RAZIEH EHSANI

PhD Thesis

2018

KeNet: A COMPREHENSIVE TURKISH WORDNET AND ITS APPLICATIONS IN TEXT CLUSTERING

RAZIEH EHSANI

IŞIK UNIVERSITY

2018

KeNet: A COMPREHENSIVE TURKISH WORDNET AND ITS APPLICATIONS IN TEXT CLUSTERING

RAZIEH EHSANI
M.S., Computer Engineering, ISTANBUL TECHNICAL UNIVERSITY, 2013

Submitted to the Graduate School of Science and Engineering
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in
Computer Engineering

IŞIK UNIVERSITY
2018

IŞIK UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

KeNet: A COMPREHENSIVE TURKISH WORDNET AND ITS APPLICATIONS IN TEXT CLUSTERING

RAZIEH EHSANI

APPROVED BY:

Prof. Olcay Taner YILDIZ (Thesis Supervisor)	Işık University	_____
Prof. Ercan SOLAK	Işık University	_____
Prof. Fikret GÜRGEN	Boğaziçi University	_____
Prof. Tunga GÜNGÖR	Boğaziçi University	_____
Assoc. Prof. Mustafa Taner ESKİL	Işık University	_____
Assist. Prof. Nilgün GÜLER BAYAZIT	Yıldız Teknik University	_____

APPROVAL DATE: 07/06/2018

KeNet: A COMPREHENSIVE TURKISH WORDNET AND ITS APPLICATIONS IN TEXT CLUSTERING

Abstract

In this thesis, we summarize the methodology and the results of our efforts to construct a comprehensive WordNet for Turkish. Most languages have access to comprehensive language resources. Traditional resources like bilingual dictionaries, monolingual dictionaries, thesauri and lexicons are developed by lexicographers. As computer processing of languages gain popularity, a new set of resources become necessary. One such resource is WordNet which was initially constructed for English language in Princeton University. A WordNet contains much of the information contained in a classic dictionary, but it also contains additional relationship information. These relations go beyond synonym relation and give information about relations such as a word being"is-a" or "is-a-part-of" another. These semantic relations are used in many text analysis tasks. A WordNet also categorizes words under common concepts. These concepts are called as synsets. As a result of all these, WordNet is a comprehensive dictionary which is readable by the computers and a useful language resource for text analysis and other research based on human language.

In Turkish language, our WordNet is not the first. The previous WordNet is part of BalkaNet project which is a multilingual WordNet including Turkish and Balkan languages. BalkaNet contains only common words between these languages, as such BalkaNet does not contain all Turkish words and suffers from top-down constructing method disadvantages. BalkaNet project has not been updated or expanded in recent years.

In this work we construct a Turkish WordNet from scratch using a bottom-up method. In general there are two methods for constructing WordNets. Bottom-up method means that we create the WordNet from scratch while top-down approach uses other WordNets by translating them. We use Turkish Contemporary Dictionary (CDT) which is an online Turkish dictionary provided by Turkish Language Institute. Bottom-up approach has its own difficulties, since constructing a WordNet from scratch requires more resources and a lot of effort.

In this work, we extract synonyms from CDT and ask experts to match common meanings for pairs of synonyms. We developed an application which makes annotation step easier and more accurate. We also use two groups of annotators to measure inter-annotator agreement. We used some automatic approaches to extract semantic relations from Turkish Wikipedia (Vikipedi) and Vikisözlük. We processed CDT to extract candidate synonyms and used rule based approaches to find synonym sets. There is no thesaurus for Turkish, so as an application we construct a thesaurus automatically and measured accuracy with our manually constructed synsets. We named our WordNet "KeNet".

Finally, in this thesis we developed a novel approach to represent a text document in a vector space. This approach uses WordNet semantic relations. This part of thesis is an application of KeNet. We used our approach to represent text documents and implemented two different clustering algorithms over these vectors. We tested our method over Turkish Wikipedia articles, domains of which are labeled by Wikipedia.

Keywords:WordNet, Turkish NLP, Semantic, Text Analysis, Graph-based, Sense

KeNet: KAPSAMLI TÜRKÇE WORDNET VE METIN KÜMELEMEDE KULLANILMASI

Özet

Bu tez, kapsamlı bir Türkçe WordNet yapımının aşamalarını, zorluklarını ve son olarak da onu bir doğal işleme alanında uygulamasını özetliyor. Her dilin kendine özel dil kaynakları vardır, örneğin tek dilli sözlükler, iki dilli sözlükler, lugatnameler klasik dil kaynaklarıdırlar ve dilbilimciler tarafından geliştirirlirler. Bu kaynaklar genellikle bir dil kurumu tarafından desteklenir ve denetlenir. Günümüz bilgisayarların hayatımızın her alanına girmesi ile birlikte, dil kaynaklarının da bilgisayarlar tarafından okunabilirliği ve bilgisayar uygulamalarında kullanılabilmeleri için geliştirilmeleri bir gereksinim haline gelmiştir. Bu bilgisayar tarafından okunabilir kaynaklardan biri WordNettir, WordNet ilk kez İngilizce için Princeton Üniversitesinde geliştirilmiştir. WordNet klasik sözlüklerin özelliklerini taşımakla birlikte kelimeler arasında bazı anlamsal ilişkileri de içerir. Bu anlamsal ilişkiler eş anlamlılıktan öte, bir kelime diğerinin bir türüdür, veya bir kelime diğer kelimenin bir parçasıdır gibi anlamsal ilişkileri de içerir. Bu anlamsal ilişkiler yazı analizlerinde kullanılmaktadır. WordNet kelimeleri gerçek dünyadaki kavramlarına göre tek bir kümede toplar, bu kümelere synset denir. Sonuç olarak WordNet, kapsamlı ve bilgisayar tarafından okunabilir bir dil kaynağıdır ve yazı analizlerinde oldukça faydalı bir kaynaktır.

Türkçe için bizim çalışmamızdan önce kapsamlı olmayan bir WordNet geliştirilmiş. Bu WordNet, BalkaNet projesinin adı altında geliştirilmiştir. BalkaNet çokdilli bir WordNettir ve Balkan dilleri ve Türkçeyi içermektedir. BalkaNet aşamalar sırasında geliştirilmiş ve anlamsal ilişkiler eklenmiştir, fakat son yıllarda herhangi bir güncelleme yapılmamıştır.

Bu çalışma, sıfırdan Türkçe için bir WordNet yapımını anlatmaktadır. Genel olarak, WordNet yapımı için iki yöntem vardır, aşağı-yukarı yöntem ve yukarıdan-aşağı yöntem. aşağı-yukarı yöntem herhangi başka bir WordNeti çevirmeden veya kullanmadan sıfırdan ve sözlük kullanarak WordNet yapımıyla uğraşır, yukarı-aşağı yöntemde ise, sıfırdan yapmak yerine başka dillerde mevcut olan WordNetleri birebir çevirerek ve dahasında geliştirerek veyahut değiştirmeyerek WordNet yapımıyla uğraşır. Bizim Çalışmamız Türk Dil Kurumunun Güncel Türkçe Sözlüğünü kullanarak aşağı-yukarı yöntem ile WordNet yapımıdır.

Bu çalışma sırasında, TDK sözlüğünden eşanlamlı kelimeleri çıkartıp ve bir grup insana bu kelimelerin ortaklaşa paylaştıkları anlamları işaretlemelerini istedik. Bu işaretleme için geliştirdiğimiz bir yazılım kullanarak sürecin kolaylaşmasını ve hata payının düşürülmesini sağladık. Ayrıca Türkçe için herhangi bir eşanlamlılar sözlüğü mevcur olmadığı için, Türkçenin ilk eşanlamlılar sözlüğünü otomatik olarak oluşturduk. İşaretleyiciler arasında anlaşmayı ölçüp ve ayrıca otomatik oluşturduğumuz eşanlamlılar sözlüğünü elle işaretlenmiş eşanlamlılar kümelerile ölçtük.

Son olarak, bu çalışmada geliştirdiğimiz WordNeti Vikipedi makalelerini kümelemesi için kullandık. Bunun için öncelikle her yazı dosyasını bir vektöre çevirdik ve bunun için kendi özel yöntemimizi kullandık.

Anahtar kelimeler: WordNet, Türkçe doğal dil işleme, Yazı Çözümleme, Graph tabanlı çözümleme, Anlam

Acknowledgements

"All that is solid melts into air...", a period of my life is about to be completed, one that I often thought would never end. Dealing with words, dissecting them gave me a liking which, once upon a time, I found in writing. As with most things, this moment would never be possible without the help and support of many people. I have to start with my supervisor Professor Olcay Taner Yıldız who supported me with his positive thinking, smiles and his humble character. Next I would like to thank my advisor Professor Ercan Solak, his help goes beyond this thesis. I would like to thank to him for his kindness, wisdom, patience. Thanks to my jury members for their valuable feedbacks. Many thanks to Anssi Yli-Jyrä for all the hopes and motivations that he gave to me, I really appreciate his kindness and support. Next I would like to thank my family who always encourage me, thanks my father for his inspiring perspective of life and my mother for her seemingly unending caring and to my brothers, Araz, Aref, Babak for their all support and encouragement. Many thanks to splendid people at Işık University especially to Berke Özenç and my room-mates, Esin Tetik and Sevde Ceren Yıldız for their support and friendship.

Finally I would like to give thanks to my husband, who makes me feel precious, strong and special.

This study was supported by The Scientific and Technological Research Council of Turkey (TÜBİTAK) Grant No: 116E104

To those words ...

Table of Contents

List of Tables

List of Figures

List of Abbreviations

CDT	**C**ontemprary **D**ictionary of **T**urkish
AI	**A**rtificial **I**ntelligence
NLP	**N**atural **L**anguage **P**rocessing
POS	**P**art **O**f **S**peech
MA	**M**orphological **A**nalysis
MD	**M**orphological **D**isambiguation
WSD	**W**ord **S**ense **D**isambiguation
PWN	**P**irinceton **W**ordNet
MWE	**M**ulti **W**ord **E**xpression
FST	**F**inite **S**tate **T**ransducer
TLI	**T**urkish **L**anguage **I**nstitute

Chapter 1

Introduction

During the last decades, with the development of fast computers, many new fields have surfaced and have started to develop. A common goal is to automatize as much work as possible using computers for decreased drudgery and increased efficiency. From automatic translation among languages, to understanding the underlying sentiments, various ambitious goals motivate development in this area. Using a language is already a complicated task for humans; training a computer for understanding a language is a NP-complete problem in artificial intelligence (AI) [33]. Thus, natural language processing has become a popular field of computer science.

In linguistics, there are five major categories: phonology, morphology, syntax, semantic and discourse. Each of these fields still remain a challenge, but NLP also deals with high level language tasks. Machine translation, summarization, topic detection, information extraction are some of these tasks. In summary, NLP tasks are divided in two parts, first part is the basic level which deals with low level NLP like morphology or syntactic analysis, and the second part is high level NLP like machine translation which employs low level processes.

Although there are many works in natural language processing that deal with the language without considering linguistic properties, understanding those properties remain inevitably critical. The amount of low level processing involving linguistic properties also depends on the language itself, for example the word segmentation

problem is more nuanced in Chinese than in English. In this thesis, we will deal with Turkish which is a language with a complicated morphology. This property makes high level language processing difficult compared to English. It is very obvious that accuracy in each low level language processing task in Turkish effects other levels. For example, accuracy in morphological analysis stage effects syntactic analysis and that in turn effects the semantic stage.

1.1 Turkish language

Dealing with natural language processing in Turkish is not an easy task. Turkish is a member of Altaic language family with a complex morphological structure. This property of the Turkish language leads to vast amounts of different surface structures in texts. In a corpus of ten million words, the number of distinct words exceeds four hundred thousand, [44]. Many words in Turkish have more than one possible morphological roles in different contexts. One word in Turkish can be Noun and Verb at the same time. This morphological property leads to "Morphological Ambiguity".

"Adam kadını teleskopla gördü" is a Turkish sentence, which has two syntactic analyses : first is "adam, kadını teleskopla gördü" and second is "adam kadını, teleskopla gördü". The first one means "The man, saw the woman carrying the telescope", while the second one means "The man saw the woman, through the telescope". This type of ambiguity is called "Syntactical Ambiguity".

Multiple possible meanings of a word also causes ambiguity at the semantic stage. As an example, consider "onun ocağı söndü". In this Turkish sentence there are no ambiguity in morphology and syntax stages but this sentence can take two different meaning semantically. First is "his oven was turned off" and second is "his family has dissolved". This type of ambiguity is called "Semantic Ambiguity".

Ambiguity is a major problem in many fields of NLP. Unlike human mind which can handle ambiguities in language using its prior knowledge, a computer cannot

as easily deal with it. Generally ambiguity in a lexical form is divided in two parts: syntax and semantics. Syntactical disambiguation performs with a higher accuracy in proportion to semantic disambiguation. In natural languages lexical form of a word may have more than one meaning. These meanings may be sometimes fairly similar or completely different. A NLP task which deals with this problem is Word Sense Disambiguation (WSD). WSD is the process of determining the sense of a polysemic word. Nowadays, WSD algorithms use computer readable dictionaries to solve the problem. Generally, they use WordNets as a comprehensive reference and dictionary. Beside WordNet, semantically labelled data also have a wide use in WSD. Providing these data, especially in the languages which are less developed in NLP, is an arduous work. Turkish is one of these languages and in this thesis we worked on Turkish language.

1.2 WordNet

One of the primary aims of NLP is extracting semantics which is discovering the underlying meaning of a processed sentence. Dictionaries are indispensable tools for human language processing and many have been made accessible through online services. A word often has multiple definitions associated with it and hence human language processing goes through a disambiguation step. Likewise, disambiguating homonymous words is an important task in computational semantics and digital dictionaries provide the means. Two words are homonymous when having same form and reading but have two different meanings.

A word may have more than one sense. Also, a sense may be shared among different words. In NLP, finding the words that share a sense and identifying in which of their senses they mean the same thing is the task of WordNet construction.

A WordNet is a graph data structure where the nodes are word senses with their associated lemmas (and collocations in the case of multi-word expressions) and edges are semantic relations between the sense pairs. Usually, the multiple senses

corresponding to a single lemma are enumerated and are referenced as such. For example, the triplet

$$(w_2^5, w_3^7, r_1)$$

represents an edge in the WordNet graph and corresponds to a semantic relation r_1 between the second sense of the lemma w^5 and the third sense of the lemma w^7. The direction of the relation is usually implicit in the ordering of the elements of the triple. For synonymy, the direction is symmetric. For hypernymy, as a convention, the first sense is an hyponym of the second.

The most pervasive relation in a WordNet is the synonymy. We take two lemmas as synonyms if there is a linguistic context in which they are interchangeable [46].

WordNets provide semantic ontologies that are used as inputs to many automated document analysis tasks, such as summarisation and classification [38, 37, 71]. WordNet is a computational lexicon of a language based on psycholinguistic principles.

Constructing a WordNet is a labour intensive undertaking. Annotating in a WordNet requires lexicographical competency as well as familiarity with the modes of use of words in different domains.

Turkish is a rich language influenced by languages like Arabic and Persian in its evolution. During the last century and after language reforms, French and English languages also had an impact on Turkish. Turkish agglutinative morphology enables new words to be generated using suffixes. These are some of the reasons why a large pool of homonymous words exists. This property of Turkish language leads to semantic ambiguity and make computational semantics challenging. A traditional way of solving this problem is via WordNet, as conceptual dictionaries are much more useful than alphabetical ones.

We named our WordNet KeNet, using the two initial letters of "kelime" (word in Turkish). KeNet covers a much larger vocabulary, not limited to those shared with other languages, including those in BalkaNet and some common spellings

of words additionally. Existing WordNets usually take the words directly from Princeton WordNet (PWN) [47], translating them first if necessary. This approach is beneficial to reduce manual labour by reusing existing work; however it can cause severe restrictions on the constructed WordNet. Many such constructions are not maintained and became obsolete, and for many others the data itself is not publicly available.

1.3 Scope of the Thesis

In the rest of this thesis, we begin with a literature review on WordNets and their construction, especially BalkaNet as a Turkish WordNet. Next, we describe the language resource Contemporary Dictionary of Turkish (CDT) which we used to construct KeNet from scratch. Problems that we encountered during the use of this resource are also highlighted. In Section 3.2, we discuss KeNet structure and how we extracted synonyms from CDT and handled multi word expressions. Section 3.1 discusses manual annotation and shows some statistics of inter-annotators agreements. In Section 3.3, we show how we constructed synsets and some statistics of those synsets. Section 3.4 is about semantic relations and how we extracted them from our language resources. In Chapter 4, we describe some rule based approaches to construct a Turkish thesaurus automatically. In Section 4.2, we compare synonym sets of automatically generated thesaurus with a manually generated one. Also we compare KeNet synsets with BalkaNet synsets. Chapter 5 contains a brief review of clustering methods and text analyzing methods. In Chapter 6, we discus about preprocessing Turkish text and preparation data for clustering. Section 6.2 and Section 6.3 show steps of constructing a textual graph and the graph-based algorithms which we use on these graphs. We also show results of using these algorithms in example textual graphs. In Chapter 7, we introduce a novel approach to convert a text document to a vector and show our clustering results using Vikipedi articles using the vectors obtained in the previous chapter.

Chapter 2

WordNets in Other Languages

The first WordNet project was Princeton WordNet (PWN) which was initiated in 1995 by George Miller, [47]. Over time, PWN evolved to become a comprehensive relational representation of the word senses of English. Currently in version 3.1, the latest release of PWN, has 117,000 synsets and 206,941 word-sense pairs. A more detailed history and description of PWN is given in [23].

Shortly after the release of PWN, WordNets for other languages were constructed. Many WordNets for other languages use the leverage of PWN by translating its synsets and extending where necessary. Such a case is the Finnish WordNet which has the same number of sysnsets as PWN [40]. The version 3.0 of Polish WordNet, plWordNet is larger than PWN by about a thousand words, [53].

EuroWordNet (EWN) [69] is a multilingual WordNet developed for seven European languages which is based upon PWN and translates it to other languages. Japanese WordNet [30] translates PWN to Japanese covering about 62,832 synsets. They started with about 3500 core synsets from PWN and enlarged their WordNet by translating more frequent synsets in PWN. To improve efficiency, they also translated from Spanish and French WordNet, to Japanese WordNet.

Arabic WordNet [8] follows a similar approach to EuroWordNet and focuses on manually extracting sets of concept, and maximizing compatible relations between

pairs of WordNets. The mapping is done using PWN version 2.0. They use Arabic special morphological properties that carry semantic information, such as "performer", "the performed work", etc. to extend Arabic WordNet.

