LİNGO ALGORİTMASININ KÜMELERLE İLİŞKİLİ DOKÜMANLARIN BELİRLENMESİ VE KÜME ETİKETLERİNİN ÇIKARILMASI AŞAMALARININ İYİLEŞTİRİLMESİ

ENHANCING THE CLUSTER CONTENT DISCOVERY AND THE CLUSTER LABEL INDUCTION PHASES OF THE LINGO ALGORITHM

SEYFULLAH DEMİR

PROF. DR. HAYRİ SEVER

Tez Danışmanı

Hacettepe Üniversitesi

Lisansüstü Eğitim – Öğretim ve Sınav Yönetmeliğinin

Bilgisayar Mühendisliği Anabilim Dalı İçin Öngördüğü

YÜKSEK LİSANS TEZİ

olarak hazırlanmıştır.

2013

SEYFULLAH DEMİR'in hazırladığı **"Lingo Algoritmasının Kümelerle İlişkili Dokümanların Belirlenmesi ve Küme Etiketlerinin Çıkarılması Aşamalarının İyileştirilmesi"** adlı bu çalışma aşağıdaki jüri tarafından **BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI**'ında **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Başkan

(Doç. Dr. Ebru Akçapınar Sezer)

Üye (Danışman)

(Prof. Dr. Hayri Sever)

Üye

(Yrd. Doç. Dr. Nazlı İkizler Cinbiş)

Üye

(Yrd. Doç. Dr. İbrahim Aykut Erdem)

Üye

(Öğr. Gör. Dr. Murat Hacıömeroğlu)

Bu tez Hacettepe Üniversitesi Fen Bilimleri Enstitüsü tarafından **YÜKSEK LİSANS TEZİ** olarak onaylanmıştır.

Prof. Dr. Fatma SEVİN DÜZ

Fen Bilimleri Enstitü Müdürü

ETİK

Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, tez yazım kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- tez içindeki bütün bilgi ve belgeleri akademik kurallar çerçevesinde elde ettiğimi,
- görsel, işitsel ve yazılı tüm bilgi ve sonuçları bilimsel ahlak kurallarına uygun olarak sunduğumu,
- başkalarının eserlerinden yararlanılması durumunda ilgili eserlere bilimsel normlara uygun olarak atıfta bulunduğumu,
- atıfta bulunduğum eserlerin tümünü kaynak olarak gösterdiğimi,
- kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- ve bu tezin herhangi bir bölümünü bu üniversite veya başka bir üniversitede başka bir tez çalışması olarak sunmadığımı

beyan ederim.

20.09.2013

Seyfullah DEMİR

ÖZET

LİNGO ALGORİTMASININ KÜMELERLE İLİŞKİLİ DOKÜMANLARIN BELİRLENMESİ VE KÜME ETİKETLERİNİN ÇIKARILMASI AŞAMALARININ İYİLEŞTİRİLMESİ

SEYFULLAH DEMİR

Yüksek Lisans, Bilgisayar Mühendisliği Bölümü

Tez Danışmanı: Prof. Dr. HAYRİ SEVER

Eylül 2013, 75 Sayfa

Arama Sonucu Kümeleme (ASK) algoritmaları, kullanıcıların arama motorları üzerinde aradıkları sonuçlara daha kolay erişebilmeleri için geliştirilen algoritmalardır. İyi bir ASK algoritmasının hem arama sonuçlarını doğru kümelemesi, hem de oluşturduğu kümelere; kümeleri temsil edebilen, anlaşılır ve anlamlı etiketler üretmesi beklenir.

Lingo algoritması her iki kritere de önem veren popüler bir ASK algoritmasıdır. Lingo algoritması, oluşturduğu kümeler için bahsedildiği şekilde başarılı etiketler üretebilmektedir; ancak kümelerin elemanlarını belirleme konusunda bazı eksiklikleri bulunmaktadır. Algoritmada uygulanan kümelere doküman atama stratejisinin sonucu olarak, küme etiketlerindeki terimleri içermeyen; ancak, aslında etiketlerle anlamsal olarak ilişkili olan dokümanlar ilgili kümelere

atanamamaktadır. Ayrıca, sonuç küme etiketlerinin belirlenmesi için kullanılan yöntem, az sayıda ilgili sonuç içeren kümelerin ortaya çıkmasına neden olmaktadır. Bu eksiklikler düşük anma (recall) değerine sebep olmaktadır.

Bu tezde, Lingo algoritmasındaki sözkonusu eksiklikleri gidererek iyileştirme sağlamayı amaçlayan; ilki kümelere ilişkili dokümanların atanması aşamasında, diğeri kümelerin etiketlerinin çıkarılması aşamasında olmak üzere iki değişiklik önerisi sunulmaktadır. Deney sonuçları, önerilen değişikliklerin anma değerini büyük oranda iyileştirdiğini göstermektedir.

Anahtar Kelimeler: bilgi erişim dizgeleri, arama sonucu kümeleme, küme elemanlarını belirleme, küme etiketleme

ABSTRACT

ENHANCING THE CLUSTER CONTENT DISCOVERY AND THE CLUSTER LABEL INDUCTION PHASES OF THE LINGO ALGORITHM

SEYFULLAH DEMİR

Master of Science, Department of Computer Engineering

Supervisor: Prof. Dr. HAYRİ SEVER

September 2013, 75 Pages

Search Results Clustering (SRC) algorithms are developed so that users can reach to the results that they search for easier. A good SRC algorithm is expected to correclty cluster the search results, and also to be able to generate representative, understandable and meaningful cluster labels for the produced clusters.

The Lingo algorithm is a popular SRC algorithm that notice both two criterions. It is able to generate successful cluster labels as expected; however, it has some shortcomings about determining the cluster contents. As a consequence of its cluster content assignment strategy, semantically relevant documents that do not contain the terms of the cluster labels could not be assigned to the related clusters. Moreover, the method that is used to select final cluster labels results in

clusters containing small number of relevant results. These shortcomings cause low recall values.

In this thesis, two modification proposals that aim to overcome the shortcomings of the Lingo algorithm are presented. The first modification proposal is for the cluster content discovery phase, and the other is for the cluster label induction phase. The experiment results show that the proposed modifications improve the low recall values to quite higher values.

Keywords: information retrieval, search results clustering, cluster content discovery, cluster labeling

TEŞEKKÜR

Tez çalışmaları süresince bilgi ve tecrübelerini esirgemeyen, tez danışmanım Prof. Dr. Sn. Hayri Sever'e,

Tez çalışmalarında vermiş olduğu destek ve yardımlarından dolayı Doç. Dr. Sn. Ebru Akçapınar Sezer'e,

Fikir ve görüşleriyle çalışmama destek olan Burak İbrahim Sevindi, Gonca Hülya Doğan, Özgür Bağlıoğlu, Onur Erdoğan ve Çağlar Günel'e,

Yüksek lisans çalışmalarımı destekleyen kurumum TÜBİTAK – BİLGEM – YTE'ye ve iş arkadaşlarıma,

Hiçbir zaman desteğini esirgemeyen sevgili arkadaşlarıma,

Her zaman yanımda olan, maddi manevi desteklerini esirgemeyen, benim için yaptıkları büyük fedakarlıklarla başarılarımın en büyük kaynağı olan başta anne ve babam olmak üzere bütün aileme teşekkür ederim.

İÇİNDEKİLER DİZİNİ

Sayfa

ÖZET i
ABSTRACT iii
TEŞEKKÜR v
İÇİNDEKİLER DİZİNİ vi
ŞEKİLLER DİZİNİ viii
ÇİZELGELER DİZİNİ ix
SİMGELER VE KISALTMALAR DİZİNİ x
SÖZLÜK DİZİNİ xii
1. GİRİŞ 1
2. İLGİLİ ÇALIŞMALAR 4
3. TEMEL BİLGİ 10
3.1. Arama Sonucu Kümeleme 10
3.1.1. Arama Motoru API'si 11
3.1.2. Vektör Uzay Modeli 11
3.1.3. Uzaklık Ölçüm Yöntemleri ve Sorgulama 14
3.1.4. Gizli Anlamsal İndeksleme 15
3.2. Carrot^2 Çatısı 17
3.3. Lingo 19
3.3.1. Orijinal Lingo Algoritması 19
3.3.2. Güncel Lingo Algoritması 23
4. YÖNTEM 29
4.1. Kümelere İlişkili Dokümanların Atanması Aşamasındaki Değişiklik 29
4.2. Kümelerin Etiketlerinin Çıkarılması Adımındaki Değişiklik 33
5. DENEY 39
5.1. Kullanılan Veri Seti 39
5.2. Kullanılan Metrikler 40
5.3. Yapılan Deney 42
5.4. Deney Sonuçları 43
6. SONUÇ VE ÖNERİLER 59
KAYNAKLAR 62
EKLER 67

ÖZGEÇMİŞ 74

ŞEKİLLER DİZİNİ

Sayfa

Şekil 3.1 Arama Motoru API'si Sonucu Örneği 11
Şekil 3.2 Tekil Değer Ayrışımı Sonuç Matrisleri..... 17
Şekil 3.3 Carrot2 Doküman Kümeleme Tezgahı Programının Örnek Ekran Görüntüsü 19
Şekil 3.4 Orijinal Algoritmasına Genel Bakış 20
Şekil 3.5 Güncel Lingo Algoritmasının Gerçekleştirimindeki Temel Java Sınıflarını İçeren Sınıf Diyagramı..... 25
Şekil 4.1 Güncel Atama Stratejisi Örneği 30
Şekil 4.2 Önerilen Atama Stratejisi Örneği 30
Şekil 4.3 Önerilen Doküman Atama Stratejisi Öncesi Sınıf Diyagramı 31
Şekil 4.4 Önerilen Doküman Atama Stratejisi Sonrası Sınıf Diyagramı..... 32
Şekil 4.5 Güncel Soyut Kavram-Etiket Adayı Eşleştirme Stratejisi 33
Şekil 4.6 Soyut Kavram-Etiket Adayı Eşleştirmesi İçin İlk Öneri..... 34
Şekil 4.7 Dokümanların Soyut Kavramlarla İlişkilendirilmesi İçin Önerilen Yöntemler..... 35
Şekil 4.8 Soyut Kavram-Etiket Adayı Eşleştirmesi İçin İkinci Öneri - Örnek Eşleştirme 36
Şekil 4.9 Önerilen Soyut Kavram - Etiket Adayı Eşleştirme Stratejisi Öncesi Sınıf Diyagramı..... 37
Şekil 4.10 Önerilen Soyut Kavram - Etiket Adayı Eşleştirme Stratejisi Sonrası Sınıf Diyagramı..... 38
Şekil 5.1 Bütün Başlıklar için Ortalama F-Ölçütü, Duyarlılık ve Anma Grafiği 44
Şekil 5.2 Bütün Başlıklar İçin Ortalama Küme Kirliliği ve Normalleştirilmiş Ortak Bilgi Grafiği..... 46
Şekil 5.3 Beagle Başlığı için Adım 1 ve Adım 8'de Üretilen İlk 10 Sonuç Küme Etiketleri 53
Şekil 5.4 Labyrinth Başlığı için Adım 1 ve Adım 8'de Üretilen İlk 10 Sonuç Küme Etiketleri 54
Şekil 5.5 Jaguar Başlığı için Adım 1 ve Adım 8'de Üretilen İlk 10 Sonuç Küme Etiketleri 54

ÇİZELGELER DİZİNİ

Sayfa

Çizelge 3.1 Güncel Lingo Algoritmasındaki Bazı Önemli Parametre ve Eşik Tanımları 27
Çizelge 5.1 Duyarlılık, Anma ve F-ölçütü 40
Çizelge 5.2 Sıralı Deney Adımlarının Tanımları 42
Çizelge 5.3 Bütün Başlıklar için Ortalama Metrik Değerleri 43
Çizelge 5.4 Beagle Başlığı İçin Doğru Küme Tanımları 48
Çizelge 5.5 Labyrinth Başlığı İçin Doğru Küme Tanımları 48
Çizelge 5.6 Jaguar Başlığı İçin Doğru Küme Tanımları 49
Çizelge 5.7 Beagle Başlığı İçin Adım 1 ve Adım 8 Sonuçları 50
Çizelge 5.8 Labyrinth Başlığı İçin Adım 1 ve Adım 8 Sonuçları 51
Çizelge 5.9 Jaguar Başlığı İçin Adım 1 ve Adım 8 Sonuçları 52
Çizelge 5.10 Her bir Başlık için Sonuçların Ağırlıklı Ortalama Değerleri 55
Çizelge 5.11 Her bir Başlığın Sonuçların Ağırlıklı Ortalama Değerlerinin Adım 8'de Adım 1'e göre Değişim Miktarı 57

SİMGELER VE KISALTMALAR DİZİNİ

AMBIENT : AMBIguous ENTries

AJAX : Asynchronous JavaScript and XML

API : Application Programming Interface

ASK : Arama Sonucu Kümeleme

CC : Cluster Contamination

DK : Doğru Küme

F : F-Measure

HTTP : HyperText Transfer Protocol

JAWS : Java Api for Wordnet Searching

LINGO : Label INduction Grouping algOrithm

LNMF : Local Non-negative Matrix Factorization

LSI : Latent Semantic Indexing

NMF-ED : Non-negative Matrix Factorization with Euclidean Distance

NMF-KL : Non-negative Matrix Factorization with Divergence algorithm

NMI : Normalized Mutual Information

ODP : Open Directory Project

P : Precision

R : Recall

REST : REpresentational State Transfer

SAK : Sonek Ağacı Kümeleme

SHOC : Semantic, Hierarchical, Online Clustering

SRC : Search Results Clustering

STC : Suffix Tree Clustering

SVD : Singular Value Decompositon

TF : Term Frequency

TF-IDF : Term Frequency – Inverse Document Frequency

URL : Uniform Resource Locator

VSM : Vector Space Model

XML : EXtensible Markup Language

SÖZLÜK DİZİNİ

Açık Dizin Projesi	: Open Directory Project
Aday Etiket Eşik Değeri	: Candidate Label Threshold
Anında	: On-the-fly
Anma	: Recall
Arama Sonucu Kümeleme	: Search Results Clustering
Aşama	: Phase
Bağlantı	: Link
Başlık	: Topic
Başlık	: Title
Belirsiz	: Ambiguous
Belirsizlik Giderme Sayfaları	: Disambiguation Pages
Belirtkeleme	: Tokenization
Bilgi Erişimi	: Information Retrieval
Çok-aracılı	: Multi-agent
Devrik	: Transpose
Dışarı-bağlantı	: Out-link
Dikkat Çekici	: Salient
Doküman Atama Eşik Değeri	: Snippet Assignment Threshold
Duyarlılık	: Precision
Etiket Benzerlik Eşik Değeri	: Label Similarity Threshold
Etkisiz Etiket	: Stop Label
Etkisiz Kelimeler	: Stop Words
F-ölçütü	: F-measure
Geçici	: Temporal
Gizli Anlamsal İndeksleme	: Latent Semantic Indexing

İçeri-bağlantı : In-link

İkili Getirim Modeli : Boolean Retrieval Model

İleri-işleme : Post-processing

Kavramsal : Conceptual

Kelime Torbası : Bag of Words

Kısa Özet : Snippet

Konu : Topic

Kök-çözümleyici : Lemmatizer

Küme Birleştirme Eşik Değeri : Cluster Merging Threshold

Küme Kirliliği : Cluster Contamination

Matris Çarpanlarına Ayırma : Matrix Factorization

Normalleştirilmiş Ortak Bilgi : Normalized Mutual Information

Ön işleme : Preprocessing

Örtüşen Kümeler : Overlapping Clusters

Sezgisel : Heuristic

Sıkılama : Stemming

Sıkılayıcı : Stemmer

Sıralama : Ranking

Sonek Ağacı Kümeleme : Suffix Tree Clustering

Sonek Dizisi : Suffix Array

Soyut Kavram : Abstract Concept

Tanımlama-Farkındalığı-Olan : Description-Aware

Tanımlama-Merkezli : Description-Centric

Tanımlama-Önce-Gelir : Description-Comes-First

Tekil Değer Ayrışımı : Singular Value Decomposition

Tolerans Kaba Küme Kümelemesi : Tolerance Rough Set Clustering

Uygulama Programı Arabirimi : Application Programming Interface

Vektör Uzay Modeli	: Vector Space Model

1. GİRİŞ

Günümüzde web, insanlar için çok önemli bir bilgi kaynağı haline gelmiştir. Öyle ki, bir bilgiye ihtiyaç duyulduğunda, başvurulacak kaynakların başında internet üzerinden erişilebilen popüler arama motorları gelmektedir. Kullanıcılar bir sorgu metni belirleyerek arama motorları üzerinde sorguyu çalıştırmakta ve dönen sonuçlar üzerinden aradıkları bilgiye ulaşmaya çalışmaktadırlar.

Klasik arama motorları, sorgulama işlemi sonucunda, sorgu metniyle ilişkili sonuçları kısa özetleriyle (snippet) birlikte ve sıralı olarak kullanıcıya ulaştırır [1]. Kullanıcılar bekledikleri sonuçlara erişmek için bu sıralı listedeki sonuçları teker teker gözden geçirmek durumundadır. Bu yüzden, benzer ya da aynı özellikteki sonuçlar gruplanmadığı için, birden fazla anlamı olan ya da çok genel terimler içeren sorgu metinleri ile sorgulama yapıldığında, ilgili sonuçlara ulaşmak zorlaşabilmektedir.

Klasik arama motorlarındaki bu dezavantajı ortadan kaldırmak için, Arama Sonucu Kümeleme (ASK) algoritmaları geliştirilmiştir. Popüler ASK algoritmaları, benzer özellikteki arama sonuçlarının birlikte kümelenmesini ve bu kümelere; kümeleri temsil edebilen, anlamlı ve anlaşılır etiketler üretilmesini amaçlar. Başarılı ASK algoritmaları sayesinde kullanıcılar, sorgulama sonucunda arama sonuçlarını etiketlenmiş, sıralı kümeler halinde görebilir ve aradıkları sonuçlara daha kolay ulaşabilirler.

Lingo algoritması, özellikle; kümeleri tanımlayıcı, anlamlı ve anlaşılır etiketler üretme konusuna odaklanmış popüler bir ASK algoritmasıdır. Öyle ki, algoritma akış sürecinde klasik kümeleme algoritmalarından farklı olarak; önce küme etiketleri belirlenmekte, kümelere atanacak sonuçlara ise daha sonra karar verilmektedir. Bu özelliğinden dolayı literatürde tanımlama-merkezli (description-centric) anlayışındaki algoritmalar arasında gösterilmekte [2], geliştiricileri tarafından ise bir tanımlama-önce-gelir (description-comes-first) algoritma olarak takdim edilmektedir [1]. Lingo algoritması, arama motorundan sorgulama sonucu olarak ilgili arama sonuçları döndürüldükten sonra devreye giren bir ileri-işleme (post-processing) algoritmasıdır.

Lingo algoritmasında küme etiketleri, bütün arama sonucu listesinde sık geçen ifadeler içinden seçilmektedir. Bu doğrultuda, algoritmanın ön işleme

(preprocessing) aşamasında sık geçen ifadeler belirlenmekte ve etiket adayı olarak işaretlenmektedir. İlerleyen aşamalarda, Gizli Anlamsal İndeksleme (Latent Semantic Indexing) yönteminden faydalanılarak tüm sonuç listesindeki soyut kavramlar (abstract concept) keşfedilmekte ve her bir soyut kavram için bir etiket adayı ile eşleştirme yapılmaktadır. Bu aşamada, soyut kavramlarla eşleşen etiket adayları algoritma sonucunda elde edilecek kümelerin etiketi olarak belirlenmektedir. Etiketleri belirlenmiş kümeler için küme elemanlarını belirleme işlemi ise, kümelerin etiketlerindeki terimleri içeren sonuçların ilgili etiketlerle temsil edilen kümelere atanması yaklaşımıyla gerçekleştirilmektedir.

Lingo algoritmasında iki ana eksiklik farkedilmiştir. Bunlardan ilki, küme etiketleriyle anlamsal olarak ilişkili olduğu halde, etiketlerdeki terimleri içermediğinden dolayı ilgili kümelere atanamayan sonuçların düşük anma değerlerine sebep olmasıdır. Diğer eksiklik ise soyut kavramlar ile küme etiket adaylarının eşleştirilme stratejisinde göze çarpmaktadır. Eşleştirme yapılırken soyut kavramlar ve etiket adaylarının terim vektörleri, vektörel olarak karşılaştırılmaktadır; ancak, etiket adayları genelde az sayıda terimden oluştuğundan bu karşılaştırmanın çok sağlıklı sonuç vermeyebileceği düşünülmüştür.

Bu tezde, söz konusu iki eksikliği gidermeyi amaçlayan iki ayrı değişiklik önerisi sunulmaktadır. İlk öneri kapsamında, WordNet [3] sözlüğünden faydalanılarak küme etiketlerindeki terimleri içermese de etiketlerle anlamsal olarak ilişkili olan sonuçların da ilgili kümelere atanabilmesi hedeflenmektedir. Önerilen yönteme göre kümelere, yalnızca küme etiketindeki terimleri içeren sonuçların değil, aynı zamanda bu terimlerin WordNet sözlüğüne göre eş anlamlılarını içeren sonuçların da atanabilmesi sağlanacaktır. İkinci öneri ile ise, soyut kavramların, kendilerine daha çok benzeyen etiket adaylarıyla eşleştirilebilmeleri amaçlanmıştır. Bu doğrultuda, öncelikle soyut kavramlarla ilişkili sonuçların belirlenmesi sağlanacak, ardından soyut kavramlar ve etiket adayları, benzerlik ölçütü olarak ortaklaşa ilişkili oldukları sonuç sayısı kullanılacak şekilde karşılaştırılarak, eşleştirilecektir.

Deney sonuçları, önerilen değişikliklerin mevcut Lingo algoritmasına göre daha başarılı sonuçlar sağladığını göstermektedir.

Dokümanın ilerleyen bölümlerindeki akış şu şekildedir:

2. Bölüm'de ASK algoritmaları ile ilgili literatürdeki çalışmalar özetlenmiştir.

3. Bölüm'de Lingo algoritmasıyla ilişkili olan temel kavramlar kısaca anlatılmıştır.

4. Bölüm'de tez kapsamında önerilen değişiklikler açıklanmış ve 5. Bölüm'de deney için kullanılan veri seti ve metrikler, yapılan deneyler ve deneylerin sonuçları açıklanmıştır.

Son olarak, 6. Bölüm'de tez çalışmasının sonucu özetlenmiş ve gelecek çalışmalar için öneriler sunulmuştur

2. İLGİLİ ÇALIŞMALAR

Arama Sonuçları Kümeleme (ASK), doküman kümeleme probleminin bir alt çalışma alanıdır. Çevrimiçi kümeleme gerektirdiği için, doküman kümeleme probleminden farklı olarak daha etkili ve anında (on-the-fly) çözümler gerektirir [4].

Carpineto ve arkadaşları [2], ASK sistemleri ile ilgili detaylı bir araştırma sunmuşlardır. Bir ASK sisteminin; sorgu sonucundaki arama sonuçlarının elde edilmesi, sonuçların ön işleme işleminden geçirilmesi, kümelerin oluşturulması ve etiketlenmesi ve sonuçların görsel olarak sunulması olmak üzere 4 ana bileşenden oluştuğunu ifade etmişlerdir. Kümelerin oluşturulması ve etiketlenmesi bileşeninde kullanılan stratejiye göre ASK algoritmalarını; veri-merkezli [5][6][7][8][9][10], tanımlama-farkındalığı-olan [11][12][13][14][15] ve tanımlama-merkezli [1][16][17][18] olmak üzere üç sınıfa ayırmışlardır.

Web arama sonuçlarının kümelenmesi fikri, ilk kez Scatter-Gather sistemi ile ortaya atıldı [5][6]. Ardından, ASK algoritmaları içerisinde, dokümanların benzerliği için tekrar eden ifadelerin kullanımının ilk kez önerildiği Sonek Ağacı Kümeleme (Suffix Tree Clustering) algoritması tanıtıldı [19][11][20][12]. Grafiksel kullanıcı arayüzü sağlayan, çevrimiçi çalışan Grouper sisteminde SAK algoritmasının gerçekleştirimi yapıldı [12].

SAK algoritması ve türevlerinde genel olarak, sık geçen ifadeleri ortaklaşa içeren dokümanların birlikte gruplanması yaklaşımı esas alınmıştır. SAK algoritmasının dezavantajlarını ortadan kaldırmak için çeşitli çalışmalar yapılmıştır [15][21].

Çince dili için geliştirilmiş SHOC algoritmasında, SAK algoritmasındaki düşük kaliteli ifade problemini de ortadan kaldırmak için sonek ağacı yerine sonek dizisi (suffix array) [22] kullanımı tercih edilmiştir [23][24]. SHOC algoritması; tam ifadelerin çıkarılması, baz kümelerin belirlenmesi ve küme hiyerarşisinin oluşturulması olmak üzere üç ana fazdan oluşmaktadır. İlk aşamada, sonek dizisi veri yapısı kullanılarak tam ifade (complete phrase) olarak adlandırılan yapıdaki ifadelerin tespiti yapılır. İkinci aşamada **T**ekil **D**eğer **A**yrışımı (**S**ingular **V**alue **D**ecomposition) yöntemi ile baz kümeler belirlenirken, üçüncü ve son aşamada baz kümelerin hiyerarşik bir yapıda birleştirilmesi işlemi gerçekleştirilir.

Janruang ve Kreesuradej [25], SHOC ile benzer şekilde, orijinal SAK algoritmasında üretilen kesilmiş olması muhtemel küme etiketlerini saf dışı

bırakarak, daha başarılı etiketler üretmeye çalışmışlardır. Bu amaçla, ilk etapta oluşturulan baz kümelerin önerdikleri yeni küme birleştirme tekniğiyle birleştirilmesini sağlayarak, daha az sayıda ve daha uzun ve anlaşılır etiketlere sahip baz kümeler oluşturmuşlardır.

Wen ve arkadaşları [26], SAK algoritmasında sonuç kümeler oluştuktan sonra, her bir küme için olası etiketlerden hangisinin küme etiketi olarak seçileceğine karar verilirken Gizli Anlamsal İndeksleme metodunun kullanılmasını önermişlerdir. Her küme için, hangi etiket adayının indirgenmiş anlamsal uzayda daha çok terim karşılığı varsa, o etiketin ilgili kümeyle ilgili daha çok anlamsal bilgi taşıdığını savunmuşlardır.

