörlerini (VE, VEYA, DEĞİL) kullanmaktadır. Bu modelde, koleksiyondaki bir belgede, sorgu terimlerinin belgede görünüp görünmediği göz önüne alınarak, verilen sorgu ile alakalı veya alakasız olduğu değerlendirilir.

Bu model sıralama mekanizmasından yoksundur. Belgeler alınır veya alınmaz, ancak sonuç kümesindeki belgeler sıralanmaz. Bu nedenle, tüm belgelerin eşit derecede önemli olduğu kabul edilir. İkili karar sistemi kullandığı için, alaka düzeyi arasındaki farkı ayırt edemez ve kısmi eşleşmeleri geri getiremez. Bu nedenle, Baeza-Yates ve Ribeiro-Neto tarafından bir bilgi alma modelinden çok veri alma modeli olarak kabul edilmektedir [39].

3.4.2 Vektör uzay modeli (Vector Space Model)

Vektör Uzay Modeli, belgeleri belirli bir sorguya uygunluk sırasına göre sıralamak için ikili olmayan bir ağırlıklandırma sistemine dayanmaktadır. Her terime, her belgeye göre bir ağırlık verilmekte ve her belge daha sonra bu ağırlıkların n-boyutlu bir vektörü ile temsil edilmektedir. Burada n, belge koleksiyonundaki tekil terim sayısıdır.

Salton ve McGill [45] belge gösterimlerini ve sorguyu yüksek boyutlu bir Öklid uzayında tanımlı vektörler olarak kabul etmiştir. Her bir terim ayrı bir boyutla temsil

edilir, böylece uzayın boyutu, indeksteki toplam benzersiz terim sayısına eşittir. Bir sorgu ve bir belge arasındaki benzerlik daha sonra kendi vektörleri arasındaki açının kosinüsü olarak hesaplanır. Denklem 3.1, ilişkiyi tahmin etmek için kullanılan iki vektör $\overrightarrow{d}$ ve $\overrightarrow{q}$ arasındaki θ açısının kosinüsüdür.

$$score(\overrightarrow{d}, \overrightarrow{q}) = \cos(\theta) = \frac{\overrightarrow{d} \cdot \overrightarrow{q}}{|\overrightarrow{d}| \times |\overrightarrow{q}|} = \frac{\sum_{i=1}^{N} d_i \times q_i}{\sqrt{\sum_{i=1}^{N} (d_i)^2} \times \sqrt{\sum_{i=1}^{N} (q_i)^2}} \quad \textbf{(3.1)}$$

Burada $\cos(0°) = 1$ ve $\cos(90°) = 0$ olduğu dikkat edilmelidir. Vektör uzay modelinde, vektör bileşenlerinin değerleri tanımlanmamıştır. Vektör bileşenlerine uygun ağırlıkların atanması sorunu, **terim ağırlıklandırma (term weighting)** olarak bilinir. Muhtemelen en ünlü terim ağırlıklandırma, belge içi terim frekansı tf ve ters belge frekansı idf'nin bir birleşimi olan $tf \cdot idf$ ağırlıklarıdır. tf-idf ağırlıklandırma şeması tarafından D belgesinde t terimine verilen ağırlık Denklem 3.2'de verilmiştir. N, koleksiyondaki toplam belge sayısı ve $df(t)$ ise belge sıklığıdır.

$$weight(t, D) = tf(t, D) \times \log_2 \frac{N}{df(t)} \quad \textbf{(3.2)}$$

3.4.3 Olasılık modeli (Probabilistic Model)

Olasılık modeli, belgeler alakaya uygunluk olasılıklarına göre sıralandığında, optimum performansın elde edildiğini belirten Olasılık Sıralama İlkesine (OSİ) (Probability Ranking Principle) dayanmaktadır [41]. Bu da, ilgili ve ilgisiz belgelerde terimlerin dağılımına dayanarak tahmin edilir. Bu nedenle, bu model belge kümesi üzerinde en uygun sıralamayı sağlamakla ilgilenmektedirler. Bu model, koleksiyonlardaki belgeleri $P(R|D)$ alaka düzeyini azaltmak için sıralar, yani belge D 'ye verilen alaka R olasılığıdır. OSİ'nin genel formu, Robertson ve Jones [42] tarafından önerilen ve RSJ modeli olarak adlandırılan model Denklem 3.3'te verilmiştir.

$$S(Q, D) = \log P(R|D) = \sum_{t \in Q \cap D} \frac{P(D_t = 1|R) \cdot P(D_t = 0|\bar{R})}{p(D_t = 0|R) \cdot P(D_t = 1|\bar{R})} \quad \textbf{(3.3)}$$

Burada R, $\{R, \bar{R}\}$ değerlerini alan , R= ilgili ve $\bar{R}$= ilgisiz olan, rastgele bir değişkendir. $P(R)$, ilgililik olasılığını gösterirken, $P(\bar{R})$ ise ilgisizlik olasılığını göstermektedir. D_t, {0, 1} değerlerini alan rastgele bir değişkendir; $D_t = 0$, D belgesinin t terimini

içermediği ve $D_t = 1$ ise D belgesinin t terimini içerdiği anlamına gelmektedir. Dolayısıyla, $P(D_t = 1|R)$, D belgesinin alaka düzeyi verilen t terimini içerme olasılığıdır. Robertson ve Walker [43] bu dağıtımlar için 2-Poisson modelini benimsemiştir ve halen en iyi performans gösteren terim ağırlıklandırma algoritmalarından biri olan ünlü Okapi BM25 terim ağırlıklandırma modelini geliştirmişlerdir.

BM25 modeli, aşağıdaki formülü kullanarak bir belge-sorgu çiftini puanlar:

$$score(D,Q) = \sum_{t \epsilon Q \cap D} IDF(t).\frac{tf_{t,D} \cdot (k_1 + 1)}{tf_{t,D} + k_1 \cdot (1 - b + b \cdot \frac{|D|}{avgdl})} \quad \textbf{(3.4)}$$

Burada k_1 ve b, sırasıyla terim-frekans etkisini ve belge uzunluğu normalizasyonunu kontrol eden serbest parametrelerdir. BM25 iki serbest parametre (k_1 ve b) içerdiğinden, aslında her durumda bir dizi ağırlıklandırma şemasını temsil etmektedir.

3.4.4 Dil modeli (Language Model)

Dil modeli [37], terim oluşumlarıyla ilgili olasılıkları kullandığından olasılıklı Bilgi Erişim modelleri ailesinin bir parçası olarak görülebilir. Bununla birlikte, verilen örnek bir sözlü cümlede en olası kelime sırasını tahmin etmeye çalışan konuşma tanıma tekniklerinden elde edilir. Bilgi Erişim alanında, bu model belgeleri dil modelleri olarak görür ve belirli bir dil modelinin verilen sorguyu üretme olasılığını tahmin eder.

Dil modellerinin Bilgi Erişimine uygulanması [22, 29, 37], aslında otomatik konuşma tanıma alanından alınan bir yeniliktir. Bilgi Erişim'de, dil modeli yaklaşımında her belgenin kendi dil modeli vardır. Ardından sorgunun bu doküman tarafından oluşturulma olasılığı, her doküman için dil modeli ile hesaplanır. Bir $unigram$ modelinde, bu işlem, Denklem 3.5'de verilen, bireysel terimlerin olasılıklarının çarpımıdır. Daha sonra bu olasılık verilen bir sorgu için belgeleri sıralamak için kullanılır.

$$P(t \epsilon Q|D) = \prod_{t \epsilon Q} P(t_i|D) \quad where \quad P(t|D) = \frac{tf_{t,D}}{|D|} \quad \textbf{(3.5)}$$

Ancak bu denklem, tüm sorgu terimlerini içermeyen bir belgeye sıfır olasılık atar ($t \epsilon Q$). Sıfır olasılıktan kaçınmak için genellikle pürüzsüzleştirme ($smoothing$) adı verilen bir yöntem kullanılır, burada sıfır olmayan bir olasılık, puanlanan belgede oluşmayan herhangi bir terime atanır.

3.4.5 Rastlantısallıktan farklılık (Divergence from randomness)

Bilgi edinmenin olasılıksal modellerini türetmek için Amati ve Van Rijsbergen [6] tarafından bir çerçeve sunulmuştur. Bu modeller, dil modeli yaklaşımında elde edilen parametrik olmayan Bilgi Erişim modelleridir. Terim ağırlıklandırma modelleri, gerçek terim dağılımının, rastgele bir işlem altında elde edilenden farklılığının ölçülmesiyle türetilir. Rastlantısallıktan Farklılık modelinde, bir belgenin önemli terimlerinin, frekansları Poisson, Hyper-Geometric, Bose-Einstein gibi temel bir rastgelelik modeli tarafından önerilen frekanstan ayrılan terimler olduğu varsayılmaktadır. En popüler Rastlantısallıktan Farklılık modellerinden biri, özellikle yüksek hassasiyetli işlerde etkili olan PL2'dir. PL2, normalize edilmiş terim frekansı (tfn) dağılımları için bir Poisson dağılımını varsayar ve Denklem 3.6'da verilen Normalizasyon 2'yi, belge içi terim frekansı ($tf_{t,D}$). olarak kullanır.

$$tfn = tf_{t,D} \cdot \log_2(1 + c \cdot \frac{avdl}{|D|}) \tag{3.6}$$

Burada c, serbest bir parametredir ve avdl, koleksiyondaki ortalama belge uzunluğudur. PL2 modeli, aşağıdaki formülü kullanarak bir belge-sorgu çiftini puanlar:

$$PL2(D,Q) = \sum_{t \epsilon Q \cap D} \frac{1}{tfn+1}\left(tfn \cdot \log_2 \frac{tfn}{\lambda} + (\lambda - tfn) \cdot \log_2 e + 0.5 \cdot \log_2(2\pi \cdot tfn)\right) \tag{3.7}$$

Burada λ bir Poisson dağılımının varyansı ve ortalamasıdır. λ, koleksiyon içindeki terim sıklığının koleksiyondaki toplam doküman sayısına bölünmesiyle elde edilir.

3.4.6 Bilgi tabanlı modeller (Information-based models)

Bilgi Erişimi için bilgi tabanlı modeller ailesi, Clinchant ve Gaussier [16–18] tarafından ortaya koyulmuştur. Bu modeller, bir kelimenin belgedeki davranışlarındaki ve koleksiyon seviyelerindeki fark, belgedeki kelimenin anlamıyla ilgili bilgi getirir hipotezine dayanmaktadır. Modelin en başarılı örneklemesi Denklem 3.8'de verilen, normalleştirilmiş terim frekansı (tfn) dağılımları için bir log-lojistik dağılım varsayan LGD'dir.

$$LGD(D,Q) = \sum_{t \epsilon Q \cap D} -\log(\frac{\lambda_t}{\lambda_t + tfn}) \quad where \quad tfn = tf_{t,D} \cdot \log_2(1 + c \cdot \frac{avdl}{|D|}) \quad \textbf{(3.8)}$$

3.4.7 Bağımsızlıktan farklılık (Divergence from independence)

Kocabaş, Dinçer ve Karaoğlan [26], belgelerin terimlerden bağımsızlık derecesini, bunların ortaya çıkma sıklıkları bakımından ölçmelerine dayanan, otomatik bir terim ağırlıklandırma yöntemi ortaya koymuşlardır. Bağımsızlıktan farklılık, sağlam temelli bir istatistiksel teoriye sahiptir. Bu model, verilen bir belgede D, t teriminin beklenen terim frekansını hesaplar. Doküman koleksiyonu, doküman uzunluğu $|C|$ olan tüm koleksiyonun toplam terim sayısı olan büyük bir doküman C olarak kabul edilir. t terimi, uzunluğu $|C|$ olan yapay belgede tf(t,C) kez görülür. Oran orantı kullanılarak, beklenen bir terim sıklığı , Denklem 3.9'daki gibi hesaplanır. Daha sonra, beklenen terim frekansı e, gerçek belge içi terim sıklığı tf(t,D) ile karşılaştırılır. Eğer tf(t,D) ≤ e ise, sıfır puan verir.

$$\frac{tf(t,C)}{|C|} = \frac{e}{|D|} \quad \textbf{(3.9)}$$

Her biri farklı temellere dayanan üç temel Bağımsızlıktan farklılık ölçümü vardır:

- DFIB= $\log_2\left(\frac{tf(t,D)-e}{e} + 1\right)$ doymuş bağımsızlık modeline dayalı *(saturated model of independence)*
- DFIC= $\log_2\left(\frac{(tf(t,D)-e)^2}{e} + 1\right)$ normalleştirilmiş ki kare mesafesine dayanan *(normalized chi-squared distance from independence)*
- DFIZ= $\log_2\left(\frac{tf(t,D)-e}{\sqrt{e}} + 1\right)$ standardizasyona dayanan *(standardization)*

3.5. Yeniden Sıralama

Yeniden sıralama, eşleştirme işleminden sonra uygulanan bir işlemdir. Eşleştirme sürecinde, standart terim ağırlıklandırma modeli (PL2, BM25 vb.), belirli bir sorgu için bir belge listesi döndürür. Referans terim ağırlıklandırma modeli tarafından döndürülen listedeki en iyi K belgelerinin makine öğrenme tekniklerini kullanarak yeniden sıralanmasına, Sıralamayı Öğrenme (Learning to Rank) [31] denir. Sıralamayı öğrenme, bir ör-

nek belge için çıkarılan çeşitli özelliklerin etkin öğrenilmiş modellere dağıtılmasını içerir ve bu daha sonra yeniden sıralamak için kullanılır [34]. Etkili bir öğrenme modelinde birçok özellik bir araya getirilmiştir. Bunlar hem sorgudan bağımsız belge özellikleri (gelen bağlantılar, URL derinliği, vb.), hem de sorguya bağlı olan belge özellikleri (başlık, gövde, bağlantı metni vb.) [34]. Sıralamayı öğrenme, Bilgi Erişim topluluğunda önemli bir ilgi kazanmıştır, verilen bilgiyi sıralama yöntemleriyle geliştirmek ve karşılaştırmak için bir çok araştırma yapılmıştır [47].