There exists other bilingual WordNets such as Catalon [6], Spanish [4], which follow a manual approach but perform automatic extraction from EuroWordNet. The set of base concepts are put to use as the starting point for the construction of the Catalan WordNet. Catalan WordNet contains Noun and Verb concepts represented in hierarchical structure.

The Persian WordNet [62] also uses an automatic approach based on PWN. They translated PWN to Persian using bilingual dictionaries. During translation one of two possible outcomes can happen. In the first case, Persian word may have only one translation in English and therefore has only one corresponding synset in PWN. For this, they take the synset directly from PWN. In the second case, if at least two translations correspond to a word, they calculate a score for each candidate based on semantic distance in their corpus. They have a success rate of %76 in ambiguous cases.

The current state of the development of various WordNets can be found on the website for Global WordNet Association [3].

For Balkan languages, BalkaNet [67] is the most comprehensive work up to date. It represent semantic relations in Balkan languages individually but each language uses the same database of vocabulary as a shared ontology and finally all WordNets are linked to each other.

For Turkish WordNet part of BalkaNet [7], the researchers automatically extracted synonyms, antonyms and hyponyms from a monolingual Turkish dictionary. A similar monolingual dictionary mining approach was recently used to extract hypernyms for Russian WordNet, [2].

In BalkaNet, developers started with a core set of words that are deemed to be common across several languages, which is considered a top-down approach.

Instead, we followed a bottom-up approach in our development. Rather than starting from a core set of lemmas and the relations among them, we started with the whole set of lemmas in a monolingual dictionary of Turkish. This is particularly in contrast to the approach used in BalkaNet as well as several similar WordNets. Starting from a common set for multiple languages as was done in BalkaNet poses problems for translating lemmas in one language to other. For example, a sense that is expressed using a single word form in one language can only be expressed using a non-idiomatic phrase in another language. For example, the word "payday" in PWN is translated as a phrase "maaş ödeme günü" (literally, salary paying day) in Turkish BalkaNet. This phrase is not a collocation to warrant its inclusion in a monolingual Turkish dictionary. To the best of our knowledge, no other WordNet construction efforts have used such a bottom-up approach as ours.

In the work detailed in the present thesis, we kept the data format compatible with that used in BalkaNet. Therefore, our final data can easily be used in extending BalkaNet to cover a larger number of synsets.

In our approach to WordNet construction for Turkish, we mined a comprehensive on-line dictionary of Turkish for synonym candidates. We then manually annotated the whole set of candidates to verify the synonymy and pair the particular senses of the verified synonyms. Thus, we obtained a graph where the nodes are senses and the edges are synonymy relations. We found the clusters in this graph and arrived at synsets. We compared the resulting set of synsets with an automatic thesaurus that we constructed as well as the smaller set of synsets obtained in BalkaNet. Although BalkaNet project is very useful for less studied Balkan languages, the common vocabulary usage is a big limitation. In Table 2.1 there are statistics about Part of Speech (POS) tag distributions in BalKaNet and KeNet. In Table 2.1 we show that our number of nouns (in KeNet) is roughly 6 times larger. Similarly, verb count in KeNet is about 11 times larger than that of BalkaNet. This is even a bigger difference than it seems, since they have verbs like "hareketsiz durmak" which is not an entry in CDT. BalkaNet contains many

POS tag	# of synsets in KeNet	# of synsets in Balkanet
Noun	66 266	10 370
Verb	25 170	2 359
Adjective	12 932	770
Adverb	2 587	40
Other	6 262	-
Total	113 217	13 499

Table 2.1: POS tag distribution of KeNet and Balkanet

verbs which are non-idiomatic in Turkish. KeNet adjective count is about 17 times bigger than BalkaNet adjective count. A big difference between adverb counts is another disadvantage of BalkaNet's top-down approach. Compared to BalkaNet, our approach yields a WordNet that is larger and more consistent. Moreover, as we detail in the rest of the thesis, our double manual annotation increases the reliability of the resulting synsets. We also mined Turkish Wikipedia for hypernym relations which increased the set of such relations obtained using only a dictionary.

Chapter 3

Manual WordNet construction

3.1 Lexical resource

The main lexical source for KeNet is the Contemporary Dictionary of Turkish (CDT) (Güncel Türkçe Sözlük) published online and in paper by Turkish Language Institute (TLI) (Türk Dil Kurumu), a public organization. Among other literary and academic works, TLI publishes specialized and comprehensive dictionaries. These dictionaries are often taken as authoritative references by other dictionaries. The online version of CDT contains 65944 lemmas. Although TLI publishes a separate dictionary of idioms and proverbs, CDT still contains some Multi-word expression (MWE) entries that have idiomatic senses. In Section 3.2.2, we discuss how we handle MWEs in KeNet.

The first edition of CDT was published in 1945. Since then, it has been revised and updated many times. Currently, CDT's 2011 print edition is in circulation. Its online version is revised more often.

In our work, we used a reduced snapshot of the CDT online. In following sections we will describe what kind of reduction we implemented on CDT.

The CDT has a pretty straightforward structure without any special markups. For example, synonyms are not marked up but are instead embedded within the sense definitions. As in following example shown, this causes a difficulty in separating synonyms between words inside sense definition.

Synonym candidates for "acı" (suffering) :

1. Ölüm, yangın, deprem vb. olayların yarattığı üzüntü, keder, elem
    - Feelings that come after events like death, earthquake, fire, grief, pain
2. çarpıcı, göz alıcı (renk)
    - Stunning, attractive (color)

In the first sense there is no delimitation between "earthquake", "fire" as a part of sense " Feelings that come after events like death,earthquake, fire" and synonyms "grif" and "pain". In the second sense when word "stunning" refer to color, there is no mark to delimit it. These problems lead to some problem during CDT processing. Homonyms of a word are enumerated under the same entry for the lemma. Most lemmas have no homonyms. The entries with the largest number of homonyms have 5. These are "bel" (sign, waist, sperm, spade, sound magnitude unit) and "bar" (a folk dance, pub, pressure unit, bitterness in mouth, stick).

The fields of an entry in CDT are as follows. Multiplicity and optionality of the field values are given at the end of each field description. Figure 3.1 also shows a screenshot of CDT.

```
<lexicon>
    <word name="ab" lex_class="isim, eskimiş" origin="Farsça">
        <meaning>Su</meaning>
    </word>
    <word name="aba" lex_class="isim, halk ağzında">
        <meaning>Abla</meaning>
        <meaning>Anne</meaning>
    </word>
    <word name="aba" lex_class="isim" origin="Arapça">
        <meaning>Yünün dövülmesiyle yapılan kalın ve kaba kumaş</meaning>
        <meaning>Bu kumaştan yapılmış yakasız ve uzun üstlük</meaning>
        <meaning class="sıfat">Bu kumaştan yapılmış olan</meaning>
        <meaning class="eskimiş">Dervişlerin giydiği abadan yapılmış, önü açık hırka</meaning>
    </word>
    <word name="aba güreşi" lex_class="isim">
        <meaning>Aba giyilerek ve bele kuşak bağlanarak yapılan bir tür güreş</meaning>
    </word>
```

Figure 3.1: A screenshot of CDT

- ALTERNATION: Indications of orthographic changes in suffixation. Optional.

- DOMAIN: Whenever an entry has a technical sense, its domain is given. Multiple, optional.

- DEFULT POS: Most common POS of the lemma. It can be empty for MWEs. It has one of the following values: verb, auxiliary verb, conjunction, postposition, common noun, adjective, pronoun, adverb, proper noun, exclamation. Multiple, optional.

- ORIGIN: Source language for loan words. It can be multiple valued for MWEs. Multiple, optional.

- CONTEXT: Indication of usages like argot, mockery etc. Multiple, optional.

- SENSES: An enumerated list of senses. Each sense might have its own POS and domain fields when they are different than those of the default.

In Table 3.1 we show frequencies of loan words from different origins and in Table 3.2 show other field values of CDT.

Even though the CDT is the main authoritative lexical resource in Turkish, it poses some difficulties when used for NLP tasks. Below we examine some of the issues we encountered in our WordNet construction.

Origin	# of lemmas
Arabic	6,044
French	4,920
Farsi	1,855
Italian	606
English	458
Greek	382
Total	14,400

Table 3.1: Loan lemmas distribution of CDT

Field	Possible values
domain	anatomy, anthropology, military, computer science, botanic, biology, geography, maritime, grammar, linguistics, theology, literature, economics, pedagogy, philosophy, physics, physiology, geometry, astronomy, zoology, law, geology, chemistry, mining, logic, mathematics, meteorology, architecture, minerology, music, psychology, cinema, sports, history, technical, commerce, theater, sociology, TV, medicine
POS	verb, auxiliary verb, conjunction, postposition, common noun, adjective, pronoun, adverb, proper noun, exclamation
context	mocking, argot, old usage, insult, popular, vulgar, metaphor, familiar, joking

Table 3.2: Field values for CDT

3.1.1 Sense granularity

CDT is prone to some of the common lexicographic problems that afflict many dictionaries. For WordNet construction, the most relevant is the proliferation of senses where the distinction among the senses are debatable.

For example, the senses of the word “yüz” (hundred) are given in CDT as follows.

1. The name of the number after ninety nine.
2. The name of the numerals 100 and C that denote this number.
3. Ten times ten, one more than ninety nine.
4. A word that, when used together with “times” and “fold”, exaggeratedly expresses the multitude of something done.

This particular entry is somewhat extreme but it highlights the problems of lexicography in CDT. The first three senses can easily be collapsed without any loss of precision into a single sense with a short definition "The number 100." Only the fourth sense is sufficiently distinct and similar to its use in English.

In contrast, the online Oxford dictionary [55] lists only a single sense with the definition "The number equivalent to the product of ten and ten; ten more than ninety; 100," thus practically collapsing the first three senses given in CDT. The finer distinctions among the senses of "hundred" are given as usages under its single main sense in the online Oxford dictionary.

The sense granularity of PWN and its effect on sense clustering tasks was investigated in [63, 49].

3.1.2 Productive derivations

Turkish is an agglutinative language with a highly productive derivational morphology. The derivations pose interesting problems for lexicographers. An important problem is to decide whether to include a derivation as separate entry in the dictionary. The cases where the derived form undergoes a sense drift away from the one that the derivational morpheme nominally entails are distinguished. If the drift is so large that the sense of the derivation can not be inferred from those of the root and the suffixes, then a new sense needs to be added to the dictionary.

There are about 40 highly productive derivational suffixes in Turkish. One particular example is the deverbal noun suffix -mA, with the semantics "the action of the verb". The CDT has about 5400 entries for deverbal nouns with suffix -mA where the definition has the single obvious sense of "the act of verb." Similarly, the CDT includes separate entries for causative and reflexive forms of verbs. For example, the CDT has an entry for the verb "sor" (ask) as well as separate entries for the deverbal noun "sor-ma" (the act of asking), "sor-dur" (cause to ask) and

"sor-ul" (be asked). Each of those entries has the single obvious sense which can be trivially inferred from the semantics of the root "sor" and the suffixes -mA (deverbal noun), -DIr (causative) and -Il (passive).

In parsing the dictionary, we had to decide whether to include these obvious productions with single senses as nodes in KeNet or leave them out to be dealt with derivational morphology. In the initial version of KeNet, we decided to keep these derived nodes as singleton synsets.

3.2 Processing the Dictionary

We use CDT to extract synonyms and other semantic relations to construct Turkish WordNet. In this chapter we discuss the preprocessing tasks needed to provide annotators with synonym candidates.

3.2.1 Synonym candidates

In this section we discuss extraction of synonym candidates from CDT. Our goal is to prepare the data for manual annotation. There is no thesaurus for Turkish and we use an automatic rule-based approach to extract synonym candidates from CDT. CDT definitions include synonyms in many cases. Although the synonyms are not specially marked, as we discuss in the Chapter 3.1, the structures of most definitions are consistent enough to enable the development of heuristics for automated synonym extraction. Synonyms are usually listed towards the end of a definition, separated by commas. In many cases, the definition itself is a single word or a multi word expression (MWE), yielding unambiguously a synonym candidate. In other cases, we slice the definitions at commas. We eliminate the slices that do not have entries of their own in the dictionary. What remains is a list of synonym candidate lemmas associated with a dictionary entry. Although in this way we get lemmas which are not word's synonym but part of definition, we try to solve this problem during manual annotation. After extracting synonym

candidates, we store these candidate pairs for manual annotation as detailed in Section 3.2.3.

3.2.2 Handling MWEs

Many dictionaries contain MWEs that are idiomatic to some degree. CDT is no exception. In pruning the dictionary, we had to decide which of these MWEs to keep and which ones to discard, possibly relegating them to a specialized graph of idioms and their usages.

For Turkish, of particular interest are the verbal compounds. Verbal compounds are formed by the combination of a (possibly case-marked) noun or adjective with one of the auxiliary verbs. Most common auxiliary verbs in Turkish verbal compounds are "etmek" (to do) and "olmak" (to be). CDT lists 14 verbs as having at least one sense in which it has an auxiliary function. Auxiliary verbs in Turkish are listed in Table 3.3.

Verb stem	Closest translation	MWE count in CDT
et	do	1227
ol	be	298
ver	give	88
gel	come	85
kal	stay	58
git	go	51
yap	do	45
geç	pass	43
getir	bring	30
göster	show	20
dur	stay, stand	11
kıl	render	5
yaz	write	2
eyle	do	1
Total		1964

Table 3.3: Auxiliary verbs in Turkish and their frequencies.

Moreover, verb compound formation is the main mechanism through which foreign verbs are borrowed in to the language. Basically, the infinitive form of the

borrowed word in the foreign lexicon is compounded with "etmek" to construct the MWE infinitive form. Examples are, "tasnif etmek" (from Arabic, "tsnyf", to classify), "lanse etmek" (from French, "lancer", to launch), "dizayn etmek" (from English, to design).

Such MWEs appear as lexical entries in CDT. They also commonly appear in sense definitions. In mining for synonym candidates in CDT definitions, we included a MWE as a synonym candidate only if it has a dedicated entry and one or more senses are listed.

CDT also includes MWE templates as separate entries. An example is the entry "... duygusu uyandırmak", (literally, "... the feeling-of to wake", meaning, "to arouse a feeling of ..."). There are about 50 of such template entries. We discarded these in our construction as it not possible the encounter them in template form in any sense definition.

3.2.3 Manual Annotation

As a first stage, we constructed an initial set of synsets automatically. An example subset of KeNet is shown in Figure 3.2. From CDT dictionary, we extract all possible meanings of words. Each possible meaning of a word is considered as a synset. The constructed synsets have the following properties represented as tags:

- ID: Each word meaning gets a unique ID as synset ID, so that each synset is associated with only one meaning. An example ID is, "TUR10-0000030", where "TUR10" represents the version of the Turkish WordNet and "0000030" represents the index number of the synset (Line 12 of Figure 3.2).
- SYNONYM: If more than one word has the same meaning, these words and their meaning indexes are stored in a SYNONYM tag. Each SYNONYM tag is composed of:

- LITERAL: The name of the word whose meaning is the same as the current synset. For example, the first literal of synset with ID "TUR10-0000050" is "aba" (Line 14 of Figure 3.2).
- SENSE: The sense index of the word. The sense indexes of each word start from one and are incremented by one for each different meaning of the word. For example, the first literal of synset with ID "TUR10-0000050" is the second sense of "aba" (Line 14 of Figure 3.2).

- POS: Part of speech tag of this synset. 'n' stands for nouns, 'a' stands for adjectives, 'v' stands for verbs, and 'b' stands for adverbs. For example, the POS tag of synset with ID "TUR10-0000080" is 'a' (adjective) (Line 17 of Figure 3.2).

Figure 3.2: A screenshot of the KeNet in XML format

There are some other keywords which determine word type or usage domain such as "mathematics", "metaphor", "slang", etc. We assign categories to a word by scanning the definition of the word in the CDT dictionary. The category of a word is represented as a semantic relation in KeNet. A semantic category in KeNet is represented with SR tag and contains the following information in order: (i) synset id of the semantic category, (ii) type of the semantic relation, i.e. CATEGORY. For example, the synset with ID "TUR10-0000120" is in the semantic category Metaphor, which has the synset ID "TUR10-0531320" (Line 21 of Figure 3.2).

All possible semantic categories and their distributions in KeNet are shown in Table 3.4.

Category	Synsets	Category	Synsets	Category	Synsets
Mathematics	655	Sport	656	Music	637
Botanic	2164	Plural	1101	Marine	559
Theology	582	Zoology	1621	Metaphor	1316
Astronomy	372	Geography	363	Grammar	700
Physics	901	Philosophy	759	Medical	783
Economy	372	Law	561	Anatomy	665
Business	255	Pedagogy	81	Technology	153
Literature	450	Cinema	185	Television	112
Technical	144	Sociology	295	Biology	395
Geology	269	Informatics	54	Physiology	24
Psychology	331	Military	554	Theater	139
Geometry	32	Logic	142	Architecture	170
Mineralogy	153	Slang	612	History	637
Chemistry	865	Meteorology	42		

Table 3.4: Semantic categories and their distributions in KeNet

3.2.4 Special Cases

We mentioned in Chapter 1 that Turkish is influenced by other languages. One of these influences is different 'A' types. First type is normal 'A' and second type is 'Â'. 'Â' is used to indicate the consonant before 'A' is palatalized, as in **istiklâl** (independence). It is also used to indicate /a:/ in words where the long vowel changes the meaning, as in **adet** (pieces) and **âdet** (tradition) or **hala** (aunt) and **hâlâ** (still). In many new Turkish texts second type is removed. We put both spellings of 'A' in KeNet.

KeNet also has synsets representing general categories. Assigning a single synset for all these categories will cause a significant loss of semantic information, hence we represent them using a separate synset for each. These categories are: Proper noun (Line 1 of Figure 3.2), time (Line 3 of Figure 3.2), date (Line 4 of Figure 3.2), hashtag (Line 5 of Figure 3.2), e-mail (Line 6 of Figure 3.2), number (Line 7 of Figure 3.2), percentage (Line 8 of Figure 3.2), fractional number (Line 9 of

Figure 3.2), range (Line 10 of Figure 3.2), and real number (Line 11 of Figure 3.2).

In processing CDT, we sliced the sense definitions at the commas. Thus, we expect to find synonym literals among the slices. Of course, not every slice is a synonym. We used the following procedure to manually select the synonym sense when present.

