VARLIK İSİMLERİ ARASINDAKİ İLİŞKİLER KULLANILARAK HABERLERİN ÖBEKLENMESİ

SALİH ATILAY OTO

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ

TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ARALIK 2012
ANKARA

Fen Bilimleri Enstitü onayı

_____

Prof. Dr. Ünver KAYNAK
Müdür

Bu tezin Yüksek Lisans derecesinin tüm gereksinimlerini sağladığını onaylarım.

_____

Doç. Dr. Erdoğan DOĞDU
Anabilim Dalı Başkanı

SALİH ATILAY OTO tarafından hazırlanan VARLIK İSİMLERİ ARASINDAKİ İLİŞKİLER KULLANILARAK HABERLERİN ÖBEKLEMESİ adlı bu tezin Yüksek Lisans tezi olarak uygun olduğunu onaylarım.

_____

Doç. Dr. Erdoğan Doğdu
Tez Danışmanı

Tez Jüri Üyeleri

Başkan : Yrd. Doç. Dr. Osman ABUL _____

Üye : Doç. Dr. Erdoğan DOĞDU _____

Üye : Yrd. Doç. Dr. Hakan Gültekin _____

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada orijinal olmayan her türlü kaynağa eksiksiz atıf yapıldığını bildiririm.

Salih Atılay OTO

Üniversitesi : **TOBB Ekonomi ve Teknoloji Üniversitesi**
Enstitüsü : **Fen Bilimleri**
Anabilim Dalı : **Bilgisayar Mühendisliği**
Tez Danışmanı : **Doç. Dr. Erdoğan Doğdu**
Tez Türü ve Tarihi : **Yüksek Lisans – Aralık 2012**

Salih Atılay OTO

VARLIK İSİMLERİ ARASINDAKİ İLİŞKİLER KULLANILARAK HABERLERİN ÖBEKLENMESİ

ÖZET

Haberlerin öbeklenmesi, birbirleriyle içerik olarak benzer olan haberlerin bir araya getirilmesi işlemidir. İnternet'in büyümesiyle çok dağıtık ve devamlı güncellenen haberlerin öbeklenmesi önemli bir problemdir. Bu problem için çok çeşitli çalışmalar yapılmaktadır. Bu çalışmaların başında Google News gelmektedir. Google News, dünya genelinde binlerce kaynaktan haberleri topladıktan sonra, birbirleriyle ilgili olanları kümeleyip kullanıcıya sunar. Haber öbekleme probleminin çözümü için en bilindik yöntem "bag of words" yöntemidir. Bu yöntem, doküman içinde geçen kelimelerin sıklıklarına bakarak öbekleme yapmaktadır. Son zamanlarda haberleri Wikipedia ve WordNet gibi kaynaklardan çıkartılan bilgilerle ilişkilendirerek öbekleme yapan yöntemler de vardır. Bu yöntemlerin "bag of words" yöntemine göre daha iyi sonuç verdiği gözlemlenmiştir. Anlamsal ağın (Semantic Web) gelişiyle birlikte internet artık bir dokümanlar ağından (web of documents), bir veri ağına (web of data) dönüşmektedir. Bu teknoloji internette aranılan bilgiye daha hızlı bir şekilde erişmemize olanak sağlayacaktır. Açık Verilerin Bağlanması (Linking Open Data) projesi, internette açık verilerin semantik web yaklaşımı ile yayınlanması ve birbirine bağlanması için geliştirilen bir projedir. Bu proje sayesinde semantik web veri ağı ve bağlı veriler (linked data) gün geçtikçe büyümektedir. Bu bağlı veriler (linked data), pek çok bilgi çıkarma ve yapay zeka uygulamasında kullanılmaya başlamıştır.

Bu tez çalışmasında, haberlerde geçen ve bağlı verilerde (linked data) bulunan varlık isimlerinin (yer, kişi, olay, vb.), birbirleriyle olan anlamsal ilişkilerini kullanarak haber öbekleme problemine yeni bir yaklaşım geliştirilmiştir. Yaptığımız testler geliştirdiğimiz yaklaşımın “bag of words” yöntemine göre daha iyi sonuçlar verdiğini göstermektedir.

Anahtar Kelimeler: Varlık İsmi, Haber öbekleme Anlamsal Ağ(Semantic Web), Açık Veri (Linked Data)

University : **TOBB Economics and Technology University**

Institute : **Institute of Natural and Applied Sciences**
Science Programme : **Computer Engineering**
Supervisor : **Associate Professor Dr. Erdoğan DOĞDU**
Degree Awarded and Date : **M.Sc. – December 2012**

Salih Atılay OTO

NEWS CLUSTERING USING RELATIONS BETWEEN NAMED ENTITIES

ABSTRACT

News clustering is the process of bringing together news articles which are similar in content. By the growth of the Internet, it is an important problem to cluster distributed and constantly updated news. There are a lot of studies for this problem. Google News is one of the major works to cluster news articles. After collecting thousands of news from the sources in the world, it provides user those that relate to each other. "Bag of wrods" is the most well-known method for solving news clustering. This method makes clustering operation by looking at the frequency of words in the document. Recently, there are some approaches for clustering documents by the information extracted from news sources such as Wikipedia and Wordnet. It is observed that these kind of methods gives beter results than "bag of words". With the advent of Semantic Web, Internet is no longer document network (web of document) and becomes data network (web of data). This technology allows us to reach information more quickly on the internet. Linking Open Data, is a project developed for the publication and interconnection of open data with the approach of the semantic web. Through this project the semantic web data network and linked data is growing day by day. Linked data was used in several information extraction and artifical intelligence application.

In this thesis, we developed a new approach to the problem of news clustering by using semantic relations of the named entites in the document. According to the experiments, it is obtained that, our approach shows beter results than "bag of word" approach.

Keywords: Named Entity, News Clustering, Semantic Web, Linked Data

TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla beni yönlendiren hocam Doç. Dr. Erdoğan DOĞDU' ya, öğrenimim süresince ücret muafiyeti ve yaşam katkı payı verdiği için TOBB ETÜ'ye ve her konuda desteklerini esirgemeyen ailem, arkadaşlarıma teşekkürü bir borç bilirim.

İçindekiler

ÇİZELGELERİN LİSTESİ

ŞEKİLLERİN LİSTESİ

1. GİRİŞ

Haberlerin öbeklenmesi, birbirleriyle içerik olarak benzer olan haberlerin bir araya getirilmesi işlemidir. İnternet'in büyümesiyle çok dağıtık ve devamlı güncellenen haberlerin öbeklenmesi önemli bir problemdir. Haberlerin öbeklenmesi veri kümesi haberler olan bir doküman öbeklenmesi problemidir. Bu problem için çok çeşitli çalışmalar yapılmaktadır. Bunların başında kelimeler çantası (bag of words) yaklaşımı ile öbekleme yapan çalışmalar vardır [61][62]. Bu yaklaşımda sadece dokümanların içerisinde geçen ortak olan kelimeler kullanılır. Bu yöntemde dokümanlar, kelime sıklıklarını içeren vektörler biçimde ifade edilirler ve çok boyutlu uzayda bu vektörlerin karşılaştırılması ile öbekleme yapılır [58].

Son yıllarda yapılan çalışmalarda sadece ortak kelime kullanmak yerine kelimeleri bir veri kaynağı (corpus) kullanarak zenginleştirilmesi, öbeklemenin kalitesini arttırdığı gözlemlenmiştir [42]. Bu veri kaynaklarının başında Wikipedia[1] ve Wordnet[2] gelmektedir. Bu tarz yaklaşımlarda dokümanlar sadece kelimeler değil kelimelerin tür, kategori bilgileri ile ya da eşanlamlı olan kelimelerin dokümana eklenmesi ile zenginleştirilir.

World Wide Web'i icat eden Tim Berners Lee tarafında ortaya atılan anlamsal ağın (semantic web) gelişiyle birlikte internet artık bir dokümanlar ağından (web of documents), bir veri ağına (web of data) dönüşmektedir. Bu teknoloji internette aranılan bilgiye daha hızlı bir şekilde erişmemize olanak sağlayacaktır [51]. Bağlı veri (Linked data[3]) ise bu veri ağını oluşturmak için yapılan bir projedir. Bu projede herkese açık olan veri kümelerinde bulunan kaynaklar birbirlerine birer etiket ile bağlanmaktadır bu sayede veri ağı sağlanmış olmaktadır ve gün geçtikçe genişleyen bir yapıdadır [57].

[1] http://www.wikipedia.org/
[2] http://wordnet.princeton.edu/
[3] http://linkeddata.org/

Bu tez çalışmasında daha önce yapılan çalışmalardan farklı olarak varlık isimlerinin (named entity) aralarında bulunan bu anlamsal bağlantılar (semantic relations) kullanarak haber öbekleme problemine farklı bir yaklaşım geliştirilmiştir. Bu tezin konusu olan "öbekleme", "haber öbekleme", ve "anlamsal web" konusunda giriş bilgileri aşağıda verilmiştir.

1.1 Öbekleme

Öbekleme, her biri sayısal ölçüm kümeleri ile tanımlanan bir grup elaman için, aynı sınıfta bulunan elemanların bazı hususlarda birbirlerine benzediği ve diğer sınıflardaki elemanlardan da farklı olacağı şekilde bir gruplandırma işlemidir [50]. Bu elemanlar çok farklı yöntemler ile organize edilebilir. Bir grup öbek bir kaç özellik ile belirtilebilir. Öbek üyelikleri özel (exclusive), örtüşme (overlapping), bulanık (fuzzy) olabilir [49]. Özel üyelikte bir eleman sadece bir öbeğe ait olabilir, örtüşmede elemanlar birden fazla öbeğe ait olabilirler, bulanık üyelikteyse bir elemanın bir öbeğe ait olma derecesi belirtilir. Öbekleme, istatistiksel veri analizi ve veri madenciliğinin ana konularından biridir. Makine öğrenme (machine learning), örüntü tanıma (pattern recognition), görüntü analizi (image analysis), bilgi çıkarımı (information retrieval), biyoinformatik (bioinformatic) gibi çeşitli alanlarda kullanılmaktadır.

Doküman öbekleme, daha genel bir alan olan veri öbekleme konusunun bir alt dalıdır. Web'in yaklaşık %80'i metin içeriklidir. Bu yüzden metin madenciliği (text mining) çok önemli bir konu haline gelmiştir. Doküman öbekleme, bilgi çekme (information retrieval) dünyasının en önemli konu başlıklarından birisidir. Gerçek dünyada kullanım alanı çok fazladır. Örnek verecek olursak arama motorları (search engines) bir sorgu için binlerce sonuç getirebilmektedir. Kullanıcılar için bu sonuç dokümanları arasındaki gezinmek ve ilgili bilgiyi bulmak zordur. Öbekleme yöntemleri gelen dokümanları otomatik bir şekilde anlamlı gruplara bölmek için kullanılır. Böylece istenilen bilgiye ulaşmak daha kolay bir hal alır.

Son yıllarda, Facebook[4], Twitter[5] gibi sosyal ağların (social network) gelişimiyle web'de içerik çok fazla artmış bulunmaktadır. Bu bilgilerin anlamlandırılması, otomatik olarak sınıflandırılması, yada bilgi çıkarımı önemli bir problem olarak ortaya çıkmaktadır. Örneğin Twitter'da günde yaklaşık 95 milyon tweet atılmakta ve öbekleme yöntemleri bu tweetleri belirli bir konu altında toplamak için kullanılmaktadır.

1.2 Haber Öbekleme

Web gibi bilgi teknolojilerinin gelişmesiyle birlikte, günümüzde çeşitli bilgi kaynaklarına web üzerinden rahat bir şekilde erişilir hale gelmiştir. Bu tezin yazıldığı 2012 yılında web'de yaklaşık 644 milyon web sitesi, 5 trilyon web sayfası ve toplamda 20.000.000 GB bilgi bulunmaktadır [66]. Bu bilgi kaynaklarından şüphesiz en önemlilerinden biri haber siteleridir. Haber siteleri çok farklı kaynaklardan gelen dağıtık ve devamlı güncellenen haberleri sunan bir yapıdadır ve şüphesiz bu belgeleri bütünleştiren sistemlere ihtiyaç vardır. Bu tarz sistemlerin çeşitli fonksiyonları içermeleri gerekir. Mesela, kullanıcı sadece ilginç veya kendi ilgi alanı ile ilgili haberlerin gösterilmesini isteyebilir. Bu nedenle sistemler kullanıcının tercihlerine göre haber seçebilmelidir. Haberlerin spor, ekonomi, teknoloji gibi kategorilere ayrılması bunu sağlayan ilk adım olarak gösterilebilir. Fakat bu tür sınıflandırma işlemleri çok genel bilgi sağlamaktadır ve ilgili haberleri bir araya toplamak için çok uygun değildir. Çünkü haberler genellikle olay odaklı olduğu için genel bir sınıflandırma işlemi, birbirleriyle ilgili haberleri kümelemesi konusunda başarılı değildir. Bu sebeple sistemler aynı olayla ilgili olan haberleri kümeleyebilmelidir. Böylece kullanıcılar aynı olayla ilgili haberleri bir arada görebilir, aynı olayı anlatan farklı bakış açılarıyla yazılan haberleri bile elde edebilir. Böylece bilgi anlamsal olarak daha bütün hale gelir ve daha istenilen kapsamlı bilgiye daha kolay bir şekilde erişilir. Google News, bu tarz sistemlere verilebilecek örneklerin başında geliyor.

[4] http://www.facebook.com/
[5] https://twitter.com/

Google News, dünya genelinde binlerce kaynaktan haberleri topladıktan sonra, birbirleriyle ilgili olanları öbekleyip kullanıcıya sunar.

1.3 Anlamsal Web (Semantic Web) ve Bağlı Veri (Linked Data)

Anlamsal Web sözü ilk olarak World Wide Web'i (www) icat eden ve World Wide Web konsorsiyumunun (W3C[6]) yöneticisi olan Tim Berners-Lee tarafından ortaya atılmıştır. Anlamsal Web, web içeriklerinin sadece doğal dillerde değil, aynı zamanda ilgili yazılımlar tarafından anlaşılabilir, yorumlanabilir ve kullanılabilir bir biçimde ifade edebileceği, böylece yazılımların veriyi kolayca bulmasını, paylaşmasını ve bilgiyi birleştirmesini sağlamayı amaçlayan gelişen bir internet eklentisidir [51]. Anlamsal Web, W3C tarafından tanımlanan protokoller ve ilişkili teknolojilerden oluşur. Bu protokollerden bazıları Kaynak Tanımlama Çerçevesi (Resource Description Framework – RDF[7]), Kaynak Tanımlama Çerçevesi Şeması (Resource Description Framework Schema– RDFS[8]) ve Web Ontoloji Dilidir (OWL[9]).

RDF, çeşitli söz dizim biçimlerinde bilgi modellemek için kullanılan genel bir üst-veri (meta-data) modelidir [52]. RDF ve diğer anlamsal veriler XML[10], N3[11], Turtle[12] gibi farklı formatlarda gösterilebilmektedir. RDFS, başka bir deyişle RDF sözlüğü, ontolojileri tanımlamak için kullanılan temel elemanları sağlar [54]. Daha açıklayıcı bir ontoloji tanımlama gereksiniminden dolayı OWL geliştirilmiştir [55].