3.6. Bilgi Erişiminde Değerlendirme

Bilgi Erişim sistemlerinin değerlendirilmesi, bir sistemin, kullanıcılarının bilgi gereksinimlerini ne kadar iyi karşıladığını değerlendirme sürecidir [51]. Bilgi Erişim değerlendirme forumları, test koleksiyonları oluşturur ve Bilgi Erişim sistemlerinin standart bir şekilde değerlendirilmesini sağlar. Bunlardan bazıları Tablo 3.1'de listelenmiştir. Bunlardan en popüleri, NIST ve ABD Savunma Bakanlığı tarafından ortaklaşa düzenlenen TREC'dir [52].

Bilgi Erişim sistemlerini karşılaştırmalı olarak değerlendirmek için, geleneksel TREC stili (Cranfield paradigması olarak da adlandırılır) değerlendirme metodolojisi benimsenmiştir. Değerlendirme metodolojisinde, bir belge koleksiyonu, bir dizi bilgi gereksinimi (konular veya sorgular) ve hangi belgelerin hangi konularla alakalı olduğunu gösteren bir dizi uygunluk değerlendirmesi (doğru cevaplar) olması gerekmektedir.

Tablo 3.1. *Bilgi Erişim değerlendirme forumları*

Forum	İsim	Referans
TREC	Text Retrieval Conference	`trec.nist.gov`
CLEF	Conference and Labs of the Evaluation Forum	`www.clef-initiative.eu`
FIRE	Forum for IR Evaluation	`fire.irsi.res.in`
NTCIR	NII Testbeds and Community for Information Access Research	`ntcir.nii.ac.jp`

Bir bilgi ihtiyacının bir örneği ve örnek konunun sorgu alaka düzeyi kararlarından bir kısmı sırasıyla Tablo 3.2 ve Tablo 3.3'te verilmiştir.

Sorgu alaka düzeyi kararları dosyası dört sütundan oluşur: TOPIC #, ITERATION, DOCUMENT # ve RELEVANCY.

1. TOPIC# konu numarasıdır.
2. ITERATION geri besleme yinelemedir. (neredeyse her zaman sıfırdır ve kullanılmaz)
3. DOCUMENT# resmi belge tanımlayıcısıdır.
4. RELEVANCY 1'den küçük olduğu zaman ilgisiz olduğunu ve 0'dan büyük olduğu zaman ilgili olduğunu gösterdiği bir tam sayı kodudur.

Alaka düzeyi etiketleri (Relevance labels), ikili (1= ilgili veya 0= ilgisiz) veya derecelendirilmiş (2= yüksek derecede alakalı; 1= alakalı; 0= alakalı olmayan; -2= spam / önemsiz) olabilmektedir.

Tablo 3.2. *Bilgi İhtiyacı örneği*

```
<topic number="250" type="single">
   <query>ford edge problems</query>
   <description>
   What problems have afflicted the Ford Edge car model?
   </description>
</topic>
```

Tablo 3.3. *Sorgu alaka düzeyi karar örneği (Query Relevance Judgements)*

```
250 0 clueweb12-0704wb-77-25051 1

250 0 clueweb12-0705wb-39-02251 0

250 0 clueweb12-0813wb-43-02321 0

250 0 clueweb12-0815wb-63-08988 2
```

3.6.1 Değerlendirme ölçümleri

Bilgi Erişim etkinliğini ölçmek için çeşitli değerlendirme ölçümleri önerilmiştir. Kesinlik ($precision$) ve hassasiyet ($recall$) Bilgi Erişim etkinliğinin iki ana ölçüsüdür.

Kesinlik, alınan ilgili belgelerin, alınan tüm belgelere oranı iken, hassasiyet, alınan ilgili belgelerin, koleksiyondaki ilgili belgelere oranıdır. Yani kesinlik, "getirilen bilginin ne kadarı, istenilen bilgiyle ilgilidir?" sorusunun, hassasiyet ise, "getirilmesi gereken bilginin ne kadarı getirilmiştir?" sorusunun cevabıdır. Örneğin, bir sistem yalnızca bir tanesi ilgili olan iki belge verirse, o sistemin kesinlik %50 (1/2) olur. Ancak, söz konusu koleksiyonun toplamda 10 ilgili dokümanı varsa, hassasiyet %10 (1/10) olacaktır.

$$kesinlik \ \ (precision) = \frac{\#\ konuyla\ ilgili\ alınan\ belgeler}{\#\ alınan\ belgeler} \tag{3.10}$$

$$hassasiyet \ \ (recall) = \frac{\#\ konuyla\ ilgili\ alınan\ belgeler}{\#\ koleksiyondaki \ \ ilgili\ belgeler} \tag{3.11}$$

F-measure, kesinlik ve hassasiyetin harmonik ortalamasıdır.

$$F_{measure} = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{3.12}$$

Mean Average Precision (MAP): Sabit bir sıradaki kesinlik ($precision$) k (P@k), ilgili belgelerin en iyi sonuçlar arasında elde edilen oranıdır, örneğin, ilk 20 sıradaki Kesinlik (P@20) şeklinde ifade edilmektedir. Ortalama kesinlik (Average precision (AP)), alınan her belgenin alınmasından sonra hesaplanan kesinlik puanlarının ortalamasıdır, alınmayan ilgili belgelerin kesinliği için sıfır kullanılır [10].

$$AP(q) = \frac{1}{|R_q|} \sum_{r \epsilon R_q} P@rank(r), \tag{3.13}$$

burada R_q, q sorgusu ile ilgili belgeleri temsil eder.

Tüm sorgu kümesinde ($q \in Q$), AP ortalaması alındığında, MAP elde edilir. MAP, ikili alaka düzeyi değerlendirmeleri için standart bir ölçümdür. Daha yakın zamanlarda, NIST'teki belge-sorgu çiftlerini değerlendirmek için altı puanlı derecelendirme ölçeği (4, 3, 2, 1, 0, -2) kullanılmıştır ve detaylı açıklamaları TREC 2014 Web Track raporunda anlatılmıştır [19]. Burada ilgililik dereceleri 1, 2, 3, 4 ilgili, 0 / -2 dereceleri konuyla ilgisiz olarak değerlendirilmektedir.

Normalized Discounted Cumulative Gain (NDCG@k) : NDCG@k, dereceli alaka kararlarını yerine getirebilen yaygın olarak kullanılan bir ölçüdür [24]. k sırasındaki DCG

Denklem 3.14'deki gibi hesaplanır:

$$DCG@k = \sum_{i=1}^{k} \frac{2^{g_i} - 1}{log_2(1+i)}, \tag{3.14}$$

burada g_i, i. sıradaki belgenin alaka derecesidir. NDCG, Denklem 3.14'in normalleştirilmiş sürümüdür ve $\frac{DCG@k}{ideal\,DCG@k}$ formülü ile hesaplanır.

3.6.2 Ölçü hesaplama için değerlendirme komut dosyaları

TREC organizatörleri tarafından sağlanan diğer bir önemli bileşen, bir Bilgi Erişim etkinliğini hesaplamak için kullanılan standart değerlendirme araçlarıdır. Bu bileşen, Bilgi Erişim etkinliğinin ölçümünün hesaplanmasında, bir standart belirlemek için önemlidir. En yaygın değerlendirme araçları aşağıdaki gibidir:

`trec_eval`[1], verilen sonuç dosyası ve standart bir değerlendirilmiş sonuç kümesi (qrels) göz önüne alındığında, Bilgi Erişim çalışmasını değerlendirmek üzere TREC organizatörleri tarafından yayınlanan ilk değerlendirme aracıdır.

`gdeval.pl`, trec-web 2014[2] indirilebilen NDCG@k ölçülerini hesaplamak için kullanılan bir değerlendirme aracıdır.

`statAP_MQ_eval_v4.pl`[3], TREC'in Million Query (MQ)[4] parçalarında kullanılan sorgu betiği kararlarının geleneksel dört sütun `qrels` dosya formatı yerine beş sütun `prels` dosya formatı olarak yayınlandığı değerlendirme script'idir. Prels dosya formatının beş sütunu şunlardır: TOPIC#, DOCUMENT#, ALGORITHM# ve INCLUSION_PROBABILITY.

[1] `http://trec.nist.gov/trec_eval/`
[2] `http://github.com/trec-web/trec-web-2014`
[3] `http://ir.cis.udel.edu/million/statAP_MQ_eval_v4.pl`
[4] `http://trec.nist.gov/data/million.query.html`

4. KARAKTER KODLAMASI

Karakter kodlama, bir çeşit kodlama sistemi kullanılarak oluşturulmuş karakter gruplarını temsil etmektedir. Kullanıldığı bağlama bağlı olarak karakterlere karşılık gelen kod noktaları ve bunların oluşturdukları kod alanı şeklinde tanımlanabilir. Bilgisayar ortamında her şey 0 ve 1 ile temsil edildiğinden, bilgisayar ile metin işlemleri yapabilmek için sayıların belirli birer karakterlerle eşleştirilmesi gerekmektedir. Bu yüzden metinsel verilerin depolanması, işlenmesi ve iletimi esnasında karakter kodlamaları kullanılmaktadır. Karakterler kodlanırken, önceden belirlenmiş uluslararası standartlara uyulması, metinlerin tüm dünyada elektronik olarak değiştirilmesine olanak tanımaktadır.

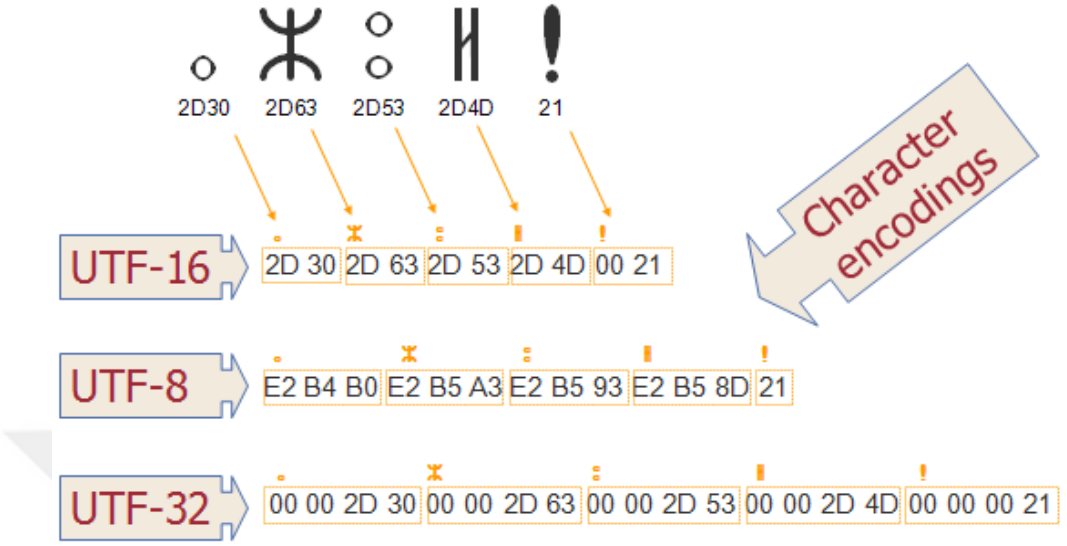
ASCII (American Standard Code for Information Interchange)(Bilgi Değişimi İçin Amerikan Standart Kodlama Sistemi): Harflerin bilgisayar ortamında saklanması ve taşınması ile ilgili geliştirilen ilk sistem ASCII sistemi olmuştur. Latin alfabesi üzerine kurulu 7 bitlik bir karakter kümesidir. 7 bit ile yazılabilen en büyük sayı 127'dir ve 128 adet sayı (0 ile 127 arası) temel noktalama işaretleri ve Amerikan İngilizcesi harflerini kodlamak için yeterli olmuştur. Latin alfabesi kullanan Avrupa ülkelerinin alfabelerindeki " ü, ö ,ç, é, ä, æ , £, ¥ " gibi harflerin de kodlanması ihtiyacı ortaya çıktığından genişletilmiş ASCII denilen yeni bir sistem kullanılmıştır. İçinde binlerce farklı karakter barındıran Çince veya Japonca gibi dillerin harfleri için fazladan gelen 128 karakterlik kapasitenin de yeterli olması olanaksızdır. Bu yüzden ASCII sistemi yerini Unicode'a bırakmıştır.

UNICODE : Unicode Konsorsiyum [1] organizasyonu tarafından geliştirilen ve her karaktere bir sayı değeri karşılığı atayan bir endüstri standardıdır. Unicode'un amacı farklı karakter kodlama sistemlerinin birbiriyle tutarlı çalışmasını ve dünyadaki tüm yazım sistemlerindeki metinlerin bilgisayar ortamında tek bir standart altında temsil edilebilmesini sağlamaktır. Unicode, modern ve antik dünyanın tüm yazı sistemleri için tüm karakterleri kapsamakta ve aynı zamanda teknik semboller, noktalamalar ve metin yazarken kullanılan birçok karakteri de içermektedir.