Let $C(l)$ denote the set of such slices extracted from the sense definitions of lemma l. Let $S(l)$ denote the set of sense definitions of lemma l. For each l_i in CDT and for each j such that $l_j \in C(l_i)$, we present the human annotator the sets $S(l_i)$ and $S(l_j)$ as two lists. The annotator picks one sense from each, thus creating a synonym sense pair. The annotator may choose not to pair any of the senses. This means that the annotator judges that the lemmas l_i and l_j are not synonymous in any of their senses.

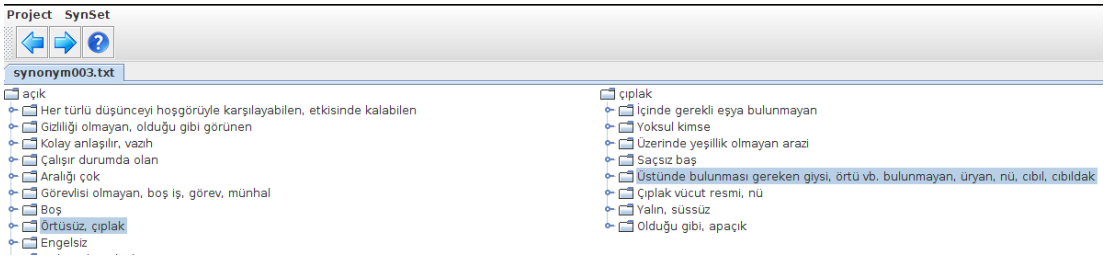

Figure 3.3: A screenshot of the synset reduction tool

Figure 3.3 shows a screenshot of the synonym pairing tool that we constructed for the manual annotation task. It shows the sets S(açık) and S(çıplak) for lemma "açık" and 'çıplak." In this particular case, the annotator paired 8th sense of "açık" with 5th sense of "çıplak." This means "çıplak" and "açık" are synonym because they share a common meaning and annotator chose this common meaning from sense lists. Because of sense granularity problem which we mentioned in previous Chapter 3.1 sometimes there are more than one common sense and the annotator had to decide to choose one of them. We will show an example of this problem in the following sections.

Note that, 8th sense of “açık” has “çıplak” as a comma separated slice and that is the reason why these two sets are shown the annotator. We instructed the annotators to disregard this bit as a pairing clue and use their own linguistic competence in deciding which senses to pair or whether to pair at all.

We annotated each pair of synonym candidate lemmas twice using two different annotators. There were a total of 9 annotators. Each annotator was given a different segment of the dictionary. The annotators were native Turkish speakers in their senior university years.

Then we determined the set of pairs where two annotators disagreed. An expert annotator went over this disagreement set and re-annotated the pairs, not necessarily agreeing with one of the annotators. In the case of agreements, the expert did not modify the pair. The expert annotator is the author of the present thesis, a native speaker of Turkish.

The total number of annotated pairs is 49 774. Of these, 42 615 had annotator agreements. In 92.15% of the pairs with disagreements, the expert annotator agreed with one of the annotators. For the rest, the expert chose a different pair which we accepted as the authoritative choice.

The annotation tool that we used in available for download in KeNet webpage [22].

3.2.5 Inter-annotator agreement

After manually pairing lemmas with their matching senses, we collect the pairs to form maximal synsets. First, we decided on a procedure on how to treat the cases where annotators disagreed.

Table 3.5 gives the statistics of the inter-annotator agreement statistics for the pairs. A and B denote the annotators and E denotes the expert.

	A & B	A & E	B & E	E only	Total
# of pairs	42 615	1 759	4 838	562	49 774
agreement percentage	85.62	3.53	9.72	1.13	100

Table 3.5: Inter-annotator agreement statistics.

In constructing the synsets, we took an edge to be valid if either both annotators marked the edge or the expert marked the edge. First condition corresponds to the first column in Table 3.5. The second condition corresponds to its next three columns.

In order to measure the inter-annotator agreement against agreement by change, consider two lemmas l_i and l_j presented to two annotators for pairing. Let $S(l_i)$ and $S(l_j)$ be the sets of senses of the lemmas l_i and l_j, respectively. The probability of chance agreement for this pair is given as $p_c(i,j) = 1/(|S(l_i)||S(l_j)|+1)$. Averaged over all pairs we obtain the probability of chance agreement as $p_c = 0.28$. Reading off the agreement probability from the first column of Table 3.5 as $p_a = 0.85$, we calculate the kappa measure as

$$\kappa = \frac{p_a - p_c}{1 - p_c} = 0.79.$$

We illustrate a common source of disagreements with an example. For the lemma "akbaba", the CDT gives 3 senses. The definition for the first sense is the description of the animal "vulture." The second sense definition is a single word, "ihtiyar" (elderly, old person). The last sense definition is the phrase, "çıkarı için başkalarını sömüren" (someone who exploits others for his/her own benefit) The annotators are asked to pick a pair of senses from 3 senses of "akbaba" and 5 senses of "ihtiyar". Among the sense definitions of "ihtiyar", two are quite close: "old person" and "elderly". While one annotator chose the first, the other annotator chose the second sense, creating a disagreement. Thus, the similar senses of a lemma is a common source of disagreement.

3.3 Synset construction

After manual annotation and verifying the pairs, using inter-annotator agreement, we want to find synsets from them. A synset is a set of synonyms which can be used interchangeably in a context without changing the meaning. It may consist of a single member, or more. A synset is sometimes also called a synonym set. In this chapter we use synset as set of meanings.

The simplest synset construction method is to find the connected components of the graph where the nodes are senses and the edges are the pairs of senses marked by the annotators as matching. Such a simple construction assumes that all the edges have the same confidence levels. Actually, the edges differ in their confidence strength. If the two teams agree on a pair, the confidence level is high. On the other end of the scale, if they disagree and the expert annotator chooses a pairing different than both, the confidence level is lowest. In between, the expert annotator might concur with one of the teams.

3.3.1 Synset statistics

Once we have the pairs of senses matched by the annotators, we constructed the synsets by finding the connected components of the undirected graph where the nodes are the senses and the edges are the manual sense pairings. In Figure 3.4, we give the distribution of the sizes of synsets.

The synset sizes in Figure 3.4 follows a Zipfian distribution, which is typical in linguistic observations [73]. Note that most of the senses are singletons. There are 49 361 synsets with only a single sense. Also, there is a huge synset with 7 906 senses. Obviously, this can not represent the true state of the sense relations in Turkish. In the rest of this section we will describe this problem and its possible solutions.

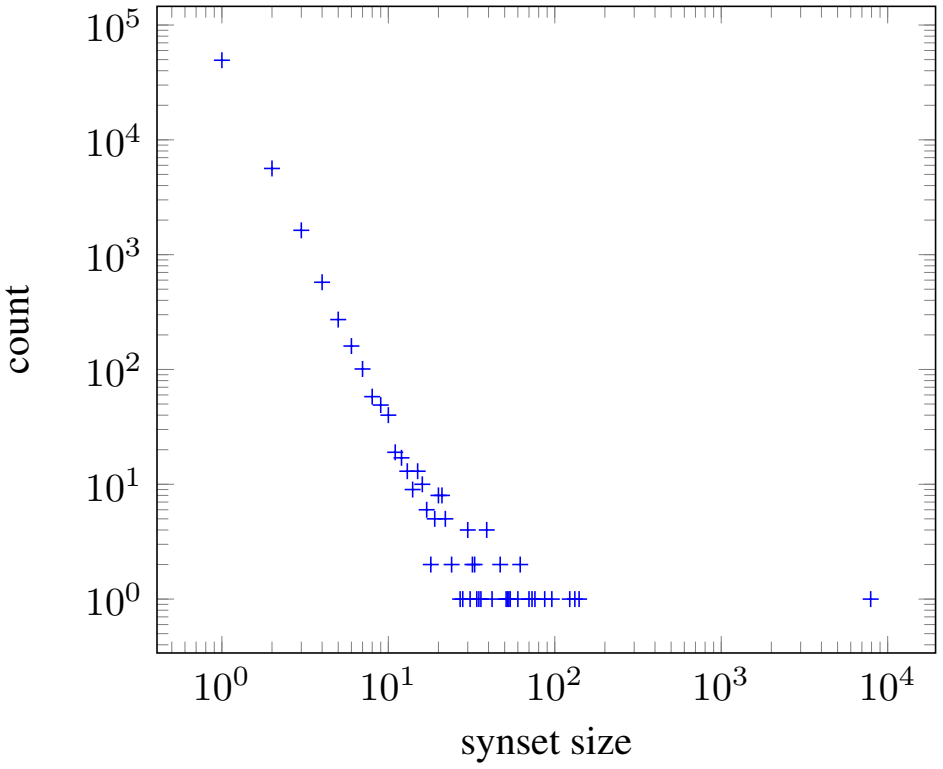

Figure 3.4: The distribution of synset sizes.

There are a few reasons for obtaining such a big synset. The main one is the sense drift introduced in the definitions of CDT, see Chapter 3.1. In CDT, often a definition for a sense cites lemmas with close senses. In some cases, this is done to confine and illustrate the sense by providing several close senses and emphasizing their intersections. However, taken in isolation, each such sense represents a small drift away from the original sense. When the annotators are presented such explanatory senses as synonym candidates, they tend to mark them as synonyms. When done in tandem, this drift creates the huge artificial synset that we observe above.

There are other big synsets due to the sense drift. The second biggest synset has 140 lemmas. The huge difference between the sizes of the largest and the second largest synsets indicates the presence of a property that joins smaller synsets. In terms of the graph structure, there are edges that connect small, densely connected sub-graphs. In order to explore this issue further, we used random walk based partitioning of the synonym candidate graph, [41]. We recursively

re-clustered the synsets that are larger than 100 until all the synsets are smaller than 100. The final distribution of synset sizes are given in Figure 3.5. The figure displays the sizes only for synsets which are not singletons.

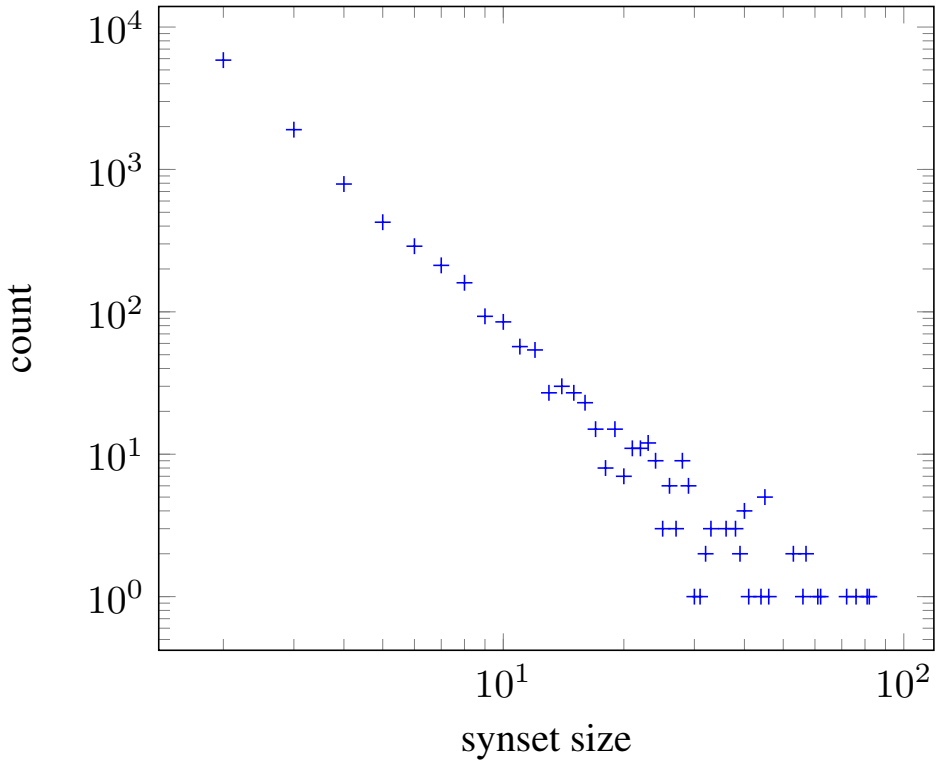

Figure 3.5: The distribution of synset sizes after recursive partitioning with random walk.

In Figure 3.6 we see two big sub-graph which are loosely connected. It's obvious that more popular adjectives in Turkish like "iyi", "kötü", "çok" appear in many definitions and cause more connectivity between senses. Before we use random walk for clustering we tried to remove such kind of adjectives which have high degree in the graph of synset. We list high degree nodes inside synset graph and remove these adjectives or words which lexicographers prefer to use to define word meanings.

3.4 Semantic relations

To investigate relations among synsets in KeNet, we confined the set of relations to the following three: antonym, hypernym and domain. We determined the

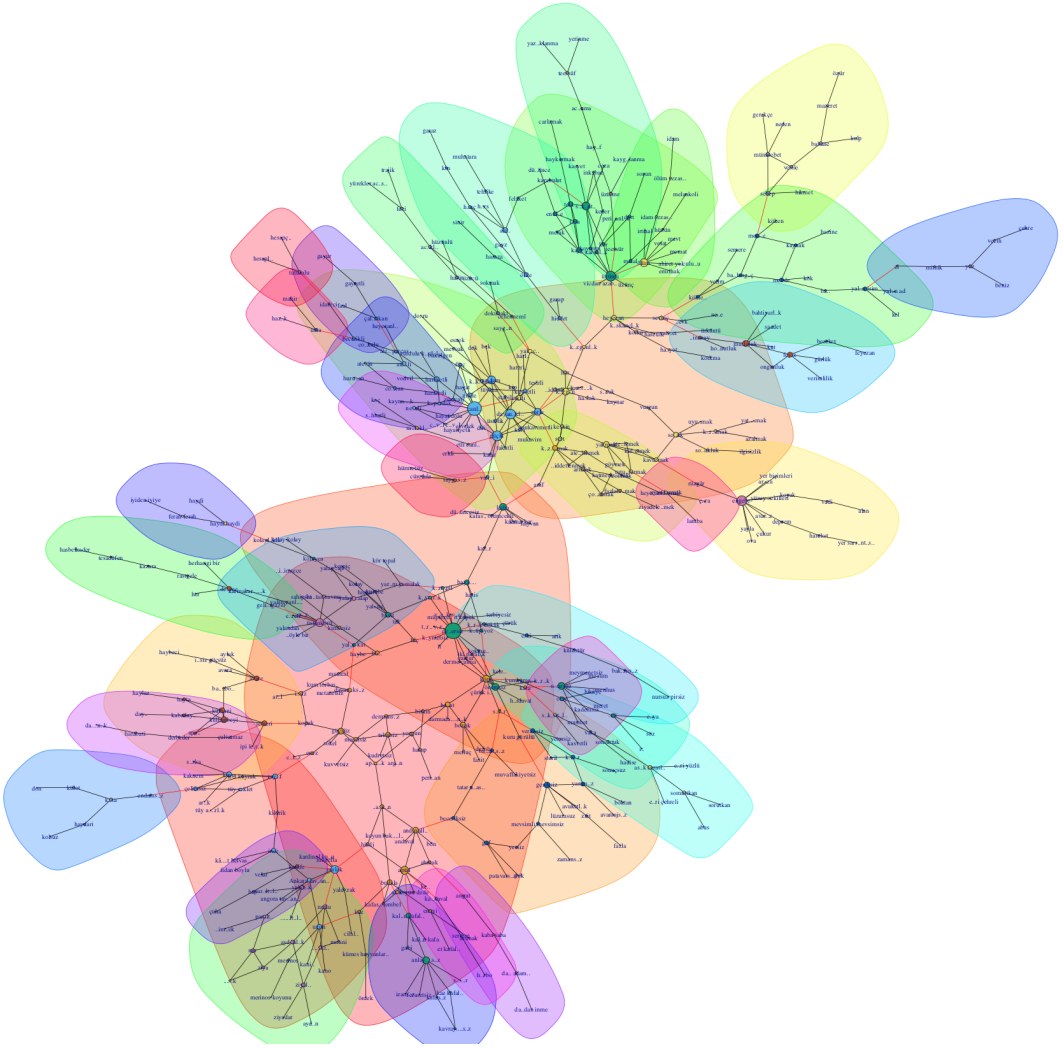

Figure 3.6: Random walk over on a big synset

candidate pairs for these relations by rule-based processing of CDT and Turkish Wikipedia (Vikipedi) [59]. Human annotators reviewed the automatic results and eliminated the false and ambiguous candidates. Table 3.6 summarizes the results.

Relation	Source	Count
Antonym	CDT	376
Hypernym	CDT	1420
Hypernym	Wikipedia	2764

Table 3.6: Statistics for the semantic relations.

Since hypernym and hyponym relations are inverses of each other, we detect them

simultaneously. So, if we detect a pair (a, b) as a being hypernym of b, we take that we also detected b as hyponym of a.

The rule-based search generates a list of candidate pairs between the lemmas, not senses. When reviewing the candidates, the human annotators determined the pair of particular senses for which the relation holds.

In the rest of this section we give the details for each type of rule-based search.

3.4.1 Antonyms

There are three categories of antonyms in lexical semantics, [25]. Gradable antonyms are the pairs that represent the opposite ends of continuous scale. For example "young" and "old" represent two ends of age scale. This type of antonym relations are most common in adjectives and nouns. Complementary antonyms, on the other hand, do not share a common scale but rather they represent the replacement of a parameter with its opposite. Commonly the parameter is direction as in (rise, fall), (enter, exit) etc. Such an antonymy is common with verbs. The last category is the relational antonyms where the antonyms represent the participants in the restricted context of a binary relationship as in (husband, wife).

The first two categories represent a quality of definability in terms of one sense defining its antonym. So "young" can be defined as the opposite of "old" but "husband" cannot be be so easily be viewed as the opposite of "wife."

We use this defining characteristic of first two antonym categories in searching for antonym candidates within CDT. We search patterns that represent opposition in sense definitions. The most common opposite pattern in CDT is

C karşıtı (opposite of C)

The agglutinative nature of Turkish enables the derivations that indicate negations as well as the presence or absence of an attribute. An example is

saygılı (respectful)
saygısız (disrespectful)

Both words appear as CDT entries. However, we do not include such pairs in our list of antonyms as the semantics of the relationship resides in the morphology rather than in the base lexicon.

3.4.2 Hypernyms and hyponyms

In this Section we describe how we find hypernym-hyponym relations. As an example "anchovy" is a kind of "fish" and actually "anchovy" is hyponym of fish and "fish" is a hypernym of "anchovy". In fact, hypernym-hyponym relation is a hierarchical relation and we can show this relation in a tree based structure 3.7.