Zeng ve arkadaşları [18], arama sonucu kümeleme problemini bir dikkat çekici (salient) etiket sıralama problemi olarak ele almışlardır. Bunun için, öncelikle dikkat çekici ifadeleri tespit edip küme etiket adayı olarak belirlemiş, daha sonra insanlarca etiketlenmiş eğitim verisinden öğrenilen bir regresyon modeline dayalı olarak bu ifadeleri sıralamışlardır.

Mecca ve arkadaşları [27], genel yaklaşımdan farklı olarak arama sonuçlarının özetleri üzerinde ileri-işleme yapmak yerine, bütün dokümanlar kümesi üzerinde Gizli Anlamsal İndeksleme yapmayı önermişlerdir. Çalışmalarında ayrıca, optimum sayıdaki tekil değerlere erişebilmek için geliştirdikleri; dinamik ve özgün bir Tekil Değer Ayrışımı stratejisini tanıtmışlardır.

Bazı araştırmacılar, web sayfaları ya da arama sonuçları arasındaki bağlantıların (link), kümeleme problemi için iyi bir özellik olacağını düşünmüşlerdir [28][29][30][31][32][33]. Bu bağlamda, Wang ve Kitsuregawa [32]; metin içeriği, içeri-bağlantı (in-link) ve dışarı-bağlantı (out-link) özelliklerinin kombinasyonunu kullandıkları bir öneri sunmuşlardır. Metin içeriğinin daha önemli bilgi sağlamasına karşın; hem bağlantı bilgilerinin, hem de metin içeriğinin web arama sonucu kümeleme için önemli olduğunu göstermişlerdir.

Duhan ve Sharma [33], hem içerik, hem de bağlantı tabanlı benzerliği kullanarak kümeleme ve sıralamaya dayalı bir algorirtma önermişlerdir. Kümeleme işleminin içerik benzerliğine göre gerçekleştirildiği, ardından her bir küme için, küme elemanlarının içerik ve bağlantı benzerliğine göre sıralandığı bir yaklaşım sunmuşlardır.

Jayanthi ve Prem [34], hem içerik, hem de oturum tabanlı bir ASK algoritması önermişlerdir. Bu kapsamda, sorgu metinleri için arama sonuçlarında seçilen sonuçları ve hangi sorgu metinleri için benzer sonuçların seçildiği bilgisini kullanmışlardır. Kullanıcı loglarına ve metin içeriğine dayalı algoritmaların, bunlardan yalnızca birine dayalı olan algoritmalara göre daha başarılı olduğunu göstermişlerdir.

Alonso ve Gertz [35], arama sonuçlarının oluşturulma ve değiştirilme tarihi gibi geçici (temporal) metaveri değerlerini ve arama sonuçlarının metinsel içeriklerindeki tarih, zaman bilgisi gibi geçici referansları kullanan bir kümeleme yaklaşımı önermişlerdir. Bu yaklaşımda, aynı geçici değerleri paylaşan sonuçların birlikte gruplanması önerilmiştir.

Alam ve Sadaf [36], sezgisel (heuristic) arama ve Gizli Anlamsal İndeksleme yaklaşımlarının birlikte kullanıldığı bir algoritma önermişlerdir. Algoritmaya göre, web arama sonuçları grafiği kullanılarak sezgisel bir yaklaşımla kümeler bulunmakta, ardından kümelerdeki sonuçlar ve sorgu metni vektörü için LSI metodu uygulanarak, kümelerdeki sonuçların sorgu metniyle anlamsal benzerlik derecelerine göre sıralanması sağlanmaktadır.

Bazı araştırmacılar, Wikipedia [37] kaynağını bilgi tabanı olarak kullanan yaklaşımlar önermişlerdir [4][38][39].

Çallı [4], küme etiketleri ve dokümanlar arasındaki anlamsal ilişkileri Wikipedia [37] yardımıyla ortaya çıkarıp, gürültülü etiketlerin filtrelenmesini ve SAK algoritmasının birleştirme aşamasını iyileştirmeyi amaçladığı çalışmasını paylaşmıştır. Wikipedia'daki anlamsal bilgilerin SAK'a entegrasyonu için öneriler sunmuştur. Bu çalışma ayrıca, ASK çalışmalarında Türkçe dili ile çalışılması konusunda da öncü bir çalışma olmuştur.

Carmel ve arkadaşları [39], verilen kümelere etiket belirleme işlemi için Wikipedia'dan faydalanmışlardır. Kümelerin metinsel içerikleriyle benzeyen Wikipedia makalelerinin kategori ve başlık gibi metaveri bilgilerinin, metinsel içerikli belgelerin kümeleri için çok başarılı etiketler üretebildiğini göstermişlerdir.

ASK algoritmalarında; anlamlı, anlaşılır ve kümeleri temsil edebilen etiketler üretebilmek çok önemli bir konudur. Bu konuya önem vererek, öncelikli olarak başarılı etiketler üretmeye dayalı yaklaşımlar güden algoritmalar tanımlama-

merkezli algoritmalar olarak sınıflandırılmaktadır [2]. Bu sınıftaki yaklaşımlar içinde öncü olan yaklaşım Vivisimo'ya [16] göre, iyi bir küme; iyi ve okunabilir bir tanımlamaya sahip olan kümedir. Vivisimo'nun "tanımlama-önce-gelir" sloganını benimseyen en başarılı algoritmalardan biri Lingo [1][40] algoritmasıdır.

Metin kümeleme algoritmalarının büyük çoğunluğunda; önce küme içeriklerinin belirlenip, daha sonra içeriğe dayalı olarak küme etiketlerinin belirlenmesi şablonuna uyulurken, Lingo algoritmasında bunun aksine, önce küme etiketleri belirlenip, daha sonra etiketlere dayalı olarak küme içeriklerine karar verilmektedir [1]. Lingo algoritması, SHOC algoritmasından ilham alınarak ve bu algoritmanın eksik görülen yanları iyileştirilerek geliştirilmiştir [40]. SHOC'ta önerilen sonek dizisine dayalı ifade çıkarımı yaklaşımı, Lingo algoritmasında adapte edilerek kullanılmıştır [40]. Lingo algoritmasında SHOC'tan farklı olarak:

- Ön işleme aşamasında, her bir arama sonucu için kullanılan dilin tespiti adımı eklenmiştir.
- Etiketlerin belirlenmesi işlemi küme içeriklerinin belirlenmesinden önce ve Tekil Değer Ayrışımı yaklaşımı kullanılarak gerçekleştirilmiştir.
- Kümelerin içeriğine karar verme işlemi Tekil Değer Ayrışımı yaklaşımına dayalı olarak değil, **V**ektör **U**zay **M**odeli (**V**ector **S**pace **M**odel) yaklaşımı kullanılarak ve küme etiketlerine dayalı olarak gerçekleştirilmiştir.
- Kümelerin belirlenmesinden sonra küme birleştirme işlemi gerçekleştirilmemiştir.

Osinski [40], Lingo'nun ilk kez sunulduğu yüksek lisans tezinde, Lingo için son kullanıcıların yorumlarına dayalı olarak detaylı bir değerlendirme sunmuştur. Ardından, Osinski ve Weiss [41], **A**çık **D**izin **P**rojesi'ndeki (**O**pen **D**irectory **P**roject) [42], insanlar tarafından önceden kategoriler halinde dizinlenmiş sayfaları kullanarak yaptıkları değerlendirmeyi paylaşmışlardır. Rastgele seçilen kategorilerdeki sonuçları bir araya getirerek test kümeleri oluşturup, Lingo'nun bu sonuçları nasıl ayrıştırdığını incelemişlerdir. Deneysel sonuçlara dayalı olarak yaptıkları analiz, her ne kadar çoğunlukla sayısal bir analiz olmasa da, Lingo'nun arama sonuçlarındaki konuları (topics) ayrıştırmada en azından SAK algoritmasına göre daha başarılı olduğunu ortaya koymuşlardır [41]. Osinski ve Weiss [43], daha sonra küme kirliliği (cluster contamination) [44] açısından Lingo ve SAK

algoritmalarını karşılaştırarak sayısal analize dayalı bir değerlendirme sunmuşlardır. Bu çalışma ile Lingo'nun SAK'tan çok daha saf, başka bir deyişle çok daha az kirlenmiş, kümeler oluşturduğunu göstermişlerdir. İlerleyen dönemde, Osinski [45], ASK algoritmasında tercih edilen matris çarpanlarına ayırma (matrix factorization) yönteminin etkilerini araştırmıştır. Bu çalışma kapsamında; konu ayrıştırma, aykırı değer tespiti ve etiket kalitesi açısından, aşağıda listelenen 4 ayrı metodu değerlendirmiştir:

- Tekil Değer Ayrıştırma (**S**ingular **V**alue **D**ecomposition),
- Negatif Olmayan Matris Çarpanlarına Ayırma (**N**on-negative **M**atrix **F**actorisation),
- Yerel, Negatif Olmayan Matris Çarpanlarına Ayırma (**L**ocal **N**on-negative **M**atrix **F**actorisation),
- Kavram Ayrıştırma (**C**oncept **D**ecomposition)

Bu çalışma sonucunda Osinski [45], negatif olmayan matris çarpanlarına ayırma yöntemlerinin SVD de dahil olmak üzere diğer metotlardan daha başarılı olduğunu göstermiştir. Ayrıca, deneysel sonuçlara göre, Lingo algoritmasında matris çarpanlarına ayırma yöntemi olarak öklit uzaklığı minimizasyonu kullanılan negatif olmayan matris çarpanlarına ayırma yöntemi (NMF-ED) kullanıldığında, konu ayrıştırma ve aykırı değer tespiti açılarından, SAK ve Tolerans Kaba Küme Kümelemesi (Tolerance Rough Set Clustering) [7] algoritmalarına göre önemli derecede daha iyi sonuçlar alındığı gösterilmiştir.

Lingo algoritmasının küme içeriklerini belirleme aşamasındaki, küme etiketlerindeki terimleri içermeyen, ancak anlamsal olarak kümelerle ilişkili sonuçların ilgili kümelere atanamaması problemini gidermek için bazı çalışmalar gerçekleştirilmiştir [46][47].

Sameh ve Kadray [46], Lingo algoritmasının Sık Geçen İfadelerin Belirlenmesi aşaması için bir değişiklik önerisi sunmuşlardır. WordNet [3] sözlüğünü kullanarak mevcut etiket adayları listesine etiket adaylarının eş anlamlılarını da eklemişlerdir. Etiketlerin eş anlamlılarından aynı zamanda küme içeriğine karar verilirken de yararlanmışlardır.

Alsulami ve arkadaşları [47], Sameh ve Kadray'ın [46] çalışmasındakine benzer şekilde WordNet aracılığıyla küme etiket adaylarının, eş anlamlı ifadeleriyle çeşitlendirilmesini önermişlerdir. Farklı olarak, sorgu metinlerinin eş anlamlı ifadelerinin de etiket adayı olabilmesine imkan vermişler, ayrıca farklı arama motorlarından sonuç getirebilen çok-aracılı (multi-agent) bir sistem sunmuşlardır. Bir diğer önemli özgün yaklaşımları ise, ön-işleme aşamasında sıkılayıcı (stemmer) yerine kök-çözümleyici (lemmatizer) kullanmaları olmuştur.

3. TEMEL BİLGİ

Bu bölümde tez için temel oluşturan bazı bilgi erişimi (information retrieval) kavramları ve Lingo ASK algoritması özetlenmiştir. İlk kısımda Lingo algoritmasına altlık eden bazı genel kavramlar anlatılırken, sonrasında Lingo algoritmasının gerçekleştirimin yapıldığı Carrot^2 çatısı ve son olarak da Lingo algoritmasının orijinal ve güncel versiyonları anlatılmıştır.

3.1. Arama Sonucu Kümeleme

ASK algoritmalarının ana amacı, arama motorlarından dönen arama sonuçlarının, benzer sonuçlar birlikte kümelenecek şekilde kümelere ayrıştırılarak uygun etiketlerle kullanıcılara sunulmasıdır. Weiss [48], ASK algoritmalarının genel gereksinimlerini aşağıdaki gibi sıralamıştır:

- Hızlı, tercihen doğrusal-zamanlı ve artımlı olmalıdır.
- Kısa metin parçalarıyla dahi iyi sonuçlar üretmelidir.
- Örtüşen kümelerin oluşmasına imkan sağlamalıdır; çünkü arama sonuçları genelde birden fazla konu içermektedir.
- Küme etiketleri sezgisel olarak anlaşılabilir olmalıdır; öyle ki kullanıcılar üretilen etiketlere gore küme içeriklerini tahmin edebilmelidir.

Web arama sonuçları sayfaların kısa özetlerinden oluştuğu için, kısıtlı oranda metinsel verilerle ifade edilen belgeler kümelenmeye çalışılmaktadır. Arama sonuçlarındaki bu kısıt, ASK algoritmalarında arama sonuçları için metinsel bilgi dışında; metaveri, içeri-bağlantı, dışarı-bağlantı, harici bilgi tabanı kavramları ile anlamsal ilgililik değerleri gibi diğer özelliklerin de değerlendirilmeye çalışılması sonucunu doğurmuştur. Metinsel bilginin kullanıldığı yaklaşımlarda, arama sonuçları genellikle vektör uzay modelinde temsil edilerek kümelenmektedir. Vektör uzay modelinde her bir belge bir vektör olarak ifade edilir ve belgelerin benzerliği vektör benzerlik yöntemlerine göre değerlendirilir. Bu başlık altında devam eden kısımlarda, arama motorlarının genel API'si, vektör uzay modeli, uzaklık ölçüm yöntemleri ve gizli anlamsal indeksleme alt başlıklarında ASK algoritmalarında kullanılan genel kavramlar detaylandırılmıştır.

3.1.1. Arama Motoru API'si

ASK sistemlerinde, arama sonuçlarının elde edilmesi aşamasında, sonuçların getirileceği arama motorlarında sorgular çalıştırılarak sonuçlar alınır. Bu işlem ilgili arama motorlarının **U**ygulama **P**rogramı **A**rabirimi (**A**pplication **P**rogramming **I**nterface) kullanılarak gerçekleştirilir.

Bir arama motoru API'sinde temelde sorgu metni girdi olarak alınırken; ilişkili sayfaların kısa özeti (snippet), başlık, URL gibi özellikleriyle sağlanan arama sonuçlarının listesi sonuç olarak döndürülür. Şekil 3.1'de HTTP istemi aracılığıyla kullanılan Ajax tabanlı bir Google arama motoru API'sinin, "arama motoru" sorgu metnine karşılık dönen sonuç listesinden bir sonuç örneği verilmiştir.


```
{
        "GsearchResultClass":"GwebSearch",
        "unescapedUrl":"http://tr.wikipedia.org/wiki/Arama_motoru",
        "url":"http://tr.wikipedia.org/wiki/Arama_motoru",
        "visibleUrl":"tr.wikipedia.org",
        "cacheUrl":"http://www.google.com/search?q\u003dcache:-9nwgmj4GVAJ:tr.wikipedia.org",
        "title":"\u003cb\u003eArama motoru\u003c/b\u003e - Vikipedi",
        "titleNoFormatting":"Arama motoru - Vikipedi",
        "content":"\u003cb\u003eArama motoru\u003c/b\u003e, İnternet üzerinde bulunan içeriği
        aramak için kullanılan bir  mekanizmadır. Üç bileşenden oluşur: web robotu, arama indeksi ve
        kullanıcı  arabirimi."
}
```


Şekil 3.1 Arama Motoru API'si Sonucu Örneği

3.1.2. Vektör Uzay Modeli

Bilgi Erişim sistemlerinde metinsel dokümanları temsil edebilmek ve işlemek için çeşitli matematiksel yöntemler önerilmiştir. Dokümanların temsili için kullanılan yaygın yaklaşım kelime torbası (bag of words) yaklaşımıdır. Bu yaklaşımla dokümanlar, kelimelerin dokümandaki sırası gözetilmeksizin içerdikleri kelimelerle temsil edilir. Bir dilin sıkça kullanılan, tek başına anlamı olmayan, cümle ögelerini birbirine bağlamak için ya da çeşitli yardımcı görevleri yerine getirmek için kullanılan, metinsel belgeler için ayırt edici bir özellik olarak kullanılamayan

kelimeler etkisiz kelimeler (stop words) olarak değerlendilir ve kelime torbalarına dahil edilmezler. Daha hızlı işlem yapılabilmesinin sağlanması da etkisiz kelimelerin kelime torbalarına dahil edilmemesinin başka bir sebebidir. Farklı amaçlar için farklı etkisiz kelime listeleri kullanılabilir. *Ama*, *ancak*, *bazı*, *çoğu* gibi kelimeler Türkçe dili için etkisiz kelimelere örnek olarak verilebilir.

Dokümanların kelime torbası yaklaşımı ile ifade edilebilmesi için vektör gösterimi kullanılabilir. Bir doküman kümesi aynı mantıkla, kelime-doküman matrisiyle ifade edilebilir. Bu yapıda, her doküman kelimeler uzayında bir nokta ile, kelime-doküman matrisinde ise bir vektör ile temsil edilir. Bilgi erişim sistemlerinde, bu yapıda bir sorgulama ve erişim probleminin gerçekleştirilmesi için önerilen yaklaşımlardan temel ikisi; İkili Getirim Modeli (Boolean Retrieval Model) ve Vektör Uzay Modeli'dir (Vector Space Model).

İkili Getirim Modeli'nde doküman vektörlerinin, dokümanların içerdiği kelimelere karşılık gelen hücrelerinde 1, diğer hücrelerinde ise 0 verisi saklanır. Farklı bir bakış açısıyla, kelime-doküman matrisinde, her kelime için, o kelimeyi barındıran dokümanlara karşılık gelen hücrelerde 1, diğer hücrelerde 0 saklanır. Bu modelde bir sorgu işletmek için, terim-doküman matrisinde sorguda geçen kelimelere karşılık gelen satırlar seçilir ve bu satırlar üzerinde sorgu kelimeleri arasındaki ikili işleçlere karşılılık gelen işlemler uygulanır. İşlemler sonucunda 1 değerine sahip hücrelere karşılık gelen dokümanlar sorgu sonucunda ilişkili olan dokümanları oluşturur.

Kelimeler uzayında, doküman kümesinde geçen bütün kelimelerin kullanılması yerine, kelimeler üzerinde sıkılama (stemming) işlemi uygulanarak elde edilen kelime gövdeleri kullanılır. Aynı kavramı ifade eden; ancak sonuna eklenen ve anlamını değiştirmeyen eklerden dolayı farklı bir kelime olarak ortaya çıkan kelimelerin, doküman benzerliği değerlendirilirken dezavantaj oluşturmaması için bu kullanım tercih edilmektedir. Dokümanın ilerleyen bölümlerinde bu detayın kafa karıştırıcı etkisini ortadan kaldırmak için kelime-doküman matrisi yerine, sıkılama işlemi sonucu elde edilen terimlerin kullanıldığının kastedildiği, terim-doküman matrisi ifadesi kullanılmaktadır.

Vektör Uzay Modeli'nde, İkili Getirim Modeli'nden farklı olarak terim-doküman matrisinde her bir hücrede terimlerin dokümanlarla ilişkililik derecesine göre

ağırlıklandırılmış değerler saklanır. Sorgu metni de dokümanlarla benzer şekilde aynı terimler uzayında bir vektörle ifade edilir. Sorgu vektörüyle, doküman vektörlerinin benzerlikleri hesaplanarak sorgulama sonucunda ilişkili dokümanların belirlenmesi sağlanır. Bu yaklaşım sayesinde sonuç dokümanlar, sorguyla benzerlik derecesine göre azalan sırada olacak şekilde (ranking) kullanıcıya sunulabilmektedir. Vektör Uzay Modeli, bilgi erişim modelleri arasında kullanılması en kolay olan ve en verimli modeldir [49].

Terim-doküman matrisinde hücrelerin ağırlıklandırılması için kullanılan yöntemler aşağıda özetlenmiştir. Daha detaylı bilgi [49]'da verilmiştir.

3.1.2.1. Terim Ağırlıklandırma Yöntemleri

Terim ağırlandırma, terimlerle dokümanlar arasındaki ilişkilerin derecesinin hesaplandığı bir işlemdir. Bu bölümde, en popüler üç terim ağırlıklandırma metodu anlatılmıştır. Bu metotlar, terim *i* ile doküman *j* arasındaki ilişkinin derecesinin a_{ij} ile gösterildiği varsayılarak anlatılmıştır.

İkili Ağırlıklandırma (Binary Weighting)

Terim *i*, doküman *j* içerisinde geçiyorsa $a_{ij} = 1$, aksi takdirde $a_{ij} = 0$ olarak ağırlandırma yapılır. Bu ağırlıklandırma yönteminde, terim-doküman ilişkilerinde terimin dokümanda geçip geçmediği belirlenebilmekte; ancak aralarındaki ilişkinin derecesi ile ilgili bir bilgi elde edilememektedir.

Terim Frekansı Ağırlıklandırması (Term Frequency Weighting)

Bu yöntemde tf_{ij}, terim *i*'nin doküman *j* içerisinde kaç defa geçtiğini ifade etmek üzere, $a_{ij} = tf_{ij}$ olarak ağırlıklandırma yapılır. İkili ağırlıklandırma yöntemi ile karşılaştırıldığında, terimler ve dokümanlar arasındaki ilişkinin derecesini verebiliyor olması ile farklılaşır. Ancak bu metot, doküman yerelinde bir ağırlıklandırma metodu olmasının dezavantajına sahiptir. Kısaca *tf* olarak ifade edilir.

Terim Frekansı-Ters Doküman Frekansı Ağırlıklandırması (Term Frequency-Inverse Document Frequency Weighting)

Bu yöntem; terim frekansı ağırlıklandırma yöntemindeki dezavantajı ortadan kaldırarak, terimlerle dokümanlar arasındaki ilişkinin derecesinin, tüm doküman kümesine bağlı olacak şekilde hesaplanmasına olanak veren bir yöntemdir. Terimlerin dokümanlardaki yerel ve genel bulunma sıklığını dengeleyici bir yaklaşım kullanılır. Örneğin, bir dokümanın bütün dokümanlarda çokça geçen bir

terim ile olan ilişkisinin derecesinin, yalnızca o dokümanda ve diğer terimden daha az sayıda geçen bir terimle olan ilişkisinin derecesinden daha küçük olmasına olanak verir. Bu yöntemde, df_i terim *i*'nin geçtiği toplam doküman sayısı ve *N* de toplam doküman sayısı olmak üzere, $a_{ij} = tf_{ij} \cdot log\ (N / df_i)$ olarak ağırlıklandırma yapılır. Formülde, $log\ (N / df_i)$ olarak hesaplanan değer terim *i*'nin global ağırlıklandırması için hesaplanır ve ters doküman frekansı olarak da bilinir. Bu formülün terim frekansı-ters doküman frekansı (*tf-idf*) olarak adlandırılmasının sebebi de budur.

3.1.3. Uzaklık Ölçüm Yöntemleri ve Sorgulama

İki metnin karşılaştırılması için literatürde Hamming uzaklığı, Levenstein uzaklığı gibi pek çok farklı yöntem önerilmiştir [50]. Vektör Uzay Modeli'nde sorgu vektörleri ile doküman vektörlerinin, ya da iki doküman vektörünün birbiriyle benzerliklerinin karşılaştırılması için de farklı yöntemler önerilmiştir. Jaccard katsayısı, öklit uzaklığı ve kosinüs uzaklığı, vektörlerin benzerliğini hesaplamak için kullanılan en popüler yöntemlerdendir. Bu başlık altında öklit uzaklığı ve kosinüs uzaklığı yöntemleri açıklanmıştır. Ardından kosinüs benzerliği yöntemi kullanılarak vektör uzay modelinde sorgulama işleminin nasıl gerçekleştirildiği anlatılmıştır.

3.1.3.1. Öklit uzaklığı

İki nokta arasındaki doğrusal uzaklıktır. $P = (p_1, p_2, \ldots, p_n)$ ve $Q = (q_1, q_1, \ldots, q_n)$, *n* boyutlu bir uzayda iki nokta olmak üzere, bu noktalar arasındaki öklit uzaklığı $\sqrt[2]{\sum_{i=1}^{n}(p_i - q_i)^2}$ formülüyle hesaplanır [51].

3.1.3.2. Kosinüs uzaklığı

İki vektör arasındaki açısal yakınlığın baz alındığı bir benzerlik yöntemidir. Vektörlerin benzerlik derecesini 0 ile 1 kapalı aralık değerlerinde ifade edebilmek için, benzerlik vektörler arasındaki açının kosinüs değeriyle ifade edilir. Örneğin, birbiriyle çakışan, başka bir deyişle aynı yöndeki iki vektör arasındaki açının kosinüs değeri $cos\ 0^\circ = 1$'dir ve bu vektörlerin birbirlerine tamamen benzediği söylenebilir.

İki vektörün kosinüs uzaklığı yöntemine göre benzerlik derecesi, vektörlerin nokta çarpımı formülü üzerinden hesaplanabilir. Nokta çarpımının cebirsel yöntemle ve geometrik olarak tanımları aşağıda verilmiştir.

Cebirsel Tanım: $P = (p_1, p_2, \ldots, p_n)$ ve $Q = (q_1, q_1, \ldots, q_n)$, n boyutlu iki vektör olmak üzere, bu vektörlerin nokta çarpımı $P \cdot Q = \sum_{i=1}^{n} \mathrm{p_i} q_i$ olarak hesaplanır.

Gemetrik Tanım: Aynı *P ve Q* vektörlerinin nokta çarpımı geometrik tanımla, $P \cdot Q = \|P\|\|Q\| \cos\theta$ formülüyle ifade edilir.

Nokta çarpımının geometrik tanımla verilen formülünde, $\cos\theta$ değerini yalnız bıraktığımızda vektörlerin kosinüs benzerliği yöntemine göre hesaplanan, 0 ile 1 arasında değerler alabilen benzerlik değerini veren formülü, $\cos\theta = \frac{P \cdot Q}{\|P\|\|Q\|} = \frac{P}{\|P\|} \cdot \frac{Q}{\|Q\|}$ olarak elde ederiz. Formüldeki $\frac{P}{\|P\|}$ ve $\frac{Q}{\|Q\|}$, *P ve Q* vektörlerinin birim vektörleridir. Bu bilgi ışığında kosinüs uzaklığı yöntemine göre benzerlik değerinin, iki vektörün birim vektörlerinin nokta çarpımıyla elde edildiğini söylebiliriz. Birim vektörleri *n x 1*'lik matris yapılarıyla ifade edildiğinde vektörlerin nokta çarpımı, ilk matrisin devriğiyle (transpose) ikinci matrisin kendisinin matris çarpımıyla da hesaplanabilmektedir.

