LEXICAL COHESION ANALYSIS FOR TOPIC SEGMENTATION, SUMMARIZATION AND KEYPHRASE EXTRACTION

A DISSERTATION SUBMITTED TO
THE DEPARTMENT OF COMPUTER ENGINEERING
AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Gönenç Ercan
December, 2012

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Prof. Dr. Fazlı Can(Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Prof. Dr. İlyas Çiçekli(Co-supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Prof. Dr. Özgür Ulusoy

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Prof. Dr. Ferda Nur Alpaslan

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Assoc. Prof. Dr. Hakan Ferhatosmanoğlu

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Assist. Prof. Dr. Seyit Koçberber

Approved for the Graduate School of Engineering and Science:

Prof. Dr. Levent Onural
Director of the Graduate School

ABSTRACT

LEXICAL COHESION ANALYSIS FOR TOPIC SEGMENTATION, SUMMARIZATION AND KEYPHRASE EXTRACTION

Gönenç Ercan
PhD. in Computer Engineering
Supervisor: Prof. Dr. Fazlı Can
December, 2012

When we express some idea or story, it is inevitable to use words that are semantically related to each other. When this phenomena is exploited from the aspect of words in the language, it is possible to infer the level of semantic relationship between words by observing their distribution and use in discourse. From the aspect of discourse it is possible to model the structure of the document by observing the changes in the lexical cohesion in order to attack high level natural language processing tasks. In this research lexical cohesion is investigated from both of these aspects by first building methods for measuring semantic relatedness of word pairs and then using these methods in the tasks of topic segmentation, summarization and keyphrase extraction.

Measuring semantic relatedness of words requires prior knowledge about the words. Two different knowledge-bases are investigated in this research. The first knowledge base is a manually built network of semantic relationships, while the second relies on the distributional patterns in raw text corpora. In order to discover which method is effective in lexical cohesion analysis, a comprehensive comparison of state-of-the art methods in semantic relatedness is made.

For topic segmentation different methods using some form of lexical cohesion are present in the literature. While some of these confine the relationships only to word repetition or strong semantic relationships like synonymy, no other work uses the semantic relatedness measures that can be calculated for any two word pairs in the vocabulary. Our experiments suggest that topic segmentation performance improves methods using both classical relationships and word repetition. Furthermore, the experiments compare the performance of different semantic relatedness methods in a high level task. The detected topic segments are used in

summarization, and achieves better results compared to a lexical chains based method that uses WordNet.

Finally, the use of lexical cohesion analysis in keyphrase extraction is investigated. Previous research shows that keyphrases are useful tools in document retrieval and navigation. While these point to a relation between keyphrases and document retrieval performance, no other work uses this relationship to identify keyphrases of a given document. We aim to establish a link between the problems of query performance prediction (QPP) and keyphrase extraction. To this end, features used in QPP are evaluated in keyphrase extraction using a Naive Bayes classifier. Our experiments indicate that these features improve the effectiveness of keyphrase extraction in documents of different length. More importantly, commonly used features of frequency and first position in text perform poorly on shorter documents, whereas QPP features are more robust and achieve better results.

Keywords: Lexical Cohesion, Semantic Relatedness, Topic Segmentation, Summarization, Keyphrase Extraction.

ÖZET

KONU BÖLÜMLEME, ÖZETLEME VE ANAHTAR KELİME ÇIKARMA İÇİN KELİME BÜTÜNLÜGÜ ANALİZİ

Gönenç Ercan
Bilgisayar Mühendisliği, Doktora
Tez Yöneticisi: Prof. Dr. Fazlı Can
Aralık, 2012

İnsanlar bir fikri veya hikayeyi anlatırken birbiriyle anlam olarak ilişkili kelimeleri kullanmaktan kaçamazlar. Bu fenomenden iki farklı bakış açısıyla faydalanmak mümkündür. Kelimeler açısından bakıldığında, anlam olarak ilişkili kelimelerin istatistiksel dağılımı ve anlatımda kullanımlarına bakarak anlam olarak ilişkili kelimeleri tanımlamak mümkün olabilir. Anlam bütünlüğüne anlatım açısından baktığımızda da kelimelerin anlam ilişkilerindeki değişime bakarak bir metnin yapısını modellemek ve bu modeli farklı doğal dil işleme problemlerinde kullanmak mümkündür. Bu araştırmada anlam bütünlüğü, bu iki açıdan da incelenmektedir. Önce kelimeler arası anlam ilişikliğinin ölçülmesi için anlam bütünlüğü kullanılmış daha sonra bu kelime ilişkileri konu bölümleme, özet çıkarma ve anahtar kelime çıkarma problemlerinde kullanılmıştır.

Kelimelerin anlam ilişikliğinin ölçülmesi için bir bilgi dağarcığı gerekmektedir. Araştırma kapsamında iki farklı bilgi dağarcığından faydalanılmaya çalışılmıştır. Birinci kelime dağarcığı kelime ilişkilerinin elle girildiği bir anlam ağıdır. İkinci yöntem ise kelimelerin düz metin derlemindeki kullanım dağılımlarını kullanmaktadır. Araştırma kapsamında bu yöntemlerin birbirine göre başarımı ölçülmekte ve kapsamlı bir analiz yapılmaktadır.

Konu bölümleme probleminde kelime bütünlüğü kullanan farklı yöntemler literatürde kullanılmaktadır. Bunların bazıları sadece kelime tekrarlarından faydalanırken, bazıları da eş anlam gibi güçlü anlamsal ilişkilerden faydalanmaktadır. Fakat şu ana kadar çok daha kapsamlı olan kelime ilişikliği yöntemleri bu problemde kullanılmamıştır. Yapılan deneyler göstermektedir ki konu bölümleme probleminin başarımı kelime ilişikliği kullanılarak arttırılabilmektedir.

Ayrıca deneyler farklı kelime ilişikliği ölçüm yöntemlerini karşılaştırmak için kullanılabilmektedir. Konulara göre bölümlenmiş metinler otomatik özet çıkarma probleminde kullanılmış ve kelime zinciri tabanlı yöntemlere göre daha başarılı sonuçlar elde etmiştir.

Son olarak kelime bütünlüğü analizi anahtar kelime bulma probleminde araştırılmaktadır. Geçmiş araştırmalar anahtar kelimelerin belge getirme ve navigasyon için başarılı araçlar olduğunu göstermektedir. Her ne kadar bu araştırmalar anahtar kelime ve belge getirme arasında bir ilişki olduğunu gösterse de, başka bir çalışmada anahtar kelimeleri bulmak için onların belge getirme başarım tahmini kullanılmamıştır. Bu araştırmada sorgu başarım tahmini yöntemlerinin anahtar kelime bulmada kullanımı incelenmiştir. Bunun için sorgu başarı tahmininde kullanılan öznitelikler anahtar kelime bulma probleminde Naive Bayes sınıflandırıcı ile birlikte kullanılmıştir. Yapılan deneyler bu özniteliklerin farklı boyuttaki belgelerde başarımı arttırdığını göstermektedir. Daha da önemlisi bu özniteliklerin yaygın olarak kullanılan deyim geçme frekansı ve belgede ilk kullanım yeri özniteliklerinin tersine kısa belgelerde daha başarılı olduğunu göstermektedir.

Anahtar sözcükler: Kelime bütünlüğü, Anlamsal ilişiklilik, Konu Bölümleme, Özetleme, Anahtar Kelime Çıkarma.

Acknowledgement

I would like to thank Prof. İlyas Çiçekli who has supervised and supported me since the day I started my graduate studies. I am indebted to Prof. Fazlı Can for all the encouragement and guidance he has provided me. Their support, expertise and invaluable recommendations were crucial in this long academic journey.

I would like to thank all my committee members for spending their time and effort to read and comment on my dissertation.

Many thanks to the former and current members of the Computer Engineering Department. It was both a pleasure and honour to get to know and work with you all.

I would like to express my deepest gratitude to my family, Nurhayat-Sadık Ercan and İlkay-Görkem Ercan and Berrin Gözen for their patience and support throughout this long journey. Finally I thank and dedicate this dissertation to my wife Pınar who has given me both strength and courage to take on this challenge.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Acknowledged as the information age by many, the 21^{st} is characterized by providing large amount of data available to masses. Digital revolution enabled this ability by improving almost all aspects of information creation and distribution. Information can now be created in electronic format by anyone through the use of personal computers. The internet not only accommodates massive amounts of data, but also functions as a gateway to share this information.

Information overload is a side-effect of digital revolution that should be treated. As Internet became virtually boundless, a user with an information need is exposed to a large set of data and is not able to utilize this knowledge-base effectively. Information Retrieval techniques aim to increase this effectiveness by helping the user to find the most significant information related to his/her needs. One example for these tools is full-text search engine, which tries to resolve the problem by limiting the focus of the user to documents that contain the queried phrase. Unfortunately natural language is complex and involves ambiguity in different levels. An ambiguous query phrase that has multiple meanings in different contexts can retrieve too many unrelated documents. It is not always easy or possible to clearly express the information need by using query phrases, and this method may fail to narrow the information presented to the user to a manageable level. Summaries and keyphrases are particularly useful in such cases as they concisely indicate the relevance and content of a text document. A user can quickly browse through the documents using keyphrases and summaries, and find documents with relevant information and eliminate others easily. However most of

electronically available content lacks summaries or keyphrases and for this reason creating them automatically is an important task with different applications.

While they are useful, it is a difficult task to automatically create summaries or keyphrases. The difficulty is inherited from the problems of natural language understanding and generation. Turing test [1] states that in order to test if a machine is "intelligent", it should be able to fool a human by imitating another human in a natural language conversation. Even after more than 60 years, it is not even possible to confidently argue if a machine will ever pass this test. In order for a machine to convincingly participate in a natural language conversation, it must successfully perform both natural language understanding and generation. Ideally a summarization system requires these components in order to perfectly mimic summarization capabilities of a human. The Natural Language Understanding (NLU) component tries to map a discourse (text or speech) to a computational model and the Natural Language Generation (NLG) component maps the computational model to natural language. For both NLG and NLU, we humans resolve these ambiguities and relate the discourse to our prior knowledge. A machine on the other hand is not by itself capable of resolving ambiguities efficiently and effectively. Furthermore, it is possible to argue that the problem of organizing and storing prior knowledge and relating the new content to this knowledge is equal to the problem of building a general artificial intelligence.

Even though it is tempting to attack the NLU and NLG problems, it is not fruitful because of the mentioned difficulties. Simplified models for natural language are usually adopted. One such simplified model is based on Lexical Cohesion phenomena seen in discourse. Lexical Cohesion states that in a discourse, the words used are related to each other. For example, in a text about the transportation system in a city, the terms *bus*, *train*, *rail* and *boat* are repetitively (*i.e.* there are multiple instances of each term) used. Lexical cohesion imposes that in a text the terms that make up the text should be related to each other in some way, either in the local context or universally. Since the seminal work of Halliday and Hassan [2], lexical cohesion is widely used in natural language processing tasks such as automated text summarization, malapropism detection and topic segmentation. For a machine, focusing on words instead of sentences or the discourse as a whole simplifies the task greatly. Instead of dealing with all the ambiguities at all levels of natural language, only the terms and their meanings are considered. Organizing prior knowledge in a semantic space of words

An **Australian historian** proposed that the key to understanding **Australia** was "the **tyranny** of **distance** ". **Australians** were **far** removed from their **British ancestors** , **far** from the **centres** of **power** in **Europe** and **North America** and far from each other - with the major **cities** separated by **distances** of some 800 km. Time, however, has broken down that sense of **distance** . **Australians** today do not see **London** or **New York** as the **centre** of the **world** . The **proximity** to **Asian economies** like **China** is an **economic strength** . **Transportation** and **communications links** have taken away the sense of **remoteness** felt by **past generations** . However, the **technology** that truly promises to end the **tyranny** of **distance** is **high speed broadband** , whose benefits we are still only beginning to understand though it has already been a decade since the frenzied **dotcom era** . That is why the **Australian government** is rolling out the world's most ambitious **broadband** project - a national **network** that bring **fibre** to homes in a more than 1,000 **cities** and **towns** covering 93% of **residences** . Next generation **wireless** and **satellite technologies** will cover the other 7%. The **network** will operate at lighting speeds and involve an estimated **investment** of $40 billion through an independent state-owned **enterprise** in **partnership** with the **private sector** .

Figure 1.1: Exemplary text with content words highlighted

and their relationships, is a simpler but effective strategy for modeling the prior knowledge.

Figure 1.1 shows an example text, where the content words are highlighted. Even when only the words in bold are considered, it is possible to see the topical changes and to guess what the text is about in general. In this research, methods described will be based on this observation and will use the words, their meanings and relationships with each other.

Semantic Relatedness

In automated lexical cohesion analysis the first question to be addressed is how to determine the semantic relatedness between term pairs. One approach is to use relationships between term pairs coded manually in a network, as in WordNet. WordNet models the semantic information between words by predefined classical relationships: synonymy(same meaning), antonymy(opposite meaning), hyponymy/hypernymy (generalization/specification) and meronymy/holonymy (member of/has a member). In our previous research utilizing WordNet classical relationships [3, 4, 5], we became aware of shortcomings and limitations of the Thesauri and Ontology based solutions. Knowledge-bases like Thesauri and Ontologies require arduous manual work by humans to create and maintain the database. This challenges research on languages with limited resources, as in

Turkish language. Research on semantic relatedness functions [6, 7] empirically reveal that, in direct applications like multiple-choice synonymy and analogy questions, corpus based functions achieve significantly superior accuracy compared to Thesaurus based approaches. Considering all these reasons, we decided to work on methods that measure semantic relatedness of term pairs from a raw text corpora instead of manually built knowledge-bases. Although this knowledge is not as refined as a manually built ontology, it is more extensive and effective in real life data. In order to further investigate this, different semantic relatedness functions are evaluated in topic segmentation, summarization and keyphrase extraction problems. These experiments give us an opportunity to compare corpus statistics and WordNet based functions.

There is a large body of research on explicitly modeling semantic relatedness between words [8, 7, 6]. Even though WordNet is used for summarization [4, 9], topic segmentation [10, 11] and keyphrase extraction [5], to the best of my knowledge explicit semantic relatedness functions are not used or evaluated in such tasks.

In the literature, evaluation of semantic relatedness (SR) functions are usually intrinsic, where the relatedness scores calculated by automated methods are evaluated by human judgments. Chapter 3 performs an intrinsic evaluation of both English and Turkish SR methods. Having performed an intrinsic evaluation of SR methods, their performance in different applications are investigated. This is called as extrinsic evaluation, where the semantic space is used in a high level NLP task.

Topic Segmentation

Topic segmentation aims to decompose a discourse into different segments, where each segment discusses a different topic. In other words, it finds the topical changes in the text. The relation of topic segmentation with this research can be explained in two folds. First of all, artificial datasets for the evaluation of topic segmentation can be built without much effort, by concatenating different text documents. In such dataset algorithms are expected to detect the original document boundaries. Furthermore, it is possible to prepare a dataset formed by concatenating different sections of a single document/book. In Chapter 4 the

performance of SR methods in topic segmentation task are evaluated for both languages.

The second reasoning tying the topic segmentation problem to this research is the relation between topic segmentation and summarization. Topic segmentation can be considered as an important preprocess of text summarization. The lexical chaining approaches of Ercan [3] and Barzilay et al. [9] use WordNet classical relationships to model topics. Topics are identified by observing the disruptions in the lexical cohesion. Especially in some genres the salient sentences are usually positioned at the start of a topic. Empirical results show that news articles are one such genre [4, 9]. The relationship between segmentation and summarization is issued in the early work by Salton *et al.* [12].

In this research topic segmentation is used to divide a given text into segments of subtopics that contribute to a more general topic. While these segments could intentionally be structured by the author, they could also be the consequence of coherency. In Chapter 4 a new feature called differential lexical cohesion analysis (DLCA) that detects points of lexical cohesion change is introduced. DLCA is a measure that uses any SR method for topic segmentation. These segments are utilized in the summarization method described in Chapter 5.

Summarization

Characteristics of a summarization system are highly affected by the characteristics of its input and output. Jones [13] emphasizes that no single criterion exists for summarization, and different summaries can be considered as "good" with respect to different context factors. She classifies these factors in three main groups: the input factor, the purpose of generating summary and the output factors. A summarization system needs to be defined, developed and evaluated considering these factors.

Naturally, the type of the input is the first factor that should be considered. The quantity of the input determines if the problem is an instance of multi-document summarization or single-document summarization. In multi-document summarization a co-related set of documents is summarized, while in single-document summarization a single document is processed. A common example

of multi-document summarization is in news portals, where news articles gathered from different sources are aggregated in a single summary representing an event. Different genres have different properties. News articles are short texts of few paragraphs, which usually narrate a single incident/event. Novels are long text documents formed of loosely coupled parts. Research articles are mediocre size text documents with a distinguished structure imposed to authors by the publishers.

Purpose of forming the summary affects all aspects of summarization. A generic summary is a term used to address summaries that are formed to only represent significant information in the original document, without any bias. Query-biased summaries are formed in order to answer a question or query, and thus only the significant information related to the query are included in the summary. Update type news summaries are formed from multiple news articles about a single event in order to update the knowledge of a user who has read some of the articles.

The format of the summary is an important constraint. A table of information extracted from the original content can be considered as the most appropriate summary. The most common format usually associated with the term summary is a short coherent running text not more than half of the original content [14]. A summary that is formed of sentences that appear in the original document is called an extract. A summary containing generated sentences that do not appear in the original document is called an abstract. Building abstracts usually involves natural language generation from a model, transformation or compression of the sentences of the original document. In Chapter 5 the summarization system building extracts is introduced and evaluated for both Turkish and English text documents. This summarization system is built on the topic segments found via the algorithm introduced in Chapter 4. The focus of this research is on forming extract type summaries that contain the most salient sentences of the original content, rather than the natural language generation aspects of the problem.

Keyphrase Extraction

One of the most compact representations of a document is a list of keyphrases. Keyphrases defining their original document can be used as indicative summaries,

which can be used for browsing and retrieval. The term keyphrase is preferred to the more common term keyword in order to emphasize that keyphrases can be formed of phrases as in "machine learning". One of the most dominant uses of keyphrases is in research articles. Authors assign keyphrases that best describe their work. The assigned keyphrases do not necessarily appear in the original document. Systems that are able to find even such keyphrases are called keyphrase generation systems. If only keyphrases that appear in the document are targeted, then this system is called a keyphrase extraction system.

Chapter 6 introduces a novel keyphrase extraction method developed in this research, which is based on lexical cohesion analysis. While in both segmentation and summarization the amount of lexical cohesion within a text document is modeled, in keyphrase extraction the aim is to determine in how many different contexts a phrase occurs and how similar these contexts are to each other. The similarity of these contexts are used to measure the phrases' ambiguity.

A typical keyphrase extraction system forms a candidate keyphrase list from phrases that appear in the original document, and evaluates each of these using the observations acquired from the original document. Two of the most effective features to date are frequency and first occurrence position in the given text. Nevertheless, not all keyphrases appear in the original source text and a keyphrase generation system must be able to identify these keyphrases as well. Keyphrase generation is challenged by two difficulties. First, candidate phrases not occurring in the text must be added to candidate phrase lists from external knowledge bases without cluttering the list with irrelevant phrases. Second, the features used in the state-of-the-art systems depend on the frequencies of the phrase in the source document, which cannot be estimated for phrases not appearing in the source document. This work addresses the second problem by introducing features that can also be calculated for phrases unseen in the original document.

Keyphrases have been used as a tool for browsing digital libraries. An example of their application is Phrasier [13], which indexes documents by only using keyphrases, and reports no negative impact in retrieval. Furthermore, Gutwin et al.[15] discuss the use of keyphrases as a tool for browsing a digital library. Although keyphrases are utilized in retrieval and for browsing applications, to the best of our knowledge, no other work utilizes the retrieval performance of phrases for identifying keyphrases.

With these observations, the methods proposed are based on the assumption that when a keyphrase is searched in a general corpus, the retrieved set of documents are expected to be relatively focused on a single domain with high similarity to the original document. For example, while "machine" and "learning" are two ambiguous terms that appear in documents from different domains, the phrase "machine learning" appears in a document set that is more concentrated on a single domain.

Our method evaluates each candidate phrase by its performance in document retrieval, which is ideally measured with respect to a user's information need. In order to evaluate the performance of information retrieval systems the documents retrieved by the queries are evaluated against document sets that are marked as relevant by human judges. The percentage of retrieved documents, which are also selected as relevant by the human judge, is called precision. The percentage of relevant documents, which are also retrieved by the query, is called recall. Precision and recall values indicate the performance of the retrieval system. However, since it is not possible to create human judgment sets of relevancy for each potential query, an estimation of the query performance must be used. The problem of estimating the retrieval performance is a fairly studied problem in the literature known as Query Performance Prediction (QPP) [16]. In QPP, a good performing query is expected to retrieve a document set with a single large or few subset(s) that is/are highly cohesive.

QPP has gained popularity as its applications proved to be beneficial for search engines by improving the document retrieval performance. Applications of QPP are selective query expansion [17, 16], merging results in a distributed retrieval system, and missing content detection [18] to improve efficiency and effectiveness of document retrieval systems. QPP features can be exploited in keyphrase extraction and lead to a keyphrase generation system. In most keyphrase extraction algorithms, features that depend on the intrinsic properties of the given document are used. On the contrary, QPP features proposed are not intrinsic properties of the given document, but they are properties calculated from a background corpora.

An Overall Look at the Applications

Although this dissertation deals with four distinct tasks, namely semantic relatedness, topic segmentation, summarization and keyphrase extraction, these tasks are actually components of a larger system able to produce summaries and keyphrases for a given document. Figure 1.2 shows the overall structure of the general system. The semantic relatedness component provides the prior knowledge to the segmentation, summarization and keyphrase extraction problems. The segmentation algorithm detects topical changes in a text and passes this information to the summarization system. Keyphrase extraction and summarization systems produce the end-products of the general system; summaries and keyphrases. The summarization system enriched by SR methods and topic segments selects the most salient sentences from the original document. The keyphrase extraction system using the background corpora produces keyphrases and significant phrases able to represent the original document. As can be seen from the figure, there are four different artifacts evaluated in the system. These extensive evaluations ensure a detailed analysis of lexical cohesion and the built system.

1.1 Goals and Contributions

The ultimate goal of this research is to provide a summarization and keyphrase extraction system that can be adapted to different languages by observing unstructured text corpora. This is demonstrated by experiments carried out in two different languages, Turkish and English.

The contributions of this research can be outlined as follows;

- Comparison of semantic relatedness functions. This research compares WordNet based, Vector space model based and Dimension reduction based SR methods. Although each are evaluated separately in the literature, no work compares their performance in the same task/data in detail.
- A test-bed for word association task for Turkish. SR methods has been evaluated using word association in English and German languages, however

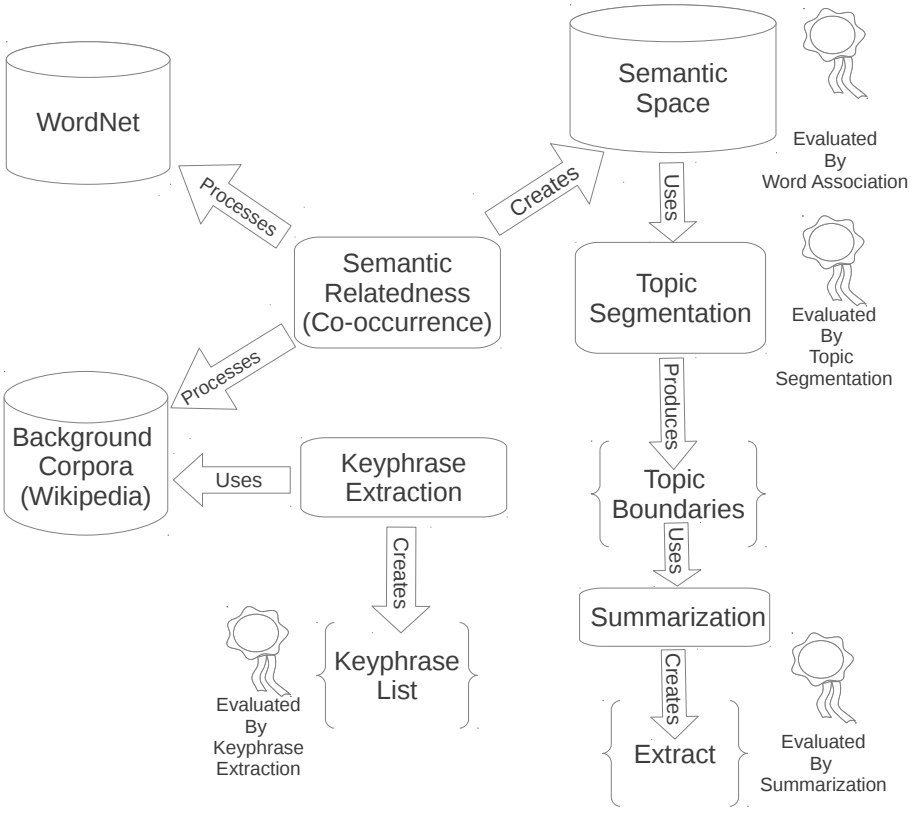


Figure 1.2: The overview of all components developed in this dissertation, their relations and performed evaluations

no similar work exists in the literature for Turkish. The word association dataset built in this research aims to fill this gap.

- Construction of semantic spaces in order to calculate the SR of words, using Turkish and English Wikipedia articles.
- An investigation of the automatically built semantic spaces with respect to WordNet classical relationships. Determining the types of classical relationships identified as closely related by automated methods. Synonyms, hypernyms/hyponyms, siblings or meronyms in WordNet are also identified as closely related in the semantic spaces. In the results, it is possible to observe that this evaluation strategy enables to investigate different properties of semantic relatedness methods when compared to commonly used evaluation methods available in the literature.
- Topic segmentation algorithm using SR functions. Although a large body of research exists for semantic relatedness methods, its applications in high level tasks are scarce and limited to malapropism detection [19]. In this work, the use of SR methods in topic segmentation problem is investigated. This defines not only a novel topic segmentation algorithm, but also an evaluation method for SR methods.
- An investigation of the relationship between summarization and topic segmentation.
- Keyphrase extraction system using features extracted from a background corpora, which can be seen as a first step towards building a keyphrase generation system.

1.2 Outline

In Chapter 2 related work and state-of-the-art methods for each task are discussed. In order to support this discussion and the explanation of work done, important background information is presented. Furthermore, how the ideas in this research are tied to established linguistic theories are discussed.

Following a bottom-up approach, the most atomic measure in the analysis, semantic relatedness is discussed in Chapter 3. The two resources used in the

analysis, namely WordNet and Wikipedia Raw Text articles are introduced. The presentation of the methods are categorized with respect to the resources they use. The datasets used to evaluate SR measures are defined, followed by the results and their discussion.

One level higher from SR methods, the topic segmentation problem is discussed in Chapter 4. Two novel algorithms are introduce that can use any SR measure discussed in this research. Following the presentation of the method, evaluations with respect to different SR methods and state-of-the-art topic segmentation algorithms are presented. Proceeding with the ability to segment a text into topics, a summarization algorithm is defined and evaluated in Chapter 5.

The final task attempted in this research, keyphrase extraction is introduced in Chapter 6. The features originally defined for QPP problem and how these are used in a classifier are introduced. Again the dataset used is presented, followed by the results of experiments performed.

Finally Chapter 7 discusses the work done and results obtained in this research. This chapter creates an opportunity to tie the results in different tasks, considering the relations between different tasks and how they contribute to the literature. The questions and research opportunities that are raised or became more evident to us are also presented in this chapter.

Chapter 2

Related Work

In this research the primary tool used is lexical cohesion, which formulates the problems attacked as a function of words used and their semantic relationships with each other. This is of course a simplification as the meaning or relationships underlying the text are complex. In order to justify this simplification, this chapter first introduces some linguistic theories that model the meaning. The challenges in deep analysis of natural language text and how focusing on lexical semantics alone avoids such problems are discussed. Following this discussion, relevant work in the literature for each application is presented.

2.1 Related Linguistic Theories

Semantics is the study of meaning, and in linguistics it focuses on how meaning can be conveyed by language. Many theories have been proposed in the literature to explain how language can entail a meaning, as well as how to define or extract this meaning from observed text or speech. One of the primary goals of Natural Language Processing (NLP) is to create computational models that can accommodate tasks that involve understanding and generation of natural language. Ideally this can only be accomplished by considering the research in semantics which has different aspects in Linguistics, Cognitive Science and Computer Science. While Linguistics and Computer Science aspects of the problem are obvious, this may not be the case for Cognitive Science. Cognitive Science

deals with the question "How does human mind work?" This question is especially important for NLP as how humans interpret a document can be used as a reference for defining algorithms.

In fact the science discipline called Neurolinguistics investigates the mechanisms of human mind related to the interpretation of discourse and human ability to communicate. While an important body of research is being carried out in Neurolinguistics, still the mechanisms of human mind are not revealed and its impact on NLP are yet to be observed.

While the aim of this section is not to comprehensively cover all linguistic theories, it introduces historical and theoretical foundations of the methods applied. Defining and surveying the theories of lexical semantics must be based on philosophy, cognition and linguistics, and is beyond the scope of this research.

2.1.1 Semiotics and Meaning

One of the earliest theories trying to explain the semantics of natural language is introduced by Saussure [20] in 1916. This theory forming the basis of structuralist linguistics and semiotics is characterised by the terms signified and signifier. The signifier is the sequence of symbols in a text or sounds that we can perceive. The signified is the underlying concept described by the signifier. Saussure argues that the system formed by signifiers and the relationships between them forms a system known as the language. Note that this definition primarily focusses on words used and their meanings. He further argues that the signifiers are arbitrary and differ from language to language, whereas the signifieds are common for languages.

Before the definition of generative grammar, most of the research on linguistics was focused on morphology and phonetics. With the introduction of generative grammar, a shift in attention towards grammar is evident. Later in late 1970s an interest re-emerged towards the structuralist theory with the works of Cruse [21], Halliday and Hassan [2], and Miller et al. [22]. These works can be considered as the basis of WordNet based methods that try to define the concepts, the words signifying these concepts and the relationships between the concepts.

2.1.2 Analytic View

Although can be classified as a subcategory of structuralist theory, quantitative approaches defining the distributional hypothesis were introduced as early in 1950s. To define what words are, Wittgenstein [23] argues that their use in language is important. An argument supporting this view is that a person cannot repeat the dictionary definition of a word, but is able to use it in the right context to convey a meaning.

In his book Harris [24] hypothesizes the concept of distributional similarity. When the distribution of words are observed in a language it is possible to model the semantics of words. While Harris argues that there is a close relation between the distribution of words and semantic properties, he avoids defining the natural language phenomena solely by this distribution, and instead posits that it is observable.

Another philosopher and linguist following the same idea is Firth [25] who is well-known for the phrase "You shall know a word by the company it keeps." Firth argues a very similar idea with both Harris and Wittgenstein. Also the work of Osgood [26] supports the use of distributional properties of words in order to model lexical semantics. Whether or not how these theories are able to explain the natural language is not the main concern of this dissertation, instead how the distributional properties of words can be exploited in real life applications is the primary concern.

2.1.3 Generative Grammar and Meaning

Chomsky [27] introduced generative grammar in order to explain the natural language phenomena. He argues that there is a two level generative model for language, namely deep level and surface level. The deep level is the idea or the thought that is to be expressed. The surface level is the natural language we observe, represented by words or sounds. The surface level is transformed from the deep level by rules.

Another essential component Chomsky defines is the "universal grammar", which is a grammar that is common to all languages and is an innate ability of

human mind. This description of universal grammar both explains why humans are able to learn a language while animals are not. However, it is not helpful in terms of building computational models, as what these innate abilities are not known.

From a practical point of view, Chomsky's theory is important as it uses the Context-Free Grammar (CFG) to model the deep level and surface level of languages. The CFG defined by Chomsky is expressively strong, and has been used to define formal languages such as programming languages. Two important challenges for computers can be observed with a CFG model. First of all, in CFG there can be ambiguity in a grammar, i.e. multiple ways of building a parse tree for the same sentence. Second, as Chomsky [27] argues, the number of sentences that can be generated is infinite.

The methods described in this dissertation can be categorised under structuralist theory based methods. Since lexical semantics is modeled by quantitative methods, i.e. distributional properties of words, it can be considered as an attempt to combine two streams of research in Computational Linguistics.

2.2 Linguistic Background

Natural language whether in the form of speech or text (discourse) is used as a means to express an opinion, fact or concept. It tries to convey a meaning expressed by its author or speaker. The meaning in the form of natural language is organized in terms of words, the ordering of the words and the punctuation marks or tones in speech. Order of words called as syntax can change the meaning as in the examples "*Man bite dog*" and "*Dog bite man.*" Placement of a comma can change the meaning of a sentence completely. Consider the change in meaning in the following examples: "*Careful, children crossing,*" "*Careful children crossing.*" Note that syntax is closely related to the generative grammar first proposed by Chomsky [27]. These two examples can be used to argue that even though the same set of words are used, the meaning can change depending on the grammar.

Natural language contains ambiguity at different levels. A discourse can be interpreted in different ways, which can only be resolved considering the context of the communication. For example, "Flying planes can be dangerous" which

can be interpreted as either “flying a plane is dangerous” or “flying planes are dangerous and can fall on you.” Thus, the meaning of the same sentence can vary in different contexts depending on the intention of the writer/speaker. This example also shows that grammatical category of the word “flying” plays an important role in the semantics of the sentence.

Furthermore, most of the time human mind interprets a discourse with the help of prior knowledge. For example, in the sentence “Kenny is afraid of the water” makes more sense when the interpreter has the prior knowledge of ”the possibility to drown in water”. For this reason the machine should be able to relate the discourse to prior knowledge, or knowledge given previously in the same discourse.

Language is usually processed at different levels. These levels are categorized in terms of text units they consider. For instance morphology deals with the structure of words and what forms they have, while syntax deals with the ordering of words to form a phrase, expression or sentence.

2.2.1 Morphology

Morphology analyses and classifies morphemes in a language. A morpheme is the smallest unit in text, realized as words, suffixes, affixes or infixes. For example the word “kids” is composed of two morphemes, the suffix “-s” which adds plural meaning to the singular form of the word “kid”.

Words in dictionaries and other resources are catalogued by a specific form of the word. For instance, the words “run”, “runs”, “running” are mapped to the same form of the word “run”. This form of the word is known as the lemma. A lemma is transformed to different forms through addition of morphemes, namely suffixes and prefixes. Both in English and Turkish the lemmas are the singular form of the words. The term lexicon defines the word stock of a language, and is formed of lemmas.