Unicode da, karakterleri kodlamak için birden çok kodlama sistemi kullanmaktadır. Unicode ile kullanılabilen en yaygın kodlama sistemleri UTF-8, UTF-16 ve UTF-32 dir. UTF-8, karakterleri temsil etmek için 1 bayt ile 4 bayt arasında değerler kullanmaktadır. UTF-16, ek karakterler için 4 bayt kullanırken bunun dışındaki diğer karakterler için 2

[1] `http://unicode.org/consortium/consort.html`

bayt kullanmaktadır. UTF-32, tüm karakterler için 4 bayt kullanmaktadır. Şekil 4.1'de farklı karakter kodlama sistemleri gösterilmiştir.



Şekil 4.1. *Karakter kodlama [23]*

Unicode kendi başına dilleri değil, diller için grafik işaretler kümesini (script) kodlamaktadır. Script, bir veya daha fazla yazı sisteminde metin bilgisini temsil etmek için kullanılan harflerden ve diğer yazılı işaretlerden oluşan bir koleksiyondur. Örneğin, Rusça Kiril script'inin bir alt kümesi ile yazılırken, Ukraynaca farklı bir alt kümesiyle yazılmıştır. Japonca yazı sistemi birkaç script kullanmaktadır. Birçok script (özellikle Latin script) çok sayıda dil yazmak için kullanılmaktadır. Unicode, aşağıdaki script'lerle yazılabilecek tüm dilleri kapsamaktadır: Latin, Greek, Cyrillic, Armenian, Hebrew, Arabic, Syriac, Thaana, Devanagari, Bengali, Gurmukhi, Oriya, Tamil, Telugu, Kannada, Malayalam, Sinhala, Thai, Lao, Tibetan, Myanmar, Georgian, Hangul, Ethiopic, Cherokee, Canadian Aboriginal Syllabics, Khmer, Mongolian, Han (Japanese, Chinese, Korean ideographs), Hiragana, Katakana, and Yi.

Unicode karakter kodlamasında her bir karaktere 16 bitlik birer kod verilmiştir. Unicode kodlama sistemi kod blokları şeklinde düzenlenmiştir. Tablo 4.1'te örnek unicode kod blok aralıkları verilmiştir. Unicode karakter kümesindeki ilk 65.536 kod noktası Temel Çok Dilli Düzlemi (Basic Multilingual Plane) oluşturmaktadır. Unicode karakter seti ayrıca yaklaşık bir milyon ek kod noktası konumu için alan içermektedir. Bu ikinci aralıktaki karakterlere ek karakter (supplementary characters) denilmektedir ve "surrogate" bir UTF-16 kod değeridir. UTF-16 için, tek bir ek karakteri temsil etmek için bir "surrogate pair" gereklidir. İlki (high surrogate), U + D800 ila U + DBFF aralığındaki 16 bitlik bir

kod değeridir. İkinci (low surrogate), U + DC00 ila U + DFFF aralığında bir 16 bit kod değeridir. Surrogate mekanizmasını kullanarak, UTF-16 1.114.112 potansiyel Unicode karakterlerini destekleyebilmektedir.

Tablo 4.1. *Örnek Unicode kod blokları*

Kod bloğu	**Alfabe**
0020 – 007F	Temel Latin
0100 – 017F	Latin Genişletilmiş-A
0180 – 024F	Latin Genişletilmiş-B
0370 – 03FF	Yunanca ve Kıpti
0400 – 04FF	Kiril

5. DENEYSEL METODOLOJİ

5.1. Giriş

Bu bölümde deneysel değerlendirme için benimsediğimiz metodoloji açıklanmaktadır. Ayrıca kullanılan veri setlerini, sorgu setlerini, analiz için kullanılan değerlendirme ölçütlerini, terim ağırlıklandırma modellerini ve Apache Lucene anlatılmaktadır.

5.2. Veri Setleri

Deneysel analizler için, çeşitli boyut ve içerikte çok sayıda standart TREC Web koleksiyonu üzerinde deneylerimizi gerçekleştirdik. Metzler ve Kurland [36] tarafından önerilen ClueWeb09 ve ClueWeb12 veri setleri ve GOV2 veri setini kullandık. ClueWeb09 veri seti, Ocak ve Şubat 2009'da toplanan yaklaşık 1 milyar Web sayfasından oluşmaktadır. ClueWeb09B, ClueWeb09A'nın "Kategori B" kısmı olan tüm veri kümesinin ilk 50 milyon İngilizce sayfasından oluşmaktadır. ClueWeb09B ve ClueWeb09A aynı konu setini kullanmaktadır. ClueWeb12 veri seti, 10 Şubat 2012 ve 10 Mayıs 2012 tarihleri arasında toplanan 733,019,372 İngilizce Web sayfasından oluşmaktadır. ClueWeb12, ClueWeb09 Web veri setinin tamamlayıcısı veya halefidir. ClueWeb12'nin dağıtımı Ocak 2013'te başlanmıştır. ClueWeb12-B13 veri kümesi, tüm veri seti içerisinde bulunan her bir dosyadan 14.belgeyi alarak oluşturulmuştur. Bu yüzden tüm veri setinin %7'sini temsil eden bir örneğidir. GOV2 veri seti, Bilgi Erişimi ve ilgili teknolojiler üzerinde yapılan araştırmaları desteklemek için yaklaşık 25 milyon Web sayfasından oluşmaktadır. Çoğunlukla İngilizce dili ile yazılmış .gov uzantılı Web sayfalarından 2004 yılının başlarında toplanarak oluşturulmuştur [3]. Veri setlerine ait istatistiksel bilgiler Tablo 5.3 ve Tablo 5.4'de gösterilmektedir.

5.3. Sorgu Setleri

Bu bölümde deneysel çalışmalarımızda kullandığımız sorgu setleri anlatılmaktadır. Kullanılan sorgu setleri TREC ve NTCIR organizasyonları tarafından oluşturulmuştur.

5.3.1 Terabyte track

Terabyte Track'in temel amacı, terabayt ölçekli belge koleksiyonları için bir değerlendirme metodolojisi geliştirmektir. Bu amaç için ilk olarak 2004 yılında TREC 2004 [14] de 49 adet yeni sorgu oluşturulmuştur. Daha sonra TREC 2005 [15] ve TREC 2006 [11] da 50 şer adet yeni sorgu daha eklenerek toplamda 149 adet sorgu oluşturulmuştur. Bu sorgu kümeleri belge koleksiyonu olarak Gov2 veri setini kullanmaktadır.

5.3.2 Million query

Million Query(MQ), ilk olarak TREC 2007'de iki amaca hizmet etmek için tasarlanmıştır [3]. Bunlardan birincisi, geniş bir belge koleksiyonunda geçici bilgi alımının (ad-hoc retrieval) keşfetmek, ikincisi ise birçok yüzeysel veya daha az kapsamlı karar kullanarak değerlendirmenin daha iyi olup olmadığını ve küçük karar gruplarının tekrar kullanılabilir olup olmadığını araştırmak amacıyla yapılmıştır.

TREC 2007 Million Query (MQ07[1]) [3] , belge koleksiyonu olarak GOV2 veri setini kullanmaktadır. MQ07 , 1524 adet sorgu içermektedir. TREC 2008 Million Query (MQ08[2]) [2], belge koleksiyonu olarak GOV2 veri setini kullanmaktadır. MQ08 , 564 adet sorgu içermektedir. TREC 2009 Million Query (MQ09[3]) [12], belge koleksiyonu olarak ClueWeb09B veri setini kullanmaktadır. MQ09 koleksiyonu, ilk 50 tanesi Web Track 2009 (WT09)'dan alınmış 561 adet sorgu içermektedir. MQ09'un sorgu alaka düzeyi kararları, dört sütun $qrels$ dosyası yerine beş sütun $prels$ dosyası olarak yayınlanmıştır. MQ'lerin değerlendirilmesinde $statAP\_MQ\_eval\_v4.pl$[4] script'i kullanılmaktadır.

5.3.3 Web track

TREC Web Track, Web bilgi alma teknolojisini büyük Web veri koleksiyonları üzerinden incelemek ve değerlendirmek amacıyla 2009-2014 tarihleri arasında oluşturulmuştur. Her yıl yaklaşık 50 yeni sorgu oluşturularak ClueWeb09 ve ClueWeb12 gibi büyük Web veri koleksiyonları üzerinde deneyler ve değerlendirmeler yapılmaktadır.

[1] `https://trec.nist.gov/data/million.query07.html`
[2] `https://trec.nist.gov/data/million.query08.html`
[3] `http://ir.cis.udel.edu/million/data.html`
[4] `http://ir.cis.udel.edu/million/statAP_MQ_eval_v4.pl`

5.3.4 Tasks track

TREC Tasks Track, kullanıcının bir sorgu göndermesini sağlayan temel görevi ne kadar iyi anlayabileceklerini ve kullanıcıların görevlerini tamamlamalarına yardımcı olmak için ne kadar yararlı olduklarına göre bilgi alma sistemlerinin kalitesini değerlendirmek amacıyla 2015 ve 2016 yıllarında oluşturulmuştur [50, 53]. Deney ve değerlendirmelerini ClueWeb12 Web veri koleksiyonu üzerinde gerçekleştirmişlerdir.

5.3.5 NTCIR-13 we want web task

NTCIR-13 We Want Web Task Track [33], TREC'in Web Track işine son vermesi üzerine NTCIR tarafından, Web Bilgi Erişim probleminin tamamen çözülmüş bir problem olmadığını düşünerek, bunu devam ettirmek amacıyla 2017 yılında oluşturulmuştur. Çince ve İngilizce olmak üzere iki farklı alt görev üzerinde değerlendirme yapılmıştır. İngilizce için 100 sorgu oluşturularak deneyleri ClueWeb12B-13 Web veri koleksiyonu üzerinde gerçekleştirmişlerdir. NTCIR-14 ve NTCIR-15'in de 2019 ve 2020 yıllarında yapılması planlanmaktadır. Sorguların istatistiği ve bunlara ilişkin alaka düzeyleri Tablo 5.1'de verilmektedir.

Tablo 5.1. *Sorgu istatistikleri*

Veri Seti	Track	Yıl	Sorgu Sayısı	Ortalama Sorgu Uzunluğu	Ortalama Sorgu Başına Alakalı Belge	Alaka Düzeyi
Gov2	Terabyte	2004	49	3.1	216.7 (± 172.1)	3
	Terabyte	2005	50	3.1	208.1 (± 148.0)	3
	Terabyte	2006	50	3.1	117.9 (± 101.0)	3
	Million Query	2007	1524	3.8	12.3 (± 9.6)	3
	Million Query	2008	564	5.2	5.2 (± 6.7)	2
ClueWeb09B	Million Query	2009	562	2.6	15.5 (± 25.7)	3
ClueWeb09	Web	2009	49	2.1	140.0 (± 79.0)	3
	Web	2010	48	2.0	109.0 (± 70.7)	5
	Web	2011	50	3.4	63.1 (± 63.7)	5
	Web	2012	50	2.3	70.5 (± 55.3)	6
ClueWeb12	Web	2013	50	3.3	83.0 (± 65.1)	6
	Web	2014	50	3.3	113.3 (± 74.9)	6
	Tasks	2015	35	3.7	28.7 (± 14.8)	4
	Tasks	2016	50	3.7	50.4 (± 25.6)	4
ClueWeb12-B13	We Want Web	2017	100	2.2	145.3 (± 66.4)	5

5.4. Terim Ağırlıklandırma Modelleri

Yaptığımız deneylerin Bilgi Erişim başarım etkinliğini sekiz terim ağırlıklandırma modeli ile karşılaştırdık. Bu modeller Tablo 5.2'te sıralanmıştır:

Tablo 5.2. *Terim ağırlıklandırma modelleri*

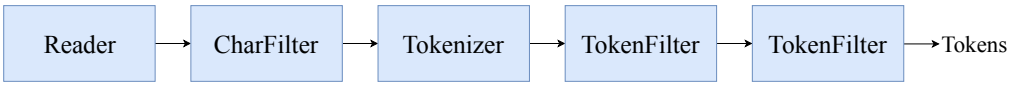
Sıra	Model	Aile	Referans
1	BM25	Olasılık	Robertson and Zaragoza [40]
2	Dirichlet	Dil modeli	Zhai and Lafferty [54]
3	DFIC	Parametrik Olmayan	Kocabaş et al. [26]
4	DFRee	Hipergeometrik	Amati [5]
5	DLH13	Hipergeometrik	Amati [4]
6	DPH	Hipergeometrik	Amati [4]
7	LGD	Bilgi Tabanlı	Clinchant and Gaussier [18]
8	PL2	Rastlantısallıktan Farklılık	Amati and Van Rijsbergen [6]

5.5. Apache Lucene

Apache Lucene[5], tamamen Java dilinde yazılmış, ölçeklenebilir, yüksek performanslı indeksleme ve güçlü, doğru ve verimli arama algoritmaları sağlayan açık kaynaklı bir arama kütüphanesidir [9]. Lucene, tüm dünyaya yayılmış gönüllü geliştiriciler sayesinde, hızla gelişmekte ve büyümektedir. Bugün Lucene'nin en popüler ve en çok kullanılan açık kaynaklı yazılımlardan biri olduğu söylenebilir. Ayrıca Apache Software Foundation[6] ile de işbirliği yapmaktadır. Son zamanlarda, Lucene'nin akademik çalışmalarda kullanımı da ivme kazanmaktadır [8].