Vektör uzay modelinde, sorguların ve dokümanların terimler uzayında vektörlerle ifade edildiği belirtilmişti. Aynı terimler uzayında, terim – doküman matrisiyle ifade edilen doküman vektörleri ve sorgu vektörü birim uzunluğa göre normalize edildiğinde, sorgu vektörünü ifade eden matrisi (terim – sorgu matrisi) *q* ile, terim – doküman matrisini ise *A* ile gösterirsek $q^T A$ matris çarpımı, sorgunun her bir dokümanla benzerlik derecesini veren sorgu – doküman matrisini verir. Önceden belirlenmiş bir benzerlik eşik değerinden daha yüksek derecede bir değerle sorguya benzeyen dokümanlar, sorguyla ilişkili dokümanlar olarak belirlenebilir [40].

3.1.4. Gizli Anlamsal İndeksleme

Klasik Vektör Uzay Modeli'ndeki sorgulama işleminde, sorguyla ilişkili dokümanların getirilmesi, dokümanların sorguda kullanılan terimleri içermesine bağlıdır. Sorguda belirtilen terimleri içermemesine rağmen, aslında kavramsal (conceptual) olarak sorguyla ilişkili olan dokümanlar bu model yaklaşımıyla sorgu sonucu olarak getirilememektedir. Sorgulama mantığının terimlere bağımlı olması, klasik VSM'in zayıflıklarından biridir.

Gizli Anlamsal İndeksleme (**L**atent **S**emantic **I**ndexing) yaklaşımı ile, klasik VSM'in yukarıda belirtilen zayıflığının aşılması hedeflenmiştir. Dokümanların terimlerle

ifade edilmesinin yetersiz olabileceği, aksine kavramlarla (concepts) olan ilişkilerine göre de değerlendirilebileceği düşünülmüştür. Bu doğrultuda LSI yaklaşımı, sorgulama işlemlerinde terimsel benzerliğe dayalı ilgililik yerine, kavramsal benzerliğe dayalı ilgililiği temel almıştır. Bu yaklaşım sayesinde, sorgu metni ile kavramsal olarak benzerlik gösteren dokümanlar, sorgu terimlerini içermese bile sorgu sonucu olarak getirilebilmektedir [40][52].

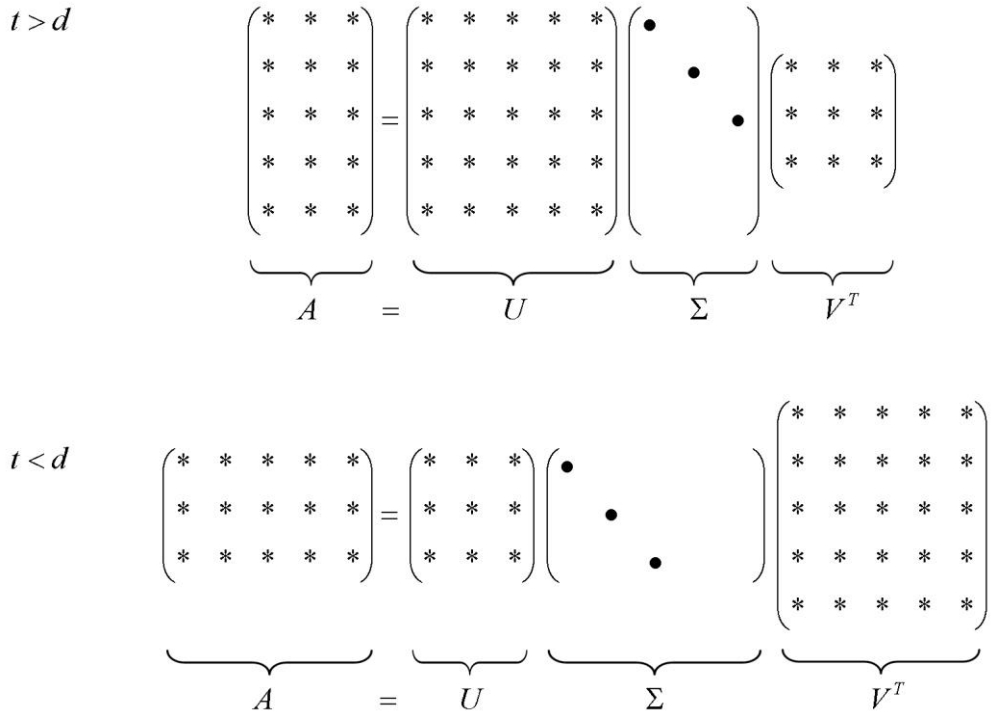
LSI yalnızca sorgulama amaçlı değil, başka amaçlarla da kullanılabilmektedir [40]. Sorgulama için de temel oluşturan bir diğer kullanım amacı, doküman kümesinde geçen soyut kavramların tespit edilebilmesidir [1]. Bir doküman kümesinde açıkça belli olmayan, gizlenmiş anlamsal bilgiyi açığa çıkartarak, ilgili soyut kavramların tespitini mümkün kılmaktadır. Kelimeler arası anlamsal benzerliklerin tespiti, filtreleme, çok boyutlu matrislerin boyutunun gürültü olarak nitelendirilebilecek özelliklerin temizlenerek daha düşük boyutlara indirgenmesi gibi amaçlar için de kullanılabilmektedir.

3.1.4.1. Tekil Değer Ayrıştırma

Terim-doküman matrisi kullanılarak, LSI yaklaşımıyla soyut kavramları tespit edebilmek için cebirsel bir matris çarpanlarına ayırma metodu olan Tekil Değer Ayrıştırma (**S**ingular **V**alue **D**ecomposition) metodundan faydalanılabilir. SVD yöntemi; *t* terim sayısı, *d* de doküman sayısı olmak üzere *t x d* boyutlarındaki bir *A* terim-doküman matrisini, $A = U \Sigma V^T$ olacak şekilde *U*, *Σ* ve *V* matrislerine dönüştürür [1]. *U* matrisi *t x t* boyutlarında bir ortogonal matristir ($UU^T = I$) ve kolon vektörleri *A*'nın sol tekil vektörleri olarak adlandırılır. Benzer şekilde *V* matrisi *d x d* boyutlarında bir ortogonal matristir ($VV^T = I$) ve kolon vektörleri *A*'nın sağ tekil vektörleri olarak adlandırılır. *Σ* matrisi ise koşegeni boyunca *A*'nın tekil değerlerinin azalan sırada yer aldığı *t x d* boyutlarında bir matristir [1]. SVD işlemi sonucu oluşan matrislerin yapısı Şekil 3.2'de gösterilmiştir.

LSI yaklaşımında beklenen kaliteye göre bir *k* değerine karar verilerek yalnızca en büyük ilk *k* tekil değerlerinin değeri alınır ve diğer tekil değerler sıfır olarak belirlenir. Bu işlemin sonucunda orijinal *A* matrisinin benzeri olan bir A_k matrisi elde edilir. A_k matrisi, *A* matrisinin ilk *k* tekil vektörlerinin oluşturduğu uzayda, *A* matrisinin bir izdüşümüdür. A_k matrisi $A \approx A_k = U_{txk}\, \Sigma_{kxk}\, V^T_{kxd}$ formülüyle elde edilir.

Lingo algoritmasında bu ayrışım ile soyut kavramların tespiti yapılmaktadır. Ayrışım sonucunda oluşan matrislerden U_{txk} matrisi terim–soyut kavram matrisi ve V_{kxd}^{T} matrisi soyut kavram-doküman matrisidir [1]. U_{txk} matrisi, doküman kümesindeki soyut kavram vektörlerini verir.



Şekil 3.2 Tekil Değer Ayrışımı Sonuç Matrisleri[1]

3.2. Carrot² Çatısı

Carrot² [53][54], açık kaynak kodlu bir Arama Sonuçları Kümeleme (ASK) çatısıdır. Girdi olarak sağlanan bir doküman kümesini alt kümelere ayrıştırmak için kullanılabilir. Hâlihazırda, K-Means, STC ve Lingo olmak üzere üç popüler ASK algoritmasının gerçekleştirimini içermektedir. Akademik ya da ticari amaçlı olarak bir ASK sistemi geliştirmek için gerekli olan eforu azaltır [4]. Carrot²'nin sağladığı hizmetlerden bazıları aşağıda listelenmiştir [53]:

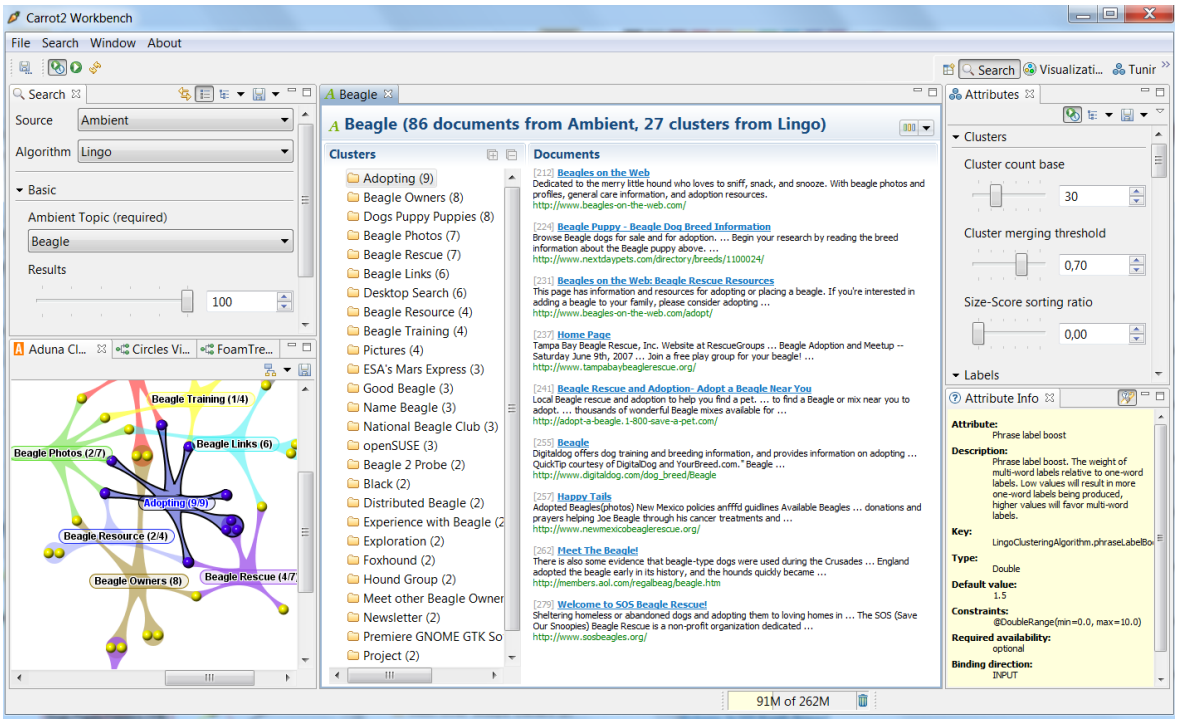
- Kümelenmesi istenen doküman listesini belirlemek için farklı seçenekler sunar. Bing arama motoru, eTools meta-arama motoru, yerel dokümanların indekslendiği arama motorları, OpenSearch standardına uyan arama

[1] [40]'dan alınmıştır.

servisleri ve uygun formatta hazırlanmış XML dosyasındaki sonuçlar, ASK algoritmalarına girdi doküman listesi sağlaması için kullanılabilen bazı seçeneklerdir.

- ODP239 [55] ve AMBIENT [56] veri setlerinin test amaçlı kullanılabilmesine olanak veren bir altyapı sağlanmaktadır.
- ASK algoritmalarının değerlendirilmesinde kullanılan; duyarlılık, anma, f-ölçütü, küme kirliliği, normalleştirilmiş ortak bilgi kalite ölçüm metriklerinin değerlerinin sonuç kümelere göre otomatik olarak hesaplanması sağlanır.
- Metin işleme problemi ile ilgili; etkili belirtkeleme (tokenization), trigram tabanlı dil tespiti, etkisiz kelimelerin filtrelenmesi ve sıkılama gibi yetenekleri vardır. Snowball [57] tarafından desteklenen diller ve Arapça, Lehçe ve Çince de dahil olmak üzere toplamda 32 dil için destek sağlanmaktadır.
- Çalışma süresi ve bellek kullanımı ile ilgili ASK algoritmasının performansının ölçülebilmesi sağlanmaktadır.
- ASK algoritmasının kümeleme işlemi sonucunda oluşturduğu kümelerin grafiksel kullanıcı arayüzünde görüntülenmesi sağlanmaktadır.

Carrot^2 projesinde Java, C# / .NET ve REST API'si sağlanmaktadır [58]. Carrot^2 çatısını hızlı bir şekilde istenen verilerle denemek, gerçek zamanlı olarak parametrik değerleri değiştirerek bu değişikliklerin etkisini gözlemlemek için Carrot^2 Doküman Kümeleme Tezgahı (Carrot^2 Document Clustering Workbench) kullanılabilir. Carrot^2'de ayrıca, komut satırı arayüzü ve geliştirilen ASK'ya göre özelleştirilebilen web tarayıcısı eklentisi ve web uygulaması mevcuttur. Şekil 3.3'de Carrot^2 Doküman Kümeleme Tezgahı programının bir örnek ekran görüntüsü gösterilmiştir.



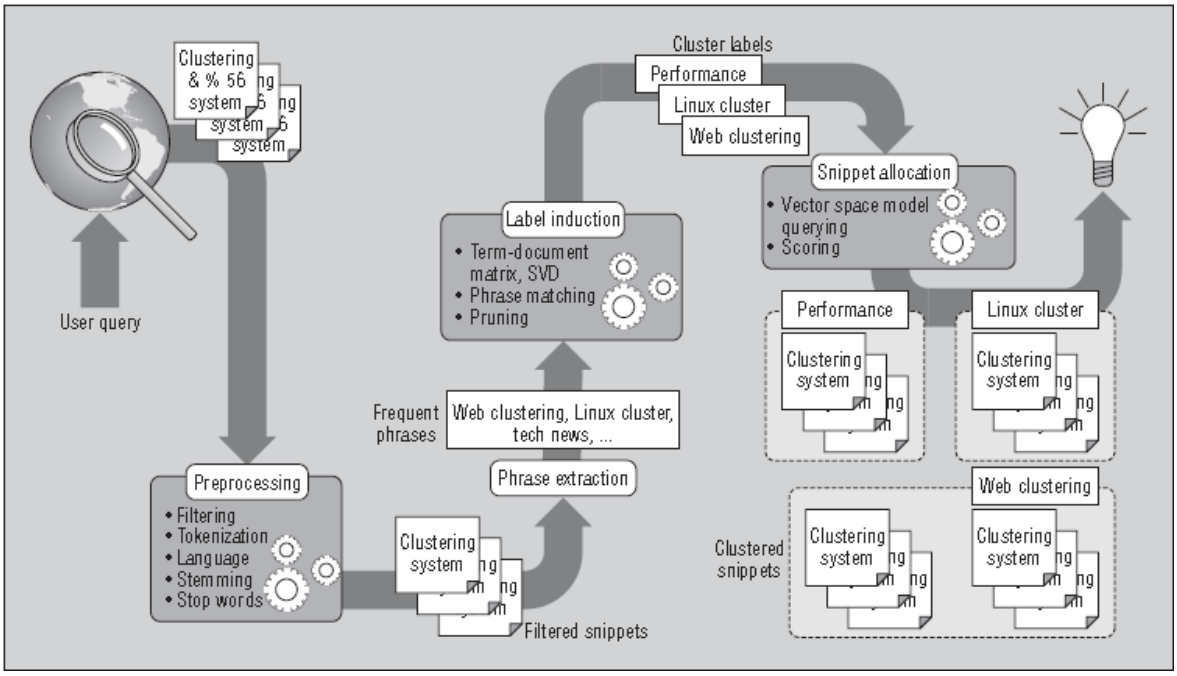
Şekil 3.3 Carrot2 Doküman Kümeleme Tezgahı Programının Örnek Ekran Görüntüsü

3.3. Lingo

Bu bölümde öncelikle orijinal Lingo algoritması [1][40], ardından Carrot2 çatısı kapsamında yayınlanan 3.7.1 sürüm etiketli kütüphane tarafından sağlanan güncel Lingo algoritması anlatılmıştır [59]. Web ASK problemi için kümelenecek örnekler, arama sonuçlarının kısa özetleridir; ancak Lingo genel ASK problemi için de kullanılabildiği için, bu bölümde ve dokümanın ilerleyen bölümlerinde kümelenecek örneklerden kısa özet yerine doküman olarak bahsedilmiştir.

3.3.1. Orijinal Lingo Algoritması

Orijinal Lingo algoritması; Ön işleme, Sık Geçen İfadelerin Belirlenmesi, Kümelerin Etiketlerinin Çıkarılması, Kümelere İlişkili Dokümanların Atanması ve Kümelerin Son Düzenlenmesi olmak üzere 5 ana aşamada incelenebilir. Bu başlıkta, her bir aşama bir alt başlık altında incelenmiştir. Şekil 3.4'de orijinal Lingo algoritmasına genel bakış görülmektedir.



Şekil 3.4 Orijinal Algoritmasına Genel Bakış[2]

3.3.1.1. Ön işleme

Bu aşamada öncelikle her doküman için; html kodları filtrelenir ve dokümanda kullanılan dil belirlenir. Dokümanın dili belirlenirken etkisiz kelime sözlüklerinden faydalanılmaktadır. Hangi dilin etkisiz kelimeleri dokümanda en çok geçiyorsa, ilgili dokümanın dili o dil olarak belirlenmektedir. Dil belirlendikten sonra, dokümanı oluşturan kelimeler üzerinde belirlenen dile göre sıkılama işlemi ve etkisiz kelimeleri işaretleme işlemi gerçekleştirilir. Etkisiz kelimeler ilerleyen aşamalarda oluşturulan terim-doküman matrisine dahil edilmez; ancak kümelerin etiketleri belirlenirken etiket içinde anlam bütünlüğüne katkıları olacağı için etiket ifadelerine dahil edilirler.

3.3.1.2. Sık Geçen İfadelerin Belirlenmesi

Lingo algoritmasının belkemiği olan aşamalardan biridir. İlerleyen aşamalarda belirlenecek kümeler için küme etiket adayları, dokümanlarda belirli bir frekans eşik değerinden daha sık geçen ifadelerden seçilir. Sık geçen ifadeler, sonek dizileri [22] aracılığıyla belirlenmektedir [40]. Bu ifadeler birden fazla kelimeden oluşabileceği gibi, tek kelimelik ifadeler de olabilir. Sık geçen bir ifadenin küme etiket adayı olarak seçilebilmesi için aşağıdaki koşulları yerine getirmesi beklenir:

[2] [43]'den alınmıştır.

- Belirli bir frekans eşik değerinden daha sık geçmelidir.
- Birden fazla cümleye yayılmamalıdır.
- Bir tam ifade [24] olmalıdır. Tam ifadelerin parça ifadelere gore kümeleri daha iyi tarif edebileceği düşünülmektedir (*Senatör Hillary* yerine *Senatör Hillary Rodham Clinton*) [40].
- Herhangi bir etkisiz kelime ile başlamamalı ve bitmemelidir.

Sık geçen ifadelerin küme etiket adayları olarak belirlenmesi yaklaşımının altında, araştırmacıların bir konu hakkında bir metin yazılırken konuyu tanımlayan anahtar ifadelerin okuyucunun dikkatini çekebilmek adına metin içerisinde sık sık tekrar edildiğini düşünmeleri yatmaktadır [1].

3.3.1.3. Kümelerin Etiketlerinin Çıkarılması

Bu aşama, terim-doküman matrisinin oluşturulması, doküman kümesindeki soyut kavramların belirlenmesi (abstract concept discovery), her bir soyut kavram için o kavramla en çok benzeyen etiket adayının küme etiketi olarak eşleştirilmesi ve benzer etiketlerin filtrelenmesi adımlarından oluşur [40].

LSI yaklaşımıyla, girdi doküman kümesinde bulunan soyut kavramlar keşfedilir [1]. SVD metodu aracılığıyla çıkarılan soyut kavramlar, ASK işlemi sonucu olarak ortaya çıkacak kümelerin belirleyicileri olarak düşünülebilir. Sık Geçen İfadelerin Belirlenmesi aşamasında seçilen etiket adaylarının, SVD ayrıştırma işlemi sonucu oluşan soyut kavram vektörlerini tanımlama konusunda başarılı olabileceği düşünüldüğünden [40], her bir soyut kavram bir etiket adayıyla eşleştirilmektedir. Başka bir deyişle, LSI ile keşfedilen her bir kümeye etiket adayları içerisinden bir etiket atanmaktadır.

Bu aşamada ilk olarak, etkisiz kelimeler (stop word) dışındaki ve terim frekans eşik değerinden daha sık geçen kelimelerin sıkılanmış (stemmed) terim karşılıkları, standart *tf-idf* yöntemiyle ağırlıklandırılarak terim-doküman matrisi oluşturulur. Takiben, bu matris üzerinde SVD işlemi uygulanarak kolon vektörlerinin soyut kavram vektörleri olduğu terim-soyut kavram matrisi (U) ve diğer SVD sonuç matrisleri elde edilir. Matris indirgeme işleminin kalitesini, algoritmanın parametrelerinden biri olan aday etiket eşik değeri (candidate label threshold) belirler. Eşik değer ne kadar yüksek seçilirse o kadar çok sayıda soyut kavram, başka bir deyişle o kadar küme ortaya çıkar.

Bir sonraki adımda, etiket adayları için de, ifade vektörleri soyut kavram vektörleri ile aynı terim uzayında yer alacak şekilde, bir terim-ifade matrisi oluşturulur. Terim-soyut kavram matrisi ve terim-ifade matrisinin kolon vektörleri kosinüs benzerliği yaklaşımıyla karşılaştırılarak, her bir soyut kavram vektörü için o kavrama en yüksek skorla (etiket skoru) benzeyen ifade vektörü belirlenir. Bu işlem ile her bir soyut kavramın temsil ettiği kümeye, soyut kavram vektörüyle eşleşen ifade vektörünün temsil ettiği etiket adayı küme etiketi olarak atanır. Bu işlem sonucu oluşan küme etiketleri, benzer bir şekilde kosinüs benzerliği yöntemiyle birbirleriyle karşılaştırılarak, birbirine belli bir etiket benzerlik eşik değerinden (label similarity threshold) daha çok benzeyen etiketlerden yalnızca en yüksek skorlu olan kalacak şekilde filtrelenir. Bu aşama sonucunda algoritmanın oluşturacağı sonuç kümelerin etiketleri belirlenmiş olur.

3.3.1.4. Kümelere İlişkili Dokümanların Atanması

Bu aşamada, bir önceki aşamada etiketleri belirlenmiş kümelere ilişkili dokümanlar atanır. Bu işlem, orijinal terim-doküman matrisi ile terim-ifade matrisinin doküman ve ifade vektörlerinin kosinüs benzerliği yöntemiyle karşılaştırılarak, belli bir doküman atama eşik değerinden (snippet assignment threshold) daha yüksek bir skorla etikete benzeyen dokümanların ilgili etiketin kümesine atanmasıyla gerçekleştirilir. Bu yaklaşımın bir sonucu olarak, bir doküman birden fazla kümeye atanabilir. Başka bir deyişle örtüşen kümeler (overlapping clusters) oluşabilir. Herhangi bir etikete atanamayan dokümanlar, "diğer konular" (other topics) adı verilen yapay bir kümeyle ilişkilendirilir.

Bu aşamayla ilgili önemli bir nokta, anlamsal olarak etiketin temsil ettiği kümeye dahil olması gereken; ancak etiketteki terimleri içermeyen dokümanların ilgili kümelere atanmasının bu yaklaşımda mümkün olmadığıdır. Bu probleme çözüm olarak getirilen yaklaşımlardan biri, kümelere doküman atama işlemi için LSI yaklaşımından faydalanmaktır. Ancak web arama sonucu kümeleme alanında, girdi olarak kullanılan dokümanlar arama sonuçların kısa özetlerinden oluştuğu için LSI yaklaşımının çok başarılı sonuçlar vermediği tecrübe edilerek, Lingo algoritmasında kümelere atanacak dokümanların belirlenmesi için LSI yerine klasik VSM yaklaşımı benimsenmiştir [40].

3.3.1.5. Kümelerin Son Düzenlenmesi

Sonuç kümeler, son kullanıcıya gösterilmeden önce, etiket skoru ve kümeye atanan doküman sayısının çarpımının sonuç değerine göre azalan sırada sıralanır. Bu yaklaşım, daha anlaşılır etiketli ve daha çok doküman içeren kümelerin daha üst sıralarda listelenmesini sağlar.