3lü gösterim (N3) ise RDF modelinin daha iyi okunabilmesini sağlayan özne (subject), nesne (object) ve yüklem (predicate) şeklinde olan gösterim modelidir

[6] http://www.w3.org/
[7] http://www.w3.org/RDF/
[8] http://www.w3.org/TR/rdf-schema/
[9] http://www.w3.org/2004/OWL/
[10] http://www.w3.org/XML/
[11] http://en.wikipedia.org/wiki/Notation3
[12] http://www.w3.org/TeamSubmission/turtle/

[56]. SPARQL[13] ise RDF veri formatında saklanan veriyi çekmek ve işlemek için geliştirilmiş bir sorgulama dilidir. Aşağıda Anlamsal web'de kişi bilgilerinin tanımlanması için kullanılan bir sözlük (vocabulary) olan FOAF[14] kullanılarak kişilerin isim ve elektronik posta (e-mail) adreslerinin çekilmesini sağlayan sorgu gösterilmiştir:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1>
SELECT ?name ?email
WHERE{
       ?person a foaf:Person.
       ?person foaf:name ?name.
       ?person foaf:mbox ?email.
}
```

Bağlı veri ise anlamsal web vizyonuna giden bir ara basamak olarak görülebilir. Bağlı veride hedef verilerin nerede tutulduklarından bağımsız olarak RDF temelinde Web verisini birbirleriyle bağlı (linked) hale getirmektir. Bağlı verinin avantajları:

- Farklı uygulama alanlarındaki (insanlar, şirketler, kitaplar, filmler, müzikler) verilerin birbirine bağlanması ile webin gelişmesine katkıda bulunur.
- Çoklu, dağıtık ve heterojen kaynaklardan getirilen verilerin birleşmesini sağlar.
- Verilerin birbirlerine bağlanarak büyük bir ağ oluşturması sayesinde aramalara ve sorgulamalara daha etkin cevap verilmesi sağlanır.
- Ontoloji dilleri ile gerçekleştirilmeleri sayesinde sadece insanlar değil yazılımlar da bu avantajlardan faydalanır [57].

Bağlı veri (Linked data) ise doküman ağından daha çok veri ağı anlamına gelmektedir. Bağlı veri şimdiki doküman ve dokümanlar arasında bağlantı bulunan web'i çok daha ileri bir düzeye çıkarmaktadır. Buradaki link yapısı şu anki dokümanlar arasındaki anlam içermeyen bağlantı (link) yapısından biraz farklı olarak

[13] http://www.w3.org/TR/rdf-sparql-query/
[14] http://xmlns.com/foaf/spec/

bu bağlı verideki bağlantılar birer etiket içermektedir. Bu sebeple bağlı veri yapısal olarak daha güçlü ve makinelerin daha rahat anlayabileceği bir yapıdadır. Doküman ağında ya da başka bir deyişle şimdiki web'de bilgiyi arama ve bilgiye erişme işlemini dokümanları parçalayarak ve eşleşen kelime veya kelime gruplarını bularak gerçekleştiriyor. Bu haliyle doğru bilgiye ulaşmak hem çok fazla işlem hem de çok daha uzun zaman gerektiriyor. Bağlı veri hedefini hızlandırmak için ilk başta Wikipedia verisi RDF olarak dışarıya açılmıştır (DBpedia[15]). Bağlı veri ile ilgili yapılan en önemli projelerin başında Bağlı Veri (Linked Data) gelmektedir. Bu projenin asıl amacı, web'de bağlı verilerin bulunduğu veri kümelerini birbirine bağlamaktır. Şu an Bağlı veri'de (Linked data) 300'ün üzerinde veri kümesi ve bu veri kümelerinde RDF şeklinde tanımlanan milyarlarca veri bulunmaktadır [45].

DBpedia: DBpedia, Açık Bağlı Veri'de (LOD) bulunan en merkezi veri kümelerin başında gelmektedir (Şekil **1**). DBpedia, dijital ortamdaki en büyük ansiklopedi sayılan Wikipedia'da bulunan yapısal verinin (structured data) RDF veri modeli haline dönüştürülmüş şeklidir. Bu tezin yazıldığı anda 15 farklı dilde 416.000 kişi, 526.000 yer, 106.000 müzik albümü gibi 3,5 milyonun üzerinde veri bulunmaktadır.

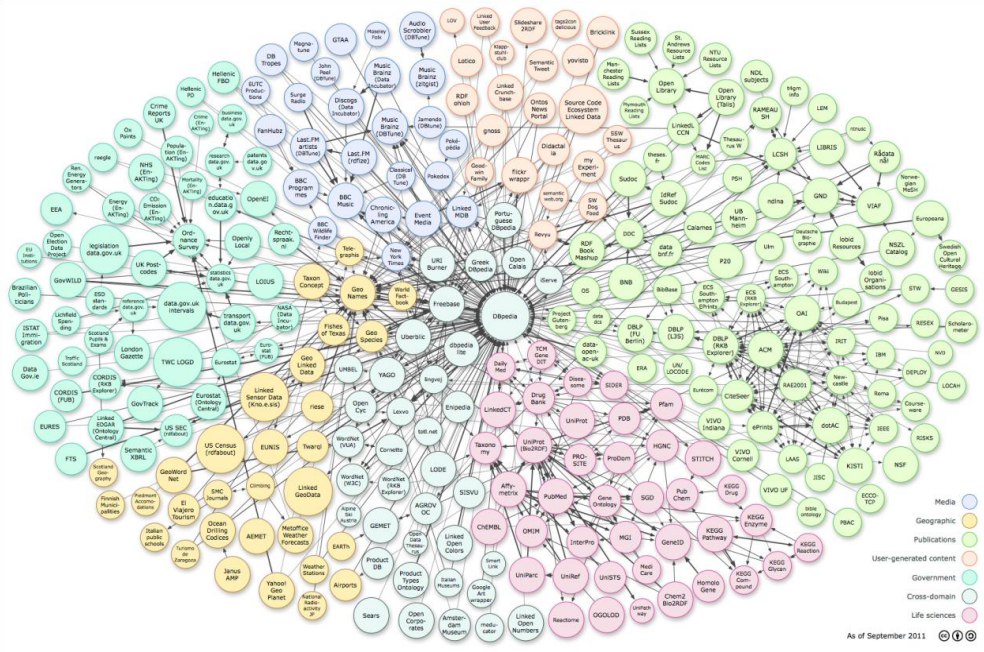
Yago: Açık Bağlı Veri'de bulunan diğer büyük veri kümelerinden biri de Max-Plank Enstitü'sü[16] tarafından geliştirilen Yago[17] veri kümesidir. Yago'da 20 milyondan fazla varlık ve bu varlıklarla ilgili 120 milyondan fazla RDF verisi bulunmaktadır. Yago özellikle varlıkların tür (type) ilişkilerinin bilgisi konusunda çok kapsayıcı bilgi sunmaktadır.

[15] http://dbpedia.org/
[16] http://www.mpg.de/en
[17] http://www.mpi-inf.mpg.de/yago-naga/yago/

1.4 Tezin Katkıları

Bu tez çalışmasında, haber makalelerini öbekleme problemi, yapılan diğer çalışmalardan farklı olarak, giderek büyümekte olan bir anlamsal web (semantic web) projesi olan Açık Veri (Linked Data) kullanılarak çözülmeye çalışılmıştır.

- Haber makalelerinde geçen varlık isimlerinin ve bu varlıkların türlerinin bulunması için bir yöntem geliştirilmiştir.
- Haber makalelerinin varlık isim ve türleri kullanılarak öbeklenmesi için bir yöntem geliştirilmiştir.
- Geliştirilen yöntemlerin performans değerlendirmesi için tarayıcı eklentisi geliştirilmiştir.
- Ayrıca web'den bir haber makale veri kümesi oluşturulmuş (download link ver) ve bu verilerle geliştirilen yöntemler test edilmiş ve sonuçların analizi yapılmıştır.



Şekil 1. LOD'nin 2011'de görünümü[18]

[18] http://rivuli-development.com/linked-data/

Tezin 2. bölümde, doküman öbekleme ile ilgili yapılan çalışmalar, dokümanların gösterim modelleri, öbekleme yöntemlerinden bahsedilmiştir. 3.bölümde, haber makalelerinin öbeklenmesi için geliştirdiğimiz yöntem açıklanmıştır ve sonrasında bir örnek senaryo verilmiştir. 4. bölümde, kullanıcı veri kümesini oluşturmak için geliştirdiğimiz tarayıcı eklentisi ve sunucu tarafının mimarisi açıklanmıştır. 5. bölümde, veri kümeleri üzerinde öbekleme çalışmaları yapılmıştır ve performans analizleri gösterilmiştir. 6. bölümde ise elde edilen sonuçlar değerlendirilmiştir.

2. İLİŞKİLİ ÇALIŞMALAR

Haber makalelerin öbeklenmesi, tanım kümesi haberler olan bir doküman öbekleme problemidir. Haber veya benzer dokümanları öbeklemek için çeşitli çalışmalar yapılmıştır. Çoğu doküman öbekleme çalışması kelime çantası (bag of words) yaklaşımına dayanır [61][62]. Kelime çantası yaklaşımı kelimeleri, özellik (feature) olarak belirtip, her dokümanı çok boyutlu kelime uzayında, birer kelime sıklığı vektörü olarak göstermeyi amaçlar [58]. Doküman içerisinde geçen terimlerin sadece o dokümanda bulundukları şekilleriyle öbekleme yapmayı amaçlamıştır. Herhangi bir anlamsal ilişki (tür, kavram, kategori, eş anlamlılık vs.) içermezler. Kelime çantası yöntemindeki en önemli sorunlardan biri seyrekliktir (sparsity). Çoğu doküman sözlük terimlerinin %5'inden azına sahiptir [59]. Diğer bir önemli sorun ise gürültüdür (noise). Web sayfaları, konuşma kayıtları (chat logs) veya elektronik posta gibi dokümanlarda çok fazla heceleme hatası (spelling error) ve kısaltmalar mevcuttur [60].

Yapılan son çalışmalarda "bag of words" yöntemi yerine anlamsal ilişkileri içeren yöntemler tercih edilmiştir. Bu anlamsal ilişkileri elde edebilmek için Wordnet, Wikipedia gibi veri kaynakları (corpus) tercih edilmiştir. Veri kaynaklarının doküman öbeklemesinde kullanılması başarıyı daha da arttırdığı gözlemlenmiştir [42].

Wordnet, bir sözlük veritabanıdır. Wordnet kelimelerin kısa ve genel tanımlamalarını gösterir ve eşanlamlı olanları gruplar. Ve aynı zamanda bu eşanlam gruplarının birbirleriyle olan çeşitli anlamsal ilişkilerini (semantic relations) de belirtir. Bu ilişkilerden bazıları:

- Kapsayıcılık (hypernyms): Y X' i kapsar, eğer X Y'ni bir türüyse (hayvan, kuşu kapsar).
- Ait olma (hyponyms): Y X'e aittir, eğer Y X'in bir alt türü ise (kuş, hayvanın bir alt türüdür).

- İçerme (holonym): Y X'i içerir, eğer X'in içinde Y varsa (Arabanın içinde direksiyon vardır).
- Bulunma (meronym): Y X'in içinde bulunur, eğer Y X'in içinde varsa (Direksiyon araba içerisinde vardır).

Wordnet ontolojisi kullanarak doküman öbeklemesi yapan çeşitli çalışmalar bulunmaktadır [43][44]. Wordnet'te varlık isimleri bulunmamaktadır ve kelimelerin sadece dilbilgisel ilişkilerini taksonomi şeklinde gösterir.

Budanitsky vd. göre, kelimeler arasındaki benzerliği Wordnet taksonomisindeki uzaklığa göre hesaplamıştır [43]. Bazı çalışmalarda ise öbekleme Wikipedia ontolojisi kullanılarak gerçekleştirilmiştir [9-12].
Strube vd. ise dokümanda geçen w_1, w_2 kelimeleri alıp bu kelimelerin Wikipedia konu başlıklarında geçen makalelerini (p_1, p_2) bulduktan sonra bu iki kelime arasındaki benzerlik, elde edilen p_1, p_2 makaleleri arasındaki benzerlikle hesaplanır. Bu benzerlik metin benzerliği ve makaleleri ait oldukları kategorilerin Wikipedia kategori hiyerarşisindeki yol uzaklığı (path distance) ile hesaplanır [12].

Gabrilovich vd. dokümanda geçen her kelimenin Wikipedia'da ait oldukları kavramları (concept) bulup, öbekleme işlemini kavram vektörlerinin kosinüs benzerliklerini karşılaştırarak yapar. Bu yöntemde Wikipedia kavramlarını bulmak için kelime, cümle, alt cümle, paragraf ya da tüm doküman kullanılabilmektedir [10].
Hu vd., Gabrilovich vd. ile benzer bir yöntem kullanarak öbekleme yapmıştır. Gabrilovich'den farkı sadece kelime odaklı çalışmaktadır. Fakat her kelimenin kendisinin term-frequency inverse-document-frequency (tf-idf) değeri ile ağırlıklandırılır. Böylece kelimelerin karşılık gelmiş oldukları Wikipedia kavramları da ağırlıklandırılmış olur [9].

Bağlı veri (Linked Data), kullanarak öbekleme yapan çalışmalar da mevcuttur. Szczuka vd. DBpedia veri kümesi üzerinde çalışmıştır [46]. Dbpedia Wikipedia'daki bilgiden yapısal bir içerik oluşturup web'de bu yapısal içeriğin ve ilişkilerinin

sorgulanabilmesini sağlayan bir veri kümesidir. Szczuka vd. de Gabrilovich gibi doküman içindeki kelimelerin DBpedia kavram makalelerini bulduktan sonra kavram vektörlerin kosinüs benzerlikleri ile öbekleme yapmıştır. Kelimelerin ilgili oldukları kavramları bulmak için gerekli olan makale metinleri Dbpedia ontolojisindeki özet (abstract) bilgisi (DBpedia property[19]) kullanılarak elde edilmiştir.
Aşağıdaki bölümlerde doküman öbeklemesi için kullanılan yöntemler ayrıntılı olarak açıklanmıştır.

2.1 GÖSTERİM MODELLERİ

Gösterim modeli genel anlamda bir nesnenin nasıl ifade edildiğini ifade eder. Bir doküman normalde konusu, cümleler ve bu cümleleri oluşturan kelimelerden oluşur. Bir insan normalde dokümanın bu gösterim şekliden dokümanın ne anlattığını gayet net bir şekilde anlayabilir ve benzer şekilde haberleri basit bir şekilde gruplayabilir. Fakat makineler bu şekilde çalışmazlar bunun için makinelerin anlayabileceği gösterim modelleri geliştirilmiştir.

Doküman içinde geçen terimleri, kelime çantası (bag of words) varsayıp diğer dokümanlarla karşılaştırmak en bilinen yöntemlerden biridir [41]. Fakat kelimeleri bir çanta içine atıp karşılaştırma yöntemi yeterli değildir. Bu bölümde dokümanların gösterimi için geliştirilmiş farklı gösterim modelleri anlatılacaktır.

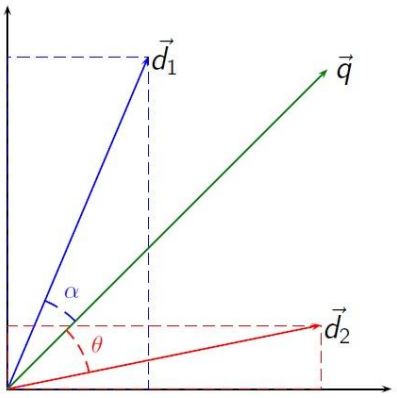
2.1.1 Vektör Uzayı Modeli (Vector Space Model)

1970'lerin başında Salton vd. tarafından otomatik dizinleme yapmak için kullanılan yöntem şu an doküman öbekleme için kullanılan standart yöntemlerden biri haline gelmiştir [3]. Vektör uzayı modeli metin dokümanları ya da herhangi bir nesneyi

[19] http://dbpedia.org/ontology/abstract

vektörler cinsinden ifade edebilen bir cebirsel yöntemdir. Genellikle bilgi filtreleme (information filtering), bilgi çekme (information retrieval), dizinleme gibi çeşitli olanlarda kullanılmaktadır.

Dokümanlar, $d_j = (w_{1,j}, w_{2,j}, ..., w_{t,j})$, sorgular da, $q = (w_{1,q}, w_{2,q}, \dots, w_{s,q})$, gibi vektör şeklinde gösterilebilir. w, j dokümanındaki ve q sorgusundaki kelimeleri gösterir. Vektör işlemleri dokümanlarla arama sorgularını (q) karşılaştırmak için kullanılır. Her boyut farklı bir terim olarak ifade edilir. Eğer bir terim doküman içinde geçiyorsa onun değeri sıfırdan farklı bir değerdir. Terimleri ağırlıklandırmak için çok çeşitli yöntemler geliştirilmiştir. Bu yöntemler arasında en fazla bilineni terim sıklığı, ters doküman sıklığı (tf-idf) ağırlıklandırma yöntemidir. Terimin anlamı uygulamaya göre değişebilir. Bazen tek bir kelime, anahtar kelime veya daha uzun cümle parçacıkları da olabilir. Eğer terimler kelime olarak seçilirse dokümanın vektörünün boyut sayısı doküman içinde geçen tekil kelime sayısı kadardır. Dokümanların birbirleriyle olan benzerlikleri her doküman vektörünün birbirleriyle ve orijinal sorgu vektörü q ile yaptığı açıların karşılaştırılması ile hesaplanır (Şekil **2**). Açı hesaplama yerine vektörler arasındaki oluşan açının kosinüsünü hesaplamak daha kolaydır [63].



Şekil 2. Vektör gösterimi[20]

[20] http://en.wikipedia.org/wiki/File:Vector_space_model.jpg

$$\cos\theta = \frac{d_2 * q}{||d_2|| * ||q||} \quad (2.1)$$

$d_2 * q$ doküman ile orjinal sorgu vektörülerinin kesişimidir (nokta çarpımı). $||d_2||$, d_2 vektörünün mutlak halidir. $||q||$ ise q vektörünün mutlak halidir.
Vektörlerin mutlak halinin hesaplanması ise aşağıdaki gibi gösterilir.

$$||q|| = \sqrt{\sum_{i=1}^{n} q_i^2} \quad (2.2)$$

Terim sıklığı – ters doküman sıklığı (tf-idf) ağırlıklandırması

Model, terim sıklığı – ters doküman sıklığı şeklinde bilinir. Doküman içindeki terimlerin ağırlığı bazı yerel ve global parametrelerin çarpımlarıyla oluşur [3].
Bir dokümanın ağırlık vektörü $V_d = [w_{1,d}, w_{2,d}, \dots, w_{N,d}]^T$, olarak T boyutlu gösterilebilinir.

$$w_{t,d} = \mathrm{tf_{t,d}} . \log \frac{|D|}{\{\{d' \in D | t \in d'\}\}} \quad (2.3)$$

$\mathrm{tf_{t,d}}$ t teriminin d dokümanı içerisindeki sıklığını,

$\log \frac{|D|}{|\{d' \in D | t \in d'\}|}$, ters doküman sıklığını,

$|\{d' \in D | t \in d'\}|$ ise t terimini içeren doküman sayısını ifade eder.
Kosinüs kullanılarak d_j ve q arasındaki benzerlik aşağıdaki şekilde hesaplanmıştır [3].

$$sim(d_j, q) = \frac{d_j . q}{\|d_j\| . \|q\|} = \frac{\sum_{i=1}^{N} w_{i,j} w_{i,q}}{\sqrt{\sum_{i=1}^{N} w_{i,j}^2} \sqrt{\sum_{i=1}^{N} w_{i,q}^2}} \quad (2.4)$$

Vektör uzay modeline dayanan ve genişletmeye yönelik çalışmalar da mevcuttur. 2.1.5'e kadar olan bölümlerde bu çalışmalardan bahsedilecektir.

2.1.2 **Genelleştirilmiş Vektör Uzay Modeli (Generalized Vector Space Model)**

Genelleştirilmiş vektör uzay modeli, ikili grupların dikgenliğini varsayımı yerine terimlerin birbirleriyle olan ilişkilerini inceler. Burada her terim vektörü t_j m_r' in 2^n kadar kombinasyonu ile oluşur r = 1...2^n [1].

$$sim(d_k, q) = \frac{\sum_{j=1}^{n}\sum_{i=1}^{n} w_{i,k} * w_{j,q} * t_i . t_j}{\sqrt{\sum_{i=1}^{n} w_{i,k}^2} * \sqrt{\sum_{i=1}^{n} w_{i,q}^2}} \quad (2.5)$$

t_i, t_j 2^n, boyutlu uzayda birer vektördür.

Terimlerin birbirleriyle olan ilişkilerini değerlendirmek için 2 ana yol vardır:

1) Terimler arasındaki anlamsal ilişkiler,

2) Terimlerinin geniş bir metin havuzunda (corpora) birlikte bulunma (co-occurence) sıklıklarını ölçerek hesaplanabilir.

Tsatsoranis vd. genelleştirilmiş vektör uzayındaki terimler arasındaki anlamsal ilişkileri Wordnet kullanarak çıkartmıştır [2].

2.1.3 **Konu Temelli Vektör Uzay Modeli (Topic Based Vector Space Model)**

Konu temelli vektör uzay modeli genel vektör uzay modelindeki gibi terimler arasında benzerlik bulunmasına göre çalışmaz. Bu yöntemde terim vektörleri arasındaki dikgenlik (orthogonal) olma gibi bir kısıt yoktur [4]. Dikgen olma koşulu doğal dillerde eşanlamlılık ve çok ilişkili kelimeler arasında hata üretecektir. Bu sebeple bu yöntemde durdurucu kelimeler (stop-words) tespiti, kök bulma ve Wordnet gibi genel sözlük kullanma metotları kullanılır. Bu yöntemde uzaydaki her boyut esas bir konuyu belirtir. Terim vektörü *t'*nin uzayda özel bir ağırlığı vardır. Önemli terimlerin ağırlıkları fazla, durdurucu kelimeler ve ilgisiz kelimelerin ağırlığı daha azdır. Doküman terim vektörlerinin toplamı şeklinde gösterilebilir. Geliştirilmiş konu temelli vektör uzay modelinde Polyvyanyy vd. terim vektörlerinin bir ontoloji

(Wordnet) kullanarak oluşturulması yöntemi denenmiştir [5]. Doküman benzerlikleri hesaplama konusunda başarılı olduğu gözlemlenmiştir.

2.1.4 **Gizli Anlamsal Analiz ((Latent Semantic Analysis))**

Gizli Anlamsal Analiz, doğal dil işlemede kullanılan doküman ve terimlerin kavramlarını (concept) çıkartarak dokümanlar arasındaki benzerliği bulmayı amaçlayan yöntemdir. LSA'da benzer anlamlı terimlerin metinin benzer parçalarında yer alması varsayılır. Tekil değer ayrışımı (Singular value decomposition) yöntemi kullanılarak büyük bir metinden her paragraftaki kelime sıklıklarının tutan bir matris (satırlar kelimeleri sütunlar ise her bir paragrafı temsil eder) oluşturulur. Kelimeler arasındaki benzerlik her satırdaki oluşan vektörlerin kosinüs açılarından hesaplanır. Değeri 1 yakın olanlar çok benzer kelimeleri, değeri 0'a yakın olanlar ise az benzer kelimeleri ifade eder [18].

2.1.5 **Terim Ayrıştırması (Term Discrimination)**

Bu metot tf-idf metoduna benzemektedir. Fakat daha çok, bilgi çıkarma (information retrieval) için uygun olan kelimelerin tespit edilmesi önemlidir. Varoluş matrisinden (occurrence matrix) daha az yoğun matris kullanılmasını hedefler. En uygun dizin terimi iki tane farklı dokümanı birbirinden ayırabilen ve iki tane benzer dokümanı birbiriyle ilişkilendirebilen değerdir. Fakat yarı uygun dizin terimi iki benzer dokümandan iki farklı dokümanı ayırt edemez.

Ayrıştırma değeri, varoluş matrisinin vektör-uzay yoğunluğundan, aynı varoluş matrisinin dizin terim yoğunluğunun çıkarılması ile elde edilir [19].

A varoluş matrisi, A_k, *k* dizin terimi olmadan oluşturulan varoluş matrisi ise, *Q(A),* *A*'nın yoğunluğu ise, ayrıştırma değeri, $DV_k = Q(A) - Q(A_k)$ olur.

2.1.6 **Çizge Modeli (Graph Model)**

Vektör uzay modeli dokümanların gösterilmesinden çok popüler bir yöntem olmasına karşın, bazı yazarlar doküman gösterim modeli için çizge modeli gibi farklı yaklaşımlar denemiştir [6-8]. Bir çizge düğümlerden (vertice) ve kenarlardan (edge) oluşur. Genel gösterimi $G = \{V,E\}$ şeklindedir. V düğümleri, E ise kenarları temsil eder.

