APPLICATION OF VECTOR SPACE MODELS TO DETECT SEMANTICALLY NON-COMPOSITIONAL WORD COMBINATIONS IN TURKISH

LEVENT TOLGA EREN

AUGUST 2016

APPLICATION OF VECTOR SPACE MODELS TO DETECT SEMANTICALLY NON-COMPOSITIONAL WORD COMBINATIONS IN TURKISH

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF

NATURAL AND APPLIED SCIENCES OF

IZMIR UNIVERSITY OF ECONOMICS

BY

LEVENT TOLGA EREN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

IN THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

AUGUST 2016

M.S. THESIS EXAMINATION RESULT FORM

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Osmihan Bayramoğlu
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Cem Evrendilek
Head of Department

We have read the thesis entitled **"Application of Vector Space Models to Detect Semantically Non-compositional Word Combinations in Turkish"** completed by **LEVENT TOLGA EREN** under supervision of **Asst. Prof. Dr. Senem KUMOVA METİN** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Senem KUMOVA METİN
Supervisor

Examining Committee Members Date: 24.08.2016

Asst. Prof. Dr. Senem KUMOVA METİN
Dept. of Software Engineering, IUE

Asst. Prof. Dr. Kaan KURTEL
Dept. of Software Engineering, IUE

Asst. Prof. Dr. Tarık KIŞLA
Dept. of Computer Education and Instructional Technologies, Ege U.

ABSTRACT

APPLICATION OF VECTOR SPACE MODELS TO DETECT SEMANTICALLY NON-COMPOSITIONAL WORD COMBINATIONS IN TURKISH

LEVENT TOLGA EREN
M.S. in Department of Natural and Applied Sciences
Graduate School of Natural and Applied Sciences
Supervisor: Asst. Prof. Dr. Senem KUMOVA METİN
August 2016

The semantic compositionality defines the relation between the meanings of word combinations and their components. In non-compositional expressions, the words combine to generate a different meaning. The identification of non-compositional expressions may support several natural language processing tasks such as machine translation, word sense disambiguation and language generation. The objective of the thesis is exploring the performance of vector space models in detection of non-compositional expressions in Turkish.

In this thesis, a data set of 2229 two-word combinations that is built from six different Turkish corpora is utilized. Three sets of five different vector space models are employed in the experiments. The evaluation of models is performed using three metrics: precision, recall and F-measure. The experimental results show that the model that measures the similarity between the vectors of word combination and the second composing word produced higher average F-scores for all testing corpora.

Keywords: semantic compositionality, vector space model, natural language processing.

ÖZ

TÜRKÇEDE ANLAMSAL BİRLEŞİMİ OLMAYAN KELİME GRUPLARININ TESPİTİNDE VEKTÖR UZAY MODELLERİNİN UYGULANMASI

LEVENT TOLGA EREN
Fen Bilimleri Enstitüsü, Yüksek Lisans
Fen Bilimleri Enstitüsü
Tez Danışmanı: Asst. Prof. Dr. Senem KUMOVA METİN
Ağustos 2016

Anlamsal birleşimlilik, kelime kombinasyonları ve bunların parçalarının anlamları arasındaki ilişkiyi tanımlamaktadır. Anlamsal birleşimli olmayan ifadelerde kelimeler bir araya gelerek farklı anlamlar meydana getirmektedir. Anlamsal birleşimli olmayan ifadelerin tanımlanması makine çevirisi, kelime anlamını belirginleştirme ve dil üretme gibi birçok dil işleme görevlerini destekleyebilmektedir. Bu tez çalışmasının amacı, Türkçe'de anlamsal birleşimli olmayan ifadelerin tespitinde uzay vektör modellerinin performanslarını araştırmaktır.

Bu tezde altı farklı Türkçe derlemden elde edilen 2229 adet ikili kelime kombinasyonu içeren bir veri kümesi kullanılmıştır. Yapılan deneylerde beş farklı vektör uzay modeli içeren üç küme kullanılmıştır. Bu modeller duyarlılık, anma, ve F-ölçümü ölçütleriyle değerlendirilmiştir. Deneylerde tüm test derlemleri için kelime kombinasyonu ve kombinasyonu oluşturan ikinci kelimeye ait vektörler arası benzerliği ölçen modelin daha yüksek F değerleri ürettiği görülmüştür.

Anahtar Kelimeler: anlamsal birleşimlilik, vektör uzay modeli, doğal dil işleme.

ACKNOWLEDGEMENT

I would like to take the opportunity to thank some people who supported me during the process of this work.

First of all, I would like to thank to my supervisor Asst. Prof. Dr. Senem KUMOVA METİN. I am so lucky to have a chance to work with such a wise person. She helped me all the time even I was having troubles.

Secondly, I would like to thank to my colleges Serhat UZUNBAYIR, Mehmet Berkehan AKÇAY, and Erdem OKUR. They were never stopped believing in me.

Lastly, I would like to thank to my father Bülent Tolga EREN, my mother Resmiye EREN, and my sister Güniz Göze EREN for their trust and endless love.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Humans are gifted to handle and process the speech in natural language. We can easily understand the meaning of each component in the speech and predict its meaning. But how about machines ? Can they truly mimic humans about this ? Today this is a formidable task for machines while humans are sufficient in understanding multiword units like phrases. Currently, in semantic field technologies like search engines hold importance on human life. But these technologies are approaching human language at word level. Existing systems does not possess higher level semantic technologies like phrases. We can give some potential applications such as question answering, intelligent search engines, bio-medical applications. Vocabulary of a language is limited but its generative ability for combinatorial expressions is not. Thus word level methods fail to model phrasal semantics no matter how many word we use. Therefore a good model which aims to capture language should be generative. Compositional semantics involves here to take word level research to phrasal semantics. In the following sections, we are going to explore semantic compositionality, then we will give details about non-compositionality and multiword expressions. Finally we will state objective of the thesis and our contributions.

1.1 Semantic Compositionality

In principle of compositionality, Pelletier [22] states that definition of an expression[1] is a function of the meaning of its components (words) and the way in which the parts are combined. Baldwin [7] describes compositionality as the degree to which the features of the parts of an expression combine to predict the features of the whole. Also he states that though the compositionality is generally considered in context of semantic compositionality, it is possible equally to talk about lexical, syntactic and pragmatic compositionality.

In this thesis, the notion of compositionality is limited to semantic compositionality of expressions that is composed of two consecutive words defined as *bigram*. In this perspective, compositionality is the degree of relation between the meaning of expression and the individual meanings of its constituents. In compositional expressions, the meaning of expression can be predicted from the meanings of its constituents. For example, the two-word expression *trafik ışığı* is a compositional expression (to some degree). The term *trafik ışığı* corresponds to signalling devices positioned at road intersections, pedestrian crossings etc. to control the flow of traffic. A person who knows the meanings of the words *trafik* and *ışığı* may guess that the term points to an object that includes *ışık* and is related somehow to *trafik*. In non-compositional expressions the combined meaning of words is unrelated to individual meanings of its components. For instance, the two-word expression *kanı bozuk* is a fully non-compositional expression. Even if a person is a native speaker of Turkish, he may not predict the meaning of *kanı bozuk* by the meanings of *kanı* and *bozuk*.

[1] Expression is a group of words.

1.2 Non-compositionality and Multiword Expressions

The concept of compositionality/non-compositionality is closely related to the notion of *multiword expressions*. Multiword expressions are defined to be groups of words that inclined to co-occur more frequently than by chance and they are either idiosyncratic or decomposable into multiple words [7]. Multiword expressions appears often in the natural languages. Identifying multiword expressions in random text clusters is a formidable problem. The identification of multiword expressions in text is important for a variety of areas in computer science such as information retrieval, machine translation, language generation, question answering, part of speech tagging and parsing. For example, if the expression *ağzından baklayı çıkarttı* is not considered as a single unit of meaning in the sentence *Ayşe sonunda ağzından baklayı çıkarttı* in machine translation, the sentence is translated to *Ayşe finally remove the beans from her mouth* erroneously. The correct translation is *Ayşe finally spilled the beans*.

Extraction of multiword expressions is a challenging task since there are no known rules that formulates the construction of all type of multiword expressions. Therefore, we will address the notion by spotting some characteristics of multiword expressions that are given in the study of Baldwin [7]. These notions include semantic or pragmatic idiomaticity, lexico-syntactic idiomaticity, situatedness, institutionalisation and translatability [7].

Semantic or pragmatic idiomaticity concerns multiword expressions whose contents greatly differ from the semantics or pragmatics of its components appearing distinctly. Constituents of non-compositional multiword expressions inclined to co-occur with some specific words within a wide set of synonyms. These preferences are called selectional preferences. For example, *sert kahve* is a multiword expression though *katı kahve* that includes the synonym of the first word in the example is not a multiword expression.

Situated multiword expressions are related with a constant pragmatic point.

These expressions used in specific circumstances like a period, a place or by people that have a special characteristic. For example, *iyi şanslar* is a multiword expression which is used in certain circumstances.

Institutionalised multiword expressions (e.g. *tuzu biberi*) are recognized as lexical terms, through continuous use over time. These multiword expressions are semantically and syntactically compositional, but statistically idiosyncratic.

Multiword expressions are not usually word to word translatable into another languages. For example, *etekleri zil çalmak* is translated to English *skirts play bell* by word to word translation. Though the correct translation meaning should give someone is getting overly exited.

Aforementioned multiword expression features indicate that in a majority of the multiword expressions the meaning of the multiword expression is not only directly connected to the individual meanings of the constituents. For example:

1. When a word in a multiword expression is exchanged with its synonym, the new combination is not a multiword expression. For example, *sert kahve* $\rightarrow$ *katı kahve*.
2. If the order of the words change in a multiword expression, the new combination is not a multiword expression (e.g. *ver elini* $\rightarrow$ *elini ver*)
3. When a multiword expression is used in a different context (situation) the expression is not a multiword expression. (e.g. *Adamı ayağının tozuyla kodese tıktılar.* $\rightarrow$ *Eve ayağının tozuyla girme, halılar kirleniyor.*)

The multiword expressions with characteristics that force the properties such as the use of same word (not even synonym), same word order, same periods (similar situation) are accepted as examples of non-compositional expressions.

We accepted that a majority of multiword expressions (idioms, technical terms, named entities, some phrasal verbs) are non-compositional expressions as in the study of Bu et al. [9] and Choueka [12].

1.3 Objective of Thesis

The objective of the thesis is exploring the performance of vector space models in detection of non-compositional expressions in Turkish. Inline with this objective, a dataset of 4800 bigrams are extracted from 6 different Turkish corpora by the use of occurrence frequency methods (chi square, occurrence frequency counts, pointwise mutual information and t-test). This dataset is annotated by 4 different human judges. The dataset is utilized in the experiments of vector space models that are previously proposed to measure the semantic compositionality/non-compositionality in different languages. The contribution of the thesis that vector space models in detection of semantic compositionality in Turkish is firstly studied. In Figure 1.1, the flowchart gives a overview to the general structure of the thesis.