3.3.2. Güncel Lingo Algoritması

Bu bölümde 3.7.1 sürüm etiketli Carrot2 kütüphanesi kapsamında sağlanan güncel Lingo algoritması antılmaktadır [59]. Güncel algoritma kısaca şu şekilde verilebilir:

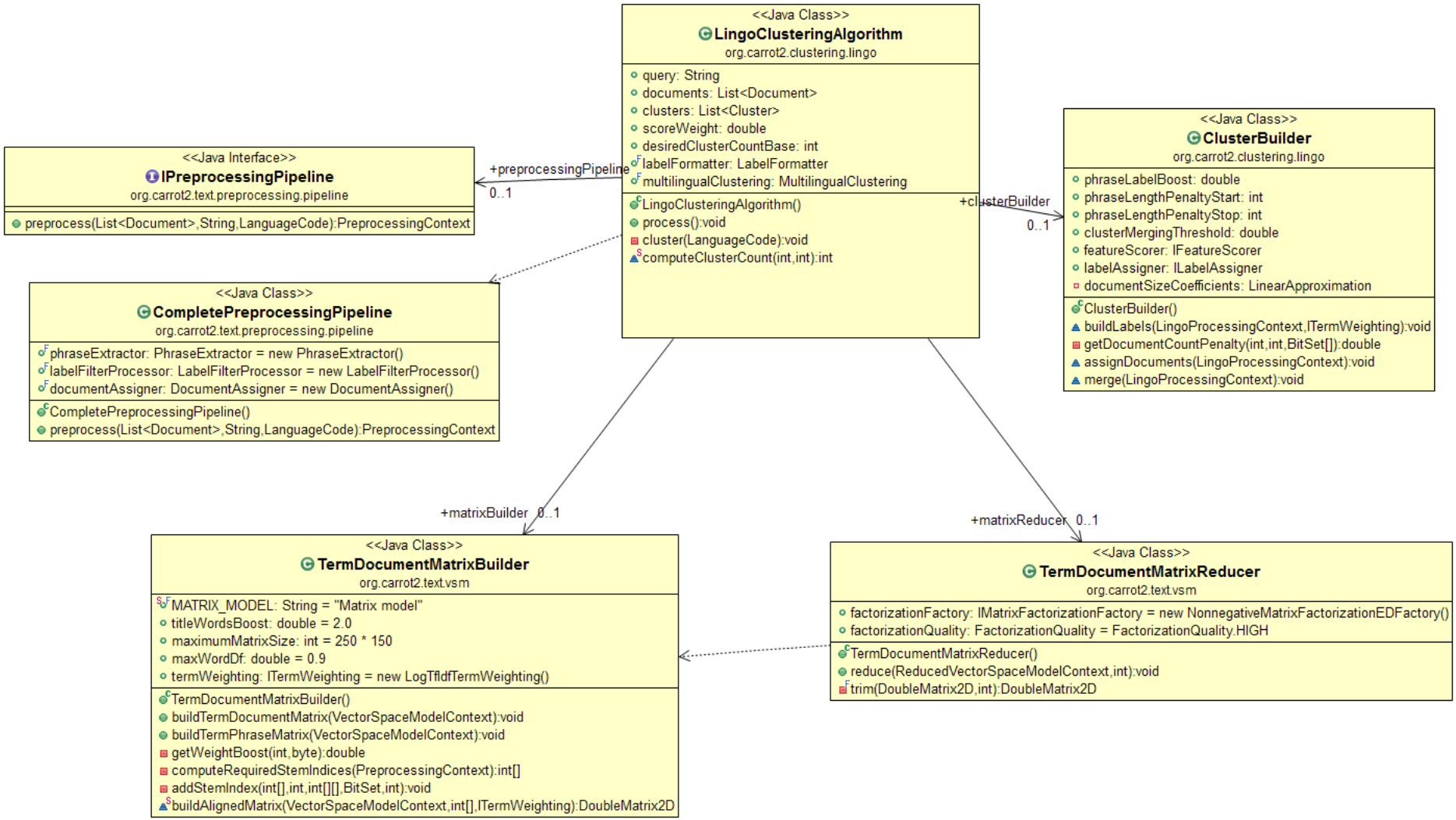
1. Girdi dokümanları ön işleme sürecinden geçir.
    1.1. Frekans eşik değerinden daha sık geçen kelime ve ifadelerden oluşan küme etiket adaylarını seç.
    1.2. Her bir etiket adayı için; etikette geçen etkisiz kelimeler dışındaki terimlerin hepsini birden içeren dokümanları etiket adayının temsil edeceği kümeye ata.
    1.3. Minimum küme elemanı sayısından daha az sayıda doküman atanmış etiket adaylarını ele.
2. Etiket adaylarını oluşturan etkisiz kelimeler dışındaki terimlerin sıkılanmış hallerinden oluşan terimler uzayında, terim-doküman matrisini oluştur.
3. Terim-doküman matrisinin doküman vektörleriyle aynı terimler uzayında yer alacak şekilde, etiket adayları için terim-ifade matrisini oluştur.
4. Terim-doküman matrisini, parametre olarak sağlanan matris çarpanlarına ayırma yöntemini kullanarak; terim-soyut kavram, soyut kavram-soyut kavram ve doküman-soyut kavram matrislerine ayır.
5. Soyut kavram ve ifade vektörlerini kosinüs benzerliği yöntemiyle karşılaştırarak, her bir soyut kavramı ona en yüksek skorla benzeyen ifadeyle eşleştir.
6. Soyut kavramlarla eşleşen etiket adaylarını küme etiketleri olarak belirle.
7. Küme birleştirme eşik değerinden (cluster merging threshold) daha büyük bir oranda ortak doküman içeren kümeleri birleştir.

8. Kümeleri, etiketlerinin eşleştikleri soyut kavramla benzerlik skoruna ve içerdikleri doküman sayısına göre, yüksek skorlu ve çok elemanlı kümeler daha önde olacak şekilde sırala.

Güncel algoritma, orijinal algoritma ile karşılaştırıldığında 3 temel fark göze çarpmaktadır:

1. Küme Etiketlerinin Çıkarılması aşamasında, varsayılan matris çarpanlarına ayırma yöntemi olarak SVD yerine NMF-ED kullanılmaktadır. NMF-ED yöntemi, karşılaştırılan diğer matris çarpanlarına ayırma metotlarına göre en iyi sonuçları verdiği için [45] bu değişiklik gerçekleştirilmiştir.
2. Kümelere doküman atama stratejisi olarak VSM yaklaşımı yerine küme etiketleriyle dokümanlar arasındaki ikili (binary) benzerliğe dayalı bir yaklaşım kullanılmaktadır. Bu yaklaşıma göre, küme etiketlerindeki etkisiz kelimeler dışındaki bütün terimleri içeren dokümanlar kümelere atanmaktadır.
3. Küme birleştirme aşaması eklenmiştir. Küme birleştirme eşik değerinden daha büyük bir oranda ortak doküman içeren kümeler üzerinde küme birleştirme işlemi gerçekleştirilmektedir.

Şekil 3.5'de Güncel Lingo algoritmasının Carrot^2 çatısındaki gerçekleştiriminde kullanılan temel Java sınıflarını içeren sınıf diyagramı verilmiştir.



Şekil 3.5 Güncel Lingo Algoritmasının Gerçekleştirimindeki Temel Java Sınıflarını İçeren Sınıf Diyagramı

3.3.2.1. Güncel Lingo Algoritmasının Parametreleri

Carrot^2 çatısındaki ASK algoritmalarının gerçekleştirimlerinde, algoritmaların işleyişlerinin kullanım ihtiyacına göre özelleştirilebilmesi amacıyla algoritmaların değişken değerleri için parametre ve eşik tanımlamaları yapılmıştır. Bu parametreler Carrot^2 Doküman Kümeleme Tezgahı uygulamasında da her sorgu işlemi bazında dinamik olarak değiştirilebilmektedir. Güncel Lingo algoritmasında kullanılan önemli parametre ve eşik tanımlarından bir kısmı Çizelge 3.1'de verilmiştir.

Güncel Lingo algoritmasında, tek kelimelik veya birden fazla kelimenin dizisinden oluşan ifade etiket adaylarıyla alakalı parametre ve eşik tanımlarının değerlendirilmesiyle ilgili önemli birkaç nokta [60] aşağıda açıklanmıştır.

- Sık geçen ifadeleri saptayan birim, ifadeyi oluşturan kelimelerin farklı çekimlenmiş varyantlarını tek bir ifadede toplar. Örneğin, İngilizce dili ile yazılmış bir metinde 2 defa geçen *computing sciences* ve 4 defa geçen *computer sciences* ifadelerini aynı ifade olarak değerlendirip, yalnızca diğerinden daha sık geçen *computer sciences* ifadesini sık geçen ifade olarak belirler ve tekrar etme sayısını 6 olarak kaydeder.
- Sık geçen kelimeleri saptayan birim, kelimelerin büyük / küçük harfli varyantlarını benzer şekilde tek bir kelimede toplar.
- Aynı şekilde sıkılanan ama farklı çekimlenmiş ik kelime için farklı kelime kayıtları saklanır; ancak terimlerin geçtiği dokümanların tutulduğu vb. diziler kapsamında bu iki kelime için tek bir dizi tutulur.
- Etiket adayları için çeşitli filtreler kullanılabilmektedir. Örneğin; etkisiz etiket (stop label) olarak düzenli ifade tanımı yapılmış şablonlara uyan, belli bir değerden daha az sayıda karakterden oluşan, yalnızca sorgu metninde geçen kelimelerden oluşan ifadeler için tanımlı filtreler kullanılabilmektedir.

Çizelge 3.1 Güncel Lingo Algoritmasındaki Bazı Önemli Parametre ve Eşik Tanımları

Parametre / Eşik Değer	**Açıklama**	**Varsayılan Değer**	**Minimum Değer**	**Maksimum Değer**
Kelime Frekans Eşiği	Bir kelimenin etiket adayı olabilmesi için girdi doküman kümesinde minimum tekrar etme sayısıdır.	1	1	100
İfade Frekans Eşiği	Bir ifadenin etiket adayı olabilmesi için girdi doküman kümesinde minimum tekrar etme sayısıdır.	1	1	100
Minimum Küme Hacmi	Bir etiket adayının içermesi gereken minimum atanmış doküman sayısı.	2	1	100
İfadenin Kendisinin İçerilmesine Göre mi Atama Yapılmalı	Etiket adaylarına doküman ataması yapılırken, dokümanlarda etiket adaylarındaki ifadenin kendisi aynı şekilde mi aranmalı yoksa etiket adayının içerdiği terimler ayrı ayrı aranabilir mi?	Hayır	-	-
Maksimum Matris Hacmi	Terim-doküman matrisinin maksimum hacim değeridir. Terimler uzayında etiket adaylarının terimlerinden olabildiğince fazlası kullanılır.	250 * 150	50 * 100	-
Maksimum Kelime Frekans Eşiği	Kelimelerin terimler uzayında oluşturulan matrislerdeki terim satırlarına dahil edilebilmesi için tüm girdi doküman kümesinde bulunabileceği maksimum doküman oranı.	0,9	0,0	1,0
Başlık (Title) Kelimelerinin Desteklenmesi	Arama sonuç dokümanlarının başlıklarında bulunan terimlerin, terim-doküman matrisindeki ağırlıklandırmalarının kaçla çarpılacağını belirler.	2,0	0,0	10,0

Terim Ağırlıklandırma Yöntemi	Terim-doküman ve terim-ifade matrisleri oluşturulurken kullanılacak terim ağırlıklandırma yöntemidir. Alabileceği değerler: Lineer tf-idf, Log tf-idf, tf.	Log Tf-Idf	-	-
Küme Sayısı Tabanı	Oluşturulacak küme sayısının hesaplanmasında kullanılan taban değerdir.	30	2	100
Matris Çarpanlarına Ayırma Yöntemi	Kullanılacak matris çarpanlarına ayırma yöntemi seçimidir. Alabileceği değerler: K-Means, LNMF, NMF-ED, NMF-KL ve Partial SVD.	NMF-ED	-	-
Matris Çarpanlarına Ayırma Kalitesi	Kullanılan matris çarpanlarına ayırma yönteminin kalitesi belirlenir. Alabileceği değerler: Düşük, Orta, Yüksek	Yüksek	-	-
İfade Etiketlerin Desteklenmesi	İfade etiket adaylarının, tek kelimelik adaylara göre desteklenme katsayısıdır.	1,5	0,0	10,0
Küme Birleştirme Eşiği	İki kümenin birleştirilmesi için ortaklaşa içermeleri gereken doküman oranı.	0,7	0,0	1,0

4. YÖNTEM

Bu bölümde Lingo algoritmasında düşük anma değerine sebep olan eksiklikleri gidermek için önerilen yöntemler anlatılmaktadır. Bu tez kapsamında biri Kümelere İlişkili Dokümanların Atanması, diğeri ise Kümelerin Etiketlerinin Belirlenmesi aşamasında olmak üzere iki değişiklik önerisi sunulmuştur.

4.1. Kümelere İlişkili Dokümanların Atanması Aşamasındaki Değişiklik

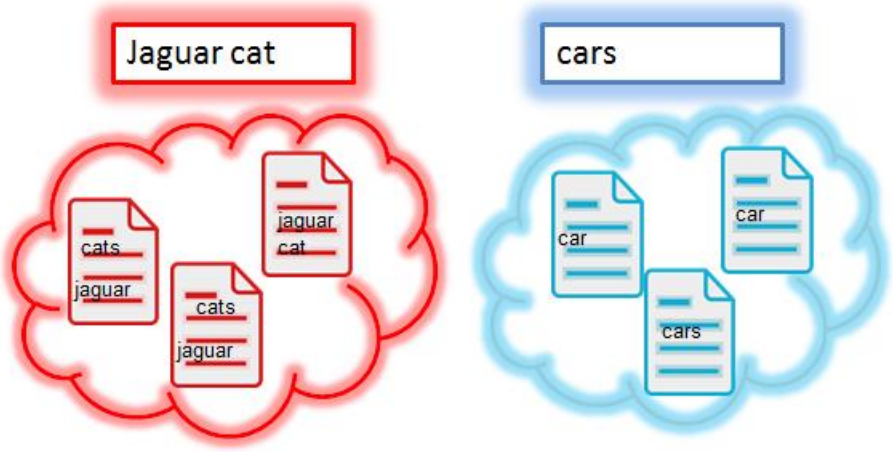
Güncel Lingo algoritmasında kümelere ilişkili dokümanların ataması işlemi etiket adayları tespit edildikten sonra yapılmaktadır. Her etiket adayının temsil ettiği küme adaylarına, etiketlerdeki etkisiz kelimeler dışındaki bütün kelimelerin sıkılanmış hallerini (stem) içeren dokümanlar atanmaktadır. Bu yaklaşım, kümelerdeki doküman sayısının kontrol edilebilirliği açısından orijinal Lingo algoritmasında kullanılan klasik VSM yaklaşımına göre daha az esneklik sağlıyor olsa da, aynı zamanda duyarlılık açısından daha başarılı sonuçlar verdiği için tercih edilmiştir. Bu yaklaşımda karşılaşılabilecek iki temel sorun vardır:

1. Etiketlerde geçen kelimeleri içerse de aslında anlamsal olarak etiketlerle ilişkili olmayan dokümanlar kümelere atanabilir.
2. Etiketlerde geçen kelimeleri içermemesine rağmen aslında anlamsal olarak etiketlerle ilişkili olan dokümanlar kümelere atanamaz.

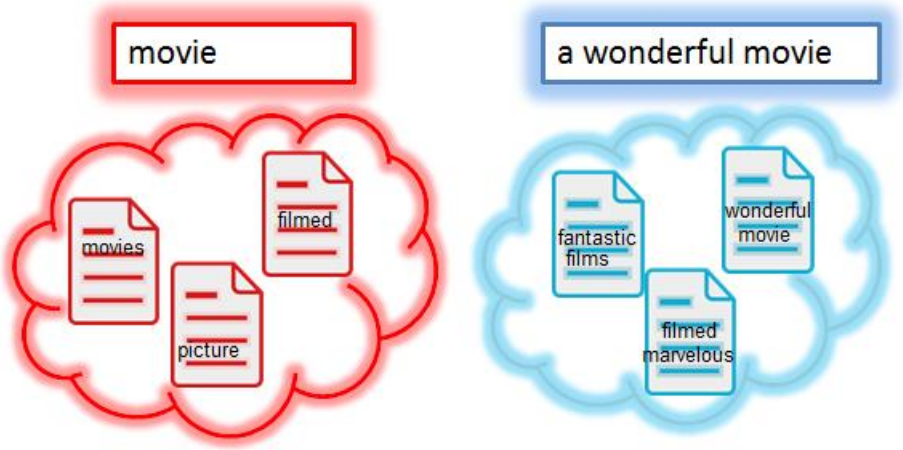
Bu problemlerden ilki duyarlılık değerlerinde, ikincisi ise anma değerlerinde düşüşe sebep olmaktadır. Algoritmanın bu aşaması için 2. sorunun çözülebilmesine yönelik bir öneri sunulmaktadır. Bu bağlamda, etiket adayları için doküman ataması yapılırken, etiketlerdeki kelimelerin WordNet [3] veritabanı aracılığıyla erişilebilecek eş anlamlılarından da faydanılması önerilmektedir. Önerilen yönteme göre, etiketteki her bir kelime için; ya kelimenin kendisinin ya da kelimenin bir eş anlamlısının sıkılanmış halini içeren dokümanlar ilgili etiket adayının temsil ettiği aday kümeye atanabilecektir. WordNet veritabanından gerekli sorgulamayı yapabilmek için JAWS [61] kütüphanesi kullanılmıştır.

Güncel atama stratejisi ve önerilen atama stratejisi sırasıyla Şekil 4.1 ve Şekil 4.2'de örneklenmiştir. Şekillerde bir arama sorgusu için elde edilen "Jaguar cat", "cars", "movie" ve "a wonderful movie" etiketlerine sahip aday kümeler örnek elemanlarıyla gösterilmektedir. Küme elemanı olan dokümanlarda (arama sonuçları); Şekil 4.1 için küme etiketlerindeki kelimelerin köklerini (stem) içeren

kelimeler, Şekil 4.2 için küme etiketlerindeki kelimelerin kendilerinin ya da eş anlamlılarının köklerini (stem) içeren kelimeler doküman sembollerinin üzerlerinde gösterilmiştir.

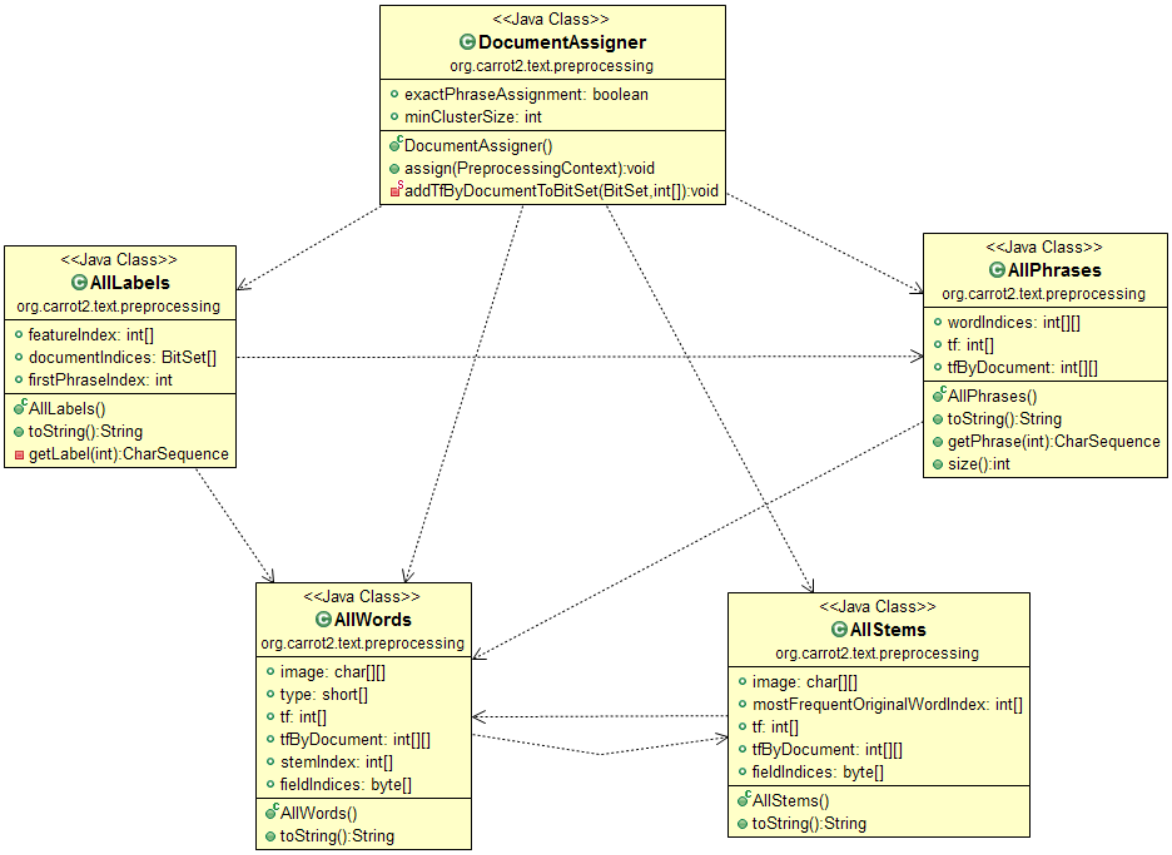


Şekil 4.1 Güncel Atama Stratejisi Örneği

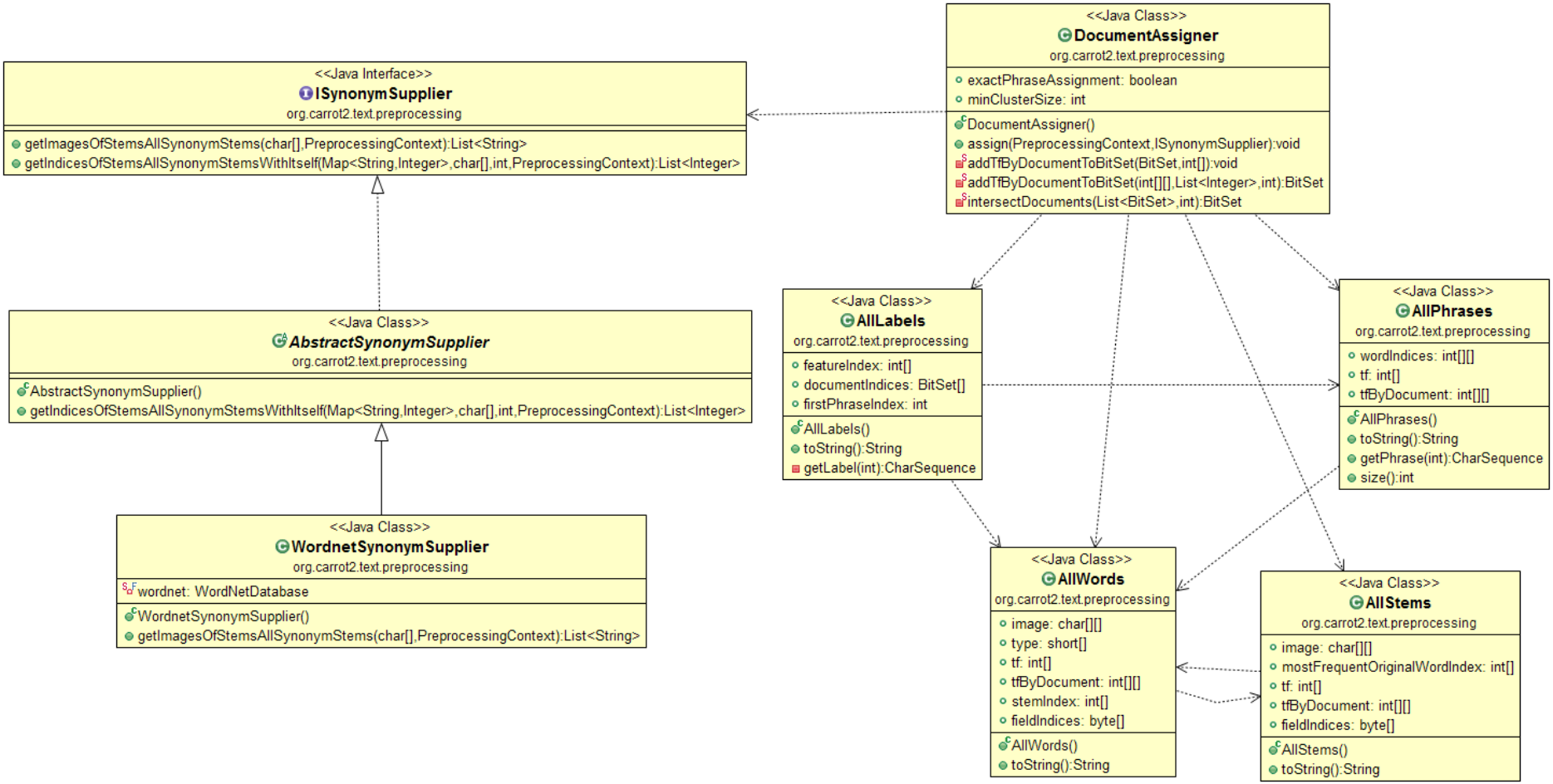


Şekil 4.2 Önerilen Atama Stratejisi Örneği

Güncel Lingo algoritmasının gerçekleştirimin olduğu Carrot[2] çatısındaki, önerilen değişikliğin öncesi ve sonrasındaki ilişkili sınıfların gösterildiği sınıflardan oluşan sınıf diyagramları sırasıyla Şekil 4.3 ve Şekil 4.4'de gösterilmiştir.



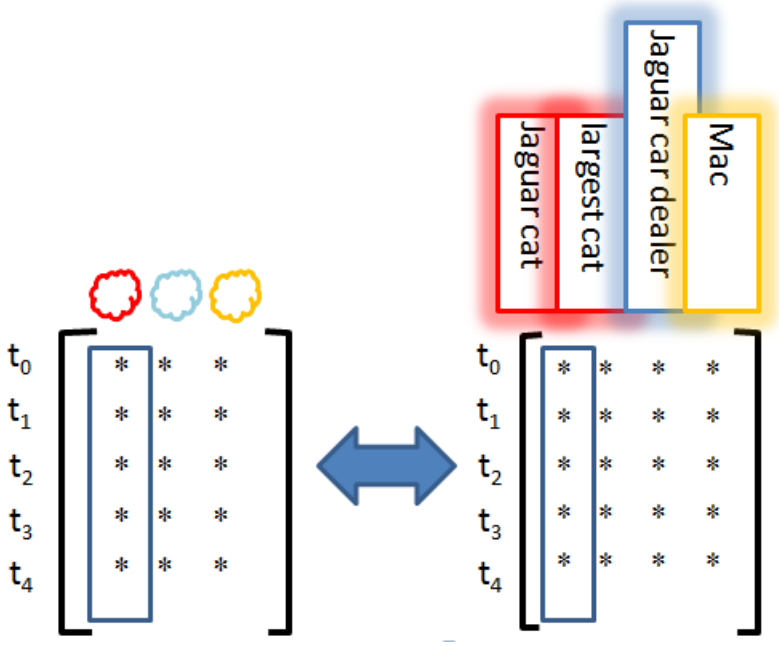
Şekil 4.3 Önerilen Doküman Atama Stratejisi Öncesi Sınıf Diyagramı



Şekil 4.4 Önerilen Doküman Atama Stratejisi Sonrası Sınıf Diyagramı

4.2. Kümelerin Etiketlerinin Çıkarılması Adımındaki Değişiklik

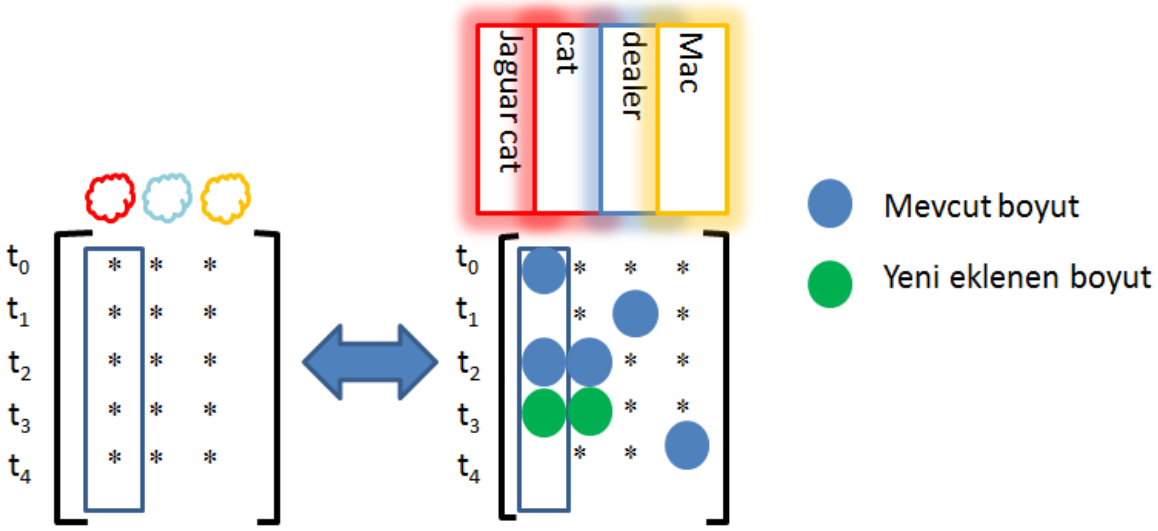
Kümelerin Etiketlerinin Belirlenmesi aşamasında terim-soyut kavram matrisi ve terim-ifade matrisleriyle ifade edilen soyut kavram ve ifade vektörleri kosinüs benzerliği yöntemiyle eşleştirilmektedir. Bu eşleştirme Şekil 4.5'de görsel olarak örneklenmiştir. Şekilde soyut kavramlar bulut şekilleriyle, etiket adayı olan ifadeler ise dikdörtgenlerle gösterilmektedir.