Lemmas are categorized to different classes. Some words called as open class words are known to be contributing more to the meaning of the discourse. Typically nouns, verbs, adjectives and adverbs are classified as open class words. The

Class	Description	Turkish	English
Nouns (N)	Used to name a person, place, thing or idea	kişi, uçak	person, plane
Verbs (V)	Used to refer to an action	koşma, gelme	run, come
Adjective (JJ)	Used to describe a noun	**güzel** kalem, **kötü** rüya	**good** pencil, **bad** dream
Adverbs (RB)	Used to describe other classes other than nouns	**güzel** anlamak, **kötü** yapılmış	**well** understood, **badly** done

Table 2.1: List of open classes

Class	Description	Turkish	English
Pronoun (P)	Used to refer to nouns, by substitution	o, bu	it, this
Conjunctions (C)	Used to connect phrases and sentences	ve, veya	and, or
Determiners (DT)	Used to specify references to nouns	-im, -in	the, my
Adpositions (PP)	Used to specify the relation between nouns	benim **için**, bana **doğru**	**to** me, **towards** me

Table 2.2: List of closed classes

term open class denotes that these words can be derived to produce new words through the use of suffixes. Closed class words on the other hand are mostly used for grammatical purposes. Pronouns, conjunctions, determiners, prepositions and postpositions belong to this category. Since it is not possible to derive new words from these, the number of words belonging to closed classes is smaller than open class words.

Our research is primarily focused on Turkish and English languages, thus the discussion will be focused on these two languages only. Table 2.1 summarizes the open classes with examples from Turkish and English. The symbols given in parenthesis are the Penn-Tree bank [28] tags of each class. Table 2.2 summarizes the closed classes with examples from Turkish and English. As can be seen from these tables, the structure of languages posses some similarities and differences. While the open class words are similar, the determiners and adpositions differ in two languages. One major difference between the two languages is that Turkish is an agglutinative language, which makes use of suffixes in order to derive

new words and provide grammatical constructs. For example the determiners, "**your** pencil" can be translated to Turkish as "kalem**in**". Another difference is in adpositions, where English uses prepositions as in "**for** me", while Turkish uses postpositions "benim **için**". Also in Turkish, suffixes can be used instead of adpositions, for example "**to** the plane" can be translated as "uçağ**a**".

Both of the languages use suffixes to change the meaning of the lemma. It is possible to categorize the suffixes into two distinct set as inflectional and derivational. The main distinction between these two types is that the former does not map the word it is applied to, to another lemma. In other words, inflectional suffixes refer to the same concept, and specify the relation of the word to other words in the text as in the example "kalemim", which specifies that the pencil belongs to me. Derivational suffixes however change the meaning of the word, as in the example "kalemlik", which transforms the word "kalem" meaning pencil to "kalemlik" a container holding pencils.

Some common affixes are given, as they are used in the analysis in the following chapters. However this list is by no means exhaustive and complete. The readers are suggested to see Goksel and Kerslake [29], Lewis [30], and Istek [31] for further information on Turkish morphology, and to Carstairs [32], Jurafsky and Martin [33] for English morphology.

Derivational suffixes transform a lemma to another lemma. For example, in English the lemma "stable" is transformed into "stabilization" by the two suffixes "-ize" and "-ation". Furthermore, some derivations transform the word in one class (such as noun) to another class (such as verb). For example, the verb "perform" is transformed to a noun "performance" using the "-ance" suffix.

When compared to Turkish, English language is limited in the number of derivational affixes. Also in contrast to agglutinative languages like Turkish and Finnish, the number of affixes applied to a root lemma is usually low and affixes are seldomly chained to produce a long word. For example, in Turkish the word "yaban-cı-laş-tır-ıl-ma" derived using five suffixes from the word "yaban" is a correct word and can be encountered in a text. While in English the same is possible but not commonly observed in contemporary English.

Some common derivational affixes are given in Table 2.3 with the class they transform from and the class they transform to. Table 2.4, shows a non-exhaustive

Affix	From PoS	To PoS	Example
-ly	JJ	RB	hard-ly, soft-ly
-ess	N	N	wait-(er)-ress, prince-ss
-hood	N	N	mother-hood, neighbor-hood
-ist	N	N	theor(y)-ist
-ity	JJ	N	pur(e)-ity, equal-ity
-ism	JJ	N	ego-ism, conserv(e)-at-ism
-ment	V	N	commit-ment, develop-ment
-er	V	N	paint-er, sing-er

Table 2.3: Some English derivational suffixes

Affix	From PoS	To PoS	Example
-e	V	N	sür-e, diz-e
-ç	V	N	süre-ç, gül-eç
-mi	V	N	geç-miş, yem-iş
-t	V	N	bağın-tı, doğrul-tu
-n	V	V	kaç-ın, gör-ün
-u	V	V	uç-uş, koş-uş
-a	N	V	kan-a, tür-e
-la	N	V	tuz-la, un-la
-de	N	N	göz-de
-den	N	V	sıra-dan
-n	N	N	yaz-ın
-e	N	N	komut-a, göz-e

Table 2.4: Some Turkish derivational suffixes

list of Turkish derivational suffixes. When compared to English, the Turkish derivational suffixes are rich. One important disadvantage of this is the level of ambiguity in Turkish. It is possible to use the same suffix to derive different classes from different classes. For example, the suffix "ın" can transform a V to produce V, or can be used to transform a V to produce N.

Inflectional suffixes are used in a sentence, to define the relationships between the words in the sentence. They are usually used to express tense, person and case. In English there are only few inflectional suffixes, while in Turkish number of different inflectional suffixes is high. In Turkish language inflections can be

Sağlam+laş1+tır2+mak3 (sağlamlaştırmak = to strengthen) Sağlam +Noun +A3sg +Pnon +Nom ĴDB +Verb +Become1 ĴDB +Verb +Caus2 +Pos ĴDB+Noun +Inf13 +A3sg +Pnon +Nom

Figure 2.1: Morphological analysis of the word *sağlamlaştırmak*

used to specify grammatical properties such as the tense of the discourse, possession or the direction of the action. It is important to note that some words in Turkish while derived through inflectional suffixes were semantically changed in time to produce a new word. For example, the words "göz-de", "sıra-dan" have gained additional meanings related to their inflections and are accepted as new words through repeated use. This fact creates an additional challenge for algorithms that model the meaning of the words by posing the question should corpus statistics algorithms process occurrences of "gözde" as an inflected form of "göz" or as a separate entry.

Morphological analysis can be performed using Finite State Automata (FSA). The language accepted by the FSA can be decomposed into morphemes. Morphological analysis tools can be used to convert a surface representation of a word to deeper form which explicitly expresses the morphemes in the word. For example, in Figure 2.1 the types of morphemes for the word is given, where DB stands for the derivational boundary and the following suffixes are inflectional.

For a word's surface representation there can be many different derivations possibly with different parts of speech. This is known as the ambiguity at the morphological level. In such instances the true morphological structure of the word can be determined from the context which it appears in. Considering the two sentences "Bu senin düşün" and "Merdivenden numaradan düşün", they use the same surface representation in different parts of speech, which can be resolved by considering the syntax of the sentence.

While morphological analysis is useful in order to have an in-depth knowledge about the uses of words, it requires disambiguation and additional computational cost. In Information Retrieval simpler methods known as stemmers are commonly used. A stemmer does not have a lexicon and simply strips common inflectional affixes from the words. For example, the Porter Stemmer [34] correctly strips the plural affix of the word "boys" to "boy", however the word "boy" is transformed to "boi" removing the "y" which can be used to convert nouns to adjectives. As this example shows, stemmers are not trying to be linguistically accurate.

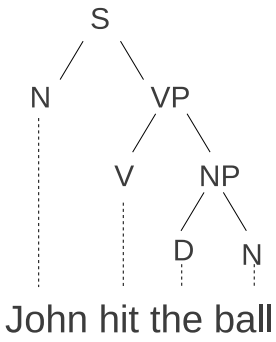


Figure 2.2: Syntax tree of the sentence "John hit the ball"

However, they are consistent as the same form of the word is transformed exactly to the same stem. When compared to English, Turkish and other agglutinative languages are rich in terms of affixes. As a consequence, the number of ambiguous constructs and the computation cost required to resolve them increase. In our research we have experimented with an additional morphological analyser finding the most commonly used form of words for Turkish.

2.2.2 Syntax

Syntax studies how words are composed to form larger semantic text units, for example to form noun phrases. Chomsky [27] defines the syntax trees by Context-Free Grammar, which is largely adopted by the Linguistic and Computer Science community. Practically the end-product of syntax analysis of a sentence is the Syntax tree, which shows how the words are composed to build the sentence.

Figure 2.2 shows an example of a syntactic tree. The noun phrase formed of a determiner "*the*" and "*ball*" is used to form the verb phrase "*hit the ball*". However, at the syntax level there is also ambiguity, meaning that multiple trees can be derived for a sentence.

For growing a syntax tree morphological analysis is required to determine the possible classes each word can take. Through morphological analysis it is possible to narrow the search space for building a syntax tree. However, since it is possible to have ambiguity in both syntax and morphological analysis this is a challenging

1. `The child wept all night.`

2. `The cheese wept all night.`

Figure 2.3: Two sentences having the same syntax tree, where the latter is not meaningful

task, where clues at both levels should be considered to resolve the ambiguities.

Syntax trees can be used to infer the semantic roles of each word in the sentence. Using syntax alone is not enough to model the semantics of a language. Consider the two sentences in Figure 2.3: Both sentences can be derived using the same syntax tree. Nevertheless while the former sentence is both syntactically correct and meaningful, the latter is not meaningful as a "weeping cheese" is not.

The meaning of a discourse depends both on its lexical and syntactic structure. Natural language is filled with ambiguity and counter-examples to rules that can be used in semantic analysis. The computational cost of growing syntax trees is high, and for less studied languages like Turkish, syntax trees and corpora are not readily available. For these reasons simplified models such as lexical semantics and cohesion are attractive options that should be considered in practical applications.

2.2.3 Coherence

With the seminal work of Halliday and Hassan [2] practical applications of structural theory of linguistics emerged in NLP literature. In my Master's Thesis [3] I have reviewed and used a method related to structural theory. The structural theory in linguistics defines semantics as a system of relationships between the text units. In this model coherence is defined as the hidden element in a discourse, which defines the general meaning. The structure of ideas and flow of the document is defined by coherence. Modelling coherence is a difficult task as it is hard to define general patterns without actually interpreting the text.

The example in Figure 2.4 can be considered to clarify the difficulties in coherence. In the first example sentence 1 is coherent with 2. In the second

1. [John is living in a neighbourhood with a very high crime rate. $_1$] [His house was robbed 4 times last year. $_2$]
2. [John is living in a neighbourhood with a very high crime rate. $_1$] [He likes spinach. $_3$]
3. [John is living in a neighbourhood with a very high crime rate. $_1$] [I bought a movie about a murderer. $_4$]

Figure 2.4: Examples of coherence and lack of coherence

example it is not possible for the reader to establish a link between sentence 1 and sentence 2. However, in the presence of a third sentence or prior knowledge like "*Spinach is easy to find in that neighbourhood*", these two sentences become coherent. Also the third example is not coherent even though the words "*crime*" and "*murder*" are related with each other.

2.2.4 Cohesion

Cohesion is the term defining the relationships in a text that are more concrete and observable. It focuses on relatively smaller units of text when compared to coherence. All the cohesion relationships contribute to coherence. Halliday and Hassan [2] define five types of cohesion relationships:

- **Conjunction** - Usage of conjunctive structures like "and" to present two facts in a cohesive manner.
- **Reference** - Usage of pronouns for entities. In the example "***Dr. Kenny*** *lives in London.* ***He*** *is a doctor.*" the pronoun "*he*" in the second sentence refers to "*Dr. Kenny*" in the first sentence. These are also known as anaphora in linguistics.
- **Lexical Cohesion** - Usage of related words. In the example sentence "***Prince*** *is the next* ***leader*** *of the kingdom*", "*leader*" is more general concept of "*prince*".
- **Substitution** - Using an indefinite article for a noun. In the example "*As soon as John was given a* ***vanilla ice cream cone****, Mary wanted* ***one***

It is easy to see that the **dog**'s *family tree* has it's *roots* from **wolves**. In fact, their connection is so close and recent, the position of **wolves** on the *tree* would be located somewhere on the *branches*. Any breed of **dog** can have fertile offspring with a **wolf** as a mate. The only physical trait found on a **wolf** that is not found on a domesticated **dog** is a scent gland located on the outside base of a **wolf**'s **tail**. Every physical trait on a **dog** can be found on a **wolf**. **Wolves** might not have the **coat** pattern of a **Dalmatian**, but there are **wolves** with black **fur** and there are **wolves** with white **fur**.

Figure 2.5: Lexical cohesion example in a discourse, where the content bearning words are highlighted

too." the word "*one*" refers to the phrase "*vanilla ice cream cone.*"

- **Ellipsis** - Implying noun without repeating it. In the sentence "*Do you have a **pencil**? No I don't*" the word "*pencil*" is implied without repeating in the second sentence.

The cohesion relationships conjunction, reference, substitution and ellipsis can only be detected by the use of syntax. A deeper level of analysis is required. Also there are many ambiguous examples where identifying the true relationship may not be so obvious even for humans. Lexical cohesion on the other hand can be carried out by simpler models, with a surface level analysis. In the case of lexical cohesion the relationships observed are simply words or phrases.

2.2.5 Lexical Cohesion

Cruse [21] discusses the issues and importance of lexical semantics in his book. The work of Cruse shows different issues related to both grammar and lexical semantics. From the context it is possible to infer additional relationships and model semantics more appropriately. Especially in problems such as textual entailment or question answering the role of these contextual clues are important. However in topic segmentation, summarization and keyphrase extraction tasks only a rough estimate of the density of lexical cohesion may suffice without further analysis such as syntactical analysis. This observation is exploited in numerous research [9, 10, 4, 5].

In terms of lexical semantics, a viable approach could be to classify the semantic relationships in two categories. The first category is local semantic relations, which are established in the context of the discourse analysed. Second category is global, where semantic relationships commonly known to exist between

lexical items are considered. For example, in Figure 2.5, two sets of words in the text that are globally related are underlined. The first set is represented by italic and underlined words, and the second set is shown by bold and underlined words. These two sets are: {wolf, dog, Dalmatian, fur, coat, tail} and {family, tree, branches, roots}. A closer look at the example text shows an example of contextual relationship between these two sets, where the 'tree' and 'wolves' are associated by the given text. While a more sophisticated analysis can be more useful to identify both global and contextual semantic relationships, it is a much more difficult task. In our previous work a simple clustering approach is used to exploit this observation [4].

Nevertheless, the global relationships can be used with ease to model the density of lexical semantics in topic segmentation, summarization and keyphrase extraction. With this motivation, it is important to model the global semantic relationships common in the language. Chapter 3 deals with this question and compares two major alternatives. The first being related to structural theory following the work of Halliday and Hassan [2], Miller [22] uses classical relationships that can be defined and stored in Thesauri. The second alternative is related to analytic view of structural theory which follows the distributional hypothesis of Osgood [26], Harris [24] and Firth [25].

2.3 Literature Survey

This section reviews previous work on semantic relatedness, topic segmentation, summarization and keyphrase extraction. Of course this review is distilled in a way to concentrate on algorithms that use some form of lexical cohesion and lexical semantics. However, since most of the research on these problems can be tied to lexical cohesion by some means, it covers the most significant state-of-the-art algorithms.

The presentation is organized in the same order with the chapters of the Dissertation. First related work on semantic relatedness is introduced. This is followed by the literature on topic segmentation, summarization and keyphrase extraction.

2.3.1 Semantic Relatedness

Semantic relatedness is concerned with building a function that is able to correlate with global relations between word pairs. If it is a similarity function, it should report high values for well-known synonym pairs, but report low values if a human judge would classify the pair as unrelated. This problem is highly related with the problem of lexical cohesion and also with lexical semantics.

Algorithms for semantic relatedness can be categorized according to their source of information. These algorithms will be presented in three main categories, where the first category is based on co-occurrence statistics calculated from corpora, the second category is based on dictionaries, taxonomies or ontologies. Finally, the third category uses links available in web based resources such as Wikipedia.

2.3.1.1 Distributional Hypothesis Based Methods

From a theoretical point of view, different theories have common ideas pointing to a correlation between the semantic relatedness of terms and their distribution in a large corpora. The distributional hypothesis which argues that semantically related words appear in similar contexts can be attributed to Harris [24].

One of the earliest uses of the term semantic space is in psychology by Osgood et al. [26]. In their work, they describe an experiment on human subjects, where each participant assigns scores to polar word-pairs to describe the semantic properties of a concept. The scores are between 0 an 7 recorded by a scale as in Figure 2.6. The semantic space is the Euclidean hyper-space formed of concepts as the points and the scale of polar words are the dimensions. For example the concept "mouse" can be differentiated from "mountain" using the polar word pair "small-large". However Osgoood et al.'s methodology requires to define the dimensions manually which is an arduous work expected from human subjects.

Another attempt of applying these ideas was made by Rubenstein and Goodenough [35]. In their work they built a dataset formed of word pairs and obtained judgements from 51 different subjects. Each subject assigned a score for each word pair in this list, where a score of 4 represents highly synonymous and a

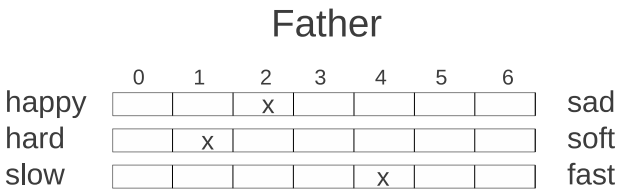


Figure 2.6: Example of Osgood's concepts and polar words differentiating the concepts.

score of 0 represents semantically unrelated. Rubenstein and Goodenough have investigated if there is a correlation between contextual similarity and semantic similarity. Building a corpus of 4.5 million words, contexts are identified for each word, where they define the context as sentences, meaning that for a word in the test set, all other words appearing in the same sentence are considered as a member of the word's context. Representing these contexts as sets, they investigated the correlation between the human synonymy judgements and context overlap. The context overlap is defined in a set theoretic way, and the number of shared context words are used for comparing two words' contexts.

In information retrieval the distributional hypothesis is used to characterize and retrieve documents from a large collection. The bag-of-words model defines a document as a set of words and neglects the order of words [36]. In this method, the context observed in the analysis is the document. Documents are characterized by the words forming them, and a relevancy to a query is formulated as the similarity of word distributions. The seminal work of Deerwester et al. [37] proposes the use of Singular Value Decomposition (SVD) in this bag-of-words model to improve the performance of document retrieval.

Landauer and Dumais [8] also use the term-by-document occurrence matrix to build the semantic space for semantic relatedness between words. It is the first instance of using SVD to reduce the dimensionality of the semantic space. In their evaluation using the near synonymy questions of TOEFL, this method achieves better results than the average non-native college applicant. Hyperspace Analogue to Language (HAL) [38] uses the term-by-term matrix to build the semantic space and does not perform any dimension reduction for inferring semantic relatedness. A comparative study investigating different parameters and co-occurrence context definitions in full-dimensional space (without any dimension reduction) is

by Bullinaria and Levy [39]. Also the work of Terra and Clarke [40] investigates additional information theoretic measures in full-dimensional semantic space.

Rapp [7] combines the ideas of using term-by-term matrices and SVD dimension reduction and reduces the term-by-term matrix using SVD. The method implemented in this research is based on Rapp's work. Until very recently comparisons between the full-dimensional semantic spaces and reduced spaces have been lacking in the literature. Bullinaria et al. [41] also investigate the work of Rapp [7] in more detail and compare the performance of full-dimensional and dimension reduced semantic spaces. While their work and the experiments in this research overlap in certain respects, there are distinguishing differences in both the tasks used in evaluation and the parameters investigated. In the next chapter, these models will be formally defined and elaborated further.

The use of search engines for semantic relatedness is proposed by Cilibrasi et al. [42]. In their work the semantic relatedness of words is formulated by the number of documents retrieved from three queries, queries for the words individually and as a whole combined with an OR operator. It can be argued that this method is a variant of closed-corpus methods described above which uses a significantly larger corpus, for instance, the index of Google search engine.

In contrast to bag-of-words assumption made on all the above research, there is an increasing interest for integrating syntactic information to the general distributional hypothesis. Pado and Lapata [43] use syntax patterns to define the context of words. In their work instead of simply using a co-occurrence window, a set of syntactic patterns are used to define the context. While this is an interesting idea, its applicability in languages that does not have syntactic parser is limited. Also the computational costs of building syntax trees limit the corpora size. Although integrating syntax is a technique that promises to avoid the noise introduced from the loose definition of context as in word based windows, we have opted for using large corpora.

An issue challenging the evaluation of the algorithms for distributional hypothesis is that the effectiveness of the methods depends on two factors: the algorithm applied, and the corpora used. In the experiments a common comprehensive corpora is used making it possible to focus on investigating the first factor, i.e. the algorithms used.

2.3.1.2 Ontology Based Methods

Dictionaries are built by linguists as a common source of lexical semantics. In its most simplistic form a dictionary composed of words and their definitions. Kozima and Forogori [44] use the Longman Dictionary of Contemporary English to build a semantic network. In their algorithm the words used in the definitions are transformed into links between the words. This transformation can be considered as calculating the similarity between the word definitions. This idea is later applied to WordNet glosses by Pedersen et al. [45].

The Roget's Thesaurus is also used as a knowledge base for semantic relatedness research. It is formed of a category structure grouping words. Morris and Hirst [46] use Roget's Thesaurus to model the semantic relatedness between words, using the overlap of categories each word is a member to. Jarmasz and Szpakowicz [47] use the hierarchy of the categories to model semantic relatedness and achieve promising results.

The WordNet project [22] is a semantic network formed of different relationships between the words and their meanings. It contains different relationships commonly referred to as classical relationships [21]. The structure of WordNet and variety of connections in the network attract a substantial interest in this knowledge-base. In the next chapter different semantic relatedness measures are explained in more detail.

2.3.1.3 Link Based Methods

In search engine domain, the links between web pages are exploited extensively to improve the results [48]. The links between web pages point to both importance and a coherency between documents. Due to its Website nature, Wikipedia makes use of links extensively. Articles contain links to each other and to the category hierarchy.

Strube and Ponzetto [49] use the category hierarchy in Wikipedia to measure semantic relatedness. Their method is based on the paths between the articles in the hierarchy. The distance between articles containing the words are used to calculate semantic relatedness. Gabrilovich and Markovitch [50] also exploit both

the links and content of Wikipedia to model the semantic relatedness between words. The concepts identified by the titles of Wikipedia articles are compared to each other by observing the paths connecting their articles. Similar methods are applied by Milne and Witten [51], Mihalcea and Csomai [52].

The effectiveness of these algorithms are adversely affected by their ability to map a word or concept to the articles. While occurrence of a word in the text of the article is not enough to associate the word to the article, using more refined textual content such as titles of the articles limits the coverage of the semantic space.

It should also be noted that the category structure in Wikipedia is not actually a hierarchy, but a large graph containing cycles. In the category structure there are cross cutting categories, for example "Events in 1980" groups otherwise unrelated articles together. Since Wikipedia is a collaboratively built encyclopedia with voluntary editors and there are no restrictions for creating links/assigning categories, there is noise in its structure that should be tackled explicitly.

2.3.2 Topic Segmentation

Topic segmentation task tries to decompose a text into segments discussing thematically different topics. Ideally identifying a topic requires a detailed coherence analysis. However as a clue cohesive ties can also be exploited. A plausible methodology for topic segmentation is by modelling the lexical cohesion for the text in search for disruptions in their density.

This idea has been dominant in the algorithms attacking topic segmentation. Naturally how lexical cohesion is modelled determines the effectiveness of algorithms. On the most simplified level lexical cohesion can be modelled by a function of word repetitions. However, word repetition can be both misleading and over simplification of lexical semantics. Another alternative is integrating the classical relationships present in WordNet for topic segmentation. While the spectrum of lexical semantics exploited increases by using classical relationships, it is still limited as only the strongly related word pairs are considered. In my algorithm, instead of relying on these pre-defined relationships, the whole spectrum of semantic relatedness is used.

This section groups related work in topic segmentation in three categories. First category presents algorithms based on observing the word repetition in text. The second category discusses methods using classical relationships in WordNet. Finally, a recent methodology which learns probabilistic topic models from training data, and tries to segment the text by associating the segments to the topic models.

2.3.2.1 Word Repetition Based Methods

Youmans [53] tracks the first uses of terms in the text, and assumes that topic boundaries should be on the concentration points of the first uses of words. It is possible to criticize this approach as the first use of terms will concentrate mostly on the beginning of the document, and therefore will suffer on long documents.

Hearst [54] introduces the TextTiling algorithm that combines two scores for gaps between adjacent text block pairs. TextTiling uses the cosine similarity between two adjacent text blocks and combines it with the average number of new terms introduced in the two blocks. A low value in this score points to a boundary.

Salton et al.[55] assume topic segments as text parts with strong internal relationships, disconnected from other adjacent parts. With this assumption topic segmentation is treated as an optimization task searching for the set of topic boundaries that maximizes the similarity within each segment requiring a strict integrity within each topic. Similarity matrices are common in such works that build on this idea. Choi [56] starts with a similarity matrix built using cosine similarity. He argues that cosine similarity is unreliable in short texts such as sentences, thus he uses a rank based transformation to avoid this problem. Using the rank based scores for each sentence, he obtains a segmentation by a divisive clustering algorithm. Ji and Zha [57] use a technique called anisotropic diffusion on the sentence similarity matrix to enhance the patterns of lexical cohesion by removing noise. Then, they process the modified similarity matrix with a dynamic programming algorithm to come up with the final segments. The distributional properties of lexical cohesion can also be exploited using statistical methods. Utiyama et al. [58] introduce a language model that defines the probability of a segmentation given the terms in the segment.

LCSeg algorithm [11] builds on lexical chains that are mostly utilized with WordNet classical relations [9, 4, 5]. Lexical chains are basically the sequence of related words spanning the text. In their definition the semantic relations are neglected and solely word repetitions in the text are considered. The start and end points of lexical chains are used to model the topic segments. It is also important to note that they use a supervised classification scheme which requires training.

Malioutov and Barzilay [59] model the text as a graph, where nodes are sentences and edges are their pairwise cosine similarities. Topic segmentation is formulated as the normalized cut problem in this graph, which maximizes the density of segments (inter-similarity) and minimizes the intra-similarities between segments. Although normalized cut is a NP-Complete problem in general graphs, since in topic segmentation the segments have a natural ordering a Dynamic programming solution can be formulated.

2.3.2.2 WordNet Based Methods

Lexical chaining approaches of Stokes et al. [10] can be considered as an extension of Youmans [53] research. Their work simply extends the idea of tracking the change in terminology by integrating the semantic relationships between words to create chains of related terms. The concentration of chain start and end positions point to a topic shift. The summarization algorithms based on lexical chains [9, 4] implicitly segment the topics in the text with WordNet relations. However these methods are not evaluated in the topic segmentation task explicitly.

Jobbins [60] model the topic change as a trough in a lexical cohesion function enriched with collocation, reiteration and lexical cohesion relations. Instead of only processing the right context, the DLCA method proposed in Chapter 4 processes both the left and right contexts of each text block. This enables DLCA to observe both a low level of lexical cohesion as the topic on left ends, and a high level of lexical cohesion formed with the topic on right.

2.3.2.3 Topic Model Based Methods

Eisenstein and Barzilay [61] define topic segments by probabilistic models with Dirichlet distribution. The parameters of the probabilistic models are inferred from a training corpus. In their model they also integrate the cue phrases able to reflect continuation of topics such as "because" and "However". The cue phrases are also inferred from the training set.

Brants et al. [62] use probabilistic latent semantic analysis [63] to define the topic segments with respect to topics that are learned by fitting the word distributions in the text to a probabilistic model. The probabilistic model defines the occurrence of words as a three level probability, where the first level is the word, hidden variable topic is in the middle and the third level is the sentence. The probabilities for individual words and topics are learned by an expectation maximization procedure.

Misra et al. [64] further investigate the topic model based segmentation algorithms by the use of Latent Dirichlet Allocation (LDA). LDA is trained with a similar document set (in fact a portion of the test data), to build the topic models observed in the domain. Then, the topics in the LDA model are fitted (the probability of generating words in the segment using the topic in LDA is greater than the other topics) to different possible segmentations using a dynamic programming algorithm.

Despite the fact that it is not a probabilistic topic model, the follow-up work of Choi [65] which uses SVD based LSA on sentence-term matrix is related to this category, since the reduction in LSA transforms and associates words in the semantic space similar to the notion of topics in LDA or PLSA. Although SVD is used in both Choi's and our algorithm, two algorithms differ significantly. In their work the SVD is applied to sentence-term matrix to form a representation similar to topic models. On the other hand in this research the SVD is applied to term-by-term matrix built from a background corpora in order to explicitly model semantic relatedness. Bestgen [66] criticizes Choi for building the semantic space from a corpora including the test documents. Bestgen shows that inclusion of the test data in semantic space greatly improves the results even though the training set size is relatively larger.

2.3.2.4 Evaluation of Topic Segmentation

Topic segmentation algorithms can be evaluated by a test corpora of documents with topic segments explicitly annotated by humans. However the experiments of Hearst [54] shows that this task involves some level of subjectivity as a disagreement between different human annotators is observed. An alternative evaluation strategy builds artificial datasets by concatenating different articles to form a longer document. In the concatenated documents the segment boundaries are simply taken as start and end points of each article.

As a measure of topic segmentation, the recall and precision values for the generated topic boundaries can be calculated with respect to ground truth topic boundaries. The recall value can be defined as the proportion of the number of matches between the boundary sets to the number of topics in the ground truth set. However this evaluation metric is criticized as it does not differentiate between a system that assigns topic boundaries in the middle of a topic and a system that assigns boundaries close to the true boundaries.

Two measures are proposed for measuring the performance of topic segmentation algorithms, error rate [67] and WindowDiff [68]. The error rate P_k passes a sliding window of words or sentences through the document, which is of size equal to the average segment length of the ground truth segmentation. As this window passes through the text, the agreement between ground truth and generated segmentations is calculated. If the two ends of the window are in the same segment in both ground truth and generated segmentation, then it is considered as an agreement. Likewise, if the ends are in different segments in both of the segmentations it is also considered as an agreement. All the other conditions where there is a dispute between the two segmentations, it is classified as an error. The probability of disagreement is the error rate.

Pevzner and Hearst [68] criticizes P_k by pointing to five different problems in measurement. As an alternative, they define the agreement as the difference of number of segments in both ground truth and evaluated segmentations. They show that this reformulation resolves the problems in P_k.

2.3.3 Summarization

Summarization has been an active research area since 1950s. Summarization task could be thought as a two level process: content selection/importance identification and text generation/smoothing extracts. Previous work for these two phases are presented separately.

2.3.3.1 Content Selection and Importance Identification

A summarization system tries to identify significant information that is important enough to be in the summary. The way we write documents, how we form the content model, and how we emphasize certain content are a phenomena. There are no strict rules, but there are clues that could be exploited to identify important topics and ideas. Summarization research investigates these different clues. It is not possible to claim that any of the features that are used in summarization yields the best results for all text genre.

2.3.3.1.1 Methods Using Position in Text Authors tend to follow some patterns on positioning the important content. Although this depends on the genre and domain of the text, a general belief is that important content is usually positioned in the first sentences. In fact, a very simple and surprisingly successful method for summarization is the selection of the first sentences in text. Brandow, Mitze and Rau [69] have achieved very good results in news articles by selecting the first sentences as summaries. Edmundson [70], Kupiec, Pederson and Chen [71], Teufel and Moens [72] all experimented with similar algorithms. They report that this simple technique gives the best results in news articles and scientific reports. As a matter of fact in Document Understanding Conference 2004, the baseline algorithm which simply extracts the first sentences, has been one of the best scoring algorithms when the target summary is limited to 75 characters.

Lin and Hovy [73] provides an extensive research for deriving the optimum position policy for different domains. They report that different text genres have different focus positions.

2.3.3.1.2 Methods Using Cue-Phrases and Formatting Some phrases are used to emphasize their importance in text, and these phrases are called **bonus phrases**. Some clue phrases reflect that the sentence is not important, and these phrases are called **stigma phrases**. *"significantly"*, *"in conclusion"* and *"last but not the least"* are few examples of bonus phrases while *"hardly"* and *"impossible"* are examples of stigma phrases. Teufel et al.[72] use cue phrases on science articles while Kupiec et al.[71] and Edmundson[70] use cue-phrases to improve existing summarization systems. Exploiting the formatting features like bold words, headers could also improve the summarization performance. Edmundson[70] and Teufel et al.[72] have shown that simple heuristics taking advantage of format features improves the success of the summarizer. Overlap between the sentences and the titles, bold phrases could be used as a clue for importance.

2.3.3.1.3 Methods Using Word Frequency Luhn [74] claims that important sentences contain unusually frequent words in the text. This has not been proven in any research. In fact, word frequency decreased the performance of some summarization systems. Edmundson [70] and Kupiec et al.'s [71] indicate that integrating word frequency to their summarization systems decreased the accuracy of the summarization system. However, word repetition by itself is a lexical cohesion type and there are lexical cohesion based summarization systems that reported successful results. Using word frequency by itself is not proven to be a powerful clue. Some systems takes advantage of word repetitions with information retrieval techniques, but the theory behind these algorithms is more sophisticated and we preferred to classify them as lexical cohesion based summarization systems.

2.3.3.1.4 Methods Using Coherence Much of the research on Coherence based summarization is focused on Rhetorical theory. Marcu's method [75] is an example of coherence based summarization. Marcu uses rhetorical parsing to model the discourse structure in the text. He models the discourse structure of the text using a tree like structure. From local structures to whole text, all relationships between clauses are determined. Forming this tree like structure takes advantage of cue phrases. Figure 2.7 shows an example from Marcu [75] and Figure 2.8 shows the discourse structure for this text.