Lucene, metin araması gerektiren çoğu uygulama için uygun bir teknolojidir. İçerisinde bir çok farklı özellik bulundurmaktadır. Örneğin, gelen içerik ve sorguların analizi, indeksleme ve depolama, arama ve sayısız faydalı modüller gibi özellikler içermektedir. Bu çalışmada çoğunlukla Lucene'nin analiz, indeksleme ve arama özellikleri kullanılmıştır.

Analiz, serbest biçimli metni en temel indekslenmiş gösterimine yani terimlerine dönüştürme işlemidir. Analiz işlemi Tokenizer ile başlar ve bir dizi TokenFilters ile devam eder. Bir çözümleyici (analyzer), analiz işleminin enkapsülasyonudur. Çözümleyiciler hem indeksleme hem de sorgu ayrıştırma için kullanılır. Çözümleyicinin görevi, verilen verileri token adı verilen temel indeks terimlerine dönüştürmektir. Bir çözümleyici üç bölümden oluşur: (i) sıfır veya daha fazla karakter filtresi (char filters), (ii) tek bir tokenizer, (iii) sıfır veya daha fazla token filtresi (token filter). Bir çözümleyici zinciri (analyzer chain) yapısı Şekil 5.1'de gösterilmektedir.



Şekil 5.1. *Çözümleyici zinciri (Analyzer chain)*

ICUTokenizer: ICU[7](International Components for Unicode - Unicode için Uluslararası Bileşenler), temeli IBM firması tarafından atılmış, şu anda birçok firmanın destek verdiği C/C++ ve Java programlama dilleri ile yazılan uygulamalar için evrensel kod ve

[5] `http://lucene.apache.org/`
[6] `http://www.apache.org`
[7] `http://site.icu-project.org/`

küreselleşme desteği sağlayan olgunlaşmış bir kütüphanedir. ICU, hem ticari yazılımlarla hem de diğer açık kaynaklı ya da özgür yazılımlarla kullanım için uygun, kısıtlayıcı olmayan açık kaynaklı bir lisans altında yayımlanmaktadır. ICU tarafından sağlanan hizmetlerden bazıları aşağıda listelenmiştir:

- **Kod Sayfası Dönüştürme:** Metin verilerini Unicode'a veya Unicode'ları metine dönüştürebilir ve bunları neredeyse diğer tüm karakter kümelerine veya kodlamaya dönüştürebilir.
- **Karşılaştırma:** Metinleri, belirli bir dil, bölge veya ülkenin standartlarına göre karşılaştırabilir.
- **Biçimlendirme:** Seçilen bir yerel ayarın düzenine göre sayıları, tarihleri, saatleri ve para birimlerini formatlayabilir.
- **Zaman Hesaplamaları:** Geleneksel Gregoryen takviminin ötesinde birden fazla takvim türü sağlar.
- **İki yönlü (Bidirectional):** Soldan sağa (İngilizce) ve sağdan sola (Arapça veya İbranice) veri içeren bir metni işleme desteği sunar.
- **Unicode Desteği:** ICU, Unicode standardını yakından takip ederek, birçok Unicode karakter özelliğine, Unicode Normalizasyonuna, Case Folding ve Unicode Standardında belirtilen diğer temel işlemlere kolay erişim sağlamaktadır.

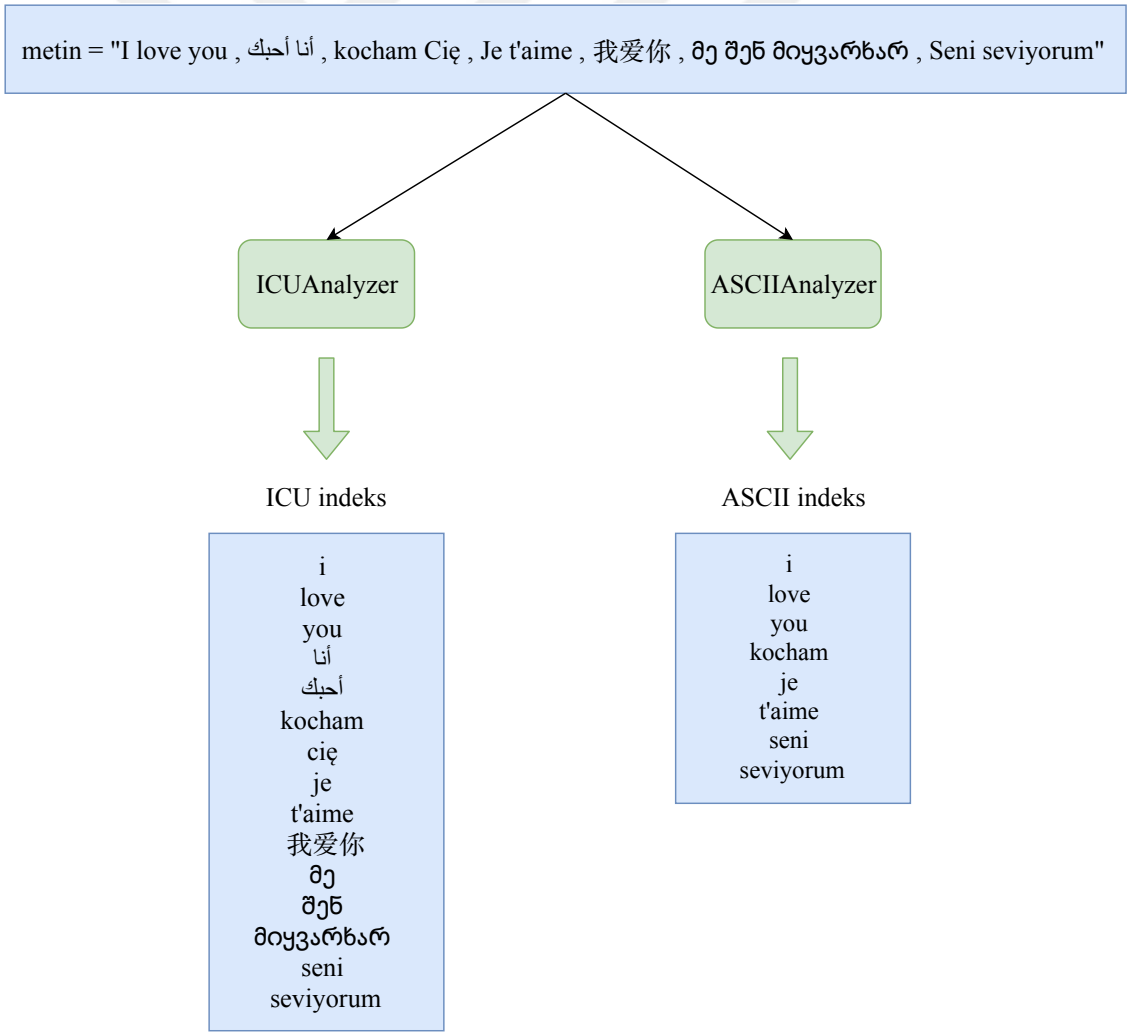
ICUTokenizer, standart tokenizer ile aynı Unicode Metin Segmentasyonu algoritmasını kullanmaktadır. Ancak Tayland, Laos, Çince, Japonca ve Korece kelimeleri tanımlamak için sözlük tabanlı bir yaklaşım kullanarak ve bazı kelimeleri ayırmak için özel kurallar kullanarak Asya dilleri için daha iyi destek sağlamaktadır.

5.6. Kullanılan Metodoloji

Deneysel çalışmalarımızda Apache Lucene 7.7.1 versiyonu kullanılmıştır. Veri setlerinde bulunan dokümanlar Web belgesi olduğu için öncelikle bu belgelerde bulunan HTML etiketleri jsoup[8] kütüphanesi kullanılarak temizlenmiştir. Daha sonra her koleksiyon için *"ICU indeks"* ve *"ASCII indeks"* olmak üzere iki farklı indeks oluşturulmuştur.

[8] `https://jsoup.org`

ICU indeks, koleksiyonlardaki tüm dokümanların orjinal içerikleri (Latin ve Latin dışı içerikler dahil) indekse eklenerek oluşturulmuştur. *ASCII indeks* ise, bir ön işleme yöntemi uygulanarak dokümanlardaki temel Latin alfabesi dışındaki alfabelerle yazılmış içerikler çıkarıldıktan sonra, geriye kalan temel Latin alfabesiyle yazılmış içerikler indekse eklenerek oluşturulmuştur. Yani ASCII indekste sadece kod noktası 0 ile 127 arasında olan karakterlerle yazılmış içerikler tutulmaktadır. Şekil 5.3'de ASCII sisteminde bulunan temel Latin karakterler gösterilmektedir. Sadece bu tablodaki karakterlerden oluşmuş kelimeler *ASCII indekste* yer almaktadır. Bu tablo dışında herhangi bir karakteri içeren token/kelimeler göz ardı edilmektedir. Her iki indeks için ön işleme aşamasında, tokenlara ayırmak için ICUTokenizer, gövdeleme işlemi için KStem [28], küçük harflere dönüştürme işlemi için LowerCaseFilter kullanılmıştır. Şekil 5.2'de örnek indeksleme süreci gösterilmektedir.



Şekil 5.2. *İndeksleme süreci*

0	<NUL>	32	<SPC>	64	@	96	`
1	<SOH>	33	!	65	A	97	a
2	<STX>	34	"	66	B	98	b
3	<ETX>	35	#	67	C	99	c
4	<EOT>	36	$	68	D	100	d
5	<ENQ>	37	%	69	E	101	e
6	<ACK>	38	&	70	F	102	f
7	<BEL>	39	'	71	G	103	g
8	<BS>	40	(	72	H	104	h
9	<TAB>	41	)	73	I	105	i
10	<LF>	42	*	74	J	106	j
11	<VT>	43	+	75	K	107	k
12	<FF>	44	,	76	L	108	l
13	<CR>	45	-	77	M	109	m
14	<SO>	46	.	78	N	110	n
15	<SI>	47	/	79	O	111	o
16	<DLE>	48	0	80	P	112	p
17	<DC1>	49	1	81	Q	113	q
18	<DC2>	50	2	82	R	114	r
19	<DC3>	51	3	83	S	115	s
20	<DC4>	52	4	84	T	116	t
21	<NAK>	53	5	85	U	117	u
22	<SYN	54	6	86	V	118	v
23	<ETB>	55	7	87	W	119	w
24	<CAN>	56	8	88	X	120	x
25	<EM>	57	9	89	Y	121	y
26	<SUB>	58	:	90	Z	122	z
27	<ESC>	59	;	91	[	123	{
28	<FS>	60	<	92	\	124	|
29	<GS>	61	=	93	]	125	}
30	<RS>	62	>	94	^	126	~
31	<US>	63	?	95	_	127	<DEL>

Şekil 5.3. *ASCII karakter tablosu*

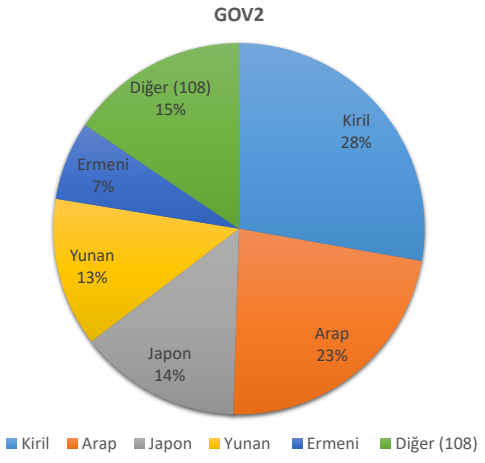
Şekil 5.4'de ClueWeb12A veri setinde en çok geçen Latin dışı alfabelerin terimleri gösterilmektedir. Terimlere ait doküman frekans ve toplam terim frekans değerleri EK-1 de verilmiştir.