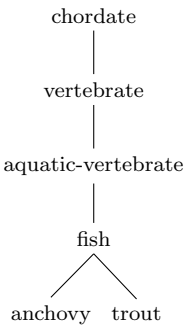

Figure 3.7: Example for Hypernym and Hyponym relation

3.4.3 Hypernym-hyponym in CDT

BalkaNet employs a rule based method to extract hypernym and hyponym relationship from Turkish dictionary. They use keywords like "kind of" ("bir çeşidi", "bir tür"), "-giller" to detect hypernym and hyponym cases, [7]. Another approach merges rule-based methods with statistical tools using a large corpus, [72]. Our set of keywords are similar to that of Bilgin et.al. [7], but additionally we also use "ve benzeri" and "ve diğeri" to find hypernyms inside CDT.

Finding hypernym relations in a descriptive text is easier than it is for hyponyms. In a sense definition in CDT, hypernym sense is often referred to with the assumption that the more abstract sense of the hypernym would be more readily available in the mental lexicon of the reader. Definitions often describe the peculiarity of the present entry and towards the end mention the hypernym. For example the definition of "flamingo" in CDT is "Leyleksigillerden, tüyleri beyaz, pembe, kanatlarının ucu kara, eti yenir bir *kuş*"
"An edible *bird* from the stork family with white and pink feathers and black wing-tips." In Table 3.6 we show some statistics about hypernym-hyponym relation in CDT.

Table 3.7: Example patterns for hypernym candidates.

pattern in Turkish	pattern in English
SUP-A verilen genel (ad,isim)DIr	is the general name given to SUP
bir SUP-DIr	is a SUP
SUP kavramlarından birisidir	is one of SUP concepts
SUP (çeşidi, türü, birisi)DIr	is a (kind, one of) SUP
SUBlArIn (bütünü, tümü)dür	is the whole of SUBs

3.4.4 Hypernym-hyponym in Vikipedi and Vikisözlük

Beside CDT we also use "Vikipedi" and "Vikisözlük" to extract hypernym-hyponym relations. Both of these are encyclopedias and contain descriptive texts. We take 65 000 words from CDT and look for its Vikipedi entry. There are only 9 624 entries in Vikipedi. We observed that hypernyms are in definition part of text, but there is no special structure which show the definition. Firstly we try to recognize the definition part. Normally, definitions finish with copula, we take sentences before "dIr" copula. After this step we take 6 100 entries which have copula in the first paragraph.

For the next step, we use some patterns to detect hypernym-hyponyms as we did in Section 3.4.3. There are variations to the patterns of referring to the hypernyms. SUP refers to the hypernym in the pattern and SUB refers to a hyponym. Some examples of patterns are listed below :

- SUB . . . SUP verilen genel (addır, isimdir). (is the general name given to SUP)
  - Rupi, bazı Asya ülkelerinde kullanılan paralara verilen genel addır.
    SUP: para, SUB: rupi
- SUB . . . bir SUPDIr. (is a SUP)
  - Bezelye, baklagiller familyasından tırmanıcı bir bitkidir.
    SUP: bitki, SUB: bezelye
- SUB . . . bir SUP çeşididir. (is a (kinds one of) SUP)
  - Kadayıf, isim benzerliğine karşın tel kadayıftan çok farklı birer tatlı çeşididir.
    SUP: tatlı, SUB: kadayıf
- SUB . . . SUPlerden birisidir. (is one of SUP concepts)
  - Frenk maydanozu, genelde mutfakta hafif yemeklere aroma vermek için kullanılır ve Fransız mutfağında aromatik bitki karışımının içindeki bitkilerden birisidir.
    SUP: bitki, SUB: frenk maydanozu
- SUB . . . SUP (bütünüdür, tümüdür). (is the whole of SUBs)
  - Tedavi, sağlığı bozulmuş olan bireyi sağlıklı du- ruma kavuşturma amacıyla yapılan tıbbi işlemler bütünüdür.
    SUP: işlem, SUB: tedavi
- SUB . . . SUPlArın (bütünüdür, tümüdür). (is the whole of SUBs)
  - Dekor, bir oyun sırasında sahnede kullanılan ve oyunu tamamlayan aksesuarların tümüdür.
    SUP: aksesuar, SUB: dekor

Here detecting hyponyms in sense definitions is more difficult as they rarely refer to subclasses. Still, since hypernym/hyponym relations are inverses of each other, we generate hyponym relations wherever we detect hypernyms.

Vikisözlük has a different structure than Vikipedi and its structure is similar to a dictionary. As an example, Vikisözlük contains "synonym" and "category" tags. We try to find entries for 6 500 words from CDT and we find only 11 410 entries inside Vikisözlük. Under tag "category" sometimes hypernym of word is written but sometimes domain of word is written. For example for word "football", under "category" tag there is "sports" when it should be "game". We pick 3 075 domains such as "sentence", "expression", "name", "person", etc. After this filtering we have 8 335 word definitions and from these we finally get 718 hypernyms. There are some examples which are listed in Table 3.8.

Hyponym	Hypernym
abanoz(ebony)	bitki(plants)
alüminyum(aluminum)	element(element)
altıparmak(six-fingered)	bitki(plants)
anakonda(anaconda)	yılan(snake)
antilop (antelope)	boynuzlugiller (hornlugs)
arapça (arabic)	dil(language)
gül (rose)	çiçek(flower)
mango(mango)	bitki (plants)
üzüm(grapes)	meyve (fruit)
altıntop (grapefrui)	bitki (plants)
balina (whale)	memeli (mammal)

Table 3.8: Hypernym examples in Vikisözlük

3.4.5 Domain

Domain is the field that a word is used in. For example "football", is a term which is used in "sports" field. Domain gives us useful information about a word. In CDT domains are tagged under "domain". We extract domain from CDT as we mentioned in Section 3.4.4 and we also extract some domains from Vikisözlük. Some of the important extracted domains and their numbers are shown in Table 3.9.

Domains	Numbers
Tıp (Medicine)	291
Din (Religion)	248
Müzik (Music)	237
Matematik (Mathematics)	242
Coğrafya (Geography)	202
Edebiyat (Literature)	191
Hukuk (Law)	188
Toplum bilimi (Community knowledge)	179
Biyoloji (Biology)	176
Anatomi (Anatomy)	158
Akrabalık (Kinship)	53
Eğitim (Education)	52
Asker (Military)	40

Table 3.9: Domains from CDT and Vikisözlük

Chapter 4

Automatic WordNet Construction

One of the most powerful tools in information access services is thesaurus. A thesaurus contains information about words that can be used for alternative synonym words. For example, checking synonymous terms can be beneficial for a search engine when the initial query does not yield results. However creating a thesaurus requires expert human labour. Some thesari are constructed using statistical methods. Statistical properties of co-occurance and other features are used in constructing such thesauri automatically, [50], [64]. These type of thesauri are called co-occurrence thesauri and although the associations are not necessarily semantically as strong as the ones in handcrafted thesauri, they are often useful in encoding the semantic structure of the text, which can not easily be detected manually [66]. Often these thesauri are generated for use in information retrieval from unstructured documents [15] [14].

In order to compare the performances of our manual synset construction procedures, we constructed a synonym thesaurus using a fully automatic, rule-based processing of CDT dictionary. Our aim in this comparison is to see how necessary it is to manually annotate the synonym candidate pairs extracted from the dictionary. We confine the comparison to the lemmas for which the CDT lists a single sense.

In this Chapter, we give the details of automatic construction. The details of the comparison are given in Section 4.2.

4.1 Automatic thesaurus

Let $S(w)$ denote the definition of the ith sense of the entry for lemma w as given in CDT.

Let R denote a deterministic rule that generates the list of candidate lemmas from a given sense definition in the dictionary. An example rule would be to slice the definition at commas and keep only the slices that are cited as the dictionary entries. For illustration, consider the definition of the first sense of the lemma "boğaz" (neck) given in CDT as "Boynun ön bölümü ve bu bölümü oluşturan organlar, imik, kursak"

For every entry in the dictionary, we define the set $C(w)$ of candidate synonym lemma for the lemma w as

$$C(w) = \{v|\ \exists i, v \in R(S(w))\}.$$

Namely, $C(w)$ represents the candidate lemmas obtained through running rule R over all sense definitions of w.

We next define the notion of strong synonymy with respect to a dictionary D and rule R as follows.

Definition 4.1. Literals w_1 and w_2 are strongly synonymous with respect to dictionary D and rule R if

$$w_1 \in C(w_2) \wedge w_2 \in C(w_1).$$

Note that we do not distinguish among different senses of a lemma and instead, consider all the synonym candidate lemmas collected across all senses.

This definition of synonymy is very restrictive. It requires the lexicographer to be complete when they write the sense definitions and include all synonym candidates symmetrically.

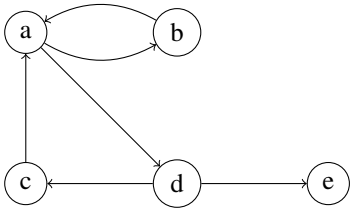

Figure 4.1: The synonym candidacy relations on the word graph.

A weaker definition allows longer cycles in mapping lemmas to synonym candidates. For this, we first define the longer synonym candidacy relation among lemmas. Let us define the set of n-synonym candidates $C_n(x_0)$ of the lemma x_0 as

$$\begin{aligned} C_n(x_0) = \{x_n| \ \ \exists x_1, x_2, \ldots, x_{n-1}, \text{ where } x_i \neq x_j, \ 1 \leq i, j < n, \\ x_1 \in C(x_0), x_2 \in C(x_1), \ldots, x_n \in C(x_{n-1})\}. \end{aligned} \tag{4.1}$$

Combining all the paths up to length n, we define the weakly n-synonym candidate set $\bar{C}_n$ for a lemma x_0 as

$$\bar{C}_n(x_0) = C_1(x_0) \cup C_2(x_0) \ldots \cup C_n(x_0).$$

Now we can define weakly n-synonymy.

Definition 4.2. Two lemmas w_1 and w_2 are weakly n-synonymous with respect to dictionary D and rule R if

$$w_1 \in \bar{C}_n(w_2) \wedge w_2 \in \bar{C}_n(w_1).$$

Obviously, weakly 1-synonymy is the same as strong synonymy.

It is easy to visualize these new synonymy relations if we view the lemmas as the nodes of a graph and the synonym candidacy relations as directed edges between the nodes. Figure 4.1 illustrates the various relations. In the graph, a and b are strong synonyms. The lemmas a and c are weakly 2-synonyms. Note that the path from c to a and the path from a to c are of different length.

Definition 4.3. Two lemmas w_1 and w_2 are weakly synonymous if there is an integer $n \geq 1$ such that w_1 are w_2 are weakly n-synonymous.

Thus, weak synonymy is the weakest of all synonym relations. In automatically finding lemmas which happen to fall in the same synset, we treat a synset as an equivalence class where the equivalence relation is weak synonymy. Note that our definition of weak-synonymy is different than the near-synonymy given in [20] where the proximity is defined of over aspects such as style and indirectness. In our case, the proximity is restricted to occurrence in the dictionary definitions of each other.

In order to evaluate the performance of fully automatic synset construction, we experimented with the following two rules.

R_1**:** Slice the definition at the commas. A slice is a synonym candidate if it has an entry in the dictionary.

R_2**:** Slice the definition at the commas. Take as candidates all the slices that are to the right of the last slice that does not have dictionary entry.

R_1 is more inclusive than R_2. However, R_2 is more aligned with how the definitions are given in typical Turkish dictionaries that do not markup the synonyms. In CDT, usually a longer descriptive definition is given first which is then followed by comma separated synonyms or closely related terms. Of course, the descriptive part may also include commas. R_2 tries to eliminate such cases of false synonyms.

As an illustration, suppose the definition of a sense is

$$w_1\ w_2\ w_3,\ w_4,\ w_5\ w_6,\ w_7,\ w_8$$

where w_i are words. Further, for simplicity, suppose that only single words have dictionary entries.

R_1 generates w_4, w_7 and w_8 as synonym candidates while R_2 generates only w_7 and w_8 since the expression $w_5\ w_6$ does not have an entry in the dictionary.

We used R_1 and R_2 and weak synonymy relation to construct the synset graph and determine the synsets by finding the connected components of the resulting directed graph.

Figure 4.2 shows the distribution of synsets for two rules R_1 and R_2 we described above.

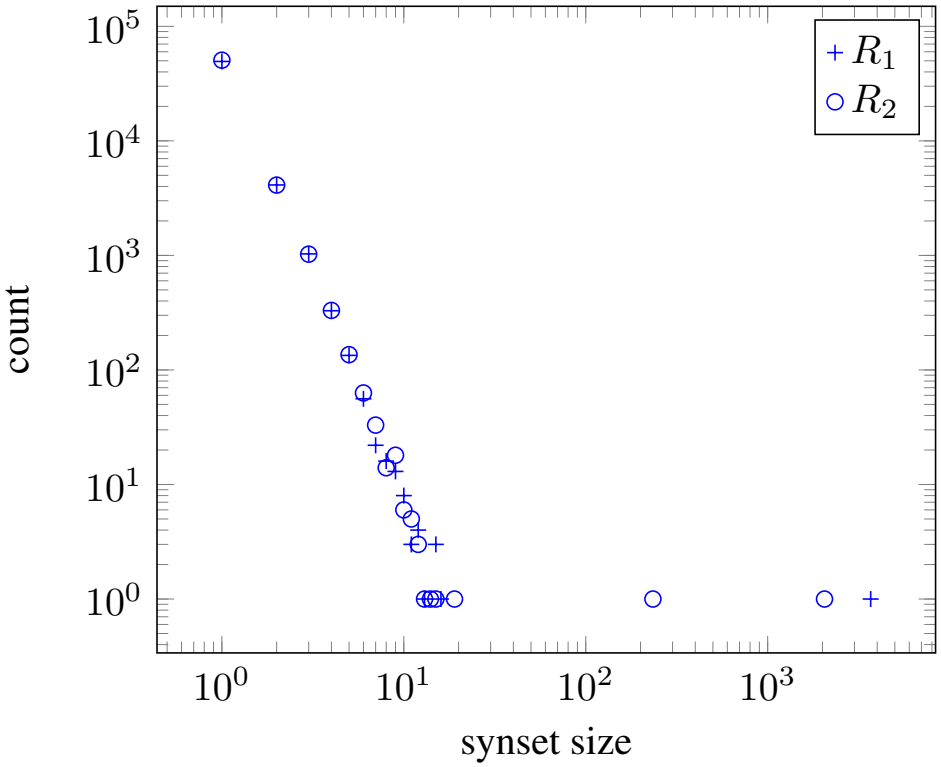

Figure 4.2: The distribution of synset sizes for R_1 and R_2

4.2 Comparison of Synsets

Given a dictionary with one or more senses assigned to each lemma, constructing synsets corresponds to a clustering of the set of senses. Thus, different synset construction methods yield different clusterings of the set of senses. In this section, we evaluate the agreement among clusters formed using KeNet, BalkaNet and automatic thesaurus.

For measuring the distance between two clusterings, we use the variation of information which satisfies the properties of a metric, [43]. Given a set A of size n and

its two partitions $X_1, X_2, \ldots, X_k$ and $Y_1, Y_2, \ldots, Y_l$, the variation of information VI between the two is defined as

$$VI(X,Y) = -\sum_{i,j} r_{ij} \left[\log \frac{r_{ij}}{p_i} + \log \frac{r_{ij}}{q_j} \right],$$

where $p_i = \frac{|X_i|}{n}$, $q_j = \frac{|Y_j|}{n}$ and $r_{ij} = \frac{|X_i \cap Y_j|}{n}$.

Since the automatic construction given in the previous Chapter yield synsets of lemmas rather than those of senses, we make the comparisons over the sets of lemmas. However this reduction poses a problem for manual synsets. Two distinct synsets might contain a common lemma and when we consider only the lemmas, the two synsets would need to be joined, yielding an unnaturally large synset. In order to be able to discard such cases, for manual construction we considered only the lemmas with a single sense. Thus, synsets of senses and synsets of lemmas become the same.

We have the following 6 distinct methods of synset construction and each one yields a different partitioning.

MS: We use the set of manually determined pairs given in Table 3.5 as edges and find the connected components of the resulting sense graph. We eliminate the pairs where one of the lemmas has more than one sense in CDT.
BS: We use the synsets in Balkanet where each lemma has only a single sense.
ASR_1**:** We first prune the dictionary by discarding the lemmas with more than one sense. We construct the synsets by automatically detecting weak synonym pairs under rule R_1.
ASR_2**:** The same as ASR_1 except we use R_2 instead of R_1.
AMR_1**:** The same as ASR_1 except we do not prune the dictionary but keep the lemmas with multiple senses.
AMR_2**:** The same as AMR_1 except we use R_2 instead of R_1.

Table 4.1 gives the variation of information between the pairs of synset partitionings constructed using the methods above. Note that not every method works with the same set of lemmas. This is especially apparent for the case of Balkanet where the set of lemmas is considerably smaller than the other methods. In order to make a fair comparison, we measured the variation of information over the set of lemmas that is common to both methods in a pairwise comparison. So, in a comparison, if a particular lemma occurs in a synset in one partitioning but not in the other, we remove the lemma from the synset.

Table 4.1: Variation of information among different synset construction methods.

	Synset construction method				
	BS	ASR_1	ASR_2	AMR_1	AMR_2
MS	0.138	0.066	0.100	0.527	0.326
BS		0.134	0.161	**0.607**	0.384
ASR_1			**0.030**	0.241	0.155
ASR_2				0.265	0.158
AMR_1					0.272

Table 4.1 highlights a couple of similarities and differences among construction methods. Among the automated methods, the variation distances seem to align them on a line as $ASR_2 < ASR_1 < AMR_2 < AMR_1$. Thus, ASR_2 and ASR_1 are quite similar since they confine their search within the set of lemmas which have a single sense. On the other hand, AMR_2 and AMR_1 are not as close. This is intuitively expected as when we consider multiple senses, determining synonym candidates with comma splitting or right splitting tend to make a larger difference in the resulting synsets.

The same alignment can be observed when we compare BS and MS to automated methods. For both, the automated method that comes closest is ASR_1.

Finally, we see that BS and MS are quite similar when projected onto the set of single-sensed lemmas appearing in Balkanet.