[With its distant orbit $_1$] [50 percent farther from the sun than Earth $_2$] [and slim atmospheric blanket, $_3$] [Mars experiences frigid weather conditions. $_4$] [Surface temperatures typically average about -60 degrees Celsius (-76 degrees Fahrenheit) at the equator$_5$] [and can dip to -123 degrees C near the poles. $_6$]

Figure 2.7: Text fragment to demonstrate coherence based techniques

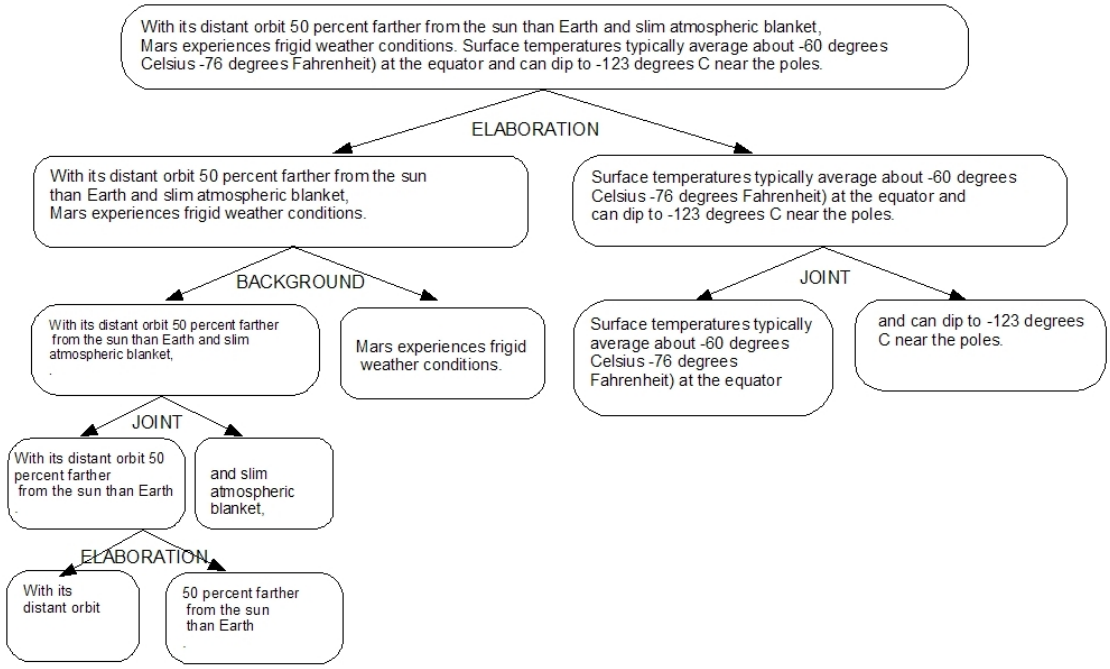


Figure 2.8: Example Discourse structure for the text in Figure 2.7

From the discourse structure, Marcu derived a scoring function for each unit depending on relation types and depth of the tree below each node. Marcu's work has achieved good results and is considered as one of the best summarization algorithms available. However, building discourse trees is a difficult problem. Performance of building the discourse tree structure is questionable. This method is blocked by the difficulties in modelling the coherence structure.

2.3.3.1.5 Methods Using Lexical Cohesion Radev et al. [76] attack automated summarization problem using information retrieval techniques. They use vector space model and clustering to find the central and salient sentences. They use weighted vectors of $tfxidf$ values to represent sentences. **TF** is **term frequency** and IDF is **inverse document frequency**. IDF is the frequency of the word in all documents in the corpus. Note that this approach depends on

word frequencies, so they are only taking advantage of word repetition. Word repetition is one of the lexical cohesion types. Erkan [77] improved the performance of the summarizer by introducing a Google's Pagerank [48] like algorithm for the selection procedure. This summarization system is a part of MEAD summarization toolkit [78] and is a milestone in automated summarization research literature.

Lexical chains are structures for modeling lexical cohesion computationally. They are sets of related words. Halliday [2] presents one of the first works on lexical cohesion. Morris and Hirst [46] discuss an algorithm for building lexical chains. Hirst and St.Onge presents [79] the first algorithm where lexical chains are built using WordNet. They use lexical chains to detect and correct malapropisms[1]. Barzilay [9] presents her lexical chaining algorithm and uses lexical chains to extract summaries. Her algorithm has achieved good results in evaluations. Usually, in algorithms using lexical chains, text units that are traversed by the strongest lexical chains are selected. Following Barzilay's algorithm there have been many lexical cohesion based summarization techniques. Silber and McCoy [80] present an efficient summarizer based on lexical chains. Their algorithm is focused on improving the running time of lexical chaining algorithm. Brunn et al. [81, 82] propose a different sentence selection procedure using lexical chains.

2.3.3.2 Text Generation, Text Compression and Smoothing

Ideally, a summarization system should interpret the text, transform it into a semantic representation and generate the summary from the semantic representation. Interpreting the text is a hard problem. Extensive domain knowledge is required for interpretation. Some researchers tried to fill some predefined templates to create summaries, by treating summarization as information extraction problem [83]. However, this approach is too domain specific and it is not possible to generalize it.

Paraphrasing or reducing the sentences extracted by extractive summarization systems could provide more coherent and shorter summaries. Knight and Marcu [84] present a text compression algorithm. Their work uses probabilistic models

[1]Malapropism is the unintentional misuse of a word by confusion with one that sounds/spells similar

and describes an EM (expectation maximization) algorithm to reduce sentences to shorter ones using syntactic parse trees. Their algorithm is also able to fusion multiple sentences into one.

Mani et al. [85] define a summary revision system which takes in an extract and produces a shorter and more readable version for it. Their system tries to resolve dangling references. Carbonell and Goldstein [86] describes a system called Maximal Marginal Relevance (MMR). Their metric identifies similarity between sentences and represents the repetition in the summary.

Barzilay and McKeown [87] describe a sentence fusion algorithm, which is a text-to-text generation algorithm. This algorithm is very important in the sense that it can paraphrase sentences. With such a tool, it is possible to convert extracts into abstracts, without understanding the text. Their algorithm takes in similar sentences and outputs a fusion of these sentences.

2.3.3.3 Evaluating Summarization Systems

Evaluation of summaries is a hard task as summaries are subjective. Different people would write different summaries for the same document. Evaluation of summaries is a research area by itself. Evaluation methodologies are divided into two main categories. Intrinsic evaluations try to measure the quality of the summary, by defining quality metrics for the summary text. For example, an intrinsic evaluation of selected content's importance is usually done by comparing system generated output summaries to model summaries written by humans. Evaluation is done by measuring the overlap between the model and the automatically extracted summary. ROUGE [88] is such an algorithm. Coherence of the summary is usually evaluated by human judges as there are no automatic evaluation methods for coherence.

Extrinsic evaluations are done by using the summaries in different tasks. For example, human annotators use the output summaries to categorize documents. Accuracy of the humans determine the quality of the summaries. Mani et al. [89] describe an extrinsic evaluation methodology based on usefulness of the system summaries.

2.3.4 Keyphrase Extraction

Keyphrase extraction algorithms typically score each candidate phrase with a keyphraseness scoring function, and sort the candidates accordingly. This function is implemented by either a supervised or an unsupervised learning algorithm. Supervised keyphrase extraction algorithms train a keyphraseness function automatically from observations in a corpus, whereas unsupervised keyphrase extraction algorithms depend on scoring functions built on assumptions and observations. Our keyphrase extraction algorithm is a supervised learning algorithm that uses QPP features.

GenEx [90] is a supervised keyphrase extraction algorithm formed of two components: an extractor and a genetic algorithm. The extractor is a text-processing tool controlled by parameters and rules. For example, the aggressiveness of the stemmer and maximum number of the terms allowed in an extracted keyphrase are two parameters of the extractor. GenEx uses a genetic algorithm and a training set to find the most suitable parameter values for a domain. The population of the genetic algorithm is formed of parameter sets representing a configuration of the extractor. The fitness function is the precision of the extractor executed with the processed parameter set/configuration. The output of the genetic algorithm is the set of rules suitable for the corpus domain and genre. GenEx uses the frequency and first occurrence position of terms in order to score each phrases importance.

The Kea algorithm uses Naive Bayes to learn a keyphraseness probability from the in-document features distribution in the training data. The outline of the Kea algorithm is identical to the outline of the system used in our experiments. Frank et al. [7] report the results of Kea and GenEx [6] to be statistically indifferent. I also use Naive Bayes in order to learn the keyphraseness probabilities of our QPP features.

Recent research aims to improve the effectiveness of keyphrase extraction by integrating additional features such as those exploiting the structural patterns of a document, syntactic patterns of phrases, semantic knowledge, and the relationships between extracted keyphrases. For instance, syntactic features are able to eliminate candidate phrases that are unlikely to be keyphrases (as in the example

of the phrase machine learning introduces, which ends with a verb, and is uncommon for a keyphrase). Hulth [91] investigates the effectiveness of using part of speech (PoS) tags in keyphrase extraction. She evaluates several candidate phrase extraction strategies with and without PoS tags. Using a rule induction method, PoS tag patterns for keyphrases are learned. Hulths experiments indicate that keyphrases have common PoS tag patterns. Furthermore, Barker et al. [92] use a chunker to detect noun phrases using simple syntactic patterns, and assign a score to each phrase depending on the tfxidf value of each phrases head noun. The in-document features are used to evaluate phrases extracted from the document by a noun phrase chunker. Recent work including ours, uses syntactic filters in finding the candidate keyphrases.

The cohesive ties between keyphrases are exploited in keyphrase extraction by using external knowledge obtained either from thesauri or from statistical information extracted from corpora. Turney [93] notes that there should be cohesion between the extracted keyphrases. He measures the pairwise semantic relatedness between two keyphrases by mining the results of a search engine. The cohesion between the keyphrases produced by Kea is reclassified with a second classifier using the cohesion features. This method depends on the output of Kea and in-document features. For extracting keywords, Ercan and Cicekli [5] use the WordNet thesaurus [22] to integrate semantic relationships between phrases and features extracted from the lexical chains of the original document. Thus their method is not able to handle keyphrases of length greater than one, and is limited to keyword extraction. Nguyen et al. [94] integrate both PoS tags and structural features in the feature set of Kea, where the distinguished structural properties of research articles are important cues for keyphrase extraction. They use the occurrence distribution of phrases with respect to the sections of the processed article. For example, a phrase appearing in the title or abstract is more likely to be a keyphrase than a phrase appearing only in the middle of the document. This method is designed specifically for research articles, and is not a generic solution.

Mihalcea et al. [95] model the text as a graph, where the vertices are terms appearing in the given text, and an edge exists between two terms if they co-occur within a distance. Keyphrases are extracted using a social ranking algorithm called TextRank on this network, which is based on the PageRank algorithm [16]. The phrase co-occurrence graph is usually very sparse, especially in short documents where term frequencies are low. Recently, Wan et al. [96] have enriched

this graph by integrating co-occurrence statistics observed in similar documents, that is, the nearest neighbours of the processed text (NN-TextRank). They evaluate their algorithm using news articles that are shorter than research articles, and report improvement over TextRank and a baseline algorithm that scores phrases using tf*idf values. Although both QPP features and NN-TextRank use a background corpus in order to handle short documents, these algorithms differ both in their methods and in the features applied.

Chapter 3

Semantic Relatedness

Lexical cohesion analysis can be used in a range of tasks, such as malapropism detection [19], summarization [4, 9], topic segmentation [10, 11], information retrieval [36]. In lexical cohesion analysis, the relationships between words are used to model the discourse and topics discussed. In its most simplistic form, word repetition can be used to model the document's content [54, 36]. This can be generalized by considering semantic relationships in a knowledge-base like Thesauri[9, 4, 10, 11]. However, mostly these methods only use strong relationships like synonymy but neglects weaker ones. In a way the analysis is constrained to only a small portion of the available data. As an alternative, semantic relatedness measures that can be calculated for any two pairs in the vocabulary can be used in such tasks.

In this chapter the semantic relatedness (SR) is defined with a discussion of how it should behave as a function. Following this discussion, automated methods for SR functions are introduced in two categories. In the first category WordNet Thesauri based methods are introduced. In the second category, semantic spaces are defined and methods to build these are discussed. Given the algorithms from both categories, the effectiveness of these methods with different variations are evaluated using Word Association, WordNet classical relationship mapping and TOEFL near synonymy questions tasks. Among these, the WordNet mapping task is first introduced in this research, and promises to be a more stable and comprehensive evaluation method compared to Word Association and TOEFL near synonymy questions.

3.1 Semantic Relatedness Methods

Semantic relatedness is a measure used to quantify the level of the relationship between lexical items, but is neglectful of how they are related. The measure does not try to identify the type of the relationship, but only its strength. Formally semantic relatedness can be defined as a symmetric function $SR(w_i, w_j)$, which returns its maximum value when the two words are maximally related in meaning and returns its minimum value when the two words are completely unrelated. Note that this definition is still vague as how and to what extent words can be related to each other is not clear and is subjective. From a statistical point of view, the ground truth in relatedness can be defined as the average of human judgements, i.e. word pairs that are commonly identified as related by humans should get high scores. Mostly word pairs denoting concepts with similarities can be classified as related, but relatedness is not simply limited to similarity.

The need to distinguish similarity and relatedness is observed for antonym word pairs, e.g. "black" and "white". Since these words have opposite meanings, they should be considered as dissimilar concepts. However, in the context of semantic relatedness, these terms are considered as semantically related with a strong relation. From this aspect, semantic relatedness as a measure should be perceived as different from semantic similarity. Nevertheless, if it is possible to enumerate all semantically related and similar words of a given word, probably the set of related words would be a superset of similar ones.

A computational model to measure semantic relatedness requires a prior knowledge-base, which can provide information about the use of the words in different contexts. The knowledge-base should somehow encapsulate the average human judgements about word pairs. This research explores two such knowledge-bases, where the first is a manually built electronic Thesauri and the second is a semantic space built using the statistics gathered from a raw text corpora. WordNet based semantic relatedness measures are introduced in different works since 90's [19, 97, 98, 99, 100]. There is also extensive research on semantic relatedness measures built using statistics gathered from raw text corpora [40, 39, 38, 8, 6, 7]. In this work methods, classified under both of these categories are evaluated and compared. Literature lacks such comparisons, as most of the previous works focus on a single category. The experiments in this research are valuable as the effectiveness and efficiency of each method is discussed in detail. The tests are

carried out on two different languages, Turkish and English, in contrast to the large body of research on semantic relatedness that focuses on the English language. The experiments and the system built in this research to the best of our knowledge is a pioneering work for Turkish language. Due to the agglutinative nature of Turkish language, it is interesting to investigate if the ideas effective in English are also effective in Turkish.

3.1.1 WordNet Based Semantic Relatedness Measures

WordNet is a network and a Thesaurus manually built by linguists. Given this knowledge base, different measures have been proposed for quantifying the semantic relatedness between two terms. For completeness these measures are defined, and compared in terms of effectiveness and efficiency.

3.1.1.1 Structure of WordNet

Before introducing the SR methods, it is useful to discuss the knowledge-base itself, the structure of WordNet. In the context of WordNet, a word denotes a specific symbolic representation, i.e. spelling or pronunciation of a word. A sense denotes a meaning or concept. In the semiotics literature a word is defined as the signifier and the meaning is defined as the signified. The signifier (word) is the text or sound we use in discourse, and the signified is the concept (sense) which is intended by the signifier.

WordNet is formed of words, which can point to different senses. The relationship between words and senses is many-to-many, where a word can be associated with different senses and a sense can be associated with multiple words. This is a consequence of linguistic phenomena of lexemes, as there is homonomy in which a word can have multiple meanings, and synonymy where the same sense can be represented by different words. Hirst [101] gives the example of "bark" for homonomy, where one of its sense is "the noise a dog makes" and the other is "the stuff on the outside of a tree." Another related term is polysemy, which is used to describe words that have different but related meanings. For example Table 3.1, lists different senses of the word *mouth* defined in WordNet. While the first two meanings are polysemous with each other as they practically refer

ID	Synonym Set	Meaning
1	mouth, oral cavity, oral fissure, rima oris	the opening through which food is taken in and vocalizations emerge
2	mouth	the externally visible part of the oral cavity on the face and the system of organs surrounding the opening
3	mouth	an opening that resembles a mouth (as of a cave or a gorge)
4	mouth	the point where a stream issues into a larger body of water
5	mouth	a person conceived as a consumer of food
6	mouthpiece, mouth	a spokesperson (as a lawyer)
7	sass, sassing, back talk, lip, mouth	an impudent or insolent rejoinder
8	mouth	the opening of a jar or bottle

Table 3.1: Different noun senses for the word "mouth" defined in WordNet

Relation	Examples	In Categories
Synonymy	N: car, automobile, V: run, escape, JJ: black, dim	N, V, JJ, R
Hypoymy	N: car, motor vehicle, V: run, hurry	N, V
Hpernymy	N:car, motor vehicle, V: hurry, run	N, V
Meronymy	N: car, accelerator	N
Holonymy	N: accelerator, car	N
Antonymy	N: dead, living, V: die, be born , JJ: black, white	N,V,JJ
Troponymy	V: walk, march	V

Table 3.2: Relationship types in WordNet and the categories they are defined in

to the same concept, it is harder to classify the third as it has some relation with the first two senses but still refers to a completely different concept. As polysemy pertains some similarities between different concepts, it is harder to identify or even define them when compared to homonomy.

As in any dictionary, the senses are categorized by word classes, namely nouns, verbs, adjectives and adverbs. What differentiates WordNet from a dictionary is its sense graph, where vertices are senses and edges are relationships between these senses. The existing relationships in WordNet known also as classical relationships are defined according to the word classes. As depicted by Table 3.2,

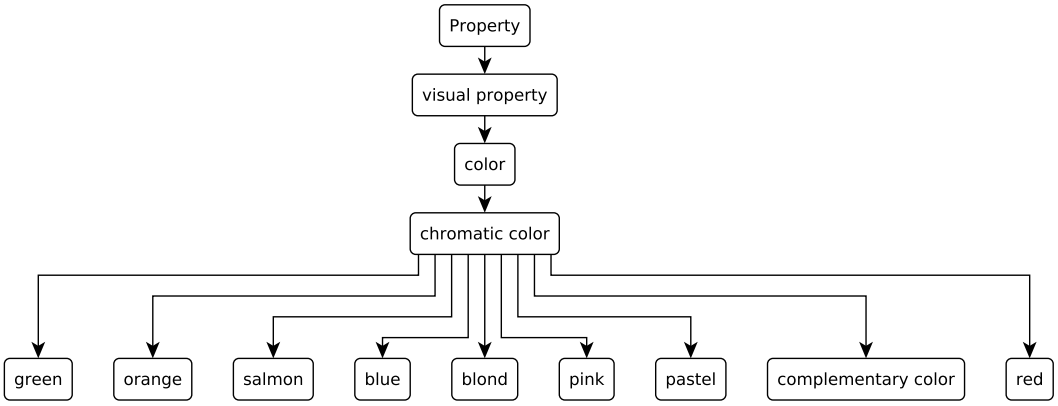


Figure 3.1: The excerpt of WordNet related to the concepts of colors

most of the relations are between nouns and verbs. These two grammatical categories contribute more to the meaning than adjectives and adverbs. However it should be noted that polysemy is more common in verbs, and distinguishing the intended sense in a text is harder when compared to nouns.

The most prominent relationship types in WordNet are hyponymy and its inverse hypernymy. These two relationships basically form a hierarchy which can be thought as a taxonomy of concepts grouping concepts by their attributes. For two senses g_i and g_j, if there is a hyponym relationship from g_i to g_j, then g_i is of type g_j. The inverse of the relation, the edge from g_j to g_i is classified as hypernymy, which indicates that g_j is the type of g_i. For example consider the subgraph from WordNet shown in Figure 3.1, in which the term *red* is a hyponym of *colour* and *colour* is a hypernym of *red*. These relations are transitive, as if g_i is a type of g_j and if g_j is a type of g_k, then immediately follows that g_i is a type of g_k. Following the *colour* example, *colour* is a type of $visual\_property$, and thus *red* is also a $visual\_property$.

The hypernym/hyponym hierarchy, does not contain any cycles. Given the transitive property, if there is a cycle in hyponym/hypernym hierarchy containing a node g_i, then g_i would be both its own hyponym and hypernym, which contradicts with the definition as something cannot be both generalization and specification of the same concept at the same time. For this reason, the graph formed of these relations is acyclic. The senses g_i and g_j which are direct hyponyms of a common g_k sense are usually considered to be highly related to each other.

For example, *red* and *blue* are considered as highly related to each other as they are both hyponyms of *chromatic_color*. These are known as hyponym/hypernym siblings.

Another relationship type between nouns is meronymy and holonymy. Meronymy refers to a more complete concept formed of different parts, while holonymy denotes that a concept is a part of another concept. For example *wheel* is a part of *car*, i.e. *car* is a meronym of *wheel*, and *wheel* is a holonym of *car*. Meronyms are further classified into subgroups as is made from and formed of members. In a way, meronyms are a way to define the attributes of nouns, through a well defined relationship. For each meronym there is a holonym, and vice versa. Meronyms are also transitive, since a part can also be composed of parts. For example $finger$ is a part of *hand* that is a part of arm, thus $finger$ is a part of arm.

The meronyms of a concept g_i, is inherited by any direct or indirect hyponym (through the transitive property). This is more obvious when considering meronyms as attributes of a concept and hyponyms as a more specific type of the same concept. For example *coat* and *hair* are two meronyms of *mammal* and *vertebre* inherits these meronyms, and defines its own meronyms *rib*, *chest* and *tail*. With inheritance it is possible to define shared attributes of concepts, without explicitly defining additional relationships for each level of the hyponym hierarchy.

Another relationship type in WordNet is antonymy, which exists between concepts that have opposite meanings. For example, *black* and *white* or *increase* and *decrease* are antonyms of each other. These concepts are usually dissimilar from each other, but this dissimilarity is usually considered as a high level of semantic relatedness. Antonyms are not specific to nouns, and can exist between verbs, adjectives and adverbs.

The second dominant class of words is verbs. Two specific relationship types exist for verbs, entailment and troponymy. A verb can entail another verb. For example *walking* may entail *stepping*, as *walking* is done through *stepping*. This relationship is similar to the meronym/holonymy relationship in nouns. Troponymy relationship describes a "manner" in an action. For example *marching* is a troponymy of *walking*, where in *marching* the steps taken are regular and the *walk* is fast. While these relationships can be useful in analysis, the number

of such instances is low and contain more polysemy creating a higher level of ambiguity harder to resolve through computational models. Thus, we have chosen to ignore verbs in the algorithms described in this research.

Adjectives are used to describe the nouns, and the concepts usually have a noun form. The relationships between adjectives are scarce, and does not contribute much to the meaning of a text. Adverbs are used to describe the word classes other than nouns. In lexical cohesion analysis using WordNet, adjectives, adverbs and verbs are almost always ignored. In the lexical chaining algorithms of Barzilay and Elhadad [9] as well as Ercan and Cicekli [4] only considers nouns. Following this paradigm, discussion is limited only to nouns.

Within the scope of this research, a WordNet library is implemented. This library is implemented with the objectives of; providing a uniform interface for both Turkish and English WordNets (and potentially other Thesauri), to implement different semantic relatedness functions and to provide a library capable of answering queries fast by storing the sense graph in memory. The English version of WordNet is loaded from the database files of Princeton WordNet [22]. The Turkish version of WordNet is loaded from the XML files provided by BalkaNet Turkish Project [102].

Table 3.3 compares the English and Turkish WordNets by some statistics, hinting to their coverage and completeness. The number of words in English is approximately 5 times larger than Turkish WordNet, and number of senses is about 8 times larger. This alone is an important indicator that Turkish WordNet is far from being complete, and unable to cover many concepts in the language. The relationship density of the sense graph can be calculated by observing the number of relations relative to the number of senses. The average number of relationships for a sense is higher for Turkish. We think that this is also a consequence of incompleteness of Turkish WordNet as more common concepts highly related with each other are present, but less common and domain specific concepts are absent.

Property	Turkish	English
Number of words	15,556	81,426
Number of senses	14,795	117,097
Number of relationships	35,802	211,156
Number of Hyponym/Hypernym relationships	25,794	167,298
Number of Meronym/Holonym relationships	10,008	43,858

Table 3.3: Comparison of nouns in Turkish and English WordNets

3.1.2 WordNet based Semantic Relatedness Functions

With WordNet it is trivial to identify the semantic relatedness of two words that are directly connected to each other through a classical relationship. For example for two synonyms *data* and $information$ it is easy to report a high level of semantic relatedness. However a semantic relatedness function should be able to measure the semantic relatedness of all possible sense pairs. The relatedness function should cover the complete n^2 space, where n is the number of senses. It should be able to differentiate the level of SR between *data* and *chicken* from *data* and *mathematics*.

In a graph, the distance between two nodes is usually calculated using the length of shortest path between them. The length of the shortest path between two senses will be denoted by the function $len(g_i, g_j)$. While using shortest paths is an intuitive and straightforward method to calculate SR, it is not effective. The problem arises as the paths in WordNet are influenced by the taxonomy hierarchy, which is not consistent and homogeneous throughout the graph. For example $len(country, Greece) = 3$ and $len(country, Turkey) = 1$ as depicted in Figure 3.2. A human would probably give a similar score for these two relationships, as they are both related to country through the same relationship. Even further while $len(Turkey, Greece) = 4$, the shortest path to *Jamaica* is $len(Turkey, Jamaica) = 2$. A human would probably expect that Turkey is more related to Greece instead of Jamaica, given its geographic and historical relationships. In this example, since *Greece* is classified as a $European\_country$ and also as a $Balkan\_country$ the path is larger, which is correct when the aim is to classify and categorize the concepts. It could be argued that since hyponymy and hypernymy are transitive an edge exists between *Greece* and *country*, thus the path length should be considered as 1. However in this case both $len(country, Greece)$ and $len(entity, Greece)$ (all senses are hypernyms of *entity*) should be 1, which

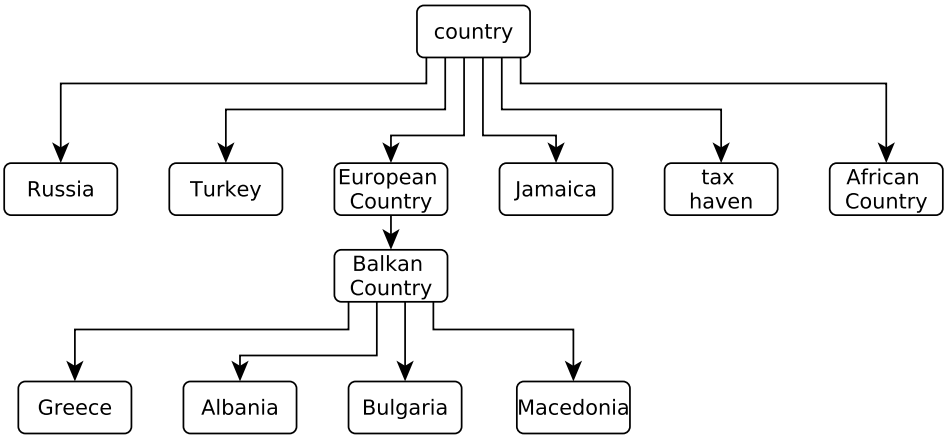


Figure 3.2: The excerpt of WordNet related to the concepts of *country*, *Greece* and *Turkey*

would certainly contradict human judgements.

Budanitsky and Hirst [19] cover most of the WordNet based semantic relatedness functions investigated in our research. For the sake of completeness the semantic relatedness functions used in our experiments will be introduced in this section. The hyponym/hypernym hierarchy in WordNet 1.5 consisted of multiple root nodes, but in 2.1 version there is only one root node *entity*. Budanitsky and Hirst uses WordNet 1.5 and introduces a new virtual root node connected to the multiple roots in order to simplify the discussion. Since WordNet 2.1 is used in our experiments the root node is *entity*.

In order to cope with problems caused by the non-uniform paths in the graph, SR measures are designed to be aware of the level of the senses in the hierarchy. Formally the depth of a sense is calculated using Equation 3.1, where the path from root node (entity) to g_i is calculated in the hypernym/hyponym hierarchy. As in the example of $len(entity, Greece)$, if one of the senses is in higher levels of the hierarchy, strength of the relationship is usually low as the concept is too general.

$$dp(g_i) = len(entity, g_i) \tag{3.1}$$

When comparing two concepts located in two different sub-hierarchies, their

lowest common ancestor in the hierarchy can be used to account for their relatedness. The function $lso(g_i, g_j)$ finds the lowest common ancestor of g_i and g_j. In a way $dp(lso(g_i, g_j))$ measures how specific the relationship of two concepts is. For example, $lso(Turkey, Greece) = country$, $dp(lso(Turkey, Greece)) = 9$ and $lso(chicken, Greece) = object$, $dp(lso(chicken, Greece)) = 4$, which is an indicator of low relatedness between *chicken* and *Greece*.

In a discourse, either in a text or speech, the used words are not mapped to their intended senses. They should be disambiguated from the context which they appear in. One possible strategy is to choose the senses, which are highly related to their neighbouring senses. In the case of semantic relatedness between two arbitrary words, where the context is defined only by two words, this is equal to finding the two senses with the maximum semantic relatedness score. Equation 3.2 is used in intrinsic evaluation of semantic relatedness methods, where $Senses(w_i)$ is the sense set of word w_i.

$$SR(w_i, w_j) = max_{g_k \in Senses(w_i) \wedge g_l \in Senses(w_j)}[SR(g_k, g_l)] \tag{3.2}$$

3.1.2.1 Sussna's Depth Relative Scaling

Sussna [97] uses weights in order to account for the problems in the taxonomy. The weight of each sense is determined considering the depth of the vertices and the relation type. Each node contributes to the edge weight depending on the relation type (hypernymy/hyponymy, meroynym/holonymy or antonymy), and the number of outgoing edges from the node with the same relationship. Equation 3.3 is a node's contribution to the edge weight with respect to a relationship type r, where max_r and min_r are the maximum and minimum scores for r and the $edges_r(g_i)$ is the number of relations leaving the concept g_i. The contribution of the concept decreases as the number of relationship the node participates in increases.

$$wt_r(g_i) = max_r - \frac{max_r - min_r}{edges_r(g_i)} \tag{3.3}$$

The weight value of the edge between g_i and g_j is calculated using Equation

3.4. The notation r' is used to emphasize that for an edge from g_i to g_j although hypernymy, hyponymy, meronymy or holonymy are bi-directional their inverse relationships exist from g_j to g_i. In this notation, if r is equal to antonymy, then r' is again antonymy. The final edge weight is the average vertex contribution, divided by the depth of the concept.

$$weight(g_i, g_j) = \frac{wt_r(g_i) + wt_{r'}(g_j)}{2 \times max\{dp(g_i), dp(g_j)\}} \tag{3.4}$$

The semantic distance value $SR_{Sussna}(g_i, g_j)$ is simply the shortest path between the two concepts $len(g_i, g_j)$. The shortest path is calculated according to the edge weights calculated using Equation 3.4. Since the edges are not uniform and non-negative, Dijkstra algorithm can be used to calculate the shortest path. In order to convert the semantic distance to semantic relatedness (similarity function) the distance is subtracted from the maximum distance possible in WordNet.

3.1.2.2 Wu and Palmer Similarity

Wu and Palmer [98] use $lso(g_i, g_j)$, the most specific common ancestor to calculate the similarity between two concepts. The length of paths from each node to the common ancestor is scaled by the depth of the common ancestor. Semantic relatedness should increase as the depth of the common ancestor increases, and as the length of paths increases from g_i and g_j it should decrease. Equation 3.5 quantifies these motivations by considering the length of the two paths passing from $lso(g_i, g_j)$ and leading to each concept. The measure divides the paths into two parts, where the first is from the root node to the common ancestor, and the second is from the ancestor to the concept. Sum of the proportion of the first part of the paths to the whole path is the similarity.

$$SR_{WP}(g_i, g_j) = \frac{2 \times dp(lso(g_i, g_j))}{len(g_i, lso(g_i, g_j)) + len(g_j, lso(g_i, g_j)) + 2 \times dp(lso(g_i, g_j))} \tag{3.5}$$

3.1.2.3 Leacock and Chodorows Normalized Path Length

Leacock and Chodorow [99] propose to scale the shortest path between two concepts by the maximum depth in WordNet, as shown in Equation 3.6. Since $MaxDepth$ is the maximum length of the shortest path in the hierarchy, the fraction in the logarithm is between 0 and 1. The logarithm function exponentially increases in magnitude and thus concept pairs that are connected to each other by less number of nodes are rewarded by this similarity function.

$$SR_{LC}(g_i, g_j) = -log(\frac{len(g_i, g_j)}{2 \times MaxDepth}) \tag{3.6}$$

3.1.2.4 Resnik's Information-based Approach

The above SR measures base their scoring functions only on the internal structure of WordNet to give a penalty to general concepts located in higher levels of the hierarchy. However the problem of non-uniform paths exists at lower levels of the hierarchy as some concepts are relatively richer in terms of number of senses. One example is the concept of *animals*, since the categorization and taxonomic classification of *animals* is well studied. The hierarchy below the sense *animals* consists of many levels. It is possible to argue that path based methods will give higher scores for concepts that are less common or categorized.

One natural way to account for such problem is by degrading the influence of concepts that are in the hierarchy for the accuracy of the taxonomy, but are rarely used in discourse. Considering the path connecting $fish$ to the root of the hierarchy is formed of concepts such as *aquatic_vertebrate*, *vertebrate* and *chordate* that are important for the taxonomy, but are rarely used in common discourse.

Resnik [100] proposes to integrate the occurrence probability of concepts, in-order to decrease the effect of such intermediate concepts. The number of occurrence for a concept g_i is the sum of all occurrences of g_i and the sum of occurrences of child concepts in the hierarchy g_j. This count can be recursively calculated by first counting all occurrences of each concept, and summing these counts from the leaf nodes to the root. In this way, the final count of the root

concept is equal to N, which is the total number of words appearing in the corpus. The probability of occurrence $p(g_i)$ is the final count of g_i divided by the total number of words in the corpus. In this setting, the root node will have a probability of 1, as it subsumes all the concepts in the hierarchy.

Resnik [100] defines the semantic relatedness between two concepts as the information content of their lowest common ancestor. Equation 3.7 is Resnik's SR function. The similarity of the concepts $Turkey$ and $Greece$ is then the information content of *country*, which is the lowest common ancestor of these concepts.

$$SR_{Resnik} = -log(p(lso(g_i, g_j))) \tag{3.7}$$

One important drawback of this measure is the sensitivity of the scores produced by this measure. The same relatedness values are assigned to any two concepts that have g_i as their common ancestor. Following our example $SR_{Resnik}(Turkey, Greece)$, $SR_{Resnik}(Turkey, European\_Country)$ and $SR_{Resnik}(Turkey, Balkan\_Country)$ are equal to each other as their lowest common ancestor is *country*.

In our experiments the probability values for this and the following information theoretic method are calculated from Wikipedia articles corpora for both Turkish and English. Resnik [100] gathers this statistics from the Brown Corpus of American English [103], which is a corpus of approximately one-million words. The Wikipedia corpus is larger in size, and covers articles from different domains as it is a comprehensive encyclopedia.

3.1.2.5 Jiang and Conraths Combined Approach

Following the work of Resnik [100], Jiang and Conrath [104] criticizes it for ignoring the link structure in WordNet and using it to only determine the lowest common ancestor of two concepts. Instead they propose to use the information theoretic scores as edge weights between the nodes, and formulates semantic relatedness as the length of the shortest path connecting the concepts.