Arap	Ermeni	Kiril	Devanagari	Gürcü
العربية	հայերեն	русский	हिन्दी	ღ
عربي	վ	українська	मराठी	ქართული
فارسی	դ	беларуская	हिंदी	ლ
و	ե	български	ॐ	ჟ
من	ւ	и	नेपाली	ღღ
اردو	ծ	в	में	ŋ
في	ա	македонски	महाराष्ट्र	ъ
الله	ղ	я	टाइम्स	საქართველო
على	խ	с	नवभारत	მარგალური
ما	ժ	език	इकनॉमिक	ღღღ

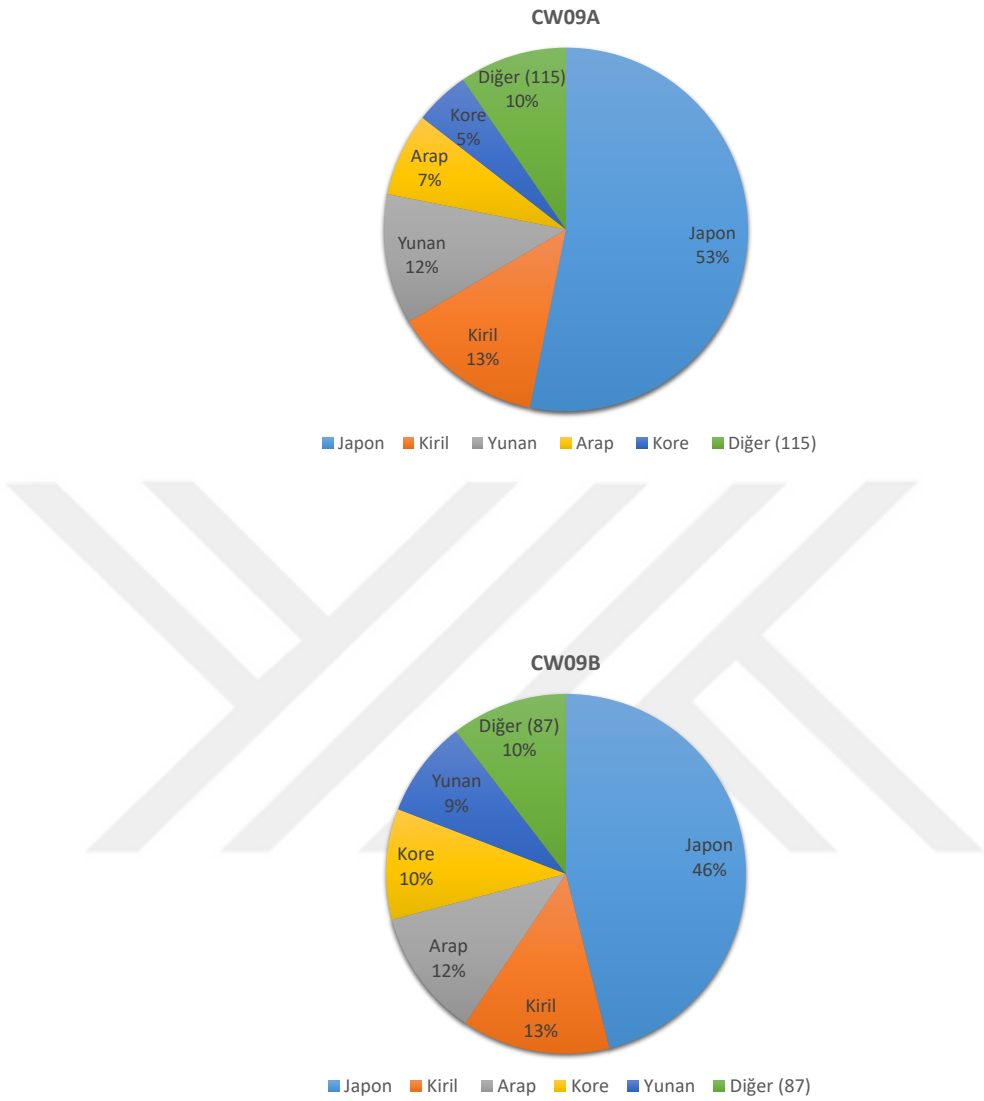
Yunan	Kore	İbrani	Japon	Thai
ελληνικά	한국어	עברית	日本語	ไทย
α	한국	יידיש	中文	ภาษา
τ	킹	נ	简体	ค
σ	우리말	ע	繁體	ที่
ι	수	פ	の	ประเทศไทย
υ	이	γ	汉语	การ
ω	대한민국	ישראל	は	ใน
ε	로그인	ר	的	ของ
δ	한글	ס	に	คน
ν	월	של	を	ใหม่

Şekil 5.4. *ClueWeb12A veri setinde en çok geçen Latin dışı alfabelerin ilk 10 terimleri*

Şekil 5.5'de GOV2 veri setinde en çok geçen Latin dışı alfabeler gösterilmektedir. Bu veri setinde en fazla geçen Latin dışı alfabenin Kiril alfabesi olduğu görülmektedir. Kiril alfabesinden sonra sırayla Arap, Japon, Yunan, Ermeni alfabeleri gelmektedir. GOV2 veri setinde Latin alfabesi dışında toplam 113 farklı Latin dışı alfabe bulunmaktadır.

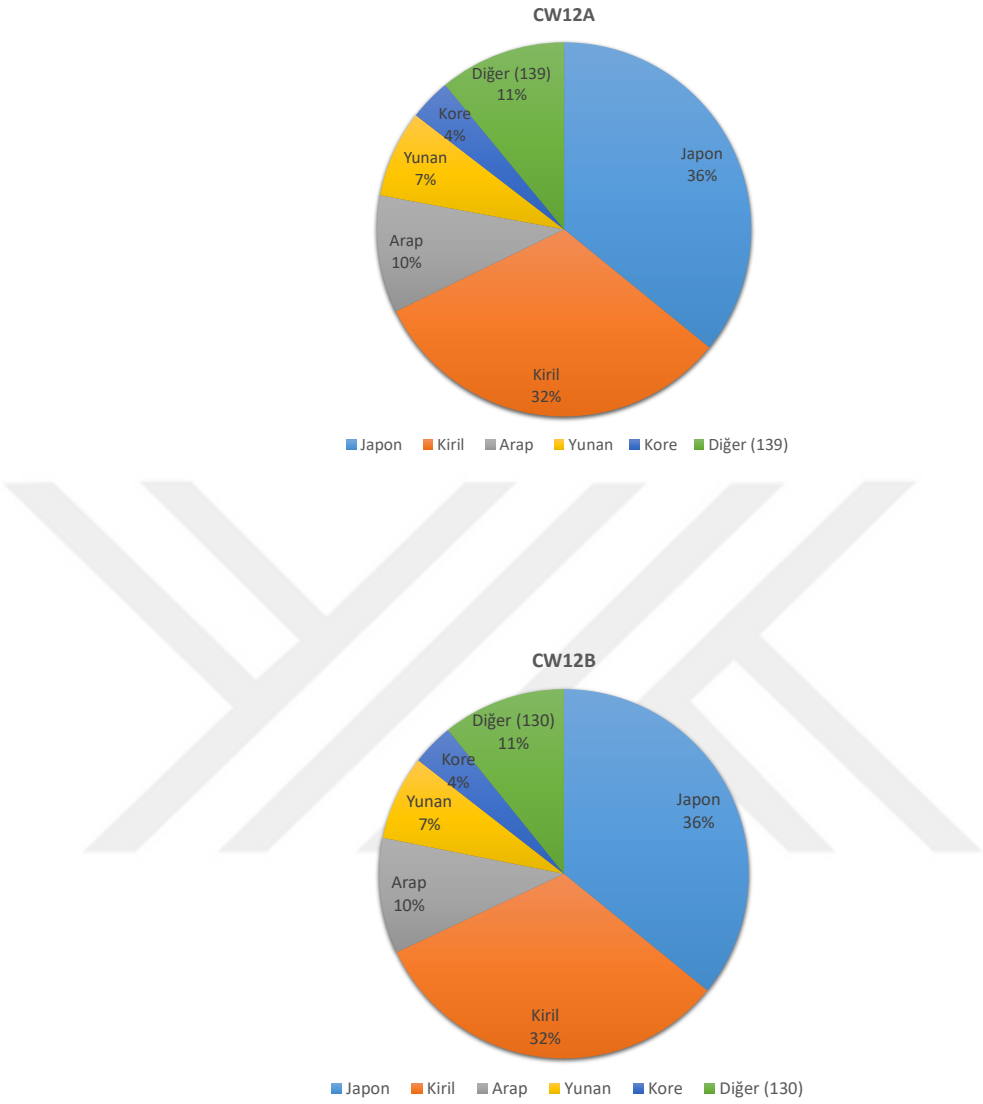


Şekil 5.5. *GOV2 veri setinde en çok geçen Latin dışı alfabelerin dağılımı*



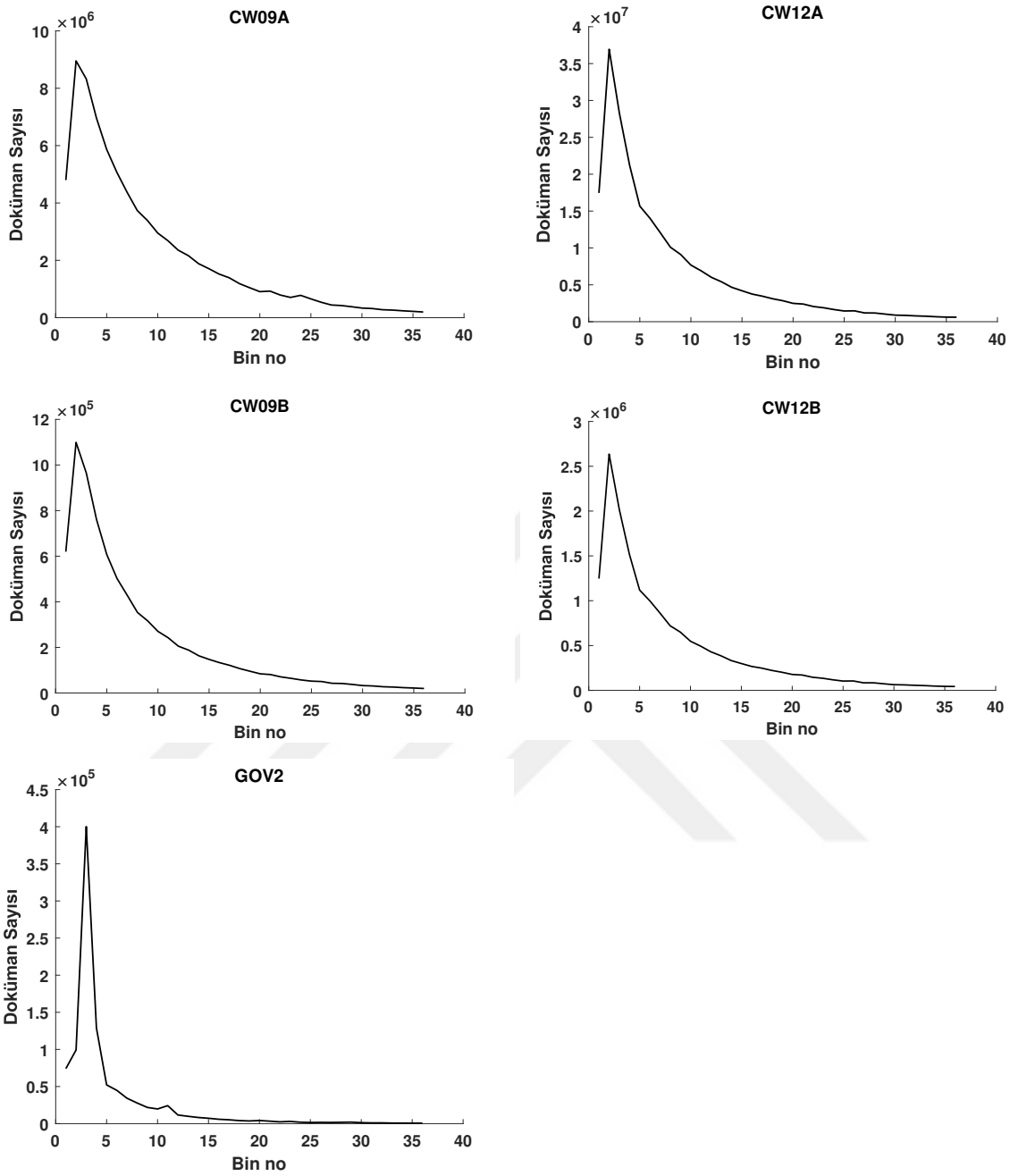
Şekil 5.6. *ClueWeb09 veri setinde en çok geçen Latin dışı alfabelerin dağılımı*

Şekil 5.6'de ClueWeb09A ve ClueWeb09B veri setlerinde en çok geçen Latin dışı alfabeler gösterilmektedir. Bu veri setlerinde en fazla geçen Latin dışı alfabenin Japon alfabesi olduğu görülmektedir. Japon alfabesinden sonra ClueWeb09A'da sırayla Kiril, Yunan, Arap, Kore alfabeleri, ClueWeb09B'de sırayla Kiril, Arap, Kore, Yunan alfabeleri gelmektedir. ClueWeb09A ve ClueWeb09B'de en çok geçen Latin dışı alfabelerin aynı alfabeler olduğu fakat farklı oran ve sırada bulunduğu görülmektedir. ClueWeb09A veri setinde Latin alfabesi dışında toplam 120 farklı Latin dışı alfabe, ClueWeb09B veri setinde Latin alfabesi dışında toplam 92 farklı Latin dışı alfabe bulunmaktadır.



Şekil 5.7. *ClueWeb12 veri setinde en çok geçen Latin dışı alfabelerin dağılımı*

Şekil 5.7'da ClueWeb12A ve ClueWeb12B veri setlerinde en çok geçen Latin dışı alfabeler gösterilmektedir. Bu veri setlerinde en fazla geçen Latin dışı alfabenin Japon alfabesi olduğu görülmektedir. Japon alfabesinden sonra ClueWeb12A ve ClueWeb12B'de sırayla Kiril, Arap, Yunan, Kore alfabeleri gelmektedir. ClueWeb12A ve ClueWeb12B'de en çok geçen Latin dışı alfabelerin aynı alfabeler olduğu görülmektedir. ClueWeb12A veri setinde Latin alfabesi dışında toplam 144, ClueWeb12B veri setinde Latin alfabesi dışında toplam 135 farklı Latin dışı alfabe bulunmaktadır.



Şekil 5.8. *Veri setlerindeki Latin dışı terimlerin frekans dağılımları*

Şekil 5.8'de veri setlerinde geçen Latin dışı terimlerin frekans dağılımları gösterilmektedir. Burada, belge uzunluklarını normalleştirmek amacıyla göreceli terim frekanslarını kullanıyoruz. Göreceli terim frekansı, bir belgedeki bir terimin geçme sayısının belgelerin uzunluğuna oranını ifade etmektedir. Ham terim frekansları ayrık bir dağılımı, göreceli terim frekansları sürekli bir dağılımı oluşturmaktadır. Gerekli ayrık dağılımı elde etmek için, herhangi bir terimin hesaplanan göreceli terim frekans değerleri, sınırlı sayıda 1000 bin şeklinde gruplandırılmıştır. Göreceli terim frekansları 0 ile 1 arasında değişe-

bildiği için 0'ın göreceli frekans değeri dışında, bu aralık eşit uzunluktaki 1000 aralığa (0.000-0.001], (0.001-0.002], vb. bölünmüştür. Grafiklere baktığımızda genel olarak Latin dışı terimlerin en çok ‰2-‰3 geçtiği doküman olduğunu söyleyebiliriz.