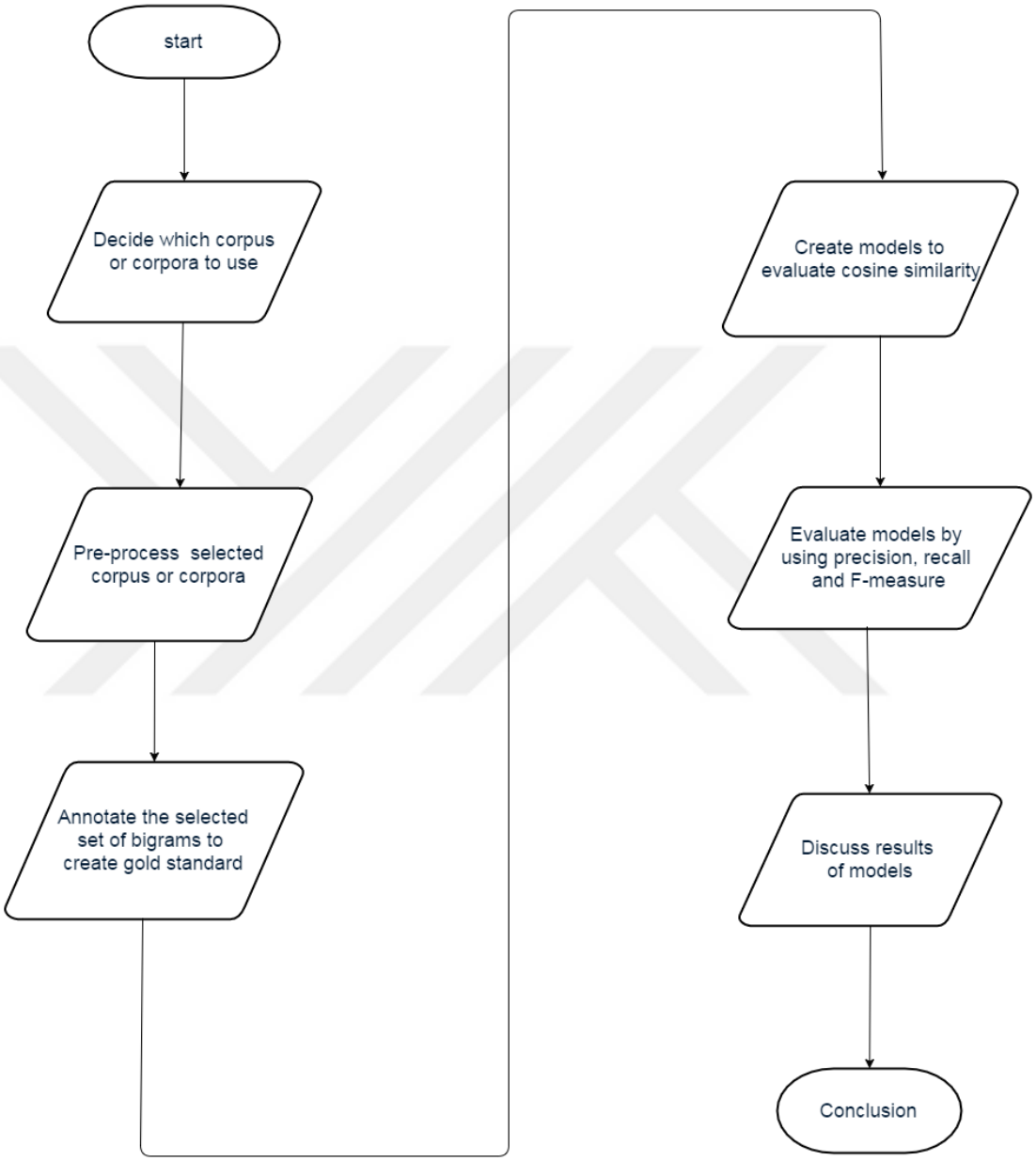

Figure 1.1: Overview of the thesis.

Chapter 2

Background Research and Related Work

The chapter covers the definition of vector space models and major works on measuring the non-compositionality in language.

2.1 Vector Space Models

In distributional hypothesis, Rubenstein & Goodenough [26] states that words with similar meanings will occur with similar neighbours if enough text material is available. Also Firth [15] states that you shall know a word by the company it keeps. Distributional hypothesis is also called as distributional semantics.

In distributional semantics, a word is expressed by its neighbouring words targeted with their occurrence frequency and the context of a target word is defined as its neighbouring words in a fixed window size. One can tell for a given two words are similar if they have a similar distribution of contexts. For example *ev* and *apartman* generally occur with context words like *kira*, *balkon*, *satılık* etc., gives a pattern to computational models that *ev* and *apartman* may be similar.

	ingilizce	yöre	kitap	büyük	dernek
yabancı	9	3	8	2	1
dil	7	4	7	3	1
yabancı dil	6	1	6	1	0

Table 2.1: Co-occurence vectors of *yabancı dil* and its components.

In distributional semantics, Vector Space Models (VSM) have become a conventional structure for describing meaning of words [29]. It is accepted that the words that tend to co-occur frequently with the targeting word/word combination constitute the meaning of the target. In VSM, each word/word combination may be expressed as a multi dimensional context vector where each dimension stores the co-occurrence frequency of a neighbouring word. The neighbouring words are the words that co-occur with the target word/word combination in a predefined windows size. The window size may vary and is commonly limited to the sentence or the text length. In Table 2.1, vectors for the targets *yabancı*, *dil* and *yabancı dil* are given as an example. In Table 2.1, the target word *yabancı* is co-occurring with the words *ingilizce* for 9 times, *yöre* for three times. In this example, *ingilizce* and *kitap* are given as good representatives, *yöre* and *dernek* are given as disturbing (bad) representatives for the *yabancı dil* concept.

2.2 Related Work

In recent years, there has been a growing awareness in the NLP field about problems related to compositionality. Several special interest workshops have been arranged and discussed issues like automatically acquiring semantic compositionality [8]. In Table 2.2, proposed methods in Distributional Semantics and Compositionality 2011 (*DiSCo 2011*) are summarized.

Any NLP system that does semantic processing relies on the assumption of semantic compositionality: the meaning of a phrase is determined by the meanings of its parts and their combination. However, this assumption does not hold for lexicalized phrases such as idiomatic expressions. In particular, while distributional methods in semantics have proved to be very efficient in tackling a wide

Applied Methods	Institution	Team	Approach
Duluth-1 Duluth-2 Duluth-3	Dept. of Computer Science, University of Minnesota	Ted Pedersen	statistical association measures: t-score and pmi
JUCSE-1 JUCSE-2 JUCSE-3	Jadavpur University	Tanmoy Chakraborty, Santanu Pal Tapabrata Mondal, Tanik Saikh, Sivaju Bandyopadhyay	mix of statistical association measures
SCSS-TCD:conf1 SCSS-TCD:conf2 SCSS-TCD:conf3	SCSS, Trinity College Dublin	Alfredo Maldonado-Guerra, Martin Emms	unsupervised WSM, cosine similarity
Cosine-Add/Mult/Alm	Center for Mind/Brain Sciences, University of Trento	Eva Maria Vecchi, Marco Baroni, Roberto Zamparelli	cosine similarity
UCPH-simple.en	University of Copenhagen	Anders Johannsen, Hector Martinez, Christian Rishøj, Anders Søgaard	support vector regression with COALS-based endocentricity features
UoY: Exm UoY: Exm-Best UoY: Pro-Best	University of York, UK; Lexical Computing Ltd., UK	Siva Reddy, Diana McCarthy, Suresh Manandhar, Spandana Gella	exemplar-based WSM prototype-based WSM
UNED-1: NN UNED-2: NN UNED-3: NN	NLP and IR Group at UNED	Guillermo Garrido, Anselmo Peñas	syntactic VSM, dependency-parsed UKWaC, SVM classifier
DTK DDTK	DISP University of Rome	Fabio Massimo Zanzotto, Lorenzo DellArciprete	distributed tree vector distributed kernel tree vector
MMI	University of Cambridge	Tim Van de Cruys	multi-way co-occurences

Table 2.2: Distributional Semantics and Compositionality Shared Task (DiSCo) 2011 participants with their applied methods and approaches.

range of tasks in natural language processing, e.g., document retrieval, clustering and classification, question answering, query expansion, word similarity, synonym extraction, relation extraction, textual advertisement matching in search engines, etc., they are still strongly limited by being inherently word-based. While dictionaries and other lexical resources contain multiword entries, these are expensive to obtain, not available for all languages to a sufficient extent, the definition of a multiword varies across resources and non-compositional phrases are merely a subclass of multiwords. The workshop brings together researchers that are interested in extracting non-compositional phrases from large corpora by applying distributional models that assign a graded compositionality score to a phrase as well as researchers interested in expressing compositional meaning with such models. This score denotes the extent to which the compositionality assumption holds for a given expression. The latter can be used, for example, to decide whether the phrase should be treated as a single unit in applications. Approaches that employ prefabricated lists of non-compositional phrases should consider a different venue.

Biemann & Giesbrecht [8] developed a compositionality dataset using various human judges. The dataset includes 133 V-OBJ, 74 V-SUBJ, 144 ADJ-NN

expressions identified with compositionality score. Each the set is annotated by judges in range 0-100.

The majority of studies on non-compositionality are presented in DiSCo 2011 is English and German language. The data is separated into three classes based on the compositionality score. Low in compositional ($0<score<37$), medium in compositional ($36<score<75$) and high in compositional ($74<score$). These expressions are labelled with related class labels, also called as coarse labels. There are 96 V-OBJ, 56 V-SUBJ 102 ADJ-NN expressions which have scores in the specified score range. All other expressions are not classified and are not involved in evaluation for coarse-grained labels. The final dataset is split into 50% test, 10% validation and 40% training datasets.

The study of Vecchi et al. [31] introduced an approach to characterize the semantic aberrance of complex expressions. In their work, they have used vector based semantic space to look properties of adjective-noun complex expressions. To do that, they have come up with several models to show compositionality levels of adjective-noun expressions. Multiplicative models, additive models and linear-map based models are used by them. From the targeted corpus, they have generated composite vectors for a set of adjective-noun expressions. This set contains either semantically acceptable adjective-noun expressions or not. Then they have tested this set with stated models. Multiplicative and additive models gave remarkable results compared to other models.

The study Zanzotto & DellArciprete [32] investigated distributed representation theories that is branching into distributed meaning and structure. They constructed an absolute distributed tree and a distributional distributed tree. Constructed trees are used for manipulating tree kernels by using recognition of textual entailment. Their results show that constructed distributional distributed tree kernels correlate with distributed tree kernels and performed better than distributional distributed tree kernels in recognition of textual entailment. Harder part is including distributional vectors in distributed structure.

In Cruys [30] work, he explored nature and usefulness of point wise mutual

information in extraction of subject-verb-object triplets. Pointwise mutual information in normal state is restricted with only two way co-occurrences. In his work, pointwise mutual information explored as two multivariate generalizations.

The study of Johannsen et al. [17] introduced a COALS-based endocentricity score method. In their research, they focused on compositionality prediction for word pairs, compositionality scores based on distributional clusters, hyphenation, statistics about wordnet-induced paraphrases and the likelihood of long translation equivalents in other languages. Their work greatly correlated with human compositionality scores and support vector regression experiments.

The study of Pedersen [21] introduced three systems that evaluated distributional methods of measuring semantic compositionality. These systems addressed semantic compositionality as a problem of collocation identification, where strong collocates are assumed to be minimally compositional.

According to Chakraborty et al. [11] the measurement of relevant compositionality of bigrams is important to identify multiword expressions in Natural Language Processing (NLP) tasks. The paper performs the experiments provided as part of the participation in the shared task. The experiments based on different collocation-based statistical approaches to measure the relative compositionality of three models of bigram phrases (Adjective-Noun, Verb-subject and Verb-object combinations). The experimental results in terms of both coarse-grained and fine-grained compositionality scores have been assessed with the human annotated gold standard data. Fair results have been obtained in terms of average point difference and coarse precision.

The aim of work in the study of Garrido & Peña [16] is to predict compositionality judgements indicated by human judges to candidate phrases, in English and German, from three general grammatical relations: adjective-noun, subject-verb and subject-object. Garrido & Peñas [16] explored the use of syntactic-based contexts collected from large corpora to develop classifiers that model the compositionality of the semantics of such pairs.

Maldonado-Guerra & Emms [18] developed a system for measuring the compositionality of collocations within the structure of the shared task of the Distributional Semantics and Compositionality workshop is presented. The system utilizes the intuition that a highly compositional collocation would tend to have a significant semantic overlap with its components whereas a collocation with low compositionality would share little semantic content with its constituents. This intuition is formed via three configurations that exploit cosine similarity measures to identify the semantic overlap between the collocation and its constituents.

In the study of Reddy et al. [25] difficulties of polysemy in word space models of compositionality detection is pointed out. Most models express each word as a single prototype-based vector without addressing polysemy. They prepared an exemplar-based model which is designed to manage polysemy. This model is tested for compositionality detection and it is seen to outperform existing prototype-based models.

Chapter 3

Measuring non-compositionality in Turkish

This chapter involves the preparation of dataset and the methods applied in this thesis. Section 3.1 gives the definitions of data sources. Section 3.2 presents the tasks performed to build dataset that is used in thesis.

3.1 Datasource

The Turkish corpus *BilCol* [10], *Bilkent* [28], *Ege*[1], *Leipzig* [23], *METU* [27], and *Muder* [14] are used in this thesis to construct the data set. Their contents are explained in the following subsections.

3.1.1 BilCol

The corpus is built in Bilkent University using the following sources through the year 2005 [10].

[1]This corpus is collected in Ege University, International Computer Institute in order to be used in natural language processing studies.

1. CNN Türk [1],
2. Haber 7 [2],
3. Milliyet Gazetesi [3],
4. TRT [5],
5. Zaman Gazetesi [6].

The corpus contains labels for days, hours, minutes of every news. Figure 3.1 gives the different portions in corpus [10].