In Resnik's semantic relatedness score, the information content is used to assign a weight to the common ancestor node. On the contrary, Jiang and Conrath assign the weight to the edges. Naturally this weight should consider the information content of the concepts in both ends of the edge. Instead of using $p(g_i)$, the conditional probability $p(g_i|par(g_i))$ is used, where $par(g_i)$ is the parent of g_i. In the same manner as Resnik's method, the occurrences are calculated from a general text corpora using Equation 3.8

$$p(g_i|par(g_i)) = \frac{p(g_i \& par(g_i))}{p(par(g_i))} = \frac{p(g_i)}{p(par(g_i))} \tag{3.8}$$

Note that since every occurrence of g_i is considered also as an occurrence of its parent, the probability of occurring both g_i and $par(g_i)$ is equal to the probability of occurring g_i. The edge weight is the information content of this probability, which is calculated by Equation 3.9.

$$wt(g_i, par(g_i)) = -log(p(g_i|par(g_i))) = log(p(par(g_i))) - log(p(g_i)) \tag{3.9}$$

Jiang and Conrath [104] integrate the edge density and depth scaling to the weight by Equation 3.10. The number of edges leaving a node is calculated by $E(par(g_i))$ and $\bar{E}$ is the average number of edges. The parameters α and β control the contribution weight of depth scaling and density scaling.

$$weight(g_i, par(g_i)) = \left(\beta + (1-\beta)\frac{\bar{E}}{E(par(g_i)}\right)\left(\frac{dp(par(g_i)+1}{dp(par(g_i))}\right)^{\alpha} wt(g_i, par(g_i)) \tag{3.10}$$

3.1.2.6 Efficiency of WordNet based Measures

Different semantic relatedness measures using WordNet, are mostly derived from the paths defined in the sense graph. Table 3.4 lists the SR functions and their computational complexity. The complexity of the algorithms are dominated by the graph search algorithm used to find the shortest path between nodes. The

Method	Graph Algorithm	Complexity
Sussna's Depth Relative Scaling	Dijkstra	$O((\|E\| + \|V\|)log(\|V\|))$
Wu&Palmer Similarity	BFS	$O(\|E\| + \|V\|))$
Leacock&Chodorows Normalized Path Length	BFS	$O(\|E\| + \|V\|))$
Jiang&Conraths Similarity	Dijkstra	$O((\|E\| + \|V\|)log(\|V\|))$
Resnik Semantic Similarity	BFS	$O(\|E\| + \|V\|))$

Table 3.4: Comparison of the complexity of WordNet based SR methods

complexity of algorithms using modified edge weights instead of uniform cost edge weights use Dijkstra's algorithm to find the shortest path. When compared to linear algorithms using BFS, this is computationally more expensive.

In practice, especially for English the cost of Dijkstra is not feasible for algorithms that require thousands or even millions of similarity calculations. In practical high level tasks such as topic segmentation this increases the running time of the algorithm significantly. For these reasons, it may be necessary to execute the Dijkstra algorithm once for each word in the text, and calculate shortest paths leaving the node from a single execution of Dijkstra.

3.1.3 Corpora Based Semantic Relatedness Measures

Corpora based semantic relatedness measures depend on the distributional hypothesis, which dates back to 1950's [24, 25]. Distributional hypothesis states that words occurring in similar contexts are semantically related. While this is both intuitive and can be observed as lexical cohesion in daily life discourse, the tools and methods to exploit this idea is still an active research area.

The main motivation behind corpora based semantic relatedness methods depends on the lexical cohesion phenomena. In different contexts the semantically related terms are expected to appear together. If co-occurrence probabilities are calculated from a large corpora, the term pairs consistently appearing together or with the same set of words are expected to be semantically related. As with any stochastic method, better approximations of the real data distribution should be based on sufficiently large number of observations.

Vector space models (VSM) are used in Information retrieval, since they are

both fast and can accommodate large number of observations. VSM considers each context in a $|V|$ dimensional hyperspace. When context is defined as documents, the document-term vectors are used. An important thing to note is that each document-term vector is sparse as usually only a small portion of V appears in a document. Sparsity increases significantly with $|V|$. Data structures for sparse data enables VSM to efficiently tackle large data sets. This is an essential attribute for co-occurrence analysis since an accurate model of semantic relatedness must be based on a large corpora.

When using VSM, the words are viewed as points in a hyperspace, where the location of points in the hyperspace is organized by the usage statistics gathered from raw text corpora. If the distributional hypothesis is valid, then in this hyperspace words with similar meanings should be located near to each other. The term semantic space is used in order define such a hyperspace. This following section presents a technique to build a semantic space from a raw text corpora.

3.1.4 Building the Vocabulary

In WordNet based methods, the words are manually defined in the Thesauri by linguists. However in automated methods the words that will form the model are not known prior to the analysis. Naturally, the first step is to identify the words that will define the hyperspace used in further analysis.

Words in a dictionary may appear in discourse in their derived forms. The inflection affixes are grammatical constructs used to define the singularity/plurality of the concept used, or to link the concept to the other components of the sentence. Even though it is possible to consider different derived forms as separate words, this will result in a very sparse matrix and large vocabulary size.

Derivational affixes on the other hand are used to convey new meanings from a word, or to change the part of speech of a word in a context (e.g. transforming a noun to adjective). In Section 2.2.1 common affixes are defined for both Turkish and English. The problem of determining the correct base form of a word involves ambiguity, which must be resolved by considering the grammatical construction of the sentence. Morphological disambiguation is a task studied in NLP, which requires a cost to analyse each sentence. Even though morphological

disambiguation can improve the effectiveness of semantic relatedness, it is hard to scale the computational cost to very large corpora. In other words, there is a trade-off between morphological accuracy and corpora size. Furthermore, for the Turkish language morphological disambiguation tools are not publicly available, and requires further research or implementation effort. For these reasons, we have chosen to experiment with larger corpora and used simpler dictionary construction strategies.

For the English language two different dictionary construction strategies are evaluated. The first method simply does not fold-in any words, and considers all word forms appearing in the corpora as a different word. Second method uses the Porter stemming algorithm [34], which simply strips common inflectional suffixes from each word occurring in the corpora. With this stemming strategy it is possible to map a word to a wrong base form. However this strategy can still be beneficial as it is possible to map majority of words to correct base forms.

For the Turkish language three different stemming strategies are explored. The first method simply does not fold-in any words, and considers all words appearing in the corpora as a different word. The second uses a suffix stripping method [105], removing common inflectional suffixes from each word. Third method uses a lexicon based morphological analyser Zemberek[1], and finds the most commonly used base word, stripping all derivational and inflectional suffixes.

In information retrieval, some words are removed from the dictionary as they are functional words (e.g. determiners or pronouns) and do not provide any useful information in analysis. This applies to semantic relatedness methods as well, as a stop word will co-occur with many different words and may create noise in the data observed. Even if stop-words do not have a negative impact on the effectiveness of the system, removing them will improve efficiency as the number of dimensions will decrease and more importantly the sparsity of the matrix will increase. Stop-words are common, mostly grammatical words that can co-occur with almost any word in the vocabulary. In fact in English one of three words is a stop-word on average. In order to clearly explore the effect of removing stop-words, the semantic spaces with stop-word removal and without stop-word removal are considered.

[1]Available at http://code.google.com/p/zemberek

Even though Wikipedia articles are checked and edited by volunteers, there are many typos, spelling errors, uncommon proper nouns and some uncommon phrases such as nicknames of contributors. For this reason the number of unique words in Wikipedia corpora is more than millions, where majority of these only occur once or few times. Furthermore, it is not possible to observe a pertinent association for seldom occurring words, and thus are not useful in the analysis. Removing the least frequent words from the dictionary removes words that are mostly typos, and degrades the vocabulary size $|V|$ by few orders of magnitude. While this removal procedure enables us to experiment with larger corpora, it does not degrade the effectiveness of the system as it is revealed by the experiments.

3.1.5 Building the Co-occurrence Matrix

Document-term matrices are used in search engines, to retrieve documents, where the similarity between the document vectors (rows of the matrix) and the vector of an issued query are calculated to return the most similar document. For semantic relatedness the column vectors of the same matrix can be used, where the semantic relatedness between w_i and w_j is equal to the similarity between their vectors. However, a document can be of varying length containing hundreds to thousands of words, possibly formed of sections covering different subjects. This in return creates a noise for the relatedness problem. Landauer [8] proposes to overcome this problem by truncating each document, and retaining only the first 2,000 characters (on average 151 words). While this degrades the noise introduced from the whole document, it does not utilize the corpora. Smaller contexts such as sentences or paragraphs can be used instead of documents, however the number of dimensions increases as does the computational demands of the algorithms.

An alternative method proposed by Burgess and Lund [38] uses term-by-term matrices instead of term-by-context matrices. Semantic relatedness models are built using a co-occurrence matrix C, which is formed of raw co-occurrence counts for word pairs. The number of co-occurrences of w_i and w_j is denoted by C_{ij}. Term-by-term matrix C is symmetric, and is of $|V| \times |V|$ dimensions. The number of dimensions can be reduced by removing infrequent words from the vocabulary. The context used by Burgess and Lund [38] is based on a sliding window passed through the corpora, and C_{ij} is incremented whenever w_i and w_j appear in the same window. While sentences or paragraphs can also be used as the context,

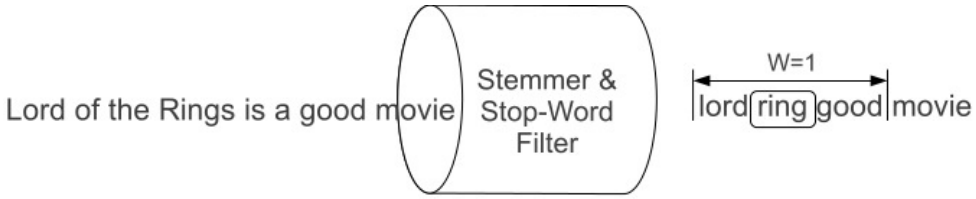


Figure 3.3: Filtered word stream and sliding window centered on the word *ring*

the variations in their lengths can create additional noise in the observations.

Given the vocabulary V, each text document is processed by a filtered word stream as shown in Figure 3.3, which uses a rectangular window of size 1. The filtered word stream first tokenizes the text to words, removing all the punctuations and other symbols. The words are first converted to lower-case and stemmed with one of the methods described in Section 3.1.4, and only non-stop words are included in the stream, and others are filtered. The window operates on this filtered stream, and as in the figure *lord* and *good* occur in the sliding window centred on *ring*, thus the co-occurrence values for *ring-lord* and *ring-good* are incremented. The window slides to the next word, and calculates the occurrences of each word in the stream.

The sliding window technique is flexible and some variations can be explored. The window can be symmetric including words appearing in both left and right contexts or it can be asymmetric including only the left context or the right context. Another variation can use a decaying factor where the co-occurrence strength degrades as the distance between the center word and neighbour word increases. Since in our experiments no significant improvement was achieved only the results of symmetric sliding window are reported.

The size of the sliding window, denoted by W, is a factor in the effectiveness of the system that should be investigated. Different values of W are explored with the experiments, as W increases the density of the co-occurrence matrix increases, as well as the memory and computation demands of the algorithm increases.

3.1.5.1 Weighing the Co-Occurrence Matrix

Given the raw co-occurrence matrix, it is possible to quantify semantic relatedness between words using vector similarity functions. However, raw co-occurrence counts are misleading since the deviation in their values are high and they are biased towards more common words that have a higher probability of occurring with any other word. In order to avoid this problem, the co-occurrence counts should be normalized by the total number of occurrence of both co-occurring words.

Following this motivation, the C matrix is converted to the weighed co-occurrence matrix A, which is symmetric and is of the same dimensions as C. Since the weighing function is used to determine the association level between two words, statistical tests evaluating the association strength are extensively used in the literature. The probability $p(w_i, w_j)$ is tested with an alternative hypothesis that considers $p(w_i)$ and $p(w_j)$ to be independent. Where N is the total number of words in the corpora, the probabilities are calculated by the following equations.

$$p(w_i, w_j) = \frac{C_{ij}}{N} \tag{3.11}$$

$$p(w_i) = \frac{\sum_j C_{ij}}{N} \tag{3.12}$$

One such measure is the Pointwise Mutual Information (PMI). It primarily is a hypothesis test, where the co-occurrence probability is evaluated by the null-hypothesis that assumes independence for each word's occurrence. PMI is negative when the co-occurrence of the two terms can be explained by luck, i.e. randomly choosing the two words independently. If PMI is a high positive number, than it points to a strong association between the two words that can not be explained by an independence model. A variation of PMI is Positive PMI (PPMI), which only retains positive PMI scores and simply replaces negative values with zero. This variation is desirable in terms of computational efficiency as it decreases the density of A. In the experiments only PMI is reported as PPMI produces very similar results.

$$PMI(w_i, w_j) = log(\frac{p(w_i, w_j)}{p(w_i)p(w_j)}) \tag{3.13}$$

Another measure first introduced by Landauer [8] for document-by-term matrices and also used by Rapp [7] is Entropy based weighting function. The information content of the co-occurrence probability is multiplied by the Entropy of the second term.

$$A_{ij} = log(1 + C_{ij})\Bigg(- \sum_k p(w_i, w_k) log(p(w_i, w_k))\Bigg) \tag{3.14}$$

The weighed A matrix can be directly used as a semantic relatedness measure with the cosine similarity function. In the literature these two weighing functions are commonly used, however no other work compares their effectiveness in both dimension reduced space and full-dimensional space.

3.1.6 Dimension Reduction

Following the work of Landauer [8] Latent Semantic Analysis (LSA) using Singular Value Decomposition (SVD) became a popular tool in Information Retrieval research. The most interesting property in LSA based information retrieval is that it promises to resolve problems caused by the use of synonymous words in different documents. For example in a document d_1 the term *data* is used, while in another document d_2 the word $information$ is used, the query $information$ is not able to return d_1. After applying SVD and reducing the dimensions the LSA is able to retrieve d_1, as LSA is able to form higher degree associations. Second important property of LSA is its ability to remove noise from the document-term matrix leaving the most significant document-term relationships.

The core of LSA is SVD, which is a method from Linear Algebra that can be applied to any matrix. SVD is a decomposition, in which the original matrix A is decomposed to three matrices as shown in Equation 3.15, where r is the rank of matrix A. The matrix U is formed of row singular vectors in its rows, and the matrix V contains the column singular vectors. The singular vector matrices U and V are orthonormal matrices, where the row vectors are orthogonal to each

other. The Σ matrix is a diagonal matrix, which contains the singular values in its diagonal. Most SVD tools return the singular values in reverse natural order. The singular values indicate the degree of variance along its associated singular vectors. Multiplication of these matrices yields the original matrix A without any loss in data.

$$A_{m\times n} = U_{m\times r}\Sigma_{r\times r}V^T_{r\times n} \tag{3.15}$$

There is a relationship between SVD and eigenvalues and eigenvectors of the AA^T matrix. The eigenvalues of the AA^T matrix are squares of the singular values of A matrix. Through this relationship it is possible to use SVD in different problems [106]. Given the singular values and vectors, it is possible to project the original matrix to a lower dimensional space. Only retaining the top k singular values and vectors, and truncating the matrices to dimensions of k reduces the number of dimensions from $|V|$ to k. This reduction technique is referred to as SVD truncation. The reduction is able to produce a projection, which minimizes the Frobenius norm distance between the full-dimensional A and k dimensional A'_k matrix. In other words SVD is able to find the best approximation of the matrix A in a k-dimensional space, in terms of $\|A - A'_k\|$, where $\|...\|$ stands for the Frobenius norm. This property hints on SVDs ability to remove noise as it retains the most pertinent information in the matrix.

For semantic relatedness SVD is applied to the weighed co-occurrence matrix and truncated to k dimensions. The value of k is a parameter of the system, in which no automated method exists to determine its optimal value. In the experiments different truncation values between 300 to 3200 are explored. Note that since the resulting matrix is dense, larger dimensions are not feasible in analysis because of the large space requirements of the matrices.

While there are no conclusive findings explaining why SVD is able to establish semantic relationships between the words when applied to a term-by-term or term-by-document matrix, it is possible to hypothesize that when word vectors are represented in a lower dimension, there is a folding in effect. For example consider two vectors v_1 and v_2 of dimension n that have some common non-zero dimensions, but are not identical. The uncommon dimensions of the vectors contributing to the dissimilarity of the matrix. When n is large consider the

minimal hypercube containing all the term vectors. In the n dimensional space the hypercube is sparse containing large areas without any vectors, as there are many word combinations not observed in the corpora. However when these vectors are projected into a lower dimensional space, by pertaining the most significant information in the original vectors, these empty areas decrease and it is possible to observe stronger relationships between the words. Note that with this hypothesis, I am arguing that the ability of establishing latent meaning is a consequence of noise removal combined with reducing sparsity of the matrix. Related to this hypothesis, in the projected k-dimensional space it is possible to observe higher order associations between words using cosine similarity. For two words w_i and w_j that never co-occur with each other, i.e. $A_{ij} = 0$, if they co-occur with common words there is a possibility to project the vectors of these words to similar k-dimensional vectors establishing a direct relationship for higher order associations. While these are hypothesis trying to explain the behaviour of SVD when used in term vectors, it is possible to simply view it as a powerful noise removal technique.

Efficient computations of SVD is a research area by itself. In the implementation the SVDLIBC[2] is used, as it supports sparse matrices and is an efficient and accurate implementation. The implementation uses Lanczos method, which is an iterative algorithm used to calculate the eigenvalues and eigenvectors of the matrix AA^T. It should be noted that, the numerical stability of the SVD calculations can be effective in the results, especially with higher $|V|$ and k values. While Lanczos method is known to be one of the most numerically stable methods for calculating SVD [107], other SVD calculation methods promise to be more efficient in running time and can be calculated incrementally and by distributed algorithms [108, 107]. The relationship between different SVD computation methods and semantic relatedness should be investigated. However in the scope of this Dissertation, I have chosen to neglect the effects of numerical inaccuracies of SVD calculations.

[2]Available from http://tedlab.mit.edu/ dr/SVDLIBC

3.1.7 Similarity of Term Vectors

Given the full dimensional matrix A or the truncated SVD matrix A'_k, the semantic relatedness is formulated as the similarity between the term vectors of two words w_i and w_j. Different similarity measures used in IR and NLP are investigated in prior research [39, 40]. However both in the experiments performed for this Dissertation and in previous research cosine similarity is the most effective similarity function.

Cosine similarity which calculates the cosine of the angle between two vectors is widely used in information retrieval. The cosine similarity exploits the geometric relationship between two vectors. It is essentially the dot product of the unit word vectors. In order to convert word vectors to unit vectors, the vectors are normalized by their Euclidean norms. If the angle between the two word vectors is 0 then its cosine is 1, indicating a high similarity. If the two vectors are orthogonal then the cosine is 0, indicating a dissimilarity between the words. The cosine similarity for a weighted matrix X is given in Equation 3.16, where m is the number of columns in X. In the case of full-dimensional matrix $X = A$ and m is equal to $|V|$, when the SVD truncated matrix is used $X = A'_k$ and $m = k$.

$$cos(w_i, w_j) = \frac{\sum_t^m X_{it} X_{jt}}{\sqrt{\sum_t^m X_{it}^2}\sqrt{\sum_t^m X_{jt}^2}} \tag{3.16}$$

3.1.8 Raw Text Corpora

In the experiments, Wikipedia articles are used for building the semantic space, as it is a general domain corpus covering different subjects. Wikipedia is available separately as both Turkish and English languages. In order to process the Wikipedia articles the database dump of the articles are retrieved from Wikimedia foundation[3]. From each corpora the articles with less than 200 words are removed, as these articles are usually redirections or disambiguation pages without a content. Table 3.5 compares the Turkish and English corpora by some statistics.

[3]Retrieved from http://dumps.wikimedia.org/. For English dump of 01/06/2012 is used, for Turkish dump of 24/05/2012 is used

	English	Turkish
Number of Articles	4,722,440	361,468
Number of Words	2,159,633,497	87,248,254
Average Article Length	457.312	241.372
Number of unique Words	10,597,320	2,159,985

Table 3.5: Comparison of Turkish and English Wikipedia corpora

	0	5	10	15	20
No Stemming	2,159,985	413,703	274,198	211,088	174,396
Lexicon	1,612,708	250,725	161,685	122,001	99,502
Stemmer	1,418,623	231,542	151,161	114.650	93,884

Table 3.6: Number of unique words in Turkish Wikipedia with different morphology analysis and filtering levels

Number of unique words is in the order of millions even if a stemming algorithm is used. In order to reduce the computational cost to a feasible level, words that appear less than the given number of times are removed from the co-occurrence matrix.

Table 3.6 lists the number of unique words occurred in Turkish Wikipedia, where three stemming algorithms and the filter's frequency threshold ψ are given. Table 3.7 lists the percentage of words in WordNet that are present in the corpora. The Turkish WordNet is covered by approximately 87 percent, even when words that appear less than 20 times are removed. Some examples of missing words that are defined in WordNet are; *akselerometre*, *agorafobi*, *darwinist*, *delge* and *dinazorlamak*. The coverage of raw vocabulary (no stemming applied) is lower than the others, indicating that some of the words in WordNet never appears in their base forms, but appears in inflected forms.

Table 3.8 lists the number of unique words occurred in English Wikipedia, where two stemming algorithms and the filter's frequency threshold Psi. Table 3.9 lists the percentage of words in WordNet that are present in the corpora. The English WordNet is covered by approximately 64.84 percent, even when words that appear less than 200 times are removed. When compared to Turkish this coverage is low. The majority of filtered words are archaic, proper names (such as

	5	10	15	20
No Stemming	86.55	82.53	79.40	76.64
Lexicon	92.66	90.62	88.78	87.08
Stemmer	92.78	90.87	89.20	87.57

Table 3.7: Percentage of Turkish WordNet nouns covered by Wikipedia corpora

	0	100	200	300	400	500
No Stemming	10,597,320	282,739	175,616	114,240	103,102	85,230
Stemmer	9,861,046	238,348	144,990	109,203	89,497	76,544

Table 3.8: Number of unique words in English Wikipedia with different morphology analysis and filtering levels

people or city names), uncommon derivations of words or domain specific words that are not common in general domain documents. For example *abampere*, *abarticulation*, *abdominocentesis*, *abstractedness*, *unneighborliness* are absent from the corpora, but are defined in WordNet. Although these are valid and defined words, how valuable they will be in co-occurrence analysis is questionable as they are infrequent.

The size of the vocabulary is important as it determines the memory requirements and running time of the implementation. With a vocabulary of size greater than hundred thousand the memory requirements of the co-occurrence matrix becomes greater than a manageable level, especially with a high value of W the size of the co-occurrence window. Taking the WordNet's vocabulary as a reference, the word removal can be chosen as 20 for Turkish and 300 for English. In both languages both vocabularies are around hundred thousand terms. In Turkish 87 percent of the words in WordNet is included in this vocabulary, and in English this is 62 percent. Through different evaluations it is shown that removing words

	100	200	300	400	500
No Stemming	70.30	62.53	60.92	57.06	54.32
Stemmer	72.26	64.84	62.69	59.34	56.52

Table 3.9: Percentage of English WordNet nouns covered by Wikipedia corpora

from the vocabulary does not degrade the effectiveness, but improves it.

3.2 Intrinsic Evaluation of Semantic Relatedness Measures

Evaluation of semantic relatedness is a subjective task and must be performed by comparing human judgements to the scores of automated methods. This type of evaluation is called intrinsic, as the semantic relatedness itself is compared in its most atomic level, in pairwise word similarities. For intrinsic evaluation, three different tasks are used. For Word Association and WordNet synonym recall tasks experiments are carried out for both Turkish and English language, for the TOEFL synonym questions only English semantic spaces are evaluated.

3.2.1 Word Pair Judgements

The most straight forward method for semantic relatedness evaluation is the task of assigning scores for word pairs. For this task, a set of human subjects are asked to give scores to word pairs. Subjects are asked to assign scores to each word pair between 0 and 4, where 4 represents closely related and 0 represents unrelated. Since this task is subjective, variance between human annotators is expected, especially for word pairs that are vaguely related with each other (scores closer to 2).

The scores of the automated methods are compared to the average of the scores assigned by human subjects. The comparison is performed using Pearson's correlation coefficient calculated as in the Equation 3.17, where n is the total number of pairs in the test set, SRG_i is the mean of ground truth human scores for word pair i, $\overline{SRG}$ is the mean of SRG_i values, SRA_i is the score of the semantic relatedness function for pair i and $\overline{SRA}$ is the mean of the system's scores.

$$cor = \frac{\sum_1^n (SRG_i - \overline{SRG})(SRA_i - \overline{SRA})}{\sqrt{\sum_1^n (SRG_i - \overline{SRG})^2}\sqrt{\sum_1^n (SRA_i - \overline{SRA})^2}} \tag{3.17}$$

The correlation coefficient is ranged between -1 to 1, where a positive score indicates a correlation between the two scores. A high negative correlation coefficient indicates an inverse relationship between the two scoring functions, i.e. whenever one increases the other decreases. Naturally the scores of an effective semantic relatedness method should have a high positive correlation coefficient with the scores assigned by human subjects.

The word pairs are selected in order to investigate different levels of semantic relatedness values, including both highly related and unrelated word pairs. For comparison the average correlation of individual human subject scores to their average scores are reported. Since Pearson correlation coefficients are not additive, the average of correlation coefficients is calculated by first transforming each correlation value to Fisher's z values, and the average of the z values are calculated. The average correlation coefficient is calculated by converting the mean z value back to correlation coefficient. In order to decide on a model, the t significance test is performed on the correlation values as described by Steiger [109, 110].

While word association task is used in the literature [19, 50], it is criticized by its coverage of the semantic space, as the words in the association lists are limited in size when compared to the size of the vocabulary. Since it requires a manual work by human subjects, it is difficult to build larger datasets. Another problem with this evaluation method is its coverage, as it contains either highly related or unrelated pairs, missing pairs that are in-between.

A word pair list and manually associated human judgement scores are first built by Rubenstein and Goodenough [35]. This set is used to evaluate different algorithms in the literature [19, 50]. The list consists of 65 word pairs. An excerpt of the data is given in Table 3.10 with the average human assigned scores. The average human correlation is 0.80 [111], which can be thought as a score representing average human performance in this task. The number of human subjects participated in the experiment is 51.

In order to evaluate the semantic relatedness functions for Turkish, a word pair list similar to Rubenstein and Goodenough [35] was built. In fact, the original 65 pairs are translated into Turkish, and additional 36 pairs are included to increase the number of pairs to 101. 74 human annotators are asked to assign scores for the word pairs. The average human correlation is 0.762, which is close but slightly

TR Word Pair	EN Word Pair	TR Mean Score	EN Mean Score
araba-yolculuk	car-journey	2.95	1.55
bilge-büyücü	sage-wizard	0.89	2.46
birader-delikanlı	brother-lad	2.20	2.41
öğlen-akşam	midday-noon	3.00	3.68
kıyı-ağaçlık	shore-woodland	1.11	0.90
takı-mücevher	gem-jewel	3.79	3.94
sırıtma-delikanlı	grin-lad	0.30	0.88
yemek-meyve	food-fruit	2.42	2.69

Table 3.10: Example Word pairs used in Word Association task with their corresponding average human scores for both Turkish and English.

lower then the inter human correlation for English.

In order to give an idea about the differences in semantic relatedness in the two languages, the correlation between the human judgement scores of Turkish and English word pairs is calculated. The correlation coefficient between the two languages is 0.82, which indicates that although the language is changed semantic relatedness is preserved.

3.2.2 Semantic Space Neighbors to WordNet Mapping

WordNet can be considered as a dataset annotated by humans, as it is formed by linguists manually. Following this idea, the word pairs with high semantic relatedness scores are compared to WordNet direct classical relationships. The recall of WordNet synonyms, hypernyms/hyponyms, meronyms and siblings are calculated for the k-nearest neighbours of words in the semantic space. This not only evaluates the accuracy of the semantic relatedness function, but also provides an ability to investigate the types of relationships that influence the semantic space. Using this method it is possible to investigate if the synonyms of words get a higher semantic relatedness score than direct hyponyms and meronyms or not.

In semantic relatedness scores using a semantic space, the observation is based on the statistics gathered from raw text. A semantic space built from corpus

statistics can be biased towards the most common sense of the word, as it observes more examples for these senses. WordNet stores the sense of words in the order of how common each sense is. Using this information in WordNet it is possible to investigate the percentage of k-nearest neighbours in semantic space to the words related to the most common sense in WordNet.

For a word $w_i \in V \cap WordNet$ let $knn(w_i)$ be the list of top k words, in descending order of $SR(w_i, w_j)$ where $w_j \in V \cap WordNet$ and $w_j \neq w_i$. Note that the k-nearest neighbours are found from the intersection of the vocabulary of WordNet and the semantic space, i.e. the words in semantic space but not in WordNet are excluded. The top k semantically related words are calculated by a brute-force method, which simply calculates pairwise similarity scores and maintains top scoring k neighbours denoted as $knn(w_i, k)$ for each word.

Given the set $knn(w_i, k)$ recall with respect to WordNet classical relationships can be calculated. Since in WordNet relationships are between senses, but in semantic space relationships are between words should be mapped to their intended sense. This task of mapping the words to their senses is called as word sense disambiguation which is a difficult task. However WordNet maintains two data structures mapping both the words to senses and senses to words. Let $Senses(w_i)$ be the list of different senses in WordNet for the word w_i and $Synset(s_j)$ be the list of words used for the sense s_j. These two lists are ordered by how common they are used, i.e. the first word in $Synset(s_j)$ is the most common word used in text for intending the sense s_j. All synonyms registered for a word w_i through different senses can be calculated using the Equation 3.18. When only the first s_j value in $senses(w_i)$ is considered, the synonyms of the most common sense of the word w_i are retrieved, which will be denoted by $Synonyms1(w_i)$. Given that the set R denotes a set of words such as $Synonyms(w_i)$ or $Synonyms1(w_i)$, it is possible to calculate recall using Equation 3.19.

$$Synonyms(w_i) = \bigcup_{s_j \in senses(w_i)} synset(s_j) \tag{3.18}$$

$$Recall(R, k) = \frac{|knn(w_i) \cap R|}{|R|} \tag{3.19}$$

In a similar fashion it is possible to extend this idea to different classical

relationships defined in WordNet, namely; Synonymy, Hyponymy, Hypernymy, Siblings, Meronymy, Holonymy and the union of all these sets. In the evaluations of different semantic spaces, recall values with respect to these relationship types are reported. For the whole dataset, macro average of all the words will be reported. The macro average is calculated as the proportion of all the correctly identified words to all the ground truth related words in WordNet.

In order to have a baseline algorithm providing a reasonable lower-bound for this evaluation method, a completely random method selecting k words for each word in the dataset is also reported. The problem is formulated as an urn experiment, where k balls are selected from an urn filled with $|V|$ balls without replacement and the probability of selecting $|R|$ marked balls is to be calculated. In this formulation R stands for the related words in WordNet and k is the number of randomly selected neighbours in semantic space. This can be modeled by the hypergeometric distribution. In hypergeometric distribution the expected value is calculated by the Equation 3.20. Since the size of $|R|$ differs for each w_i, expected value for each word is calculated. Again the macro average of these random selections are calculated and reported as the baseline algorithm.

$$E(w_i) = \frac{k|R|}{|V|} \tag{3.20}$$

3.2.3 Near Synonymy Questions

A common evaluation task in semantic relatedness research is the use of Test Of English as a Foreign Language (TOEFL) questions [8, 112, 39, 41, 7]. These tests are particularly designed to test the vocabulary and comprehension abilities of students, which learn English as a second language. Since these questions are also solved by humans, it is possible to compare the performance of automated methods with humans.

The test consists of 80 near synonymy questions taken from TOEFL exam, where a word is given along with 4 choices and the student is asked to find the most related word from these choices. Figure 3.4, shows an example question from this test. The effectiveness measure in this test is accuracy, which is simply the proportion of the number of correct answers to the number of questions. The

Query Word: enormously (a) tremendously (b) appropriately (c) uniquely (d) decidedly

Figure 3.4: An example question from TOEFL synonymy questions where the correct answer is "tremendously"

performance of humans who has taken this test is 64% [8].

Unfortunately this test or a similar question set is not available for Turkish. Although there are few similar questions in University Entrance Exam, these questions are mostly analogy questions, where the words are related to each other through their similar attributes, but otherwise they are not related to each other.

3.3 Results and Model Selection

The parameters of the semantic space is an important factor in the effectiveness and efficiency of the semantic relatedness performance. There is a large body of research focusing on evaluating different semantic spaces [40, 39, 19, 41, 7], however still there are important questions to be addressed in the literature. Since the effectiveness of algorithms depends on the raw text corpora used, the results obtained from independent experiments are not comparable with each other. For example in Terra and Clarke [40] experiments are carried out with different weighing functions and similarity functions in full-dimensional semantic space, however these experiments are not comparable with the results of Rapp [7] which uses SVD truncated semantic space, as the raw text corpora used are different. In the experiments carried out in this research, different techniques are investigated and compared with each other in a controlled setting allowing pairwise comparisons between the methods. Also to the best of our knowledge no comparison between semantic space methods, full-dimensional methods and WordNet based methods exists. Our evaluation is detailed and uses 3 different intrinsic evaluation methods and 1 extrinsic evaluation method. Using this evaluation strategy different semantic relatedness measures are compared with each other.

Word Removal ψ	Word Assoc.	WordNet Synonym Recall
5	0.7350	0.4854
10	0.7356	0.4987
15	0.7362	0.5143
20	0.7365	0.5233
25	0.7361	0.5330
30	0.7352	0.5369
35	0.7374	0.5420
40	0.7397	0.5445

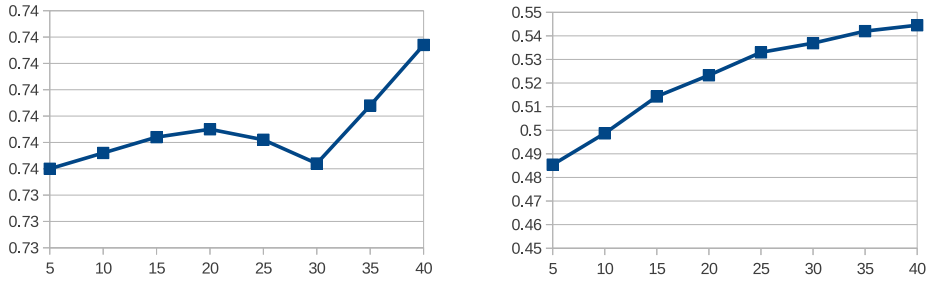
Table 3.11: Results of Word Association and WordNet Synonym Mapping for Turkish Language when ψ value is varied

3.3.1 Effect of Pruning the Vocabulary

In order to reduce the vocabulary size, words occurring less than ψ times are pruned from the co-occurrence matrix. This section investigates the relationship between this removal strategy and the effectiveness of the algorithms in both Turkish and English languages. All the other parameters are fixed to same values in order to investigate the effect of the ψ parameter.