Şekil 4.5 Güncel Soyut Kavram-Etiket Adayı Eşleştirme Stratejisi

İfadeler genelde çok fazla terim içermediği için ifade vektörleri, soyut kavram vektörlerine göre daha az terim boyutuna yayılabilmektedir. Başka bir deyişle terim-soyut kavram matrisi daha sık bir matrisken, terim-ifade matrisi daha seyrek bir matristir. Bu çerçevede, bu karşılaştırma yaklaşımıyla soyut kavramlar ve etiket adaylarını eşleştirmenin sağlıksız sonuçlar verebileceği düşünüldüğünden, soyut kavram-etiket adayı eşleştirmesi için iki ayrı değişiklik önerisi sunulmuştur.

İlk olarak, soyut kavram vektörleriyle karşılaştırılan ifade vektörlerinin, ifadedeki etkisiz kelimeler dışındaki kelimelerin eş anlamlı terimleri kullanılarak zenginleştirilmesi önerilmektedir. Zenginleştirme işlemi, etiketlerde geçen kelimelerin eş anlamlılarından yalnızca terimler uzayında yer alan kelimeler için yapılabilecektir. Bu öneri Şekil 4.6'da görsel olarak örneklenmiştir.

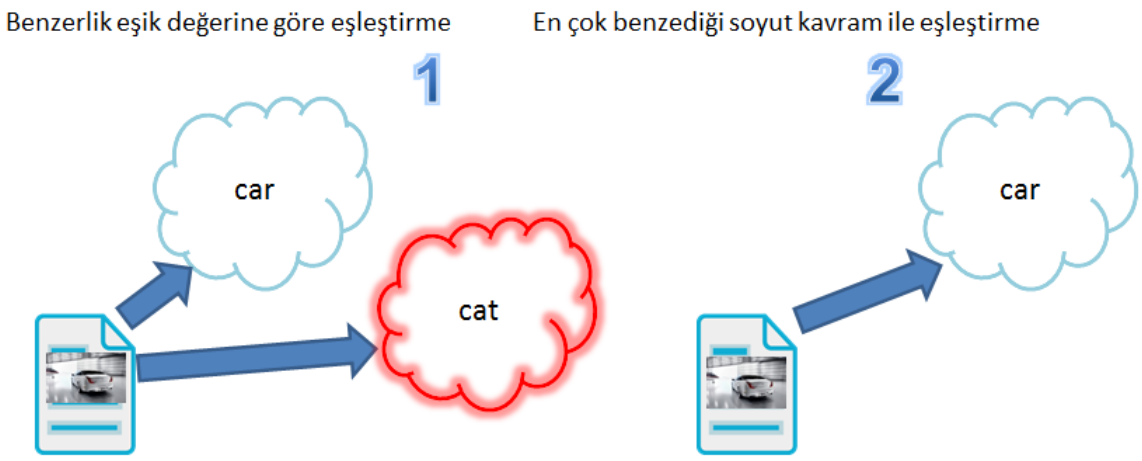


Şekil 4.6 Soyut Kavram-Etiket Adayı Eşleştirmesi İçin İlk Öneri

İkinci olarak LSI yaklaşımından da faydalanılarak yeni bir eşleştirme yöntemi önerilmektedir. Bu yöntemde, soyut kavramlarla etiket adayları eşleştirilirken, benzerlik ölçütü olarak, ilişkili oldukları ortak doküman sayısının kullanılması önerilmektedir. Bu yöntem, ön koşul olarak soyut kavramlarla ilişkili olan dokümanların saptanmasını gerektirmektedir. Bu ilişkilendirme terim-doküman matrisinin matris çarpanlarına ayrılması işleminin bir sonucu olarak ortaya çıkmaktadır.

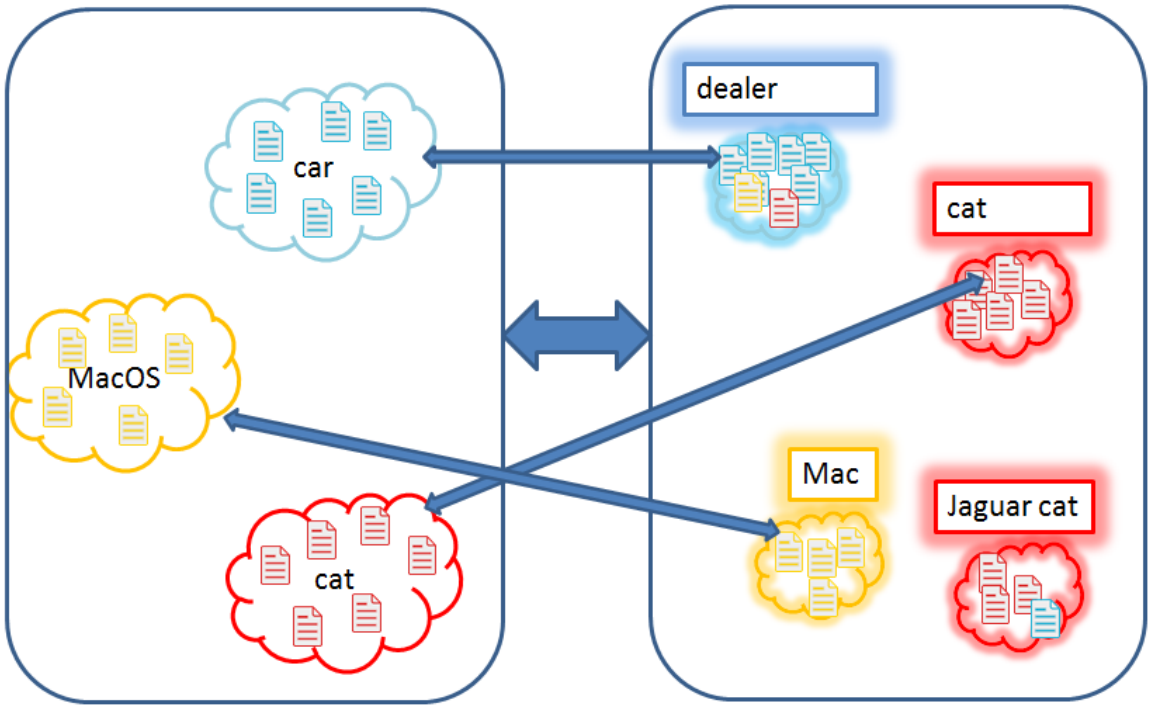
Matris çarpanlarına ayırma işlemine tabi tutulan terim-doküman matrisi; terim-soyut kavram matrisi, soyut kavram-soyut kavram ve soyut kavram-doküman matrislerine ayrıştırılmaktadır. Soyut kavram-doküman matrisinde doküman vektörleri terimler yerine soyut kavramlarla ifade edilmektedir. Matrisin her bir hücresinde dokümanlarla soyut kavramların ilişkililik dereceleri yer alır. Bu bilgiden faydalanılarak her bir soyut kavramla ilişkili olan doküman ya da dokümanlar belirlenebilir. Bu aşamada her bir soyut kavramla ilişkili olan dokümanların belirlenmesinde iki farklı yöntem önerilmiştir:

1. Her bir dokümanı belli bir benzerlik eşik değerinden daha yüksek derecede ilişkili olduğu soyut kavramlarla ilişkilendir.
2. Her bir dokümanı yalnızca en çok ilişkili olduğu soyut kavramla ilişkilendir.



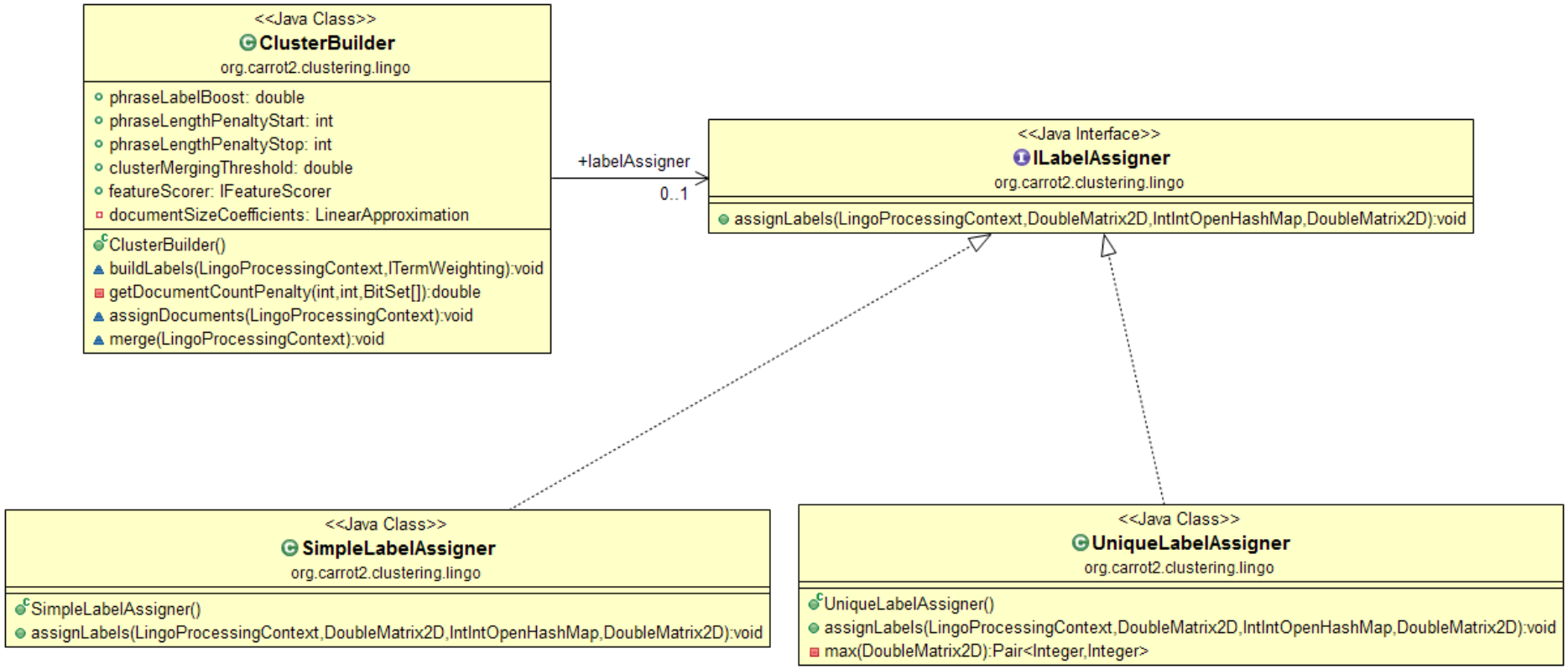
Şekil 4.7 Dokümanların Soyut Kavramlarla İlişkilendirilmesi İçin Önerilen Yöntemler

Şekil 4.7'de, yukarıda anlatılan iki yöntem görsel olarak örneklenmiştir. Bu yöntemlerden biri kullanılarak, dolaylı yoldan her bir soyut kavram için soyut kavramların ilişkili oldukları dokümanlar belirlendikten sonra, her bir etiket adayına atanan dokümanlar da hâlihazırda belli olduğu için, soyut kavramlar ve etiket adayları ortaklaşa içerdikleri doküman sayısı değeri benzerlik ölçütü olarak kullanılacak şekilde karşılaştırılıp eşleştirilebilir. Şekil 4.8'de, soyut kavramlar ve etiket adaylarının bu öneriye göre örnek bir eşleştirmesi verilmiştir. Şekilde sol sütundaki bulutlar soyut kavramları, sağ sütundaki bulutlar ise üzerlerindeki dikdörtgenlerde verilen ifadelerle etiketlenmiş aday kümeleri temsil etmektedir. Sağ sütunda yer alan ve bir soyut kavramla eşleşen aday kümeler ("dealer", "cat" ve "Mac"), kümeleme işleminin sonucu olarak ortaya çıkan sonuç kümeleri oluşturmaktadır.

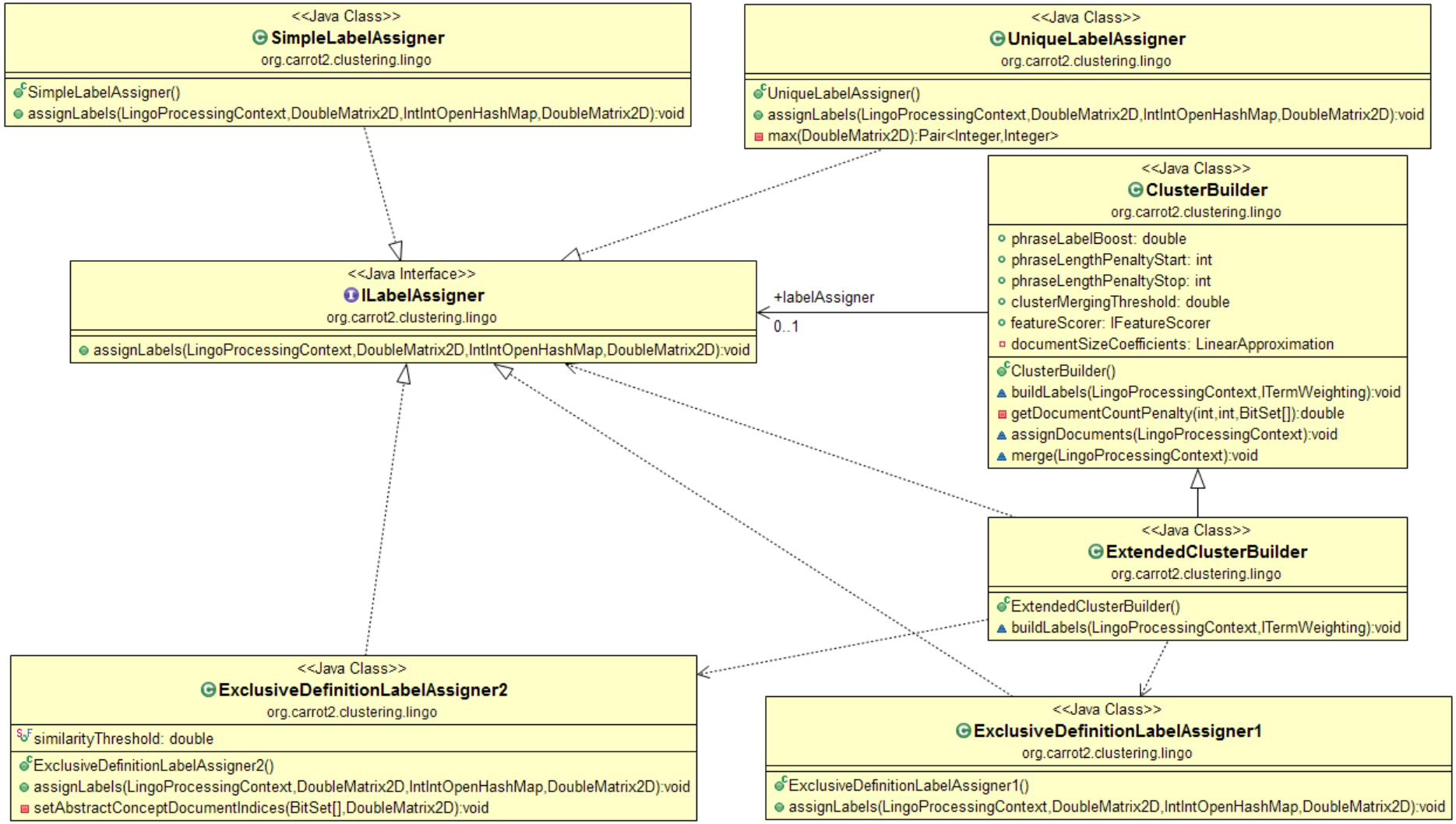


Şekil 4.8 Soyut Kavram-Etiket Adayı Eşleştirmesi İçin İkinci Öneri - Örnek Eşleştirme

Güncel Lingo gerçekleştiriminde önerilen değişikliğin uygulanmasının öncesi ve sonrasındaki ilişkili sınıfların gösterildiği sınıflardan oluşan sınıf diyagramları sırasıyla Şekil 4.9 ve Şekil 4.10'de gösterilmiştir.



Şekil 4.9 Önerilen Soyut Kavram - Etiket Adayı Eşleştirme Stratejisi Öncesi Sınıf Diyagramı



Şekil 4.10 Önerilen Soyut Kavram - Etiket Adayı Eşleştirme Stratejisi Sonrası Sınıf Diyagramı

5. DENEY

Tez çalışması kapsamında güncel Lingo algoritmasının eksikliklerinin giderilmesi için 4. bölümde detaylı olarak anlatılan öneriler, Carrot^2 kütüphanesi kapsamında sağlanan Java API'sinin kaynak kodları [59] üzerinde gerçekleştirilmiştir.

5.1. Kullanılan Veri Seti

Deneylerde, Carrot2 çatısında sorgulama ve sorgu sonuçlarının Lingo algoritmasına girdi olarak verilmesi için gerekli altyapısı sağlanan, AMBIENT [56] veriseti kullanılmıştır.

AMBIENT veriseti Wikipedia'nın [37] belirsizlik giderme (disambiguation) sayfalarından [62] seçilen toplam 44 adet belirsiz (ambiguous) başlıkla ilgili alt başlıklar ve her bir alt başlık için ilişkili arama sonuçlarını içerir. Wikipedia'da belirsiz başlıkla ilgili bir belirsizlik giderme sayfasında, ilgili başlığı belirsiz yapan bütün alternatif kullanımları alt başlıklarla listelenir. AMBIENT veriseti için seçilmiş 44 belirsiz başlığın her biri için, başlıkların belirsizlik giderme sayfalarındaki tüm alternatif alt başlıkları, verisetine başlıkların alt başlıkları olarak dahil edilmiştir. Ayrıca, her bir başlık için, başlığın sorgu metni olarak kullanılmasıyla 2008 yılının Ocak ayı içerisinde bir arama motorundan alınan ilk 100'er arama sonucu verisetinde toplanmıştır. Bu arama sonuçları her bir başlık için, başlığın alt başlıklarından hangisiyle ilişkili ise o alt başlıkla ilişkilendirilmiştir [56]. Bu yaklaşımın sonucu olarak verisetinde; her bir başlık için kimi alt başlıklarla ilişkili hiçbir arama sonucu bulunamayabilirken, bazı arama sonuçları da hiçbir alt başlıkla ilişkilendirilmemiş olabilmektedir.

Verisetinde arama sonuçları yalnızca; URL, başlık ve sonuç sayfaların kısa özetlerinden (snippet) oluşmaktadır. Güncel ve önerilen Lingo algoritmasının ikisinde de, kümeleme için yalnızca bu bilgiler kullanılmaktadır. Tez genelinde arama sonuçlarından doküman veya arama sonucu olarak bahsedilmekte, her iki ifade ile de yalnızca bu bilgilerden oluşan arama sonuçları kastedilmektedir.

Verisetiyle yapılan deneylerde; her bir başlık bir arama sorgusu olarak kullanılmış ve her bir başlığın en az bir arama sonucuyla ilişkilendirilen alt başlıkları ise ilişkili oldukları arama sonuçlarıyla birlikte, deneylerde algoritmanın oluşturması beklenen doğru kümeler olarak baz alınmıştır.

AMBIENT veriseti, ASK problemi için oldukça uygun bir verisetidir. Sorgu metni olarak kullanılacak her bir başlık için elde edilen arama sonuçlarının, doğru bir şekilde beklenen alt başlıklara ayrılıp ayrılamadığının sınanabilmesine olanak vermektedir.

5.2. Kullanılan Metrikler

Deney sonuçlarını, beklenen sonuçlarla kıyaslamak için 5 farklı metrik kullanılmıştır. Bunlardan 3 tanesi klasik bilgi erişim metriklerinden olan duyarlılık (precision), anma (recall) ve f-ölçütü (f-measure) metrikleri, diğerleri ise; küme kirliliği (cluster contamination) ve normalleştirilmiş ortak bilgi (**n**ormalized **m**utual **i**nformation) metrikleridir.

Verisetindeki her bir başlık için; duyarlılık, anma ve f-ölçütü değerlerinin ağırlıklı ortalamaları hesaplanmış, veriseti genelindeki sonuçlar ise, bu değerlerin tüm başlıklar kapsamında aritmetik ortalama değerleri alınarak elde edilmiştir.

Duyarlılık, anma ve f-ölçütü metriklerinin tanımı, sonuçlardaki Çizelge 5.1'de gösterilen Doğru Pozitif, Yanlış Pozitif ve Yanlış Negatif durumundaki sonuç sayılarına göre hesaplanır.

Çizelge 5.1 Duyarlılık, Anma ve F-ölçütü

	Doğru Kümede	**Doğru Kümede Değil**
Kümede	Doğru Pozitif	Yanlış Pozitif
Kümede Değil	Yanlış Negatif	Doğru Negatif

K_i, K kümesinde ve DK_i de DK doğru kümeler kümesinde bir küme iken DK_i, K_i kümesiyle eşleşen doğru küme olmak üzere:

- Doğru Pozitif, Yanlış Pozitif ve Yanlış Negatif; (5.1), (5.2) ve (5.3) ile verilen denklemlerle tanımlanabilir.

  - $\text{Doğru Pozitif} = K_i \cap DK_i$ (5.1)
  - $\text{Yanlış Pozitif} = K_i \setminus DK_i$ (5.2)
  - $\text{Yanlış Negatif} = DK_i \setminus K_i$ (5.3)

- Duyarlılık, Anma ve F-Ölçütü; (5.4), (5.5) ve (5.6) ile verilen denklemlerle tanımlanabilir.

$$\text{Duyarlılık} = \frac{|\text{Doğru Pozitif}|}{|\text{Doğru Pozitif} \cup \text{Yanlış Pozitif}|} \quad (5.4)$$

$$\text{Anma} = \frac{|\text{Doğru Pozitif}|}{|\text{Doğru Pozitif} \cup \text{Yanlış Negatif}|} \quad (5.5)$$

$$\text{F} - \text{ölçütü} = 2 \cdot \frac{(\text{duyarlılık} \cdot \text{anma})}{(\text{duyarlılık} + \text{anma})} \quad (5.6)$$

- Ağırlıklı Ortalama Duyarlılık, Ağırlıklı Ortalama Anma ve Ağırlıklı Ortalama F-Ölçütü ise; (5.7), (5.8) ve (5.9) ile verilen denklemlerle tanımlanabilir.

$$\text{Ağırlıklı Ort. Duyarlılık} = \frac{\sum_{K \in DK}(\text{Duyarlılık}_K \cdot |K|)}{\sum_{K \in DK}|K|} \quad (5.7)$$

$$\text{Ağırlıklı Ort. Anma} = \frac{\sum_{K \in DK}(\text{Anma}_K \cdot |K|)}{\sum_{K \in DK}|K|} \quad (5.8)$$

$$\text{Ağırlıklı Ort. F} - \text{ölçütü} = \frac{\sum_{K \in DK}(\text{F}-\text{ölçütü}_K \cdot |K|)}{\sum_{K \in DK}|K|} \quad (5.9)$$

Duyarlılık, anma ve f-ölçütü değerleri; (5.7), (5.8) ve (5.9) ile verilen ağırlıklı ortalama denklemlerinden de anlaşıldığı gibi, DK'daki her bir doğru küme için hesaplanır. Her bir DK_i doğru kümesi için, bu kümeyle en iyi f-ölçütü değerini veren $K_{\text{en iyi}}$ kümesi seçilir ve DK_i için metriklerin değerleri $K_{\text{en iyi}}$ kümesindeki sonuçlara göre hesaplanır.

Küme kirliliği metriği, oluşturulan kümelerin saflığını değerlendirmek için kullanılan bir metriktir. Bir küme, yalnızca bir doğru kümenin elemanlarından oluşuyorsa tamamen saf bir küme olarak düşünülür. Küme kirliliği değerinin artması kümelerin daha çok kirlendiğini gösterirken, değerinin 0 olması mutlak saflıkta kümeler oluşturulduğunu gösterir. Bir kümeleme işlemi sonucunda üretilen her bir küme için bir küme kirliliği değeri hesaplanır ve tüm kümeler birlikte değerlendirilerek, kümelerin eleman sayılarına göre küme kirliliğinin ağırlıklı ortalama değeri hesaplanır. Küme kirliliği metriğinin değerleri Carrot[2] çatısında [44]'de anlatıldığı şekilde hesaplanmaktadır.

Üretilen kümelerin kalitesinin değerlendirilmesi için kullanılan son metrik normalleştirilmiş ortak bilgi metriğidir. Değerinin 1 olduğu durum, üretilen kümelerin doğru kümelerle tamamen eşleştiği, kümeleme işleminin mükemmel

olarak nitelendirilebileceği durumdur. Değerleri Carrot[2] çatısında [63]'de tarif edilen şekilde hesaplanmaktadır.

Küme kirliliği ve normalleştirilmiş ortak bilgi metriklerinin veriseti genelindeki değerleri de; duyarlılık, anma ve f-ölçütü metriklerindeki gibi, bu değerlerin tüm başlıklar kapsamında aritmetik ortalama değerleri alınarak elde edilmiştir.

5.3. Yapılan Deney

4. bölümde açıklanan önerilerin güncel Lingo algoritması üzerindeki etkilerini gözlemlemek için, birbirini takip eden 8 adımlık bir deney gerçekleştirilmiştir. Her adımda, güncel algoritma üzerine önerilen değişikliklerin farklı kombinasyonları sınanmıştır. Adım 1'den Adım 8'e kadar numaralandırılmış olan adımlar, açıklamalarıyla birlikte Çizelge 5.2'de verilmiştir. Tüm adımlarda, algoritmanın mevcut parametre ve eşik değer tanımları için güncel gerçekleştirimde tanımlı olan varsayılan değerler kullanılmıştır.