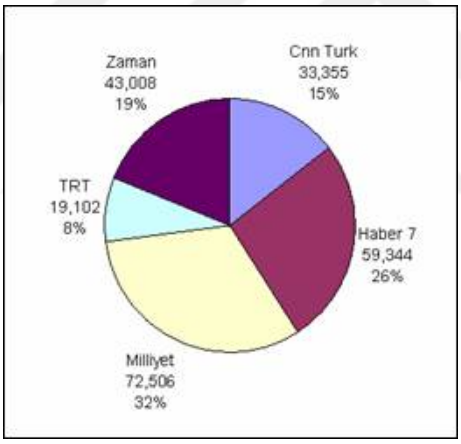

Figure 3.1: Distribution of news between sources.

3.1.2 Bilkent

Bilkent corpus [28] is compiled in Bilkent University to be used in computational linguistics research. The corpus is automatically annotated by a finite state machine. The corpus is morphologically analysed by a finite state machine [28], [19].

3.1.3 Ege

Ege corpus is constructed in Ege University in International Computing Institute for Natural Language Processing (NLP) studies. The corpus includes 875 texts that are classified in 9 major topics (e.g science, religion etc.).

3.1.4 Leipzig

The Leipzig Corpora Collection (LCC) is a collection of corpora of similar sources and equivalent processing for more than 250 languages. According to their sources, the corpora are classified in three dimensions:

1. Language (sometimes in connection with the country of origin)
2. Genre (currently: news texts, random web texts, and Wikipedia texts)
3. Time: year of download

Turkish newspaper corpus is built based on material of the year 2005 [23].

3.1.5 METU

METU Turkish Corpus [27] is a compilation of 2 million words of post-1990 written Turkish samples. METU Turkish Corpus is XCES tagged at the typographical level. The words of METU Turkish Corpus were taken from 10 various genres. At most 2 samples from one source are used; each sample is 2000 words or the sample ends when the next sentence ends [27].

3.1.6 Muder

This corpus is built in Muğla Sıtkı Koçman University [14]. It contains approximately over 40000 sentences and 670000 tokens.

3.2 Dataset Preparation

Linguistic approaches for term recognition and collocation extraction include plenty of linguistic processing components. Many of these components correspond to fundamental Natural Language Processing tasks which were studied in the past and were solved adequately. They aim to eliminate textual noise and in general, modify the input raw text so as ease extraction of required information from text.

Preprocessing methods vary from very simple ones, such as utilizing all characters of the text into lower-case, to complicated ones, such as resolving abbreviations and syntax normalization.

In this thesis, following tasks are performed in data set construction:

1. Removal of punctuation
2. Tokenization
3. Application of occurrence frequency methods (chi-square, occurrence frequency counts, pointwise mutual information and the t-test)
4. Annotation of set

Below subsections give the details on these tasks. Moreover, in Figure 3.2 the flowchart diagram for the dataset preparation is shown, its purpose is to give a better understanding of the structure.

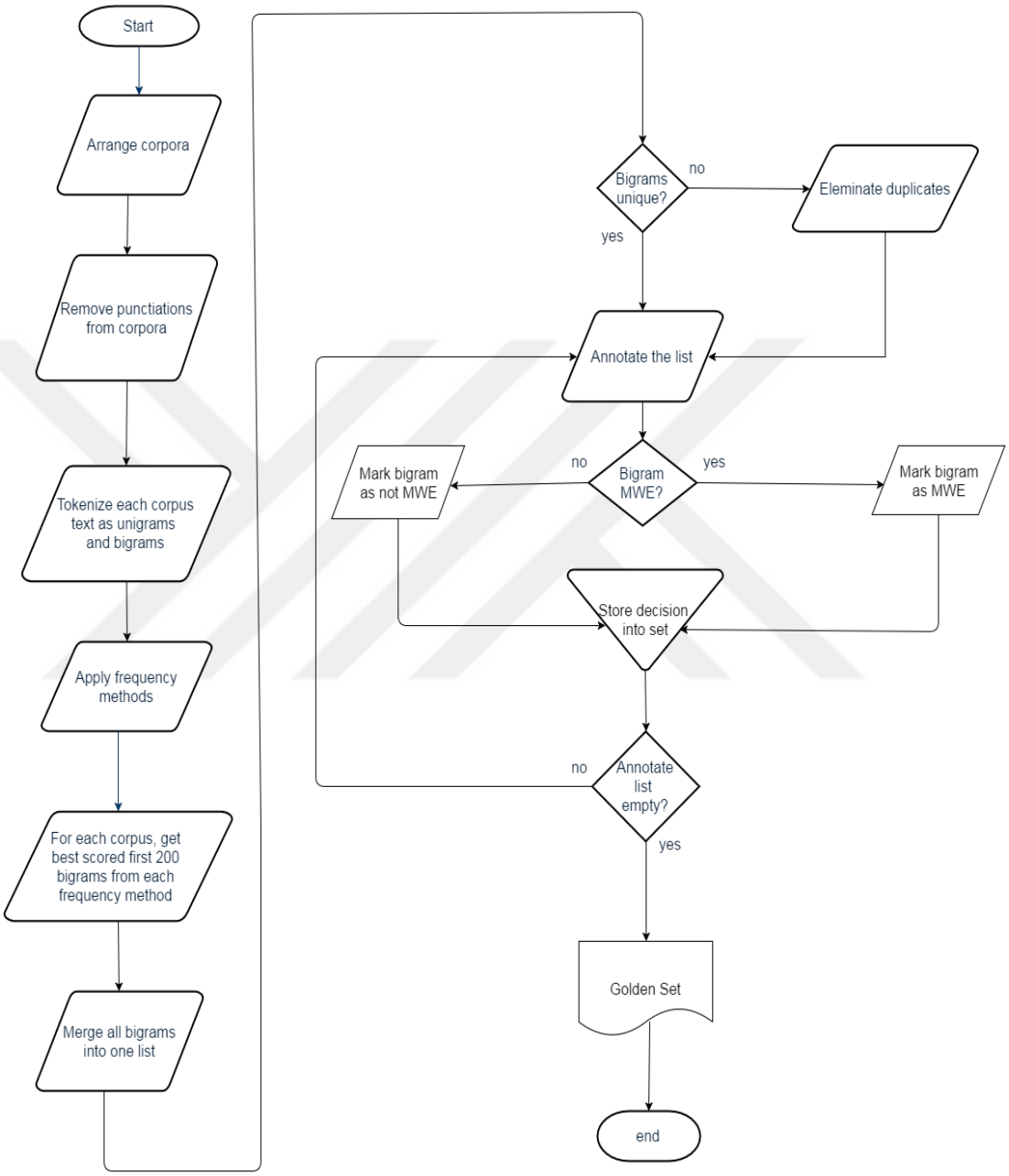

Figure 3.2: How pre-processing and annotation of set work.

3.2.1 Removal of punctuation

The corpora that are used in natural language process studies may contain irrelevant data such as non-alphanumeric tokens and Extensible Markup Language

(XML) tags. For example, BilCol corpus contains XML tags such as in Listing 3.1. Such irrelevant data not only bias the amount of data in corpus but also disturb the performance of natural language processing methods. In Table 3.1, we presented raw data counts of each corpora in our study. To get rid of these irrelevant data, we cleared up all XML tags and non-alphanumeric tokens.

Corpora	**# Word Counts**	**# Char Counts**
BilCol	44,150,213	347,734,602
Bilkent	767,132	5,111,377
Ege	2,449,664	17,365,833
Leipzig	14,279,547	110,628,416
METU	1,984,634	15,222,700
Muder	679,750	5,391,177

Table 3.1: Corpora statistics before removal of punctuation.

```
<DOC>
<DOCID> 0 </DOCID>
<SOURCE> Haber7 </SOURCE>
<DATE> 2005-01-01 00:00:00 </DATE>
<TITLE> Maliye gece denetiminde </TITLE>
<TEXT>
Vatan Caddesi'ndeki maliye kompleksinden saat 20:00 sıralarında
ayrılan, İstanbul Defterdarlığı Vergi Denetmenleri Bürosu Başkanı
Ali Baş idaresindeki 800 kişilik denetleme ekibi, 70 araçla,
gruplar halinde önceden berlirlenen bölgelere dağıldı. AA
</TEXT>
</DOC>
```

Listing 3.1: An example sentence of BilCol corpus.

Corpora	# Word Counts	# Character Counts
BilCol	42,414,743	320,071,109
Bilkent	706,443	5,358,042
Ege	2,465,285	18,353,348
Leipzig	13,389,049	101,313,193
METU	1,987,447	14,715,263
Muder	638,547	4,909,231

Table 3.2: Corpora statistics after removal of punctuation.

Table 3.2 shows statistics of corpora data after cleaning up from punctuation.

3.2.2 Tokenization

Tokenization is the one of first steps of preprocessing and corpus preparation. Tokenizers are components that get text as input and separate the sentences and words in text. Generally, after tokenization individual words are called tokens or unigrams. Tokenizers output a list of tokens for each input sentence.

In this study, before tokenizing corpora, we have changed all upper-case letters to lower-case letters. For example, the words *Kare* and *kare* treated as two different tokens by machine in a case sensitive situation. To fix that issue, one must first change all words to lower-case form.

Tokens are first sorted in alphabetical order and then each token's unique count (occurrence frequency) is calculated (e.g. *parasal* token have been observed in BilCol corpus 520 times). Total unique token counts in other words unigram counts can be found in Table 3.3.

After unique tokens are found, we produced bigrams from them. This process follows the pattern: Each unique token is tailed with its following token to create bigrams. For instance, *word1 word2 word3* are three distinctive words that creates *word1 word2* and *word2 word3* unique bigrams. Total unique bigram counts can be found in Table 3.4.

Corpora	# Unigrams
BilCol	984,434
Bilkent	94,552
Ege	259,196
Leipzig	745,446
METU	212,853
Muder	82,145

Table 3.3: Total unique unigram counts of corpora.

Corpora	# Bigrams
BilCol	11,759,532
Bilkent	507,758
Ege	1,637,055
Leipzig	7,350,443
METU	1,388,722
Muder	437,826

Table 3.4: Total unique bigram counts of corpora.

In the following subsection we explain the application of occurrence frequency methods for extraction of candidates.

3.2.3 Application of occurrence frequency methods

Statistical approaches for identifying and handling multiword expressions employ frequency counts of words, N-grams, co-occurrences of words, etc. Statistical approaches handle the frequency counts and context distributions in numerous distinct ways and output decisions on multiword expressions or output scores that quantify useful characteristics of multiword expressions in terms of compositionality.

In the following subsections, we present the statistical approaches that we employed in this study. These statistical approaches refer to the degree of strength of association between words. Following methods try to detect whether the components of a candidate term form a collocation rather than co-occurring by only just chance.

Frequency	w_1	w_2
560	ya	da
321	diye	konuştu
308	böyle	bir
299	yeni	bir
285	bu	arada
275	bu	konuda
269	büyük	bir
255	en	büyük
251	insan	hakları
242	bir	şey

Table 3.5: First 10 most frequent bigrams in Bilkent corpus.

3.2.3.1 Occurrence frequency counts

The easiest method of finding multiword expressions (collocations) in a corpus is counting the number of occurrences of word combinations in the text. If a word combination is occurring frequently in a corpus, it is assumed to be a multiword expression.

In frequency based models, frequencies are actually the co-occurrence counts of words or word combination. Co-occurrence counts are calculated to estimate how strong is the relationship between two words in a given word combination. For example, assume that *gömlek* occurs more frequently than *üniforma* in some corpora. But, if *polis* co-occurs with *üniforma* in numerous sentences than it does with *gömlek*, then one can decide that the relationship between *polis* and *üniforma* is stronger.

In this study, we have extracted both unigram and bigram (two words that occur consequently) occurrence frequencies 200 bigrams that have the highest occurrence counts. Then, for each corpus we have selected

From Bilkent corpus, the most frequently observed first 10 bigrams can be found in Table 3.5.

PMI	$\mathbf{f(w_1)}$	$\mathbf{f(w_2)}$	$\mathbf{f(w_1w_2)}$	$\mathbf{w_1}$	$\mathbf{w_2}$
16.1145	9	6	6	gelmediğini	ölçecek
15.7401	10	7	6	gözlenmesi	kararlaştırılan
15.6995	9	8	6	mehtap	projesinde
15.6995	12	6	6	hükmünde	kararnamenin
15.6995	12	6	6	cezaya	başvurmalıdır
15.6026	7	11	6	tüme	varım
15.5475	10	8	6	ayşe	bacı
15.2844	16	6	6	düzenlenip	düzenlenmediğini
15.2844	16	6	6	dergisinin	kapağında
15.2255	10	15	9	külçe	simli

Table 3.6: First 10 pointwise mutual information scores of Muder corpus.