The plot given in Figure 3.5 for the results in Table 3.11 shows the change in correlation values, when the frequency removal threshold ψ varies from 5 to 40. The results are produced by a semantic space using a lexicon based stemming algorithm, truncated SVD with top 400 singular values retained and entropy based term weighting function. A two tailed t-test on the correlation values reveals that these correlation values are statistically indifferent with a confidence factor of 0.99. This shows that removing infrequent words from the vocabulary does not degrade the effectiveness of the system. In fact, the highest word association score is slightly higher when the ψ value is 40, but this is not statistically significant. However when the vocabulary is excessively pruned, i.e. ψ value is increased as high as 500, since the words included in the word pairs list is not in the semantic space, the performance of the system naturally drops.

In the English dataset, the same experiment is performed with a system using Porter Stemmer, stop word removal, truncated SVD with top 400 singular values retained and entropy based term weighting function. Figure 3.6 shows the results



(a) Word Association Values

(b) WordNet Recall of Synonyms

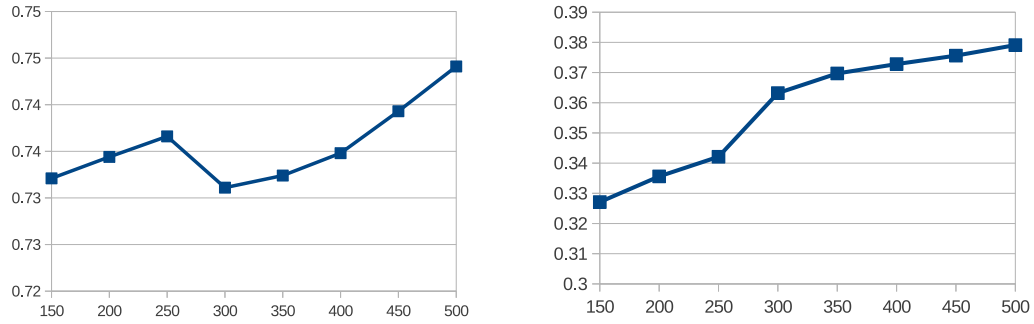
Figure 3.5: Plot of the Word Association correlation values and WordNet Synonym Mapping recall Value in Turkish language when ψ is varied

Word Removal ψ	Word Assoc.	WordNet Synonym Recall
150	0.7321	0.3271
200	0.7344	0.3356
250	0.7366	0.3421
300	0.7311	0.3632
350	0.7324	0.3697
400	0.7348	0.3728
450	0.7393	0.3756
500	0.7441	0.3791

Table 3.12: Results of Word Association and WordNet Synonym Mapping for English Language when ψ value is varied

in Table 3.12, when the ψ values are changed from 150 to 500. Although the results are similar to the results in Turkish experiments, the results obtained with $\psi = 500$ is significantly better than the results obtained with $\psi < 450$.

These results obtained in both languages shows that removal of low frequency words does not degrade the performance of the system, in fact it improves the results. After the removal, the dimensions of the co-occurrence matrix decreases. In the case of no word removal the algorithm would be executed on a square matrix of million dimensions, which is not computationally feasible. However as Zipf's law is able to explain the distribution between word frequencies and



(a) Word Association Values

(b) WordNet Recall of Synonyms

Figure 3.6: Plot of the Word Association correlation values and WordNet Synonym Mapping recall value in English language when ψ is varied

number of words, removing low frequencies greatly reduces this cost. The improvements in terms of effectiveness observed in the results of both languages can be attributed to two different reasons. First reason is related to noise. In the co-occurrence matrix the neighbours of the low frequency terms are relatively few and the patterns observed are based on a small set of instances. It is not possible to determine a strong recurrent pattern between word pairs if one of them seldom occurs in the corpora. Thus, these are merely useful in the analysis, but they create noise, as the observed co-occurrences may as well be the consequence of chance. Second reason is related to the so-called "curse of high dimensionality". As the number of dimensions increases the performance of methods analysing these data degrades as well. When the number of dimensions are high, the sparsity of the co-occurrence matrix increases. As sparsity increases it is harder to get a more reliable estimate through numerical methods. When the number of dimensions is reduced to a more manageable size, the benefits of both SVD and cosine similarity increases.

3.3.2 Effect of Dimension Reduction and Weighting Function

In the literature the term-by-term co-occurrence matrix is used in semantic relatedness measurements both with [8, 7, 41] and without dimension reduction

Semantic Space	Word Assoc.	WordNet Synonym Recall
pmi-full	0.3049	0.3560
entropy-full	0.3342	0.3717
pmi-svd	0.7250	0.4430
entropy-svd	0.7480	0.5190

Table 3.13: Comparison of using different term weighting functions and dimension reduction in English language

Semantic Space	Word Assoc.	WordNet Synonym Recall
pmi-full	0.5248	0.4420
entropy-full	0.4812	0.4414
pmi-svd	0.6776	0.4686
entropy-svd	0.7627	0.6459

Table 3.14: Comparison of using different term weighting functions and dimension reduction in Turkish language

[38, 40, 39]. However, these experiments are not comparable with each other as they use different corpora with different characteristics (corpora size, domain). In the experiments, the difference between models with or without dimension reduction are compared. Furthermore, the term weighting function PMI and PPMI used in the experiments of Terra and Clarke [40], Bullinaria et. al. [39] are compared to the Entropy based term weighting function used in the experiments of Rapp [7] in models with dimension reduction. While recently Bullinaria et. al. [41] compares the performance of SVD truncated semantic space with full dimensional space, they lack the comparison between using PMI term weights and Entropy term weights.

Table 3.14 shows the observed correlation of the judgements of the semantic spaces to human judgement scores and the results of WordNet mapping in Turkish language experiments. Entropy based method produces significantly better scores compared to PMI when dimension reduction is applied, however in the full-dimensional case (without any dimension reduction) it is only slightly better. The difference between Entropy based method and PMI in SVD truncated experiment is significant as the hypothesis of equal means is rejected with a t-test and a confidence factor of 0.99. In the full-dimensional space the difference is not

significant.

Table 3.13 shows the results of the same experiment in English language. In word association problem, when using truncated SVD semantic spaces both of the weighting functions achieve similar correlation values. However in WordNet mapping problem Entropy based weighting function achieves a better recall value. Again in reduced dimensional space models are statistically different when a t-test is performed with a confidence factor of 0.99, but they are indifferent in full-dimensional space.

When considering the results in both languages, it is observed that although PMI and Entropy weights achieve competitive results in full-dimensional semantic space, in the SVD truncated space Entropy based weighting function is significantly better. In Bullinaria et. al. [41] the PMI is used with SVD, while in Rapp [7] entropy based weighting function is used but they are not comparable as the corpora used are different. In document-by-term matrices for information retrieval Dumais [113] reports a similar finding, where three different weighing functions are compared and entropy based weighting function achieves the best results in SVD truncated spaces. The results of PMI weighting scheme in full-dimensional semantic spaces should not overshadow the use of Entropy based weighing schemes in the SVD truncated spaces.

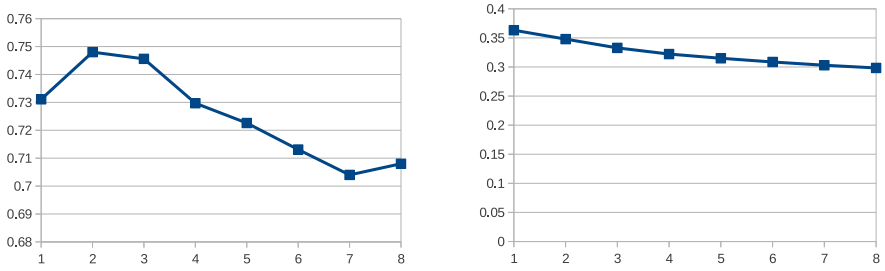
3.3.3 Effect of Co-occurrence Window Size

The window size determines the size of the co-occurrence context, when its value increases the density of the full-dimensional co-occurrence matrix increases. The size of the context window determines the number of observations of the algorithm. From one point of view it could be argued that with a higher window size, it is possible to observe more distant (in terms of number of words in-between) interactions between the words providing more knowledge. From another point of view, this increases the amount of noise introduced to the system.

Table 3.15 shows the results of varying the window size in English language experiment. Although varying the window size decreases the effectiveness of the system in both Word Association and WordNet synonym mapping problems, there is no significant difference between the results with a confidence factor of

Context Window Size	Word Assoc.	WordNet Synonym Recall
1	0.7311	0.3632
2	0.7480	0.3479
3	0.7456	0.3328
4	0.7297	0.3222
5	0.7226	0.3150
6	0.7131	0.3086
7	0.7040	0.3030
8	0.7080	0.2983

Table 3.15: Results of Word Association and WordNet Synonym Mapping for English Language when co-occurrence window size W is varied



(a) Word Association Values

(b) WordNet Recall of Synonyms

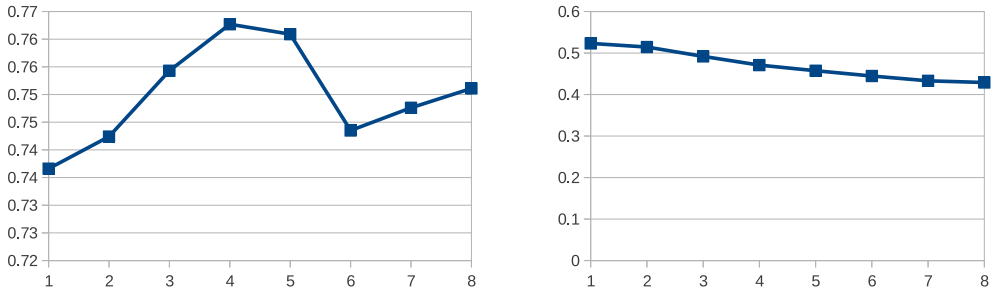
Figure 3.7: Plot of the Word Association correlation values and WordNet Synonym Mapping for English Language, when window size is varied

99% when tested with paired t-test. However as it can be seen from Figure 3.7 there is a decline in effectiveness as the window size increases. This can be attributed to the noise introduced to the model as the observed context gets larger.

Table 3.16 shows the results of varying the window size in Turkish language experiment. When the window size increases, the effectiveness degrades as can be observed from Figure 3.8. In terms of window size the results are similar in both English and Turkish experiments. The slight difference between optimal co-occurrence window sizes can be attributed to the stop word rates in the languages. While in English the number of grammatical stop words are used more frequently, in Turkish these constructs are conveyed through affixes. With more stop-word

Context Window Size	Word Assoc.	WordNet Synonym Recall
1	0.7366	0.5233
2	0.7424	0.5145
3	0.7543	0.4920
4	0.7627	0.4710
5	0.7609	0.4574
6	0.7435	0.4447
7	0.7476	0.4331
8	0.7511	0.4290

Table 3.16: Results of Word Association and WordNet Synonym Mapping for Turkish Language when co-occurrence window size W is varied



(a) Word Association Values

(b) WordNet Recall of Synonyms

Figure 3.8: Plot of the Word Association correlation values and WordNet Synonym Mapping recall values for Turkish language, when window size is varied

removal the effective window size increases in English, for this reason the optimal co-occurrence window in English (3) is less than Turkish (4).

3.3.4 Effect of the Number of Dimensions Retained

In SVD based dimension reduction, the target dimension size is an important parameter. Since the resulting matrices from SVD are dense, the upper bound for its values is constrained by the space complexity and running time complexity of the algorithm. For example with a vocabulary of 100K words and a dimension reduction of 10K dimensions will result in a matrix of 8 GB, which will not

Context Window Size	Word Assoc.	WordNet Synonym Recall
200	0.7138	0.3018
300	0.7382	0.3290
400	0.7480	0.3479
500	0.7463	0.3604
600	0.7522	0.3707
700	0.7657	0.3787
800	0.7563	0.3859
1600	0.7494	0.4114
3200	0.7251	0.4164

Table 3.17: Results of Word Association experiment as correlation coefficients and WordNet synonym mapping for English language under the variation of SVD reduction factor

fit into memory in commodity hardware. While there is no theoretical method for determining the number of dimensions of the latent semantic space, in the literature usually a number between 800 and 200 achieves the best results [8].

Table 3.17 shows the effectiveness of the semantic space in Word Association and WordNet synonym mapping problems in English language. The effectiveness in Word association problem is the highest when the reduction factor is 700, but drops after this value. However in WordNet synonym mapping, this is not the case and recall of synonyms increases even with 3200. In terms of statistical significance, all of the models are statistically indifferent when tested with t-test using a confidence factor of 0.99.

Table 3.18 shows the effectiveness of the semantic space in Word Association and WordNet synonym mapping problems in Turkish language. The effectiveness in Word association problem is the highest when the reduction factor is 400, but drops after this value. However in WordNet synonym mapping, this is not the case and recall of synonyms increases even until reduction is 800. In terms of statistical significance the results of 1600 and 3200 are statistically lower with a confidence factor of 0.99.

In the experiments an discrepancy between the Turkish and English languages is observed. The saturation point for Turkish is lower for reduction factor when compared to English. This is in fact due to the discrepancy in the corpora size. Since the English Wikipedia is larger than Turkish, the signal to noise ratio in the co-occurrence matrix is higher.

Context Window Size	Word Assoc.	WordNet Synonym Recall
200	0.7320	0.4215
300	0.7585	0.4497
400	0.7627	0.4710
500	0.7559	0.4822
600	0.7492	0.4876
700	0.7482	0.4914
800	0.7440	0.4967
1600	0.6924	0.4747
3200	0.6274	0.4159

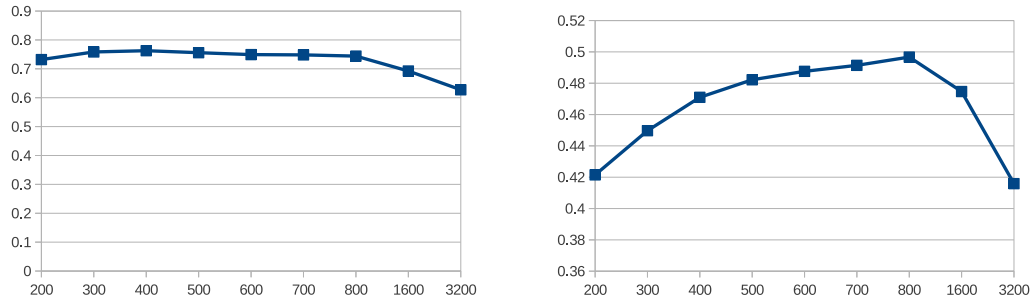
Table 3.18: Results of Word Association experiment as correlation coefficients and WordNet synonym mapping for Turkish language under the variation of SVD reduction factor

3.3.5 Comparison of Semantic Space and WordNet based Measures

Unfortunately since WordNet based measures are extracted from WordNet, it is not reasonable to test them using the WordNet synonym mapping test, as they would achieve almost perfect scores. However it is possible to make a comparison through Word Association and TOEFL near synonymy problems. A further comparison will be presented through extrinsic evaluation with the topic segmentation problem.

Table 3.19 compares the performance of WordNet based methods in English language Word Association problem. The results of the semantic space is calculated using word removal of 500, stemming with stop-word removal, Porter stemmer, SVD truncated to 1600 dimensions, co-occurrence windows size equal to 1 and using entropy weighting scheme.

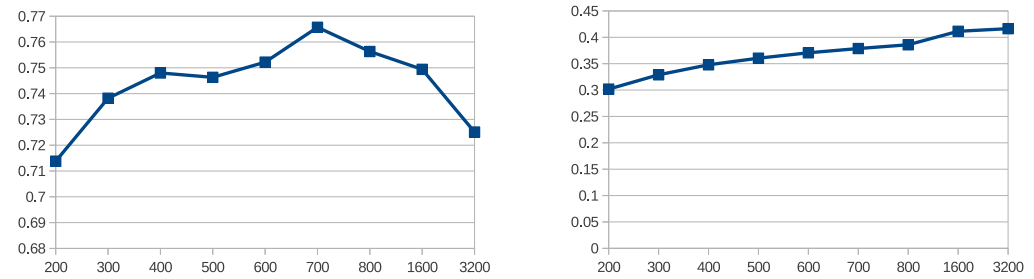
In the Word Association problem, the WordNet based methods achieve significantly higher results with a confidence factor of 0.99. However in the TOEFL synonymy questions, the semantic space achieves significantly better results. With these results it is not possible to confidently decide which method is more effective. While the TOEFL synonym questions is becoming a standard test for semantic relatedness evaluation, I criticize its use as it only contains 80 questions



(a) Word Association Values

(b) WordNet Recall of Synonyms

Figure 3.9: The effect of SVD reduction factor in Turkish language



(a) Word Association Values

(b) WordNet Recall of Synonyms

Figure 3.10: The effect of SVD reduction factor in English language

formed of mostly adjectives and verbs. The classical relationships in WordNet are low in number for adjectives and verbs, thus it is natural for WordNet based methods to achieve worse scores in a dataset dominated by adjectives and verbs. Solely from these experiments it is not possible to conclude WordNet based or semantic space methods are better in semantic relatedness. In the next chapter, another comparison in topic segmentation task will shed more light to this question.

In Turkish no questions similar to TOEFL synonym questions exist. In the Word Association task the results are as low as 0.40. This is mainly due to the fact that the WordNet in Turkish is not complete and there are components in sense graph. For a given two pair a path may not exist between them, as they are

Method	Word Assoc.	TOEFL Synoynmy
Semantic Space	0.7791	0.9367
Wu&Palmer	0.8266	0.3871
Resnik Semantic Distance	0.8416	0.3226
Leacock&Chodorow	0.8578	0.38710

Table 3.19: Comparison of WordNet based methods with Semantic Space model in English language

in different components and thus semantic relatedness can not be calculated. For such cases we have modified the algorithms to return low semantic relatedness score by default. These results shows that with a WordNet that is not complete such as Turkish, it is not possible to use commonly used semantic relatedness functions.

3.3.6 Comparison to the State-of-the-art Methods

TOEFL synonym questions are used in the literature for comparing different semantic relatedness scores. Table 3.20 shows the performance of state-of-the-art algorithms. The best performing algorithm Rapp [7] actually uses the same method but a different corpus. In the experiment Rapp uses the BNC [114] corpus of 100 million words, which is smaller than Wikipedia corpus. An expectation would be to have significantly better results than Rapp's method as a larger corpus is used. However it should be noted that the quality of Wikipedia is lower than BNC, as there are artificial repetitions and patterns common to Wikipedia. For example "external links" appears in most of the articles, or InfoBoxes summarizing different information are repeated in many articles. One word in TOEFL questions is "figure", which is repeated in many articles.

3.3.7 Semantic Spaces and Classical Relationship Types

Using the classical relationships in WordNet it is possible to further investigate the kinds of relationships in the semantic spaces. Figure 3.11 shows the average recall values for each classical relationship for different k values, where k is the

Method	Word Assoc. %	Based on
WikipediaSVD	93.67	Corpus (SVD)
Rapp	92.50	Corpus (SVD)
Matveeva et al.	86.25	Corpus (Full-Dimensional)
Terra and Clarke	81.25	Corpus (Full-Dimensional)
Jarmasz and Szpakowicz	78.75	Roget's Thesaurus
Bullinaria et al.	85.00	Corpus (Full-Dimensional)
Average Human Performance	64.50	Average non-English US college applicant

Table 3.20: Comparison of Wikipedia SVD Truncated to state-of-the-art Algorithms

number of neighbours considered in semantic space. This plot resembles the shape of the logarithm function, in which as k becomes larger, the increase in recall value diminishes. An elbow shape is formed in the plot where the recall value becomes saturated, indicating that words classically related to each other are usually the nearest neighbours in the semantic space. As expected, stronger relationship Synonymy, is more focused on most similar neighbours when compared to weaker relationships like Hyponymy, Meronymy and Siblings. The Siblings are more scattered in the semantic space as the growth of the plot is closer to linear.

Figure 3.12 compares the Synonym frequency of nearest neighbours to random selection. The random selection is linear and when 500 nearest neighbours from the semantic space are selected, only upto 6 percent of the synonyms are covered. On the other hand, the semantic space neighbours contain about 60 percent of the synonyms.

On average when the nearest neighbour of word w_i in semantic space is selected, it is directly connected to w_i about 47 percent of the time. The complete distribution is shown in 3.13. Along with synonyms hypernym/hyponym siblings are the most dominant classical relationship that can be identified by the semantic space, however this is natural as the number of such siblings is relatively large when compared to other classical relationships. The relation of about 53 percent of the nearest neighbours are unknown to WordNet. These nearest neighbours can be collocations (common phrases), related to the words by multiple node paths in WordNet or related through non-classical relationships as in the example of "cop" and "doughnut".

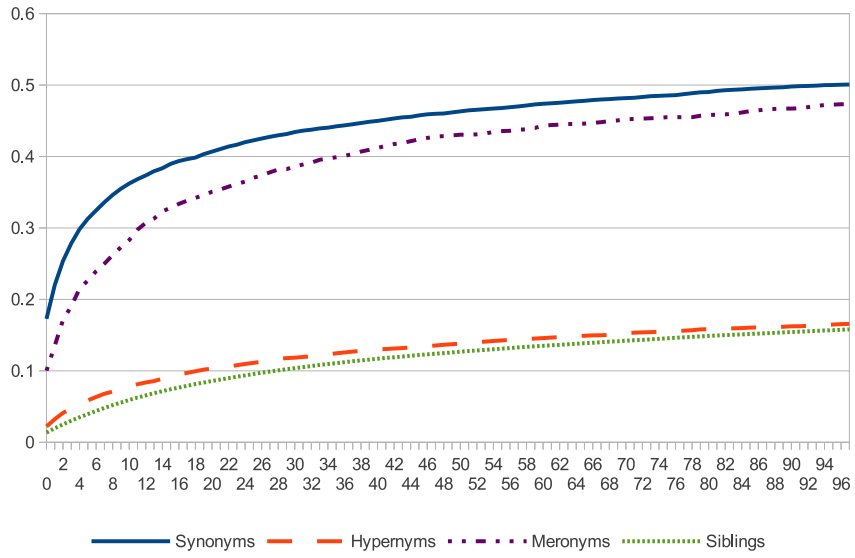


Figure 3.11: Recall of WordNet Relationships versus k-nearest neighbours in Semantic Space

WordNet stores the words and their associated meanings in reverse sorted order of commonality. The most common sense used in discourse for a word is stored as the first sense of the word. Using this information it is possible to investigate if the semantic space is biased towards the most common sense. As expected since the most common sense is observed more in the data, the semantic space is biased towards this sense. For the nearest neighbours, about 86.29 percent of the relationships found in WordNet are related to the most common sense of the word in Turkish experiment. In English experiment this is slightly lower 78.08 percent. Experiments in both languages confirm that there is a bias towards the most common sense in semantic space.

3.4 Discussion of the Results

This chapter investigates different aspects of different semantic relatedness functions built for Turkish and English languages using intrinsic evaluation methods. First it is evident that for a language other than English, which only has an

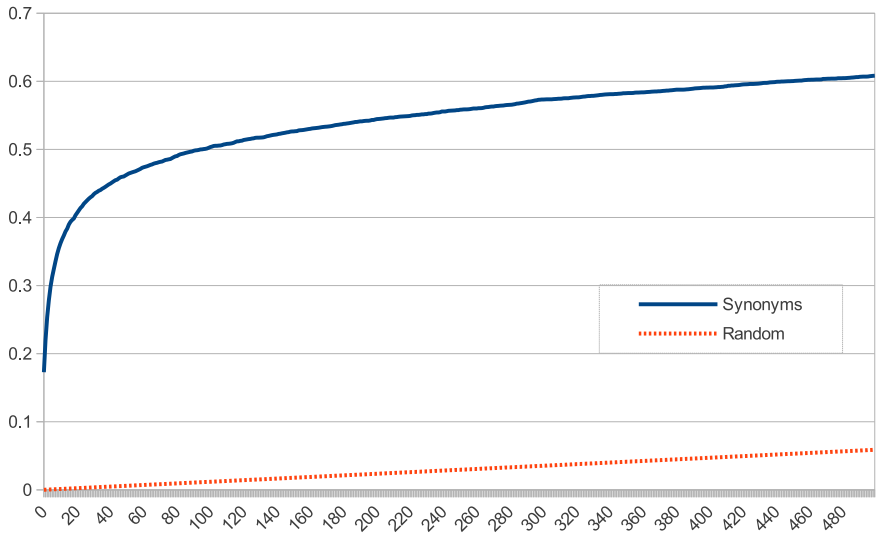


Figure 3.12: Comparison of Random Selection with Semantic Space Neighbours

incomplete Thesaurus, the effectiveness of WordNet based semantic relatedness methods is low when compared to the performance of semantic spaces. In the English experiments, conflicting results are obtained as in Word Association WordNet based methods perform better, but in TOEFL synonymy questions semantic spaces are significantly more effective. However this will be investigated further in the next chapter through topic segmentation task.

In the literature of SVD based semantic spaces there is a shift towards the use of PMI as an edge weighing scheme. However while PMI achieves competitive results in full-dimensional space, its performance is below Entropy based weighing scheme in SVD truncated space. Also the SVD truncated semantic space significantly outperforms the full-dimensional semantic space. Since reduced semantic space is more efficient in terms of memory and computational complexity, in practical applications of semantic relatedness, SVD based dimension reduction is more desirable.

In order to evaluate semantic spaces, common tasks such as Word Association and TOEFL synonymy questions are not robust as they only consider a limited portion of the vocabulary. The results of WordNet based evaluation tends to be more robust as it is possible to observe a stable trend when a parameter's value is varied. Similar evaluation techniques that involve larger number of semantic

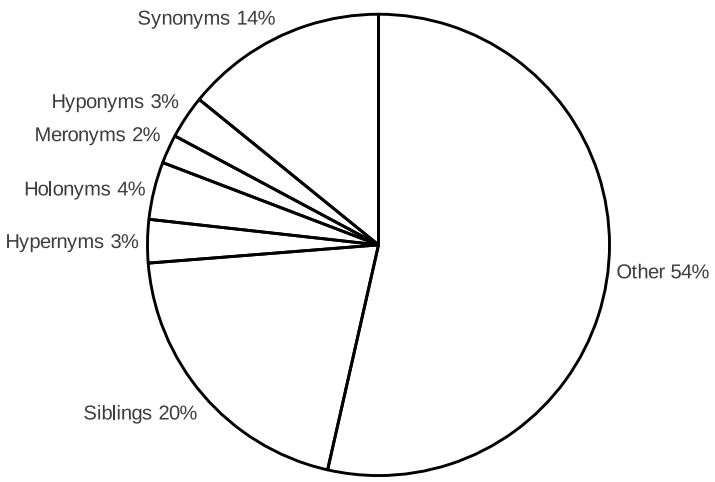


Figure 3.13: Distribution of WordNet Relationships for the Nearest Neighbours in Semantic Space

relatedness calculations should be preferred to compare different methods.

The experiments carried out in both Turkish and English show that there is no major discrepancy between the results, depending on the structure of the languages. From the experiments using stemming, stop word removal, small co-occurrence window size, large SVD reduction factor and Entropy weighing scheme seems to achieve the best results.

Chapter 4

Topic Segmentation

In topic segmentation problem, the flow of the article is modelled, and thematic changes in the document are detected. For example, when ran on the text of this chapter, an ideal topic segmentation algorithm is expected to return the sections of this chapter. Topic segmentation is performed in order to evaluate semantic relatedness measures and use it as an input for the summarization algorithm presented in the next chapter.

While there is a large body of research for topic segmentation, to the best of our knowledge no other algorithm uses semantic relatedness for this task. The semantic relatedness function is used to measure the density of lexical cohesion in consecutive sentence blocks. In this algorithm semantic relatedness of a context is measured for its left and right contexts. If the lexical cohesion is focused heavily on a single direction then a topic boundary is assumed to be on the opposite direction. Using this intuition, two different algorithms are formulated and described. The semantic relatedness function can be any semantic similarity function, as described in Chapter 3.

Next section formally introduces the topic segmentation algorithms developed, followed by the description of the datasets used in the experiments and the results. Different semantic relatedness functions described in Chapter 3 and the segmentation performance of the system built in this research are compared with the state-of-the-art algorithms in the literature.

4.1 Semantic Relatedness Based Topic Segmentation Algorithm

A topic segmentation algorithm takes a document of m text blocks (sentences or paragraphs) and outputs t intervals that span the whole text. The number of topics t is either determined from the document or given as a parameter. In order to avoid confusion, hereinafter the text blocks will be referred to as sentences. However, in the presence of paragraph boundaries they can be used instead of sentences.

In text segmentation algorithms that process the document by a sliding window, such as TextTiling [54], Choi [56] and Brants et al. [62], the lexical cohesion level between consequent text blocks namely left and right contexts are measured. A disruption at the level of lexical cohesion indicates a topical change. Instead we propose to process the text as a sequence of three different contexts, with the addition of middle to commonly used left and right contexts. Each context is defined as a set of words, and there are semantic relationships between the members of neighbouring sets. In fact, these relationships are defined by any semantic relatedness $SR(w_i, w_j)$ function between the words w_i and w_j. Both WordNet and semantic space based methods defined in Chapter 3 can be utilized. Figure 4.1, depicts this idea of using three contexts. These three sets form a complete tri-partite graph, where wc_i is connected to all the words in L and R. Given this framework two algorithms are presented, where the first algorithm takes in the number of topic segments to be identified as a parameter, while the second automatically determines the number of topics.

4.1.1 Number of Topics is Known

Given a document formed of sentences $\{s_1, s_2...s_m\}$, the algorithm passes a sliding window through the sentences. Let L_i, C_i and R_i be the contexts defined with respect to s_i. C_i is the set of words that occur in sentence s_i. L_i is the **left context** formed of words occurring in any of the sentences $\{s_{i-\Omega}...s_{i-1}\}$, where Ω is the window size. Similarly, the **right context** of the ith sentence denoted by R_i is the set of word that occur in any of the sentences $\{s_{i+1}...s_{i+\Omega}\}$. Since these are defined as sets, a word may occur only once in any of L_i, R_i and S_i,

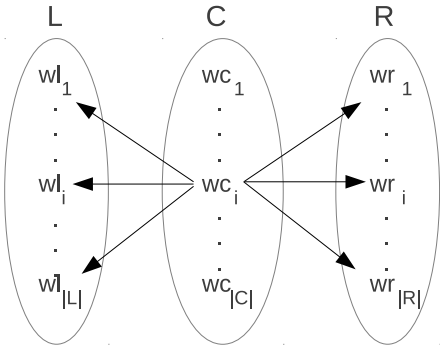


Figure 4.1: Model of three consequent contexts

even though they may appear more than once in their corresponding sentences.

A topic boundary, by definition, is between the last sentence of one topic and the first sentence of subsequent topic. Intuitively the lexical cohesion of these two sentences should concentrate on opposite sides. For each sentence a score reflecting the lexical cohesion change is calculated by comparing C_i with L_i and R_i with a vote casting scheme, where each wc_i casts its vote to left or right contexts. The votes are weighted by the difference of maximum semantic relatedness.values in L_i and R_i contexts. This difference denotes both the strength and direction of lexical cohesion of C_i. Since this difference characterizes the algorithm, it will be referred to as Differential Lexical Cohesion Analysis (DLCA) in the further discussion. Figure 4.2 presents the context score calculation for a sentence s_i with DLCA algorithm. If the lexical cohesion is concentrated towards the left context, then $score_i$ is negative, and it is positive if it is towards the right context.

Figure 4.3 shows a visualization of the scores for a document for both hypothetical and real data. In Figure 4.3(a), expected perfect result of the DLCA method is depicted. If all the lemmas in C_i are associated (i.e. the maximum value of semantic relatedness value) with words in R_i, then the color is purely white. On the other hand, when they are all associated to lemmas in L_i, then the sentence is in black. In such an ideal set, the maximum level of contrast is expected to be in the topic boundary. The intra-topic sentences should slowly become darker as the right context leaves the topic while the left context begins to span the topic. For a long topic that is greater than the window size Ω, both right and left contexts will span the same topic at some point. In this case the

1: $score_i, scoreL_i, scoreR_i \leftarrow 0$
2: **for** each $wc_j \in C_i$ **do**
3: $lMax = max_{wl_k \in L_i}(SR(wc_j, wl_k))$
4: $rMax = max_{wr_k \in R_i}(SR(wc_j, wr_k))$
5: **if** $lMax > rMax$ **then**
6: $scoreL_i += lMax - rMax$
7: **else if** $lMax < rMax$ **then**
8: $scoreR_i += rMax - lMax$
9: **end if**
10: **end for**
11: $score_i += scoreR_i - scoreL_i$

Figure 4.2: Algorithm calculating the sentence context scores based on a voting scheme

semantic relatedness will not have a significant bias to the right or left context. The window size Ω can be selected as a number that is higher than the average topic size.

Figure 4.3(b) shows the visualization of four concatenated news articles. While there are many fluctuations in the semantic relatedness direction within a topic, the topic boundaries exhibit a stable pattern. For example, 7^{th} sentence of the third topic is more related to its left context, which may in fact point to a subtopic within the news article. The contrast, i.e. semantic relatedness difference surrounding the topic boundary, is almost always significant. In order to further exploit this pattern, numerical differentiation of $score_i$ is used. Since $score_i = scoreR_i - scoreL_i$ the numerical differentiation $score'_i$ using two points results in Equation 4.1. The values with a high positive value in $score'_i$ is considered to be a topic's last sentence. Thus, the sentences with the top $score'_i$ scores are selected as topic boundaries by the DLCA algorithm.

$$score'_i = scoreR_{i+1} - scoreL_{i+1} - scoreR_i + scoreL_i \tag{4.1}$$

The $score'_i$ scores of twenty-five concatenated Reuters news articles are shown in Figure 4.4. The article boundaries usually exhibit high values of $score'_i$. In the evaluations, when a cut-off point of 5.22 is used (the dashed horizontal line), a recall of 79% and a precision of 73% is achieved. As it can be seen, most of the real topic boundaries have a high $score'_i$, and only a small amount of them are just below the cut-off point. There is also a small amount of false positives

(a) Semantic Relatedness Visualization of Hypothetical Data

(b) Semantic Relatedness Visualization of Real Data

Figure 4.3: Visualization of semantic relatedness: he set of sentences is represented by rectangles, and the boundaries are denoted by a line. Sentences with high semantic relatedness to L_i are denoted by dark colors whereas high relatedness to R_i is colored with bright colors.

whose $score'_i$ values above the cut-off point.

Calculation of the $score'_i$ values depends only on the local contexts, limiting the space and running time complexity of the algorithm to the number of words in the local context. This is an advantage when compared to the algorithms in the literature that require similarity calculations between all sentences and use dynamic programming [56, 58, 61, 59]. In other words, this algorithm is more suitable for segmenting long documents containing thousands or even millions of sentences. This local analysis is also performed in TextTiling [54], which only uses word repetition for lexical cohesion analysis.