Hammouda vd. yönlü bir çizge olan doküman dizin çizgesi (document index graph) kullanmıştır [7]. Her düğüm doküman içerisindeki bir kelimeyi temsil etmektedir. Her iki düğüm ($v_{i,}\ v_j$) arasındaki kenar ilişkisi, eğer doküman içerisinde v_j, v_i'den sonra geliyorsa kurulur. Çizge içerisinde bulunan düğümler aynı zamanda, kelimenin bulunduğu dokümanın izini tutar. Hangi kenarın nereye gittiği ve hangi dokümanda tutulduğu farklı bir dizinde tutulur. v_1'den v_n'e uzanan yol n uzunluğunda olan cümleyi ifade eder. Aynı alt cümleleri içeren cümleler G çizgesi üzerinde ortak yollara sahiptirler. Yönlü çizge içerisinde bulunan dokümanlar arasındaki cümle parçacıklarının eşleşme oranı daha sonra dokümanların benzerliğinin hesaplanmasında kullanılır.

Zamir vd. web arama motorundan gelen web kesitleri (web snippet) için doküman dizin çizgesine benzer olan "son ek ağaç öbekleme" (suffix tree clustering) kullanmıştır [8]. Doküman dizin çizgesi yöntemindeki aynı amaçla cümle parçacıkları yapısını ya da web kesitlerini kullanır. Sonek ağacı, kökü olan yönlü bir ağaçtır. Bu yapıda aşağıdaki özellikler bulunmaktadır:

- Her iç düğümün en az iki tane alt düğümü vardır.
- Ağaç üzerindeki her kenar doküman içerisindeki biricik bir cümle parçacığıyla etiketlenmiştir.
- Bir kenardan sonra gelen kenarlar aynı kelime ile başlayamaz.
- Doküman içerisindeki her sonek s için ağaçta bir tane sonek düğümü bulunmaktadır.

- Yapraklar kendi cümle parçacığının kökeni bilgisine sahiptir.

Zamir vd. genellikle web kesitleri ile ilgilenirken, Schenker vd. gibi HTML sayfalarıyla çalışanlar da vardır [6]. Web dokümanlarını göstermek için 3 farklı yöntem ortaya koymuştur. Doküman dizin çizgesi ve sonek ağaç öbekleme yöntemlerinin aksine, her doküman yönlü bir çizge ile gösterilir. Doküman içerisindeki her tekil terim bir düğüm ile gösterilir. v_i'den v_j'ye giden her kenar sadece ve sadece metin içide terim t_j terim t_i den önce geliyorsa, oluşturulur.

2.1.7 **Olasılı Konu Modeli (Probabilistic Topic Model)**

Doküman gösterim yöntemlerinden bir diğeri de gösterim modelinin bazı rastgele işlemler sonucu üretilmiş olanlarıdır.
Doküman konuların dağılımı şeklinde oluşturulduğunu varsayalım. Bu dağılımdan rastgele bir konu seçilir. Sonra seçilen konunun benzer anlamları da bu konuya eklenir. Mesela rastgele seçimlerden tilki elde edilmiş olsun, kurt ve kurnaz kelimeleri de bu konuya eklenir. Genelde doküman bir kaç tane konunun birleşiminden oluşur. Bu gösterim modeli ilk olarak Hofmann vd. tarafından ortaya atılmıştır [21]. Daha sonraları Blei vd. tarafından değiştirilmiştir [22][23]. Konu modellemede yapılan asıl sorun doküman içindeki konu dağılımının ve konu-kelime dağılımının tam olarak bilinmemesidir. Bu bilinmeyen özellikler saklı değişken olarak adlandırılır. Doküman içindeki kelimelere bakarak arka çıkarsama yöntemleri uygulanarak gizli yapı çıkarılabilir [22].

P(z) ye *d* dokümanı içerisindeki *z* tane konunun dağılımı diyelim. Önceden bahsedildiği gibi doküman içerisindeki kelimeler bir konunun örneklemesi ve sonrasında konu-kelime dağılımı yapılarak elde ediliyordu. $P(z_i = j)$ doküman içerisinde j'ninci konu i'ninci kelimenin örneklenmesinden elde edilmesinin olasılığıdır. $P(w_i|z_i = j)$, w_i'ninci kelime j'ninci konu altında örneklendirilmiş i'ninci kelimedir

$$P(w_i) = \sum_{j=1}^{k} P(w_i|z_i = j)\, P(z_i = j) \quad (2.6)$$

k konu sayısını temsil eder. Bu işlem tüm dokümanlar için tekrar edilir.

2.2 ÖBEKLEME YÖNTEMLERİ

Doküman öbekleme, daha genel bir alan olan veri öbekleme konusunun bir alt dalıdır. Deb vd. göre, çoğu öbek algoritmaları elemanlar arasındaki uzaklık, veri uzayındaki yoğunluk, aralık veya özel istatistiksel dağılımlar kullanılarak çalışır. Bu sebeple öbekleme çok amaçlı optimizasyon ile formüle edilebilir [25]. Dokümanların öbeklenmesinde kullanılan 2 genel algoritma vardır. Bu algoritmalardan ilki hiyerarşik öbekleme algoritmasıdır. Toplayarak ya da bölerek dokümanlar hiyerarşik bir yapıda öbeklenir. Fakat bu yaklaşımda etkinlik (efficiency) problemi vardır. Diğer algoritmalar ise K-means öbekleme algoritması ve bunun türevleridir. K-means algoritması, hiyerarşik öbekleme algoritmasına göre daha etkilidir (efficiency), fakat daha az doğru sonuç (accuracy) verir [65]. Aşağıdaki bölümlerde bu algoritmalar daha ayrıntılı bir şekilde verilmiştir.
En uygun öbekle yöntemi kullanılan veri seti ve sonuçların kullanım şekillerine göre değişmektedir. Aşağıda öbekleme için kullanılan çeşitli yöntemler sıralanmıştır:

- Bağlantılı model (Conectivity model), örnek hiyerarşik öbekleme [26].
- Merkezi model (Centroid model), örnek k-means öbekleme [27].
- Dağıtık model (Distrubition model), örnek beklentiyi maksimize etme (expectation maximization) algoritmasında kullanılan çok değişkenli normal dağıtım [28].
- Yoğunluk modeli (Density model), örnek olarak DBSCAN ve OPTICS [30], [31].
- Alt uzay modeli (Subspace model), örnek olarak ikili öbekleme Biclustering, [32].

Öbekleme veri kümesindeki tüm elemanların belirli gruplar içerisinde toplanmasıdır. Hiyerarşik öbeklemede ise öbeklerin birbirleriyle olan ilişkileri de belirtilir. Öbekleme algoritmaları temel olarak iki farklı yaklaşıma dayanır [63]:

- Katı öbekleme (Hard clustering), her eleman bir kümeye ait olup olmaması ile ilgilenir
- Yumuşak öbekleme (Soft clustering), her elemanın belirli bir kümeye ait olması derecesi ile ilgilenir

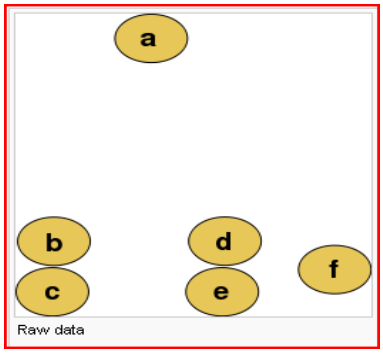
Doküman öbekleme işlemi için en çok tercih edilen yöntem hiyerarşik öbekleme ve k-means öbekleme algoritmalarıdır [33].

2.2.1 **Hiyerarşik Öbekleme**

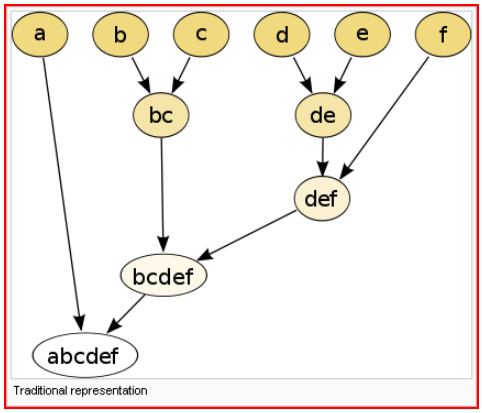
Hiyerarşik öbekleme algoritmasındaki amaç, hiyerarşi öbeği kurmaktır. Hiyerarşik öbekleme yapmak için kullanılan stratejiler 2 türlüdür [64]:

- Toplayıcı (Agglomerative): Bu yöntemde hiyerarşi aşağıdan yukarıya doğru oluşturulur. Her eleman ilk önce kendi öbeğinde başlar sonra yukarıya doğru öbekler birleşerek ilerler. En sonda bir tane öbek kalır
- Parçalayıcı (Divisive): Bu yöntemde hiyerarşi yukarıdan aşağıya doğru oluşturulur. Bütün elemanlar ilk başta bir tane öbekte başlar, özyinelemeli olarak parçalanarak her eleman kendi öbeğine yerleşir.

Hiyerarşinin sonuçları genellikle (Şekil **4**)'deki gibi dendogram üzerinde gösterilir.



Şekil 3. Sıralı Veri[21]



Şekil 4. (Şekil 3)'deki verilerin dendrogram gösterimi [22]

Genel durumda, Toplayıcı hiyerarşik öbeklemenin karmaşıklığı $O(n^3)$ dür. Bu karmaşıklı büyük veri setlerinde yavaşlığa neden olur. Parçalayıcı hiyerarşik öbekleme algoritmasının karmaşıklığı ise $O(2^n)$ dir. Bu yöntem de toplayıcı hiyerarşik öbekleme gibi yavaş çalışır. Fakat bazı özel durumlar için uygun etkili toplayıcı metotlar vardır (karmaşıklık $O(n^2)$), [34,35].

Hangi öbeklerin birleştirileceği ve hangi öbeklerin ayrıştırılacağını bilmek için elemanlar arasındaki benzeyemezlik ölçüm (dissimilarity measure) bilgilerinin olması lazım. Hiyerarşik öbeklemeyi yapmak için, elemanlar arasındaki uzaklığı ve

[21] http://upload.wikimedia.org/wikipedia/commons/b/b5/Clusters.svg
[22] http://upload.wikimedia.org/wikipedia/commons/a/ad/Hierarchical_clustering_simple_diagram.svg

öbeklerin birbirlerine bağlanma yöntemi (linkage method) gibi iki tane ölçüt lazımdır.

Uzaklık Ölçütü (Distance Metric)

Seçilen uzaklık ölçüt yöntemi oluşacak öbeklerin şekillerini etkiler. Örnek verecek olursak, (1,0) noktası ile orijin olan (0,0) noktası arası uzaklık her yöntemde 1'dir. Fakat (1,1) noktası ile (0,0) noktası arasında seçilen yönteme göre 2 (Manhattan), $\sqrt{2}$ (Öklit uzaklığı), 1 (Maksimum uzaklık) gibi uzaklıklar elde edilebilir.

Çizelge 1. Uzaklık ölçüt formülleri[23]

Ölçüt İsmi	**Formül**
Öklit Uzaklığı	$\|\boldsymbol{a}-\boldsymbol{b}\|_2 = \sqrt{\sum_i (\boldsymbol{a_i}-\boldsymbol{b_i})^2}$
Kare Öklit Uzaklığı	$\|\boldsymbol{a}-\boldsymbol{b}\|_2^2 = \sum_i (\boldsymbol{a_i}-\boldsymbol{b_i})^2$
Manhattan Uzaklığı	$\|\boldsymbol{a}-\boldsymbol{b}\|_1 = \sum_i \|\boldsymbol{a_i}-\boldsymbol{b_i}\|$
Maksimum Uzaklık	$\|\boldsymbol{a}-\boldsymbol{b}\|_\infty = \max_i \|\boldsymbol{a_i}-\boldsymbol{b_i}\|$
Mahalanobis Uzaklığı	$\sqrt{(\boldsymbol{a}-\boldsymbol{b})^T \boldsymbol{S^{-1}} (\boldsymbol{a}-\boldsymbol{b})}$, S kovaryans matrisi
Kosinüs Benzerliği	$\frac{\boldsymbol{a}.\boldsymbol{b}}{\|\boldsymbol{a}\|\|\boldsymbol{b}\|}$

[23] https://onlinecourses.science.psu.edu/stat505/book/export/html/148

2.2.1.1 Bağlama Yöntemleri (Linkage Methods)

Bağlama ölçütü öbek içindeki elemanların birbirleriyle olan uzaklık ilişkisine bakarak öbekler arasındaki uzaklığı hesaplama yöntemleridir. Böylece hangi öbeklerin birbirleriyle bağlanacağını belirler. En fazla kullanılan bağlama yöntemleri tekil bağlama, tam bağlama ve ortalama bağlama yöntemleridir. Bunların dışında:

- Tekil Bağlama (Single Linkage): en ucuz ve en anlaşılır bağlama yöntemidir. İki tane öbeğin benzerliği bu iki öbekte birbirine en yakın bulunan iki elemanı kullanılmasıyla hesaplanır. Diğer bir deyişle birbirine en yakın iki eleman öbeklerin benzerliğini gösterir. Benzerlik matrisindeki değerler sıralanarak bağlama işlemi doğrusal zamanda yapılabilir. Bu yüzden karmaşıklık $O(n^2)$ dir.
- Tam Bağlama (Complete Linkage): toplam öbek çapını minimize etmeye çalışır. Her öbekteki en uzak noktayı alır. Birleştirme işleminde ekstra işlem yapılması gerekmektedir. Karmaşıklık $O(n^2 \log n)$ dir.
- Ortalama Bağlama (Average Linkage): Sadece öbek kenarları yerine öbek içerisindeki her benzerliği hesaba katar. Diğer bir deyişle yerel benzerlik veya çap yerine öbek bağıntısını (cohesion) maksimize etmeye çalışır. Bu yöntemde benzerlik matrisinden yanında öbeklerin ortalaması bilgisine de ihtiyaç vardır. Literatürde bu yöntem grup ortalama öbekleme ya da Aritmetik Ortalama ile Ağırlıksız Parça Grup Yöntemi (UPGMA) olarak bilinir [36].

$$\frac{1}{|C_1|.|C_2|}\sum_{x \in C_1}\sum_{y \in C_2} dist(x,y) \tag{2.7}$$

C_1, C_2 iki farklı öbeği, x ve y ise bu öbeklerdeki elemanları gösterir.

Toplamalı hiyerarşik öbeklemenin asıl gücü hesaplanmasıdır. Oluşan ağaçta her düzeyde hiyerarşik olarak öbekle oluşmaktadır. Bu yöntemde önceden bir öbek sayısı

tanımlamasına gerek yoktur. Fakat dezavantajı çok fazla hafıza tutmasıdır. Bu sorun için bazı çözüm yolları geliştirilmiştir [37].

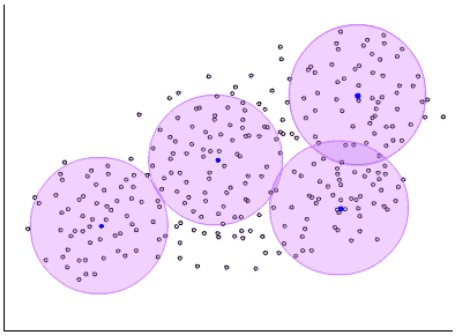
2.2.2 **K-means Öbekleme**

Veri madenciliğinde, k-means öbekleme n tane elemanı k tane öbek içerisinde gruplama işlevidir. Burada her eleman öbek merkezi (centroid) kendisine en yakın olan öbeğe ait olur. Her merkez dışarıda hiçbir eleman kalmayacak şekilde döngüsel bir yapıda güncellenir. Bu merkezler girdi elemanlarından hiçbirisine işaret etmez. Bu merkezlerin bir elemana işaret etmesini gerektiriyorsa bu algoritma k-medioid olur [67].

Girdi elemanları $(x_1, x_2, \dots, x_n)$ ve her eleman da d boyutlu bir vektör olsun. K-means öbekleme n tane elemanı k tane öbeğe yerleştirmeyi amaçlar. $(k \leq n)$, $S = \{S_1, S_2, \dots, S_k\}$ öbekleri için, öbek için karelerin toplamı (within-cluster sums of squares) minimize etmek için aşağıdaki formül kullanılır [68].

$$arg\ \min_s \sum_{i=1}^{k} \sum_{x_j \in S_i} \left\| x_j - \mu_i \right\|^2 \tag{2.8}$$

μ_i, S_i'nin orta noktasıdır.



Şekil 5. Öbek merkezleri [24](centroid)

[24] https://onlinecourses.science.psu.edu/stat505/node/148

Elimizde ilk k-means öbekleri olduğunu varsayalım $m_1^{(1)}, \ldots, m_k^{(1)}$

Algoritma iki aşamadan oluşur. İlki atama adımı, ikincisi ise güncelleme adımı [38].

- Atama adımında her elemanı, öbek merkezi kendisine en yakın olan öbeğe atanır

$$S_i^{(t)} = \left\{ x_p : \left\| x_p - m_i^{(t)} \right\| \leq \left\| x_p - m_j^{(t)} \right\| \forall\, 1 \leq j \leq k \right\} \tag{2.9}$$

  Her x_p elemanı, sadece bir tane $S^{(t)}$ öbeğine atanabilir.

- Güncelleme adımında elemanların merkezleri olacak yeni orta noktalarını hesaplama işlemi yapılır.

$$m_i^{(t+1)} = \frac{1}{S_i^{(t)}} \sum_{x_j \in S_i^{(t)}} x_j \tag{2.10}$$

  Algoritma atamalarda bir değişme olmadığı zamana kadar çalışır.

Algoritmanın çalışmasında en fazla zaman yukarıdan da anlaşılacağı gibi vektör uzaklıklarını hesaplamak için harcanıyor. Her uzaklık veya benzerlik bir defa hesaplanır ve her atama adımında o anki en iyi ile karşılaştırılır. Yeni orta noktalar her vektörü sadece bir kere ele alır. Buradan da k-means doküman sayısı ve boyut sayısı ile doğru orantılı olduğu sonucuna varılır. Eğer algoritmayı i gibi bir sabit değer ile çalıştırırsak, toplam karmaşıklık $O(iknd)$ olur, d boyut sayısı, n eleman sayısını gösterir.

3. VARLIK İSİMLERİ VE İLİŞKİLERİ KULLANILARAK HABER MAKALELERİNİN ÖBEKLENMESİ

Gösterim modelleri bölümünde bahsedildiği gibi dokümanlar genellikle vektör modeli türevi ya da çizge modeli şeklinde gösterilmiştir. Geliştirdiğimiz yöntemde daha önceki çalışmaların aksine herhangi vektör modeli türevi ya da çizge modeli doküman gösterim modeli olarak seçilmemiştir. Öbekleme yapılabilmesi için bir uzaklık ya da benzerlik fonksiyonu tanımlanması gerekmektedir ve her dokümanın birbirine olan uzaklık ya da benzerlik değerinin bilinmesi lazımdır. Geliştirdiğimiz yöntemde, her iki haber arasındaki bir çizge oluşturulmuştur. Bu çizgede düğümler (node) varlık isimleri, kenarlar (edge) ise bu varlık isimleri arasındaki ilişkilerdir. Daha sonra her iki haber makalesi arasındaki benzerlik değerleri hesaplanır sonrasında toplamalı hiyerarşik öbekleme algoritması kullanılır. Aşağıdaki bölümlerde sırasıyla ilk başta haber makalelerindeki varlık isimleri, tür ve varlık isimleri arasındaki bağlantı ilişkilerinin nasıl elde edildiğinden bahsedilmiştir. Daha sonra elde edilen bu varlık isimleri ve aralarındaki ilişkiler kullanarak çizgenin oluşturulması ve kenar ağırlıklandırılmasının nasıl yapıldığı ve son olarak öbekleme açıklanmıştır.

3.1 Yöntem

Önerdiğimiz yöntem genel hatlarıyla 3 aşamadan oluşmaktadır (Şekil **6**):

Şekil 6. Yöntem akış şeması

- ***Varlık isimlerinin (named entity), türleri ve diğer varlık isimleri ile olan ilişkilerinin çıkarılması:***

  Bu tez çalışmasında haberleri kümeleme işleminin altında yatan asıl mantık şudur. Haberlerin içinde geçen varlık isimlerinin (named entity) anlamsal olarak birbirlerine olan benzerliği, haberlerin birbirlerine olan benzerliği ile ilişkilidir. Yani iki haber içinde geçen varlık isimleri birbirlerine ne kadar yakın iseler bu iki haber de birbirlerine anlamsal olarak o kadar yakındır. Bu sebeple ilk başta varlık isimlerinin anlamsal ilişkilerini çıkarmak için Açık Bağlı Veri (LOD) veri kümeleri kullanılmıştır. Burada Yago ve DBpedia veri kümeleri seçilmiştir. DBpedia'dan varlık isimlerinin URL bilgileri ve diğer varlık isimleri aralarında var olan bağlantı ilişkileri, Yago veri kümesinden ise varlık isimlerinin ait oldukları tür bilgileri çıkarılmıştır.

- ***İki haber makalesi arasındaki anlamsal çizge (semantic graph) ve Benzerlik Matrisi (Similarity Matrix) oluşturulması:***

  Her haber içinde geçen varlık isimleri, bu varlık isimlerinin tür ilişkileri ve diğer varlık isimleri ile arasında var olan bağlantı ilişkileri çıkarıldıktan sonra, her ikili haber arasında bir çizge oluşturulur. Varlık isimleri çizgede düğümlerle (node), aralarındaki ilişkiler düğümler arası kenarlarla (edge) gösterilir. Bu çizge oluşturulurken kenar ilişkileri varlık isimlerinin birbirleriyle ilgili olan anlamsal ilişkileri ağırlıklandırılarak oluşturulur. Sonrasında bu çizge içindeki kenarların değerleri ile çizgenin ağırlığı hesaplanır. Her ikili haberin birbirlerine olan benzerlikleri (çizge ağırlığı) hesapladıktan sonra kümeleme için gerekli olan benzerlik matrisi oluşturulur.

- ***Öbekleme:***

  Önceki adımda elde edilen benzerlik matrisi (similarity matrix), toplamalı hiyerarşik kümeleme (aggloramative hierarchical clustering) metoda girdi

(input) olarak verilir. Hiyerarşik öbekleme bu input matrisi kullanarak oluşturulur. Sonuç olarak hiyerarşik olarak kümelenmiş haberler elde edilir.

Aşağıda bu 3 adım daha detaylı bir şekilde anlatılmıştır.

3.1.1 **Varlık isimleri, türleri ve bağlantı ilişkilerinin çıkarılması**

Varlık isimlerinin çıkarılması ve Alchemy API

Haberde geçen varlık isimlerinin bulunması için, Alchemy API[25] kullanılmıştır. Bu araca gönderilen basit bir sorguya parametre olarak verilen haberin bağlantısı ile (URL) gönderilir, geriye haberin içindeki metinde bulunan varlık isimleri (named entity) ve bu varlık isimlerinin terim sıklığı (count), Linked data'da bulunan varlıkların yer aldığı veri kümeleri (dbpedia, freebase, yago), bu kümelerdeki URL bilgileri, bu verilerin türleri (type) ve alt türleri (subtype) bilgileri elde edilir.

Varlık ismi bulunması (Named Entity Recognition - N.E.R) işleminde, en önemli sorunlardan birisi gelen varlıklarda anlam karışıklığının yada birden fazla anlamın bulunmasıdır. Örneğin, metin içinde geçen "*apple*" kelimesinin meyve mi yoksa şirket adı olan "*Apple*" mı olduğunun saptanması başlı başına bir sorundur. Bununla ilgili çok fazla çalışma yapılmaktadır. Alchemy API gelen varlıklarda anlam karmaşasını çözüp (disambiguation) varlık isminin asıl bahsettiği varlığı <disambiguated> etiketi altında bulunan Açık Bağlı Veri kümelerdeki URL bilgilerini gösterir.

"`http://www.cnn.com/2009/CRIME/01/13/missing.pilot`" haber bağlantısı içinde geçen varlık isimlerinin alınması için örnek bir Alchemy API sorgu çağırısı aşağıda gösterilmektedir:

[25] http://www.alchemyapi.com/