Tablo 5.3. *Veri seti istatistikleri*

Veri Seti	Toplam doküman sayısı	Toplam terim sayısı		Tekil terim sayısı		En az bir terimi olan belge sayısı	
		ICU	ASCII	ICU	ASCII	ICU	ASCII
GOV2	25,170,851	21,841,480,500	21,823,185,287	65,426,405	64,972,325	25,170,679	25,170,657
CW09A	503,892,674	334,431,387,594	333,647,417,315	679,815,158	671,349,893	503,892,225	503,892,214
CW09B	50,220,191	39,532,247,545	39,450,903,566	131,263,102	128,933,056	50,220,164	50,220,163
CW12A	731,659,408	525,922,253,287	522,678,211,666	1,368,845,183	1,347,051,214	731,559,623	731,545,224
CW12B	52,245,792	37,565,340,824	37,333,978,973	204,450,203	199,382,093	52,238,719	52,237,730

Tablo 5.4. *Ortalama belge uzunlukları*

Veri Seti	Ortalama belge uzunluğu			Toplam Latin dışı terim oranı (‰)
	ICU	ASCII	Fark	
GOV2	867.7	867.0	0.726	0.23
CW09A	663.7	662.1	1.555	0.45
CW09B	787.2	785.6	1.619	0.32
CW12A	718.9	714.5	4.420	2.11
CW12B	719.1	714.7	4.415	2.10

Tablo 5.3 ve Tablo 5.4'de, deneylerde kullandığımız veri setlerine ait istatistiksel bilgiler verilmiştir. Burada, kullanılan tokenizer yöntemine göre veri setlerinin toplam terim sayısı, tekil terim sayısı gibi istatistiksel bilgilerinin değiştiği görülmektedir. Terim sayısındaki bu değişim belge uzunluklarını etkilemektedir. Belge uzunluklarının değişmesi sonucunda ortalama belge uzunluğu değişmektedir. ASCII indeks kullanıldığı zaman ICU indekse göre veri setlerindeki terim sayısının azaldığını, bunun sonucunda ortalama belge uzunluklarının ASCII de daha küçük olduğu Tablo 5.4'de görülmektedir. Kullanılan tokenization yöntemine göre oluşan bu değişiklikler sonucunda Bilgi Erişim başarımı da

değişmektedir. Çünkü Bilgi Erişim başarım hesaplamalarında kullanılan terim ağırlıklandırma modelleri tf, doküman sayısı, doküman uzunluğu, ortalama doküman uzunluğu, token sayısı, terim frekansı, doküman frekansı gibi istatistiksel bilgiler üzerinden hesaplama yaptığı için bu bilgilerde oluşan değişiklikler Bilgi Erişim başarımının değişmesine neden olmaktadır.

- **tf:** Bir terimin bir belgede görünme frekansı
- **Doküman sayısı:** Koleksiyondaki toplam doküman sayısı
- **Doküman uzunluğu:** Koleksiyonda bulunan bir belgelenin uzunluğu
- **Ortalama doküman uzunluğu:** Koleksiyondaki toplam terim sayısının, en az bir terimi olan belge sayısına oranı
- **Token sayısı:** Koleksiyondaki toplam token sayısı
- **Terim frekansı:** Koleksiyondaki terim frekansı
- **Doküman frekansı:** Koleksiyonda bir terimin farklı dokümanlarda geçme frekansı

6. DENEYSEL SONUÇ VE ANALİZ

Yaptığımız deneylerde, geri kazanım etkinliğini ölçmek için, NDCG@20 ve MAP ölçümlerini kullanıyoruz. NDCG, derecelendirilmiş alaka düzeylerinin olduğu durumlarda akademik çalışmalarda önerilen ve yaygın olarak kullanılan bir ölçümdür [21, 49]. Web veri kümeleri üzerinde yaptığımız deneyler için NDCG@20 ve MAP sonuçları sırasıyla Tablo 6.1 ve Tablo 6.2'de gösterilmiştir.

Burada Latin dışı alfabelerle yazılmış içeriğin Bilgi Erişim başarımına etkisini incelemek amacıyla, ICU ve ASCII indeks için 8 farklı Bilgi Erişim modellerinin (BM25, DFIC, DLH13, LGD, LM, PL2, DPH, DFRee), farklı Web veri ve sorgu kümeleri üzerindeki başarımları gösterilmektedir. ICU indekste Latin ve Latin dışı alfabelerle yazılmış içerikler tutulurken, ASCII indekste sadece Latin alfabesi ile yazılmış içerikler tutulmuştur. Tablolarda ICU ve ASCII indeksten geri kazanım değeri büyük olanlar kalın olarak belirtilmiştir. Değerler arasındaki fark, *t*-test [27] ($p < 0.05$)'e göre anlamlı olanlar ise "†" işareti ile gösterilmiştir. ICU ve ASCII için veri seti ve sorgu setine göre ortalama olarak en yüksek geri kazanım değerini veren modeller farklı renkte vurgulanmıştır.

Tablo 6.1'deki NDCG@20 değerlerine baktığımızda başarımın modelden modele, veri ve sorgu kümelerine göre değişiklik gösterdiğini gözlemlemekteyiz. Sonuçların çoğunda ICU daha iyi başarım gösterirken, bazılarında da ise ASCII daha iyi sonuç vermektedir. Genel ortalamaya baktığımızda sonuçların birbirine yakın olduğu görülmektedir. Ama ICU'nun BM25 modeli için MQ08 ve MQ09 da, DPH modeli için NTCIR da ($p < 0.05$)'e göre anlamlı olarak daha iyi değerler verdiği görülmektedir. Bunun dışında hiçbir yerde ASCII'nin daha iyi olup da aradaki farkın anlamlı olduğu bir değer yoktur. Buradan şu sonuçları çıkartabiliriz: "†" işaretinin olduğu durum çok fazla değil ama ICU'yu kullanmanın herhangi bir riski yoktur. Ama ASCII için aynı şeyleri söyleyemiyoruz. Çünkü BM25 ve DPH modelinde eğer ASCII kullanırsak istatistiksel olarak daha kötü sonuç alabilmekteyiz.

Tablo 6.1. *NDCG@20 sonuçları*

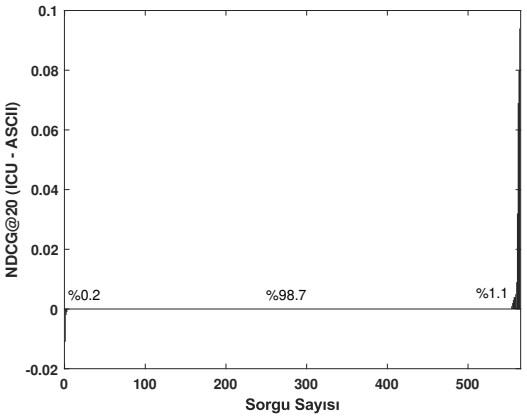
Model	İndeks	Gov2	MQ07	MQ08	MQ09	CW09A	CW09B	CW12A	CW12B	NTCIR
BM25	ICU	0.36876	**0.22888**	**0.25911**†	**0.25276**†	**0.07100**	**0.14960**	**0.14184**	**0.08599**	**0.36857**
	ASCII	**0.36884**	0.22881	0.25860	0.25220	0.07078	0.14933	0.14131	0.08572	0.36777
DFIC	ICU	**0.36182**	0.19982	0.19202	**0.21537**	**0.04194**	**0.15083**	**0.15233**	**0.10841**	0.34869
	ASCII	0.36181	**0.20009**	**0.19227**	0.21536	0.04180	0.15081	0.15186	0.10836	**0.34924**
DLH13	ICU	0.31194	**0.22348**	**0.24237**	0.23322	0.06392	**0.12941**	**0.11724**	**0.07803**	**0.32396**
	ASCII	**0.31223**	0.22332	0.24224	**0.23357**	**0.06395**	0.12925	0.11701	0.07794	0.32317
LGD	ICU	**0.35746**	**0.24917**	**0.27138**	0.24503	**0.06830**	**0.14878**	**0.18090**	**0.11009**	0.36193
	ASCII	0.35727	0.24893	0.27123	**0.24506**	0.06827	0.14868	0.18042	0.10994	**0.36350**
LM	ICU	**0.39153**	0.24912	0.25720	**0.20950**	0.03964	**0.14545**	**0.19978**	**0.13362**	**0.35247**
	ASCII	0.39149	**0.24930**	**0.25722**	0.20941	**0.03969**	0.14544	0.19962	0.13356	0.35211
PL2	ICU	0.33314	0.19728	**0.18209**	**0.23841**	0.06721	**0.13788**	**0.15698**	0.10022	0.35788
	ASCII	**0.33356**	**0.19742**	0.18205	0.23830	0.06721	0.13781	0.15663	**0.10041**	**0.35861**
DPH	ICU	0.39571	**0.26025**	**0.27499**	0.28448	0.09374	**0.17941**	0.19267	**0.10760**	**0.36598**†
	ASCII	**0.39574**	0.26013	0.27474	**0.28460**	**0.09386**	0.17925	**0.19304**	0.10746	0.36386
DFRee	ICU	**0.35526**	**0.24590**	**0.27337**	**0.27218**	0.08096	0.15772	**0.16719**	0.09647	**0.36733**
	ASCII	0.35519	0.24584	0.27320	0.27208	**0.08104**	**0.15777**	0.16691	**0.09671**	0.36678

Tablo 6.2'deki MAP sonuçlarına baktığımızda yine benzer sonuçlar olduğunu gözlemliyoruz. Burada da MQ08 için PL2 ve LGD modelinde, ClueWeb09B için DLH13 ve PL2 modelinde eğer ASCII yi kullanacak olursak istatistiksel olarak kötü sonuç almaktayız. Her ne kadar çoğu sorgu için ICU ve ASCII değerleri birbirine yakın olsa da yani fark yaratmasa da, sonuçlara baktığımızda ICU'yu kullanmak daha iyi ve risksizdir. Bu yüzden Web veri kümelerinde ASCII yerine ICU'yu kullanmayı öneriyoruz. Hem bunun pratik bir açılımı da bulunmaktadır. Örneğin sorgu kümelerinde, HTML veya Japonca karakter geçmiyor diye gerçek hayatta bu böyle olacak diye bir şey yoktur. Gerçek hayattaki kullanıcılar sorgulara Japonca karakter veya HTML etiketleri de girebilmektedir. Bu yüz-

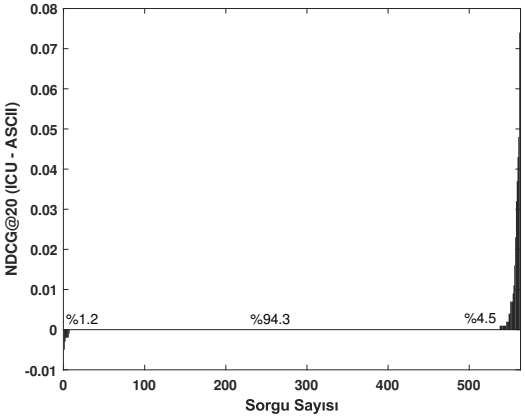
den genel olarak bu çalışmanın sonucunda, Web dokümanlarının içeriğinde bulunan Latin ve Latin dışı alfabelerle yazılmış bütün içeriğin olduğu gibi yani orijinal haliyle indekste tutmanın daha iyi olduğunu söyleyebiliriz.