3.2.3.2 Pointwise mutual information

Pointwise Mutual Information (PMI) is a measure derived from information theory and can be employed for term and collocation extraction [13]. PMI is measured as follows for tokens w_1 and w_2 :

$$PMI(w_1, w_2) = \log_2 \frac{P(w_1 w_2)}{P(w_1)P(w_2)} = \log_2 \frac{P(w_1|w_2)}{P(w_1)} = \log_2 \frac{P(w_2|w_1)}{P(w_2)} \tag{3.1}$$

In this study, we have calculated PMI values for bigrams that occur more than 5 times in corpora. Example PMI values that are obtained from Muder corpus can be found in Table 3.6. For each corpus we have selected 200 bigrams that produced highest PMI scores. These bigrams then listed for future use in our study (in total 200x6=1200 bigrams recorded for PMI).

3.2.3.3 Hypothesis testing

Hypothesis testing grants the statistical structure for analysing the frequency of occurrence of an event with the repetition of it by chance. In other words, many hypothesis testing methods evaluate whether or not something is a possible event.

The basic procedure is defined in following: Firstly the null hypothesis is set (H_0). Then the probability, p, of the event if H_0 is calculated and H_0 is rejected if p is too low (typically below a significance level of p <0.05, 0.01, 0.005 or 0.001).

For collocation multiword extraction the null hypothesis is the independence of constituents. It is described as the fact that there is no association between the words, beyond occurrences by chance. The hypothesis can be formulated for any number of words. Here, we concentrate on the bigram case. Let w_1 and w_2 be the constituent words of a collocation candidate. The independence hypothesis is:

$$\mathrm{P}(w_1 w_2) = \mathrm{P}(w_1)\ \mathrm{P}(w_2)$$

There are various hypothesis testing techniques. In this study, we have used the commonly used ones and explain their advantages and disadvantages.

3.2.3.3.1 Chi-square test In principle, chi-square test ($\tilde{\chi}^2$) checks observed values with the expected values for independence. If the variance between observed and expected frequencies is high, the null hypothesis of independence can be rejected. If O_{ij} and E_{ij} are the observed and expected values, related to the cell(i,j) of the table of frequencies, the quantity X^2 is defined as:

$$X^2 = \sum_{i,j} \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \tag{3.2}$$

In this study, we have calculated chi-square values from previously extracted bigrams. Besides, we have ignored the bigrams that have occurrence frequency less than 6. Then, for each corpus we have selected the first 200 bigrams from the list of bigrams that is sorted in decreasing order of chi-square value.

As an example, the most high scored first 10 bigrams of Bilkent corpus can be found in Table 3.7.

Chi-square	$\mathbf{f(w_1)}$	$\mathbf{f(w_2)}$	$\mathbf{f(w_1w_2)}$	$\mathbf{w_1}$	$\mathbf{w_2}$
176,599	6	6	6	sıcağı	sıcağına
176,597	7	7	7	irili	ufaklı
176,593	9	9	9	enine	boyuna
173,098	26	25	25	zülfü	livaneli
169,523	13	12	12	utku	çakırözer
168,181	11	10	10	bardağı	taşıran
166,206	8	9	8	yürütmeyi	durdurma
163,014	6	7	6	canla	başla
160,673	73	61	61	bordo	mavili

Table 3.7: First 10 chi-square scores of Bilkent corpus.

3.2.3.3.2 The t-test The t-test is a statistical test broadly used in collocation extraction. The t-test is also a function of the variation between observed and expected means, estimated by the variance. The test shows the probability of obtaining an example with the observed t-test value, considering that the example is drawn from a distribution with mean μ.

If $\overline{x}$ is the sample mean, s^2 is the sample variance, N is the sample size and μ is the mean of the distribution, t value is computed as in follows:

$$t = \frac{\bar{x} - \mu}{\sqrt{\frac{s^2}{N}}} \tag{3.3}$$

In this study, we have calculated the t-test values from previously extracted bigrams. Besides, we have ignored the bigrams that have occurrence frequencies less than 6. The best scoring 200 candidates are selected to construct the data set in the thesis.

As an example, best scoring 10 bigrams of the t-test values in BilCol corpus is given in Table 3.8.

The t-test	$f(w_1)$	$f(w_2)$	$f(w_1w_2)$	w_1	w_2
179.104	52,568	44,376	32,196	diye	konuştu
147.364	66,999	55,906	21,905	ifade	eden
139.528	23,109	245,587	19,754	ya	da
138.03	55,765	44,663	19,178	yaptığı	açaıklamada
127.065	111,539	88,728	16,642	daha	sonra
126.068	66,999	58,929	16,642	ifade	etti
125.893	23,111	20,443	16,642	daha	sonra
125.629	374,324	17,391	16,642	söz	konusu
123.242	15,941	91,213	16,642	bu	arada
122.323	230,555	77,112	16,642	bilgiye	göre

Table 3.8: First 10 the t-test scores of BilCol corpus.

Word-1	Word-2	Judge-1	Judge-2	Judge-3	Judge-4	Result
dolanım	hızı	0	0	0	0	0
cürüm	işlemek	1	1	1	1	1
canla	başla	1	1	1	1	1
savaşa	hayır	1	0	1	0	**0**
susurluk	davası	1	1	1	0	1

Table 3.9: Sample data from gold standard.

3.2.4 Annotation of set

Annotation set consists of sets that are previously obtained by application of 4 frequency methods. Each method yielded 1200 distinct bigrams from 6 different corpora. Then we collected these bigrams into one pool (which yielded 1200x4=4800 bigrams). Sorting and calculating unique count of these bigrams shrank the list to 2229 bigram candidates.

In annotation task, 4 human judges (native Turkish, MSc and Ph.D. students) are employed. Each judge decided to given candidate whether the given bigram is compositional or non-compositional. The judges are guided to annotate idiomatic expressions, named entities, technical terms, phrasal verbs and multi-worded conjunctions as non-compositional. Table 3.9 shows some sample data from the annotated *gold standard*. In gold standard 1194 bigrams are tagged as compositional and 1035 bigrams are non-compositional.

3.3 Method

In the thesis, for each bigram and composing words in the dataset, a vector that includes co-occurring frequencies of words is built. The words that composes the vector are the ones that reside in same sentence with the target (bigram or a constituents of a bigram) and the frequency is measured from the sentences where the word and the target co-occurs. The experiments are deployed on Bilkent, Muder and METU corpora in the thesis.

The first step in vector space modelling is determining the sentences in the corpus. Since the sentences are already tagged in Bilkent and METU corpora, no preprocessing is required. However, sentence segmentation of Muder corpus was to be performed. The texts in these corpora are split into sentences by pre-defined delimiters such as dot, exclamation mark. Though this automatic segmentation may fail in some cases, we believe that our next pre-processing step; elimination of sentences that are shorter than two words; may reduce the number of failing sentences.

Each bigram in the final set of 2229 bigrams (gold standard) is searched through the sentences of corpora and we kept any sentence that contains at least one of the bigrams in the gold standard. Table 3.11 gives the number of sentences where at least one of the bigrams may be observed.

We employ our VSM computations in MATLAB environment. In Figure 3.3 a flowchart diagram for MATLAB is shown for better understanding of the structure. MATLAB code explanation and details can be found in Appendix A.

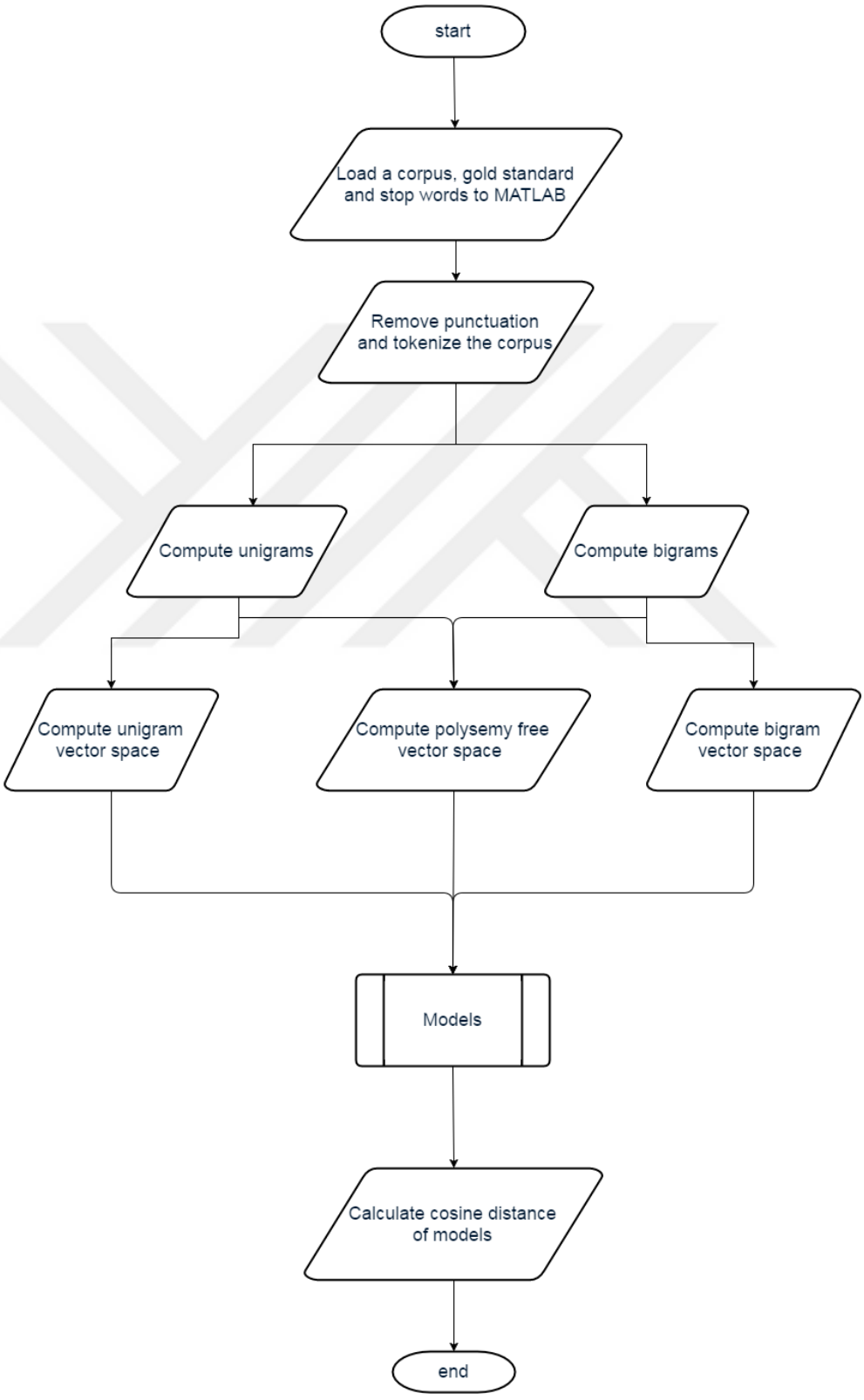

Figure 3.3: How MATLAB code works.

Corpus	# Sentence Counts
Bilkent	48,268
METU	178,417
Muder	41,864

Table 3.10: Manually and automatically generated corpora sentences.

Corpus	# Sentence Counts	# Word Counts	# Char Counts
Bilkent	43,295	745,001	5,437,711
METU	148,911	2,105,652	14,953,470
Muder	39,767	670,031	4,997,683

Table 3.11: Refined sentences and their statistics.

We created VSM vectors in a window size of 5[2] and limited them within sentence. Simply each vector includes the frequencies of co-occurring words of the target bigram or unigram. To measure similarity between these vectors, we use cosine similarity as in given below.

$$sim(\vec{V_1}, \vec{V_2}) = \frac{\vec{V_1} \cdot \vec{V_2}}{\|\vec{V_1}\|\|\vec{V_2}\|} \tag{3.4}$$

In our experiments, we have used a normalized cosine similarity function that produces values in range [0,2] instead of range [-1,1] (0 indicates the exact similarity between vectors, 2 is vice versa).

In the thesis, the compositionality for a given bigram is measured by 5 different models that can be obtained from the following equation:

$$\begin{aligned} \alpha(\vec{w_1}, \vec{w_2}) = a + b * sim(\overrightarrow{w_1 w_2}, \vec{w_1}) \\ +c * sim(\overrightarrow{w_1 w_2}, \vec{w_2}) \\ +d * sim(\overrightarrow{w_1 w_2}, \vec{w_1} + \vec{w_2}) \\ +e * sim(\overrightarrow{w_1 w_2}, \vec{w_1} * \vec{w_2}) \end{aligned} \tag{3.5}$$

[2]preceding and following 5 words of the target word/word combination

In the equation 3.5 that is proposed by Reddy et al. [25], $\overrightarrow{w_1w_2}$ corresponds to the context vector of the bigram w_1w_2 and $\vec{w_1}$, $\vec{w_2}$ corresponds to the vector of w_1, w_2 respectively. In our experiments, we have employed 3 different set of models. The brief definitions of model sets are given below.