```
http://access.alchemyapi.com/calls/url/URLGetRankedNamedEntities?
  apikey=YOUR_API_KEY&
  url=http://www.cnn.com/2009/CRIME/01/13/missing.pilot/index.html
```

Kullanıcı isteğe göre gelen varlık isimlerini ve bu varlık isimlerinin özelliklerini ister RDF/XML isterse XML veri gösterimi şeklinde alabiliyor.

```
<entity>
        <type>StateOrCountry</type>
        <relevance>0.264</relevance>
        <count>5</count>
        <text>Indiana</text>
        <disambiguated>
                <name>Indiana</name>
                <subType>Location</subType>
                <subType>PoliticalDistrict</subType>
                <subType>AdministrativeDivision</subType>
                <subType>GovermentalJurisdiction</subType>
                <subType>USState</subType>
                <geo>40,0 -86,0</geo>
                <website>http://in.gov</website>
                <dbpedia>http://dbpedia.org/resource/Indiana</dbpedia>
                <freebase>http://freebase.com/ns/guid.9202a8c0400064if80000</freebase>
                <umbel>http://umbel.org/umbel/ne/wikipedia/Indiana</umbel>
                <yago>http://mpii.de/yago/resource/Indiana</yago>
        <disambiguated>
<entity>
```

Şekil 7. Alchemy API örnek varlık ismi

$H = \{h_1,\ldots,h_n\}$ haber makalelerinin içeren haber listesi olsun. Her haber içindeki varlık isimleri listesini $V(h_i) = \{v_{i_l},\ldots,v_{i_m}\}$ küme gösterimi ile gösterebilir.

Varlık isimlerinin Yago tür bilgilerinin elde edilmesi

Anlamsal web'in (Semantic Web) en faydalı özelliklerinde birisi varlıkların tür bilgilerinin olmasıdır. Bu tür bilgisi sayesinde varlıkların hangi ana sınıfa ait olduklarını öğrenebiliriz. Çok basit bir örnek verecek olursak USA başkanı "Barack Obama"'nın türü insandır (person). Amerika'nın türü ise konumdur (location). Alchemy'den veya diğer Açık Bağlı Veri (LOD) veri kümelerindeki varlık isimlerinin tür bilgileri çok geneldir. Tür bilgilerinin çok genel olması istenilen bir durum değildir. Mesela DBpedia veri kümesindeki Barack Obama varlık isminin türü kişi (person) olarak belirtilmiştir. Bu çok genel bir bilgidir ve haber kümeleme

işleminde çok fazla gürültü (noise) üretebilecek bir şeydir. Yago'da, DBpedia'dan farklı olarak daha özel veri türleri tanımlıdır. Aşağıda "Barack Obama" varlık ismi için Yago'da bulunan türlerin bir kısmı listelenmiştir (Çizelge **2**).

Çizelge 2. Barack Obama varlık ismi için Yago tür listesi

1	21st-centuryPresidentsOfTheUnitedStates
2	AfricanAmericanAcademics
3	AfricanAmericanLawyers
4	AfricanAmericanPoliticians
5	AmericanLegalScholars
6	AmericanNobelLaureates
7	AmericanPoliticalWriters
8	AudioBookNarrators
9	CurrentNationalLeaders
10	HarvardLawSchoolAlumni
11	IllinoisLawyers
12	LivingPeople
13	NobelPeacePrizeLaureates
14	PeopleFromHonolulu,Hawaii
15	Person100007846
16	PresidentsOfTheUnitedNationsSecurityCouncil
17	PresidentsOfTheUnitedStates
18	WritersFromChicago,Illinois

Farklı veri kümelerinde aynı veriler yer alabilmektedir. Bunlar arasında bağlantılar da yerleştirilerek veya benzer bilgiler kopyalanarak daha zengin ve birbirleriyle bağlı veri kümeleri oluşturulmaya çalışılmaktadır. Örneğin DBpedia'da yer alan kaynaklar YAGO'da yer alıyorlarsa YAGO'daki tür bilgileri DBpedia'ya da aktarılmış bulunmaktadır. Yago varlık isimlerinin Dbpedia'daki karşılıklarının tür bilgileri DBpedia veri kümesi içerisinde de bulunur. DBpedia'da yer alan varlıkların türleri

SPARQL sorgusu ile alınabilir. Aşağıda "Barrack Obama"nın tür bilgilerinin DBpedia'dan sorgulanmasını sağlayan URL sorgusu gösterilmiştir:

```
http://dbpedia.org/sparql
  ?default-graph-uri=http://dbpedia.org
  &query=select+distinct?type+
  where{<http://live.dbpedia.org/resource/Barack_Obama a ?type}&format=XML
```

İki habere ait varlık isimlerinin arasındaki ilişkilerin belirlenmesi

Anlamsal Ağ'ın (Semantic Web) en önemli özelliklerinden birisi de varlıklar arasındaki ilişkilerin var olması. Hatta bu varlıkların aralarında bulunan ilişkilerin tanımları bile mevcuttur. Mesela DBpedia veri kümesinde Barack Obama'nın doğum yeri başka bir DBpedia varlık ismi olan Honolulu'dur. DBpedia, Wikipedia'nın RDF veri modeli haline dönüştürülmüş şeklidir. Yapılandırılmamış Wikipedia verisinin DBpedia RDF modeline çevrilme işi otomatik yapıldığı için varlıklar arasındaki bağlantıların var olduğu belirtilmiş, fakat tanımı yapılmamıştır. Böyle bağlantılar için DBpedia ontolojisinde http://dbpedia.org/ontology/wikiPageWikiLink ilişkisi ilave edilmiştir. Bu ilişki türü, ontolojide tanımlana bütün ilişkileri kapsar. Bunun için DBpedia veri kümesindeki iki tane varlık arasındaki ilişkinin var olup olmadığını aşağıdaki gibi bir SPARQL sorgusu ile öğrenilebilir:

```
ASK {
<http://dbpedia.org/page/Presidency_of_Barack_Obama>
<http://dbpedia.org/ontology/wikiPageWikiLink>
<http://dbpedia.org/page/Honolulu>
}
```

Yukarıdaki sorgu "Barack Obama" ile "Honolulu" arasında bir bağlantının var olup olmadığı bilgisini çıkarmayı sağlar. Bu iki varlık ismi arasında doğum yeri (http://dbpedia.org/ontology/birthPlace) ilişkisi bulunduğu için sonuç doğru (true) olarak geri dönecektir.

3.1.2 İki haber makalesi arasındaki anlamsal çizgenin oluşturulması

Haberlerin içerisindeki varlık isimleri, türleri ve bağlantı ilişkileri çıkarıldıktan sonra yapılması gereken verilen haber kümesi $H = \{h_1,...,h_n\}$ içindeki haberler arasında oluşturulacak çizgenin (graph) oluşturulması. Çizge gösterimi $G(H) = (V, E)$ şeklinde yapılabilir. H, haber makalesi kümesi, V, varlık isimleri, E ise bu varlık isimleri arasındaki ilişkiler olarak gösterilebilir.

Haberler arasında çizge oluşturulmasındaki amaç toplamalı hiyerarşik kümeleme yapabilmek için benzerlik ya da uzaklık matrisinin oluşturulması gerekliliğidir. İkili haber makaleleri arasında oluşan alt çizgelere (subgraph) bakarak matris oluşturulur. N tane haber için oluşturulacak alt çizge sayısı $\binom{N}{2}$ kadardır. Alt çizge gösterimi $G(h_i, h_j) = (V, E)$, şeklinde yapılabilir. V kümesi $V(h_i)$ ve $V(h_j)$'den oluşur, her biri içinde ilgili haberde geçen varlık isimlerinin kümesidir. E ise h_i ve h_j'de geçen varlık isimleri arasındaki ilişki kümesidir. $E = (E_v \cup E_t \cup E_b)$ olarak tanımlanabilir. E_v varlık eşitliği olan, E_t tür eşitliği olan, E_b ise varlıklar arası bağlantı ilişkilerini gösteren kenarlar kümesidir.

$w(e)$, varlık isimleri arasındaki bir kenarın ağırlığı olarak tanımlarsak, kenar ağırlıkları toplamlarından alt çizgenin ağırlığı hesaplanabilir. Bu ağırlık değeri sonraki bölümlerde benzerlik matrisinin değerlerini oluşturmak için kullanılacaktır.

Varlık isimleri arasındaki kenar (edge) bağlantılarının kurulması

İki haber arasındaki yakınlık, iki haber arasında varlık ilişkileri ile oluşturulan çizgenin ağırlığı ile belirlenecektir. Varlık isimleri arasında 3 farklı kenar ilişkisi vardır:

a) Varlık eşitliği bulunan varlık isimleri arasındaki kenarları:

Bu tür kenar ilişkileri en güçlü ilişkidir. Çünkü iki haber makalesinin birbirlerine yakın olması aynı varlık isimlerini içermesi ile yakından ilişkilidir. Mesela iki haberde de "Barack Obama" varlık ismi geçiyorsa bu iki haberin tam olarak olmasa da birbiriyle ilgili olabileceği konusunda bize bilgi vermektedir.

Çizge içerisindeki bu kenar bağlantılarını oluşturabilmek için h_i ve h_j haber makalelerinde bulunan tüm varlık isimlerinin Dbpedia URL bilgileri birbirleriyle karşılaştırılır. Eğer iki varlık ismi Dbpedia URL'si aynı ise bu iki düğüm (node) arasında bir kenar (edge) kurulur.

$v_k \in V(h_i)$, $v_l \in V(h_j)$ ve $v_k = v_l$ ise, bu iki varlık ismi arasındaki kenar $e_{v,}$ E kenar kümesine eklenir.

$$w(e) = c * tf(v_k) * tf(v_l) \tag{2.11}$$

(2.11)'deki formül kenar ağırlığı hesaplamanın genel formülüdür. 3 tür kenar ilişkisi için de aynı formül kullanılacaktır. Sadece c katsayısının değeri her tür kenar için farklı olacaktır. $tf(v_k)$, değeri h_i haber makalesinde geçen bir varlık isminin haber makalesi içerisindeki sıklığını (term frequency) belirtir. Kenar ağırlığı varlık isimlerinin sıklıkları ve c katsayısı ile çarpılarak bulunur. Varlık ismi eşitliği kenar türü için c katsayı değeri 1 olduğu için bu iki varlık ismi arasındaki kenarın ağırlığı bu varlık isminin geçtiği iki makaledeki sıklıkların çarpımı kadardır.

b) Tür eşitliği kenarları:

Bu tür kenar ilişkisi ikinci güçlü ilişkidir. İki farklı haberler içinde geçen varlık isimleri eğer aynı türe (type) bağımlıysa bu iki varlık ismi arasında bu tür bir kenar (edge) bağlantısı kurulur. Örnek verecek olursa bir makalede "Barack Obama" diğer

haber makalesinde ise "Vlademir Putin" geçiyorsa iki varlık ismi de Yago ontolojisinde tanımlı olan "CurrentNationalLeaders" tür bilgisine ait oldukları için bu iki varlık ismi arasında bir kenar ilişkisi kurulur.

$t_a \in T(v_k)$, $t_b \in T(v_l)$ ve $t_a = t_b$ ise; e_t, E kenar kümesine eklenir. t_a, v_k varlık ismine, t_b ise v_l varlık ismine ait türlerdir. T ise bir varlık ismine ait tüm türlerin bulunduğu kümedir. Bu kenarın ağırlığını hesaplamak için (2.11)'deki formül kullanılır. Tür ilişkisi ile kurulan kenar (edge) bağlantısı varlık ismi eşitliği kenar ilişkisinden daha az anlamlı olduğu için, yapılan deneylerden sonra buradaki *c* değeri $1/8$ olarak seçilmiştir. Bu kenarın ağırlığı $w(e_t)$ varlık isimlerinin her iki makaledeki sıklığının çarpımın 8'de 1'i kadardır. Diğer tür kenar ilişkilerinden farklı olarak iki varlık isminin birden fazla ortak türü olabileceği için, birden fazla kenar ilişkisi kurulabilir.

c) Varlık ilişkisi kenarları:

Bu tür kenar ilişkisi ağırlığı en az olan kenar ilişkisidir. İki farklı haber içinde geçen varlık ilişkileri arasında DBpedia veri kümesi ontolojisinde tanımlı olan "wikiPageWikiLinks" özelliği ile bağlı iseler bu iki varlık ismi arasında bu tür bir kenar (edge) kurulur. Örnek verecek olursak bir makalede "Barack Obama" diğer makalede "Honolulu" geçiyorsa bu iki varlık ismi arasında DBpedia veri kümesinde bir "wikipageWikiLinks" bağlantısı olduğu için bu iki varlık ismi arasında bu tür bir kenar ilişkisi kurulur.

$v_k \in V(h_i)$, $v_l \in V(h_j)$, eğer v_k ve v_l arasında bağlantı ilişkisi varsa, e_b, E kenar kümesine eklenir. Bu kenarın ağırlığını hesaplamak için (2.11)'deki formül kullanılır. Bağlantı ilişkisi en az önemli ilişki türü olduğu için, çeşitli deneylerden sonra *c* değeri 1/32 olarak seçilmiştir. Bu ilişki türünün katsayı *c* değerinin bu kadar az seçilmesinin nedeni, bu türden çok fazla ilişki olduğu için sonuçları olumsuz etkilediği görülmüş, bu yüzden ağırlık katsayısı düşük seçilmiştir.

h_i ve h_j haberlerinin oluşturduğu çizgenin ağırlığı tüm kenarların toplamı olarak ifade edilebilir (2.12):

$$w(G_{ij}) = \sum_{e \in E} w(e)\,,\ \forall\ e_t, e_v, e_b\ \in E \qquad (2.12)$$

3.1.3 **Benzerlik matrisinin oluşturulması**

Haber listesindeki tüm ikili haber makaleleri arasında bir çizge oluşturduktan sonra bu her makalenin diğer makalelere oluşturduğu ağırlığı $w(G_{ij})$ ($h_i\ ve\ h_j$ haberleri arsındaki çizge ağırlığı) ile benzerlik matrisi oluşturulur. Aşağıda örnek bir matris gösterimi verilmiştir. Oluşturulan bu benzerlik matrisi öbekleme işlemi sırasında toplamalı hiyerarşik öbekleme yöntemine girdi (input) olarak verilir.

	h1	**h2**	**h3**	**h4**	**h5**
h1	0.0				
h2	2.0	0.0			
h3	6.0	5.0	0.0		
h4	10.0	9.0	4.0	0.0	
h5	9.0	8.0	5.0	3.0	0.0

Şekil 8. Örnek benzerlik matrisi gösterimi[26]

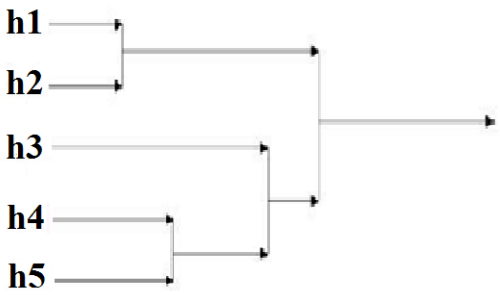
$h_i\ \in H$, $h_j\ \in H$ ve $i\ \neq j$ için $w(G_{ij})$, h_i ve h_j haber makaleleri arasındaki çizge ağırlığıdır.

.

[26] http://deepclimate.org/2011/06/07/mining-new-depths-in-scholarship-part-1/

3.1.4 **Öbekleme**

Öbekleme metodu olarak toplayıcı hiyerarşik öbekleme kullanılmıştır. Bu yöntem elemanlar arasında oluşturulan uzaklık ya da benzerlik matrisini girdi olarak alır ve seçilen bağlama yöntemine (single, complete, average) göre oluşan öbekleri toplayarak en sonunda öbekler bir hiyerarşi şeklini alır ve gösterim dendogram üzerinde yapılır [82]. Şekil 8'deki benzerlik matrisi için elde edilen dendogram Şekil 9'da gösterilmiştir.



Şekil 9. Şekil 7 için dendogram gösterimi

Hiyerarşik öbekleme yaptıktan sonra sistemin başarısını ölçmek için hiyerarşik kümelemeyi parçalı kümelemeye çevirmek gerekir (partial clustering). Bunu yapmak için hiyerarşi uygun bir seviyeden (level) kesilmesi lazım. Örnek verecek olursak Şekil 9'deki hiyerarşi 1. Seviyeden kesilirse oluşacak öbekler (h1, h2) ve (h3, h4, h5) olur. Eğer 2. Seviyeden kesilecek olursa oluşacak öbekler (h1, h2), (h3), (h4,h5) olur. 3.seviyeden kesilecek olursa (h1, h2), (h3), (h4), (h5) şeklinde oluşur. Kesilme düzeyinin belirlemek için çeşitli çalışmalar yapılmıştır. Bu tez çalışmasında uygun kesme seviyesini bulmak için "silhouette katsayısı" kullanılmıştır ve ölçüm yöntemi aşağıda gösterilmiştir [39].

$$s(i) = \begin{cases} 1 - a(i)/b(i), eğer\ a(i) < b(i) \\ 0, \qquad\qquad eğer\ a(i) = b(i) \\ b(i)/a(i) - 1, eğer\ a(i) > b(i) \end{cases} \tag{2.13}$$

$$-1 \leq s(i) \leq 1$$

$a(i)$, i elemanının bulunduğu öbek içindeki diğer elemanlarla olan ortalama uzaklığıdır. $b(i)$ ise i elemanının, diğer öbeklerle olan ortalama uzaklıkların en düşüğüdür. Hiyerarşik kümeleme döngüsel olarak her seviyeden kesilip, her seviyedeki elemanlarının (haber makaleleri) ortalama $s(i)$ değeri ölçülmüştür. En yüksek ortalama $s(i)$ değeri bulunan seviyeden hiyerarşi kesilip parçalı kümeler haline getirilmiştir.

3.2 Örnek Senaryo

Bu senaryoda 2 tane haber makalesinin içinde geçen varlık isimleri, bu varlık isimlerine ait tür (type) bilgileri ve varlık isimleri arasındaki bağlantılar çıkartıldıktan sonra, bu iki haber makalesi arasında anlamsal çizgenin (semantic graph) oluşturulması ve bu çizgenin ağırlık hesaplanması gösterilecektir. Bunun için iki tane haber makalesi seçilmiştir. Bu iki haber makalesi de Japonya ve Çin arasındaki diplomasiden bahsediyor. Birbirlerine yakın olayları anlatan haber makaleleri seçilmiştir.

1) http://www.nytimes.com/2012/11/27/world/asia/japan-expands-its-regional-military-role.html?ref=territorialdisputes&_r=1&

Tokyo— After years of watching its international influence eroded by a slow-motion economic decline, the pacifist nation of ***Japan*** is trying to raise its profile in a new way, offering military aid for the first time in decades and displaying its own armed forces in an effort to build regional alliances and shore up other countries' defenses to counter a rising ***China***. Already this year, Japan crossed a little-noted threshold by providing its first military aid abroad since the end of World War, approving a $2 million package for its military engineers to train troops in ***Cambodia*** and ***East Timor*** in disaster relief and skills like road building. Japanese warships have not only conducted joint exercises with a growing number of military forces in the Pacific and ***Asia***, but they have also begun making regular port visits to countries long fearful of a resurgence of Japan's military.And after stepping up civilian aid programs to train and equip the coast guards of other nations, Japanese defense officials and analysts say, Japan could soon reach another milestone: beginning sales in the region of military hardware like seaplanes, and perhaps eventually the stealthy diesel-powered submarines considered well suited to the shallow waters where China is making increasingly assertive territorial claims…

2) http://www.nytimes.com/2012/10/31/world/asia/in-speech-organized-by-beijing-ex-diplomat-calls-islands-dispute-with-japan-a-time-bomb.html?_r=1&