Çizelge 5.2 Sıralı Deney Adımlarının Tanımları

Adım	Tanım
Adım 1	Güncel Lingo algoritması gerçekleştirimini (v3.7.1) uygula.
Adım 2	Kümelere İlişkili Dokümanların Atanması aşaması için önerilen değişikliği; hem tek kelimelik etiket adayları hem de ifade etiket adayları için uygula.
Adım 3	Kümelere İlişkili Dokümanların Atanması aşaması için önerilen değişikliği yalnızca ifade etiket adayları için uygula.
Adım 4	Adım 2'de anlatılanları ve ilaveten Küme Etiketlerinin Çıkarılması aşaması için önerilen ifade vektörlerinin zenginleştirilmesi önerisini uygula.
Adım 5	Küme Etiketlerinin Çıkarılması aşaması için önerilen ortak doküman sayısı benzerliğine dayalı soyut kavram-etiket adayı eşleştirmesi önerisini, benzerlik eşik değerini 0,75 alarak, 1. seçenekle uygula.
Adım 6	Küme Etiketlerinin Çıkarılması aşaması için önerilen ortak doküman sayısı benzerliğine dayalı soyut kavram-etiket adayı eşleştirmesi önerisini 2. seçenekle uygula.
Adım 7	Hem Adım 3, hem de Adım 5'de önerilen değişiklikleri birlikte uygula.
Adım 8	Hem Adım 3, hem de Adım 6'da önerilen değişiklikleri birlikte uygula.

5.4. Deney Sonuçları

Deneyin her bir adımı için, her bir metriğin tüm başlıklar için elde edilen ortalama sonuç değerleri Çizelge 5.3'de verilmiştir. Şekil 5.1 ve Şekil 5.2'de ise aynı değerlerin grafiksel gösterimleri sunulmuştur. Bu değerler, 5.2 bölümünde anlatılan şekilde hesaplanmıştır.

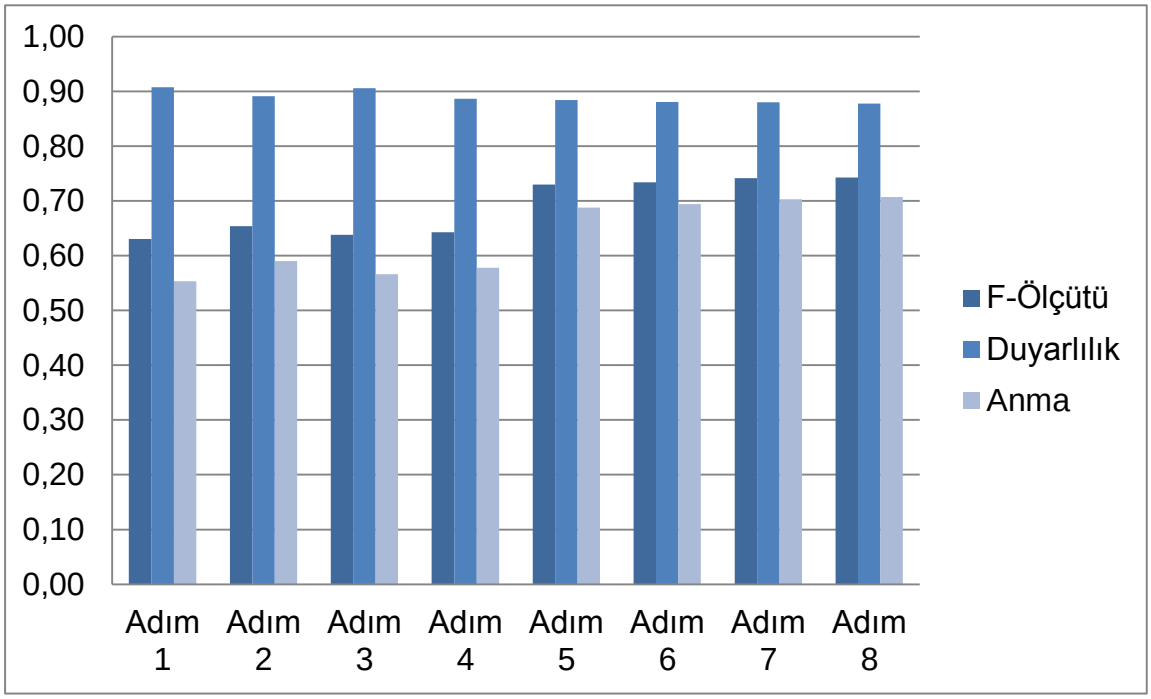
Çizelge 5.3 Bütün Başlıklar için Ortalama Metrik Değerleri

Adım	Küme Kirliliği	F-Ölçütü	Duyarlılık	Anma	Normalleştirilmiş Ortak Bilgi
Adım 1	0,260	0,630	0,908	0,554	0,651
Adım 2	0,338	0,654	0,891	0,590	0,646
Adım 3	0,281	0,638	0,906	0,566	0,646
Adım 4	0,335	0,643	0,886	0,578	0,643
Adım 5	0,291	0,730	0,884	0,688	0,691
Adım 6	0,296	0,734	0,880	0,694	0,693
Adım 7	0,298	0,741	0,880	0,703	0,692
Adım 8	0,294	0,743	0,878	0,707	0,691

Çizelge 5.3'deki değerler incelendiğinde, Adım 2'de Adım 1'e göre daha yüksek f-ölçütü ve anma değerleri elde edilirken; buna karşın duyarlılık değerinin görece daha az bir oranla düştüğü görülebilir. Bu sonuç, Kümelere İlişkili Dokümanların Atanması aşaması için önerilen atama sürecinde eş anlamlılardan faydalanma yaklaşımın, kümelere kümelerle ilişkili daha çok doküman atanması konusunda başarılı olduğunu; ancak bir yan etki olarak, görece daha az bir oranla da olsa, duyarlılık değerini düşürdüğünü göstermektedir. Adım 3'te, Adım 2'den farklı olarak uygulanan, hem tek kelimelik hem de ifade etiket adayları yerine yalnızca ifade etiket adayları için eş anlamlılardan faydalanma yaklaşımının, Adım 2'ye göre hem f-ölçütü ve anma değerlerini daha az iyileştirdiği, hem de bunun birlikte duyarlılık değerini daha az düşürdüğü gözlenebilir. Adım 2 ve Adım 3 sonuçları küme kirliği açısından karşılaştırıldığında da, Adım 3'ün Adım 2'ye göre daha iyi bir performans verdiği görülebilir. Bu bilgiler doğrultusunda, Kümelere İlişkili

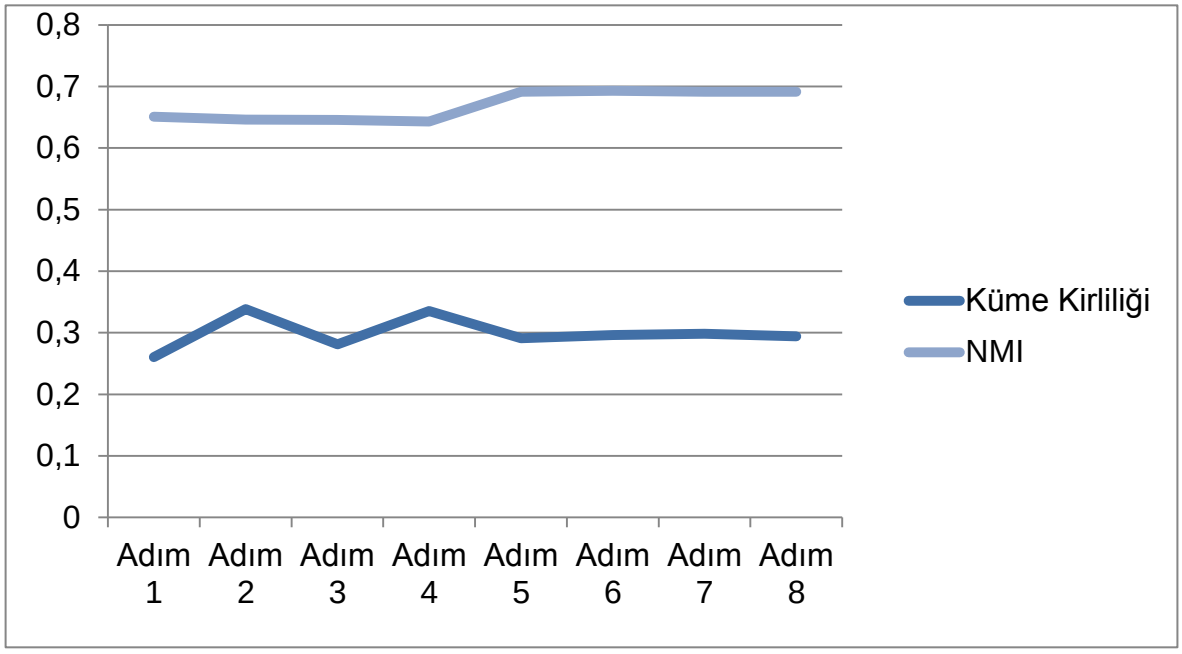
Dokümanların Atanması aşaması için önerilen değişikliğin, yalnızca ifade etiket adayları için uygulanmasının daha iyi sonuçlar verdiği söylenebilir.

Çizelge 5.3'de ayrıca, Adım 4'te uygulanan değişikliğin Adım 2 ile karşılaştırıldığında, yalnızca küme kirliliği açısından daha iyi değerler verdiği, diğer metrikler açısından bir iyileştirme sağlayamadığı görülebilir. Bu sonuca göre, Küme Etiketlerinin Çıkarılması aşaması için önerilen ifade vektörlerinin zenginleştirilmesi yaklaşımın, soyut kavram-etiket adayı eşleştirmesini iyileştirmediği söylenebilir.



Şekil 5.1 Bütün Başlıklar için Ortalama F-Ölçütü, Duyarlılık ve Anma Grafiği

Şekil 5.1 incelendiğinde; Adım 5, 6, 7 ve 8 için f-ölçütü ve anma değerlerinin diğer adımlara göre çarpıcı bir oranda daha yüksek olduğu görülebilir. Bu gözleme göre, Küme Etiketlerinin Çıkarılması aşaması için önerilen yeni soyut kavram-etiket adayı eşleştirmesi yaklaşımının, mevcut yaklaşıma göre ve ifade vektörlerinin zenginleştirilmesi yaklaşımına göre daha başarılı sonuçlar verdiği söylenebilir. Ayrıca, şekil dikkatlice incelendiğinde, Adım 6'da Adım 5'e göre az da olsa daha iyi f-ölçütü ve anma değerlerinin elde edildiği, dolayısıyla yeni soyut kavram-etiket adayı eşleştirmesi önerisininin 2. seçenek ile uygulandığında daha başarılı olduğu söylenebilir. Şekilden çıkarılabilecek başka bir sonuç, Adım 7'nin Adım 5'e göre ve Adım 8'in de Adım 6'ya göre daha başarılı sonuçlar verdiğidir. Bu sonuç, önerilen yeni soyut kavram-etiket adayı eşleştirmesi yönteminin, Kümelere İlişkili Dokümanların Atanması aşaması için önerilen ifade etiket adaylarının eş anlamlılarından faydalanılması yaklaşımıyla birlikte kullanıldığında daha da başarılı sonuçlar verdiğini göstermektedir.



Şekil 5.2 Bütün Başlıklar İçin Ortalama Küme Kirliliği ve Normalleştirilmiş Ortak Bilgi Grafiği

Şekil 5.2'de küme kirliliği ve normalleştirilmiş ortak bilgi metrikleri için alınan sonuç değerlerin her bir adım için değişimi görülmektedir. Bu şekle göre; Adım 5, 6, 7 ve 8'de uygulanan yeni soyut kavram-etiket adayı eşleştirmesi yönteminin daha başarılı NMI değerlerinin elde edilmesini sağladığı, kümelere doküman atama stratejisi için hem tek kelimelik, hem de ifade etiket adayları için kelimelerin eş anlamlılarından faydalanılması önerisinin uygulandığı Adım 2 ve Adım 4 adımları için ise daha kirli kümeler oluşturulduğu görülebilir.

Çizelge 5.3, Şekil 5.1 ve Şekil 5.2'deki sonuçlar birlikte değerlendirildiğinde tüm adımlar arasında, tüm metrikler açısından genel olarak en başarılı sonuçların Adım 8'de elde edildiği söylenebilir. Başka bir deyişle güncel algoritmaya göre en iyi iyileştirme; önerilen yeni soyut kavram-etiket adayı eşleştirimesi yönteminin 4. bölümde anlatılan seçeneklerden 2. seçenek kullanılarak uygulanması ve kümelere doküman atanması süreci için önerilen eş anlamlılardan faydalanma yaklaşımının yalnızca ifade etiket adayları için uygulanması ile elde edilmektedir. Bu yüzden daha detaylı değerlendirmeler için yalnızca; güncel Lingo algoritmasının uygulandığı adım olan Adım 1 ve en başarılı iyileştirmelerin elde edildiği Adım 8 karşılaştırılmıştır. [64]'de örnek olarak Beagle başlığı seçilmiş ve en iyi iyileştirmenin alındığı adımda, güncel Lingo algoritmasının uygulandığı

adıma göre daha iyi sonuçlar alındığı gösterilmişti. Bu çalışmada Beagle başlığı ile birlikte, Labyrinth ve Jaguar başlıkları için de detaylı sonuçlara yer verilmiştir. Çizelge 5.4, Çizelge 5.5 ve Çizelge 5.6'da sırasıyla Beagle, Labyrinth ve Jaguar başlıkları için verisetinde tanımı yapılan doğru küme tanımları verilmiştir. Peşinden; Çizelge 5.7, Çizelge 5.8 ve Çizelge 5.9'da, her bir doğru küme için; doğru kümeyle eşleşen sonuç kümelerin etiketleri ve f-ölçütü, duyarlılık ve anma değerleri, Adım 1 ve Adım 8 için verilmiştir.

Çizelge 5.7, Çizelge 5.8 ve Çizelge 5.9 incelendiğinde; Çizelge 5.7'de Beagle, Çizelge 5.8'de Labyrinth ve Çizelge 5.9'da Jaguar başlıkları için genel olarak Adım 8'de, Adım 1'e kıyasla daha başarılı sonuçlar alındığı görülebilir.

Çizelge 5.7'de Beagle başlığı için yalnızca DK2 doğru kümesi bazında Adım 1'de Adım 8'e göre daha iyi bir sonuç alındığı gözlenmektedir. Ancak, Çizelge 5.4 incelendiğinde DK2 kümesinin toplamda 86 doküman içinde yalnızca 2 doküman içerdiği görülmektedir. Bu nedenle DK2 kümesi doğru kümeler içinde bir aykırı küme olarak görülebilir.

Çizelge 5.8'de Labyrinth başlığı için DK1 dışındaki bütün doğru kümeler için Adım 1 ve Adım 8'de tamamen aynı sonuçların üretildiği ve DK1 için de Adım 8'in Adım 1'e göre daha başarılı olduğu görülmektedir.

Çizelge 5.9'da Jaguar başlığı için, aykırı küme olarak düşünülebilecek DK1 ve DK2 kümeleri için Adım 1 ve Adım 8'de aynı şekilde doğru kümelerle tamamen eşleşen kümeler oluşturulduğu görülmektedir. Başka bir az sayıda doküman içeren doğru küme olan DK6 için Adım 8'de, Adım 1'e göre 1 tane fazladan ilişkisiz doküman içeren bir küme eşleştiği için duyarlılık ve f-ölçütü değerinde azalma görülmektedir. Bu aykırı kümeler dışındaki doğru kümeler olan DK2, DK3 ve DK5 doğru kümeleri içinse, Beagle başlığında olduğu gibi [64], çarpıcı bir iyileştirme gözlenmektedir. Bu doğru kümeler için, duyarlılık değerini hiç düşürmeden, daha yüksek anma ve dolayısıyla daha yüksek f-ölçütü değerlerinin elde edildiği daha iyi kümeler ile eşleştirme yapılması sağlanmıştır.

Beagle, Labyrinth ve Jaguar başlıkları için verilen bütün sonuçlar birlikte değerlendirildiğinde, önerilen yöntemlerin (Adım 8) güncel Lingo algoritmasına (Adım 1) göre daha başarılı sonuçlar verdiği görülebilir. Bu başarının altında, anlamsal bir yöntem olan LSI ile keşfedilen soyut kavramların, kendilerini daha iyi

temsil eden etiket adaylarıyla eşleştirilebilmesi yatmaktadır. Özellikle Beagle ve Jaguar ile ilgili sonuçlar incelendiğinde, doğru kümelere karşılık olarak Adım 8'de elde edilen kümelerin etiketlerinin, Adım 1'dekilere göre doğru küme tanımlarını daha iyi temsil ettikleri görülebilir.

Çizelge 5.4 Beagle Başlığı İçin Doğru Küme Tanımları

Doğru Küme	**Tanım**	**Sonuç Doküman Sayısı**
DK1	Beagle is a dog breed.	55
DK2	HMS Beagle, the ship in which Charles Darwin undertook the travels during which he made many observations which became important for his formulation of his theory of evolution	2
DK3	Beagle 2, a failed British Mars lander named after HMS Beagle. It crashed on 25 December 2003.	11
DK4	Beagle (software), a desktop search service for GNU/Linux users.	18

Çizelge 5.5 Labyrinth Başlığı İçin Doğru Küme Tanımları

Doğru Küme	**Tanım**	**Sonuç Doküman Sayısı**
DK1	Labyrinth(film), a 1986 fantasy film directed by Jim Henson	9
DK2	Labyrinth was an elaborate maze in Greek mythology constructed by the artificer Daedalus to hold the Minotaur	4
DK3	Pan's Labyrinth, a 2006 Spanish film by Guillermo del Toro	6
DK4	Labyrinth(book), a historical novel by writer and journalist Kate Mosse	1

DK5	Labyrinth(Juno Reactor album), a 2004 album by Goa trance group Juno Reactor	1
DK6	Labyrinth(band), an Italian power metal band	5

Çizelge 5.6 Jaguar Başlığı İçin Doğru Küme Tanımları

Doğru Küme	**Tanım**	**Sonuç Doküman Sayısı**
DK1	Jaguar, the codename for Mac OS X v10	2
DK2	Jaguar(Panthera onca), a New World mammal(a"big cat") of the Felidae family native to South and Central America	22
DK3	Jaguar(car), a British luxury car manufacturer, owned by Ford as of 1990	47
DK4	SEPECAT Jaguar, a military aircraft	2
DK5	Atari Jaguar, a video game console made by Atari	5
DK6	Fender Jaguar, guitar introduced in 1962, built by Fender	2

Çizelge 5.7 Beagle Başlığı İçin Adım 1 ve Adım 8 Sonuçları

	Adım 1				**Adım 8**			
Küme	**Etiket**	**F-Ölçütü**	**Duyarlılık**	**Anma**	**Etiket**	**F-Ölçütü**	**Duyarlılık**	**Anma**
DK1	Adopting	0,281	1,000	0,164	Breed	0,571	1,000	0,400
DK2	Project	0,500	0,500	0,500	Beagle Resource	0,286	0,200	0,500
DK3	ESA’s Mars Express	0,429	1,000	0,273	Mars	0,900	1,000	0,820
DK4	Desktop Search	0,500	1,000	0,333	Search	0,643	0,900	0,500

Çizelge 5.8 Labyrinth Başlığı İçin Adım 1 ve Adım 8 Sonuçları

	Adım 1				**Adım 8**			
Küme	**Etiket**	**F-Ölçütü**	**Duyarlılık**	**Anma**	**Etiket**	**F-Ölçütü**	**Duyarlılık**	**Anma**
DK1	Henson Film	0,615	1,000	0,444	Henson	0,941	1,000	0,889
DK2	Greek Mythology	0,667	1,000	0,500	Greek Mythology	0,667	1,000	0,500
DK3	Pan's Labyrinth	0,909	1,000	0,833	Pan's Labyrinth	0,909	1,000	0,833
DK4	Ring	0,667	0,500	1,000	Ring	0,667	0,500	1,000
DK5	Music	0,667	0,500	1,000	Music	0,667	0,500	1,000
DK6	Metal	0,750	1,000	0,600	Metal	0,750	1,000	0,600

Çizelge 5.9 Jaguar Başlığı İçin Adım 1 ve Adım 8 Sonuçları

	Adım 1				Adım 8			
Küme	**Etiket**	**F-Ölçütü**	**Duyarlılık**	**Anma**	**Etiket**	**F-Ölçütü**	**Duyarlılık**	**Anma**
DK1	Mac	1,000	1,000	1,000	Mac	1,000	1,000	1,000
DK2	Jaguar Animal	0,370	1,000	0,227	Cat	0,667	1,000	0,500
DK3	Jaguar Dealer	0,433	1,000	0,277	Car	0,795	1,000	0,660
DK4	Tactical Support Aircraft	1,000	1,000	1,000	Tactical Support Aircraft	1,000	1,000	1,000
DK5	Game Database	0,571	1,000	0,400	Game	0,889	1,000	0,800
DK6	Website of Fender Musical Instruments	1,000	1,000	1,000	Music	0,800	0,667	1,000

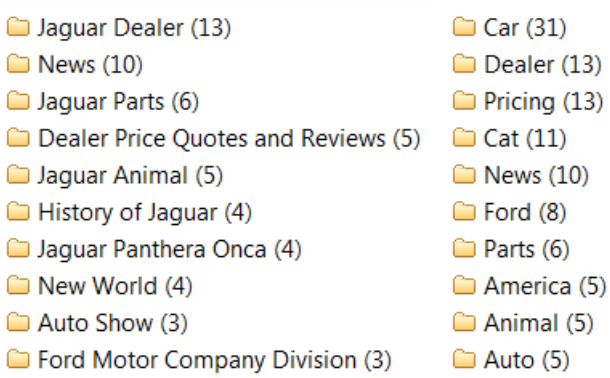
Beagle, Labyrinth ve Jaguar başlıkları için Adım 1'de ve Adım 8'de üretilen ilk 10'ar sonuç kümenin etiketleri sırasıyla Şekil 5.3, Şekil 5.4 ve Şekil 5.5'de soldan sağa verilmiştir. Bu şekiller incelendiğinde:

- Beagle başlığı için ilk 10 sonuç küme etiketleri içerisinde; Adım 8'de 1, 2 ve 5. konumlarda doğru kümelerle eşleşen toplamda 41 elemanlı 3 küme bulunmaktayken, Adım 1'de 1 ve 7. konumlarda doğru kümelerle eşleşen toplamda 15 elemanlı 2 sonuç küme bulunmaktadır.
- Labyrinth başlığı için ilk 10 sonuç küme etiketleri içerisinde; Adım 8'de 2, 3, 4, 8 ve 10. konumlarda doğru kümelerle eşleşen toplamda 20 elemanlı 5 küme bulunmaktayken, Adım 1'de de 1, 2, 6, 8 ve 9. konumlarda doğru kümelerle eşleşen toplamda 16 elemanlı yine 5 sonuç küme bulunmaktadır.
- Jaguar başlığı için ilk 10 sonuç küme etiketleri içerisinde; Adım 8'de 1 ve 4. konumlarda doğru kümelerle eşleşen toplamda 44 elemanlı 2 küme bulunmaktayken, Adım 1'de 1 ve 5. konumlarda doğru kümelerle eşleşen toplamda 18 elemanlı 2 sonuç küme bulunmaktadır.

Bu sonuçlar sıralama (ranking) açısından değerlendirildiğinde, önerilen yöntemlerin bir başka kazanımı olarak, kullanıcıların aradıkları sonuçlara daha üst sıralarda, daha hızlı erişebilmelerini sağlayacak sonuçların elde edilebilmesi görülebilir.

Şekil 5.3 Beagle Başlığı için Adım 1 ve Adım 8'de Üretilen İlk 10 Sonuç Küme Etiketleri

Şekil 5.4 Labyrinth Başlığı için Adım 1 ve Adım 8'de Üretilen İlk 10 Sonuç Küme Etiketleri



Şekil 5.5 Jaguar Başlığı için Adım 1 ve Adım 8'de Üretilen İlk 10 Sonuç Küme Etiketleri

Son olarak, Adım 1 ve Adım 8 için, Çizelge 5.10'da her bir başlık için 5.2 bölümünde anlatılan şekilde hesaplanan ağırlıklı ortalama sonuç değerleri ve Çizelge 5.11'de bu değerlerin Adım 8'de, Adım 1'e göre değişim değerleri (Adım 8'deki değerlerin Adım 1'e göre farkı) verilmiştir. Bu çizelgelerden, başlık bazındaki ağırlıklı ortalama sonuç değerlerinin, tüm başlıklar bazındaki ortalama sonuç değerleriyle paralel olduğu görülebilir. Sonuç olarak, önerilen yöntemler; anma, f-ölçütü ve NMI değerlerini yüksek oranda arttırırken, duyarlılık ve küme saflığı değerlerini görece kabul edilebilir düzeylerde azaltmaktadır.