Set 1 (S_1): This set includes 5 different models where the vectors include raw frequencies of neighbouring words that reside in same window size with the target. For example, in Model 1 the similarity of bigram and the first word in bigram is measured. In this model, the vector of first word is composed of all neighbouring words in all sentences that includes the first word.

$$
\begin{aligned}
(S_1M_1): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_1}) \\
(S_1M_2): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_2}) \\
(S_1M_3): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_1} + \vec{w_2}) \\
(S_1M_4): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_1} * \vec{w_2}) \\
(S_1M_5): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_1}) \\
&\quad + sim(\overrightarrow{w_1w_2}, \vec{w_2}) \\
&\quad + sim(\overrightarrow{w_1w_2}, \vec{w_1} + \vec{w_2}) \\
&\quad + sim(\overrightarrow{w_1w_2}, \vec{w_1} * \vec{w_2})
\end{aligned}
$$

In models S_1M_1 and S_1M_2, the semantic similarity between the bigram and its constituents are measured If the given bigram is non-compositional it is expected that the similarity score of these vectors will not be high.

In models S_1M_3 and S_1M_4, the similarity between the bigram and a combined version of vectors(pointwise addition and multiplication) for the constituents are measured as in Mitchell & Lapata [20].

Finally in model S_1M_5, the results of the previous models are summed up.

Set 2 (S_2): This set includes models where the vectors of component words are refined. In this set, the vector for each constituent is built by the sentences that includes the constituent but not the bigram. As a result the refined vector for $\vec{w_1}$ is $\vec{w_1'} = \vec{w_1} - \overrightarrow{w_1w_2}$ and $\vec{w_2}$ is $\vec{w_2'} = \vec{w_2} - \overrightarrow{w_1w_2}$.

$$
\begin{aligned}
(S_2M_1): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_1'}) \\
(S_2M_2): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_2'}) \\
(S_2M_3): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_1'} + \vec{w_2'}) \\
(S_2M_4): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_1'} * \vec{w_2'}) \\
(S_2M_5): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_1'}) \\
&\quad + sim(\overrightarrow{w_1w_2}, \vec{w_2'}) \\
&\quad + sim(\overrightarrow{w_1w_2}, \vec{w_1'} + \vec{w_2'}) \\
&\quad + sim(\overrightarrow{w_1w_2}, \vec{w_1'} * \vec{w_2'})
\end{aligned}
$$

Set 3 (S_3): In Set 3 while building the vectors of constituents irrelevant sentences in the corpus are removed. For example, building the vector of $\vec{w_1}$, only the sentences that includes both w_1 and a word that is semantically related to w_2 are considered and the other sentences that includes only w_1 are removed.

In [25], it is stated that, the composing words of a bigram may be used in a different context that may be unrelated to the regarding bigram. Reddy et.al. [25] exemplified this by the bigram *traffic light*. The composing word *light* may occur in different context in corpus. And some of the occurrences may be unrelated to notion of *traffic light*. This unrelated occurrences tend to decrease the semantic relation between the composing words; *light* and *traffic*. In order to decide relevant occurrences of *light*, a group of words that appears in similar context of *traffic* is defined. This group of words will be named as context words from now on. While building the vector of *light* the sentences where both *light* and at least one of the context words of *traffic* are selected, the other sentences where only *light* is observed are accepted to be in a context that is unrelated to *traffic light*.

In this thesis, for each composing word a group of context words is determined. The context words group includes the words that are most frequently co-occurring words with the regarding word. Simply, for each word, the most frequently co-occurring words are listed in the corpus, the stop words are removed from the list and finally the first five words are assigned as context words. The list of stop words that is given in [4] is used. Following the models in Set 3 are

given:

$$
\begin{aligned}
(S_3M_1): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_1}^r) \\
(S_3M_2): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_2}^r) \\
(S_3M_3): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_1}^r + \vec{w_2}^r) \\
(S_3M_4): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_1}^r * \vec{w_2}^r) \\
(S_3M_5): &\quad sim(\overrightarrow{w_1w_2}, \vec{w_1}^r) \\
&\quad + sim(\overrightarrow{w_1w_2}, \vec{w_2}^r) \\
&\quad + sim(\overrightarrow{w_1w_2}, \vec{w_1}^r + \vec{w_2}^r) \\
&\quad + sim(\overrightarrow{w_1w_2}, \vec{w_1}^r * \vec{w_2}^r)
\end{aligned}
$$

3.4 Evaluation

The evaluation of models is performed in 3 steps. For each model:

1. We sorted bigrams according to the similarity score that is produced by model in decreasing order. It is accepted that if the similarity score of a bigram is low, than it is non-compositional.

2. We measured precision, recall and F-measure scores in a pointwise manner. In other words, the evaluation is performed for set size N where N is sorted from 1 to the total set size.

3. Average precision, recall and F-measure scores are compared to the other averaged values.

Precision is the fraction of retrieved examples that are relevant, while recall is the fraction of relevant examples that are retrieved. Both precision and recall are therefore based on a harmony and measure of relevance. Following, the formulas are given for precision and recall.

$$Precision = \frac{\text{number of correctly identified terms}}{\text{number of identified terms}} \tag{3.6}$$

$$Recall = \frac{\text{number of correctly identified terms}}{\text{number of gold standard terms}} \tag{3.7}$$

In the statistical analysis, F-score is a measure of that considers both the precision and the recall of the results. F-score treated as a weighted average of the precision and recall, where an F-score reaches its greatest value at 1 and lowest at 0. F-score is presented as follows:

$$Fscore = 2 * \frac{precision * recall}{precision + recall} \tag{3.8}$$

Chapter 4

Experimental Results

In experiments, three sets of models are tested for 3 corpora of different sizes, Bilkent, Muder and METU corpus.

Table 4.1 presents the average precision, recall and F-measure values obtained from Bilkent corpus. The number of bigrams that reside both in gold standard and Bilkent corpus is 957 in which 63.32% of bigrams is annotated as non-compositional. In Table 4.1 the bold cells represent the highest scores for three metrics. The highest averaged F-value and recall is obtained by S_2M_2. Considering average F-values it is observed that in Bilkent corpus, Set 2 outperforms the other sets of models. As a result, it is possible to state that the refined vectors of composing words; the vectors that are built by the sentences that include the constituents but not the bigram; are better representatives to detect non-compositionality.

In Figure 4.1 F-measure curves of the models are given for Bilkent corpus.The vertical axis in figure is the F-value and the horizontal axis is the percentage of data set that is completed for the given F-value. Similar to the results that are revealed by average scores, the curves of the models in Set 2 are holding the higher F values compared to other sets. Considering F curves it is again observed while building the vectors of constituents ignoring the sentences that includes regarding bigrams increases the performance.

In Table 4.2, the average scores of evaluation are given for Muder corpus. Muder corpus includes 798 bigrams (56% non-compositional, 44% compositional) of gold standard. The maximum F-value is obtained in model S_2M_2. In Figure 4.2, F-curves of Muder corpus are presented. Similar to the results of the Bilkent corpus, models in Set 2 generate higher average F-values compared to other sets. Another important result that is examined from Figure 4.2 is that the models in different sets did not generate distinguishing F-curves opposing to the results of Bilkent corpus.

1129 of bigrams in gold standard is observed in METU corpus. In Table 4.3 and in Figure 4.3 evaluation results of METU corpus are illustrated. Considering the average values and the F-score curves, the models in Set 2 are performing better compared to the other models, supporting the results in previous corpora

Based on the overall results of 3 corpora, it is examined that Model 2 in Set 2 is succeeding in Turkish corpora in this experimental set up. Though the size of the corpus changes the evaluation scores, the best model or the best set of models do not differ according to the corpus size.

	AVERAGE PRECISION	**AVERAGE RECALL**	**AVERAGE F-MEASURE**
S_1M_1	0.630	0.494	0.501
S_1M_2	0.602	0.486	0.490
S_1M_3	0.586	0.473	0.476
S_1M_4	0.576	0.467	0.469
S_1M_5	0.588	0.476	0.479
S_2M_1	0.733	0.548	0.564
S_2M_2	0.737	**0.555**	**0.570**
S_2M_3	0.737	0.548	0.565
S_2M_4	0.710	0.542	0.557
S_2M_5	**0.741**	0.551	0.567
S_3M_1	0.636	0.495	0.503
S_3M_2	0.638	0.496	0.504
S_3M_3	0.611	0.484	0.489
S_3M_4	0.629	0.485	0.493
S_3M_5	0.622	0.490	0.496

Table 4.1: Average precision, recall and F-measure values obtained from Bilkent corpus.

	AVERAGE PRECISION	AVERAGE RECALL	AVERAGE F-MEASURE
S_1M_1	0.562	0.594	0.521
S_1M_2	0.551	0.585	0.513
S_1M_3	0.535	0.574	0.500
S_1M_4	0.541	0.576	0.503
S_1M_5	0.547	0.582	0.509
S_2M_1	0.580	0.597	0.527
S_2M_2	**0.593**	**0.608**	**0.537**
S_2M_3	0.587	0.594	0.524
S_2M_4	0.575	0.591	0.521
S_2M_5	**0.593**	0.599	0.529
S_3M_1	0.583	0.601	0.530
S_3M_2	0.585	0.603	0.533
S_3M_3	0.560	0.590	0.517
S_3M_4	0.573	0.594	0.522
S_3M_5	0.575	0.599	0.528

Table 4.2: Average precision, recall and F-measure values obtained from Muder corpus.

	AVERAGE PRECISION	AVERAGE RECALL	AVERAGE F-MEASURE
S_1M_1	0.640	0.509	0.519
S_1M_2	0.632	0.504	0.513
S_1M_3	0.613	0.491	0.499
S_1M_4	0.614	0.484	0.492
S_1M_5	0.617	0.495	0.503
S_2M_1	0.744	0.550	0.570
S_2M_2	0.740	**0.551**	0.569
S_2M_3	0.742	0.546	0.565
S_2M_4	0.736	0.548	0.568
S_2M_5	**0.748**	**0.551**	**0.571**
S_3M_1	0.658	0.511	0.523
S_3M_2	0.653	0.513	0.524
S_3M_3	0.632	0.502	0.511
S_3M_4	0.652	0.504	0.516
S_3M_5	0.646	0.510	0.520

Table 4.3: Average precision, recall and F-measure values obtained from METU corpus.

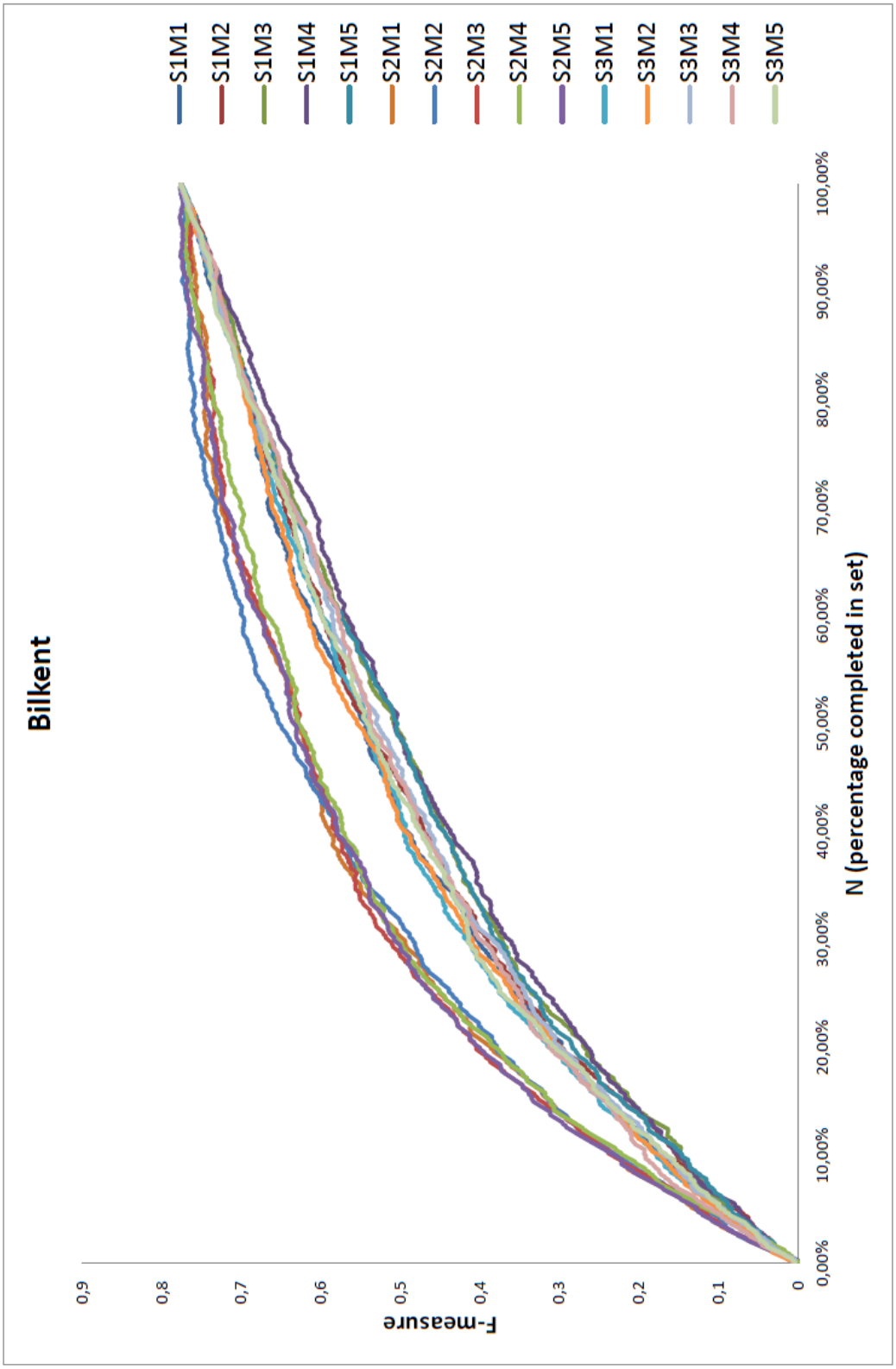

Figure 4.1: Average F-measure curves of models from Bilkent corpus.