HONG KONG — A longtime Chinese diplomat warned Tuesday that the United States is using Japan as a strategic tool in its effort to mount a comeback in ***Asia***, a policy that he said is serving to heighten tensions between China and Japan. The retired diplomat, Chen Jian, who served as an under secretary general of the United Nations and as China's ambassador to Japan, said the United States should restrain Tokyo and should focus its diplomatic efforts on bringing about negotiations between China and ***Japan*** over the disputed islands in the ***East China Sea*** known as the Diaoyu by China and the Senkaku by Japan.In an unusually biting assessment of the ***United States***, Mr. Chen said: "It is in the U.S. interest to quarrel with China, but not to fight with China."While Mr. Chen has retired from China's diplomatic service, his remarks were particularly significant because they represent the most detailed public exposition of China's views at a time when Chinese officials have been wary of making comments because of the approaching Communist Party Congress, which is scheduled to begin in ***Beijing*** on Nov. 8.In the speech, which was organized by the Chinese Ministry of Foreign Affairs and was attended by half a dozen Chinese diplomats, ***Keiro Kitagami*** held out an olive branch by urging that discussions between Japan and China should start on ways to reduce the risk of clashes between Chinese… Japanese patrol vessels that have gotten perilously close off the islands in the last month...

Yapılacak ilk işlem, bu iki haber makalelerindeki varlık isimlerini, bu varlık isimlerinin haberler içindeki sıklıklarını (term frequency) ve Linked data'daki DBpedia bağlantı (URL) bilgilerini çıkarmaktır.

Çizelge 3. 1. Haber makalesindeki varlık isimleri

Varlık İsmi	**Sıklığı**	**Dbpedia URL**
Japan	18	http://dbpedia.org/resource/Japan
China	11	http://dbpedia.org/resource/People's_Republic_of_China
United States	5	http://dbpedia.org/resource/United_States
Keiro Kitagami	2	http://dbpedia.org/resource/Keiro_Kitagami
Asia	3	http://dbpedia.org/resource/Asia
TOKYO	2	http://dbpedia.org/resource/Tokyo
East China Sea	1	http://dbpedia.org/resource/East_China_Sea
South China Sea	1	http://dbpedia.org/page/South_China_Sea
Akihisa Nagashima	1	http://dbpedia.org/resource/Akihisa_Nagashima
East Timor	1	http://dbpedia.org/resource/East_Timor
Cambodia	1	http://dbpedia.org/resource/Cambodia
Southeast Asian	1	http://dbpedia.org/resource/Southeast_Asia
Iraq	1	http://dbpedia.org/resource/Iraq
Yoshihiko Noda	1	http://dbpedia.org/resource/Yoshihiko_Noda
Keio University	1	http://dbpedia.org/resource/Keio_University
Afghanistan	1	http://dbpedia.org/resource/Afghanistan

Çizelge 4. 2. Haber makalesindeki varlık isimleri

Varlık İsmi	**Sıklığı**	**Dbpedia URL**
China	29	http://dbpedia.org/resource/People's_Republic_of_China
Japan	23	http://dbpedia.org/resource/Japan
United States	15	http://dbpedia.org/resource/United_States
Keiro Kitagami	2	http://dbpedia.org/resource/Keiro_Kitagami
Asia	2	http://dbpedia.org/resource/Asia

TOKYO	2	http://dbpedia.org/resource/Tokyo
Beijing	2	http://dbpedia.org/resource/Beijing
Congress	2	http://dbpedia.org/resource/United_States_Congress
Hong Kong	1	http://dbpedia.org/resource/Hong_Kong
Wen Jiabao	1	http://dbpedia.org/resource/Wen_Jiabao
Cambodia	1	http://dbpedia.org/resource/Cambodia
Southeast Asian	1	http://dbpedia.org/resource/Southeast_Asia
United Nations	1	http://dbpedia.org/resource/United_Nations
East China Sea	1	http://dbpedia.org/resource/East_China_Sea
Keio University	1	http://dbpedia.org/resource/Keio_University
South China Sea	1	http://dbpedia.org/page/South_China_Sea
New York	1	http://dbpedia.org/resource/New_York_City
Philippines	1	http://dbpedia.org/resource/Philippines
Renmin University	1	http://dbpedia.org/resource/Renmin_University_of_China

İkinci haberde geçen varlık isimleri, sıklıkları ve DBpedia URL bilgileri haberlerde geçen varlık isimleri Alchemy API kullanılarak çıkarıldıktan sonra sıradaki işlem elde edilen her bir varlık ismi için Yago türlerini tespit etmektir. Yago türleri DBpedia veri seti içinde de bulunduğu için DBpedia veri seti sorgulanıp gelen türleri filtreleyip sadece Yago türleri elde edilir. Bir varlık ismi için onlarca yago türü olduğu için gösterim açısından aşağıda sadece bir kaç tane varlık isminin yago türleri verilmiştir.

Çizelge 5. 1. Haber makalesindeki bazı varlık isimleri için Yago tür bilgileri

Varlık İsmi	**Yago Tür URL**
Japan	http://dbpedia.org/class/yago/OECDMemberEconomies http://dbpedia.org/class/yago/CountriesBorderingThePhilippineSea http://dbpedia.org/class/yago/CountriesBorderingThePacificOcean http://dbpedia.org/class/yago/EastAsianCountries http://dbpedia.org/class/yago/IslandCountries

	http://dbpedia.org/class/yago/G20Nations
Beijing	http://dbpedia.org/class/yago/CapitalsInAsia http://dbpedia.org/class/yago/MetropolitanAreasOfChina http://dbpedia.org/class/yago/MunicipalitiesOfThePeople'sRepuBlicOfChina http://dbpedia.org/class/yago/HostCitiesOfTheSummerOlympicGames http://dbpedia.org/class/yago/PrefecturesOfJapan
China	http://dbpedia.org/class/yago/G20Nations http://dbpedia.org/class/yago/EastAsianCountries
Keiro Kitagami	http://dbpedia.org/class/yago/KeioUniversityAlumni http://dbpedia.org/class/yago/PeopleFromTokyo

Çizelge 6. 2. Haber makalesindeki bazı varlık isimleri için Yago tür bilgileri

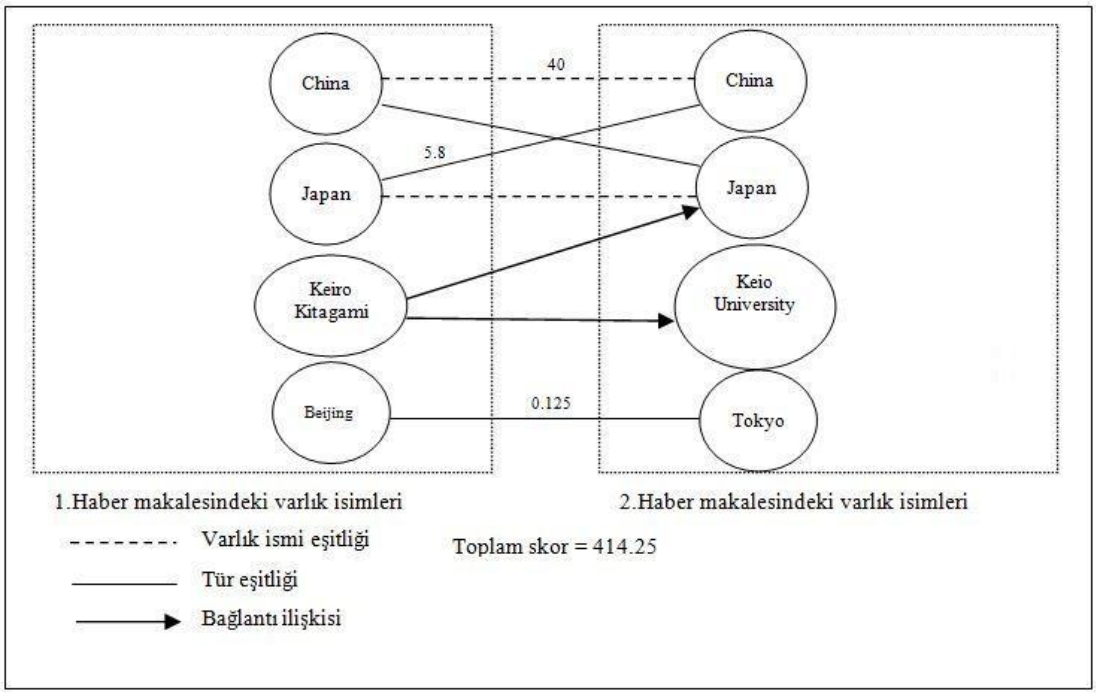
Varlık İsmi	**Yago Tür URL**
Japan	http://dbpedia.org/class/yago/OECDMemberEconomies http://dbpedia.org/class/yago/CountriesBorderingThePhilippineSea http://dbpedia.org/class/yago/CountriesBorderingThePacificOcean http://dbpedia.org/class/yago/EastAsianCountries http://dbpedia.org/class/yago/IslandCountries http://dbpedia.org/class/yago/G20Nations
Tokyo	http://dbpedia.org/class/yago/CapitalsInAsia http://dbpedia.org/class/yago/PopulatedCoastalPlacesInJapan http://dbpedia.org/class/yago/HostCitiesOfTheSummerOlympicGames http://dbpedia.org/class/yago/PrefecturesOfJapan
China	http://dbpedia.org/class/yago/G20Nations http://dbpedia.org/class/yago/CountriesBorderingThePacificOcean http://dbpedia.org/class/yago/EastAsianCountries
Keio University	http://dbpedia.org/class/yago/EduCAtionalInstitutionsEstablishedIn1858 http://dbpedia.org/class/yago/PrivateUniversitiesInJapan

Son olarak, ilk haber makalesindeki varlık isimlerinin diğer haber makalesindeki varlık isimleri ile olan DBpedia veri kümesinde var olan bağlantı ilişkilerini çıkarılır. Fakat bu tür ilişkinin diğer ilişkilerden farkı ilişkinin iki taraflı da olabileceğidir. Bu sebeple ikinci haber makalesindeki varlık isimlerinin ilk haber makalesindeki varlık isimleri ile olan bağlantı ilişkilerinin de çıkarılması gerekir.

Çizelge 7. DBpedia Veri kümesinde aralarında WikiPageWikiLinks ilişkisi bulunan bazı varlık isimleri

Varlık ismi	**Varlık ismi**
Keiro Kitagami	Japan
Japan	Tokyo
Tokyo	Japan
Japan	Keio University
Keio University	Japan
Beijing	Tokyo
Keiro Kitagami	Keio University

Varlık isimler, varlık isimlerinin türleri ve diğer varlık isimleriyle aralarında var olan bağlantı ilişkileri elde edildikten sonra iki haber arasında çizge oluşturma işlemi yapılabilir. Daha önceden belirtildiği gibi iki varlık ismi arasında üç kenar (edge) türü bulunabilir. İki farklı varlık ismi arasında bu üç çeşit kenardan yalnızca bir tanesi ile kenar ilişkisi kurulabilir.

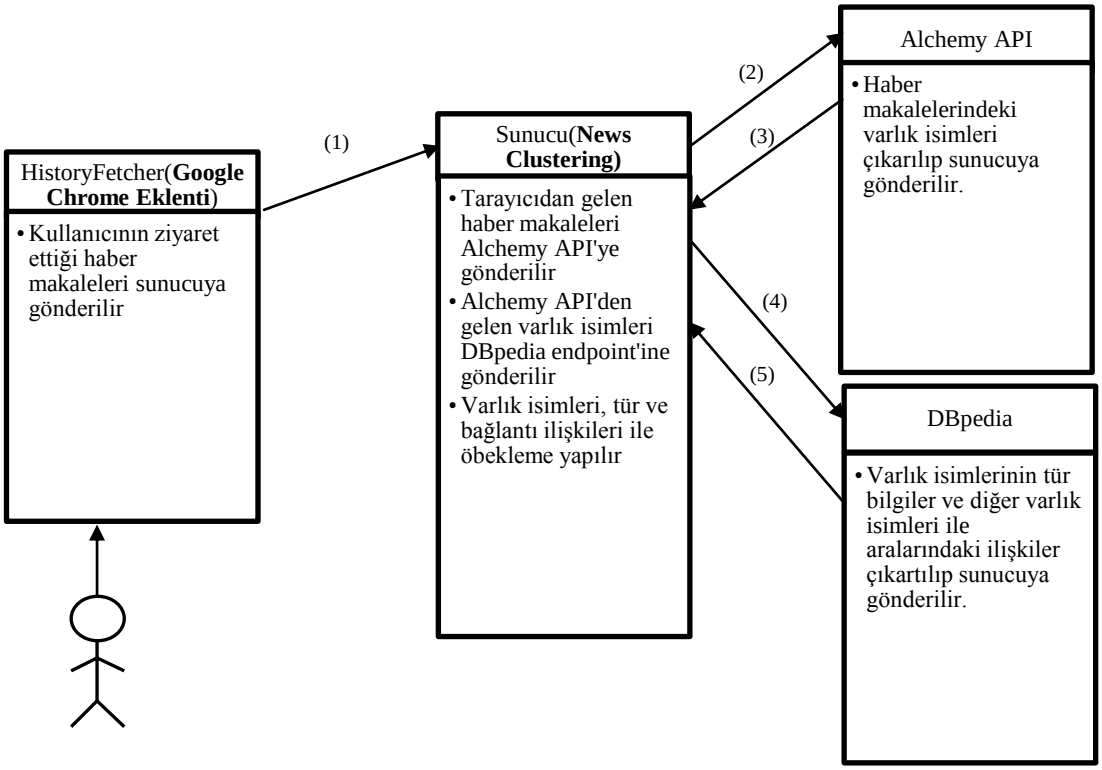


Şekil 10. İki haber makalesi içindeki bazı varlık isimlerinin aralarında oluşan çizge(varlık isimleri ve ilişkilerinin tümü gösterilmemiştir.)

İlk makalede geçen China varlık ismi ile ikinci makalede geçen China varlık ismi arasında varlık ismi eşitliği ilişkisi vardır. Bu kenarın ağırlığı: 1 (11 + 29) = 40 (2.11). Japan varlık ismi ile ikinci makalede geçen China varlık ismi arasında Yago türlerinden "G20Nations" tür ilişkisi vardır. Bu kenarın ağırlığı = $\frac{1}{8}$ (18 + 29) = 5,8 (2.11). Beijing varlık ismi ile ikinci makalede geçen Tokyo varlık ismi arasında DBpedia veri kümesi ontolojisinde tanımlı bağlantı ilişkisi vardır. Kenar ağırlığı = $\frac{1}{32}$ (2 + 2) = 0.125 (2.11). İki haber makalesi arasındaki oluşan çizgenin toplam ağırlığı bütün kenarların toplanmasıyla 414.25 olur.

4. GERÇEKLEŞTİRİM

Bu bölümde, mimari hakkında genel bilgi verilecektir ve gerçekleştirimde bulunulan 2 bileşenden bahsedilecektir. Bu bileşenlerden ilki istemci tarafında çalışan "HistoryFetcher" adlı tarayıcı tabanlı çalışan bir eklentidir. İkinci bileşen ise sunucu tarafında çalışan "NewsClustering" adlı bileşendir. Bu mimaride, bileşenlerin yapacağı işleri bölerek; istemci tarafından kullanıcının sadece gezdiği haber sayfaları gönderilecek, sunucu tarafında ise gelen verilerin filtreleme ve sonuç üretme işlemleri yapılacaktır. Ayrıca bu tarz yaklaşım bizim tarayıcıya olan bağımlılığımızı da ortadan kaldırmaktadır.



Şekil 11. Sistem şeması

Sistemin çalışması genel hatlarıyla aşağıdaki gibidir:

(1) Kullanıcı geliştirdiğimiz bir Google Chrome[27] tarayıcı eklentisi olan "HistoryFetcher" sayesinde gezindiği haber makalelerini sunucu tarafına gönderir.
(2) Sunucuya gelen haberlerin içindeki varlık isimlerinin elde edebilmesi için haber URL bağlantıları Alchemy'ye gönderilir.
(3) Alchemy gelen haber makalelerinin HTML yapısını parçalayıp varlık isimlerini DBpedia URL bilgileri ve haberin tüm metnini sunucuya geri gönderir.
(4) Sunucuya gelen varlık isimlerinin tür ve diğer varlık isimleri ile olan bağlantı ilişkilerini DBpedia kullanılarak sorgulanır.
(5) DBpedia varlık isimlerinin tür ve bağlantı ilişkileri çıkarılarak sunucuya geri gönderilir.

Aşağıda bu bileşenler daha ayrıntılı bir biçimde anlatılmaktadır.

4.1 HistoryFetcher

"HistoryFetcher" kullanıcın Google Chrome tarayıcısına yüklenebilen bir eklentidir. Google Chrome Google tarafında geliştirilen bir ağ tarayıcısıdır. Chrome %35 lik pazar payı ile en çok kullanılan web tarayıcısıdır. Chrome, eklenti geliştirmedeki kolaylık ve kullanım yaygınlığından dolayı seçilmiştir. Chrome eklentileri, JavaScript, HTML ve CSS kullanılarak geliştirilir. Bir Google Chrome eklentisi temel olarak 3 bileşenden oluşur.

a) Manifesto Dosyası (Manifest file)

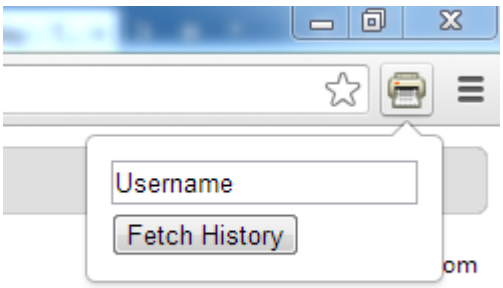
Bu dosyada gerçekleştirilecek eklenti ile ilgili genel bilgiler, eklentinin sürümü ve eklentinin kullanacağı izinler ile ilgili bilgiler verilmektedir. Bizim eklentimiz

[27] http://en.wikipedia.org/wiki/Google_Chrome

kullanıcının geçmişte gezindiği sayfaları göndereceği için, “history” iznini istemektedir.

b) Arayüz sayfaları (User interface pages)

Eklentinin ara yüz sayfalarıdır. HTML ve CSS kullanılarak oluşturulur. Tarayıcının sağ üst köşesindeki eklentinin simgesine tıklanarak bu erişilebilinir.



Şekil 12. HistoryFetcher Arayüzü

Kullanıcıya, eklenti simgesine tıkladığı zaman bir açılır pencere (popup window) gösterilir. Bu açılır pencere ile kullanıcı eklentiyi kullanabilir. Bizim eklentimizde açılır pencere, kullanıcının kullanıcı adını girebileceği bir yazı alanı (text field) ve tıklayarak gezindiği sayfaları gönderebileceği bir düğmeden (button) oluşmaktadır.

c) Arka plan Sayfaları (Background pages)

Arka plan sayfası her eklentide bir tane bulunan ve eklentinin ana mantığını gerçekleştiren görünmez sayfalardır. Bu dosya manifest dosyasında belirtilmelidir. Arka plan sayfaları diğer sayfalar ile içerik betikleri (content scripts) kullanarak mesajlaşır. Açılır penceredeki düğmeye tıklanıldığı zaman kullanıcının geçmiş bilgilerine gidip, bu bilgileri sunucu tarafına gönderme işlemi bu kısımda yapılır.

Kullanıcının geçmiş bilgilerine ulaşmak için yine Google Chrome API[28] kütüphanesindeki tanımlanmış metodlar kullanılır. Aşağıdaki metod API kütüphanesinde kullanılan kullanıcının gezindiği sayfa bilgilerine erişmeyi sağlar:

```
chrome.history.search({text: "", maxResults: 1000, startTime: 0},
onHistoryResult);
```

"maxResult" parametresi tarihsel sıra ile kaç tane sayfa bilgisinin, "startTime" parametresi ise hangi tarihten başlanılarak zamansal olarak geriye doğru gezindiği sayfa linklerine erişilmesini sağlar. En sondaki parametre ise geri çağırım (callback) metodunu belirtir, gelen linklerin kullanılacağı metot tanımıdır. Kullanıcını gezindiği sayfa listesini göndermek için basit bir XMLHttpRequest nesnesi oluşturulur ve "GET" metodu ile gerekli parametreler (geçmiş bağlantısı, kullanıcı adı) de gönderilecek bağlantıya ilave edilerek sunucu tarafında çalışan bileşene gönderilir.

4.2 News Clustering

Bu bileşen tamamen sunucu tarafında çalışan, Java programla dili kullanarak geliştirilen bir uygulamadır. Bu bileşen görev alanı itibarıyla birkaç alt iş bölümüne ayrılmıştır. Bu alt bileşenler

a) Veri toplama

Bu bileşen gelen verileri almak için 8080 kapısını dinleyen bir servlet sınıfını içerir. Servlet gelen veriyi işleyip alt bileşenleri sırasıyla başlatılması işlevini yürüten ana sınıftır. Bu sınıf ilk olarak HttpServletRequest nesnesi tarafından üretilen bir HttpSession nesnesi oluşturur. Bu HttpSession nesnesi ilk önce gelen bağlantı(URL) içindekileri parametreleri ayrıştırır. Ayrıştırılan bu parametrelerle kullanıcını kullanıcı adı ve gezindiği sayfa bilgileri alınır ve sonra bu geçmiş sayfa bilgilerine

[28] http://developer.chrome.com/extensions/api_index.html

filtreleme işlemi uygulanır. Daha sonra sistem daha önce tanımlanan sayıda kullanıcının gezindiği sayfa URL'sini elde ettiği zaman işlemini bitirip, gerekli alt bileşenleri çağırır. Bu kapsamda ilk olarak elde edilen bağlantılar (URL) Alchemy API kullanılarak, ilk önce bu sayfada geçen varlık isimler (named entity) çıkarılır, daha sonra DBpedia sorgulanarak, bu varlık isimlerin tür (type) ve bu varlık isimleri arasındaki bağlantı ilişkileri elde edilir. Kullanıcı adı, sayfalar ve bu sayfalardaki varlık isimlerinin tür ve diğer varlık isimleri arasındaki bağlantı bilgileri elde edildikten sonra bu bilgiler sonraki işlemlerde kullanılmak üzere veri tabanına kayıt edilir. Daha sonra bu veriler kullanılarak öbekleme işlemini yapacak olan alt bileşen çağırılır.

b) Alchemy API

Bu bileşen Alchemy API'yi kullanmak için gerekli olan metotları içeren kütüphane sınıfını ve bu sınıfı kullanarak istenilen verileri elde eden operasyon sınıfını içerir. Alchemy API'yi kullanabilmek için kullanıcılar Alchemy'ye üye olup birer API anahtarı (API key) aldıktan sonra üyelik kapasitesine göre sorgulama yapabilir. Alchemy, araştırma çalışmaları için günlük 30000 sorgulamaya izin vermektedir. Kullanıcıların gezindiği sayfaları Alchemy'ye gönderilerek o sayfada bulunan varlık isimlerinin yazı, bağlı açık veri (linked data) bağlantısı, sıklığı (term frequency) verileri elde edilir.

c) Dbpedia

Bu bileşen Alchemy Api'den dönen varlık isimlerinin tür ve diğer varlık isimleriyle ile ilgili bağlantı ilişkilerinin elde edildiği kısımdır. Varlık isimleriyle ilgili bu bilgiler DBpedia'nın web apisi kullanılarak elde edilir. İki varlık ismi arasındaki bağlantı ilişkileri yine Dbpedia'nın web API'si kullanılarak elde edilir. Elde edilen tür ve ilişki bilgileri veri tabanına kayıt edilir.

5. DEĞERLENDİRME

Bu bölümde geliştirdiğimiz yöntem ile yapılan çalışmaların sonuçları değerlendirilecektir. İlk önce kullandığımız veri kümelerinin nasıl oluşturulduğu ve yapısından, sonrasında ise performans analiz sonuçları ve bu sonuçların değerlendirilmesinden bahsedilecektir.

5.1 Veri Kümesi

Veri kümesi oluşturmak için 2 farklı yöntem seçilmiştir. İlk veri kümesi Google news sitesinden toplanan 150 tane haber makalesi seçilerek oluşturulmuştur. Bu haberler konularına ve benzerliklerine göre bir kullanıcı tarafından elle öbeklenmiş ve 14 gruba ayrılmıştır. İkinci veri kümesi ise gerçek kullanıcılar tarafından oluşturulmuştur. Bunun için 3 farklı kullanıcı seçilmiştir. Her kullanıcı "HistoryFetcher" tarayıcı eklentisi kullanarak gezindiği son 60 haber sayfasını sunucuya göndermiştir. Başarı ölçümü yapabilmek için her kullanıcı gezindiği bu 60 haber sayfanın öbeklemesini elle yapmıştır. Her haber makalesinin ilgili haber konularını yine okuyucu bilebileceği için her kullanıcı sadece kendisinin okuduğu haberleri kümelemiştir. Bu sebeple performans analizi kullanıcı odaklı olarak gösterilecektir. Sistem kendi başarısının "bag of words" (kelime çantası) yöntemi ile de karşılaştıracağı için haber makalelerinin metin hallerine de ihtiyaç duyulmaktadır. Sistem, kullanıcıdan "HistoryFetcher" tarayıcı eklentisi aracılığıyla gelen haber bağlantılarını (URL) Alchemy API kullanarak haberlerin metinlerini otomatik elde etmektedir.

5.2 Performans Analizi

Sistemin performansını ölçmek için hassasiyet (precision), geri çağırım (recall), F1 skoru (F1 score) ölçüm değerleri kullanılmıştır.

Tüm haberlerin listesini $H = \{h_1,...,h_n\}$, Kullanıcının tanımladığı öbek listesi $X = \{x_1,...,x_r\}$ (r kümeye ayrılmış), ve sistemimizin ürettiği çıktı öbek listesi $Y = \{y_1,...,y_s\}$ (s kümeye ayrılmış) olsun. Sistemin öbekleme başarısı Tablo 8'de gösterilen ölçüm parametreleri ile değerlendirilecektir.

Çizelge 8. Başarı ölçüm parametreleri

	Sistem aynı küme	Sistem farklı küme
Kullanıcı aynı küme	TP	FN
Kullanıcı farklı küme	FP	TN

Tabloda yer alan parametrelerin anlamları şunlardır:

- *TP* (True positive), H haber listesindeki haber ikililerinden X listesinde ve Y listesinde aynı öbekte olan haberlerin sayısı,
- *TN* (True negative), H haber listesindeki haber ikililerinden X listesinde farklı Y listesinde farklı öbekte olan haberlerin sayısı,
- *FN* (False negative), H haber listesindeki haber ikililerinde X listesinde aynı Y listesinde farklı öbekte olan haberlerin sayısı,
- *FP* (False positive) H haber listesindeki haber ikililerinden X listesinde farklı Y listesinde aynı öbekte olan haberlerin sayısını gösterir.

Hassasiyet, geri çağırım, F1 skoru metrikleri aşağıdaki gibi hesaplanır [40]:

$$Hassasiyet\ (precision) = \frac{TP}{FP+TP} \tag{2.14}$$

$$Geri\ çağırım\ (recall) = \frac{TP}{TP+FN} \tag{2.15}$$

$$F1\ skoru = 2.\frac{precision.recall}{precision+recall} \quad (2.16)$$

İki haber makalesi arasındaki çizgeyi 3 farklı kenar ilişkisi oluşturur. Çizgeyi bu 3 tane kenar çeşidinin farklı kombinasyonları ile oluşturarak öbekleme yapılmıştır. Buradaki amaç bu kenar türlerinin öbekleme performansına olan katkılarını ölçebilmektir. Bu yöntemler:

- *Varlık ismi*: Sadece haber kümesinde geçen varlık isimlerinin eşitliği ile oluşan çizgelerin ile yapılan öbekleme yöntemidir
- *Varlık ismi + tür*: Haber kümesinde geçen varlık isimlerinin eşitliği ve varlık isimlerinin tür eşitliği ile oluşan çizgeler ile yapılan öbekleme yöntemidir.
- *Varlık ismi + tür + bağlantı ilişkisi*: 3 kenar türünün de bulunmasıyla oluşan çizgeler ile yapılan öbekleme yöntemidir.

Çizelge 9, 10, 11, 12 deki 5. 6. 7. kolonlar ise öbekleme yaparken çizgeleri oluşturan kenar türlerinin tüm çizgelerin toplam puanı içindeki katkısını göstermektedir. Buradaki amaç ise denenen farklı kombinasyonlarda kenarların birbirlerin göre puanlamaya katkı oranını göstermektir.

Çizelge 9. Google News haber makaleleri için öbekleme performansı

Yöntem	**Precision**	**Recall**	**F1 skoru**	**Varlık ismi %**	**Tür %**	**Bağlantı ilişkisi %**
Bag of words	0.42	0.30	0.35	-	-	-
Varlık ismi	**0.90**	0.57	0.70	100	0	0
Varlık ismi + tür	0.84	**0.61**	**0.75**	31	69	0
Varlık ismi + tür +bağlantı ilişkisi	0.71	0.48	0.57	13	32	55

Google News haber kümesi için yapılan deneylerde en iyi sonucu veren yöntem Varlık ismi + tür yöntemi olmuştur (F1 = 0.75). En kötü sonuç veren ise “bag of

words" yöntemidir. "Bag of words" yönteminin kötü sonuç vermesinin nedeni olarak çok fazla gürültü (noise) içermesi söylenebilir. Çünkü "bag of words" yöntemi ortak kelimeler üzerinden öbekleme yaptığı için haber makalesindeki varlık isimleri haricindeki kelimelerin sıklığı ve aynı zamanda bu kelimelerin diğer haber makalelerinde de bulunması nedeniyle ilgisiz haberleri aynı öbek içerisine koyabilmektedir. Varlık ismi + tür + bağlantı yönteminin de iyi sonuç vermediği görülmüştür. Bunun nedeni ise bağlantı ilişkisinin toplam ağırlıktaki oranının (%55) fazla olması ve gürültü üretebilecek çok sayıda bağlantının bulunmasıdır.

Çizelge 9'da görüldüğü gibi varlık isimleri ile tür ilişkisinin birlikte kullanılması performansı arttırmaktadır. Bunun nedeni varlık ismi yöntemi sadece varlık isimlerini kullanarak öbekleme yapmaktadır. Mesela sadece "Volkswagen Group" marka otomobillerden bahseden haberler ile sadece "Mercedes Benz" marka otomobillerden bahseden haberler ortak varlık ismi olmadığından dolayı aynı öbek içinde bulunmazlar. Fakat Varlık ismi + tür yönteminde ise iki varlık ismi de aynı Yago türlerine sahip olduğu için (MotorVehicleManufacturersOfGermany, CarManufacturers vb.) bunlar aynı öbek altında bulunabilmektedir. Çizelge 9'da genellikle geri çağırım (recall) değerleri yüksek değildir. Geri çağırım değerinin düşük çıkması kullanıcının tanımladığı öbek listesinde aynı öbek içinde bulunan haber ikilileri sistemin ürettiği öbeklerde aynı öbekte bulunmamasından kaynaklanmıştır. Bunun başlıca iki nedeni vardır. Kullanıcı haber makalelerini çok genel öbekler içine yerleştirmiştir ve sistemin ürettiği öbekler birbirleriyle daha fazla ilişkili haberleri olduğu için kullanıcı tarafında aynı öbekte bulunan haberler sistemin ürettiği öbek listesinde aynı öbek altında bulunamamıştır. Diğer bir neden ise hiyerarşik öbeklemenin kesildiği seviye ile ilgilidir. Hiyerarşik öbeklemenin sonucu bir hiyerarşi şeklinde üretildiği için hiyerarşide birbirlerine yakın olan haberler bile hiyerarşi belli bir seviyeden kesildikten sonra farklı öbekler içinde olabilmektedir.

Çizelge 10. Google News veri kümesi için silhoutte katsayısı performansa etkisi

	Silhoutte katsayı	**F1 skoru**	**Cluster Sayısı**
Seviye 1	-	-	1
Seviye 2	0.18	0.63	10
Seviye 3	**0.26**	**0.75**	20
Seviye 4	0.10	0.56	31

Yukarıdaki Çizelgede Google News veri kümesi için "silhoutte katsayısı" nın Varlık ismi + tür yöntemi için başarı ölçümleri verilmiştir. Sistemin başarısını ölçebilmek için hiyerarşik öbeklemeyi, parçalı öbeklemeye çevirmek gerekmektedir. Daha önceki bölümlerde bahsedildiği gibi "silhoutte katsayısı" hiyerarşinin kesilme seviyesini bulmak için kullanılmıştır. Çizelge 10'da görüldüğü gibi "silhoutte katsayısı" nın en yüksek seviyesi (0.26) aynı zamanda F1 skorunun da en fazla olduğu seviyedir.

Aşağıda ise 3 kullanıcı için elde edilen sonuçlar vardır. Bu kullanıcılar için (2.11)'de kenar ağırlıklarını bulmak için kullandığımız formül değiştirilip varlık isimlerinin sıklıklarını çarpmak yerine toplama işlemi yapılarak en başarılı formül gösterimi de tespit edilmeye çalışılmıştır.

$$w(e) = c * (\, tf(v_k) + tf(v_l)) \tag{2.17}$$

Performans analizi kullanıcı temelli olduğu için 3 kullanıcı için sonuçlar ayrı Çizelgelerde gösterilmiştir.

Çizelge 11. 1. kullanıcı için öbekleme performansı

Yöntem	**Precision**	**Recall**	**F1 skoru**	**Varlık ismi %**	**Tür %**	**Bağlantı ilişkisi %**
Bag of words	0.44	0.32	0.37	-	-	-
Varlık ismi	0.87	0.41	0.56	100	-	-
Varlık ismi + tür	**0.93**	**0.70**	**0.81**	23	87	-
Varlık ismi + tür +bağlantı ilişkisi	0.59	0.69	0.63	11	33	56

Yukarıdaki Çizelge ilk kullanıcıdan elde edilen veri kümesi ile yapılan deneysel çalışmaların sonuçlarını göstermektedir. Çizelgeye bakıldığı zaman en iyi sonuç Varlık ismi + tür yöntemi ile elde edilmiştir (F1 = 0.81). Bu Çizelgede Varlık ismi + tür + bağlantı yöntemi Varlık ismi yönteminden daha iyi sonuç vermiştir. Fakat Google news veri ile yaptığımız çalışmalarda tam tersi durum söz konusudur. Buradan başarı performanslarının veri kümesi bağımlı olduğu sonucuna ulaşılmaktadır.

Çizelge 12. 2. kullanıcı için öbekleme performansı

Yöntem	**Precision**	**Recall**	**F1 skoru**	**Varlık ismi %**	**Tür %**	**Bağlantı ilişkisi %**
Bag of words	0.29	0.45	0.34	-	-	-
Varlık ismi	0.61	0.60	0.61	100	-	-
Varlık ismi + tür	**0.72**	0.69	**0.71**	34	66	-
Varlık ismi + tür +bağlantı ilişkisi	0.37	**0.81**	0.49	10	26	64

Yukarıdaki Çizelge ikinci kullanıcıdan elde edilen veri kümesi ile yapılan deneysel çalışmaların sonuçlarını göstermektedir. Bu veri kümesinde de en iyi sonuç Varlık ismi + tür yönteminde elde edilmiştir (F1 = 0.71). Varlık ismi + tür + bağlantı ilişkisi kötü sonuç vermiştir (F1 = 0.49). Bunun nedeni ise bağlantı ilişkisinin toplam ağırlıktaki oranının (%64) fazla olması ve gürültü üretebilecek çok sayıda bağlantının bulunmasıdır.

Çizelge 13. 3. kullanıcı için öbekleme performansı

Yöntem	**Precision**	**Recall**	**F1 skoru**	**Varlık ismi %**	**Tür %**	**Bağlantı ilişkisi %**
Bag of words	0.25	0.36	0.29	-	-	-
Varlık ismi	0.62	0.66	0.64	100	-	-
Varlık ismi + tür	**0.64**	**0.72**	**0.67**	33	67	-
Varlık ismi + tür +bağlantı ilişkisi	0.51	0.53	0.56	12	26	62

Bu 3 kullanıcı ile yapılan deneyler, (2.11)'deki formül için en başarılı sonuç veren gösterim şeklini bulmak için yaptığımız değişikliğin (2.17) başarı performansını pek etkilemediğini göstermiştir. Google news veri kümesi için F1 skoru 0.75 hesaplanırken 3 kullanıcı için ortalama F1 skoru 0.74 olarak hesaplanmıştır.

Aşağıdaki Çizelgeler 3 kullanıcının için varlık + tür yöntemi için "silhoutte" katsayısı performans analizi gösterilmektedir.

Çizelge 14. 1. kullanıcı için silhoutte katsayısı performansa katkısı

	Silhoutte katsayı	**F1 skoru**	**Cluster Sayısı**
Seviye 1	-	-	1
Seviye 2	0.18	0.61	10
Seviye 3	**0.52**	**0.81**	20
Seviye 4	0.35	0.54	31

Çizelge 15. 2. kullanıcı için silhoutte katsayısı performansa katkısı

	Silhoutte katsayı	**F1 skoru**	**Cluster Sayısı**
Seviye 1	-	-	1
Seviye 2	0.18	0.56	9
Seviye 3	0.32	0.65	17
Seviye 4	**0.48**	**0.71**	26

Çizelge 16. 3. kullanıcı için silhoutte katsayısı performansa katkısı

	Silhoutte katsayı	**F1 skoru**	**Cluster Sayısı**
Seviye 1	-	-	1
Seviye 2	0.20	0.49	11
Seviye 3	**0.56**	**0.67**	20
Seviye 4	0.35	0.44	32

Silhoutte katsayısını kullanımı, son 3 Çizelgeden da anlaşılacağı gibi, hiyerarşik öbeklemeyi parçalı öbeklemeye otomatik olarak çevirebilmek için kullanılabilecek

uygun bir yöntem olduğu gözlemlenmiştir. Katsayının en yüksek olduğu seviyeler F1 skorunun da en yüksek olduğu seviyelerdir.

6. SONUÇLAR