Çizelge 5.10 Her bir Başlık için Sonuçların Ağırlıklı Ortalama Değerleri

	Adım 1					**Adım8**				
	P	**R**	**F**	**CC**	**NMI**	**P**	**R**	**F**	**CC**	**NMI**
AIDA	0,839	0,433	0,532	0,373	0,434	0,717	0,567	0,570	0,401	0,521
B_52	0,915	0,267	0,402	0,208	0,305	1,000	0,493	0,656	0,202	0,453
BEAGLE	0,988	0,221	0,351	0,106	0,366	0,960	0,477	0,622	0,139	0,429
BRONX	0,783	0,355	0,432	0,297	0,337	0,837	0,553	0,644	0,286	0,320
CAIN	0,974	0,474	0,531	0,292	0,667	0,824	0,632	0,683	0,193	0,609
CAMEL	0,993	0,429	0,572	0,124	0,596	0,743	0,814	0,704	0,385	0,617
CORAL_SEA	0,880	0,605	0,682	0,334	0,458	0,838	0,721	0,754	0,367	0,550
CUBE	0,928	0,813	0,860	0,167	0,887	0,934	0,854	0,879	0,245	0,903
EOS	0,979	0,500	0,601	0,149	0,627	0,968	0,906	0,935	0,182	0,799
EXCALIBUR	0,855	0,563	0,636	0,233	0,695	0,807	0,656	0,660	0,291	0,663
FAHRENHEIT	0,929	0,582	0,675	0,161	0,709	0,899	0,761	0,796	0,230	0,818
GLOBE	0,896	0,717	0,778	0,295	0,788	0,861	0,755	0,773	0,400	0,783
HORNET	0,913	0,659	0,726	0,288	0,778	0,879	0,659	0,704	0,356	0,730
INDIGO	0,800	0,895	0,829	0,394	0,882	0,778	0,895	0,815	0,423	0,834
IWO_JIMA	0,943	0,314	0,400	0,397	0,483	0,733	0,779	0,730	0,399	0,534
JAGUAR	1,000	0,325	0,467	0,175	0,421	0,992	0,650	0,776	0,127	0,600
LA_PLATA	0,954	0,478	0,598	0,288	0,591	0,870	0,821	0,829	0,375	0,674
LABYRINTH	0,962	0,615	0,721	0,270	0,743	0,962	0,769	0,834	0,371	0,817
LANDAU	0,846	0,900	0,857	0,345	0,908	0,833	0,925	0,865	0,392	0,906
LIFE_ON_MARS	0,983	0,250	0,366	0,056	0,174	0,984	0,417	0,569	0,067	0,154
LOCUST	0,863	0,458	0,560	0,337	0,634	0,863	0,563	0,659	0,322	0,693
MAGIC_MOUNTAIN	0,899	0,585	0,634	0,356	0,699	0,889	0,878	0,869	0,240	0,834

MATADOR	0,910	0,595	0,693	0,124	0,807	0,781	0,730	0,703	0,252	0,758
METAMORPHOSIS	0,967	0,600	0,707	0,207	0,594	0,967	0,600	0,707	0,286	0,603
MINOTAUR	0,971	0,431	0,563	0,059	0,578	0,934	0,588	0,688	0,089	0,681
MIRA	0,899	0,553	0,563	0,465	0,723	0,783	0,789	0,760	0,335	0,806
MIRAGE	0,890	0,588	0,651	0,136	0,800	0,882	0,676	0,711	0,333	0,757
MONTE_CARLO	0,896	0,625	0,730	0,204	0,739	0,930	0,736	0,803	0,214	0,770
OPPENHEIM	0,919	0,756	0,809	0,296	0,840	0,951	0,829	0,870	0,252	0,904
OUT_OF_CONTROL	0,796	0,833	0,804	0,480	0,909	0,880	0,889	0,870	0,480	0,928
PELICAN	0,921	0,500	0,619	0,305	0,637	0,885	0,700	0,710	0,581	0,712
PURPLE_HAZE	0,864	0,667	0,696	0,370	0,703	0,804	1,000	0,868	0,388	0,824
RAAM	0,913	0,466	0,608	0,141	0,428	0,921	0,638	0,749	0,135	0,547
RHEA	0,981	0,442	0,565	0,159	0,659	0,981	0,692	0,804	0,144	0,725
SCORPION	0,862	0,477	0,533	0,255	0,572	0,841	0,477	0,527	0,191	0,593
THE_LITTLE_MERMAID	0,762	0,396	0,487	0,442	0,500	0,812	0,521	0,620	0,380	0,470
TORTUGA	0,966	0,759	0,832	0,248	0,805	0,865	0,897	0,838	0,407	0,829
URANIA	0,919	0,614	0,679	0,308	0,739	0,868	0,705	0,758	0,400	0,701
WINK	0,824	0,717	0,749	0,269	0,882	0,810	0,783	0,785	0,325	0,845
XANADU	0,832	0,551	0,557	0,406	0,622	0,808	0,673	0,664	0,453	0,682
ZEBRA	0,948	0,549	0,628	0,163	0,711	0,955	0,592	0,678	0,237	0,674
ZENITH	0,933	0,500	0,624	0,246	0,681	0,933	0,700	0,786	0,239	0,763
ZODIAC	0,925	0,650	0,740	0,148	0,783	0,925	0,700	0,752	0,174	0,793
ZOMBIE	0,912	0,647	0,681	0,385	0,757	0,926	0,647	0,698	0,250	0,809
ORTALAMA	0,908	0,554	0,630	0,260	0,651	0,878	0,707	0,743	0,294	0,691

Çizelge 5.11 Her bir Başlığın Sonuçların Ağırlıklı Ortalama Değerlerinin Adım 8'de Adım 1'e göre Değişim Miktarı

	Adım 1					Adım 8 – Adım 1				
	P	R	F	CC	NMI	P	R	F	CC	NMI
AIDA	0,839	0,433	0,532	0,373	0,434	-0,122	0,134	0,038	0,028	0,087
B_52	0,915	0,267	0,402	0,208	0,305	0,085	0,226	0,254	-0,006	0,148
BEAGLE	0,988	0,221	0,351	0,106	0,366	-0,028	0,256	0,271	0,033	0,063
BRONX	0,783	0,355	0,432	0,297	0,337	0,054	0,198	0,212	-0,011	-0,017
CAIN	0,974	0,474	0,531	0,292	0,667	-0,150	0,158	0,152	-0,099	-0,058
CAMEL	0,993	0,429	0,572	0,124	0,596	-0,250	0,385	0,132	0,261	0,021
CORAL_SEA	0,880	0,605	0,682	0,334	0,458	-0,042	0,116	0,072	0,033	0,092
CUBE	0,928	0,813	0,860	0,167	0,887	0,006	0,041	0,019	0,078	0,016
EOS	0,979	0,500	0,601	0,149	0,627	-0,011	0,406	0,334	0,033	0,172
EXCALIBUR	0,855	0,563	0,636	0,233	0,695	-0,048	0,093	0,024	0,058	-0,032
FAHRENHEIT	0,929	0,582	0,675	0,161	0,709	-0,030	0,179	0,121	0,069	0,109
GLOBE	0,896	0,717	0,778	0,295	0,788	-0,035	0,038	-0,005	0,105	-0,005
HORNET	0,913	0,659	0,726	0,288	0,778	-0,034	0,000	-0,022	0,068	-0,048
INDIGO	0,800	0,895	0,829	0,394	0,882	-0,022	0,000	-0,014	0,029	-0,048
IWO_JIMA	0,943	0,314	0,400	0,397	0,483	-0,210	0,465	0,330	0,002	0,051
JAGUAR	1,000	0,325	0,467	0,175	0,421	-0,008	0,325	0,309	-0,048	0,179
LA_PLATA	0,954	0,478	0,598	0,288	0,591	-0,084	0,343	0,231	0,087	0,083
LABYRINTH	0,962	0,615	0,721	0,270	0,743	0,000	0,154	0,113	0,101	0,074
LANDAU	0,846	0,900	0,857	0,345	0,908	-0,013	0,025	0,008	0,047	-0,002
LIFE_ON_MARS	0,983	0,250	0,366	0,056	0,174	0,001	0,167	0,203	0,011	-0,020
LOCUST	0,863	0,458	0,560	0,337	0,634	0,000	0,105	0,099	-0,015	0,059
MAGIC_MOUNTAIN	0,899	0,585	0,634	0,356	0,699	-0,010	0,293	0,235	-0,116	0,135
MATADOR	0,910	0,595	0,693	0,124	0,807	-0,129	0,135	0,010	0,128	-0,049
METAMORPHOSIS	0,967	0,600	0,707	0,207	0,594	0,000	0,000	0,000	0,079	0,009

MINOTAUR	0,971	0,431	0,563	0,059	0,578	-0,037	0,157	0,125	0,030	0,103
MIRA	0,899	0,553	0,563	0,465	0,723	-0,116	0,236	0,197	-0,130	0,083
MIRAGE	0,890	0,588	0,651	0,136	0,800	-0,008	0,088	0,060	0,197	-0,043
MONTE_CARLO	0,896	0,625	0,730	0,204	0,739	0,034	0,111	0,073	0,010	0,031
OPPENHEIM	0,919	0,756	0,809	0,296	0,840	0,032	0,073	0,061	-0,044	0,064
OUT_OF_CONTROL	0,796	0,833	0,804	0,480	0,909	0,084	0,056	0,066	0,000	0,019
PELICAN	0,921	0,500	0,619	0,305	0,637	-0,036	0,200	0,091	0,276	0,075
PURPLE_HAZE	0,864	0,667	0,696	0,370	0,703	-0,060	0,333	0,172	0,018	0,121
RAAM	0,913	0,466	0,608	0,141	0,428	0,008	0,172	0,141	-0,006	0,119
RHEA	0,981	0,442	0,565	0,159	0,659	0,000	0,250	0,239	-0,015	0,066
SCORPION	0,862	0,477	0,533	0,255	0,572	-0,021	0,000	-0,006	-0,064	0,021
THE_LITTLE_MERMAID	0,762	0,396	0,487	0,442	0,500	0,050	0,125	0,133	-0,062	-0,030
TORTUGA	0,966	0,759	0,832	0,248	0,805	-0,101	0,138	0,006	0,159	0,024
URANIA	0,919	0,614	0,679	0,308	0,739	-0,051	0,091	0,079	0,092	-0,038
WINK	0,824	0,717	0,749	0,269	0,882	-0,014	0,066	0,036	0,056	-0,037
XANADU	0,832	0,551	0,557	0,406	0,622	-0,024	0,122	0,107	0,047	0,060
ZEBRA	0,948	0,549	0,628	0,163	0,711	0,007	0,043	0,050	0,074	-0,037
ZENITH	0,933	0,500	0,624	0,246	0,681	0,000	0,200	0,162	-0,007	0,082
ZODIAC	0,925	0,650	0,740	0,148	0,783	0,000	0,050	0,012	0,026	0,010
ZOMBIE	0,912	0,647	0,681	0,385	0,757	0,014	0,000	0,017	-0,135	0,052
ORTALAMA	0,908	0,554	0,630	0,260	0,651	-0,030	0,153	0,112	0,034	0,040

6. SONUÇ VE ÖNERİLER

Arama Sonucu Kümeleme (ASK) problemi, kullanıcıların bir arama motorundan dönen sonuçlar içerisinde aradığı sonuçlara daha kolay ulaşabilmelerinin amaçlandığı bir çalışma konusudur. Bir ASK algoritmasının başarılı olabilmesi için sonuçların doğru bir şekilde kümelenmesi ve kümelerin son kullanıcıya anlaşılır etiketlerle sunulması gereklidir.

Bu tez çalışmasında, popüler ve başarılı bir ASK algoritması olan Lingo algoritmasının eksikliklerini gidermeye yönelik öneriler sunulmuş ve önerilerin uygulanmasıyla elde edilen sonuçlar paylaşılmıştır.

Lingo algoritması, kümeler için anlaşılırlık ve anlamlılık açısından başarılı etiketler üretebilmektedir. Bu başarısını, klasik kümeleme yaklaşımından farklı olarak; öncelikle etiketlerin belirlenmesini ve daha sonra belirlenen etiketlere göre kümelere ilişkili dokümanların atanmasını prensip edinmesine borçludur. Lingo algoritmasında; doküman kümesinde sık geçen ifadeler etiket adaylarını belirlemekte ve LSI yaklaşımıyla tüm dokümanlar içinde geçen gizli soyut kavramlar tespit edilmektedir. Daha sonra her bir soyut kavram bir etiket adayıyla eşleştirilerek kümeler boş olarak oluşturulmakta, ardından küme etiketlerindeki etkisiz kelimeler dışındaki bütün terimleri içeren dokümanlar etiketlerin temsil ettiği kümelere atanmaktadır.

Lingo algoritmasının eksikliklerinden bir tanesi, küme etiketiyle anlamsal olarak ilişkili olduğu halde, etiket terimlerini içermeyen dokümanların ilgili kümelere atanmasının mümkün olmamasıdır. Bu eksikliği gidermek için, etiketlerdeki terimlerin kendisini ya da eş anlamlısını içeren dokümanların kümelere atanması yaklaşımı önerilmiştir. Bu yöntem; hem tek kelimelik etiket adayları, hem de birden fazla kelimeden oluşan, ifade etiket adayları için denenmiş ve yalnızca ifade etiket adayları için uygulandığında ilkine göre daha başarılı sonuçlar alınmıştır. Bu önerinin sonuçlarını etkileyen birincil etken, üretilen kümelerin anma değerlerini arttırmaya çalışırken, gürültü olarak adlandırabileceğimiz, aslında kümeyle ilişkili olmadığı halde kümelere atanabilen dokümanların; duyarlılık ve küme saflığı değerlerini düşürmeleridir. Önerilen yaklaşımın bütün etiket adayları yerine yalnızca ifade etiket adayları uygulanması; gürültü mahiyetindeki dokümanların

kümelere atanması riskini azaltırken, hem de anma değerinin arttırılabilmesini sağlamıştır. Bunun sonucu olarak f-ölçütü değeri de arttırılabilmiştir.

Lingo algoritmasının bir diğer büyük eksiği, soyut kavramların etiket adaylarıyla eşleştirilmesi aşamasındadır. Terim-doküman matrisinin, LSI yöntemiyle; terim-soyut kavram, soyut kavram-soyut kavram ve doküman-soyut kavram matrislerine ayrıştırılmasının ardından, bu matrislerle aynı terim uzayında uzanacak şekilde etiket adaylarından terim-ifade (etiket adayı) matrisi oluşturulmaktadır. Ardından terim-soyut kavram ve terim-ifade matrisleriyle ifade edilen soyut kavram ve ifade vektörleri; her bir soyut kavram vektörü, kosinüs benzerliği yöntemiyle kendisine en çok benzeyen ifade vektörüyle eşleşecek şekilde eşleştirilmektedir. Ancak, ifade vektörleri genelde az sayıda terimle ifade edildiği için bu karşılaştırma çok sağlıklı sonuçlar vermeyebilmektedir.

Lingo algoritmasının soyut kavram-etiket adayı eşleştirilmesindeki bu eksikliğini gidermek için öncelikle ifade vektörlerinin, etiketlerde geçen terimlerin eş anlamlı terimlerinin de ifadeleri temsil eden vektörlere dahil edilerek zenginleştirilmesi önerilmiştir. Ancak, bu öneri beklenen başarıyı gösterememiştir. Bundan dolayı bu eşleştirme işlemi için ikinci ve yeni bir öneri sunulmuştur. Bu öneri kapsamında öncelikle, LSI yaklaşımı sonucunda oluşan doküman-soyut kavram matrisinden faydalanılarak, her bir soyut kavramla ilişkili dokümanların belirlenmesi ve ardından soyut kavramlar ve etiket adaylarının ortaklaşa ilişkili oldukları doküman sayısı değeri benzerlik ölçütü olarak kullanılacak şekilde karşılaştırılıp, eşleştirilmeleri önerilmiştir. Bu önerinin ön koşulu niteliğinde olan soyut kavramlarla ilişkili dokümanların belirlenmesi işlemi için iki alternatif önerilmiştir. Bunlardan ilkinde dokümanlar belirli bir eşik değerinden daha yüksek skorla benzedikleri bütün soyut kavramlarla ilişkilendirilebilirken, ikincisinde yalnızca en yüksek dereceyle benzedikleri soyut kavramla ilişkilendirilebilmektedir. Bu alternatif yöntemler vasıtasıyla, dolaylı olarak soyut kavramlarla ilişkili dokümanlar belirlenebilmektedir. Sonuçlar karşılaştırıldığında önerinin ikinci alternatifle uygulandığında, düşük bir farkla da olsa ilkine göre daha başarılı sonuçlar verdiği gözlenmiştir.

Deneylerde en başarılı iyileştirme, yeni soyut kavram-etiket adayı eşleştirme yönteminin ikinci alternatifle uygulanması önerisinin ve kümelere doküman atanması için önerilen yalnızca ifade etiket adayları için eş anlamlılardan

faydalanma önerisinin beraber uygulanmasıyla elde edilmiştir. Bu kombine yaklaşımın sonucunda; anma ve f-ölçütü değerlerinin yüksek oranda artmasına karşın, duyarlılık ve küme saflığı değerlerinde anma değerindeki artışın yaklaşık beşte biri kadar bir azalma olduğu ve NMI değerinin de arttığı görülmüştür. Ayrıca, verisetindeki örnek başlıklar için oluşturulan kümeler incelendiğinde, önerilen yaklaşımın güncel algoritmaya göre daha çok doğru kümeyle; daha üst sıralarda ve daha çok eleman içeren kümeleri eşleştirebildiği görülmüştür.

Gelecek çalışmalarda, Lingo algoritmasının küme birleştirme aşaması incelenerek, anlamsal olarak birbirleriyle ilişkili olan kümelerin birleştirilmesi sağlanabilir. Bu işlem, anma ve f-ölçütü değerlerinin, duyarlılık değerine zarar vermeden daha da arttırılabilmesini sağlayabilir. Ayrıca; kümelerdeki eleman sayısının kontrol altına alınabildiği, esnek ve anlamsal olarak kümelerle ilişkili olan dokümanların ilgili sonuç kümelere atanmasına imkan veren alternatif öneriler geliştirilebilir.

KAYNAKLAR

[1] S. Osi'nski, J. Stefanowski, and D. Weiss, "Lingo: Search results clustering algorithm based on singular value decomposition," in in Intelligent information processing and web mining, Springer, **2004**, pp. 359–368.

[2] C. Carpineto, S. Osi'nski, G. Romano, and D. Weiss, "A survey of web clustering engines," ACM Computing Surveys (CSUR), vol. 41, no. 3, p. 17, **2009**.

[3] "Princeton University 'About WordNet.' WordNet.," **2010**. [Online]. Available: http://wordnet.princeton.edu. [Accessed: 30-Aug-2013].

[4] Ç. Çallı, "Improving Search Result Clustering by Integrating Semantic Information From Wikipedia," MIDDLE EAST TECHNICALUNIVERSITY, **2010**.

[5] D. R. Cutting, D. R. Karger, J. O. Pedersen, and J. W. Tukey, "Scatter/Gather: a cluster-based approach to browsing large document collections," in Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval, **1992**, pp. 318–329.

[6] M. A. Hearst and J. O. Pedersen, "Reexamining the cluster hypothesis: scatter/gather on retrieval results," in Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval, **1996**, pp. 76–84.

[7] C. L. Ngo and H. S. Nguyen, "A tolerance rough set approach to clustering web search results," in in Knowledge Discovery in Databases: PKDD 2004, Springer, **2004**, pp. 515–517.

[8] Y. S. Maarek, R. Fagin, I. Z. Ben-Shaul, and D. Pelleg, "Ephemeral document clustering for web applications," **2000**.

[9] F. Giannotti, M. Nanni, D. Pedreschi, and F. Samaritani, "WebCat: Automatic Categorization of Web Search Results.," in SEBD, **2003**, pp. 507–518.

[10] B. Stein and S. M. Zu Eissen, "Topic identification: Framework and application," in Proc. International Conference on Knowledge Management, **2004**, vol. 400, pp. 522–531.

[11] O. Zamir and O. Etzioni, "Web document clustering: a feasibility demonstration," in Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval, **1998**, pp. 46–54.

[12] O. Zamir and O. Etzioni, "{Grouper}: a dynamic clustering interface to {Web} search results," Computer Networks (Amsterdam, Netherlands: 1999), vol. 31, no. 11–16, pp. 1361–1374, **1999**.

[13] P. Ferragina and A. Gullì, "The anatomy of SnakeT: a hierarchical clustering engine for web-page snippets," in Proceedings of the 8th

European Conference on Principles and Practice of Knowledge Discovery in Databases, **2004**, pp. 506–508.

[14] P. Ferragina and A. Gullì, “Experimenting SnakeT: A hierarchical clustering engine for web-page snippets,” in Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, **2004**, pp. 543–545.

[15] P. Ferragina and A. Gulli, “A personalized search engine based on web-snippet hierarchical clustering,” in Special interest tracks and posters of the 14th international conference on World Wide Web, **2005**, pp. 801–810.

[16] “Vivisimo Clustering Engine,” [Online]. Available: http://vivisimo.com. [Accessed: 30-Aug-2013].

[17] K. Kummamuru, R. Lotlikar, S. Roy, K. Singal, and R. Krishnapuram, “A hierarchical monothetic document clustering algorithm for summarization and browsing search results,” in Proceedings of the 13th international conference on World Wide Web, **2004**, pp. 658–665.

[18] H.-J. Zeng, Q.-C. He, Z. Chen, W.-Y. Ma, and J. Ma, “Learning to cluster web search results,” in Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, **2004**, pp. 210–217.

[19] O. Zamir, O. Etzioni, O. Madani, and R. M. Karp, “Fast and Intuitive Clustering of Web Documents.,” in KDD, **1997**, vol. 97, pp. 287–290.

[20] O. E. Zamir, “Clustering web documents: a phrase-based method for grouping search engine results,” University of Washington, **1999**.

[21] I. Mas\lowska, “Phrase-based hierarchical clustering of web search results,” in Proceedings of the 25th European conference on IR research, **2003**, pp. 555–562.

[22] U. Manber and G. Myers, “Suffix arrays: a new method for on-line string searches,” in Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms, **1990**, pp. 319–327.

[23] Z. Dong, “Towards Web Information Clustering,” Southeast University, Nanjing, China, **2002**.

[24] D. Zhang and Y. Dong, “Semantic, Hierarchical, Online Clustering of Web Search Results,” in in Advanced Web Technologies and Applications, vol. 3007, J. Yu, X. Lin, H. Lu, and Y. Zhang, Eds. Springer Berlin Heidelberg, **2004**, pp. 69–78.

[25] J. Janruang and W. Kreesuradej, “A New Web Search Result Clustering based on True Common Phrase Label Discovery,” in Computational Intelligence for Modelling, Control and Automation, 2006 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on, **2006**, p. 242.

[26] H. Wen, G.-S. Huang, and Z. Li, “Clustering web search results using semantic information,” in Machine Learning and Cybernetics, 2009 International Conference on, **2009**, vol. 3, pp. 1504–1509.

[27] G. Mecca, S. Raunich, and A. Pappalardo, “A new algorithm for clustering search results,” Data & Knowledge Engineering, vol. 62, no. 3, pp. 504–522, Sep. **2007**.

[28] X. He, H. Zha, C. H.Q. Ding, and H. D. Simon, “Web document clustering using hyperlink structures,” Comput. Stat. Data Anal., vol. 41, no. 1, pp. 19–45, **2002**.

[29] R. Bekkerman, S. Zilberstein, and J. Allan, “Web page clustering using heuristic search in the web graph,” in Proceedings of the 20th international joint conference on Artifical intelligence, **2007**, pp. 2280–2285.

[30] Y. Wang and M. Kitsuregawa, “Link Based Clustering of Web Search Results,” in in Advances in Web-Age Information Management, vol. 2118, X. S. Wang, G. Yu, and H. Lu, Eds. Springer Berlin Heidelberg, **2001**, pp. 225–236.

[31] Y. Wang and M. Kitsuregawa, “On combining link and contents information for web page clustering,” in Database and expert systems applications, **2002**, pp. 902–913.

[32] Y. Wang and M. Kitsuregawa, “Evaluating contents-link coupled web page clustering for web search results,” in Proceedings of the eleventh international conference on Information and knowledge management, **2002**, pp. 499–506.

[33] N. Duhan and A. K. Sharma, “A novel approach for organizing web search results using ranking and clustering,” International Journal of Computer Applications, vol. 5, no. 10, pp. 1–9, **2010**.

[34] D. S. K. Jayanthi and S. Prem, “Web Document Clustering and Visualization Results of Semantic Web Search Engine Using V Ranking,” International Journal of Computer Theory and Engineering (IJCTE), vol. 3, no. 3, pp. 462–466, **2011**.

[35] O. Alonso and M. Gertz, “Clustering of search results using temporal attributes,” in Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, **2006**, pp. 597–598.

[36] M. Alam and K. Sadaf, “Article: Web Search Result Clustering using Heuristic Search and Latent Semantic Indexing,” International Journal of Computer Applications, vol. 44, no. 15, pp. 28–33, **2012**.

[37] “Wikipedia.” [Online]. Available: http://wikipedia.org. [Accessed: 30-Aug-2013].

[38] Y. Meiyappan, N. Iyengar, and A. Kannan, “SRCluster: Web Clustering Engine based on Wikipedia,” International Journal of Advanced Science and Technology, vol. 39, pp. 1–18, **2012**.

[39] D. Carmel, H. Roitman, and N. Zwerdling, “Enhancing cluster labeling using wikipedia,” in Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, **2009**, pp. 139–146.

[40] S. Osi’nski, “An algorithm for clustering of web search results,” University of Technology, Poland, **2003**.

[41] S. Osi'nski and D. Weiss, "Conceptual clustering using lingo algorithm: Evaluation on open directory project data," in in Intelligent Information Processing and Web Mining, Springer, **2004**, pp. 369–377.

[42] "Open Directory Project." [Online]. Available: http://www.dmoz.org/. [Accessed: 30-Aug-2013].

[43] S. Osinski and D. Weiss, "A concept-driven algorithm for clustering search results," Intelligent Systems, IEEE, vol. 20, no. 3, pp. 48–54, **2005**.

[44] D. Weiss, "Cluster Contamination Measure."

[45] S. Osinski, "Improving quality of search results clustering with approximate matrix factorisations," in in Advances in Information Retrieval, Springer, **2006**, pp. 167–178.

[46] A. M. Sameh and A. M. Kadray, "Semantic web search results clustering using lingo and wordnet," International Journal of Research and Reviews in Computer Science (IJRRCS), vol. 1, no. 2, **2010**.

[47] B. S. Alsulami, M. F. Abulkhair, and F. A. Essa, "Semantic Clustering Approach Based Multi-agent System for Information Retrieval on Web," International Journal of Computer Science and Network Security, vol. 12, no. 1, p. 41, **2012**.

[48] D. Weiss, "Introduction to search results clustering," in Proceedings of the 6th International Conference on Soft Computing and Distributed Processing, **2002**, pp. 82–84.

[49] G. Salton, Automatic text processing: the transformation, analysis, and retrieval of information by computer. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., **1989**.

[50] R. Baeza-Yates, B. Ribeiro-Neto, and others, Modern information retrieval, vol. 463. ACM press New York, **1999**.

[51] Anonim, "Öklid uzaklığı - Vikipedi." [Online]. Available: http://tr.wikipedia.org/wiki/Öklid_uzaklığı. [Accessed: 08-Sep-2013].

[52] M. W. Berry, S. T. Dumais, and G. W. O'Brien, "Using linear algebra for intelligent information retrieval," SIAM review, vol. 37, no. 4, pp. 573–595, **1995**.

[53] S. Osinski and D. Weiss, "Carrot2: Design of a Flexible and Efficient Web Information Retrieval Framework," in in Advances in Web Intelligence, vol. 3528, P. Szczepaniak, J. Kacprzyk, and A. Niewiadomski, Eds. Springer Berlin Heidelberg, **2005**, pp. 439–444.

[54] D. Weiss and J. Stefanowski, "Web search results clustering in Polish: Experimental evaluation of Carrot," in in Intelligent Information Processing and Web Mining, Springer, **2003**, pp. 209–219.

[55] C. Carpineto and G. Romano, "ODP239 dataset.," **2009**. [Online]. Available: http://credo.fub.it/odp239/. [Accessed: 30-Aug-2013].