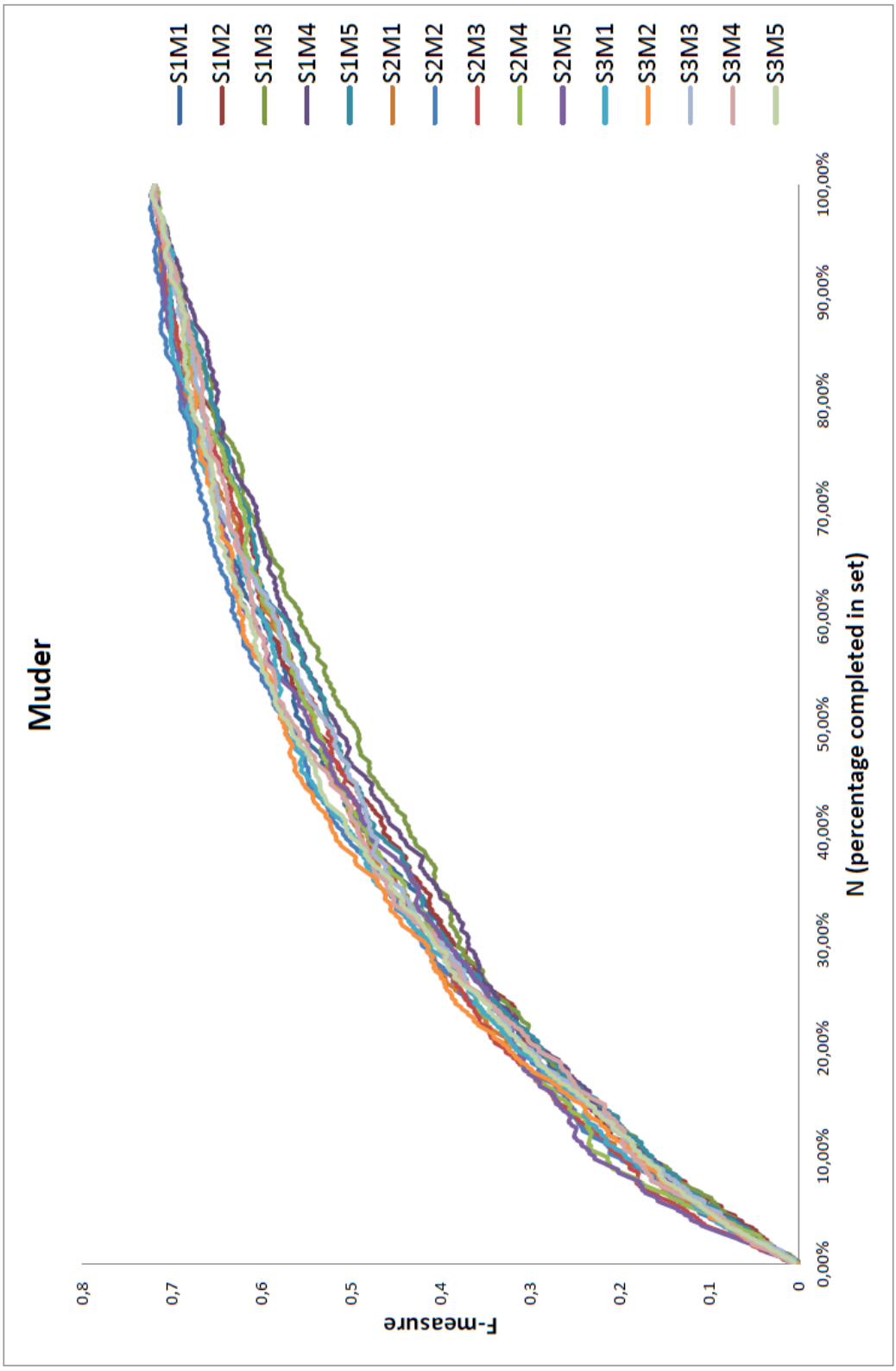

Figure 4.2: Average F-measure curves of models from Muder corpus.

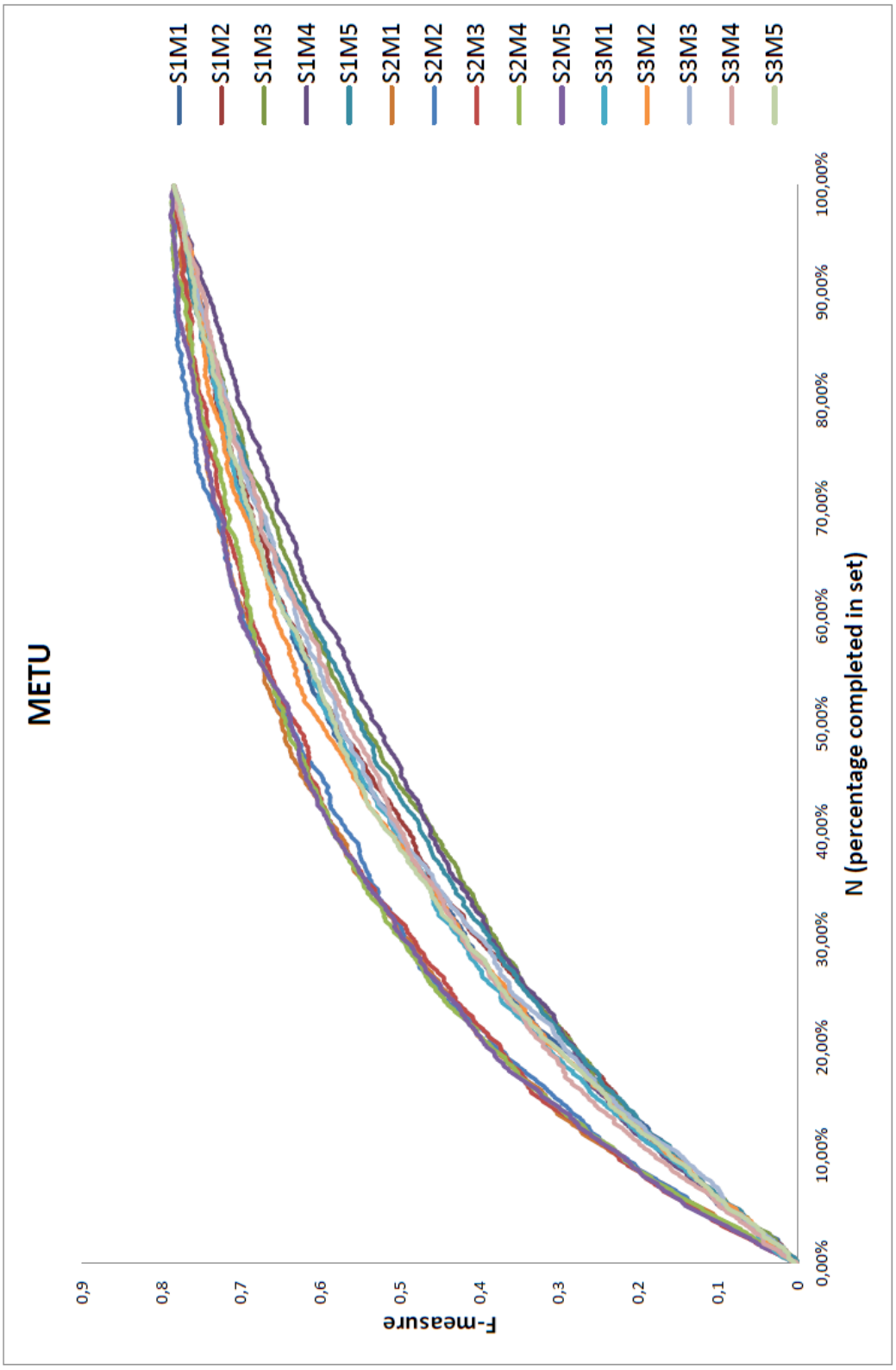

Figure 4.3: Average F-measure curves of models from METU corpus.

Chapter 5

Conclusion

In this thesis, we analyzed the semantic compositionality/non-compositionality in Turkish by vector space models. We introduced three sets of 5 different VSMs that assess the non-compositionality in Turkish. VSMs of Set 2; the models where the vector of composing words are built by ignoring the sentences that hold the word combination; are observed to provide better performance results compared to other models. It is also examined that as the size of the corpus increases, the difference in performances of successful and unsuccessfull methods becomes more significant.

Due to the high time and space complexity of the algorithms that are used to implement models, we were unable to work on larger corpus. As a future work, we are planning to repeat our experiments in a larger corpora and with different settings (e.g. windows size, stemmed/surface formed corpus, binary/weighted vectors, unigrams/bigrams/trigrams).

BIBLIOGRAPHY

[1] Web site link: http://www.cnnturk.com.

[2] Web site link: http://www.haber7.com.

[3] Web site link: http://www.milliyet.com.tr.

[4] Web site link: http://www.ranks.nl/stopwords/turkish.

[5] Web site link: http://www.trt.net.tr.

[6] Web site link: http://www.zaman.com.tr.

[7] Timothy Baldwin. Compositionality and multiword expressions: Six of one, half a dozen of the other.

[8] Chris Biemann and Eugenie Giesbrecht. Distributional semantics and compositionality 2011: Shared task description and results. In *Proceedings of the workshop on distributional semantics and compositionality*, pages 21–28. Association for Computational Linguistics, 2011.

[9] Fan Bu, Xiaoyan Zhu, and Ming Li. Measuring the non-compositionality of multiword expressions. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 116–124. Association for Computational Linguistics, 2010.

[10] Fazlı Can, Seyit Koçberber, Özgür Bağlıoğlu, Süleyman Kardas, H. Cagdaş Öcalan, and Erkan Uyar. New event detection and topic tracking in turkish. *Journal of the American Society for Information Science and Technology*, 61(4):802–819, 2010.

[11] Tanmoy Chakraborty, Santanu Pal, Tapabrata Mondal, Tanik Saikh, and Sivaju Bandyopadhyay. Shared task system description: Measuring the compositionality of bigrams using statistical methodologies. In *Proceedings of the Workshop on Distributional Semantics and Compositionality*, pages 38–42. Citeseer, 2011.

[12] Yaacov Choueka. Looking for needles in a haystack or locating interesting collocational expressions in large textual databases. In *RIAO 88:(Recherche d'Information Assistée par Ordinateur). Conference*, pages 609–623, 1988.

[13] Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29, 1990.

[14] Bekir Taner Dinçer. Türkçe için istatistiksel bir bilgi geri-getirim sistemi, 2004.

[15] John Rupert Firth. *Papers in linguistics, 1934-1951*. Oxford University Press, 1957.

[16] Guillermo Garrido and Anselmo Peñas. Detecting compositionality using semantic vector space models based on syntactic context: shared task system description. In *Proceedings of the Workshop on Distributional Semantics and Compositionality*, pages 43–47. Association for Computational Linguistics, 2011.