Chapter 5

Related work on clustering text

As the data size goes up, it becomes crucial to automatically reduce the data to manageable unit. Research on big networked data analysis, [17], and machine learning for complex networks [60] provide tools for this purpose. In this thesis, we use graph based NLP tools, [45]. Our perspective on texts is that we deal with a network of words, [18]. Many interesting applications emerge from this framework. Text analysis as an application of NLP is one of those. Text clustering is a high level NLP task. This high stage is the semantic stage which uses other stages like morphology and syntax in the preprocessing. All NLP tasks which deal with semantics try to extract the meaning of a text and underlying conceptual features. Depending on language, working in semantic stage needs some preprocessing in lower NLP stages. For example in agglutinative language verbs are almost never in their root forms in texts, whereas in WordNet, like in many dictionaries, the root forms of words are used.

In the rest of this thesis, we implement an algorithm to cluster text data embedded in a vector space. Our clustering is a content based clustering. We use ontology to achieve this. A novel approach that is efficient in high dimensional problems is also proposed. We use a graph based structure to represent text documents. In this approach we do not need corpus or any information based on

corpus. We use semantic similarity measures and test it over 6 Turkish newspapers headlines and finally we test our approach on Turkish Wikipedia (Vikipedi) data by implementing *K*-means and hierarchical clustering algorithms.

5.1 Semantic Similarity

Semantic similarity is a metric, which is defined over words or documents and determines the distance between words or documents based on their meaning. There are two main approaches to measure semantic similarity. First is topological similarity and second is statistical similarity.

5.1.1 Topological similarity

Using topological similarity as a semantic distance measure is a popular method. There are different methods to calculate topological similarity:

- **Edge-based**: Employs edges and edge features.
- **Node-based**: Employs nodes and nodal features.
- **Node-and-Relation-Content-based**: Employs nodal features as well as their relation with each other.
- **Pairwise**: Employs semantic similarity between pairs, like phrase based similarity.
- **Groupwise**: calculate the similarity directly, like Jaccard similarity which calculates similarity between group of words.

A common way to use the edge based approach is to employ a taxonomy for classification [52] . Gene antologies are the most popular source for such methods [5].

Resnik et.al. [57] propose a nodal approach that finds semantic similarity between words using an Is-a taxonomy. They go beyond just counting distances between nodes in a Wordnet and instead define a probability distribution between nodes in a Is-a taxonomy based on shared nodal information. Another related work [54] measures similarity between words, senses and texts. The graph-based approach of Couto et.al. [16] that measures semantic similarity in a gene ontology (GO) is also a node-based method.

Node and relation content based methods are usually based on Resnik similarity and are applied on ontology [19].

Opposite to the pair wise approach that tries to measure semantic similarity over pairs, group wise methods measure semantic similarity over all words or concepts. Jaccard Index is one such method. On the other hand, Bollegala et.al. [9] use certain patterns of word co-occurrences in Web and they train a Support Vector Machine with positive samples from WordNet synsets and negative samples from random matchings of non-synonym words. Both of these works measure semantic similarity between two words without considering context where the words appear.

5.1.2 Statistical similarity

A different approach to measure semantic similarity is to fit a statistical distribution on the data and extract a distance measure from it. Here are some of the important works in this field:

Latent semantic analysis (LSA), [34], is a method in distributional semantics that analyzes relationships between a set of documents and the terms therein. LSA assumes that semantically similar words will occur in similar parts of text.
Pointwise mutual information (PMI), [10], [12], is a statistical method based on a large corpus which uses search engines, like Google search engine, and cannot

measure similarity between whole sentence and document.

NASARI (Novel Approach to a Semantically-Aware Representation of Items) [13] uses Wikipedia and WordNet (in their case BabelNet [48]), and has state-of-the-art performance on multiple datasets in two standard benchmarks: word similarity and sense clustering. By having contents of communication available in the form of machine readable texts, the input is analyzed for frequencies and coded into categories for building up inferences.

5.2 Content based clustering

Unsupervised learning, clustering in particular, forms an important part of data mining. Unlike supervised learning, in this problem we work without any "label" or "class" information about the data. Main application of such an excercise is to find related groups of points in the data [31].

With increasing number of documents in many fields, specially in Web, interest on data analysis has picked up speed. Computer-based content analysis is growing in popularity. Newspaper articles, scientific articles, movie reviews, and so on, can all be subject to systematic analysis of textual data. Content analysis seeks to extract essential information in a given context [32]. The simplest form of content analysis employs text characteristics such as word frequency or length. But data sparsity and context related issues demand the use of other characteristics like synonymity and hyponym-hypernym relationship.

As an example, recommendation systems are popular and benefit from content analysis methods. Recommendation systems use clustering methods in data [1]. Content based methods also are used for social media analysis [68]. Most of these techniques use semantic similarity measures. As we described in Section 5.1, semantic similarity can be measured using ontologies or statistical approaches [57], [9], [27], [65], [51] and [70].

The following concepts are common to all clustering methods: representation model, similarity measure, clustering model and a clustering algorithm. There are two popular techniques to cluster textual data: connectivity models and centroid/spatial models. Typical examples for the first are the random walk type methods or hierarchical clustering methods and for the second is k-means algorithm and its variants. Hierarchical algorithms usually suffer from efficiency problems but they can produce depth information for detailed analysis when k-means and similar algorithms produce less information but are very efficient [42].

Some clustering algorithms may assign one document to more than one cluster while others assign one document only to a single cluster. First case is called a soft (or overlapped) clustering method and the other is called hard clustering [42]. A clustering method may employ a rule-based, statistical or a combined approach. Hotho et.al. [28] use ontology based information like synonyms and up to 5 level hypernyms to cluster documents. They show that using ontology helps increase clustering accuracy. They replace the words in a text with their assigned senses that are given by a word-sense disambiguation method.

Similarly, Sedding et.al. [61] use bag-of-words, ontology relations, POS tags of words in document, and hypernyms. Huang et.al. [29] Euclid distance, Pearson correlation coefficient or the averaged Kullback-Leibler divergence divergence measures. According to the Huang et.al. [29] Jaccard and Pearson coefficient measures find more coherent clusters. Despite most of works in text clustering which use vector space models [58], this work [39] uses more frequent words and more frequent meanings instead of the text. By using vector space model and feature extracting model, Larsen et.al. [35] offer a fast clustering algorithm. There is also a graph-based document clustering approach [27] which denotes words in documents as nodes and their co-occurrences as edges. They use HTML structure for represent documents.

Chapter 6

Textual graph

In this Chapter, we have some preprocessing tasks over data. After these steps, we introduce our graph based data representation. Using graph representation give us some opportunities to use graph based algorithms. We show some graph based analysis in the end of this Chapter.

6.1 Preprocessing data

Turkish is a agglutinative language with many word forms and a word appears with many suffix combinations in different texts. We need some preprocessing steps to find basic dictionary form of words. We also need to find POS tags to solve partial sense ambiguity. We have the following preprocessing pipeline: morphological analysis, morphological disambiguation, word conversion to the (basic) dictionary entry according Turkish phonological rules.

6.1.1 Morphological analyze

All Turkish verbs appear in different forms in text than they appear in dictionary. Often nouns take suffixes in the text like possessive, plural. We deal with the semantics and we should remove morphological parts of words because there are no semantic information in morphological part of words. Furthermore, we need

POS tags of words to do our partial sense disambiguation which we mentioned in the Section 6.2.2. All nouns and adjectives that are nominal, appear in dictionary as their root form while verbs appear in their infinitive form. Beside stemming to root form and finding the POS tag, we also need polarity of verbs. Polarity of verbs determine whether each verb is negative or positive. This information can be useful in determining exact distance between texts but in this work we deal with semantic relatedness and try to cluster texts based on their domains.

We used morphological analyzer which is based on Finite State Machine [26]. This analyzer gives us all possible analyses of a word. We used "Proper" tag if a word was not in analyzer lexicon. We use POS tag and root forms as output of morphological analyzer. As an example, we show all analyses for word "nisan" :

- nisan :nisan+Noun+A3sg+Pnon+Nom
- nisa+Noun+A3sg+P2sg+Nom

and word "yazdı" :

- yazdı : yaz+Verb+Pos+Past+A3sg
- yaz+Noun+A3sg+Pnon+Nom ^ DB+Verb+Zero+Past+A3sg

There are more than one analysis per word in this example, choosing one of them is related to morphological disambiguation step which we describe in Section 6.1.2.

6.1.2 Morphological disambiguation

In previous section 6.1.1, we showed two example words with two analyses per word. Choosing the correct analysis among many possible morphological analyses according to the context is the task of morphological disambiguation.

In other words, morphological analyzer outputs are completely independent of the context and in many cases have more than one root forms. We need to know which root form is mentioned in context. The amount of possible suffixes in Turkish implies many possible analyses in most cases. As a result, most of the words in a text have more than one POS tag. Outputs of morphological analyzer are ambiguous and need disambiguation. Morphological disambiguator gives us exactly which word is mentioned in a special context. We used a morphological disambiguation for Turkish which uses multiple conditional random fields [21]. As an example, we show the output of morphological disambiguation over two words which we give as an example in Section 6.1.1:

For word "nisan":

- nisan :nisan+Noun+A3sg+Pnon+Nom

and word "yazdı" :

- yazdı : yaz+Verb+Pos+Past+A3sg

6.1.3 Convert words to the dictionary entries

In Turkish, root form of a verb is not the form which is listed in Turkish dictionaries. In dictionary, verbs appear in their infinitive forms. Infinitive form in Turkish is generated by appending suffix "mAk" after root forms. According to phonetic rules of Turkish, "A" can be realized as "a" or "e". Using simple scripts to append the infinitive forms, we obtained infinitive form of verbs. Nouns and Adjectives, which we categorize under a common POS "Nominal", appear as their root form in the dictionaries. After this step, we can map each word in a text to a synonym set.

6.1.4 Getting rid of redundant words

Certain frequent words appear in almost all texts. These words pollute the text for our purposes and may result in miscalculation of exact distance between text documents. These words are called as "stop words". We decided to remove these words from texts. The stop words we use :

ve (and), veya (and or), ya (or), her (each), herbiri (each of), böyle (such), şöyle (like this), bir (one), de (also), dahi (even), için (for), ise (too), ile (with), kendi (itself), çok (much), daha (more), ilk (first), yıl (year), ayrıca (likewise), son (finally), iyi (good), yalnız (just), sonra (next), karşı (towards), mu, (question mark), da (also), mi (question mark), mü (question mark), sözcük (word), yoksa (otherwise), en (most), yeniden (again), o (it), biz (we), üzere (about to), hem (also), neredeyse (almost), şu (this), bu (this), etmek (do), yapmak(make), olmak (be)

We also remove punctuations from text, other than those delimiting sentence boundaries. We also removed numbers and digits from text.

6.2 Constructing textual graph

A correct representational structure is crucial for text analysis. Therefore, our first goal is to find a suitable data structure for texts. We represent a document without considering other documents or any corpus. We assume each document contains information about important words and their relation (semantic relation or occurrence relation). The data structure should enable easy and fast extraction of such information.

Many methods try to find important words by merely counting in a document or corpus, whereas our method works differently. Extracting emphatic words in a text gives valuable information about the context. An analysis using word counts

can only provide a crude picture. On the other hand, graphical structure can give a much more detailed one.

6.2.1 Representing text

Each word in a text has some relations to other words in the same text or words in other texts. These relations are divided into two types. First kind of relation arises from the context. For example, words w_1 and word w_2 are in a relationship because w_2 appears after w_1 in the text. These relations are independent from corpus and are specific for the text being analyzed. Second type of relations are the semantic relations, which are independent of the analyzed text. We use Wordnet (KeNet in our case), to extract these independent semantic relations.

Synonym, antonym and hypernym relations, are the semantic relations which we extracted from KeNet and we use most of them in text analysis. Beside these relations, we also use domain information for the word inside text. For example, if text t_1 is about "olive" and text t_2 is about "tree" with superficial information there is no similarity between them when considering the surface forms only. But considering to hypernym relation we know that, "olive" is a kind of "tree" and as a result t_1 and t_2 are similar. In the case of "antonym", if text t_1 is about "freedom" and text t_2 is about "despotism", both are talking about similar things with different polarities. Also synonymy is important because each text has its own language. In two texts with the same context, the word "compatriot" might be written as "citizen" in other text. Both of these words has the same sense but a superficial surface analysis might treat them as two different words. Sometimes, a writer might avoid using a word and prefer to use a certain synonym of the word. Here too, we need to extract semantic relations.

In this work we try to semantically analyse and cluster texts based on their "domain". Semantic relations are extracted from KeNet, as described in Chapter 3.4. We used some NLP preprocessing tools to prepare data as in Section 6. Using graph data structure gives us opportunity to use graph algorithms to extract

information. In Chapter 6.3, we show how we represent texts in a graph structure. For evaluation, we work on some Turkish newspapers and also Vikipedi (Turkish Wikipedia) data which are labeled for their domains. In Section **??**, we show similarity measures over graphs. In result Section 6.3.4, we show our clustering over texts.

Before we start analyzing a document using its network representation, we need a network of semantic relations among words independent of the documents where the word appear. We use this priori network together with the relations characterizing a document to arrive at a complete graph of a document.

Traditionally, WordNets have been used to represent semantic relations in a language. Such relations range from synonymy among nouns to entailments among verbs. In the present work, we use KeNet automatic thesaurus which we constructed during this thesis. Each cluster corresponds to a sub-graph of synonyms. The topology of the sub-graph contains indications about the representativeness of the lemmas within the cluster. For example, some lemmas appear often in the definitions of other lemmas. This is reflected in the cluster in the density of edges among the lemmas which appear in each other's dictionary definitions. We exploited this topology to find the representative lemma of a synonym cluster. We ranked the nodes of a cluster in terms of their betweenness centrality metric and choose the first lemma as the representative node of the synonym cluster. As we explain in the following sections, we use the representative lemma within the document network to improve the document connectivity.

Hypernymy is another semantic relation that we exploited in document representation. Hypernymy defines a "is-a" relationship among words. For example, "fruit" is a hypernym of "apple" and "pear." We use hypernym relation which we previously extracted in this thesis.

We also use domains which we extracted from CDT and Turkish Wikipedia (Vikipedi) and Vikisözlük.

In the Chapter 4 we explain that synonym sets refer to sets of words which have the same meanings. Assume words w^1 and w^2 have the same meaning and hence semantically related. We call this relation r. Synonym sets are defined as $S = \{w^i, w^j | r(w^i, w^j) \neq \emptyset\}$. Previously, all words from CDT have been manually annotated and we have pairs of words which are in a relation r. We can now obtain the sets using a clustering method. In this step, we use a random walk clustering algorithm.

If we represent our words as a graph G, edges in G represent semantic relation r between nodes which represent words. We implemented random walk over G. There are some paths which are not exactly correct in G. This problem arises from mismatches made during manual annotation 3.2.5 or problems in CDT, (see Chapter 3.1). This problem causes many unrelated connected components to connect to each other with mismatched edges. This is a kind of community detection problem. We use an iterative random walk algorithm to find sub-components.

6.2.2 Disambiguating synsets

Finally, we have 10 186 synonym sets which have multiple words in them. It is obvious that some words can appear in more than one sets, since a word can have possibly many senses. In that case, we use part of speech (POS) tags of words and append them to ends of words. For example, word “ekmek” have two meanings in CDT, one have POS verb and other has POS Noun, The word “ekmek” appears in two different sets. With this modification “ekmek” will be appear in one as “ekmek-verb” and in other as “ekmek-noun”. This method solves sense ambiguity problem partially.

6.2.3 Representatives for synsets

In order to name our synsets, we opted to use representative words (representatives) instead of the more common approach of assigning IDs. We believe this improves legibility of the discovered synsets. We used a graph based approach for selecting these representative words. The lexicographers generally use simple phrases or words to define words in dictionaries. This means, there is a simple word to represent a concept in real world and besides this simple word, other synonyms are either borrowed from other languages (mostly Arabic and Persian in our case) or they are old and literary words. Because of this fact, in a synonym set, we tried to select a simple word which we will use that as a representative of set. To choose this representative word, we used "Betweenness" property in a graph.

Betweenness centrality is widely used in network theory. For example in telecommunications network, a node with high betweenness centrality is a node which has more control over other nodes [24]. This concept is also applied in the scope of scientific cooperation, social networks, transport and biology. Betweenness is a centrality measure in graph theory which is based on shortest paths in the graph. Shortest path is a path between two nodes in a graph which contains minimum number of edges between or if graph is weighted takes minimum sum of edges weights between those nodes. There exists at least one shortest path between the vertices for every connected nodes. The betweenness centrality for each vertex is the number of these shortest paths that pass through the vertex. For node v the betweenness centrality is defined as :

$$g(v) = \sum_{s \neg t \neg v} \sigma_{st}(v)/\sigma_{st}$$

Where σ_{st} is the total number of shortest paths from node s to node t and $\sigma_{st}(v)$ is the number of those paths that pass through v.

If we show a synonym set as a graph, a word which has a high betweenness centrality measure to other words in the graph. The representative word is often used in CDT to define the other words. For example, in the set S= {anlaklı, zeki, havsalası geniş, zeyrek, anlayışlı, cin fikirli, mıncırık, ferasetli}, word "zeki" has the highest betweenness centrality and we choose that as the representative for the set S. In Figure 6.1, we show this synset in the graph structure and show how "zeki" is located in this graph.

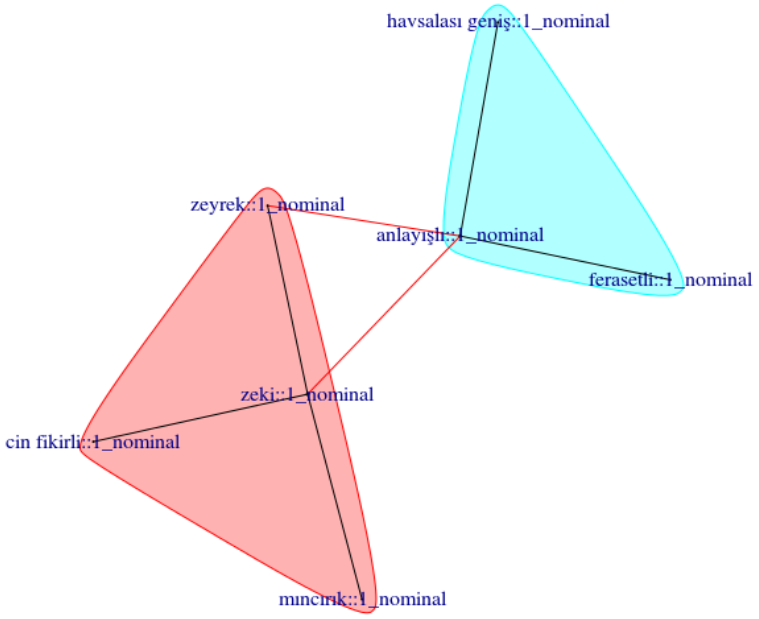

Figure 6.1: Graph structure of a synset

In case of a single word we choose that word as representative.