An important problem concerns short sentences which only contain none or few content words. At the extreme level when a sentence without any content word is occurred, its $score'_i$ value will be high as $score_{i+1}$ will be relatively higher than $score_i$ which is 0. This can result in small segments of only 1 short sentence. In order to avoid this problem a smoothing factor is integrated to the algorithm by including the immediate left and right neighbours of the sentence to calculate left skewed $score_i^{(L)}$ and right skewed $score_i^{(R)}$ scores. The score of s_i is the difference of $score_{i+1}^{(L)} - score_i^{(R)}$, which not only resolves short sentence problems but also potentially increases the difference for a true topic boundary.

The window size Ω can ideally be determined relative to the average segment size. However, using a large Ω value does not degrade the performance of the algorithm, as the maximum operation is robust to noise created by including sentences that span a further away topic. Selecting Ω as a value larger than the average segment size achieves good results.

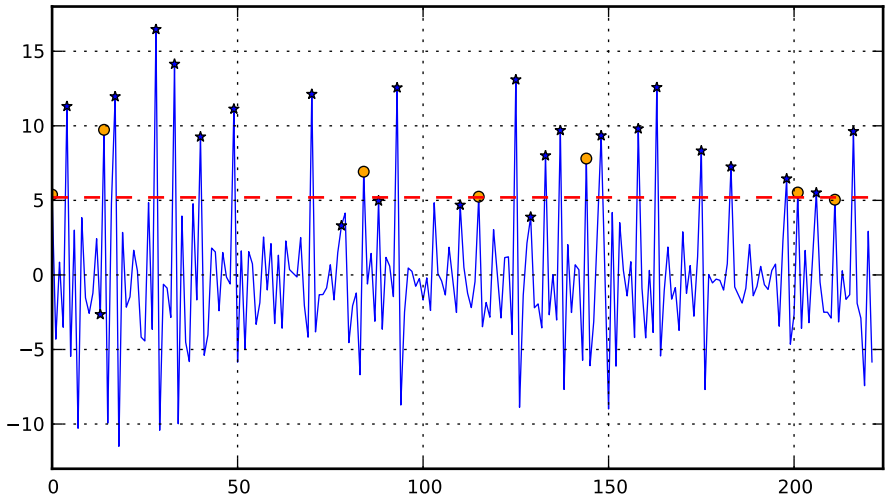


Figure 4.4: Plot of $score'_i$ for a document set from Reuters corpus where y-axis denotes the value of $f'(x)$ and x-axis is the sentence index in the text. Dashed horizontal line drawn from y=5.22 is the cut-off point used for this corpus. Star marked data points denote a true article boundary. Circles mark false positive classifications

This version of DLCA performs a linear running time local analysis on the document, with a low space complexity bounded by the window size Ω. Since its running time and space complexity is linear in terms of the number of sentences, it can be used to decompose a long document. As it will be presented in the results, the top scoring sentences are usually true topic boundaries. These two properties can be exploited in order to create smaller problems by extracting few segments with DLCA, and can be further decomposed using more demanding algorithms performing global analysis.

4.1.2 Number of Topics is Unknown

While the DLCA scores obtained above are good indicators for topic boundaries, a topic segment by itself can be composed of subtopics. Through local analysis it is not possible to identify if a segment is a subtopic or not, only a difference is modelled. Thus, it is not an easy task to identify the number of topics using only the DLCA scores. Simple methods determining a stopping criteria based on the slope of DLCA scores it was not able to infer the number of topics accurately.

Instead of using fixed sliding windows the segments are grown by repetitively merging adjacent clusters that exhibit a lexical cohesion towards either left or right neighbour. This both removes the window size parameter, and also enables to devise a stopping condition as the complete segment can be observed.

Algorithm initializes with m clusters, where each sentence is assumed to be a segment on its own. The whole document is represented as a chain of clusters, where each cluster keeps track of its left and right clusters. If a sentence contains less than 2 content words, it is immediately merged with its left cluster with the assumption that it is following a discussion introduced earlier in the text.

The decision of merging clusters is based on the semantic relatedness scores between the words. A graph is formed for each cluster C_i by the sets C_{i-1}, C_i and C_{i+1}, where the edges are defined between the members of C_i and the other words. Let w_{i1} be the first word of C_i and $w_{(i-1)1}$ be the first word of C_{i-1}. In this setting with a random walk model, the probability of moving from C_i to C_{i-1} is defined as the sum of transition probabilities starting from C_i and ending in C_{i-1} as defined in Equation 4.2. The transition probability $p(w_i \rightsquigarrow w_j)$ is the probability of transition from w_i to w_j, which will be defined with two different random walk models. Similarly the probability to right context can be defined as in Equation 4.3.

$$P(C_i \rightsquigarrow C_{i-1}) = \frac{1}{|C_i|} \sum_{w_i \in C_i \wedge w_j \in C_{i-1}} p(w_i \rightsquigarrow w_j) \tag{4.2}$$

$$P(C_i \rightsquigarrow C_{i+1}) = \frac{1}{|C_i|} \sum_{w_i \in C_i \wedge w_j \in C_{i+1}} p(wi \rightsquigarrow w_j) \tag{4.3}$$

The transition probability $p(wi \rightsquigarrow w_j)$ can be defined in terms of the semantic relatedness values as in Equation 4.4, where w' is a neighbour of w_i in $Adj(w_i)$ where it could be either C_{i-1} or C_{i+1}. This biased probability encourages transitions to semantically related words more than the other unrelated words. An intuitive but misleading algorithm might decide to merge a cluster to its left neighbour if $P(C_i \rightsquigarrow C_{i-1}) > P(C_i \rightsquigarrow C_{i-1})$ or to its right neighbour otherwise. However this strategy is bound to create a single large text segment, as it is more probable to end in a cluster with more words and larger clusters will always win against smaller clusters. Instead of this, tests of randomness gives better results

and is able to leave a cluster as it is, if there is no bias towards either left or right context.

$$p(w_i \rightsquigarrow w_j) = \frac{SR(w_i, w_j)}{\sum_{w' \in Adj(w_i)} SR(w_i, w')} \tag{4.4}$$

In order to test if $P(C_i \rightsquigarrow C_{i-1}) > P(C_i \rightsquigarrow C_{i-1})$ is larger than a completely random walk, an alternate probability model is defined by using $p'(w_i \rightsquigarrow w_j)$ as in Equation 4.5. If $P(C_i \rightsquigarrow C_{i-1}) > P'(C_i \rightsquigarrow C_{i-1})$ and $P(C_i \rightsquigarrow C_{i+1}) < P'(C_i \rightsquigarrow C_{i+1})$, then the cluster is merged with left cluster as there is a bias greater than the random model, there is no bias towards right context. Since the probability of $P(C_i \rightsquigarrow C_i)$ is also defined, a cluster spanning a complete segment will tend to have a higher probability to itself. The algorithm requires no stopping condition as a cluster stops merging if it is self-contained. When no more merges are possible the clusters are reported as the final segments.

$$p'(w_i \rightsquigarrow w_j) = \frac{1}{|C_i|(|C_{i-1}| + |C_{i+1}|)} \tag{4.5}$$

Naturally some clusters will have small divergences from the random model, while others will diverge more. Furthermore a cluster with no bias towards an adjacent cluster can have a bias if its neighbour grows. In fact, this algorithm is expected to start merging from segment boundaries towards the other end of the segment. For a cluster in the middle of a segment the probability bias will become more evident when its neighbour becomes the next segment. In order to exploit this observation, clusters with the largest probability differences will be merged earlier, as they are expected to be on the segment boundary. Using a binary heap keyed by the log likelihood ratios of $P(C_i \rightsquigarrow C_{i+1})$ or $P(C_i \rightsquigarrow C_{i-1})$ to their truly random counterparts, the cluster with the largest probability difference is processed first. When a cluster is merged only three cluster's probabilities are updated; the new merged cluster, the standing neighbour of merged cluster and the new neighbour of merged cluster.

4.2 Dataset and Evaluation

Topic segmentation is a subjective task, and even for a single document it may not be possible to reach a consensus between human judges. For instance, Hearst [54] reports the disagreement on the topic boundaries of an article called Stargazers annotated by seven human subjects. Due to this subjective nature, artificial data sets are built by concatenating news articles to form a single text. This task is called news story segmentation where the goal is to find the concatenation points, neglecting any sub-topic that may reside in one of the articles.

In order to compare the algorithms with the state-of-the-art algorithms in English, they are evaluated on a corpus used in Stokes [10]. The corpus consists of 40 files each containing 25 different news articles gathered from Reuters. For the Turkish language a new corpora is formed by concatenating Turkish news articles gathered from BilCol [115]. 25 different files are formed, each containing 50 different articles.

As a baseline algorithm, a truly random algorithm, which assigns random topic segments in regular intervals is reported. For the English experiment, Choi [56], TextTiling [54] and Select (Stokes et al.) [10] are reported. For the Turkish experiment, Choi's topic segmentation algorithm is modified to use Turkish stemming and stop word removal.

In order to measure the effectiveness of the algorithms, recall and precision is reported. Recall is the proportion of the number of truly identified topic boundaries to the number of true boundaries. Precision is the percentage of true boundaries with respect to the totally assigned boundaries. When the algorithm has access to the number of topics, its precision and recall values are exactly the same. In the literature precision and recall is criticized as it over-penalizes methods that assign topic boundaries close to the true boundaries, which can be considered as near-misses. Instead of recall and precision Word Error Rate (P_k) [67] and WindowDiff [68] evaluation metrics are used in topic segmentation. The P_k value quantifies the inconsistency between the reference segmentation and the system built segmentation in terms of words. WindowDiff on the other hand quantifies the error rate as the number of segments between the reference and system built segmentation.

4.3 Results

The topic segmentation problem is valuable from two aspects. First it sheds a light on the value of semantic relatedness algorithms in a high level task, providing a common test-bed to compare different methods. Second, it is a task that can be used as a pre-processing tool for other high-level tasks such as summarization. In order to investigate these, first a comparison between the semantic spaces and WordNet based measures will be given. Following this, performance of the algorithm is compared to other topic segmentation algorithms previously defined in the literature.

4.3.1 Effect of Semantic Space Parameters

The parameters of the semantic space play an important role in the effectiveness of the algorithm. The vocabulary size (word removal factor), number of SVD dimensions retained, stop word removal, stemming algorithm and co-occurrence window size, all determine the effectiveness of the algorithm.

In Chapter 3 the semantic spaces are evaluated with Word Association task and WordNet synonymy recall values. Although there is a level of consensus between the results of each task, in some parameters there are some deviations. While in WordNet synonymy task, the semantic space achieving the best results seems uses co-occurrence window of 1 and a reduction factor higher than 800, in Word Association better results are achieved with a semantic space with a co-occurrence window size of 3 and 700 SVD dimensions retained. In order to investigate these slightly contradicting results, the extrinsic evaluation method topic segmentation is used.

Figure 4.5 is the plot of word error rate P_k of DLCA algorithm when the number of topics is given as a parameter, where the x-axis is the co-occurrence window size in Turkish language. As can be observed, the error rate declines to 0.1188 when co-occurrence window size is 4 and continues to degrade slowly. In WordNet synonymy recall task, the best recall value is achieved when co-occurrence window is 1, but in Word Association task a peak is observed when using 4 and fluctuates for higher co-occurrence window sizes. It is possible to state that in WordNet synonymy task the best results are obtained with the smallest

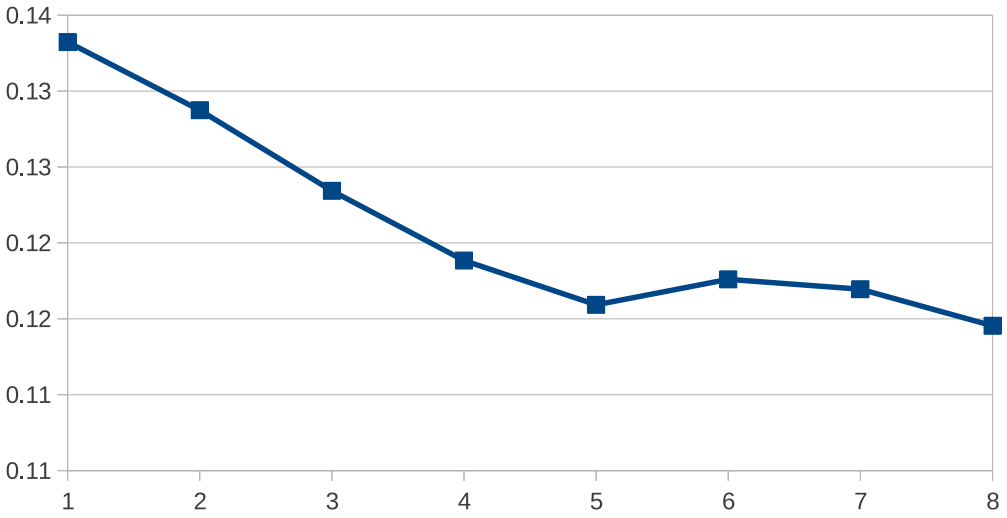


Figure 4.5: Plot of Word Error Rate P_k and the size of the Co-occurrence Window

window size equal to 1, while in other tasks a window size equal to 3-4 achieves the best results. The performance of topic segmentation exhibits a similar pattern to Word Association rather than WordNet synonymy task. Similar results are also observed in the English experiment for a co-occurrence window of size 3, which achieves the best Word Association and good topic segmentation results. In the light of these results, it is possible to conclude that when only synonyms are targeted by the semantic relatedness function a low co-occurrence window size must be preferred. However, for a general semantic relatedness function it should be as high as 3,4 words. Note that in both languages these values are similar, as this parameter does not depend on the corpora size.

The number of SVD dimensions retained is another parameter, where a difference between WordNet synonymy task and topic segmentation is observed. Figure 4.6 shows the plot of P_k values against the retained SVD dimensions. In WordNet synonymy task higher number of dimensions (i.e. 800) achieve the best results, but in Word Association there is a peak in 400 dimensions, which is in consensus with the low error rate in topic segmentation. In the English language experiments using 700 SVD dimensions achieve the best results in Word Association, while for WordNet synonymy task a value as high as 3200 achieves a good

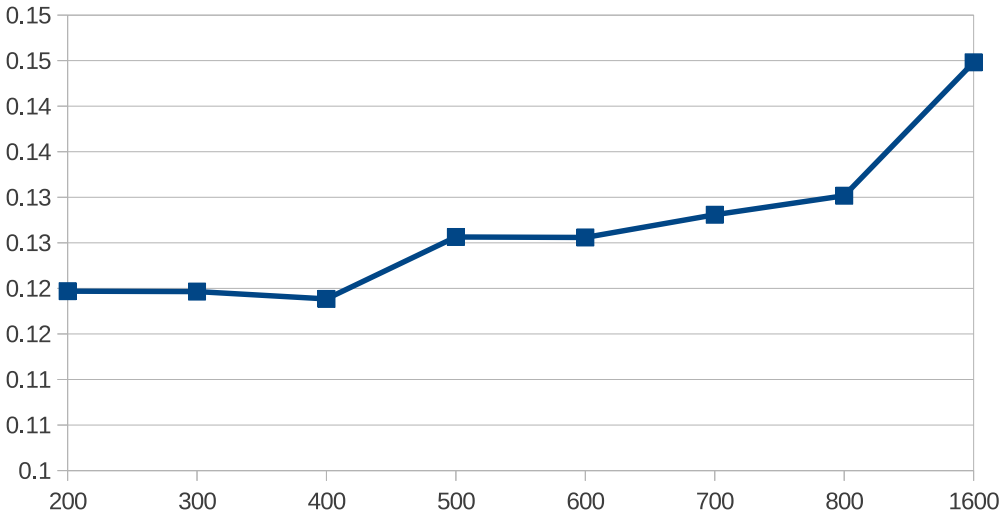


Figure 4.6: Plot of Word Error Rate P_k and the number of SVD dimensions

recall value. While Turkish achieves the best results by using a dimension reduction 400, English achieves its minimum with 700. This difference between the languages can be explained by the difference in corpora size. As English corpora is larger, the signal-to-noise ratio of the co-occurrence matrix is greater, and more dimensions can be beneficial to represent the semantic space.

The importance of stemming methods is more evident in Topic Segmentation. The word error rate increases upto 0.1616 from 0.1262 when stemming using Porter Stemmer is disabled. When only the stop word removal is disabled the error rate increases to 0.1472. Pruning the co-occurrence matrix by removing infrequent words does not have a notable effect in topic segmentation. Using PMI instead of Entropy does not change the effectiveness of the system significantly. While the word error rate of semantic space using PMI as weighting scheme is 0.0677, when Entropy weighting is used the same configuration achieves a slightly worse word error rate of 0.06944. While word error rate slightly differs, the recall is exactly the same that is 0.865 for the two weighting schemes.

Effectiveness changes significantly depending on the parameters of the semantic space. The difference between the worst and best performing parameter sets is more than 6 percent in both English and Turkish experiments. The values of

the parameters that will yield the best results depend on the characteristics of the background corpora, which the semantic space is built from. However experiments indicate that a semantic space built using 3-5 co-occurrence window, a dimension reduction of 500-700, stop-word removal, pruning the least frequent words to prune the vocabulary to 50,000 words and using a stemming algorithm achieve the best results.

The full-dimensional semantic space of $|V|$ dimensions (more than 60K for both languages) achieves a word error rate of 0.1772 and a recall of 0.693 improves significantly when only its dimensions are reduced to 400 using SVD truncation in Turkish language. The reduced semantic space is able to find 0.865 of the correct topics and has a word error rate of 0.0677. Not only the effectiveness of SVD truncated is superior to full-dimensional semantic space but also the running time is at least few magnitudes lower. Since in topic segmentation thousands or even millions of semantic relatedness score calculations are performed, the efficiency of the semantic relatedness function directly determines the running time of the whole system. The cost of cosine similarity in SVD reduced spaces is constant and usually a low value between 400-800. However, in the full-dimensional space it depends on the number of non-zero dimensions of the compared words and can be equal to $|V|$ in the worst case.

4.3.2 Comparison with WordNet based Semantic Relatedness Functions

WordNet based topic segmentation and summarization algorithms exists in the literature [10, 9, 4]. Nevertheless none of them use the semantic relatedness functions. The experiments in Chapter 3 have contradicting results for WordNet based methods. While in Word Association task the WordNet based semantic relatedness methods achieve the best results, in TOEFL synonymy tasks they achieve very poor results close to random selection.

Table 4.1 shows the word error rate P_k and precision/recall values for the DLCA algorithm when the number of topics are known. Note that the precision and recall values are the same as the algorithms have access to the number of topics. Although WordNet based semantic relatedness methods are effective in Word Association task, they are not effective in topic segmentation task. Among the

Semantic Relatedness Function	**Recall&Precision**	P_k
SVD Reduced Semantic Space	0.871	0.064
Full Dimensional Semantic Space	0.693	0.177
Wu&Palmer	0.232	0.612
Resnik Semantic Distance	0.673	0.197
Leacock&Chodorow	0.490	0.306
Random	0.093	0.490

Table 4.1: Comparison of different Semantic Relatedness Functions used in DLCA for Reuters news articles

three WordNet semantic relatedness functions, Resnik semantic distance [100], which uses occurrence counts from a corpora, is the most effective one. DLCA using WordNet based semantic relatedness functions achieves better than Random selection of topic segments. The SVD truncated semantic space built from Wikipedia is able to find 87% of the topic boundaries correctly.

When considering the performance of WordNet based semantic relatedness functions, it should be considered that most of the relationships in it are between nouns. The number of relationships between other classes are low. For example in WordNet it is not possible to define the relationships between the noun *murder* and the verb *kill*. This shortcoming plays an instrumental role in the effectiveness of WordNet based semantic relatedness functions when they are used in a high level natural language task. It should also be noted that Resnik method which outperforms the other WordNet SR functions uses occurrence statistics along with WordNet classical relationships. The performance gain can be attributed to these co-occurrence statistics.

While WordNet based methods perform better in Word Association task, both full-dimensional and reduced semantic spaces achieve better results in topic segmentation. Reduced semantic spaces have an additional advantage over both full-dimensional semantic spaces and WordNet based methods as its running time is notably lower. Performing Breadth First Search or Dijkstra algorithms on WordNet graph which contains more than 150 thousand words takes more time when compared to cosine similarity calculation in reduced semantic spaces. In terms of running time, the full-dimensional semantic space is the worst and few orders of magnitude lower than reduced semantic space.

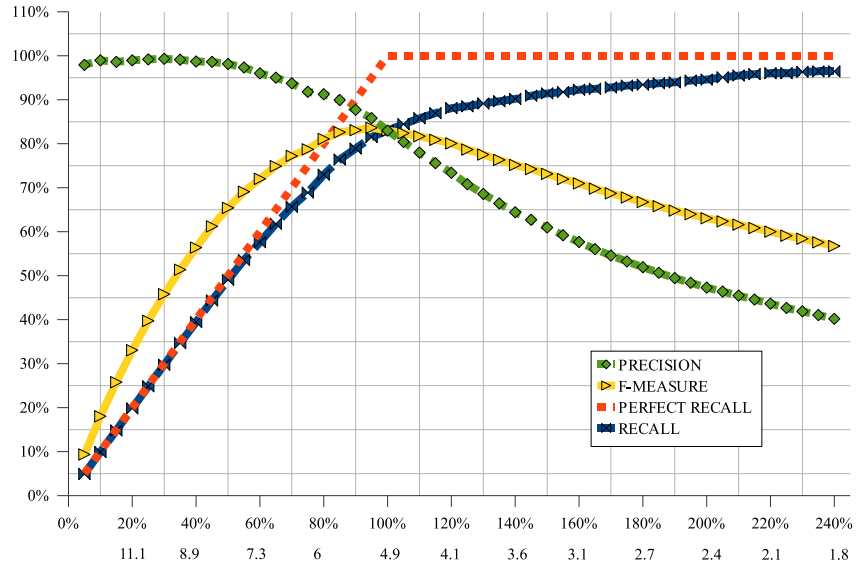


Figure 4.7: Recall, precision and f-measure of DLCA, along with the recall of a hypothetical perfect segmenter are shown in y-axis. The x-axis shows the number of sentences selected as a portion of total topic boundaries, and the corresponding DLCA scores are given below.

4.3.3 Effectiveness of Semantic Relatedness based Topic Segmentation Algorithms

Figure 4.7 demonstrates the relationship between the number of topic boundaries and the DLCA score, in which the x-axis denotes the proportion of the selected sentences to the number of actual boundaries. The value x=60% represents the sentences with top DLCA scores of at least 7.3. Selecting these sentences as topics yields a recall of approximately 55%. Perfect recall represents the scores of a hypothetical perfect segmentation system that is able to find all topics. This ideal system will have a linear increase until x=100% and will stay constant after that point. It is important to note that DLCA achieves a similar curve up to 70% with a precision higher than 95%.

Table 4.2 presents the recall, precision, and Word Error Rate P_k scores of five different systems tested in the English Reuters News articles dataset. In the table DLCA shows the method with a knowledge of number of topic boundaries and DLCA_b is the version of DLCA which determines the number of topics. These two versions of the algorithm are compared with TextTiling [54], C99 [56], SeLeCT

System	Recall	Precision	P_k
DLCA	0.871	0.871	0.064
DLCA$_b$	0.796	0.761	0.110
C99$_b$	0.749	0.724	0.128
SeLeCT$_b$	0.606	0.791	0.191
TextTiling$_b$	0.321	0.410	0.221
BayesSeg	0.886	0.886	0.043
UI	0.884	0.884	0.045
BayesSeg$_b$	0.862	0.870	0.059
UI$_b$	0.659	0.912	0.133
MinCut	0.358	0.358	0.236

Table 4.2: Topic segmentation scores for Reuters news articles

[10], BayesSeg [61], MinCut [59] and UI [58] algorithms. The parameters of the algorithms for BayesSeg, MinCut and UI are set to the same configurations as in Eisenstein and Barzilay [61]. The results of the experiment in the same corpus in Stokes [10] are reproduced.

DLCA performs a local analysis of the text by processing it in a linear fashion. These two attributes are common both for TextTiling and SeLeCT. DLCA performs its analysis on the text blocks while TextTiling uses gaps, i.e. the similarity of two adjacent text blocks. We believe that using gaps instead of text blocks is limited, since gaps do not fully observe the lexical cohesion change in both directions. Another advantage is the integration of a semantic relatedness function instead of relying solely on word re-iteration. These two main differences significantly increase the accuracy of topic segmentation. The SeLeCT algorithm uses lexical chains and WordNet classical relationships. In the SeLeCT algorithm, a concentration of lexical chain initiations points to a topic shift. We believe that this model is too strict and prone to errors in word sense disambiguation and shortcomings of WordNet. One such error may result in a discontinuity in the lexical chain, and thus in the modeled topic. The DLCA model is more flexible and does not try to model the topics, but rather it concentrates on finding the topic shifts.

When compared to more recent algorithms BayesSeg and UI that perform a

global analysis the performance of DLCA is competitive but below these algorithms. It should be noted that while the recall and precision values are close, the word error rate of DLCA is 2% lower. Since DLCA performs a local analysis and concentrates on topic shifts a false boundary is usually further away from the true boundary. In MinCut, BayesSeg and UI dynamic programming is used to model the segments. This forces the false boundary assignments to be near the true boundaries. Thus, the P_k values of DLCA are lower relative to its recall values. When the number of boundaries are not given as a parameter, DLCA outperforms UI. While using dynamic programming with semantic relatedness promises to achieve better results, it is not an easy task. The main challenge for building a dynamic programming method with semantic relatedness scores is the running time of the algorithm. Since in a dynamic programming method semantic relatedness for all pairs should be calculated, the running time increases significantly. Devising a dynamic programming based method able to consider the inter and intra segment similarities is left as an important future work.

Table 4.3 shows the results of Turkish experiment, for three different algorithms DLCA, DLCA inferring topic boundaries (DLCA_b), Choi's C99 [56] and UI [58]. DLCA achieves better results when compared to the other algorithms. When the number of topic boundaries are known it is able to find 82% of the true topic boundaries. Surprisingly even though UI is effective in English language, its performance diminishes in Turkish. Although its language dependent components, stop words and stemming methods, are modified for Turkish, its performance is relatively low.

System	**Recall**	**Precision**	P_k
DLCA	0.821	0.821	0.116
DLCA$_b$	0.681	0.745	0.163
C99$_b$	0.524	0.553	0.264
UI$_b$	0.395	0.887	0.264
UI	0.631	0.631	0.167

Table 4.3: Topic segmentation scores for Turkish news articles

The results of both experiments show that when compared to other segmentation algorithms that use only word repetition or WordNet classical relationships,

semantic relatedness functions are able to improve the topic segmentation performance. However the results of dynamic programming based algorithms indicate that there is still room for improvement. It should be noted that the performances of DLCA in both Turkish and English are close and is able to find more than 80% of the true topic boundaries. Furthermore the correct boundaries found usually have the highest DLCA scores, in cases where only the most significant (ones where the topic of the text changes sharply) topic boundaries are required, stopping the algorithm early can be useful for yielding higher precision.

Chapter 5

Automated Text Summarization

Automated text summarization aims to create concise representative texts shorter than their original documents. One of the approaches in summarization is to select important sentences from the original document to form an extract. However how to define what is important is rather vague and difficult. Topical structure of the text can be used as an important clue. When a topic is introduced, usually its first few sentences describe the new idea more generally before describing its details. Based on this observation, a summarization algorithm can select the first sentences of the most salient topic segments.

This same observation is also exploited in our previous work, which uses lexical chains [3, 4] built from WordNet classical relationships to determine the sentences where a new idea is presented. It is assumed that start and end positions of lexical chains correspond to the topic segments. Building on the same idea, instead of using only WordNet classical relationships, topic segment methods described and evaluated in Chapter 4 are deployed for summarization.

This chapter first describes the developed topic segments aware summarization algorithm, which orders topic segments with respect to their importance. Following this, Turkish and English language corpora and the experiment settings are presented. This chapter concludes with a presentation of the results and a discussion based on these results.

5.1 Segment Salience and Sentence Extraction

In summarization a document formed of sentences denoted by $S = \{s_0, s_1, ...s_n\}$, is processed and a subset of S is returned as an extract. Through topic segmentation algorithms it is possible to further decompose the document into topic regions. A topic segment T_i is a linear sequence of sentences, where b_i denotes the starting sentence's index of T_i and e_i denotes the ending sentence's index of T_i. All the sentences between b_i and e_i are in this topic segment, inclusively. Let m be the number of segments found for the document, then it follows that $b_0 = 0$ and $e_m = n$.

In order to select sentences from S, importance of the segments are determined. In a similar fashion to Radev et al. [116], saliency of a segment is formulated as its centrality. Centrality measures the similarity of a text block's vector to the centroid of the whole document. A text block with a high similarity to the centroid of the document is considered as important, as it contains the most dominant information entailed in the document.

Let $C(k, l)$ be a function which forms a vector, where the words in the document are the dimensions, and the weight of the j^{th} dimension $C(k, l)_j$ is the average tf*idf value of the j^{th} word. The tf*idf weights are composed by two components, where term frequency (tf) is the number of times the word occurs in sentences with an index between k and l. The inverse document frequency is calculated using the frequencies in a background corpora, which is Wikipedia in our case. A topic segment T_i is transformed to a vector using the function $C(b_i, e_i)$. Similarly the centroid of the document is formulated as $C(0, n)$. The centrality of T_i is the cosine similarity of $C(0, n)$ and $C(b_i, e_i)$, as in Equation 5.1.

$$Centrality(T_i) = \frac{\sum_{j=0}^{V} C(b_i, e_i)_j * C(0, n)}{\sqrt{\sum_{j=0}^{V} C(b_i, e_i)_j}\sqrt{\sum_{j=0}^{V} C(0, n)_j}} \tag{5.1}$$

In certain genre such as news articles and journal articles, sentences in the first portions of the document are usually indicative and important. In order to incorporate this clue into topic saliency, a second component representing the position of the segment is calculated. Equation 5.2 is the position component for measuring the segment importance. The position score of T_i is 1 when $i = 0$ and

gets lower scores for segments closer to the end of the document.

$$Position(T_i) = \frac{n - b_i}{n} \tag{5.2}$$

The final segment score is the linear combination of position and centrality scores. The first sentence of the top scoring segments are included in the summary, until the target summary size is reached. Since the saliency score represents the importance of the segment, the most important topics in the document are expected to be covered by following this sentence selection procedure.

5.2 Corpus and Evaluation

Evaluating summarization algorithms is a difficult task and is an active research area. A summary's quality can be considered from different aspects, such as selected contents' importance and presentation quality. Presentation quality itself is composed of two aspects: grammatical correctness and coherence. Since the algorithm used builds extracts and does not involve any natural language generation, it should be grammatically correct given the sentences selected from the original documents are. On the other hand, there can be coherency issues in the formed extracts as the coherence links tying the selected sentences may be excluded. However, since the first sentences of topic segments are extracted, defects are less likely.

Since deciding what is more important in a document is a subjective task, judgements of multiple humans is desirable. Ideally multiple human judges can evaluate and assign scores for each system generated summary. However this requires substantial manual work, which cannot be automated. Instead the human judges are asked to write abstracts for each document in the corpora, and the system generated summaries are compared with these. Although less accurate and superficial, this technique can be repeated in order to measure the improvements in summarization techniques.

One of the most widely used measures is Recall Oriented Understudy for Gisting Evaluation (ROUGE) [88]. ROUGE calculates the recall of text units by N-Grams, Longest Common Subsequence (LCS) and Weighted version of LCS

(WLCS). The N-Grams depending on the N value, counts the number of word matches. LCS finds the longest common sequences, which may be interrupted by other words, but should be in the same order. The weighted version of LCS awards common sequences of uninterrupted text units. A large overlap with the model summary is an evidence of including a content thought to be important by at least one person. However, a single model may not cover all possibly important content deserving to be included in a summary. For this reason, instead of precision, recall values are usually reported for ROUGE scores as having a sentence in the generated summary which is not included in the model summary does not necessarily mean that it is unimportant.

The summarization algorithm, is evaluated in both Turkish and English languages. For the Turkish language experiments the scientific articles corpora built in Ozsoy et al. [117] is used. The corpora consists of 2 datasets each containing 50 articles chosen from medicine, sociology and psychology journals. Each article contains an accompanying abstract written by the author of the article. These manually built ground truth abstracts are compared to the system built extracts using ROUGE scores.

The English language experiments are based on the DUC 2002 corpora of news articles. For news articles selected from Reuters, LA Times and Associated Press multiple human judges are asked to create abstracts. For evaluation these manually built abstracts are compared with the automatically built summaries. In the corpus there are 500 summaries each containing 2 manually built abstracts.

The summarization algorithm which uses the DLCA topic segmentation method is denoted by DLCA Summarizer. For comparison the results of Lexical chaining based summarization algorithms of Ercan and Cicekli [4] and the results of 13 participants of DUC 2002 single document summarization competition are reproduced for the English experiments. In order to avoid a clutter, only the best, average and worst scores of DUC participants are reported.

In the Turkish language experiments the results of DLCA summarizer that uses Turkish semantic space are reported. Ozsoy et al. [117] introduce two novel algorithms and compares different summarization techniques that use SVD. The re-implemented SVD summarization algorithms are; Gong and Liu [118], Steinberger and Jezek [119] and Murray et al. [120]. The novel methods introduced in their research are Cross and Topic methods. For comparison purposes the best

System	DS1 ROUGE-L	DS2 ROUGE-L
Best Ozsoy et al. [117]	0,320	0,274
Worst Ozsoy et al. [117]	0,195	0,189
DLCA Summarizer	0.244	0.219

Table 5.1: ROUGE scores of the summarization experiments for Turkish language. Results on two different datasets are presented

Algorithms	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**ROUGE-W**
DLCA Summarizer	0.43	0.184	0.348	0.127
Ercan&Cicekli	0.394	0.16	0.322	0.117
Average	0.401	0.179	0.329	0.12
Best System	0.485	0.231	0.4	0.147
Worst System	0.065	0.033	0.061	0.028

Table 5.2: DUC 2002 English summarization results

and worst among the results of these algorithms are reported.

5.3 Results

The ROUGE-L f-measure scores achieved in the two Turkish datasets are shown in Table 5.1. The Cross method first introduced by Ozsoy et al. [117] achieves the best results. While the results of DLCA Summarizer are below the LSA based methods in general, it is competitive with Steinberger and Jezek [119] and Gong and Liu [118].

All ROUGE recall values are reported for English in Table 5.2. DLCA Summarizer is below 8 other systems which participated in DUC 2002, however it is above the average in all ROUGE measures. When compared to the summarization algorithm which uses lexical chains proposed in our earlier work [4], there is an improvement in the results.