[56] C. Carpineto and G. Romano, "Ambient dataset.," **2008**. [Online]. Available: http://credo.fub.it/ambient/. [Accessed: 30-Aug-2013].

[57] M. F. Porter, "Snowball: A language for stemming algorithms." **2001**.

[58] “Carrot2 - Open Source Search Results Clustering Engine.” [Online]. Available: http://project.carrot2.org/. [Accessed: 30-Aug-2013].

[59] “release/3.7.1: Release 3.7.1 · carrot2/carrot2 · GitHub.” [Online]. Available: https://github.com/carrot2/carrot2/releases/tag/release/3.7.1. [Accessed: 31-Aug-2013].

[60] “PreprocessingContext (Carrot2 v3.7.1 API Documentation (JavaDoc)).” [Online]. Available: http://download.carrot2.org/stable/3.7.1/javadoc/org/carrot2/text/preprocessing/PreprocessingContext.html. [Accessed: 01-Sep-2013].

[61] “Java API for WordNet Searching (JAWS).” [Online]. Available: http://lyle.smu.edu/~tspell/jaws/. [Accessed: 30-Aug-2013].

[62] “Wikipedia - List of disambiguation pages.” [Online]. Available: http://en.wikipedia.org/wiki/Wikipedia:Links_to_(disambiguation)_pages. [Accessed: 30-Aug-2013].

[63] C. D. Manning, P. Raghavan, and H. Schütze, Introduction to Information Retrieval. New York, NY, USA: Cambridge University Press, **2008**.

[64] S. Demir, E. Sezer A, and H. Sever, “Modifications for the Cluster Content Discovery and the Cluster Label Induction Phases of the Lingo Algorithm,” International Journal of Computer Theory and Engineering, **In Press**.

EKLER

EK-1 Modifications for the Cluster Content Discovery and the Cluster Label Induction Phases of the Lingo Algorithm

EK-1 Modifications for the Cluster Content Discovery and the Cluster Label Induction Phases of the Lingo Algorithm

Modifications for the Cluster Content Discovery and the Cluster Label Induction Phases of the Lingo Algorithm

Seyfullah Demir, Ebru A. Sezer and Hayri Sever

***Abstract*—Search results clustering techniques help end users to find their related results easier. Both producing correct cluster contents and assigning descriptive, meaningful labels to the clusters are crucial for these techniques. *Lingo* is one of the most popular algorithms which consider both and it is known as a *description-comes-first* algorithm. Lingo has success on assigning descriptive, human-readable cluster labels, but it actually has a minor drawback on assigning documents to the clusters, which cause low recall values. In this paper, we propose two main modifications for the *Cluster Content Discovery* and the *Cluster Label Induction* phases of the Lingo algorithm. The evaluation of the experimental result shows that, although it causes a slight decrease in the precision, our modified Lingo algorithm provides quite higher recall and f-measure values.**

***Index Terms*—information retrieval, search results clustering, cluster content discovery, cluster labeling**

INTRODUCTION

Search engines are used in order to get the relevant results for a query. When users run a query on a search engine, a ranked list of the search results are returned with their snippets (partial content). If users run too general or ambiguous queries, it may be difficult to reach what they need, since a large number of results are returned and similar results are not grouped together. Search results clustering techniques are used to overcome this problem.

By employing search results clustering techniques, the search results are returned as labeled groups. A search result clustering algorithm should correctly cluster the results and also should produce descriptive labels for each cluster. Traditional clustering algorithms are not generally good enough at selecting descriptive labels for clusters. To overcome this problem, description-aware and description-centric algorithms [1] were developed. Zamir and Etzioni pioneered the approach of using recurring phrases in the search results clustering processes, within *Suffix Tree Clustering (STC)* algorithm in Grouper system [2]. In STC algorithm, the documents which share the same phrases are grouped together and the phrases that they share are used as cluster labels.

Later on, Osiński introduced the Lingo algorithm in his master thesis [3], and in [4] Osiński and his colleagues presented the Lingo. Lingo is a description-centric algorithm, in which the cluster labels are determined first, and then document assignments to the clusters are made. Osiński and Weiss presented an evaluation of the Lingo algorithm on the Open Directory Project (ODP) data [5]. Although their analysis was mostly non-numerical, they showed that Lingo separates the topics in the search results, better than the STC. Osiński and Weiss then showed Lingo produces significantly purer clusters than STC, by demonstrating a numerical analysis [6]. After that, Osiński investigated the effects of the matrix factorization method used in the search results clustering algorithm. He compared four different methods that are SVD (Singular Value Decomposition), NMF (Non-negative Matrix Factorisation), LNMF (Local Non-negative Matrix Factorisation) and CD (Concept Decomposition) in terms of topic separation, outlier detection and label quality [7]. He showed that NMF methods perform better than other methods including SVD, which was used in the original algorithm. According to the evaluation of his experiments, he also showed that Lingo with NMF-ED (NMF with Euclidean Distance minimisation) is significantly better than STC and TRC (Tolerance Rough Set Clustering) [8], in terms of topic separation and outlier detection.

In 2010, Sameh and Kadray proposed a modification in the *Frequent Phrase Extraction* phase of the Lingo algorithm [9]. They expanded the frequent phrases by including the synonyms which was obtained through the WordNet database [10]. They used the synonyms also when document assignments to the clusters are made. As a result, their algorithm could produce clusters that include the documents which contain the synonyms of the cluster labels, as well.

In [6], Osiński and Weiss mentioned a future work to enhance the Lingo's document assignment phase, due to low recall values. Since documents were assigned to the clusters using classic VSM (Vector Space Model) [11] approach, some irrelevant documents could be assigned to the clusters, while some semantically relevant documents could not be assigned. As the former could cause lower precision, the latter could lead to low recall. In the current implementation (v3.7.1) of the Lingo algorithm, instead of classic VSM approach, a binary similarity approach is used. Currently, a document is assigned to a cluster if it contains the stems of all words (except stop words) of the cluster label. Considering the results of the current implementation of the Lingo algorithm, we can see that the precision value is satisfactory, but the recall value can still be enhanced.

In this study we propose modifications for Lingo's Cluster Content Discovery and Cluster Label Induction phases. In Cluster Content Discovery phase, we propose a modification to overcome the weakness in the assignment process. According to our proposal, if a document contains the stem of the word itself or the stem of at least one

Manuscript received July 22, 2013.

synonym, for each word in the label**,** then it is assigned to the cluster. Additionally, in Cluster Label Induction phase, we propose a modification to match abstract concepts with cluster label candidates. In Lingo, the term-abstract concept matrix is attained, as a result of reducing the term-document matrix. Moreover, the term-label candidate matrix is built by conforming to the same term space as the term-abstract concept matrix. For the matching process of abstract concepts and labels, column vectors of both are compared via cosine similarity method. As another difference, we find the documents which are related to the abstract concepts and then match the abstract concepts with the label candidates by using the number of their common documents as similarity measure.

METHOD

In this section, firstly we give some brief information about the original and the current Lingo algorithm (v3.7.1), and then present our modification proposals for it.

Original Lingo Algorithm

A summarized version of the original Lingo algorithm is given below. More detailed algorithms can be seen in [3], [4].

1. Preprocess documents. For each document; do text filtering, identify the language of the document, apply stemming process and mark stop words.
2. Discover frequent terms and phrases as label candidates.
3. Discover abstract concepts by using SVD.
4. Match the abstract concepts with best matching label candidates. Let the matched label candidates become the cluster labels.
5. Prune similar cluster labels.
6. Determine cluster contents for each cluster label by using classic VSM approach.
7. Sort clusters according to calculated cluster scores.

Current Lingo Algorithm

A brief algorithm of the current Lingo is given below:

1. Preprocess documents
    - 1.1 Extract frequent phrases and single words as cluster label candidates.
    - 1.2 Determine the assigned documents for each label candidate.
    - 1.3 Filter out the label candidates that contain less number of documents than the minimum cluster size threshold.
2. Build the term-document matrix using the stems of the label candidates (except the stop words in the label candidates).
3. Reduce the term-document matrix to the term-abstract concept matrix according to the desired cluster count base threshold.
4. Match the abstract concepts with the cluster label candidates.
5. Select the cluster label candidates that matched with an abstract concept as the labels of the determined clusters.
6. Merge clusters that share higher percentage of documents than the cluster merging threshold.
7. Form the final clusters for presentation.

There are some major differences between the original and the current algorithm. We list the three important changes as follows:

- Default matrix factorization method used in the Cluster Label Induction was changed from SVD to NMF-ED, since the NMF-ED method performs best as showed in [7]
- Documents are not assigned to clusters using classic VSM approach anymore. Instead of the VSM approach, a binary similarity function is used to determine whether a document should be assigned to a cluster or not. A document is assigned to a cluster if the document contains the stems of all words in the cluster's label.
- Cluster merging phase was included before final clusters are formed. The clusters, which share common documents with a higher percentage than a cluster merging threshold, are merged.

Proposed Modifications for Current Lingo Algorithm

In order to enhance the low recall value of the Lingo algorithm, we propose two main modifications.

Cluster Content Discovery Phase: We propose a method which aims to provide that the documents, which do not include all of the stems of all words in a cluster's label whereas those are semantically related to it, could also be assigned to the related cluster. Our method requires a lexical database which can provide synonyms for a given word.

According to the proposed method, for a single word label candidate, the documents that include the stem of the word itself or a stem of at least one of its synonyms are assigned to the cluster label. As for phrase label candidates, the documents that contain the stem of actual word or a stem of at least one of its synonyms, for each word of the label, are assigned to the cluster. We employed the WordNet lexical database as synonym supplier component. Synonym set that is retrieved for a word includes all of the synonyms which can be members of any type and related to any sense.

Cluster Label Induction Phase: In Lingo, the abstract concepts that latently exist in the input document set are discovered by means of SVD [4]. Moreover, the frequent phrases are thought to be potentially capable of describing the abstract concepts [3]. Therefore, each abstract concept vector is matched with a frequent phrase which is a label candidate.

We propose a modification in the abstract concept-label candidate matching process. In the current Lingo algorithm, the matching process is performed by comparing the abstract concept and the label candidate vectors, which lie on the same term space, by using the cosine similarity function. Since the labels generally consist of a few words, their vectors are mostly so sparse. We noticed that comparison of abstract concept vectors with sparse label vectors might not be so successful in selecting correct labels for the abstract concepts. To overcome this, we firstly propose to enrich the label vectors, by including stems of the synonyms of the words in labels, if that stems are included in the term space.

We further propose a new abstract concept-label matching approach for Lingo. In the Lingo algorithm, the term-

document matrix is reduced to the term-abstract concept matrix (base matrix) and also the document-abstract concept matrix (coefficient matrix), via a selected factorization method. The latter actually can reveal the similarities between documents and abstract concepts. By using this information, we can determine the documents related to each abstract concept. Since we also have the assigned documents to each label candidate, we can use the number of common documents between the abstract concepts and the label candidates as a similarity measure. For each abstract concept, the top-most similar label candidate is matched. According to this approach, the number of final labels (clusters) could be less than the abstract concept number. For the proposed method, there are two options to determine the abstract concepts that a document should be assigned to:

- 1st Option: Assign a document to the abstract concepts that the document is similar with a higher score than a document-abstract concept similarity threshold.
- 2nd Option: Assign a document to the top-most similar abstract concept only.

For the first option, the coefficient matrix should be column-length-normalized prior to the assignment process.

EXPERIMENTS

An open source implementation of the Lingo algorithm is provided in the Carrot^2 [12], which is an open source search results clustering engine. The Carrot^2 is implemented in Java. The proposed methods were experimented on the Carrot^2 engine. To retrieve the synonyms from the WordNet database, the Java API for WordNet Searching (JAWS) [13] was used.

In our experiments, we used the AMBIENT (AMBIguous ENTries) dataset [14]. It contains 44 ambiguous topics that are selected from the disambiguation pages of Wikipedia. Each topic includes a set of subtopics and 100 ranked documents that were retrieved from a search engine (January 2008). In the dataset, the documents for each topic are matched with the subtopics; whereas some documents are not matched with any subtopics and some subtopics do not contain any documents. In our experiments, we compared our resulting clusters with the given clusters of AMBIENT and we used five different metrics such as contamination, precision, recall, f-measure and normalized mutual information (NMI) to evaluate the experiment results. The Carrot^2 engine provides the calculation of these metrics.

The contamination metric is used to evaluate the purity of the resulted clusters. Its weighted average value for the whole cluster set is calculated. When its value gets closer to zero, it means purer clusters are produced, and vice versa. For the precision, recall and f-measure metrics, the weighted average values are calculated, too. For each true cluster for a query, the cluster which achieves the best f-measure is selected from produced clusters, and then precision, recall and f-measure metrics are calculated. The weighted average values of these metrics are then calculated by using the size of the true clusters and their values. NMI metric is also used to evaluate the quality of the clusters. Its value will be 1, for a perfect clustering algorithm. We use the averages of the weighted average values over all topics in the dataset (TABLE 2 , Fig. 1 and Fig. 2).

Our experimental stage consists of 8 sequential steps, which we tagged them as from S1 to S8. The steps are listed in TABLE 1 with their definitions. For all steps, we used the default values for the parameters and thresholds, which are defined in the current implementation for Lingo.

In TABLE 2, the values of the experiment results are shown for each step. Fig. 1 shows the f-measure, precision and recall values and Fig. 2 shows the contamination and NMI values.

According to TABLE 2, it can be seen that S2 increases the f-measure and recall values, as it causes a slight decrease in precision, compared to S1. If S2 and S3 are compared, it can be seen that S3 produces purer and more precise clusters. According to these results, it can be said that employing synonyms in document assignment process can provide better clusters.

TABLE 1: DEFINITIONS OF SEQUENTIAL STEPS

Step	Definition
S1	The current implementation of the Lingo algorithm as of release 3.7.1
S2	The modification in cluster content discovery phase, for both single word and phrase label candidates, is employed
S3	The modification in cluster content discovery phase, for only phrase label candidates, is employed
S4	The modification used in S2 and the modification of enriching label vectors, in cluster label induction phase, are employed
S5	The modification in cluster label induction phase by using the first option with the similarity threshold value of 0.75 is employed
S6	The modification in cluster label induction phase by using the second option is employed
S7	The modifications used in both S3 and S5 are employed
S8	The modifications used in both S3 and S6 are employed

In TABLE 2, it is also shown that S4 cannot make any valuable improvement over S2. This shows that enriching label vectors does not enhance abstract concept-label candidates matching.

Fig. 1 shows that our proposed abstract concept-label candidates matching method is successful, since f-measure and recall values for S5, S6, S7 and S8 are dramatically higher than S1 and Fig. 1 also shows that using the second option, for the proposed matching method, leads slightly better results than using the first option. Moreover, it can be seen that including the modification in cluster content discovery phase provides an improvement over S5 and S6, in S7 and S8. As Fig. 2 shows, for S5, S6, S7 and S8, higher NMI and lower contamination values are achieved, compared to the S2 and S4.

The best improvement was achieved by applying the modifications for abstract concept-label candidate matching method by using second option and employing synonyms in document assignment process for the phrase label candidates (S8). Due to the size limitation, we further compare the more detailed results only for S1 and S8. Therefore TABLE 3, TABLE 5 – 8 and Fig. 3 show the related results for only S1 and S8. Moreover, the Beagle topic is selected randomly for the

comparisons.

In TABLE 4, the partitions (true clusters) for the Beagle topic are shown with their sizes and definitions. TABLE 5 - 8 show the values of f-measure, precision and recall and the labels of the best matching clusters, for each true partition. In addition, TABLE 3 shows the weighted average precision, recall and f-measure values for the topic.

From TABLE 5, 7 and 8, it can be seen that S8 significantly outperforms S1. Only for P2, S1 seems better in the results, but P2 can be seen as an outlier subtopic, since it contains only 2 of the total 86 documents. The improvements can also be seen in the TABLE 3, which demonstrates the weighted average values for the Beagle topic.

TABLE 2: AVERAGE VALUES OVER ALL TOPICS

Steps	Contamination	F-Measure	Precision	Recall	NMI
S1	0.260	0.630	0.908	0.554	0.651
S2	0.338	0.654	0.891	0.590	0.646
S3	0.281	0.638	0.906	0.566	0.646
S4	0.335	0.643	0.886	0.578	0.643
S5	0.291	0.730	0.884	0.688	0.691
S6	0.296	0.734	0.880	0.694	0.693
S7	0.298	0.741	0.880	0.703	0.692
S8	0.294	0.743	0.878	0.707	0.691

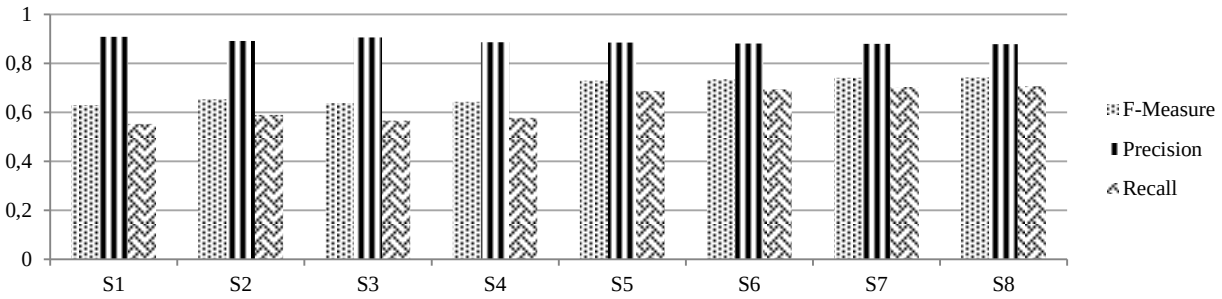


Fig. 1. Average F-Measure, Precision and Recall Values over All Topics

TABLE 3: WEIGHTED AVERAGE RESULTS OF THE SELECTED STEPS FOR THE BEAGLE TOPIC

Step	F-Measure	Precision	Recall
S1	0.351	0.988	0.221
S8	0.622	0.960	0.477

TABLE 4: TRUE PARTITIONS FOR THE BEAGLE TOPIC

Partition	Definition	Document Count
P1	Beagle is a dog breed	55
P2	HMS Beagle, the ship in which Charles Darwin undertook the travels during which he made many observations which became important for his formulation of his theory of evolution	2
P3	Beagle 2, a failed British Mars lander named after HMS Beagle. It crashed on 25 December 2003	11
P4	Beagle (software), a desktop search service for GNU/Linux users.	18

TABLE 5: RESULTS FOR THE P1 PARTITION OF THE BEAGLE TOPIC

	S1	S8
Label	Adopting	Breed
F-Measure	0.281	0.571
Precision	1.000	1.000
Recall	0.164	0.400

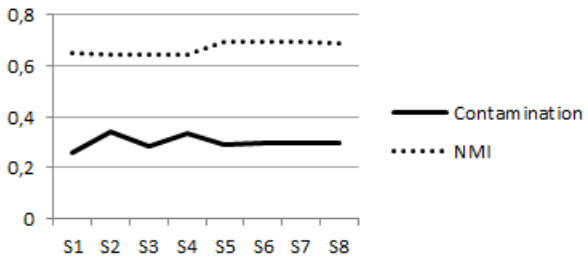


Fig. 2. Average Contamination and NMI Values over All Topics

TABLE 6: RESULTS FOR THE P2 PARTITION OF THE BEAGLE TOPIC

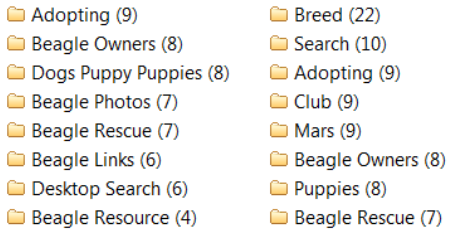
	S1	S8
Label	Project	Beagle Resource
F-Measure	0.500	0.286
Precision	0.500	0.200
Recall	0.500	0.500

TABLE 7: RESULTS FOR THE P3 PARTITION OF THE BEAGLE TOPIC

	S1	S8
Label	ESA's Mars Express	Mars
F-Measure	0.429	0.900
Precision	1.000	1.000
Recall	0.273	0.820

TABLE 8: RESULTS FOR THE P4 PARTITION OF THE BEAGLE TOPIC

	S1	S8
Label	Desktop Search	Search
F-Measure	0.500	0.643
Precision	1.000	0.900
Recall	0.333	0.500



(a) (b)

Fig. 3. The Labels of the Clusters for the Beagle Topic; (a) Generated by S1 and (b) Generated by S8

In Fig. 3, the lists of the top-8 produced clusters for the Beagle topic for S1 and S8 are demonstrated. According to it, while S1 lists two clusters, which match a true cluster, in 1st and 7th positions, S8 list three clusters in 1st, 2nd and 5th positions. These results show that the proposed methods outperformed the current Lingo algorithm.

CONCLUSION

We propose two main modifications for the Lingo algorithm, in order to eliminate the disadvantages due to the low recall values. First of them is to benefit from the synonyms for document assignments to the cluster labels. The other is to use the number of common documents between the abstract concepts and the label candidates as a new similarity measure to match the abstract concepts with the cluster label candidates. We experiment two alternatives to determine which documents should be assigned to which abstract concepts. One of them lets the documents be assigned only to one single abstract concept, while the other considers more than one abstract concept.

The experiment results demonstrate that our proposals for the Lingo algorithm lead quite better clusters, compared to the current algorithm. Despite the slight decrease in the precision values, our proposals make the recall and f-measure values increase dramatically.

As a future work, the merging process of the Lingo algorithm can be modified so that the recall values can be increased even more, without decreasing the precision and f-measure values.

ACKNOWLEDGMENT

Seyfullah Demir thanks to Stanislaw Osiński and Dawid Weiss who were always helpful and kind whenever he has questions or need advices.

REFERENCES

C. Carpineto, S. Osiński, G. Romano, and D. Weiss. "A survey of Web clustering engines.", *ACM Computing Surveys (CSUR)*, vol. 41, no. 3, pp. 1-38, July 2009

O. Zamir , O. Etzioni, "Grouper: a dynamic clustering interface to Web search results", *Computer Networks: The International Journal of Computer and Telecommunications Networking*, vol. 31, no. 11-16, pp. 1361-1374, May 1999

S. Osiński, "An Algorithm for Clustering of Web Search Results", M.S. thesis, Poznan University of Technology, Poland, 2003.

S. Osiński, J. Stefanowski, and D. Weiss, "Lingo: Search results clustering algorithm based on singular value decomposition.", *Intelligent Information Processing and Web Mining*, vol. 25, pp. 359-368, 2004

S. Osiński, D. Weiss, "Conceptual clustering using lingo algorithm: evaluation on open directory project data", *Intelligent Information Processing and Web Mining*, vol. 25, pp. 369-377, 2004

S. Osiński, D. Weiss, "A concept-driven algorithm for clustering search results," *IEEE Intelligent Systems*, vol. 20, no. 3, pp. 48-54, May-June 2005

S. Osiński, "Improving quality of search results clustering with approximate matrix factorisations", *Advances in Information Retrieval*, vol. 3936, pp. 167-178, 2006

N. C. Lang, "A tolerance rough set approach to clustering web search results", M.S. thesis, Faculty of Mathematics, Informatics and Mechanics, Warsaw University , 2004

A. Sameh, A. Kadray, "Semantic Web Search Results Clustering Using Lingo and WordNet", *International Journal of Research and Reviews in Computer Science (IJRRCS)*, vol. 1, no. 2, pp. 7-76, 2010

Princeton University "About WordNet." WordNet. Princeton University. 2010. http://wordnet.princeton.edu

G. Salton, "Automatic text processing: the transformation, analysis, and retrieval of information by computer", Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1989

S. Osiński, D. Weiss, "Carrot2: Design of a Flexible and Efficient Web Information Retrieval Framework", *Advances in Web Intelligence*, vol. 3528, pp. 439-444, 2005

Java API for WordNet Searching (JAWS), http://lyle.smu.edu/~tspell/jaws/

G. Romano, C. Carpineto, Ambient dataset, http://credo.fub.it/ambient/, 2008

Seyfullah Demir was born in Konya, Turkey. He received the bachelor degree in the Department of Computer Engineering in Hacettepe University, Turkey. Now, he is studying for Master of Science in the same department. He has been work for The Scientific and Technological Research Council of Turkey, since 2010. His research interests include information retrieval and software engineering.

Assoc. Prof. Ebru A. Sezer received the bachelor degree in the Department of Computer Engineering in Hacettepe University, Turkey in 1996. She gets the MSc and PhD degrees at the same department in 2000 and 2006. Her active research topics include information retrieval, data and web mining, fuzzy-logic systems and geographical information systems.

Prof. Dr. Hayri Sever is the head of the Department of Computer Engineering in Hacettepe University, Turkey. He is involved in several domestic and international research projects as a leader or consultant. He also has several international publications. His active research topics include information retrieval (and filter) models, vertical search engines, data and web mining, geographical information systems, and business process managements systems. His area of expertise includes database and information retrieval systems, software engineering, artificial intelligence, and internet computing.

ÖZGEÇMİŞ

Kimlik Bilgileri

Adı Soyadı	: Seyfullah Demir
Doğum Yeri	: Konya
Medeni Hali	: Bekar
E-posta	: seyfullahdemir@gmail.com
Adresi	: Nasuh Akar Mah. Ziyabey Cad.
	Yüksel Sitesi D. Blok No: 22
	Çankaya / ANKARA

Eğitim

Lise	: 2002-2005 Aksaray Fen Lisesi
Lisans	: 2005-2010 Hacettepe Üniversitesi Mühendislik
	Fakültesi Bilgisayar Mühendisliği Bölümü

Yabancı Dil ve Düzeyi

İngilizce	: İleri
İspanyolca	: Başlangıç

İş Deneyimi

2010 – Halen	: TÜBİTAK – BİLGEM – YTE
	Yazılım Teknolojileri AraştırmaEnstitüsü

Deneyim Alanları

C, C++, C#, Ada, Assembly, Objective-C, Java, GWT, JSF, JSP, Servlets, EJB, JPA, Hibernate, UML, SQL, Oracle, PostgreSQL, Microsoft Visual Studio, Eclipse, Xcode, Matlab, Git, Jira, HP Quality Center, Protege, Semantic Web, Information Retrieval

Tezden Üretilmiş Projeler ve Bütçesi

-

Tezden Üretilmiş Yayınlar

-

Tezden Üretilmiş Tebliğ ve/veya Poster Sunumu İle Katıldığı Toplantılar

Tebliğ: Modifications for the Cluster Content Discovery and the Cluster Label Induction Phases of the Lingo Algorithm

Yer : International Conference on Computer Science and Information Technology (ICCSIT 2013)