The motivation of this, after removing pos tags from words, there are two different representatives for word w with sense s_1 which is "noun" and for w with sense s_2 which is "verb". In the rest of this work we will use representatives of words instead of word themselves. About 70% of words have representatives.

6.2.4 Co-occurrence graph

After preprocessing as we describe in the Chapter 6, we convert words in texts to their dictionary format and we also remove redundant words from texts.

We convert our text documents to graphs. In this graph nodes v are words in the text and there is a edge e between node v_1 and v_2 if v_2 appears successive of v_1. We confine this co-occurrence in a sentence. We define sentence boundary set as {".","!","?",";"}. Two word are not co-occurred if there is one of these elements of boundary set between them. According to the previous Section 6.2.3, we use representatives of words, when we consider words as nodes in the graph, we consider representatives as nodes in the graph. If a word has synonyms in text all of these synonyms will be represented as a single word which is their representative word. In this method, co-occurrence graph will be more connected and number of connected components will decrease. More connectivity in the graph gives us more information about text.

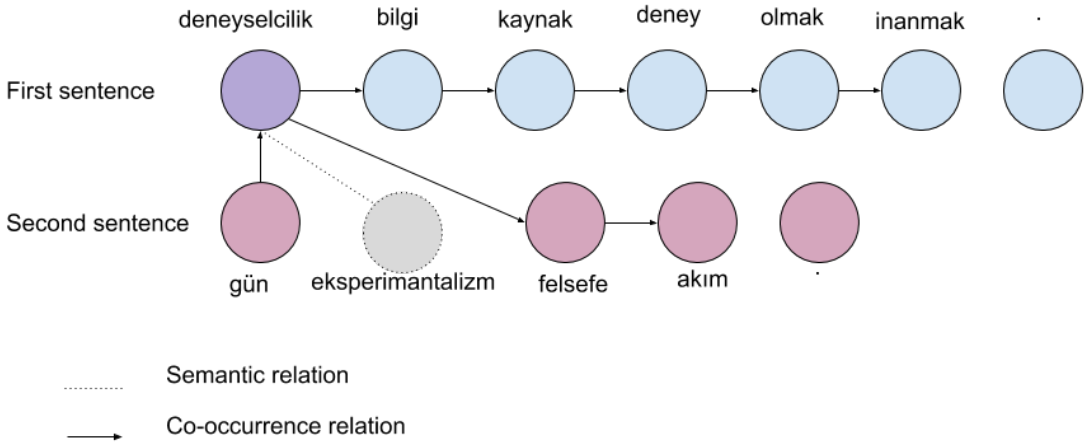

Figure 6.2: Co-occurrence graph using representative words

In Figure 6.2, there are two sentences. First is "deneyselcilik bilginin kaynağının deney olduğuna inanır." and second is "günümüzde eksperimentalizm bir felsefe akımıdır." We use root form of words and also we do not show word "bir" because we eliminate it as a redundant word in preprocessing step 6.1.4. "deneyselcilik"

and “eksperimentalizm” both are in same synonym set and and according to our algorithm, we show them as a word “deneyselcilik” which is their representative.

6.3 Textual graph analysis

In the previous Section we described how we construct a textual graph. Nodes of this graph are each word's representatives and edges are defined according to the co-occurrence graph.

In this Section, we use Jaccard semantic similarity over 5 Turkish newspapers to find similarity between these newspapers' headlines. We collect newspaper headlines from different events in Turkey. Moreover, each headline reflects the ideological position. In the upcoming sections we introduce Jaccard similarity and its variants, PageRank algorithm and discuss the related clustering results. In the Section 5.1, we described a selection of methods that measure semantic distance between words or documents. Here we use Jaccard similarity and generalized Jaccard similarity for this purpose.

6.3.1 Jaccard Similarity

Jaccard similarity coefficient is a statistical metric used to compare the similarity and diversity of sample sets. The Jaccard similarity is defined as the size of the intersection divided by the size of the union of the sample sets. We choose Jaccard similarity measure because of its simplicity and efficiency.

6.3.2 Generalized Jaccard similarity

Jaccard similarity can be used over words. With generalized Jaccard similarity, we are able to measure similarity over numeric sets. Let $\mathbf{X}$ and $\mathbf{Y}$ be two numeric vectors which can be binary when we want to represent text as a bag of words vector.

As$\mathbf{X}$= $\{x_1, x_2, x_3, ..., x_n\}$ and $\mathbf{Y}$= $\{y_1, y_2, y_3, .., y_n\}$ where, $x_i, y_i \geq 0$. the Jaccard similarity between $\mathbf{X}$ and $\mathbf{Y}$ is defined as :

$$J(X, Y) = \sum_i \frac{min(x_i, y_i)}{max(x_i, y_i)}$$

For the next section, we use PageRank scores (discussed next) of words as weights.

6.3.3 PageRank

PageRank [11] is a graph centrality algorithm. This algorithm is one of most important ranking algorithms and is used as a part of the Google search engine. The goal of the PageRank method is to find the most "important" vertex in the graph [45].

Let $G = (V, E)$ be a directed graph with set of vertices V and set of edges E which is subset of $(V \times V)$. Let also be $In(V_i)$ the set of vertices that point to V_i (predecessor) and $Out(V_i)$ be the set of vertices to which that vertex V_i points (successors). The PageRank score of vertex (or its "importance" as defined by PageRank) V_i is defined as follows [11]:

$$S(v_i) = (1 - d)/|V| + d * \sum_{j \in In(V_i)} S(V_j)/|Out(V_j)|$$

where d is a damping factor which is usually set to 0.85 [11].

In our case, after running PageRank over textual graph we will obtain a score for each word in text, the one with the highest score is the most important word inside the text.

6.3.4 Experimental results for clustering headlines

We choose 5 Turkish newspapers as listed below:

- Cumhuriyet: left liberal newspaper
- Hürriyet: right liberal newspaper
- Yeni akit: fundamentalist newspaper
- Yeni Şafak: fundamentalist newspaper
- Aydınlık : nationalist newspaper

And, we choose 5 important events in Turkey as listed below:

- 23 April, National Sovereignty and Children's Day
- 19 May, Commemoration of Atatürk
- 15 July, "Coup d'etat" day
- 16 July, 1 day after "Coup d'etat"
- 17 July, 2 days after "Coup d'etat"

After preprocessing texts, we generate a co-occurrence graph per each headline. Over this textual co-occurrence graph we run PageRank algorithm. We show the results in 4 different experiment runs on. We use generalized Jaccard similarity over 15 most important words in the headline (the first 15 high scored words after running PageRank). Same as first one but over 30 most important words. Running generalized Jaccard similarity over all words in the graph, but still using PageRank scores as word weights. The basic approach where we use only simple Jaccard similarity over words. We also use random walk algorithm over these textual graphs to cluster the headlines.

In Figure 6.3, all three approaches give similar results. The results make sense, since the most similar newspapers are found to be Cumhuriyet and Hurriyet, as expected. The similarity between the nationalist Aydinlik and the fundamentalist-nationalist Yeni Safak is more interesting and unexpected. In this case, we suspect

the shorter headlines might have caused this. Overall, the methods that use the PageRank scores perform much better than the basic approach.

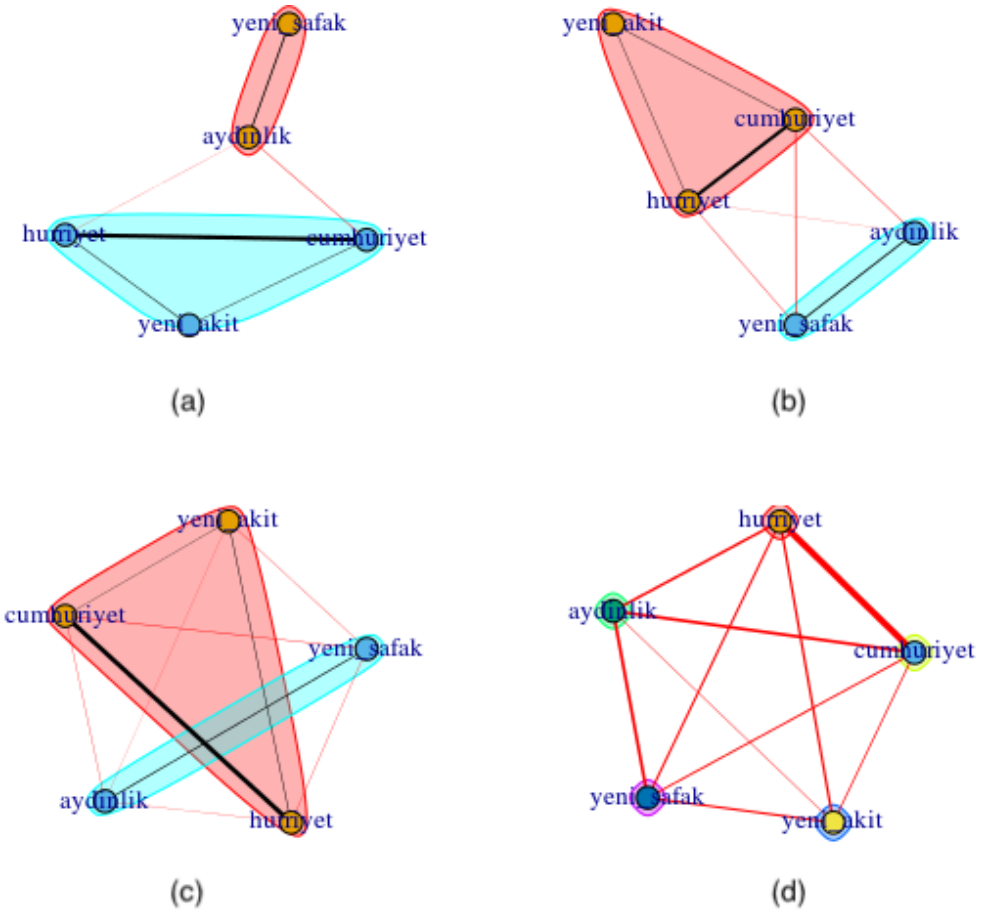

Figure 6.3: 19 May, Commemoration of Atatürk

Results in Figure 6.4, on the other hand, do not show significant difference between the basic approach and the others. Here the similarity results are even better, we observe that the fundamentalist newspapers show clear similarity on one hand, and the liberal-nationalist ones among themselves on the other.

The day of the coup d'etat (shown in Figure 6.5) was not a special date for the newspapers, since the coup d'etat happened long after the newspapers were in print. On the opposite end, the headlines on the 2nd day after the coup d'etat are clearly polarized as this was the first day the newspapers were publishing news about the event due to the censorship on the first day. Hence, in Figure 6.7,

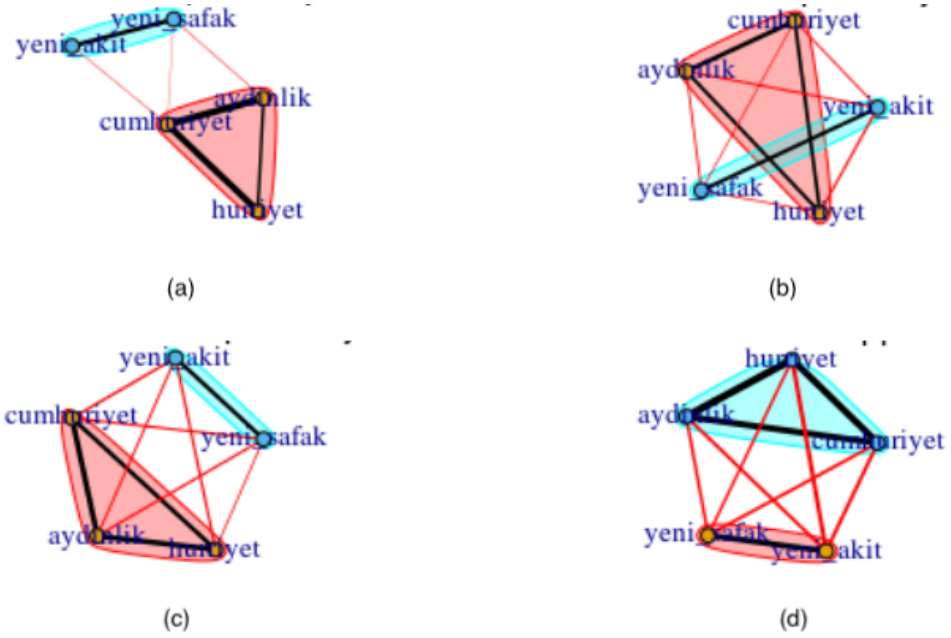

Figure 6.4: 23 April, National Sovereignty and Children's Day

we see a clear separation between the pro-government and opposition newspaper headlines.

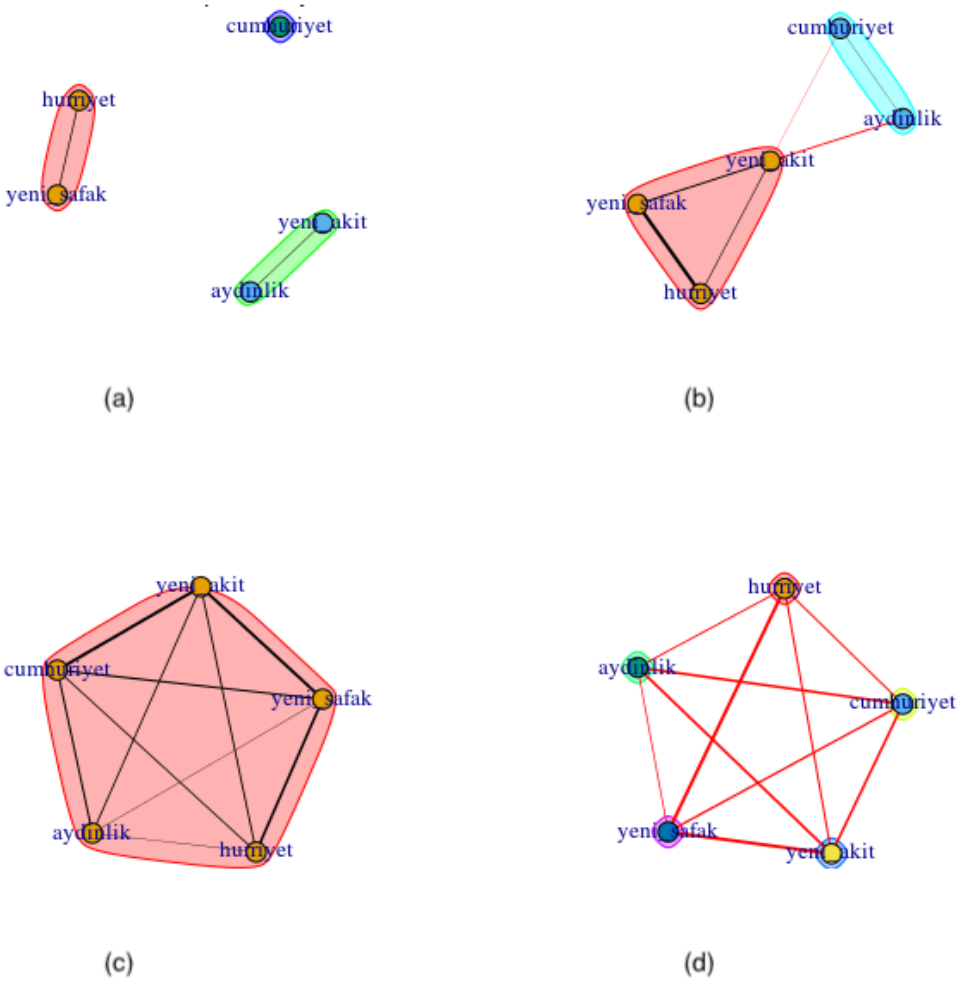

Figure 6.5: 15 July, “Coup” day

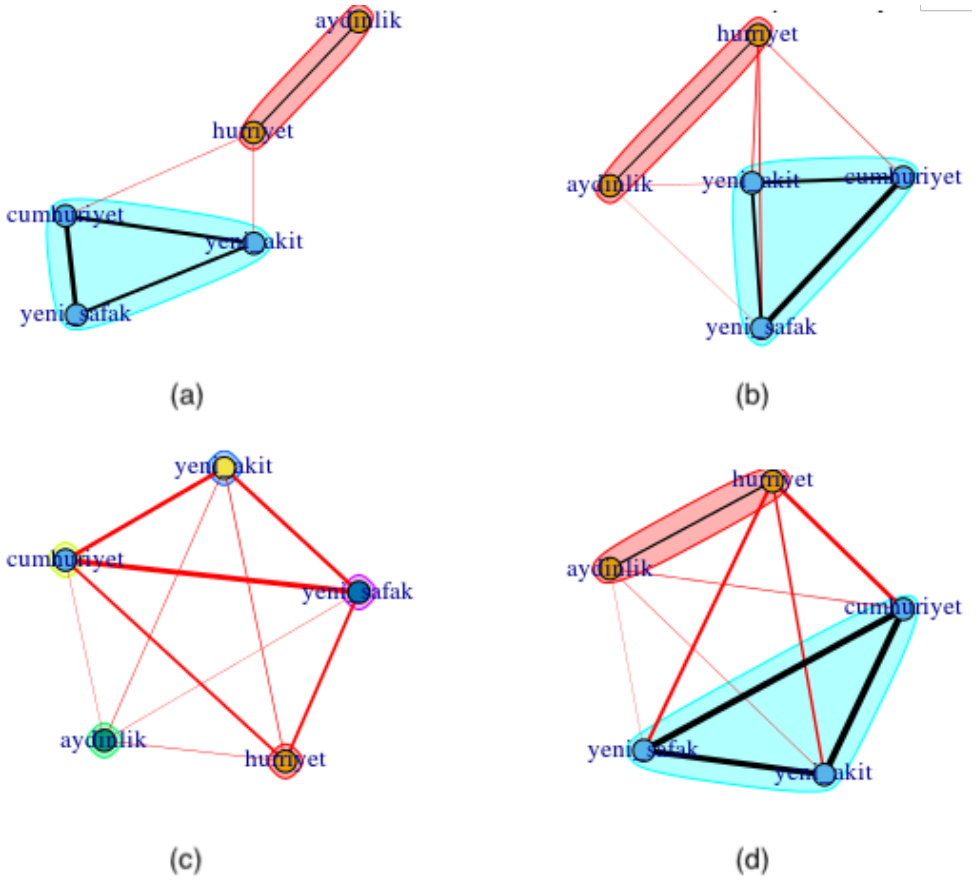

Figure 6.6: 16 July, 1 day after "Coup"

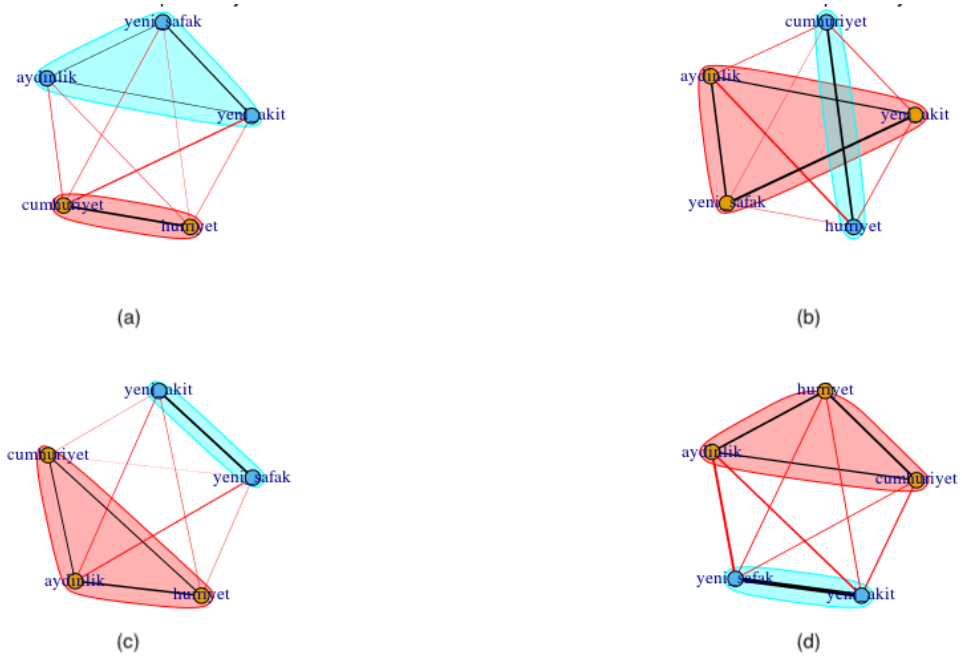

Figure 6.7: 17 July, 2 days after "Coup" day

Chapter 7

Page2Vec algorithm

In the previous Section, we see that, using PageRank algorithm to extract important words in the text can be useful. In this section, we introduce a new approach to convert a text document to a vector using its textual graph properties and semantic relations extracted from KeNet.