Bu tez çalışmasında, bir doküman öbekleme problemi olan haber makalelerinin öbeklenmesi problemi üzerine çalışılmıştır. Geliştirdiğimiz yöntemde önceki çalışmalardan farklı olarak, haber makalelerindeki varlık isimleri, tür bilgileri ve varlık isimleri arasındaki bağlantı ilişkileri kullanılmıştır. Varlık isimlerinin tür bilgileri ve bağlantı ilişkilerinin bulmak için bir Anlamsal web (Semantic web) projesi olan ve internette açık olarak bulunan verileri birbirine bağlamayı amaçlayan Bağlı veri (Linked data) veri kümelerinden yararlanılmıştır. Varlık isimleri ve bağlantı ilişkileri için DBpedia, varlık isimlerinin tür bilgileri içinse Yago veri kümesi kullanılmıştır. Yöntemimiz de, daha önceki çalışmalarda kullanılan vektör uzayı modeli (vector space model) gibi doküman gösterim modelleri tercih edilmemiştir. Haber makaleleri, makalelerde geçen varlık isimleri arasında varlık, tür ve varlıklar arası ilişki bilgilerine göre çizge ile modellenmiş ve bu çizge üzerinde kenar ağırlıkları hesaplanarak haberler arası ilişki değeri hesaplanmış; haberler arasında ikili ilişki ağırlıkları bir benzerlik matrisi oluşturularak hiyerarşik öbekleme yapılmıştır. Kullanılan varlık ilişkilerin, yönteme olan katkısını ölçmek için bu ilişkilerin 3 farklı birleşimi denenmiştir. Bunlar sadece varlık isminin, varlık ismi ve tür bilgisinin ve son olarak varlık ismi, tür bilgisi ve bağlantı ilişkisinin birlikte kullanıldığı yöntemlerdir.

Yapılan deneysel çalışmalar, geliştirdiğimiz yöntemin kelime çantası “bag of words” yöntemine göre çok daha başarılı olduğunu göstermiştir. Kelime çantası (bag of words) yönteminde herhangi bir anlamsal ilişki kullanımı yoktur. Bu yöntemde haber içerisinde geçen ama haberin asıl anlatmak istediği konuyla ilgisi olmayan kelimelerin çokluğu çok fazla gürültü üretebilmektedir. Bizim geliştirdiğimiz yöntemde haberdeki sadece kişi yer, şirket gibi haberi asıl tanımlayabilecek varlık isimleri kullanılmıştır. Böylece konu ile ilgisi olmayan ve gürültü üretebilecek kelimeler çıkarılmıştır. Kenar ağırlıklandırmada kullandığımız 3 farklı ilişki arasında en iyi sonucu veren varlık ismi ve tür bilgisinin beraber kullanıldığı yöntem olmuştur. Buradan tür bilgisinin performansı arttırıcı etkisi olduğu gözlemlenmiştir.

En kötü sonuç veren ise 3 ilişkinin de birlikte kullanıldığı yöntem olmuştur, buradan bağlantı ilişkilerinin çok fazla gürültü ürettiği sonucuna varılmıştır.

Bu tez çalışmasında, varlık isimleri arasındaki anlam ilişkilerini kullanarak haber makalelerini öbekleme problemine farklı bir yaklaşım öne sürüldü. Çalışmanın devamında, geliştirdiğimiz yöntem kullanılarak kullanıcı odaklı haber öneri sistemleri, haber makalelerindeki olay tespiti (event detection), haber analiz sistemleri gibi alanlarda çalışmalar yapılabilir. Ayrıca bu yöntemle haber makalelerinden farklı olarak Facebook, Twitter gibi sosyal ağlardaki kullanıcı bilgileri (durum güncellemeleri, arkadaşlıkları, ilgi alanları, vb.), kullanılarak birbirleriyle ilgili olan kullanıcıların öbeklenmesi ve bu kullanıcılar için öneri sistemleri üzerinde çalışmalar yapılabilir.

7. KAYNAKLAR

[1] Wong, SK Michael, Wojciech Ziarko, and Patrick CN Wong. "Generalized vector spaces model in information retrieval." In *Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 18-25. ACM, 1985.

[2] Tsatsaronis, George, and Vicky Panagiotopoulou. "A generalized vector space model for text retrieval based on semantic relatedness." In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pp. 70-78. Association for Computational Linguistics, 2009

[3] Salton, Gerard, Anita Wong, and Chung-Shu Yang. "A vector space model for automatic indexing." *Communications of the ACM* 18, no. 11 (1975): 613-620.

[4] Becker, Jörg, and Dominik Kuropka. "Topic-based vector space model." In*Proceedings of the 6th international conference on business information systems*, pp. 7-12. 2003.

[5] Polyvyanyy, Artem, and Dominik Kuropka. *A Quantitative Evaluation of the Enhanced Topic Based Vector Space Model*. Univ.-Verlag, 2007.

[6] Schenker, Adam, Horst Bunke, Mark Last, and Abraham Kandel. "Clustering of web documents using graph representations." *Applied Graph Theory in Computer Vision and Pattern Recognition* (2007): 247-265.

[7] Hammouda, Khaled M., and Mohamed S. Kamel. "Efficient phrase-based document indexing for web document clustering." *Knowledge and Data Engineering, IEEE Transactions on* 16, no. 10 (2004): 1279-1296.

[8] Zamir, Oren, and Oren Etzioni. "Web document clustering: A feasibility demonstration." In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 46-54. ACM, 1998.

[9] Hu, Xiaohua, Xiaodan Zhang, Caimei Lu, E. K. Park, and Xiaohua Zhou. "Exploiting Wikipedia as external knowledge for document clustering." In*Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 389-396. ACM, 2009.

[10] Gabrilovich, E. and S. Markovitch. 2007. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. *Proceedings of IJCAI*, 1606-1611.

[11] Wang, Pu, Jian Hu, Hua-Jun Zeng, and Zheng Chen. "Using Wikipedia knowledge to improve text classification." *Knowledge and Information Systems*19, no. 3 (2009): 265-281.

[12] Michael Strube and Simon Paolo Ponzetto. WikiRelate! Computing semantic relatedness using Wikipedia. *In AAAI'06*, Boston, MA, 2006.

[13] Szczuka, Marcin, Andrzej Janusz, and Kamil Herba. "Clustering of rough set related documents with use of knowledge from DBpedia." *Rough Sets and Knowledge Technology* (2011): 394-403.

[14] Hossain, M. Shahriar, and Rafal A. Angryk. "Gdclust: A graph-based document clustering technique." In *Data Mining Workshops, 2007. ICDM Workshops 2007. Seventh IEEE International Conference on*, pp. 417-422. IEEE, 2007

[15] Banerjee, Somnath, Krishnan Ramanathan, and Ajay Gupta. "Clustering short texts using wikipedia." In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 787-788. ACM, 2007.

[16] Phan, Xuan-Hieu, Le-Minh Nguyen, and Susumu Horiguchi. "Learning to classify short and sparse text & web with hidden topics from large-scale data collections." In *Proceedings of the 17th international conference on World Wide Web*, pp. 91-100. ACM, 2008.

[17] Fellbaum, Christiane. "WordNet." *Theory and Applications of Ontology: Computer Applications* (2010): 231-243.

[18] Dumais, Susan T. "Latent semantic analysis." *Annual Review of Information Science and Technology* 38, no. 1 (2005): 188-230.

[19] Can, Fazli, and Esen A. Ozkarahan. "Computation of term/document discrimination values by use of the cover coefficient concept." *Journal of the American Society for Information Science* 38, no. 3 (1987): 171-183.

[20] "Linked Open Data gösterimi" erişim adresi: http://www.phibetaiota.net/2012/12/talking-frog-linked-open-data-lod-101/

[21] Hofmann, Thomas. "Probabilistic latent semantic indexing." In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 50-57. ACM, 1999.

[22] Blei, David M., and John D. Lafferty. "Topic models." *Text mining: classification, clustering, and applications* 10 (2009): 71.

[23] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." *the Journal of machine Learning research* 3 (2003): 993-1022.

[24] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." *the Journal of machine Learning research* 3 (2003): 993-1022.

[25] Deb, Kalyanmoy. "Multi-objective optimization." *Multi-objective optimization using evolutionary algorithms* (2001): 13-46.

[26] Ward Jr, Joe H. "Hierarchical grouping to optimize an objective function."*Journal of the American statistical association* 58, no. 301 (1963): 236-244.

[27] Hartigan, John A., and Manchek A. Wong. "Algorithm AS 136: A k-means clustering algorithm." *Applied statistics* (1979): 100-108.

[28] Johnson, Richard Arnold, and Dean W. Wichern. *Applied multivariate statistical analysis*. Vol. 4. Englewood Cliffs, NJ: Prentice hall, 1992.

[29] Steinbach, Michael, George Karypis, and Vipin Kumar. "A comparison of document clustering techniques." In *KDD workshop on text mining*, vol. 400, pp. 525-526. 2000.

[30] Birant, Derya, and Alp Kut. "ST-DBSCAN: An algorithm for clustering spatial–temporal data." *Data & Knowledge Engineering* 60, no. 1 (2007): 208-221.

[31] Ankerst, Mihael, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. "OPTICS: ordering points to identify the clustering structure." *ACM SIGMOD Record* 28, no. 2 (1999): 49-60.

[32] Cheng, Yizong, and George M. Church. "Biclustering of expression data." In*Proceedings of the eighth international conference on intelligent systems for molecular biology*, vol. 8, pp. 93-103. 2000.

[33] Steinbach, Michael, George Karypis, and Vipin Kumar. "A comparison of document clustering techniques." In *KDD workshop on text mining*, vol. 400, pp. 525-526. 2000.

[34] Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. "Hierarchical clustering." *The elements of statistical learning* 2 (2009).

[35] Sas Institute. *SAS/Stat 9.2 User's Guide: The Glimmix Procedure (Book Excerpt)*. Edited by Publishing SAS Publishing. Sas Inst, 2008.

[36] Sokal R and Michener C. "A statistical method for evaluating systematic relationship" *University of Kansas Science Bulletin*, vol. 38 pp. 1409-1438. 1958.

[37] Jensen, E.C., Beitzel, S.M., Pilotto, A.J., Goharian, N., Frieder, O.: Parallelizing the buckshot algorithm for efficient document clustering. In: CIKM 2002: 11th int. conf. on Information and knowledge management, pp. 684–686. ACM Press, New York (2002)

[38] MacKay, David. "An example inference task: clustering." *Information Theory, Inference and Learning Algorithms* (2003): 284-292.

[39] Rousseeuw, Peter J. "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis." *Journal of computational and applied mathematics* 20 (1987): 53-65.

[40] Olson, David L., and Dursun Delen. *Advanced data mining techniques*. Springer, 2008.

[41] “Kelime çantası modeli (bag of words)” erişim adresi: http://en.wikipedia.org/wiki/Bag-of-words_model, erişim tarihi : 15 Aralık 2012.

[42] Hotho, Andreas, Steffen Staab, and Gerd Stumme. "Ontologies improve text document clustering." In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pp. 541-544. IEEE, 2003.

[43] Budanitsky, Alexander, and Graeme Hirst. "Evaluating wordnet-based measures of lexical semantic relatedness." *Computational Linguistics* 32, no. 1 (2006): 13-47.

[44] Hotho, Andreas, Alexander Maedche, and Steffen Staab. "Ontology-based text document clustering." *KI* 16, no. 4 (2002): 48-54.

[45] Bizer, Christian, Tom Heath, and Tim Berners-Lee. "Linked data-the story so far." *International Journal on Semantic Web and Information Systems (IJSWIS)*5, no. 3 (2009): 1-22.

[46] Szczuka, Marcin, Andrzej Janusz, and Kamil Herba. "Clustering of rough set related documents with use of knowledge from DBpedia." *Rough Sets and Knowledge Technology* (2011): 394-403.

[47] Issal, C., 2010, Document Clustering, *Yüksek Lisans Tezi*, *Göteborg Üniversitesi*, Göteborg İsviçre.

[48] Kim, Y., 2010, Document Clustering in a Learned Concept Space, *Doktora Tezi,* Marie Curie Universitesi, Paris Fransa.

[49] Brücher, Heide, Gerhard Knolmayer, and Marc-André Mittermayer. "Document classification methods for organizing explicit

knowledge." *Research Group Information Engineering, Institute of Information Systems, University of Bern, Engehaldenstrasse* 8 (2002).

[50] Brian S. Everitt, Sabine Landau, and Morven Leese. Cluster Analysis. Oxford University Press, fourth edition, 2001.

[51] Berners-Lee, Tim, James Hendler, and Ora Lassila. "The semantic web."*Scientific american* 284, no. 5 (2001): 28-37.

[52] Lassila, Ora, and Ralph R. Swick. "Resource description framework (RDF) model and syntax specification." (1998).

[53] "Anlamsal ağ standartları" erişim adresi: http://www.w3.org/2001/sw/wiki/Main_Page, erişim tarihi: 11 Aralık 2012

[54] Brickley, Dan, and Ramanathan V. Guha. "Resource Description Framework (RDF) Schema Specification 1.0: W3C Candidate Recommendation 27 March 2000." (2000).

[55] McGuinness, Deborah L., and Frank Van Harmelen. "OWL web ontology language overview." *W3C recommendation* 10, no. 2004-03 (2004): 10.

[56] Berners-Lee, Tim. "Notation 3." *The World Wide Web Consortium(W3C) MIT, INRIA, http://www. w3. org/DesignIssues/Notation3. html. Design Suggestion*(1998).

[57] Bizer, Christian, Tom Heath, and Tim Berners-Lee. "Linked data-the story so far." *International Journal on Semantic Web and Information Systems (IJSWIS)*5, no. 3 (2009): 1-22.

[58] van Rijsbergen, Cornelis Joost. "A theoretical basis for the use of co-occurrence data in information retrieval." *Journal of Documentation* 33, no. 2 (1977): 106-119.

[59] Dhillon, Inderjit S., and Dharmendra S. Modha. "Concept decompositions for large sparse text data using clustering." *Machine learning* 42, no. 1 (2001): 143-175.

[60] Knoblock, Craig, Daniel Lopresti, Shourya Roy, and L. Venkata Subramaniam. "Special issue on noisy text analytics." *International Journal on Document Analysis and Recognition* 10, no. 3 (2007): 127-128.

[61] Rigouste, Loïs, Olivier Cappé, and François Yvon. "Inference and evaluation of the multinomial mixture model for text clustering." *Information processing & management* 43, no. 5 (2007): 1260-1280.

[62] Li, Yanjun, Soon M. Chung, and John D. Holt. "Text document clustering based on frequent word meaning sequences." *Data & Knowledge Engineering* 64, no. 1 (2008): 381-404.

[63] Gustafson, Donald E., and William C. Kessel. "Fuzzy clustering with a fuzzy covariance matrix." In *Decision and Control including the 17th Symposium on Adaptive Processes, 1978 IEEE Conference on*, vol. 17, pp. 761-766. IEEE, 1978.

[64] Murtagh, Fionn. "A survey of recent advances in hierarchical clustering algorithms." *The Computer Journal* 26, no. 4 (1983): 354-359.

[65] Andrews, Nicholas O., and Edward A. Fox. "Recent developments in document clustering." (2007).

[66] “İnternet’in şu anki durumuyla ilgili veriler” erişimi adresi: http://news.netcraft.com/archives/2012/03/05/march-2012-web-server-survey.html erişim tarihi : 17 Aralık 2012.

[67] Han, Jiawei, Micheline Kamber, and Anthony KH Tung. "Spatial clustering methods in data mining: A survey." *Geographic data mining and knowledge discovery. Taylor and Francis* 21 (2001): 48.

[68] Witten, Daniela M., and Robert Tibshirani. "A framework for feature selection in clustering." *Journal of the American Statistical Association* 105, no. 490 (2010): 713-726.

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı : OTO, Salih Atılay

Uyruğu : T.C.

Doğum tarihi ve yeri : 23.10.1985 Ankara

Medeni hali : Bekar

Telefon : 0 (312) 292 42 94

e-mail : saoto@etu.edu.tr

Eğitim

Derece	**Eğitim Birimi**	**Mezuniyet tarihi**
Lisans	Hacettepe Üniversitesi/Bilgisayar	2009

İş Deneyimi

Yıl	**Yer**	**Görev**
2010-2012	TOBB Ekonomi ve Teknoloji Üniversitesi	Araştırma Görevlisi

Yabancı Dil

İngilizce

Yayınlar

Hakimov, Sherzod, Salih Atilay Oto, and Erdogan Dogdu. "Named Entity Recognition and Disambiguation using Linked Data and Graph-based Centrality Scoring." *Proceedings of the 4th International Workshop on Semantic Web Information Management*. ACM, 2012