[17] Anders Johannsen, Hector Martinez Alonso, Christian Rishøj, and Anders Søgaard. Shared task system description: Frustratingly hard compositionality prediction. In *Proceedings of the Workshop on Distributional Semantics and Compositionality*, pages 29–32. Association for Computational Linguistics, 2011.

[18] Alfredo Maldonado-Guerra and Martin Emms. Measuring the compositionality of collocations via word co-occurrence vectors: Shared task system description. In *Proceedings of the Workshop on Distributional Semantics and Compositionality*, pages 48–53. Association for Computational Linguistics, 2011.

[19] Senem Kumova Metin and Bahar Karaoğlan. Collocation extraction in turkish texts using statistical methods. In *Advances in Natural Language Processing*, pages 238–249. Springer, 2010.

[20] Jeff Mitchell and Mirella Lapata. Vector-based models of semantic composition. In *ACL*, pages 236–244, 2008.

[21] Ted Pedersen. Identifying collocations to measure compositionality: shared task system description. In *Proceedings of the Workshop on Distributional Semantics and Compositionality*, pages 33–37. Association for Computational Linguistics, 2011.

[22] Francis Jeffry Pelletier. The principle of semantic compositionality. *Topoi*, 13(1):11–24, 1994.

[23] Uwe Quasthoff, Matthias Richter, and Christian Biemann. Corpus portal for search in monolingual corpora. In *Proceedings of the fifth international conference on language resources and evaluation*, volume 17991802, 2006.

[24] Dragomir R Radev, Simone Teufel, Horacio Saggion, Wai Lam, John Blitzer, Hong Qi, Arda Celebi, Danyu Liu, and Elliott Drabek. Evaluation challenges in large-scale document summarization. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 375–382. Association for Computational Linguistics, 2003.

[25] Siva Reddy, Diana McCarthy, Suresh Manandhar, and Spandana Gella. Exemplar-based word-space model for compositionality detection: Shared task system description. In *Proceedings of the Workshop on Distributional Semantics and Compositionality*, pages 54–60. Association for Computational Linguistics, 2011.

[26] Herbert Rubenstein and John B Goodenough. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633, 1965.

[27] Bilge Say, Deniz Zeyrek, Kemal Oflazer, and Umut Özge. Development of a corpus and a treebank for present-day written turkish. In *Proceedings of the eleventh international conference of Turkish linguistics*, pages 183–192, 2002.

[28] Gökhan Tür, Dilek Hakkanı-Tür, and Kemal Oflazer. A statistical information extraction system for turkish. *Natural Language Engineering*, 9:181–210, 6 2003.

[29] Peter D Turney, Patrick Pantel, et al. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37(1):141–188, 2010.

[30] Tim Van de Cruys. Two multivariate generalizations of pointwise mutual information. In *Proceedings of the Workshop on Distributional Semantics and Compositionality*, pages 16–20. Association for Computational Linguistics, 2011.

[31] Eva Maria Vecchi, Marco Baroni, and Roberto Zamparelli. (linear) maps of the impossible: capturing semantic anomalies in distributional space. In *Proceedings of the Workshop on Distributional Semantics and Compositionality*, pages 1–9. Association for Computational Linguistics, 2011.

[32] Fabio Massimo Zanzotto and Lorenzo Dell'Arciprete. Distributed structures and distributional meaning. In *Proceedings of the Workshop on Distributional Semantics and Compositionality*, pages 10–15. Association for Computational Linguistics, 2011.

Appendix A

MATLAB Code for Vector Space Models

A.1 Initialization Segment

```
corpus=evalc('type C:\corpus.txt');
unigrams=evalc('type C:\unigrams.txt');
bigrams=evalc('type C:\bigrams.txt');
stop=evalc('type C:\stopwords.txt');

corpus=regexprep(corpus,'\W',' ');
unigrams=regexprep(unigrams,'\W',' ');
bigrams=regexprep(bigrams,'\W',' ');
stop=regexprep(stop,'\W',' ');

corpus=strtrim(regexprep(corpus,'\s*',' '));
unigrams=strtrim(regexprep(unigrams,'\s*',' '));
bigrams=strtrim(regexprep(bigrams,'\s*',' '));
stop=strtrim(regexprep(stop,'\s*',' '));

tempc=regexprep(corpus,' ',''',''');
tempun=regexprep(unigrams,' ',''',''');
tempbi=regexprep(bigrams,' ',''',''');
temps=regexprep(stop,' ',''',''');
```

```
20
21 eval(['words = { ''',tempc,'''};']);
22 eval(['uns = { ''',tempun,'''};']);
23 eval(['bi = { ''',tempbi,'''};']);
24 eval(['stops = { ''',temps,'''};']);
25
26 running_words = length(words);
27 vocab = unique(words);
28 vocab_words = length(vocab);
29 [vocab,void,index] = unique(words);
30 freq = hist(index,vocab_words);
31 [ranked_freq,ranking_idx] = sort(freq,'descend');
32 ranked_vocab = vocab(ranking_idx);
```

A.2 Bigrams Creation

```
1  biRS = transpose(reshape(bi,[2,2228]));
2  bimtx = sparse(length(ranked_vocab),length(ranked_vocab));
3
4  for z=2:length(words)
5      g1 = words{z-1}; g2 = words{z};
6      for t=1:length(biRS)
7          if strcmp(biRS(t,1),g1)&strcmp(biRS(t,2),g2)
8              nidx1 = find(strcmp(ranked_vocab,g1));
9              nidx2 = find(strcmp(ranked_vocab,g2));
10             bimtx(nidx1,nidx2) = bimtx(nidx1,nidx2)+1;
11         end;
12     end;
13 end;
14
15 [g1,g2,rawcount] = find(bimtx);
16 [bigram_count,bigram_idx] = sort(rawcount,'descend');
17 bigram_word1 = ranked_vocab(g1(bigram_idx));
18 bigram_word2 = ranked_vocab(g2(bigram_idx));
```

A.3 Unigram VSM

```
1  uniVSMmtx = sparse(length(ranked_vocab),length(ranked_vocab));
```

```
wsize=5;

for k=wsize*2+1:length(words)
    w6 = words{k-5};
    if find(strcmp(w6,uns))
        w1 = words{k}; w2 = words{k-1}; w3 = words{k-2}; w4 = words{k-3}; w5 = words{k-4};
        w7 = words{k-6}; w8 = words{k-7}; w9 = words{k-8}; w10 = words{k-9}; w11 = words{k-10};
        idx1 = find(strcmp(ranked_vocab,w1));
        idx2 = find(strcmp(ranked_vocab,w2));
        idx3 = find(strcmp(ranked_vocab,w3));
        idx4 = find(strcmp(ranked_vocab,w4));
        idx5 = find(strcmp(ranked_vocab,w5));
        idx6 = find(strcmp(ranked_vocab,w6));
        idx7 = find(strcmp(ranked_vocab,w7));
        idx8 = find(strcmp(ranked_vocab,w8));
        idx9 = find(strcmp(ranked_vocab,w9));
        idx10 = find(strcmp(ranked_vocab,w10));
        idx11 = find(strcmp(ranked_vocab,w11));
        if not(strcmp(w2,'EOF'))&not(strcmp(w3,'EOF'))&not(strcmp(w4,'EOF'))&not(strcmp(w5,'EOF'))&not(strcmp(w1,'EOF'))
            uniVSMmtx(idx6,idx1) = uniVSMmtx(idx6,idx1)+1;
        end;
        if not(strcmp(w3,'EOF'))&not(strcmp(w4,'EOF'))&not(strcmp(w5,'EOF'))&not(strcmp(w2,'EOF'))
            uniVSMmtx(idx6,idx2) = uniVSMmtx(idx6,idx2)+1;
        end;
        if not(strcmp(w4,'EOF'))&not(strcmp(w5,'EOF'))&not(strcmp(w3,'EOF'))
            uniVSMmtx(idx6,idx3) = uniVSMmtx(idx6,idx3)+1;
        end;
        if not(strcmp(w5,'EOF'))&not(strcmp(w4,'EOF'))
            uniVSMmtx(idx6,idx4) = uniVSMmtx(idx6,idx4)+1;
        end;
        if not(strcmp(w5,'EOF'))
            uniVSMmtx(idx6,idx5) = uniVSMmtx(idx6,idx5)+1;
        end;
        if not(strcmp(w7,'EOF'))
            uniVSMmtx(idx6,idx7) = uniVSMmtx(idx6,idx7)+1;
        end;
```

```
38          if not(strcmp(w7,'EOF'))&not(strcmp(w8,'EOF'))
39              uniVSMmtx(idx6,idx8) = uniVSMmtx(idx6,idx8)+1;
40          end;
41          if not(strcmp(w7,'EOF'))&not(strcmp(w8,'EOF'))&not(strcmp(w9,'EOF'))
42              uniVSMmtx(idx6,idx9) = uniVSMmtx(idx6,idx9)+1;
43          end;
44          if not(strcmp(w7,'EOF'))&not(strcmp(w8,'EOF'))&not(strcmp(w9,'EOF'))&not(
                strcmp(w10,'EOF'))
45              uniVSMmtx(idx6,idx10) = uniVSMmtx(idx6,idx10)+1;
46          end;
47          if not(strcmp(w7,'EOF'))&not(strcmp(w8,'EOF'))&not(strcmp(w9,'EOF'))&not(
                strcmp(w10,'EOF'))&not(strcmp(w11,'EOF'))
48              uniVSMmtx(idx6,idx11) = uniVSMmtx(idx6,idx11)+1;
49          end;
50      end;
51  end;
```

A.4 Bigram VSM

```
1   biVSMmtx = sparse(length(ranked_vocab),length(ranked_vocab));
2   wsize2=5;
3
4   for i=wsize2*2+2:length(words)
5       x6 = words{i-5}; x7 = words{i-6};
6       word1 = strcmp(bigram_word1,x7);
7       word2 = strcmp(bigram_word2,x6);
8       if (find(word1&word2))
9           index2 = find(word1&word2);
10          x1 = words{i}; x2 = words{i-1}; x3 = words{i-2}; x4 = words{i-3}; x5 =
                words{i-4};
11          x8 = words{i-7}; x9 = words{i-8}; x10 = words{i-9};
12          x11 = words{i-10}; x12 = words{i-11};
13          bidx1 = find(strcmp(ranked_vocab,x1));
14          bidx2 = find(strcmp(ranked_vocab,x2));
15          bidx3 = find(strcmp(ranked_vocab,x3));
16          bidx4 = find(strcmp(ranked_vocab,x4));
17          bidx5 = find(strcmp(ranked_vocab,x5));
18          bidx6 = find(strcmp(ranked_vocab,x6));
19          bidx7 = find(strcmp(ranked_vocab,x7));
```

```
20          bidx8 = find(strcmp(ranked_vocab,x8));
21          bidx9 = find(strcmp(ranked_vocab,x9));
22          bidx10 = find(strcmp(ranked_vocab,x10));
23          bidx11 = find(strcmp(ranked_vocab,x11));
24          bidx12 = find(strcmp(ranked_vocab,x12));
25          if not(strcmp(x2,'EOF'))&not(strcmp(x3,'EOF'))&not(strcmp(x4,'EOF'))&not(
                strcmp(x5,'EOF'))&not(strcmp(x1,'EOF'))
26              biVSMmtx(index2,bidx1) = biVSMmtx(index2,bidx1)+1;
27          end;
28          if not(strcmp(x3,'EOF'))&not(strcmp(x4,'EOF'))&not(strcmp(x5,'EOF'))&not(
                strcmp(x2,'EOF'))
29              biVSMmtx(index2,bidx2) = biVSMmtx(index2,bidx2)+1;
30          end;
31          if not(strcmp(x5,'EOF'))&not(strcmp(x4,'EOF'))&not(strcmp(x3,'EOF'))
32              biVSMmtx(index2,bidx3) = biVSMmtx(index2,bidx3)+1;
33          end;
34          if not(strcmp(x5,'EOF'))&not(strcmp(x4,'EOF'))
35              biVSMmtx(index2,bidx4) = biVSMmtx(index2,bidx4)+1;
36          end;
37          if not(strcmp(x5,'EOF'))
38              biVSMmtx(index2,bidx5) = biVSMmtx(index2,bidx5)+1;
39          end;
40          if not(strcmp(x8,'EOF'))
41              biVSMmtx(index2,bidx8) = biVSMmtx(index2,bidx8)+1;
42          end;
43          if not(strcmp(x8,'EOF'))&not(strcmp(x9,'EOF'))
44              biVSMmtx(index2,bidx9) = biVSMmtx(index2,bidx9)+1;
45          end;
46          if not(strcmp(x8,'EOF'))&not(strcmp(x9,'EOF'))&not(strcmp(x10,'EOF'))
47              biVSMmtx(index2,bidx10) = biVSMmtx(index2,bidx10)+1;
48          end;
49          if not(strcmp(x8,'EOF'))&not(strcmp(x9,'EOF'))&not(strcmp(x10,'EOF'))&not(
                strcmp(x11,'EOF'))
50              biVSMmtx(index2,bidx11) = biVSMmtx(index2,bidx11)+1;
51          end;
52          if not(strcmp(x8,'EOF'))&not(strcmp(x9,'EOF'))&not(strcmp(x10,'EOF'))&not(
                strcmp(x11,'EOF'))&not(strcmp(x12,'EOF'))
53              biVSMmtx(index2,bidx12) = biVSMmtx(index2,bidx12)+1;
54          end;
55      end;
```

```
56 end;
```