Representing a text as a graph gives us an opportunity to use graph based algorithms. Now we want to represent the text document as a vector. Each component of this vector will have values in $[0, 1]$. Word co-occurrence, synonymy and PageRank scores all contribute to this vector space representation. In Section 6.2.4, we mentioned that since we are using representative words as nodes, co-occurrence graph of the text becomes much more connected. This helps against data sparsity without any loss of meaning, for example, by increasing the connectivity of the co-occurrence graph. This co-occurrence graph is constructed using representative words as explained in Section 6.2.4. Let M_{w_i,f_j} be a matrix when w_i are words in the text document and $f_j \in \{D, H\}$ when $D = \{d_1, ..., d_m\}$ represents domains set and $H = \{h_1, ..., h_n\}$ represents hypernyms set which are both taken from KeNet. In our case n is 747 and m is 64, it means our vectors are 811-dimensional per each text document and are independent from text document length. If w_i is in domain D then $M_{w_i,d_j} = 1$, 0 otherwise. If has a hypernym in H then $M_{w_i,h_j} = 1$, 0 otherwise. Figure 7.1 shows an example M.

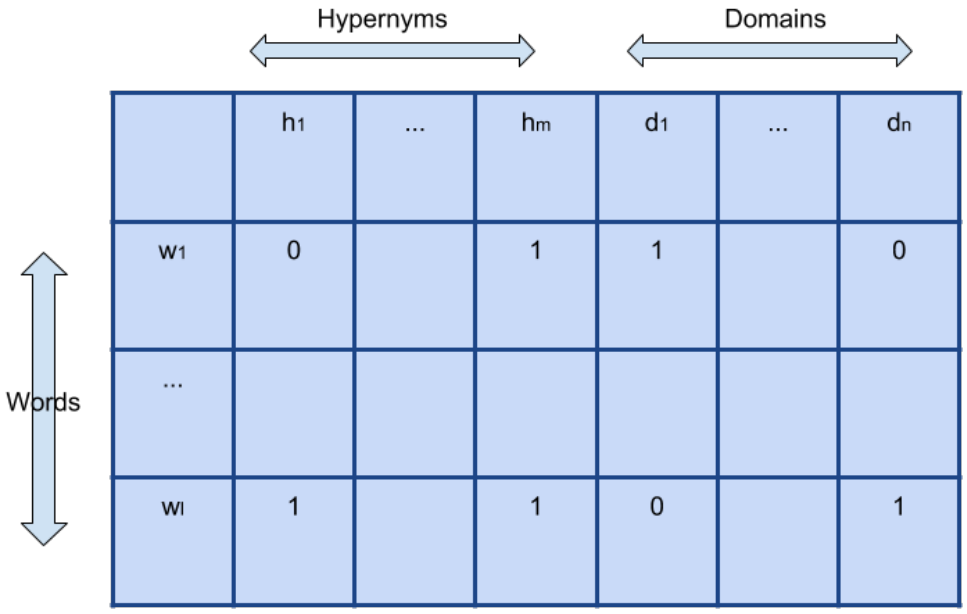

Figure 7.1: domain-hypernym feature incidence matrix

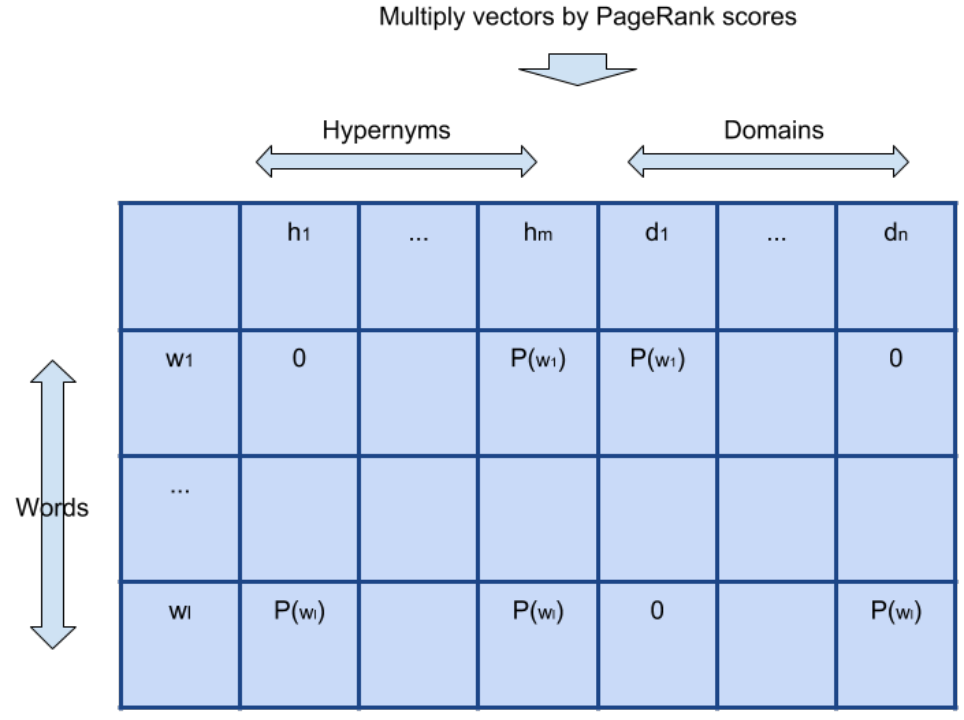

Figure 7.2: Multiply word vectors by the corresponding PageRank score

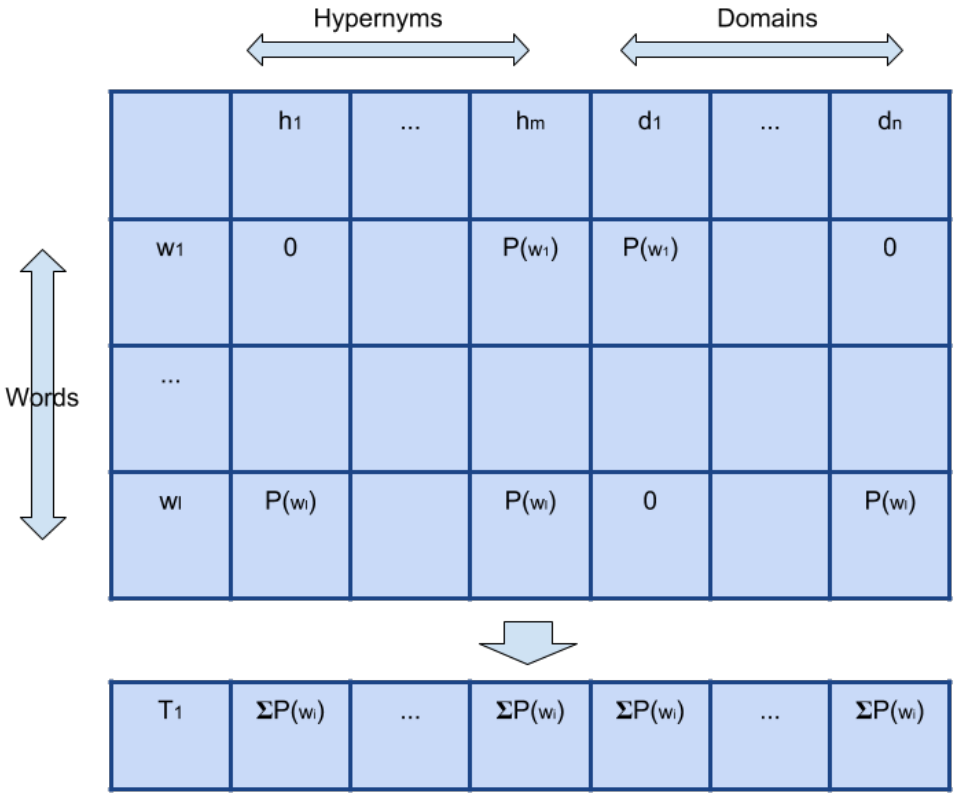

Figure 7.3: Sum over columns to find vector for text t

In the next step, we calculate $\mathcal{H}_{w_i,f_j}$, where $\mathcal{H}_{w_i,f_j} = P(w_i) * M_{w_i,f_j}$ when $P(w_i)$ is PageRank score for word w_i. Figure 7.2 we show the details.

Finally, for text t vector is calculated from $\mathcal{H}$ as $\sum_{w_i} \mathcal{H}_{w_i,f_i}$.

7.1 Experimental Results

In this Section, we use Page2Vec algorithm to convert text documents to the vectors and then cluster them. We collect Vikipedi articles in 5 different domains. Information about domains are given by Vikipedi. Each text document has a single domain. We convert these texts to a vector and run K-means and hierarchical clustering over them to find 5 clusters. Our goal is to obtain a clustering that

parallels the text domain. Domains used in this example are Sports, Politics, Literature, Nature, Geography.

7.1.1 *K*-means clustering

In Figure 7.4, we show k-means clustering result. We show the clusters in different colors and we use PCA dimensional reduction method to visualize vectors in 2-dimensions. We see only a single case of mis-clustering, where one document from Geography domain appears in Nature cluster and hence the *K*-means clustering error is 0.04.

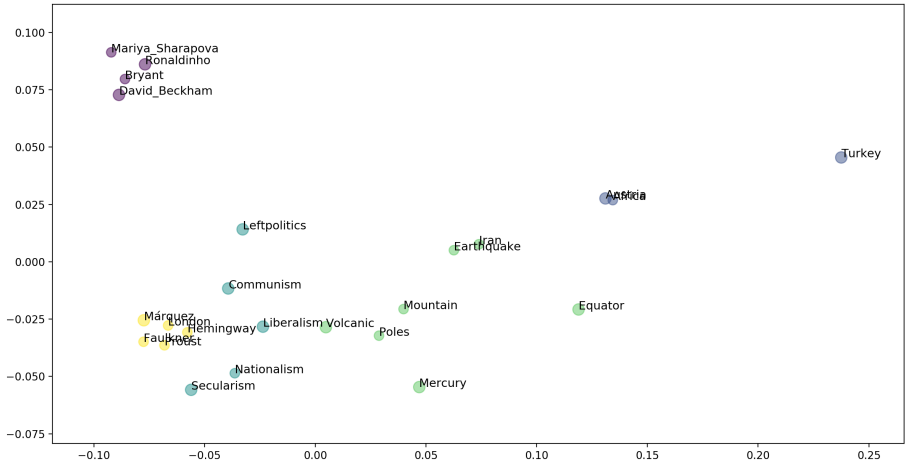

Figure 7.4: Clustering using *K*-means over Pagr2Vec outputs

A recent work on vector-space representation of text documents is by Le at.el. [36]. Their algorithm is based on neural networks, and is named Doc2Vec. This work is considered state-of-the-art in semantic text clustering.

We used gensim [56], to convert our documents to vectors. In Figure 7.5, we see results of *K*-means over Doc2Vec outputs. We see that *K*-means results using our method are more accurate than Doc2Vec method results. There is a small difference between domain "Nature" and "Geography", and these domains share

more common words among themselves. Our method results in a sharper clustering that properly differentiates these domains, while they are mixed for Doc2Vec clustering. We also observe better results for the sports domain.

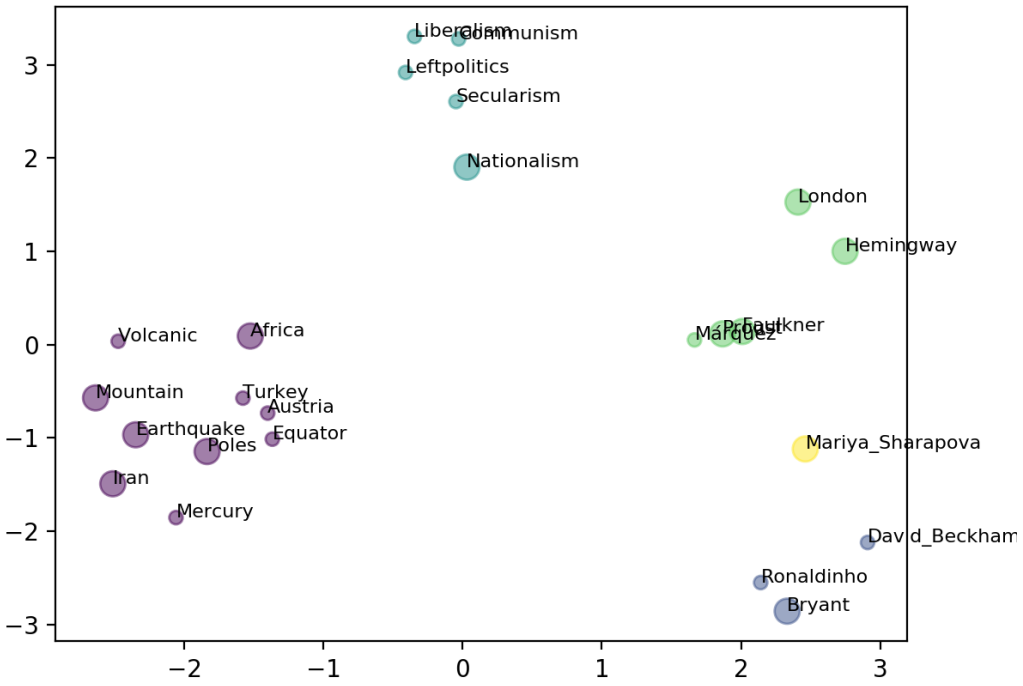

Figure 7.5: Clustering using K-means over Doc2Vec outputs

7.1.2 Hierarchical clustering

In Figure 7.6, we show hierarchical clustering results. Unlike K-means we do not need to know cluster numbers in advance. As we can see in Figure 7.6 "Nature" cluster and "Geography" cluster are grouped together at a higher level.

The selected documents are not distributed equally in all clusters, yet our results are quite accurate. This suggests that our approach might be robust to cluster imbalance. In Figure 7.7 we show the result of using Hierarchical clustering using Doc2Vec algorithm outputs. Our vectors yield a more sensible result at every clustering level. Our method groups "Literature" and "Politics" in a higher level,

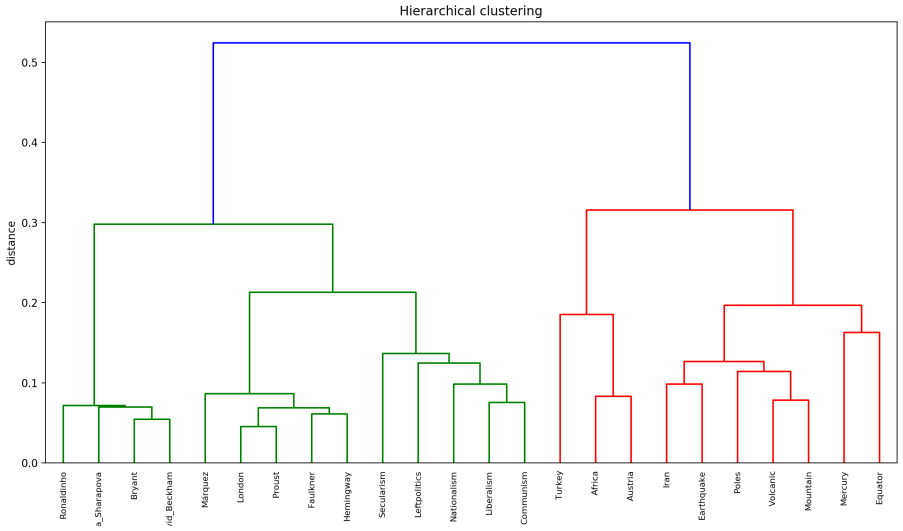

Figure 7.6: Clustering using Hierarchical clustering over Page2Vec outputs

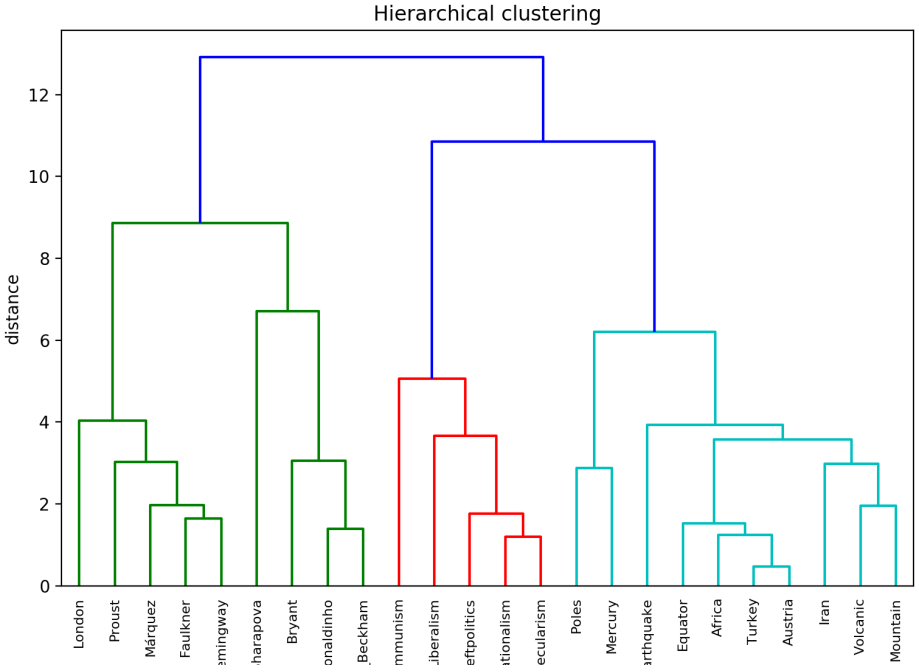

Figure 7.7: Clustering using Hierarchical clustering over Doc2Vec outputs

while Doc2Vec vectors result in “Politics”, “Nature” and “Geography” being in the same cluster.