In both languages, DLCA summarizer achieves above the average scores. To be fair, summarization by topic segmentation alone does not seem to be an adequate method. When comparing the results of DUC 2002, it should be noted

that some of the other algorithms are performing sentence reduction and anaphora resolution. In other competing algorithms, there are some systems that focus on only news article domain, tracking events. Reduction of sentences could improve ROUGE scores as summaries extracted are limited in size, and only a portion of selected sentences may be important. Resolving anaphora, improves the performance as they are not usually used in the model summaries.

The experiments in Chapter 4 shows that the performance of DLCA in topic segmentation is at promising levels. When compared to our previous work [4] and Barzilay's algorithms [9] that use lexical chains and WordNet classical relations DLCA summarizer achieves better results. This shows that there is an improvement in topic segmentation over lexical chains based methods, when semantic relatedness functions based on semantic spaces are used.

When compared to LSA based summarization algorithms evaluated in Ozsoy et al. [117], it is possible to argue that DLCA summarizer while performs better than some of the methods, it is below the performance of their best method. While both of the methods are using Latent Semantic Analysis, they are used for completely different purposes. In Ozsoy et al. [117] the SVD is applied to the sentence term matrix of the analysed document. However in DLCA summarizer, it is applied to the term-by-term matrix built from Wikipedia Turkish articles, in order to train a semantic relatedness function. The performance gain of SVD from local information resident in the analysed document is apparently more useful, when compared to DLCA model which tries to model the relationships between words.

Chapter 6

Keyphrase Extraction

Keyphrases are short descriptive phrases defining the underlying document. Automatically assigning keyphrases is a difficult task as they should reflect the meaning conveyed in the document in a concise manner. A keyphrase system able to produce a list formed of phrases appearing in the document is called a keyphrase extraction system. On the other hand, if a system is able to generate keyphrases that do not appear in the document it is called as a keyphrase generation system.

A document is formed of many words and phrases, determining which of these are salient and able to distinguish the contents of the document is a task requiring semantic analysis. Thus, this task is highly related to lexical semantics and lexical cohesion. The methods described in Chapter 3 are primarily focused on modelling the semantic relationships between words. However when individual words are combined to build a phrase, the meaning of the phrase drifts away from the contributing words. Natural language is productive in terms of building phrases, phrases can be built by combining different open class words.

The problem of determining the semantics of a phrase by considering the semantics of the words is a difficult and a new topic attracting interest from the research community in recent years [121, 122]. One approach for determining the semantics of phrases is by including them in the semantic spaces built [123, 124]. However, these models are not suitable for practical applications as semantic spaces grow exponentially. Another approach is by defining operations on the

semantic vector representations of individual words to form the semantic representation of phrases [121, 122]. These methods are still not in a level applicable in practical problems. The most significant problem appears to be the amount of corpora required to model virtually infinite number of phrases that can be built.

Fortunately keyphrases are usually phrases commonly used in a specific domain and it is possible to gather adequate statistics to model their meanings. Instead of using the semantic relatedness methods defined in Chapter 3 a simpler methodology is implemented specifically for keyphrases. Different contexts which a keyphrase appears in is retrieved from a large background corpora. The similarities between these contexts are used to determine its saliency and relevancy to the analysed document.

The saliency of a keyphrase is formulated as a function of its ambiguity. I assume that a phrase used in many different domains to convey different meanings is not suitable as a keyphrase. A keyphrase should be commonly used to represent a single meaning. Thus, different contexts in which the keyphrase appears in should be related to each other. This same idea is also true for predicting the quality of queries in search engines. Query performance prediction (QPP) methods try to formulate how a query will perform and how ambiguous it is. Even if a phrase is unambiguous and common it may not be appropriate for the document analysed. In order to ensure the relevancy of the keyphrase to the document, its similarity to the retrieved documents are also compared.

This chapter first describes the algorithm developed for keyphrase extraction. Dataset and background corpora used in the experiments are introduced. Finally the results are presented with a discussion of the contribution of the introduced features.

6.1 Keyphrase Extraction using QPP Features

The general components of the keyphrase extraction system are depicted in Figure 6.1. The gist of the system is common to some previous works [125, 94, 91, 93] as all of them perform feature extraction and supervised classification. First step in keyphrase extraction is the tokenization of the text into words and punctuations. Using the token stream, a candidate keyphrase list is formed from the phrases

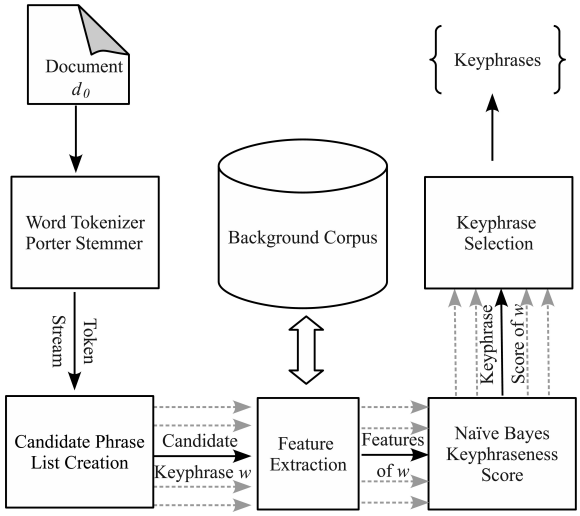


Figure 6.1: Components of the keyphrase extraction system.

appearing in the text of the original document. For each candidate phrase w the feature extraction component extracts a feature set using the background corpus and the original document d_0. The background corpus is a sufficiently large document collection, excluding the original documents which the keyphrases are extracted. A Naive Bayes classifier trained with documents of the same genre and their associated keyphrase lists calculates the probability of keyphraseness for each candidate phrase w. Keyphrases are selected using these scores, and the output of the system is a set of keyphrases. The keyphrase selection component simply sorts the phrases according to the keyphraseness score and returns the top keyphrases.

The feature extraction component is what distinguishes our keyphrase extraction system from the others. The feature extractor uses a background corpus to determine some intrinsic properties of each phrase w, as depicted in the flowchart in Figure 6.1. The feature extractor operates on a phrase w, finding the document set D' formed of all the documents that w appears in. Both for efficiency and for effectiveness, a subset of D' is selected by a sampling procedure. Features are extracted from this subset D and the original document d_0.

6.1.1 Candidate Phrase List Creation

Keyphrases usually appear as noun phrases in documents. Hulth [91] reports that nouns preceded by nouns or adjectives are the most common part of speech (PoS) tag patterns observed for keyphrases. In fact, the majority of the keyphrases in the corpora used in the experiments can be extracted with a simple grammar rule for finding noun phrases. In our system, in order to create a candidate keyphrase list, the text of the given document d_0 is tokenized, and the PoS tags are assigned using the Stanford PoS tagger [126]. Each sequence of PoS tags satisfying the regular expression $(JJ|NN)*NN$ is included in the candidate phrase list, where NN represents nouns and JJ represents adjectives. For example, in the sentence "*This is a good/JJ machine/NN learning/NN algorithm/NN*" "*good machine learning algorithm*", "*good machine learning*", "*machine learning algorithm*", "*machine learning*", "*learning algorithm*", "*machine*", "*learning*", "*algorithm*" match the regular expression and are extracted as candidate phrases. Only the candidate keyphrases matching the regular expression are retained and evaluated by the classification algorithm.

The PoS pattern method can detect more than 80% of all keyphrases in the research article corpus used in our experiments as candidate keyphrases. We compared this method with an exhaustive candidate keyphrase extraction method, which returns all consecutive terms that do not contain punctuations as candidate keyphrases. The exhaustive method extracts all the keyphrases appearing in the text as candidate keyphrases. It detects 82% of the keyphrases in the research article corpus, which is only 2% more than from the PoS pattern method. However, this method produces more candidates than the PoS pattern method where most candidates are unlikely to be keyphrases or even phrases. For example, in the corpus the exhaustive method produces approximately 35,000 candidate phrases whereas the PoS method produces approximately 2000 candidates per document. When the former method is used, the system must process a larger number of candidate keyphrases, which degrades the efficiency of the system. In addition, the effectiveness of the system is not improved, as having more candidate keyphrases creates noise for the classification algorithm. Since the PoS pattern method is as effective as the exhaustive method, it is preferred instead of the exhaustive method.

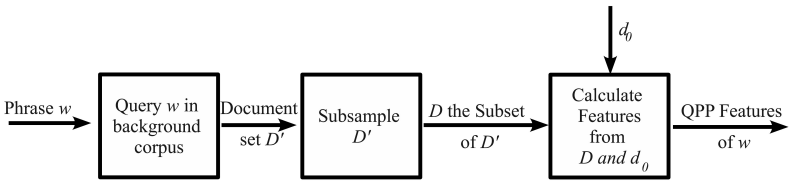


Figure 6.2: Flowchart of feature extractor.

6.1.2 Information Retrieval from Wikipedia

The information retrieval component (IRC) performs full-text search in the background corpus, which is composed of Wikipedia articles. Since the keyphrase extraction algorithm processes the document vectors and language models of retrieved documents by accessing their full texts, an offline indexing system is preferred. Document retrieval is an important factor for both the efficiency and the effectiveness of the keyphrase extraction system. This section describes how full-text search is performed from the background corpus for a candidate keyphrase given as a query, and how the returned documents are subsampled.

Given a term as a query, the IRC returns the set of articles that contain the searched term. Given a phrase (i.e. multiple terms) as a query, the IRC returns the set of articles in which the terms of the phrase appear in exactly the same configuration. In order for an IRC to support such phrase queries, either a phrase or a positional index must be maintained [127]. For example, for the phrase machine learning, documents containing machine learning are returned, whereas those containing learning machine are not. Implementation of the IRC uses the indexing mechanisms of the Lucene search engine.[1] A Lucene inverted index supports positional indices. For each term, a sorted list of articles containing the term and its position within the text are stored.

Given a phrase w formed of one or more terms, let $D' = \{d_1, d_2, .., d_m\}$ be the set of documents that contain w. If w occurs at least once in all indexed articles of the background corpus, the size of D' may be as large as the length of the collection queried. Since in-depth analysis of the m returned documents is not efficient, and may result in noise due to variations in $|D'|$ for different phrases, a subset D is sampled from D'. The sample size μ is a parameter of the system, and the size of the sample set denoted as $|D|$ is $min(|D'|, \mu)$. Higher

[1] Available at http://lucene.apache.org

values of μ increase the variance of $|D|$ for different phrases, and thus create a bias towards phrases that appear in a smaller number of documents in the background corpus. A lower value of μ is not able to represent the set $|D'|$. Our empirical evaluations show that using a sample size $\mu = 20$ avoids problems caused both by the variations in $|D|$ and by the under-representation of the set D'.

Two sampling strategies were evaluated: random and rank-based. Random sampling simply selects μ documents from D' randomly. Rank-based sampling reduces the noise caused by documents with low frequencies of w. Accordingly, documents are ranked using a function of the frequency of phrase w, and only the μ top-scoring documents are retained. Since the ranked sampling strategy achieves better results than the uniform sampling strategy, only the result of the ranked sampling strategy is reported in this article. In our experiments, the Okapi BM25 ranking function is used:

$$BM25(w, d_i) = IDF(w).\frac{c(w, d_i).(k_1 + 1)}{c(w, d_i) + k_1.(1 - b + b.\frac{|d_i|}{avdl})} \tag{6.1}$$

$$IDF(w) = log(\frac{N - |D'| + 0.5}{|D'| + 0.5}) \tag{6.2}$$

where $c(w, d_i)$ is the frequency of the phrase w in the document d_i, and the parameters b and k_1 control the scoring functions behaviour: b controls the weight of the document length, and k_1 controls the weight of term frequency in the ranking. The values $k_1 = 2$ and $b = 0.75$ are used in our experiments, as Jones [13] reports that these values correlate with human relevance judgements. The average document length in the background corpus is denoted by $avgDocLen$, and $|d_i|$ denotes the length of the document d_i.[2]

The background corpus used in our experiments is composed of English Wikipedia[3] articles that are longer than 200 characters. The number of Wikipedia articles indexed is 3,326,028, containing a total of approximately 1 billion terms. The average document length of the corpus is 237.89 non-stop words.[4] Wikipedia is a generic background corpus, since it is a comprehensive encyclopaedia relating

[2]The IDF component is used to weigh the effect of the terms, when the query is formed of multiple terms. Equation 6.1 differs from Jones [13], as it does not use the IDF component.

[3]The dump file of 30 July 2010 retrieved from http://download.wikimedia.org is used.

[4]Common English propositions and articles are excluded, and a stopword list of 452 words is employed.

to different topics. In practice, the background corpus can be domain specific. For instance, in a digital library, an index of all the articles stored can be preferred instead of the generic Wikipedia articles corpus. We chose to use Wikipedia to have a domain-independent system.

6.1.3 QPP Measures

This section defines the QPP measures utilized in this article on the basis of the assumption that keyphrases are unambiguous query phrases that retrieve documents tied to each other by a specific domain or topic. Note that this assumption is also important for a retrieval system. For example, for the query "learning" a diverse set of documents is returned, whereas for the query "machine learning" a more refined set of documents is returned. The QPP problem [16] tries to predict the effectiveness of a query, and should encourage the query "machine learning" over "learning". The experiments discussed in this paper evaluate the effects of different QPP techniques [128, 16, 17, 129, 130] in keyphrase extraction.

Table 6.1 lists the features used in keyphrase extraction. The first two of these features, $firstPosition$ and $tf * idf$, are in-document features, and the last seven are QPP features, which can be grouped in two categories depending on the methods used in their extraction. The first category uses the geometrical properties of the retrieved documents when represented by vector space models (VSM). The second class of methods uses ideas from language models (LM) and information theory. The extraction of each feature is explained for a single phrase w, using two inputs: the document set D and the original document d_0. Features from 3 to 7 are calculated using vector space models, and the last two are based on language models.

Both VSM-based and LM based methods use a bag-of-words assumption to simplify the analysis. In order to create a bag or set of words, documents are tokenized to words (terms). Words are processed with a Porter stemmer [34] to conflate inflected forms of words with their base forms. In our presentations, the set V denotes the vocabulary formed of conflated words occurring in the background corpus and d_0, the function $c(t_k, d_i)$ returns the number of occurrences of the k^{th} word of V in the i^{th} document, and f_k is the number of documents containing the word t_k. The retrieved documents inthe set D and the original

Id	Feature Name	Value-Range	Description
1	$firstPosition$	$[0, 1]$	Distance of first occurrence of phrase from the top of d_0
2	$tf * idf$	$[0, \infty]$	Term Frequency times inverse document frequency
3	$CosCentrTod_0$	$[0, 1]$	Cosine similarity of the centroid of D to d_0
4	$avgCosTod_0$	$[0, 1]$	Average cosine similarity of document in D to d_0
5	$iterSim$	$[0, 1]$	Mean of pairwise cosine similarities of documents in D
6	$CoxLewisTest$	$[0, 1]$	Cox-Lewis Clustering Tendency Test
7	$DocPertub$	$[-\|D\|, \|D\|]$	Average rank change in D under document perturbation
8	$Clarity$	$[0, \infty]$	KL-Divergence of D language model from background corpora language model
9	$KLDocsfromd_0$	$[0, \infty]$	KL-Divergence of D language model from d_0 language model

Table 6.1: Features used in Keyphrase Extraction.

document d_0 are tokenized using this method.

6.1.3.1 Vector Space Model Based Features

A vector space model defines each document as a vector in a $|V|$-dimensional space, where each dimension corresponds to a word in the vocabulary V. Let d_{ik} represent the k^{th} terms weight in the i^{th} document, where the original document is indexed by 0 and the documents in the set D are indexed from 1 to $|D|$. The weight d_{ik} is calculated as shown in Equation 6.3, where N is the number of documents in the background corpora, and f_k is the number of documents in which the k^{th} term occurs. The weighting function d_{ik} is the term frequency ($tf = c(t_k, d_i)$) times inverse document frequency ($idf = log(N/f_k)$), and is termed the $tf * idf$ weighting.

$$d_{ik} = c(t_k, d_i) log(\frac{N}{f_k}) \tag{6.3}$$

A document vector d_i consists of the weights of all terms of the vocabulary V in the i^{th} document. Two document vectors can be compared with each other through different similarity metrics. The cosine of the angle between two vectors is one such similarity function, called the cosine similarity. The cosine similarity between documents d_i and d_j is calculated using the equation

$$cosSim(d_i, d_j) = \frac{\sum_{k=0}^{|V|} d_{ik} djk}{\sqrt{\sum_{k=0}^{|V|} d_{ik}^2} \sqrt{\sum_{k=0}^{|V|} d_{jk}^2}} \tag{6.4}$$

In this model, our assumption is that a keyphrase w has to retrieve a document set that is geometrically closer to d_0, cohesive, less scattered and more concentrated. When defined in terms of similarity, all documents in D and the original document d_0 should be similar to each other. The average similarity of the retrieved documents to d_0 ($avgCosTod_0$) is calculated using

$$avgCosTod_0 = \frac{1}{|D|} \sum_{i=1}^{|D|} cosSim(d_i, d_0) \tag{6.5}$$

The inter-similarity feature (interSim) calculates the average pairwise similarity of the retrieved documents using

$$interSim = \frac{2}{|D|(|D|-1)} \sum_{i}^{|D|} \sum_{j=i+1}^{|D|} cosSim(d_i, d_j) \tag{6.6}$$

Kwok et al. [131] use a similar metric to predict the performance of queries. Since cosine similarity function is symmetric, the average can be calculated using only $|D|(|D|-1)/2$ similarities.

Calculation of $avgCosTod_0$ requires pairwise similarity calculations of each document with document d_0. Another method used in text categorization and summarization [116, 132] calculates the centroid of documents, and uses the similarity to the centroid instead. The centroid of D, denoted by $\bar{D}$, is the arithmetic mean of the document vectors, and is calculated using

$$\bar{D} = \frac{1}{|D|} \sum_{d \in D} d \tag{6.7}$$

The $CosCentrTod_0$ measure is just the cosine similarity of document d_0 and $\bar{D}$. $CosCentrTod_0$ and $avgCosTod_0$ are two similar measures. They are equal if all the input document vectors are unit vectors, that is, if the norms of vectors are 1. In our case they are not unit vectors, and thus these two values are not

equal but only similar. The $CosCentrTod_0$ feature can be interpreted as a comparison of d_0 with a virtual document formed by concatenating all the retrieved documents in D. The $avgCosTod_0$ feature takes into account the local relationships between each retrieved document and d_0, whereas $CosCentrTod_0$ is based on a more global view of the term usage in D. Furthermore, our experiments have shown that using both $CosCentrTod_0$ and $avgCosTod_0$ together achieves the best results.

Although measuring the similarity between documents is a good indicator for QPP, a coherent set may not always have a high average similarity, because of outliers and noise in D. Vinay et al. [128] define three measures: the Cox Lewis clustering tendency, query perturbation, and document perturbation. Through a modified version of the CoxLewis clustering tendency test, the first measure evaluates D for the existence of either natural groupings or randomness. Vinay et al. introduce query and document perturbation. The former modifies the query issued by a random noise, and observes the rank change in retrieval results. In our work we apply this measure by using d_0 as the issued query. Vinay et al. report that this measure is not able to predict the query performance effectively. In an affirming manner, we observe that this feature is not effective in keyphrase extraction. For this reason, we are not reporting the results of the query perturbation feature.

Different tests of clustering tendency exist in the literature. The Hopkins test [133] and the CoxLewis statistic [134] are two such tests in which the points in the original set and the randomly generated points are compared to determine the randomness of the set. If a higher similarity to random points is observed, then the original set is randomly distributed in the space.

These tests are suitable when there are only a few dimensions, but they are not directly applicable to high-dimensional hyperspaces. In a high-dimensional space such as $|V|$ dimensions, where V is typically in the order of thousands, a randomly generated point will most probably be distant from D as the probability space is large. In order to limit the probability space, Vinay et al. [128] propose using a document in D as a skeleton for the random generation, and avoid creating a random document composed of unlikely term combinations. The CoxLewis test selects a document randomly from D, and assigns random term weights to its non-zero dimensions to form a random document vector rd_i. Random

weights are between 0 and the maximum term weight appearing in set D. This generation strategy keeps the randomly generated points in a minimal hyper-rectangle containing all the documents in D.

Let nd_{i1} denote the nearest neighbour of the random vector rd_i in D, and let nd_{i2} be the nearest neighbour of nd_{i1} in D. The proportion of the similarity $cosSim(nd_{i1}, nd_{i2})$ to $cosSim(nd_{i1}, rd_i)$ tests whether the injected random vector can be more similar to a document in D than any other document in D. This test is repeated with $|D|/2$ random vectors, and the average of these tests is used as the CoxLewis score:

$$CoxLewisTest = \sum_{i=0}^{|D|/2} \frac{cosSim(nd_{i1}, nd_{i2})}{cosSim(nd_{i1}, rd_i)} \tag{6.8}$$

Document perturbation was first described by Vinay et al. [128] using VSM, and has recently been adapted to language models as rank robustness [130]. Similar to the CoxLewis test, the effect of adding random noise to the documents in D is tested. Given the document set D, when the documents are in descending order according to $cosSim(d_i, d_j)$ values that is, numbered from the most similar to the least the function $rank(d_i, D, d_j)$ is the rank of document d_j with respect to document d_i. The value of $rank(d_i, D, d_i)$ is equal to 1 with similarity 1.0 when documents are unique in D. The test modifies d_i by adding noise, and checks whether the set D contains documents more similar than d_i to perturbed d_i ($noise(d_i, \alpha)$). In a document set formed of unrelated documents, the rank of d_i does not change, whereas in a coherent set, rank change is expected to be high. Let α be a parameter controlling $noise(d_i, \alpha)$, and the function $noise(di, \alpha)$ return a vector perturbed by adding noise to each dimension of vector d_i. The noise is generated using a Gaussian distribution with mean equal to 0, and deviation equal to α. The overall rank change for a noise level is calculated by repeating the test 10 times for all documents in D

$$docPerturb(\alpha) = \sum_{i=1}^{|D|} \sum_{0}^{10} rank[noise(d_i, \alpha], D, d_i] \tag{6.9}$$

The overall *docPerturb* feature is the slope of the line that best fits the $docPerturb(\alpha)$ values. The document perturbation test uses the noise levels

$\alpha = \{0.1, 1, 10, 100\}$. If the slope is positive, then the rank changes as the noise level increases, and the set is assumed to be coherent.

6.1.3.2 Language Model Based Features

Language models are used in different applications of information retrieval research [135, 136]. Unigram language models are formulated by a bag-of-words assumption, and ordering of words is not taken into account. A simple estimate of the probability of generating a term t_i from a document d_j is the maximum likelihood estimate (MLE) that is, the relative frequency of t_i in d_j.

MLE usually results in a probability distribution with sharp changes, which assigns zero probability to terms not appearing in the document. Smoothing is a technique applied to resolve these problems. The probabilities of low or non-occurring terms are increased, and the probabilities of frequent terms are degraded. JelinekMercer smoothing combines the MLE of the whole document collection with a documents MLE, providing a smoother probability distribution. Two language models are combined by a weighted average controlled by the parameter λ. We used the same value as utilized in Townsend et al. [16], which is 0.6. The linear combination of MLE of a document d_j with the whole background collection (all the Wikipedia articles) is given by

$$P(w|d) = \lambda P_{ml}(w|d) + (1 - \lambda)P_{wiki}(w) \tag{6.10}$$

With the above probability estimate for each term, we derive a simplified clarity score motivated by the score defined by Townsend et al. [16]. The relevance of a term t to the query phrase w and original document d_0, $P(t|w, d_j)$, is modelled by

$$P(t|w, D) = \sum_{j=1}^{|D|} P(t|d_j)P(d_j) \tag{6.11}$$

where $P(t|d_j)$ reflects the probability of observing term t in the document d_j in the set D. The probability $P(d_j)$ is uniform for all documents in D, and is

equal to $1/|D|$. The clarity measure is the KullbackLeibler (KL) divergence [137] of the retrieved set D from the whole background corpus, defined as

$$Clarity = \sum_{t \in D} P(t|w, D) log(\frac{P(t|w, D)}{P_{ML}(t|Wiki)}) \tag{6.12}$$

KL divergence compares two different probability models. It is used as a document similarity function in various information retrieval tasks, such as text clustering and categorization [138, 139]. In a similar fashion to $CosCentrTod_0$ and $avgCosTod_0$ features, the relationship of the retrieved set D to the original document d_0 is investigated using the $KLDocsFromd_0$ feature. This feature is simply the KL divergence of the language model $P(t|w, D)$ from d_0, calculated according to

$$KLDocsfromd_0 = \sum_{t \in D} P(t|w, D) log(\frac{P(t|w, D)}{P_{ML}(t|d_0)}) \tag{6.13}$$

6.1.4 Learning to Classify Keyphrases

Keyphrase extraction can be considered as a classification task with two classes: keyphrase or non-keyphrase. For a specific domain or genre, a supervised machine learning algorithm analyses, learns and classifies keyphrases. Previous work on keyphrase extraction suggests that different types of corpora behave differently, and thus should be trained for each applied domain [90, 125].

The Naive Bayes learning algorithm uses the Bayesian rule to infer the probability of class membership, given the features. Using the independence assumption, the probability of keyphraseness $P(keyphrase|F_w)$ is calculated, where F_w is the feature set of phrase w. This probability is estimated from the training corpus using the Bayesian rule as given by

$$P(keyphrase|F_w) = P(keyphrase)\Pi_{f \in F_w} P(f|keyphrase) \tag{6.14}$$

The class prior $P(keyphrase)$ is low, since the proportion of keyphrases to non-keyphrases in a document is very low. As a result of this imbalance between the

classes, the probability $P(keyphrase|F_w)$ is low, and it is not possible to use strict thresholds for classification. For this reason, in contrast to other classification methods, thresholds are not applied for keyphraseness scores. When the target keyphrase size is 5, the top five ranking keyphrases are returned as the output, no matter how low the probability value is. Using this method, the prior probability can be neglected in calculations, as it will be the same for each candidate w.

Kea [125] reports a higher precision when the feature values are discretized using the minimum discrimination length(MDL) [140]. The features we have introduced behave similarly, and their precision decreases when supervised discretization is not applied to the features. Discretization is done by splitting the value ranges so as to minimize the entropy of the training population with respect to the probability of keyphraseness. For this reason, we apply MDL discretization to all the features.

6.2 Corpus and Evaluation Metrics

Keyphrases of a document should be assigned in accordance with the intention and emphasis of the text. Naturally, the author of the document is well-aware of the intentions of the text. Thus, keyphrases assigned by the author(s) of the text can be considered as the ground truth in keyphrase generation. However, in keyphrase extraction this poses a problem, since not all author assigned keyphrases appear in the documents. Some recent works use human judges to annotate the documents, using only keyphrases that appear in the documents [96, 94, 141]. In our opinion, this method is less reliable as keyphrase assignment is a subjective task depending on the background of the human judge, and when author assigned keyphrases are available for evaluation they should be preferred.

In the experiments, a corpus composed of 75 journal articles is used. The same corpus is used in several works on keyphrase extraction [5, 90, 125]. As reported in Turney [90] the corpus is composed of journal articles from different domains, which is shown in Table 6.2. About 82% of the keyphrases appear in the articles, so 18% of keyphrases cannot be detected by keyphrase extraction systems.

Journal Name	#Articles	KeyPh./Doc.	Words/Doc.
Journal of the International Academy of Hospitality Research	6	6.2	6,299
Psycoloquy	20	8.4	4,350
The Neuroscientist	2	6.0	7,476
Journal of Computer-aided Molecular Design	14	4.7	6,474
Behavioral & Brain Sciences Preprint Archive	33	8.4	17,522
All	75	7.5	10,781

Table 6.2: Corpus of journal articles and its attributes.

In order to highlight the disadvantages of systems that solely depend on in-document features, an experiment using a corpus of shorter documents is conducted. To this end, the abstracts of articles are used. The average document length in the abstracts is 156 words. 44.8% of the keyphrases appear in the text of abstracts, which means that 55.2% of keyphrases cannot be extracted.

Not all keyphrases occur in the background corpus. 14.17% of the keyphrases never appear in Wikipedia, while 16.6% appear in less than five different Wikipedia articles. This is simply caused by the productivity of languages in phrases, especially due to domain specific technical terms. It should be noted that the articles in the corpus date back to 1993. An observation that hints at the rate of phrase production is that in recent articles built for SemEval 2010 task 5 [141], a higher percentage of keyphrases never appear in Wikipedia. Yet, it is possible to solve this problem by using a larger knowledge base such as a search engine, or a domain specific corpus stored in a digital library. In most practical applications of extracted keyphrases the importance of detecting such uncommon keyphrases is low.

Unfortunately for Turkish language building a corpus is even more challenging as the keyphrases used in journal articles are not available in a background corpora such as Wikipedia. Even in our best attempt for building a corpora, only 38% of the keyphrases appears in Turkish Wikipedia. For this reason it was not possible to observe a positive impact of the method in Keyphrase Extraction task.

Keyphrase extraction systems are usually evaluated using precision and recall, which are defined in terms of sets of phrases. In a processed source document, let the set A be the author-assigned phrases, and let A' be the subset of A formed

of phrases appearing in the document. Let E be the set of phrases extracted automatically by the system. The recall is calculated according to

$$Recall(A, E) = \frac{|A \cap E}{|A'|} \tag{6.15}$$

To avoid penalizing keyphrase extraction systems for keyphrases that cannot be extracted, the recall value is calculated with respect to the set A'. The precision is calculated from

$$Precision(A, E) = \frac{|A \cap E}{|E|} \tag{6.16}$$

6.3 Results

The test and training data are chosen so as to be compatible with the experiments performed by Turney [90] and Frank et al. [125]. Of the journal articles, 20 are reserved for testing and the remaining 55 documents are used for training. In the corpus of abstracts, 53 documents are used for training and 19 for testing; three have been omitted, as they lack an abstract.

Tables 6.3 and 6.4 present the results of the full-text and abstract experiments respectively. For both of the experiments, the precision, recall and average number of correct keyphrases per document are given when 5, 10 and 15 keyphrases are extracted for each article. The results of Kea [125][5] are also provided for comparison. In Tables 6.3 and 6.4, $inDoc + QPP$ denotes the experiments using all of the features defined in Table 6.1. The feature set $inDoc$ denotes $tf * idf$ and $firstPosition$. $QPPFeats$ denotes all features, excluding those of $inDoc$.

In full-text articles, the Kea algorithm is able to extract 38% of the keyphrases appearing in the articles, when 15 keyphrases are extracted for each document. Journal articles concisely define the contribution of the document in early sections, and keyphrases are used more frequently in abstracts and introductory portions of the document. This is why the $firstPosition$ feature achieves high

[5]Kea 3.0 version, downloaded from; http://www.nzdl.org/Kea/download.html

accuracy in scientific articles.

As seen in Table 6.3, the effectiveness of $QPPFeats$ is lower than that of Kea and $inDoc$. We have observed that QPP features tend to have similar values for domain-specific phrases, keyphrases and sub- or superphrases of keyphrases. In fact, for a superset of a keyphrase, a similar set of documents is usually retrieved from the background corpus. For example, for the keyphrase "obsessive compulsive disorder" and the subphrases "obsessive compulsive", "compulsive disorder" as well as the superphrase "obsessive compulsive personality disorder", similar sets of documents are retrieved from the background corpus. Since all QPP features are calculated using the retrieved documents, the feature values are almost identical to each other. In order to tackle this problem, and to improve the effectiveness of the system, we integrated QPP features with in-document features in full-text articles. As a result, the $inDoc + QPPFeats$ system achieves the best recall values when compared to the Kea and $inDoc$ algorithms.

The results of the experiment using abstracts, as shown in Table 6.4, reveal a defect of $inDoc$ features in shorter documents. Whereas $tf * idf$ and $firstPosition$ are able to achieve high precision and recall values in full-text experiments, their performance is poor in the abstract corpus. As indicated previously, the $firstPosition$ feature, depending on the structure of the document, is effective in full-text articles. However, in shorter documents, the $firstPosition$, which is the normalized distance from the start of the document to the word, is subject to more noise. Whereas a change in distance by a few words does not change the value of $firstPosition$ in long documents, it changes the distance value significantly in short documents. $tf * idf$ values are formed of two components of term frequency and inverse document frequency. Inverse document frequency gets larger values when the phrase occurs in fewer documents. In short texts, most phrases occur once or a couple of times. Since frequencies of terms are similar, a high value of $tf * idf$ is assigned to a phrase that appears infrequently in the corpus. Thus, for short documents it is even possible to observe the highest $tf * idf$ values in spelling errors and typos.

The QPP features are not extracted directly from the document, and can be calculated for any phrase, regardless of how many times it occurs in the text, if it ever does. This makes them more robust to changes in the length of the documents. For abstracts, the recall values of $QPPFeats+inDoc$ and $QPPFeats$

are better than those of the Kea algorithm.

On the one hand, QPPFeats + inDoc correctly identifies 53% of author-assigned keyphrases appearing in the abstract when 15 keyphrases per article are extracted. On the other hand, it can be observed that in-document features degrade the effectiveness of $QPPFeats + inDoc$ in short documents, since 55% of the keyphrases are identified when only $QPPFeats$ are used. Both the $inDoc$ and Kea algorithms are able to extract only a maximum of 40% of the keyphrases, which is 15% less than $QPPFeats$ when 15 keyphrases are extracted. Their recall is about half of the QPP features when only five keyphrases are extracted.

One important advantage of $QPPFeats$ is that it is possible to calculate them for phrases not appearing in the original document. In the keyphrase generation problem, in contrast to extraction, the algorithm should be able to generate phrases not appearing in the text, and should add keyphrase candidates from a prior knowledge, such as a background corpus or taxonomy. Expanding the extracted candidate keyphrases is a research topic by itself, and is left as a future work. However, in order to demonstrate the fact that QPPFeats can be used in keyphrase generation, we have performed an additional experiment. In the abstracts corpus we have manually added the 55% of the keyphrases that do not occur in their respective abstract to extracted candidate phrase lists, and repeated the experiment. In this setting, when the 15 top-scoring keyphrase candidates are selected, the number of correct keyphrases generated is improved by 42.5%, and the recall value is increased to 78%, with a precision of 20%. The precision value is even higher than the result of $inDoc + QPPFeats$ in the full-text article experiments: that is, the $QPPFeats$ can extract more keyphrases only by observing the abstracts.

When the QPP features are studied individually, it is observed that the two features $CosCentrTod_0$ and $avgCosTod_0$ have the greatest impact on keyphrase extraction. Our experiments suggest that using these two features provides the greatest improvement in keyphrase extraction. Two features document perturbation (i.e. ranking robustness) and clarity can be successfully used to improve the results in both QPP and keyphrase extraction.