Tablo 6.2. *MAP sonuçları*

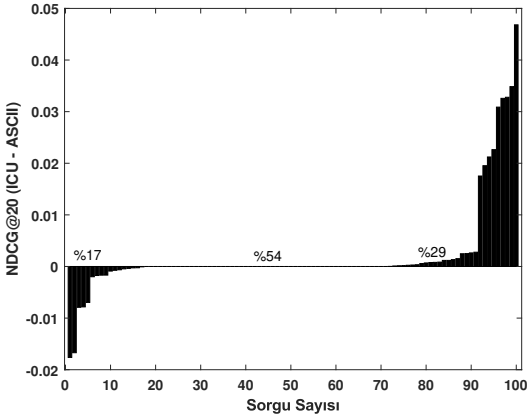
Model	İndeks	Gov2	MQ07	MQ08	MQ09	CW09A	CW09B	CW12A	CW12B	NTCIR
BM25	ICU	**0.24109**	**0.19057**	**0.26144**	0.19756	**0.06386**	**0.08794**	0.09839	0.02724	**0.25545**
	ASCII	0.24100	0.19052	0.26120	0.19756	0.06371	0.08793	**0.09855**	**0.02729**	0.25522
DFIC	ICU	**0.25040**	0.16659	**0.17449**	0.20014	0.05481	0.10219	0.11336	0.03707	0.30645
	ASCII	0.25039	**0.16697**	0.17445	**0.20032**	**0.05484**	**0.10220**	**0.11345**	**0.03723**	**0.30658**
DLH13	ICU	**0.23007**	**0.18945**	0.23367	0.19685	0.05265	**0.07425**†	**0.08701**	**0.02404**	**0.24243**
	ASCII	0.22985	0.18938	**0.23370**	**0.19701**	0.05265	0.07419	0.08697	0.02399	0.24183
LGD	ICU	**0.26336**	**0.20991**	**0.28115**†	0.20683	0.05921	**0.09050**	0.12466	0.03659	**0.28793**
	ASCII	0.26335	0.20991	0.28093	**0.20696**	**0.05924**	0.09047	**0.12474**	**0.03663**	0.28770
LM	ICU	**0.27773**	0.20996	**0.25347**	0.19935	0.05124	**0.09953**	0.14485	**0.04789**	0.32168
	ASCII	0.27770	**0.21000**	0.25336	**0.19947**	0.05124	0.09952	**0.14500**	0.04783	**0.32171**
PL2	ICU	**0.21252**	**0.16454**	**0.17663**†	0.19083	**0.05506**	**0.08341**†	0.10904	0.03347	**0.26497**
	ASCII	0.21247	0.16451	0.17646	**0.19096**	0.05504	0.08337	**0.10911**	**0.03347**	0.26470
DPH	ICU	**0.27499**	**0.22241**	**0.28542**	0.23636	0.07833	**0.10325**	0.13128	0.03469	**0.28314**
	ASCII	0.27495	0.22232	0.28529	**0.23639**	**0.07834**	0.10322	**0.13150**	**0.03478**	0.28275
DFRee	ICU	**0.26157**	0.21052	**0.28068**	0.22594	0.06915	0.09215	0.11494	0.03083	**0.27794**
	ASCII	0.26148	**0.21054**	0.28057	**0.22615**	**0.06917**	**0.09217**	**0.11506**	**0.03095**	0.27784



(a) MQ08-ndcg@20-bm25



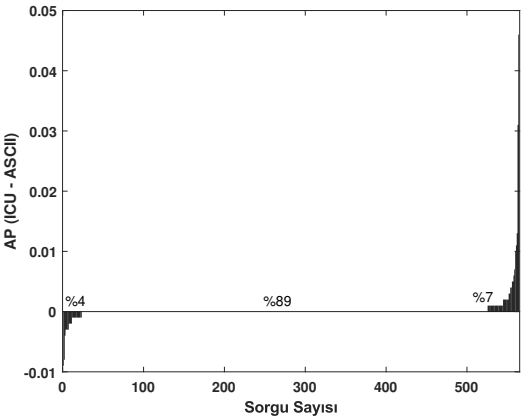
(b) MQ09-ndcg@20-bm25



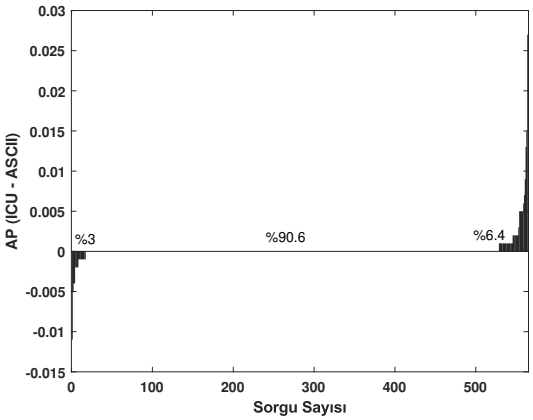
(c) NTCIR-ndcg@20-dph

Şekil 6.1. *NDCG@20 Risk grafikleri*

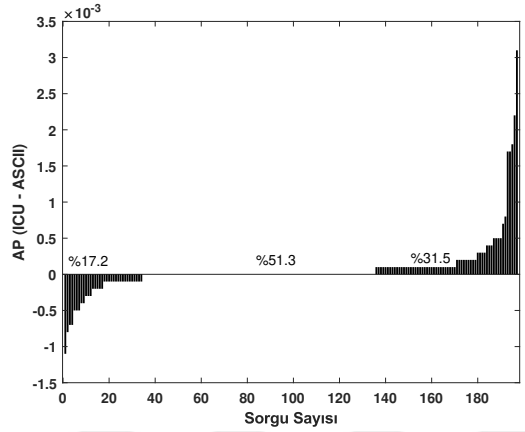
Şekil 6.1 ve Şekil 6.2 'deki grafikler ICU ve ASCII arasındaki farkın istatistiksel olarak anlamlı olduğu sonuçlara ait dağılımı göstermektedir. Bu grafikler, çoğu sorguda ICU ve ASCII değerlerinin eşit geldiğini ancak ICU'nun ASCII'den daha iyi olduğunu göstermektedir. Özellikle NDCG@20 de ICU'nun MQ08 ve MQ09 sorgularında BM25 modeli için ve NTCIR sorgularında DPH modeli için, MAP de MQ08 sorgularında LGD ve PL2 modelleri için ve ClueWeb09B sorgularında DLH13 ve PL2 modelleri için grafiklerin sağ tarafındaki sorgularda pozitif yönde puan farkları olduğu gözlemlenmektedir.



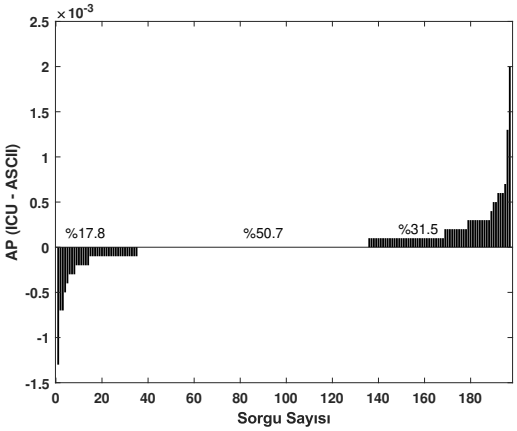
(a) MQ08-map-lgd



(b) MQ08-map-pl2



(c) ClueWeb09B-map-dlh13



(d) ClueWeb09B-map-pl2

Şekil 6.2. *MAP Risk grafikleri*

7. DEĞERLENDİRME

Bu tez kapsamında yapılan çalışmada Gov2, ClueWeb09 ve ClueWeb12 Web veri kümelerinde geçen Latin dışı alfabelerle yazılmış içeriğin Bilgi Erişim başarımı üzerindeki etkisi incelenmiştir. Deneysel çalışmalarda, tüm içeriklerin (Latin ve Latin dışı içerikler dahil) tutulduğu ICU indeks ve sadece Latin alfabesi ile yazılmış içeriğin tutulduğu ASCII indeks oluşturularak, bu iki indeksin 8 farklı Bilgi Erişim modeli (BM25, DFIC, DLH13, LGD, LM, PL2, DPH, DFRee) üzerindeki başarımları karşılaştırılmıştır.

Elde ettiğimiz sonuçlara göre Latin dışı içeriklerin ön işleme aşamasında dokümanlardan atmanın Bilgi Erişim başarımını düşürdüğü gözlemlenmiştir. Burada belgenin orijinal içeriğinde ne varsa onu korumak gerektiğini, aksi halde bunu korumazsak belgenin içeriğinde var olan bir şeyi yapay bir şekilde atarak, belgenin boyunun küçülmesine sebep olduğumuzu gözlemliyoruz. Belgenin boyunu yapay olarak küçülttüğümüzde bu belgeleri daha kısa bir belgeye dönüştürmüş oluruz. Bu da ortalama belge uzunluğunu düşürmemize neden olmaktadır. Bunun etkisini belge uzunluğuna hassas olan BM25 modeli üzerinde görebiliyoruz. BM25 modelinde, ICU ile ASCII indeks kullanımına bağlı olarak iki başarım değeri arasında MQ08 ve MQ09 sorgu kümeleri üzerinde istatistiksel olarak anlamlı bir fark olmaktadır. ICU indeks kullanıldığı zaman BM25 modelinin daha iyi sonuç verdiği görülmüştür.

Eğer veri setleri düzenli bir yapıya sahip olan makale özetleri veya gazetelerden alınmış içeriklerden oluşmuş olsaydı, bunlar için sadece Latin alfabesi ile yazılmış içeriği tutan tokenization yöntemleri kullanılırdı. Ama konu Web koleksiyonları olduğunda, Web'in kaotik yapısından dolayı Web belgelerinin içeriğinde her şey (Latin veya Latin dışı içerikler) olabilmektedir. Bu yüzden bu belgelerin içeriğinde geçen her şeyi tutan bir tokenization yöntemi kullanılmalıdır.

Sonuç olarak, eğer Web koleksiyonları üzerinde çalışıyorsak, Latin dışı alfabelerle yazılan içeriklerin, ön işleme aşamasında dokümanlardan çıkarılmadan indekse dahil edilmesinin Bilgi Erişim başarımı için daha iyi olduğu görülmektedir. Başka bir deyişle, eğer bir Web sayfasında Latin dışı alfabelerle yazılmış bir içerik doğal olarak bulunuyorsa, nasıl olsa Latin alfabesi ile yazılmış sorgular geliyor diye, var olan bu içeriği atmamalıyız. Eğer belge içerik uzayını, sorgu yazım uzayına indirgersek, örneğin sadece harflerle yazılmış sorgu uzayından rakamları atarsak, aynı ASCII indeks de olduğu gibi belgeleri yapay bir şekilde kısaltarak Bilgi Erişim başarımını olumsuz yönde etkilemiş oluruz.

KAYNAKÇA

[1] Abe, T., Chiba, Y., Suoya, Li, B., Kinoshita, T., 2006. An academic information retrieval system based on multiagent framework, in: Ng, H.T., Leong, M.K., Kan, M.Y., Ji, D. (Eds.), Information Retrieval Technology, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 497–507.

[2] Allan, J., Aslam, J.A., Carterette, B., Pavlu, V., Kanoulas, E., 2008. Million Query Track 2008 Overview. Technical Report. National Institute of Standards and Technology. URL: https://trec.nist.gov/pubs/trec17/papers/MQ.OVERVIEW.pdf.

[3] Allan, J., Carterette, B., Aslam, J.A., Pavlu, V., Dachev, B., Kanoulas, E., 2007. Million Query Track 2007 Overview. Technical Report. National Institute of Standards and Technology. URL: https://trec.nist.gov/pubs/trec16/papers/1MQ.OVERVIEW16.pdf.

[4] Amati, G., 2006. Frequentist and bayesian approach to information retrieval, in: Advances in Information Retrieval. Springer Berlin Heidelberg. volume 3936 of *Lecture Notes in Computer Science*, pp. 13–24. doi:10.1007/11735106_3.

[5] Amati, G., 2009. Divergence from Randomness Models. Springer US, Boston, MA. pp. 929–932. URL: http://dx.doi.org/10.1007/978-0-387-39940-9_924, doi:10.1007/978-0-387-39940-9_924.

[6] Amati, G., Van Rijsbergen, C.J., 2002. Probabilistic models of information retrieval based on measuring the divergence from randomness. ACM Trans. Inf. Syst. 20, 357–389.

[7] Arguello, J., Diaz, F., Lin, J., Trotman, A., 2015. SIGIR 2015 workshop on reproducibility, inexplicability, and generalizability of results (RIGOR), in: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, Santiago, Chile. pp. 1147–1148. URL: http://doi.acm.org/10.1145/2766462.2767858, doi:10.1145/2766462.2767858.

[8] Azzopardi, L., Crane, M., Fang, H., Ingersoll, G., Lin, J., Moshfeghi, Y., Scells, H., Yang, P., Zuccon, G., 2017. The Lucene for information access and retrieval research (LIARR) workshop at SIGIR 2017, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, Shinjuku, Tokyo, Japan. pp. 1429–1430. URL: http://doi.acm.org/10.1145/3077136.3084374, doi:10.1145/3077136.3084374.

[9] Białecki, A., Muir, R., Ingersoll, G., 2012. Apache Lucene 4, in: Proceedings of the SIGIR 2012 Workshop on Open Source Information Retrieval, Portland, Oregon, USA. pp. 17–24. URL: http://opensearchlab.otago.ac.nz/paper_10.pdf.

[10] Buckley, C., Voorhees, E.M., 2000. Evaluating evaluation measure stability, in: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, Athens, Greece. pp. 33–40.

URL: http://doi.acm.org/10.1145/345508.345543, doi:10.1145/345508.345543.

[11] Buttcher, S., Clarke, C., Soboroff, I., 2006. The TREC 2006 Terabyte Track. Technical Report. National Institute of Standards and Technology. URL: https://trec.nist.gov/pubs/trec14/papers/TERABYTE.OVERVIEW.pdf.

[12] Carterette, B., Pavlu, V., Fang, H., Kanoulas, E., 2009. Million Query Track 2009 Overview. Technical Report. National Institute of Standards and Technology. URL: http://trec.nist.gov/pubs/trec18/papers/MQ09OVERVIEW.pdf.

[13] Catena, M., Macdonald, C., Ounis, I., 2014. On inverted index compression for search engine efficiency, in: de Rijke, M., Kenter, T., de Vries, A.P., Zhai, C., de Jong, F., Radinsky, K., Hofmann, K. (Eds.), Advances in Information Retrieval, Springer International Publishing, Cham. pp. 359–371.

[14] Clarke, C., Craswell, N., Soboroff, I., 2004. Overview of the TREC 2004 Terabyte Track. Technical Report. National Institute of Standards and Technology. URL: http://trec.nist.gov/pubs/trec13/papers/TERA.OVERVIEW.pdf.

[15] Clarke, C., Scholer, F., Soboroff, I., 2005. The TREC 2005 Terabyte Track. Technical Report. National Institute of Standards and Technology. URL: https://trec.nist.gov/pubs/trec14/papers/TERABYTE.OVERVIEW.pdf.

[16] Clinchant, S., Gaussier, E., 2009. Retrieval constraints and word frequency distributions: A log-logistic model for IR, in: Proceedings of the 18th ACM Conference on Information and Knowledge Management, ACM, New York, NY, USA. pp. 1975–1978. URL: http://doi.acm.org/10.1145/1645953.1646280, doi:10.1145/1645953.1646280.