Chapter 8

Conclusions

This thesis consists of two related parts. In the first part, we build the most comprehensive Turkish WordNet to date, using a bottom-up approach. Instead of relying on the PWN, which is the overwhelmingly popular approach, we build our WordNet from scratch using a digital Turkish dictionary. The process consists of automatic as well as manual tasks. We believe this new WordNet will be a vital resource for future cutting-edge Turkish NLP research. Building a WordNet, of course, is an ongoing process and we expect to update and improve it. Constructing WordNet, needs human labour and supportive resources like well-designed dictionaries. Also, refers to linguistic experts are important.

Constructing a WordNet is a labour intensive undertaking. In the present thesis, we presented a summary of our work on building a comprehensive WordNet for Turkish. Our manual annotation involved a total of 9 human annotators over a period of three years.

In our WordNet construction, we mined a comprehensive dictionary of Turkish for synsets. We manually annotated the synsets twice, going over the disagreements for further reliability. We used clustering on the sense graph to find the final synsets.

For Turkish, WordNet construction is made more difficult by the lack of structured lexical resources. The most authoritative resource for Turkish lexicon is

the official Contemporary Dictionary of Turkish published by the Turkish Language Institute. As we discussed in the preceding sections, the CDT has some lexicographical issues. The most acute of these for a WordNet study is the fuzzy boundaries among the senses of a lemma. Although a certain level of imprecision is often expected in lexicography, its level in CDT makes its use in an NLP pipeline difficult. In our study, we used CDT as it is while also noting the areas where it can be improved in a further study at a more fundamental level. Such a study would best start by collapsing some close senses into a single sense.

In our work, human annotators are presented with synonym candidates automatically mined from a monolingual dictionary. Obviously, one cannot expect an unstructured general dictionary to be comprehensive in listing synonym candidates. As a further study, one can imagine mining a large corpus for synonym candidates using contextual clues. In such an analysis for Turkish, context should be made canonical by stripping inflectional morphemes off the lemmas.

For the next stage of our WordNet construction, we will devise methods to automatically break down the huge synsets using contextual clues both from the definitions and corpora. We will still need to verify the resulting components through human annotators. Such a study will also provide us with further guidance on how to structure a canonical dictionary of Turkish.

The current version of KeNet is publicly available for download [22].

In the second part of this thesis, we use the new WordNet as part of a novel method to represent text documents in a vector space. In many machine learning tasks, using data as a fixed length feature vector is very crucial. Bag of word method is one of most popular method to represent textual data. But this method suffer from two important disadvantages. Bag of word does not deal with semantic relations between words and take each words as a individual word which is semantically independent from other words. Also word ordering is not taken to account in this method. There is a novel method which does not have disadvantages of bag of words method. Word embedding, [36] use other words

positions to define a word, when uses neural network. In this method there are no lexicon based information used, instead position and ordering of words become important to represent data as a vector. This work is sate-of-the-art in clustering texts.

In this thesis, we proposed a new approach, which represent texts as a fixed length vectors. We used WordNet relations to represent texts and also we used word ordering. In our preliminary experiments, we show that our method in clustering texts has better results than Doc2Vec. This approach makes use of the domain, hypernym-hyponym and synonym relations obtained from our WordNet. This novel vector-space representation captures semantic relatedness among text documents much better than competing methods. We observed that most of verbs do not carry semantic information as nouns carry. An immediate expansion of this idea is to take into account other semantic relations, which is a topic for future research. For example, using more semantic relation like meronyms or antonyms will be useful.

We used some word sense disambiguation steps, but still there are lots of works in this area. We deal with lemmas and lemmas can be appear in multiple synsets. In this cases we chose one of synsets according to POS tag but if there is an ambiguity in POS tag we chose randomly.

References

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. on Knowl. and Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005. [Online]. Available: https://doi.org/10.1109/TKDE.2005.99

[2] D. Alexeyevsky and A. V. Temchenko, "Wsd in monolingual dictionaries for russian wordnet," in *8th Global WordNet Conference (GWC2016), Bucharest, Romania, 27-30 January 2016.*, 2016.

[3] G. W. Association, "Wordnets in the world," http://globalwordnet.org/wordnets-in-the-world/, 2017, accessed: 2017-07-01.

[4] J. Atserias, L. Villarejo, and G. Rigau, "Spanish wordnet 1.6: Porting the spanish wordnet across princeton versions." in *LREC*, 2004.

[5] S. Benabderrahmane, M. Smail-Tabbone, O. Poch, A. Napoli, and M.-D. Devignes, "Intelligo: a new vector-based semantic similarity measure including annotation origin," *BMC bioinformatics*, vol. 11, no. 1, p. 588, 2010.

[6] L. Benitez, S. Cervell, G. Escudero, M. Lopez, G. Rigau, and M. Taulé, "Methods and tools for building the catalan wordnet," *arXiv preprint cmp-lg/9806009*, 1998.

[7] O. Bilgin, Ö. Çetinoğlu, and K. Oflazer, "Building a wordnet for turkish," *Romanian Journal of Information Science and Technology*, vol. 7, no. 1-2, pp. 163–172, 2004.

[8] W. Black, S. Elkateb, H. Rodriguez, M. Alkhalifa, P. Vossen, A. Pease, and C. Fellbaum, "Introducing the arabic wordnet project," in *Proceedings of the third international WordNet conference*. Citeseer, 2006, pp. 295–300.

[9] D. Bollegala, Y. Matsuo, and M. Ishizuka, "Measuring semantic similarity between words using web search engines." *www*, vol. 7, pp. 757–766, 2007.

[10] G. Bouma, "Normalized (pointwise) mutual information in collocation extraction," *Proceedings of GSCL*, pp. 31–40, 2009.

[11] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer networks and ISDN systems*, vol. 30, no. 1-7, pp. 107–117, 1998.

[12] J. A. Bullinaria and J. P. Levy, "Extracting semantic representations from word co-occurrence statistics: A computational study," *Behavior research methods*, vol. 39, no. 3, pp. 510–526, 2007.

[13] J. Camacho-Collados, M. T. Pilehvar, and R. Navigli, "Nasari: a novel approach to a semantically-aware representation of items," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 567–577.

[14] H. Chen, B. Schatz, T. Ng, J. Martinez, A. Kirchhoff, and C. Lin, "A parallel computing approach to creating engineering concept spaces for semantic retrieval: The illinois digital library initiative project," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 771–782, 1996.

[15] H. Chen, T. Yim, D. Fye, and B. Schatz, "Automatic thesaurus generation for an electronic community system," *Journal of the American Society for information science*, vol. 46, no. 3, p. 175, 1995.

[16] F. M. Couto, M. J. Silva, and P. M. Coutinho, "Measuring semantic similarity between gene ontology terms," *Data & knowledge engineering*, vol. 61, no. 1, pp. 137–152, 2007.

[17] M. Dehmer, F. Emmert-Streib, S. Pickl, and A. Holzinger, *Big data of complex networks*. CRC Press, 2016.

[18] P. S. Dodds and C. M. Danforth, "Measuring the happiness of large-scale written expression: Songs, blogs, and presidents," *Journal of happiness studies*, vol. 11, no. 4, pp. 441–456, 2010.

[19] H. Dong, F. K. Hussain, and E. Chang, "A context-aware semantic similarity model for ontology environments," *Concurrency and Computation: Practice and Experience*, vol. 23, no. 5, pp. 505–524, 2011.

[20] P. Edmonds and G. Hirst, "Near-synonymy and lexical choice," *Comput. Linguist.*, vol. 28, no. 2, pp. 105–144, Jun. 2002. [Online]. Available: http://dx.doi.org/10.1162/089120102760173625

[21] R. Ehsani, M. E. Alper, G. Eryigit, and E. Adali, "Disambiguating main pos tags for turkish," in *Proceedings of the 24th Conference on Computational Linguistics and Speech Processing (ROCLING 2012)*, 2012, pp. 202–213.

[22] R. Ehsani, E. Solak, and O. T. Yıldız, "Kenet," http://haydut.isikun.edu.tr/kenet.html, 2017, accessed: 2017-11-01.

[23] C. Fellbaum, "ed. wordnet: an electronic lexical database," *MIT Press, Cambridge MA*, vol. 1, p. 998, 1998.

[24] L. C. Freeman, "A set of measures of centrality based on betweenness," *Sociometry*, pp. 35–41, 1977.

[25] V. Fromkin, R. Rodman, and N. Hyams, "An introduction to language," 2013.

[26] O. Görgün and O. T. Yildiz, "A novel approach to morphological disambiguation for turkish," in *Computer and Information Sciences II*. Springer, 2011, pp. 77–83.

[27] K. M. Hammouda and M. S. Kamel, "Efficient phrase-based document indexing for web document clustering," *IEEE Transactions on knowledge and data engineering*, vol. 16, no. 10, pp. 1279–1296, 2004.

[28] A. Hotho, S. Staab, and G. Stumme, "Ontologies improve text document clustering," in *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*. IEEE, 2003, pp. 541–544.

[29] A. Huang, "Similarity measures for text document clustering," in *Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand*, 2008, pp. 49–56.

[30] H. Isahara, F. Bond, K. Uchimoto, M. Utiyama, and K. Kanzaki, "Development of the japanese wordnet." in *LREC*, 2008.

[31] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, 2009, vol. 344.

[32] K. Klaus, "Content analysis: An introduction to its methodology," 1980.

[33] J. Kontos, "Artificial intelligence and natural language processing," *E. Benou, 1st Ed. Athens: E. Benou Hellas*, 1996.

[34] T. K. Landauer and S. T. Dumais, "A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge." *Psychological review*, vol. 104, no. 2, p. 211, 1997.

[35] B. Larsen and C. Aone, "Fast and effective text mining using linear-time document clustering," in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1999, pp. 16–22.

[36] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International Conference on Machine Learning*, 2014, pp. 1188–1196.

[37] S. Lee, S.-Y. Huh, and R. D. McNiel, "Automatic generation of concept hierarchies using wordnet," *Expert Systems with Applications*, vol. 35, no. 3, pp. 1132 – 1144, 2008.

[38] C. H. Li, J. C. Yang, and S. C. Park, "Text categorization algorithms using semantic approaches, corpus-based thesaurus and wordnet," *Expert Systems with Applications*, vol. 39, no. 1, pp. 765 – 772, 2012.

[39] Y. Li, S. M. Chung, and J. D. Holt, "Text document clustering based on frequent word meaning sequences," *Data & Knowledge Engineering*, vol. 64, no. 1, pp. 381–404, 2008.

[40] K. Lindén, J. Niemi, and M. Hyvärinen, *Extending and Updating the Finnish Wordnet*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 67–98. [Online]. Available: https://doi.org/10.1007/978-3-642-30773-7_7

[41] L. Lovász, "Random walks on graphs," *Combinatorics, Paul erdos is eighty*, vol. 2, no. 1-46, p. 4, 1993.

[42] C. D. Manning and H. Schütze, *Foundations of statistical natural language processing*. MIT press, 1999.

[43] M. Meilă, "Comparing clusterings by the variation of information," in *Learning Theory and Kernel Machines: 16th Annual Conference on Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003, Washington, DC, USA, August 24-27, 2003. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 173–187.

[44] E. Mengusoglu and O. Deroo, "Turkish lvcsr: Database preparation and language modeling for an agglutinative language," in *in ICASSP'2001, Student Forum, Salt-Lake City*, 2001.

[45] R. Mihalcea and D. Radev, *Graph-based natural language processing and information retrieval*. Cambridge university press, 2011.

[46] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller, "Introduction to WordNet: an on-line lexical database," *International Journal of Lexicography*, vol. 3, no. 4, pp. 235–244, 1990.

[47] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[48] R. Navigli and S. P. Ponzetto, "Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network," *Artificial Intelligence*, vol. 193, pp. 217–250, 2012.

[49] M. Palmer, H. T. Dang, and C. FELLBAUM, "Making fine-grained and coarse-grained sense distinctions, both manually and automatically," *Natural Language Engineering*, vol. 13, no. 2, pp. 137–163, Jun. 2007.

[50] Y. C. Park and K.-S. Choi, "Automatic thesaurus construction using bayesian networks," *Information Processing & Management*, vol. 32, no. 5, pp. 543–553, 1996.

[51] T. Pedersen, S. V. Pakhomov, S. Patwardhan, and C. G. Chute, "Measures of semantic similarity and relatedness in the biomedical domain," *Journal of biomedical informatics*, vol. 40, no. 3, pp. 288–299, 2007.

[52] V. Pekar and S. Staab, "Taxonomy learning: factoring the structure of a taxonomy into a semantic classification decision," in *Proceedings of the 19th international conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, 2002, pp. 1–7.

[53] M. Piasecki, S. Szpakowicz, M. Maziarz, and E. Rudnicka, "plwordnet 3.0 – almost there," in *8th Global WordNet Conference (GWC2016), Bucharest, Romania, 27-30 January 2016.*, 2016.

[54] M. T. Pilehvar, D. Jurgens, and R. Navigli, "Align, disambiguate and walk: A unified approach for measuring semantic similarity," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2013, pp. 1341–1351.

[55] O. U. Press, "Oxford living dictionaries," https://en.oxforddictionaries.com, 2017, accessed: 2017-10-20.

[56] R. Řehůřek and P. Sojka, "Software Framework for Topic Modelling with Large Corpora," in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta: ELRA, May 2010, pp. 45–50, http://is.muni.cz/publication/884893/en.

[57] P. Resnik *et al.*, "Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language," *J. Artif. Intell. Res.(JAIR)*, vol. 11, pp. 95–130, 1999.

[58] G. Salton, "Automatic text processing: The transformation, analysis, and retrieval of," *Reading: Addison-Wesley*, 1989.

[59] E. Şaşmaz, R. Ehsani, and O. T. Yildiz, "Hypernym extraction from wikipedia and wiktionary," in *Signal Processing and Communications Applications Conference (SIU), 2017 25th*. IEEE, 2017, pp. 1–4.

[60] I. Scholtes, "Understanding complex systems: When big data meets network science," *it-Information Technology*, vol. 57, no. 4, pp. 252–256, 2015.

[61] J. Sedding and D. Kazakov, "Wordnet-based text document clustering," in *proceedings of the 3rd workshop on robust methods in analysis of natural language data*. Association for Computational Linguistics, 2004, pp. 104–113.

[62] M. Shamsfard, A. Hesabi, H. Fadaei, N. Mansoory, A. Famian, S. Bagherbeigi, E. Fekri, M. Monshizadeh, and S. M. Assi, "Semi automatic development of farsnet; the persian wordnet," in *Proceedings of 5th Global WordNet Conference, Mumbai, India*, vol. 29, 2010.

[63] R. Snow, S. Prakash, D. Jurafsky, and A. Y. Ng, "Learning to merge word senses," in *EMNLP-CoNLL 2007 - Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Stanford University, Palo Alto, United States, Dec. 2007, pp. 1005–1014.

[64] P. Srinivasan, "Thesaurus construction," *Information Retrieval: data structures and algorithms*, pp. 161–218, 1992.

[65] E. Terra and C. L. Clarke, "Frequency estimates for statistical word similarity measures," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics, 2003, pp. 165–172.

[66] Y.-H. Tseng, "Automatic thesaurus generation for chinese documents," *Journal of the Association for Information Science and Technology*, vol. 53, no. 13, pp. 1130–1138, 2002.

[67] D. Tufis, D. Cristea, and S. Stamou, "Balkanet: Aims, methods, results and perspectives. a general overview," *Romanian Journal of Information science and technology*, vol. 7, no. 1-2, pp. 9–43, 2004.

[68] P. Velardi, R. Navigli, A. Cucchiarelli, and F. D'Antonio, "A new content-based model for social network analysis," in *Semantic Computing, 2008 IEEE International Conference on*. IEEE, 2008, pp. 18–25.

[69] P. Vossen *et al.*, "Eurowordnet: a multilingual database for information retrieval," in *Proceedings of the DELOS workshop on Cross-language Information Retrieval*, 1997, pp. 5–7.

[70] J. Weeds and D. Weir, "Co-occurrence retrieval: A flexible framework for lexical distributional similarity," *Computational Linguistics*, vol. 31, no. 4, pp. 439–475, 2005.

[71] T. Wei, Y. Lu, H. Chang, Q. Zhou, and X. Bao, "A semantic approach for text clustering using wordnet and lexical chains," *Expert Systems with Applications*, vol. 42, no. 4, pp. 2264 – 2275, 2015.

[72] S. Yildirim and T. Yildiz, "Automatic extraction of turkish hypernym-hyponym pairs from large corpus," *Proceedings of COLING 2012: Demonstration Papers*, pp. 493–500, 2012.

[73] G. K. Zipf, *The Psychobiology of Language*. New York, NY, USA: Houghton-Mifflin, 1935.