Most of the earlier work on keyphrase extraction focus on research articles. However, there is an increasing interest in applying keyphrase extraction in shorter documents, such as Twitter messages [142] and news articles [96]. In

Algorithms	Recall			Precision			KeyPerDoc		
	5	10	15	5	10	15	5	10	15
Kea	0.16	0.22	0.29	0.13	0.17	0.07	0.63	0.84	1.10
$inDoc + QPPFeats$	0.27	0.44	0.53	0.21	0.17	0.14	1.05	1.68	2.05
$QPPFeatures$	0.29	0.47	0.55	0.22	0.18	0.14	1.11	1.79	2.11
$inDoc$	0.18	0.29	0.40	0.14	0.11	0.10	0.68	1.11	1.53

Table 6.3: Keyphrase Extraction full-text experiment results.

Algorithms	Recall			Precision			KeyPerDoc		
	5	10	15	5	10	15	5	10	15
Kea	0.19	0.32	0.38	0.25	0.20	0.17	1.25	2.05	2.50
$inDoc + QPPFeats$	0.24	0.32	0.40	0.34	0.22	0.19	1.70	2.20	2.80
$QPPFeatures$	0.11	0.21	0.28	0.16	0.15	0.13	0.80	1.45	1.95
$inDoc$	0.11	0.27	0.36	0.15	0.19	0.17	0.75	1.85	2.50

Table 6.4: Keyphrase Extraction Abstracts experiment results.

this research, the potential problems of features commonly used in keyphrase extraction were shown through experiments. Although these features are useful in full-text articles, their effectiveness drops in short documents. Features extracted from a background corpus are able to solve this problem. We have shown that while the introduced QPP features improve keyphrase extraction in full-text articles, the improvement is much more significant for shorter documents like abstracts.

Furthermore, our features are not dependent on the occurrences of the phrases in the original document, and can be calculated for phrases that never appear in the document. All in all, this work aimed to establish a link between the problems of QPP and keyphrase extraction. We believe that this work contributes to the research on finding keyphrases by removing the constraints imposed by features directly extracted from the occurrences in the original document. A careful investigation of techniques for creating candidate keyphrase lists by mining related articles or semantically related phrases enables our algorithm to generate keyphrases. The techniques used in this research may lead to more general methods that are able to operate on different genres and perform generation instead of extraction.

Chapter 7

Conclusion

Given two word pairs "chicken-egg" and "chicken-golf ball" perhaps anyone can immediately argue that the first pair is more related, using his/her prior knowledge of the concepts. However in a context, where the reader knows that golf balls are used to fake eggs to encourage chickens to lay nests a contextual semantic relatedness emerges. In this research I have investigated if it is possible to model globally known semantic relationships such as "chicken-egg", and with this basic knowledge about concepts whether it is possible to improve the performance of algorithms in tasks such as topic segmentation, summarization and keyphrase extraction.

The semantic relatedness methods evaluated and compared in Chapter 3 utilize two different knowledge sources, namely Thesauri and corpus statistics. The question of which performs better is both an important and sophisticated question to answer. The experiments showed that while WordNet based semantic relatedness scores perform slightly better in the Word Association task when compared to semantic spaces, they achieve notably worse scores in TOEFL synonymy questions. Such discrepancies in the results can be partly attributed to the fact that relationships between different categories (verbs, adjectives and adverbs) are not modelled by the classical relationships in WordNet. In Word Association task most of the word pairs are nouns, while in TOEFL synonymy questions there are nouns, adjectives and verbs. These contradicting results makes it difficult to decide which semantic relatedness measure to use in a high level task. In such a task, like topic segmentation the performance of WordNet based methods

are much lower than statistics based methods. Only the recall of Resnik's SR method which is actually a hybrid method integrating both corpus statistics and the structure of WordNet is higher than 50% in topic segmentation.

Although semantic relatedness is an active research area gaining popularity as it is an interesting topic from both Cognitive science and Linguistics aspects, natural language processing applications utilizing them are scarce. In order to investigate how knowing the common lexical semantics helps in a high level task such as topic segmentation, novel algorithms were implemented. These algorithms enabled us to compare different semantic relatedness methods defined in the literature using real discourse in the form of text documents. It became evident from our experiments that the use of corpus statistics achieves better results than WordNet based methods. Also it is possible to observe the importance of Singular Value Decomposition as a tool for building semantic spaces which reduces the running time and increases the effectiveness.

Furthermore, in the automated text summarization task a similar result is obtained supporting the use of corpus statistics in semantic relatedness. Again in this setting it is possible to observe better results, when semantic spaces are utilized. These two applications are proofs that using lexical semantics on global level in natural language tasks can be useful. However, on the downside the results are usually lower when compared to methods modelling lexical semantics in the local context, such as the Bayesian topic model method of Eisenstein and Barzilay [61] in topic segmentation or the summarization methods of Ozsoy et al [117]. Even though the results of our experiments do not disprove the existence of algorithms only using common semantic relatedness measures, contextual semantic relatedness as in "chicken-golf ball" is crucial for achieving the best results possible.

With a slightly different approach in keyphrase extraction problem, keyphraseness of phrases are measured based on a simpler semantic relatedness measure extracted from a background corpora. This not only improves keyphrase extraction performance, but is also a step towards more general solutions for keyphrase generation. On the other hand, this research established a link between keyphrase extraction and query performance prediction. However, this method has a problem perhaps common to many information retrieval and natural language processing tasks, requiring a very large corpora. Since all keyphrases must be present

in the background corpora, the algorithm can only detect keyphrases that are present in the background corpora. While in the English corpora this is not a problem, as Wikipedia is adequate and large enough in size for this language, for Turkish most of the Keyphrases in the corpora are missing.

7.1 Future Work

Given the encouraging results achieved in these tasks, it is tempting to use semantic relatedness in different applications, where this additional lexical semantics knowledge may prove to be useful. Two of the potential tasks are in machine translation and speech-to-text tasks, where in both of the problems there are ambiguities which can benefit from lexical cohesion and semantics. In machine translation by establishing links between semantic spaces of two languages, words can be selected in order to preserve the semantic relatedness level in the translation. The links between semantic spaces can simply be established through language-to-language dictionaries (may be the inter-language links in Wiktionary). In speech-to-text, for an acoustic signal words with similar phonetic properties are determined. For a given sound it is possible to have a long list of candidate words. From this list it may be possible to select the ones maximizing the semantic relatedness level of the text.

Alternatives to latent semantic analysis by SVD are probabilistic LSA models and lately Dirichlet Latent Allocation (LDA methods). While we have done some initial experiments with PLSA and achieved similar results to SVD based LSA in document retrieval, probabilistic models are more intuitive and easier to modify. However to the best of my knowledge PLSA and LDA are primarily used for context-by-term matrices, which may not be as effective as term-by-term matrices in semantic relatedness. It would be curious to actually add comparisons to semantic relatedness methods using PLSA and LDA.

We have specifically chosen Wikipedia articles as a background corpora, in order to be able to capture more comprehensive domain independent semantic relationships. However, some relationships can only be observable only when focused on a specific domain, but otherwise will be cluttered in a corpora such as

Wikipedia. Thus, the effect of domain dependency can be investigated. Furthermore, it may be possible to build a solution, that stores a hierarchy of semantic spaces built according to the categories available in Wikipedia and for a document chooses the most relevant semantic space.

The topic segmentation algorithm in this research simply observes the local context, and derives a score able to highlight the topic shifts. However in this algorithm the decisions are made locally, the whole segmentation is never considered. This may be a factor degrading the effectiveness of the algorithms. Another alternative common to algorithms achieving the best results is trying all possible topic segmentations using dynamic programming. While it may be relatively more obvious to define a dynamic programming solution, it is difficult to find one which will limit the number of semantic relatedness calculations to an acceptable level (probably below an asymptotic complexity of $O(|V|^2)$, otherwise the running time of the algorithm will restrain its use in practical applications.

Our keyphrase extraction algorithm can be converted to a keyphrase generation algorithm if a candidate keyphrase list can be produced by including related phrases. First of all, a combination of both searching for candidates in related documents and evaluating these phrases by the introduced method can be interesting. Utilizing an algorithm similar to spreading activation can prove to be useful to grow a graph of both documents and their phrases to attack the more general problem of keyphrase generation.

Also it might be interesting to use the SVD based semantic relatedness methods which only works for words but not phrases, by an algorithm similar to Baroni and Lecci [121]. In their work, the semantic relatedness of noun phrases are defined by some operations performed on the semantic space vectors of its components (words). This method is especially interesting as a successful algorithm does not have to collect or observe the co-occurrence of the phrase, but can model its semantic relatedness by just observing its words individually. While such an extension is difficult, it promises to calculate all possible phrases that can be constructed in the language.

Bibliography

[1] A. M. Turing, "Computing Machinery and Intelligence," *Mind*, vol. 59, no. 236, pp. 433–460, 1950.

[2] M. A. K. Halliday and R. Hasan, *Cohesion in English*. Longman, 1976.

[3] G. Ercan, "Automated Text Summarization and Keyphrase Extraction," Master's thesis, 2006.

[4] G. Ercan and I. Cicekli, "Lexical Cohesion Based Topic Modeling for Summarization," in *Proceedings of Computational Linguistics and Intelligent Text Processing*, pp. 582–592, 2008.

[5] G. Ercan and I. Cicekli, "Using lexical chains for keyword extraction.," *Inf. Process. Manage.*, vol. 43, no. 6, pp. 1705–1714, 2007.

[6] P. P. Peter D. Turney, "From frequency to meaning: Vector space models of semantics," *Journal of Artificial Intelligence Research (JAIR)*, vol. 37, pp. 141–188, 2010.

[7] R. Rapp, "Discovering the Senses of an Ambiguous Word by Clustering its Local Contexts," in *Proceedings of the 28th Annual Conference of the Gesellschaft für Klassifikation*, pp. 521–528, 2004.

[8] T. K. Landauer and S. T. Dumais, "A solution to Plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge," *Psychological Review*, vol. 104, no. 2, pp. 211–240, 1997.

[9] R. Barzilay and M. Elhadad, "Using Lexical Chains for Text Summarization," in *Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization*, pp. 10–17, 1997.

[10] N. Stokes, J. Carthy, and A. F. Smeaton, "SeLeCT: a lexical cohesion based news story segmentation system," *Artificial Intelligence Communications*, vol. 17, no. 1, pp. 3–12, 2004.

[11] M. Galley, K. McKeown, E. Fosler-Lussier, and H. Jing, "Discourse Segmentation of Multi-Party Conversation.," in *ACL* (E. W. Hinrichs and D. Roth, eds.), pp. 562–569, ACL, 2003.

[12] G. Salton, J. Allan, and A. Singhal, "Automatic Text Decomposition and Structuring.," *Inf. Process. Manage.*, vol. 32, no. 2, pp. 127–138, 1996.

[13] S. Jones and M. S. Staveley, "Phrasier: A System for Interactive Document Retrieval Using Keyphrases," in *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, (New York), pp. 160–167, ACM, 1999.

[14] E. Hovy, "Text Summarization," in *The Oxford Handbook of Computational Linguistics* (R. Mitkov, ed.), Oxford Handbooks in Linguistics, pp. 583–598, Oxford: Oxford University Press, 2003.

[15] C. Gutwin, G. Paynter, I. Witten, C. Nevill-Manning, and E. Frank, "Improving browsing in digital libraries with keyphrase indexes," *Journal of Decision Support Systems*, vol. 27, pp. 81–104, 1999.

[16] S. Cronen-Townsend, Y. Zhou, and W. B. Croft, "Predicting query performance.," in *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, (New York), pp. 299–306, ACM, 2002.

[17] G. Amati, C. Carpineto, and G. Romano, "Query Difficulty, Robustness, and Selective Application of Query Expansion," in *Advances in Information Retrieval*, vol. 2997, pp. 127–137, Berlin: Springer, 2004.

[18] E. Yom-Tov, D. Carmel, A. Darlow, D. Pelleg, S. Errera-Yaakov, and S. Fine, "Juru at TREC 2005: Query Prediction in the Terabyte and the Robust Tracks.," in *Proceedings of the 2005 Text Retrieval Conferance (TREC)* (E. M. Voorhees and L. P. Buckland, eds.), vol. Special Publication 500-266, NIST, 2005.

[19] A. Budanitsky and G. Hirst, "Evaluating WordNet-based Measures of Lexical Semantic Relatedness," *Computational Linguistics*, vol. 32, no. 1, pp. 13–47, 2006.

[20] F. D. Saussure, *Cours de linguistique générale*. Paris: Bayot, 1916.

[21] D. Cruse, *Lexical semantics*. Cambridge Univ Pr, 1986.

[22] 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.

[23] L. Wittgenstein, *Philosophical Investigations*. Blackwell, 1953. Translated by G.E.M. Anscombe.

[24] Z. Harris, *Methods in Structural Linguistics*. Chicago: University of Chicago Press, 1951.

[25] J. R. Firth, "A synopsis of linguistic theory 1930-55.," vol. 1952-59, pp. 1–32, 1957.

[26] C. Osgood, *The measurement of meaning*. Univ of Illinois Pr, 1975.

[27] N. Chomsky, *Syntactic structues*. The Hague: Mouton, 1957.

[28] M. Marcus, G. Kim, A. Marcinkiewicz, R. Macintyre, A. Bies, M. Ferguson, K. Katz, and B. Schasberger, "The penn treebank: Annotating predicate argument structure," in *In ARPA Human Language Technology Workshop*, pp. 114–119, 1994.

[29] A. Goksel and C. Kerslake. Taylor & Francis Group, 2005.

[30] G. Lewis, *Turkish Grammar*. 2 ed., 2001.

[31] O. Istek, "A Link Grammar For Turkish," Master's thesis, 2006.

[32] M. Carstairs, *An Introduction to English Morphology*. Edinburgh University Press, 2002.

[33] D. Jurafsky and J. Martin, *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition*. Prentice Hall series in artificial intelligence, Prentice Hall, 2000.

[34] M. F. Porter, "An Algorithm for Suffix Stripping," *Program*, vol. 14, no. 3, pp. 130–137, 1980.

[35] H. Rubenstein and J. B. Goodenough, "Contextual correlates of synonymy.," *Commun. ACM*, vol. 8, no. 10, pp. 627–633, 1965.

[36] G. Salton, A. Wong, and C. T. Yu, "Automatic indexing using term discrimination and term precision measurements.," *Inf. Process. Manage.*, vol. 12, no. 1, pp. 43–51, 1976.

[37] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman, "Indexing by Latent Semantic Analysis," *Journal of the American Society of Information Science*, vol. 41, no. 6, pp. 391–407, 1990.

[38] K. Lund and C. Burgess, "Producing high-dimensional semantic spaces from lexical co-occurrence," *Behavior Research Methods, Instrumentation, and Computers*, vol. 28, pp. 203–208, 1996.

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

[40] E. Terra and C. L. A. Clarke, "Frequency estimates for statistical word similarity measures," in *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, (Morristown, NJ, USA), pp. 165–172, Association for Computational Linguistics, 2003.

[41] J. Bullinaria and J. Levy, "Extracting semantic representations from word co-occurrence statistics: stop-lists, stemming, and svd," *Behavior research methods*, pp. 1–18, 2012.

[42] R. L. Cilibrasi and P. M. Vitanyi, "The Google Similarity Distance," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, pp. 370–383, 2007.

[43] S. Padó and M. Lapata, "Dependency-Based Construction of Semantic Space Models," *Computational Linguistics*, vol. 33, no. 2, pp. 161–199, 2007.

[44] H. Kozima and T. Furugori, "Similarity between words computed by spreading activation on an English dictionary," in *Proceedings of the sixth conference on European chapter of the Association for Computational Linguistics*, (Morristown, NJ, USA), pp. 232–239, Association for Computational Linguistics, 1993.

[45] T. Pedersen, S. Patwardhan, and J. Michelizzi, "WordNet::Similarity: Measuring the Relatedness of Concepts," in *Proceedings of the Demonstration Papers at HLT-NAACL*, (Boston), 2004.

[46] J. Morris and G. Hirst, "Lexical cohesion computed by thesaural relations as an indicator of the structure of text," *Comput. Linguist.*, vol. 17, no. 1, pp. 21–48, 1991.

[47] M. Jarmasz and S. Szpakowicz, "Roget's thesaurus and semantic similarity," in *Conference on Recent Advances in Natural Language Processing*, pp. 212–219, 2003.

[48] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank Citation Ranking: Bringing Order to the Web.," Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[49] M. Strube and S. Ponzetto, "WikiRelate! Computing semantic relatedness using Wikipedia," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 21, p. 1419, Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.

[50] E. Gabrilovich and S. Markovitch, "Computing semantic relatedness using Wikipedia-based explicit semantic analysis," in *Proceedings of the 20th international joint conference on Artifical intelligence*, IJCAI'07, (San Francisco, CA, USA), pp. 1606–1611, Morgan Kaufmann Publishers Inc., 2007.

[51] D. Milne and I. H. Witten, "An effective, low-cost measure of semantic relatedness obtained from Wikipedia links," in *AAAI 2008*, 2008.

[52] R. Mihalcea and A. Csomai, "Wikify!: linking documents to encyclopedic knowledge," in *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, (New York, NY, USA), pp. 233–242, ACM, 2007.

[53] G. Youmans, "A new tool for discourse analysis: The vocabulary management profile," *Language*, vol. 67, no. 4, pp. 763–789, 1991.

[54] M. A. Hearst, "TextTiling: Segmenting Text into Multi-paragraph Subtopic Passages," *Computational Linguistics*, vol. 23, pp. 33–64, March 1997.

[55] G. Salton, J. Allan, and A. Singhal, "Automatic Text Decomposition and Structuring," *Information Processing and Management Journal*, vol. 32, no. 2, pp. 127–138, 1996.

[56] F. Choi, "Advances in domain independent linear text segmentation," in *Proceedings of the First Conference on North American Chapter of the Association for Computational Linguistics*, pp. 26–33, 2000.

[57] X. Ji and H. Zha, "Domain-independent text segmentation using anisotropic diffusion and dynamic programming," in *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pp. 322–329, 2003.

[58] M. Utiyama and H. Isahara, "A statistical model for domain-independent text segmentation," in *ACL '01: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pp. 499–506, 2001.

[59] I. Malioutov and R. Barzilay, "Minimum Cut Model for Spoken Lecture Segmentation.," in *ACL* (N. Calzolari, C. Cardie, and P. Isabelle, eds.), The Association for Computer Linguistics, 2006.

[60] A. C. Jobbins and L. J. Evett, "Text segmentation using reiteration and collocation," in *Proceedings of the 17th international conference on Computational linguistics*, pp. 614–618, 1998.

[61] J. Eisenstein and R. Barzilay, "Bayesian unsupervised topic segmentation," in *EMNLP '08: Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 334–343, 2008.

[62] T. Brants, F. Chen, and I. Tsochantaridis, "Topic-based document segmentation with probabilistic latent semantic analysis," in *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, (New York, NY, USA), pp. 211–218, ACM, 2002.

[63] T. Hofmann, "Probabilistic Latent Semantic Indexing," in *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 50–57, 1999.

[64] H. Misra, F. Yvon, O. Cappé, and J. M. Jose, "Text segmentation: A topic modeling perspective.," *Inf. Process. Manage.*, vol. 47, no. 4, pp. 528–544, 2011.

[65] F. Choi, P. Wiemer-Hastings, and J. Moore, "Latent Semantic Analysis for Text Segmentation," in *Proceedings of 6th EMNLP*, pp. 109–117, 2001.

[66] Y. Bestgen, "Improving Text Segmentation Using Latent Semantic Analysis: A Reanalysis of Choi, Wiemer-Hastings, and Moore (2001)," *Computational Linguistics*, vol. 32, no. 1, pp. 5–12, 2006.

[67] D. Beeferman, A. Berger, and J. Lafferty, "Statistical Models for Text Segmentation," *Machine Learning*, vol. 34, no. 1, pp. 177–210, 1999.

[68] L. Pevzner and M. A. Hearst, "A critique and improvement of an evaluation metric for text segmentation," *Computational Linguistics*, vol. 28, no. 1, pp. 19–36, 2002.

[69] R. Brandow, K. Mitze, and L. F. Rau, "Automatic condensation of electronic publications by sentence selection," *Inf. Process. Manage.*, vol. 31, no. 5, pp. 675–685, 1995.

[70] H. P. Edmundson, "New Methods in Automatic Abstracting," *Journal of the Association for Computing Machinery*, vol. 16, no. 2, pp. 264–285, 1969.

[71] J. Kupiec, J. Pedersen, and F. Chen, "A trainable document summarizer," in *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (New York, NY, USA), pp. 68–73, ACM Press, 1995.

[72] S. Teufel and M. Moens, "Summarizing Scientific Articles: Experiments with Relevance and Rhetorical Status," *Computational Linguistics*, vol. 28, no. 4, pp. 409–445, 2002.

[73] C.-Y. Lin and E. H. Hovy, "Identifying Topics by Position," in *Proceedings of 5th Conference on Applied Natural Language Processing*, (Washington D.C.), March 1997.

[74] H. P. Luhn, "The automatic creation of literature abstracts," *IBM Journal of Research and Development*, vol. 2, pp. 159–165, 1958.

[75] D. Marcu, "Improving summarization through rhetorical parsing tuning," in *Proceedings of The Sixth Workshop on Very Large Corpora*, (Montreal, Canada), pp. 206–215, August 1998.

[76] D. R. Radev, H. Jing, and M. Budzikowska, "Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies," *NAACL/ANLP Workshop on Automatic Summarization, Seattle, WA, April 30, 2000*, May 2000.

[77] G. Erkan and D. R. Radev, "LexRank: Graph-based Lexical Centrality as Salience in Text Summarization," *Journal Of Artificial Intelligence Research, Volume 22, pages 457-479, 2004*, Sept. 2011.

[78] D. R. Radev, T. Allison, S. Blair-Goldensohn, J. Blitzer, A. Çelebi, S. Dimitrov, E. Drábek, A. Hakim, W. Lam, D. Liu, J. Otterbacher, H. Qi, H. Saggion, S. Teufel, M. Topper, A. Winkel, and Z. Zhang, "MEAD - A Platform for Multidocument Multilingual Text Summarization.," in *LREC*, European Language Resources Association, 2004.

[79] G. Hirst and D. St-Onge, *Lexical chains as representations of context for the detection and correction of malapropisms*, pp. 305–332. 1998.

[80] H. G. Silber and K. F. McCoy, "An Efficient Text Summarizer using Lexical Chains.," in *INLG* (M. Elhadad, ed.), pp. 268–271, The Association for Computer Linguistics, 2000.

[81] M. Brunn, Y. Chali, and B. Dufour, "The University of Lethbridge Text Summarizer at DUC 2002," in *Proceedings of the Document Understanding Conference (DUC)*, 2002.

[82] Y. Chali and S. Dubien, "University of Lethbridge's Participation in TREC 2004 QA Track.," in *TREC* (E. M. Voorhees and L. P. Buckland, eds.), vol. Special Publication 500-261, National Institute of Standards and Technology (NIST), 2004.

[83] L. F. Rau, P. S. Jacobs, and U. Zernik, "Information extraction and text summarization using linguistic knowledge acquisition.," *Inf. Process. Manage.*, vol. 25, no. 4, pp. 419–428, 1989.

[84] K. Knight and D. Marcu, "Statistics-Based Summarization—Step One: Sentence Compression," in *Proceedings of the 17th National Conference on Artificial Intelligence*, Cambridge, MA: MIT Press, 2000.

[85] I. Mani, B. Gates, and E. Bloedorn, "Improving Summaries by Revising Them.," in *ACL* (R. Dale and K. W. Church, eds.), ACL, 1999.

[86] J. G. Carbonell and J. Goldstein, "The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries.," in *SIGIR*, pp. 335–336, ACM, 1998.

[87] R. Barzilay and K. R. McKeown, "Sentence Fusion for Multidocument News Summarization," *Computational Linguistics*, vol. 31, pp. 297–328, September 2005.

[88] C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of summaries," in *Proc. ACL workshop on Text Summarization Branches Out*, p. 10, 2004.

[89] I. Mani, *Automatic Summarization*. John Benjamins Publishing Co., 2001.

[90] P. D. Turney, "Learning Algorithms for Keyphrase Extraction.," *Information Retrieval*, vol. 2, no. 4, pp. 303–336, 2000.

[91] A. Hulth, "Improved automatic keyword extraction given more linguistic knowledge," in *Proceedings of the 2003 conference on Empirical methods in Natural Language Processing (EMNLP)*, vol. 10, (Morristown), pp. 216–223, ACL, 2003.

[92] K. Barker and N. Cornacchia, "Using Noun Phrase Heads to Extract Document Keyphrases," in *Proceedings of the 13th Biennial Conference of the Canadian Society on Computational Studies of Intelligence: Advances in Artificial Intelligence* (H. J. Hamilton, ed.), vol. 1822 of *Lecture Notes in Computer Science*, (Berlin), pp. 40–52, Springer, 2000.

[93] P. D. Turney, "Coherent keyphrase extraction via web mining," in *International Joint Conferences on Artificial Intelligence (IJCAI)*, vol. 18, pp. 434–442, LAWRENCE ERLBAUM ASSOCIATES LTD, 2003.

[94] T. D. Nguyen and M.-Y. Kan, "Keyphrase Extraction for Scientific Publications," in *Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers*, vol. 4822 of *Lecture Notes in Computer Science*, (Berlin), pp. 317–326, Springer, 2007.

[95] R. Mihalcea and P. Tarau, "TextRank: Bringing Order into Texts," in *Proceedings of 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Morristown), pp. 404–411, ACL, July 2004.

[96] X. Wan and J. Xiao, "Exploiting neighborhood knowledge for single document summarization and keyphrase extraction.," *ACM Transactions on Information Systems*, vol. 28, no. 2, 2010.

[97] M. Sussna, "Word sense disambiguation for free-text indexing using a massive semantic network," in *Proc. of 2nd International Conference on Information and Knowledge Management*, (Arlington, Virginia), 1993.

[98] Z. Wu and M. Palmer, "Verb semantic and lexical selection," in *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics (ACL-1994)*, (Las Cruces (Mexico)), pp. 133–138, 1994.

[99] C. Leacock and M. Chodorow, "Combining local context and WordNet similarity for word sense identification," in *MIT Press* (C. Fellfaum, ed.), (Cambridge, Massachusetts), pp. 265–283, 1998.

[100] P. Resnik, "Using Information Content to Evaluate Semantic Similarity in a Taxonomy.," in *IJCAI*, pp. 448–453, Morgan Kaufmann, 1995.

[101] G. Hirst, *Semantic Interpretation and the Resolution of Ambiguity*. Studies in natural language processing, Cambridge University Press, 1992.

[102] 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.

[103] W. N. Francis and H. Kucera, "Brown Corpus Manual," tech. rep., Department of Linguistics, Brown University, Providence, Rhode Island, US, 1979.

[104] J. J. Jiang and D. W. Conrath, "Semantic similarity based on corpus statistics and lexical taxonomy," *CoRR*, vol. cmp-lg/9709008, 1997.

[105] G. Eryiğit and E. Adalı, "An affix stripping morphological analyzer for Turkish," in *Proceedings of the IASTED International Conference Artificial Intelligence And Applications*, pp. 299–304, 2004.

[106] B. N. Datta, *Numerical Linear Algebra and Applications (2. ed.)*. SIAM, 2010.

[107] M. Brand, "Fast Low-Rank Modifications of the Thin Singular Value Decomposition," *Linear Algebra and Its Applications*, vol. 415, no. 1, pp. 20–30, 2006.

[108] G. Gorrell, "Generalized Hebbian Algorithm for Incremental Singular Value Decomposition in Natural Language Processing.," in *EACL* (D. McCarthy and S. Wintner, eds.), The Association for Computer Linguistics, 2006.

[109] J. Steiger, "Tests for comparing elements of a correlation matrix.," *Psychological Bulletin*, vol. 87, no. 2, p. 245, 1980.

[110] P. Chen and P. Popovich, *Correlation: Parametric and nonparametric measures*. No. 137-139, Sage Publications, Incorporated, 2002.

[111] G. Pirrò and N. Seco, "Design, Implementation and Evaluation of a New Semantic Similarity Metric Combining Features and Intrinsic Information Content.," in *OTM Conferences (2)* (R. Meersman and Z. Tari, eds.), vol. 5332 of *Lecture Notes in Computer Science*, pp. 1271–1288, Springer, 2008.

[112] P. D. Turney, "Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL," *Proceedings of the Twelfth European Conference on Machine Learning, (2001), Freiburg, Germany, 491-502*, Dec. 2002.

[113] S. T. Dumais, "Improving the retrieval of information from external sources," *Behavior Research Methods, Instruments, & Computers*, vol. 23, no. 2, pp. 229–236, 1991.

[114] G. Aston and L. Burnard, *The BNC handbook. Exploring the British National Corpus with SARA*. Edinburgh University Press, 1998.

[115] F. Can, S. Kocberber, O. Baglioglu, S. Kardas, H. Ocalan, and E. Uyar, "New event detection and topic tracking in turkish," *Journal of the American Society for Information Science and Technology*, vol. 61, no. 4, pp. 802–819, 2010.

[116] D. R. Radev, H. Jing, M. Sty, and D. Tam, "Centroid-based summarization of multiple documents.," *Information Processing and Management*, vol. 40, no. 6, pp. 919–938, 2004.

[117] M. G. Ozsoy, F. N. Alpaslan, and I. Cicekli, "Text summarization using Latent Semantic Analysis.," *J. Information Science*, vol. 37, no. 4, pp. 405–417, 2011.

[118] Y. Gong and X. Liu, "Generic Text Summarization Using Relevance Measure and Latent Semantic Analysis.," in *SIGIR* (W. B. Croft, D. J. Harper, D. H. Kraft, and J. Zobel, eds.), pp. 19–25, ACM, 2001.

[119] J. Steinberger and K. Jezek, "Text Summarization and Singular Value Decomposition.," in *ADVIS* (T. M. Yakhno, ed.), vol. 3261 of *Lecture Notes in Computer Science*, pp. 245–254, Springer, 2004.

[120] G. Murray, S. Renals, and J. Carletta, "Extractive summarization of meeting recordings.," in *INTERSPEECH*, pp. 593–596, ISCA, 2005.

[121] M. Baroni and R. Zamparelli, "Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 1183–1193, Association for Computational Linguistics, 2010.

[122] J. Mitchell and M. Lapata, "Vector-based models of semantic composition," *proceedings of ACL-08: HLT*, pp. 236–244, 2008.

[123] S. Clark and S. Pulman, "Combining symbolic and distributional models of meaning," in *Proceedings of the AAAI Spring Symposium on Quantum Interaction*, pp. 52–55, 2007.

[124] D. Widdows, "Semantic vector products: Some initial investigations," in *To appear in Second AAAI Symposium on Quantum Interaction*, vol. 26, p. 28th, Citeseer, 2008.

[125] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning, "Domain-Specific Keyphrase Extraction," in *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, (San Francisco), pp. 668–673, Morgan Kaufmann Publishers Inc., 1999.

[126] K. Toutanova, D. Klein, C. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pp. 173–180, Association for Computational Linguistics, 2003.

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

[128] V. Vinay, I. J. Cox, N. Milic-Frayling, and K. R. Wood, "On ranking the effectiveness of searches.," in *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)* (E. N. Efthimiadis, S. T. Dumais, D. Hawking, and K. Javelin, eds.), (New York), pp. 398–404, ACM, 2006.

[129] B. He and I. Ounis, "Query performance prediction," *Information Systems*, vol. 31, no. 7, pp. 585–594, 2006.

[130] Y. Zhou and W. B. Croft, "Ranking robustness: a novel framework to predict query performance.," in *Proceedings of the 15th ACM International Conference on Information and Knowledge Management (CIKM)* (P. S. Yu, V. J. Tsotras, E. A. Fox, and B. Liu, eds.), (New York), pp. 567–574, ACM, 2006.

[131] K.-L. Kwok, L. Grunfeld, N. Dinstl, and P. Deng, "TREC 2005 Robust Track Experiments Using PIRCS," in *Proceedings of the 2005 Text Retrieval Conferance (TREC)* (E. M. Voorhees and L. P. Buckland, eds.), vol. Special Publication 500-266, NIST, 2005.

[132] E.-H. Han and G. Karypis, "Centroid-Based Document Classification: Analysis and Experimental Results.," in *Principles and Pratice of Knowledge Discovery in Databases (PPKDD)* (D. A. Zighed, H. J. Komorowski, and J. M. Zytkow, eds.), vol. 1910 of *Lecture Notes in Computer Science*, (Berlin), pp. 424–431, Springer, 2000.

[133] B. Hopkins and J. Skellam, "A new method for determining the type of distribution of plant individuals," *Annals of Botany*, vol. 18, no. 2, p. 213, 1954.

[134] T. Cox and T. Lewis, "A conditioned distance ratio method for analyzing spatial patterns," *Biometrika*, vol. 63, no. 3, p. 483, 1976.

[135] J. M. Ponte and W. B. Croft, "A language modeling approach to information retrieval," in *Proceedings of the 21st Annual international ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, (New York), pp. 275–281, ACM, 1998.

[136] V. Lavrenko, J. Allan, E. DeGuzman, D. LaFlamme, V. Pollard, and S. Thomas, "Relevance models for topic detection and tracking," in *Proceedings of the Second International Conference on Human Language Technology Research (HLT)*, (San Francisco), pp. 115–121, Morgan Kaufmann Publishers Inc., 2002.

[137] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[138] B. Bigi, "Using Kullback-Leibler Distance for Text Categorization.," in *Proceedings of European Conference on Information Retrieval (ECIR)* (F. Sebastiani, ed.), vol. 2633 of *Lecture Notes in Computer Science*, (Berlin), pp. 305–319, Springer, 2003.

[139] D. Pinto, J.-M. Benedi, and P. Rosso, "Clustering Narrow-Domain Short Texts by Using the Kullback-Leibler Distance.," in *Proceedings of the Conference on Computational Linguistics and Intelligent Text Processing (CICLING)* (A. F. Gelbukh, ed.), vol. 4394 of *Lecture Notes in Computer Science*, (Berlin), pp. 611–622, Springer, 2007.

[140] U. M. Fayyad and K. B. Irani, "Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning," in *International Joint Conferences on Artificial Intelligence (IJCAI)*, (San Francisco), pp. 1022–1029, Morgan Kaufmann Publishers Inc., 1993.

[141] S. Kim, O. Medelyan, M. Kan, and T. Baldwin, "Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles," in *Proceedings of the 5th International Workshop on Semantic Evaluation*, (Morristown), pp. 21–26, Association for Computational Linguistics, ACL, 2010.

[142] X. Zhao, J. Jiang, J. He, Y. Song, P. Achananuparp, E. LIM, and X. Li, "Topical keyphrase extraction from twitter," *Research Collection School of Information Systems*, 2011.