[17] Clinchant, S., Gaussier, E., 2010. Information-based models for ad hoc IR, in: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, Geneva, Switzerland. pp. 234–241. URL: http://doi.acm.org/10.1145/1835449.1835490, doi:10.1145/1835449.1835490.

[18] Clinchant, S., Gaussier, E., 2011. Retrieval constraints and word frequency distributions a log-logistic model for IR. Information Retrieval 14, 5–25. URL: http://dx.doi.org/10.1007/s10791-010-9143-7, doi:10.1007/s10791-010-9143-7.

[19] Collins-Thompson, K., Macdonald, C., Bennet, P., Diaz, F., Voorhees, E.M., 2015. TREC 2014 Web Track Overview. Technical Report. National Institute of Standards and Technology. URL: http://trec.nist.gov/pubs/trec23/papers/overview-web.pdf.

[20] Doyle, L.B., Becker, J., 1975. Information retrieval and processing. Melville Pub. Co, Los Angeles, USA.

[21] Fuhr, N., 2018. Some common mistakes in IR evaluation, and how they can be avoided. SIGIR Forum 51, 32–41. URL: http://doi.acm.org/10.1145/3190580.3190586, doi:10.1145/3190580.3190586.

[22] Hiemstra, D., 2000. Using language models for information retrieval. Ph.D. thesis. University of Twente, Netherlands. URL: http://eprints.eemcs.utwente.nl/6563/01/thesis.pdf.

[23] Ishida, R., 2010. Character encodings: Essential concepts. URL: https://www.w3.org/International/articles/definitions-characters/. (Erişim tarihi: 16.04.2019).

[24] Järvelin, K., Kekäläinen, J., 2002. Cumulated gain-based evaluation of IR techniques. ACM Trans. Inf. Syst. 20, 422–446.

[25] Jiang, J., Zhai, C., 2007. An empirical study of tokenization strategies for biomedical information retrieval. Information Retrieval 10, 341–363. doi:10.1007/s10791-007-9027-7.

[26] Kocabaş, I., Dinçer, B.T., Karaoğlan, B., 2014. A nonparametric term weighting method for information retrieval based on measuring the divergence from independence. Information Retrieval 17, 153–176.

[27] Kreyszig, E., 1970. Introductory Mathematical Statistics. John Wiley.

[28] Krovetz, R., 1993. Viewing morphology as an inference process, in: Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, Pittsburgh, Pennsylvania, USA. pp. 191–202. URL: http://doi.acm.org/10.1145/160688.160718, doi:10.1145/160688.160718.

[29] Lafferty, J., Zhai, C., 2001. Document language models, query models, and risk minimization for information retrieval, in: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, New Orleans, Louisiana, USA. pp. 111–119.

[30] Lin, J., Crane, M., Trotman, A., Callan, J., Chattopadhyaya, I., Foley, J., Ingersoll, G., Macdonald, C., Vigna, S., 2016. Toward reproducible baselines: The open-source IR reproducibility challenge, in: Advances in Information Retrieval: 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20-23, 2016. Proceedings. Springer International Publishing, Cham, pp. 408–420. URL: http://dx.doi.org/10.1007/978-3-319-30671-1_30, doi:10.1007/978-3-319-30671-1_30.

[31] Liu, T.Y., 2009. Learning to rank for information retrieval. Foundations and Trends® in Information Retrieval 3, 225–331. URL: http://dx.doi.org/10.1561/1500000016, doi:10.1561/1500000016.

[32] Lui, M., Han Lau, J., Baldwin, T., 2014. Automatic detection and language identification of multilingual documents. Transactions of the Association for Computational Linguistics 2, 27–40. doi:10.1162/tacl_a_00163.

[33] Luo, C., Sakai, T., Liu, Y., Dou, Z., Xiong, C., Xu, J., 2017. Overview of the NTCIR-13 We Want Web Task, in: Proceedings of NTCIR-13, pp. 394–401.

[34] Macdonald, C., Santos, R.L., Ounis, I., He, B., 2013. About learning models with multiple query-dependent features. ACM Trans. Inf. Syst. 31, 11:1–11:39. URL: http://doi.acm.org/10.1145/2493175.2493176, doi:10.1145/2493175.2493176.

[35] Manning, C.D., Raghavan, P., Schütze, H., 2008. Introduction to Information Retrieval. Cambridge University Press, New York, NY, USA. URL: http://www-nlp.stanford.edu/IR-book/.

[36] Metzler, D., Kurland, O., 2012. Experimental methods for information retrieval, in: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, Portland, Oregon, USA. pp. 1185–1186. URL: http://doi.acm.org/10.1145/2348283.2348534, doi:10.1145/2348283.2348534.

[37] Ponte, J.M., Croft, W.B., 1998. A language modeling approach to information retrieval, in: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, Melbourne, Australia. pp. 275–281. URL: http://doi.acm.org/10.1145/290941.291008, doi:10.1145/290941.291008.

[38] Porter, M.F., 1997. An algorithm for suffix stripping, in: Spärck Jones, K., Willett, P. (Eds.), Readings in Information Retrieval. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 313–316. URL: http://dl.acm.org/citation.cfm?id=275537.275705.

[39] Ricardo, B.Y., Berthier, R.N., 1999. Modern Information Retrieval. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[40] Robertson, S., Zaragoza, H., 2009. The probabilistic relevance framework: BM25 and beyond. Foundations and Trends® in Information Retrieval 3, 333–389. URL: http://dx.doi.org/10.1561/1500000019, doi:10.1561/1500000019.

[41] Robertson, S.E., 1997. The probability ranking principle in IR. Journal of Documentation 33, 294–304.

[42] Robertson, S.E., Jones, K.S., 1976. Relevance weighting of search terms. Journal of the American Society for Information Science 27, 129–146. URL: http://dx.doi.org/10.1002/asi.4630270302, doi:10.1002/asi.4630270302.

[43] Robertson, S.E., Walker, S., 1994. Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval, in: Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Springer-Verlag New York, Inc., New York, NY, USA. pp. 232–241. URL: http://dl.acm.org/citation.cfm?id=188490.188561.

[44] Roy, D., Mitra, M., Ganguly, D., 2018. To clean or not to clean: Document preprocessing and reproducibility. J. Data and Information Quality 10, 18:1–18:25.

[45] Salton, G., McGill, M.J., 1986. Introduction to Modern Information Retrieval. McGraw-Hill, Inc., New York, NY, USA.

[46] Singh, V., Saini, B., 2014. An effective tokenization algorithm for information retrieval systems. doi:10.5121/csit.2014.4910.

[47] Tax, N., Bockting, S., Hiemstra, D., 2015. A cross-benchmark comparison of 87 learning to rank methods. Information Processing & Management 51, 757 – 772. URL: http://www.sciencedirect.com/science/article/pii/S0306457315000862, doi:http://dx.doi.org/10.1016/j.ipm.2015.07.002.

[48] Uysal, A.K., Gunal, S., 2014. The impact of preprocessing on text classification. Information Processing & Management 50, 104 – 112. URL: http://www.sciencedirect.com/science/article/pii/S0306457313000964, doi:https://doi.org/10.1016/j.ipm.2013.08.006.

[49] Valizadegan, H., Jin, R., Zhang, R., Mao, J., 2009. Learning to rank by optimizing NDCG measure, in: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C., Culotta, A. (Eds.), Advances in Neural Information Processing Systems 22. Curran Associates, Inc., pp. 1883–1891. URL: http://papers.nips.cc/paper/3758-learning-to-rank-by-optimizing-ndcg-measure.pdf.

[50] Verma, M., Yılmaz, E., Mehrotra, R., Kanoulas, E., Carterette, B., Craswell, N., Bailey, P., 2016. Overview of the trec tasks track 2016, in: TREC.

[51] Voorhees, E.M., 2002. The philosophy of information retrieval evaluation, in: Revised Papers from the Second Workshop of the Cross-Language Evaluation Forum on Evaluation of Cross-Language Information Retrieval Systems, Springer-Verlag, London, UK. pp. 355–370. URL: http://dx.doi.org/10.1007/3-540-45691-0_34, doi:10.1007/3-540-45691-0_34.

[52] Voorhees, E.M., Harman, D.K., 2005. TREC: Experiment and Evaluation in Information Retrieval (Digital Libraries and Electronic Publishing). The MIT Press.

[53] Yılmaz, E., Verma, M., Mehrotra, R., Kanoulas, E., Carterette, B., Craswell, N., 2015. Overview of the trec 2015 tasks track, in: TREC.

[54] Zhai, C., Lafferty, J., 2004. A study of smoothing methods for language models applied to information retrieval. ACM Trans. Inf. Syst. 22, 179–214. URL: http://doi.acm.org/10.1145/984321.984322, doi:10.1145/984321.984322.

EK-1. ClueWeb12A veri setinde en çok geçen Latin dışı alfabelerin ilk 10 terimlerinin frekans dağılımları

Terim	Toplam Terim Frekansı	Doküman Frekansı	Terim	Toplam Terim Frekansı	Doküman Frekansı
	Arap			**Ermeni**	
العربية	1,678,706	1,361,264	հայերեն	345,261	273,063
عربي	1,258,457	1,180,296	վ	332,409	124,273
فارسی	610,295	494,041	դ	373,150	122,916
و	1,229,051	259,128	ե	643,316	116,659
من	977,297	231,882	ւ	284,374	110,130
اردو	289,659	212,683	ծ	260,598	103,860
في	933,659	205,725	ա	276,145	99,957
الله	574,433	191,560	ղ	248,088	99,041
على	465,411	149,290	խ	184,036	69,042
ما	287,678	116,061	ժ	229,350	62,384
	Kiril			**Devanagari**	
русский	10,434,695	7,574,627	हिन्दी	4,378,550	4,305,767
українська	4,046,817	3,951,376	मराठी	545,780	536,021
беларуская	2,816,374	2,739,013	हिंदी	159,700	135,707
български	1,753,855	1,599,375	ॐ	138,693	69,994
и	7,533,056	1,528,168	नेपाली	52,168	42,187
в	6,128,249	1,331,034	में	163,953	31,220
македонски	1,159,283	1,141,139	महाराष्ट्र	34,862	29,858
я	3,870,214	1,127,176	टाइम्स	99,303	29,256
с	3,300,274	1,041,949	नवभारत	33,840	29,146
език	1,034,403	970,139	इकनॉमिक	31,466	29,127
	Gürcü			**Yunan**	
ღ	2,236,440	336,100	ελληνικά	5,291,740	5,102,781
ქართული	305,660	291,773	α	7,934,439	1,161,069
ლ	167,825	52,960	τ	1,609,219	881,235
ვ	50,494	24,369	σ	4,220,593	666,927
ღღ	37,730	15,353	ι	2,587,174	585,407
ე	24,551	11,964	υ	3,065,852	560,383
ბ	26,635	11,674	ω	1,174,622	485,430
საქართველო	8,391	7,946	ε	2,298,971	343,703
მარგალური	6,399	6,388	δ	872,332	336,515
ღღღ	12,836	5,686	ν	1,054,208	324,708
	Kore			**İbrani**	
한국어	4,058,199	3,800,221	עברית	3,191,200	3,081,487
한국	146,354	108,785	יידיש	165,414	157,102
킹	169,819	98,663	נ	326,726	117,207
우리말	83,839	83,605	ע	241,654	108,415
수	169,791	64,322	פ	372,625	88,413
이	133,229	60,124	ץ	372,946	77,451
대한민국	66,348	57,113	ישראל	113,388	68,54
로그인	70,978	54,883	ר	222,77	65,488
한글	58,567	51,359	ס	160,18	62,634
월	226,305	49,281	של	390,184	61,235
	Japon			**Thai**	
日本語	9,485,692	8,695,239	ไทย	2,750,046	2,534,289
中文	13,116,422	7,302,308	ภาษา	1,772,757	1,648,403
简体	5,533,573	5,317,053	ค	108,501	55,147
繁體	4,407,343	4,252,783	ที่	221,907	53,874
の	8,343,923	1,563,702	ประเทศไทย	53,803	43,307
汉语	996,225	963,016	การ	163,33	42,426
は	3,106,296	869,977	ใน	115,379	37,237
的	5,918,327	864,998	ของ	111,831	37,109
に	3,273,205	844,814	คน	64,033	35,462
を	3,780,648	840,659	ใหม่	62,109	34,777

ÖZGEÇMİŞ

Adı-Soyadı : Ahmet ALKILINÇ

Doğum Yeri ve Yılı : Afşin/KAHRAMANMARAŞ, 1992

E-posta : ahmetalkilinc@eskisehir.edu.tr

Egitim

- Eskişehir Osmangazi Üniversitesi, Bilgisayar Mühendisliği, Haziran 2014
- Elbistan Anadolu Lisesi, Haziran 2009

Meslek

- Araştırma Görevlisi, Anadolu Üniversitesi, Eskişehir, Türkiye 2017
- Araştırma Görevlisi, Mersin Üniversitesi, Mersin, Türkiye 2019

Uluslararası Toplantılara Sunulan Bildiriler

- Alkılınç, A., Arslan, A., "A Comparison of Recent Information Retrieval Term Weighting Models Using Ancient Datasets," *International Conference on Artificial Intelligence and Data Processing (IDAP)*, Malatya, 2018.
- Arslan, A., Alkılınç, A., and Dinçer, B. T. , "Büyük Veri Setlerinde Varlık Tanıma En Sık Geçen E Posta Web Adreslerinin ve Emojilerin Tespit Edilmesi," *6. Uluslararası Mühendislik ve Bilim Alanında Yenilikçi Teknolojiler Sempozyumu*, Antalya, 2018.