A.5 Initialization of Stop Words

```
1  for sin=1:length(stops)
2      if find(strcmp(ranked_vocab,stops(sin)))
3          stopindex(sin)=find(strcmp(ranked_vocab,stops(sin)));
4      else
5          stopindex(sin)=0;
6      end;
7  end;
8
9  indexx1 = zeros(1,length(bigram_word2));
10 indexx2 = zeros(1,length(bigram_word1));
11
12 for n=1:length(bigram_word1)
13     uniVSMindex = sparse(length(ranked_vocab),length(ranked_vocab));
14     indexx1(n) = find(strcmp(ranked_vocab,bigram_word2(n)));
15     indexx2(n) = find(strcmp(ranked_vocab,bigram_word1(n)));
16     td1=5;td2=5;rx1=1;rx2=1;
17     if indexx1(n)~=0&indexx2(n)~=0
18         [~,uniVSMindex(indexx1(n),:)] = sort(uniVSMmtx(indexx1(n),:),2,'descend');
19         while rx1<td1+1
20             if uniVSMindex(indexx1(n),rx1)~=stopindex
21                 uniVSMv1(n,rx1) = uniVSMindex(indexx1(n),rx1);
22                 rx1=rx1+1;
23             else
24                 td1=td1+1;
25                 rx1=rx1+1;
26             end;
27         end;
28         [~,uniVSMindex(indexx2(n),:)] = sort(uniVSMmtx(indexx2(n),:),2,'descend');
29         while rx2<td2+1
30             if uniVSMindex(indexx2(n),rx2)~=stopindex
31                 uniVSMv2(n,rx2) = uniVSMindex(indexx2(n),rx2);
32                 rx2=rx2+1;
33             else
34                 td2=td2+1;
35                 rx2=rx2+1;
```

```
36                  end;
37              end;
38          end;
39  end;
```

A.6 Polysemy Free VSM

```
1   limits = [0,find(strcmp(words,'EOF'))];
2   uniVSMR1 = sparse(length(ranked_vocab),length(ranked_vocab));
3   uniVSMR2 = sparse(length(ranked_vocab),length(ranked_vocab));
4   wsize3=5;
5
6   for j=wsize3*2+1:length(words)
7       y6 = words{j-5};
8       if find(strcmp(y6,uns))
9           y1 = words{j}; y2 = words{j-1}; y3 = words{j-2}; y4 = words{j-3}; y5 =
                words{j-4}; y7 = words{j-6};
10          y8 = words{j-7}; y9 = words{j-8}; y10 = words{j-9}; y11 = words{j-10};
11          ridx6 = find(strcmp(ranked_vocab,y6));
12          i_lower = find(limits <= j-5,1,'last');
13          i_higher = find(limits >= j-5,1,'first');
14          lnum = limits(i_lower);
15          hnum = limits(i_higher);
16          if not(j>hnum)
17              ridx1 = find(strcmp(ranked_vocab,y1));
18          else
19              ridx1=0;
20          end;
21          if not(j-1>hnum)
22              ridx2 = find(strcmp(ranked_vocab,y2));
23          else
24              ridx2=0;
25          end;
26          if not(j-2>hnum)
27              ridx3 = find(strcmp(ranked_vocab,y3));
28          else
29              ridx3=0;
30          end;
31          if not(j-3>hnum)
```

```
32          ridx4 = find(strcmp(ranked_vocab,y4));
33      else
34          ridx4=0;
35      end;
36      if not(j-4>hnum)
37          ridx5 = find(strcmp(ranked_vocab,y5));
38      else
39          ridx5=0;
40      end;
41      if not(j-6>hnum)
42          ridx7 = find(strcmp(ranked_vocab,y7));
43      else
44          ridx7=0;
45      end;
46      if not(j-7<lnum)
47          ridx8 = find(strcmp(ranked_vocab,y8));
48      else
49          ridx8=0;
50      end;
51      if not(j-8<lnum)
52          ridx9 = find(strcmp(ranked_vocab,y9));
53      else
54          ridx9=0;
55      end;
56      if not(j-9<lnum)
57          ridx10 = find(strcmp(ranked_vocab,y10));
58      else
59          ridx10=0;
60      end;
61      if not(j-10<lnum)
62          ridx11 = find(strcmp(ranked_vocab,y11));
63      else
64          ridx11=0;
65      end;
66      for jk=1:length(bigram_word1)
67          if strcmp(bigram_word1(jk),y6)
68              if find(uniVSMv1(jk,:)==ridx1|uniVSMv1(jk,:)==ridx2|uniVSMv1(jk,:)
                    ==ridx3|uniVSMv1(jk,:)==ridx4|uniVSMv1(jk,:)==ridx5|
                    uniVSMv1(jk,:)==ridx7|uniVSMv1(jk,:)==ridx8|uniVSMv1(jk,:)
                    ==ridx9|uniVSMv1(jk,:)==ridx10|uniVSMv1(jk,:)==ridx11)
```

```
69                  if not(strcmp(y2,'EOF'))&not(strcmp(y3,'EOF'))&not(strcmp(y4,'
                        EOF'))&not(strcmp(y5,'EOF'))&not(strcmp(y1,'EOF'))
70                      uniVSMR1(jk,ridx1) = uniVSMR1(jk,ridx1)+1;
71                  end;
72                  if not(strcmp(y3,'EOF'))&not(strcmp(y4,'EOF'))&not(strcmp(y5,'
                        EOF'))&not(strcmp(y2,'EOF'))
73                      uniVSMR1(jk,ridx2) = uniVSMR1(jk,ridx2)+1;
74                  end;
75                  if not(strcmp(y4,'EOF'))&not(strcmp(y5,'EOF'))&not(strcmp(y3,'
                        EOF'))
76                      uniVSMR1(jk,ridx3) = uniVSMR1(jk,ridx3)+1;
77                  end;
78                  if not(strcmp(y5,'EOF'))&not(strcmp(y4,'EOF'))
79                      uniVSMR1(jk,ridx4) = uniVSMR1(jk,ridx4)+1;
80                  end;
81                  if not(strcmp(y5,'EOF'))
82                      uniVSMR1(jk,ridx5) = uniVSMR1(jk,ridx5)+1;
83                  end;
84                  if not(strcmp(y7,'EOF'))
85                      uniVSMR1(jk,ridx7) = uniVSMR1(jk,ridx7)+1;
86                  end;
87                  if not(strcmp(y7,'EOF'))&not(strcmp(y8,'EOF'))
88                      uniVSMR1(jk,ridx8) = uniVSMR1(jk,ridx8)+1;
89                  end;
90                  if not(strcmp(y7,'EOF'))&not(strcmp(y8,'EOF'))&not(strcmp(y9,'
                        EOF'))
91                      uniVSMR1(jk,ridx9) = uniVSMR1(jk,ridx9)+1;
92                  end;
93                  if not(strcmp(y7,'EOF'))&not(strcmp(y8,'EOF'))&not(strcmp(y9,'
                        EOF'))&not(strcmp(y10,'EOF'))
94                      uniVSMR1(jk,ridx10) = uniVSMR1(jk,ridx10)+1;
95                  end;
96                  if not(strcmp(y7,'EOF'))&not(strcmp(y8,'EOF'))&not(strcmp(y9,'
                        EOF'))&not(strcmp(y10,'EOF'))&not(strcmp(y11,'EOF'))
97                      uniVSMR1(jk,ridx11) = uniVSMR1(jk,ridx11)+1;
98                  end;
99              end;
100         end;
101         if strcmp(bigram_word2(jk),y6)
```

```
102             if find(uniVSMv2(jk,:)==ridx1|uniVSMv2(jk,:)==ridx2|uniVSMv2(jk,:)
                    ==ridx3|uniVSMv2(jk,:)==ridx4|uniVSMv2(jk,:)==ridx5|
                    uniVSMv2(jk,:)==ridx7|uniVSMv2(jk,:)==ridx8|uniVSMv2(jk,:)
                    ==ridx9|uniVSMv2(jk,:)==ridx10|uniVSMv2(jk,:)==ridx11)
103               if not(strcmp(y2,'EOF'))&not(strcmp(y3,'EOF'))&not(strcmp(y4,'
                      EOF'))&not(strcmp(y5,'EOF'))&not(strcmp(y1,'EOF'))
104                 uniVSMR2(jk,ridx1) = uniVSMR2(jk,ridx1)+1;
105               end;
106               if not(strcmp(y3,'EOF'))&not(strcmp(y4,'EOF'))&not(strcmp(y5,'
                      EOF'))&not(strcmp(y2,'EOF'))
107                 uniVSMR2(jk,ridx2) = uniVSMR2(jk,ridx2)+1;
108               end;
109               if not(strcmp(y4,'EOF'))&not(strcmp(y5,'EOF'))&not(strcmp(y3,'
                      EOF'))
110                 uniVSMR2(jk,ridx3) = uniVSMR2(jk,ridx3)+1;
111               end;
112               if not(strcmp(y5,'EOF'))&not(strcmp(y4,'EOF'))
113                 uniVSMR2(jk,ridx4) = uniVSMR2(jk,ridx4)+1;
114               end;
115               if not(strcmp(y5,'EOF'))
116                 uniVSMR2(jk,ridx5) = uniVSMR2(jk,ridx5)+1;
117               end;
118               if not(strcmp(y7,'EOF'))
119                 uniVSMR2(jk,ridx7) = uniVSMR2(jk,ridx7)+1;
120               end;
121               if not(strcmp(y7,'EOF'))&not(strcmp(y8,'EOF'))
122                 uniVSMR2(jk,ridx8) = uniVSMR2(jk,ridx8)+1;
123               end;
124               if not(strcmp(y7,'EOF'))&not(strcmp(y8,'EOF'))&not(strcmp(y9,'
                      EOF'))
125                 uniVSMR2(jk,ridx9) = uniVSMR2(jk,ridx9)+1;
126               end;
127               if not(strcmp(y7,'EOF'))&not(strcmp(y8,'EOF'))&not(strcmp(y9,'
                      EOF'))&not(strcmp(y10,'EOF'))
128                 uniVSMR2(jk,ridx10) = uniVSMR2(jk,ridx10)+1;
129               end;
130               if not(strcmp(y7,'EOF'))&not(strcmp(y8,'EOF'))&not(strcmp(y9,'
                      EOF'))&not(strcmp(y10,'EOF'))&not(strcmp(y11,'EOF'))
131                 uniVSMR2(jk,ridx11) = uniVSMR2(jk,ridx11)+1;
132               end;
```

```
                end;
            end;
        end;
    end;
end;
```

A.7 Cosine Distance

```
matrix = zeros(length(bigram_word1),4);

for nx=1:length(bigram_word1)
    uniVSMidx1(nx) = find(strcmp(ranked_vocab,bigram_word1(nx)));
    uniVSMidx2(nx) = find(strcmp(ranked_vocab,bigram_word2(nx)));
    va=uniVSMmtx(uniVSMidx1(nx),:);
    vb=uniVSMmtx(uniVSMidx2(nx),:);
    vc=biVSMmtx(nx,:);
    matrix(nx,1)=pdist2(vc,va,'cosine');
    matrix(nx,2)=pdist2(vc,vb,'cosine');
    vd=va+vb;
    ve=va.*vb;
    matrix(nx,3)=pdist2(vc,vd,'cosine');
    matrix(nx,4)=pdist2(vc,ve,'cosine');
end;

matrix2 = zeros(length(bigram_word1),4);

for nx=1:length(bigram_word1)
    uniVSMidx1(nx) = find(strcmp(ranked_vocab,bigram_word1(nx)));
    uniVSMidx2(nx) = find(strcmp(ranked_vocab,bigram_word2(nx)));
    va=uniVSMmtx(uniVSMidx1(nx),:);
    vb=uniVSMmtx(uniVSMidx2(nx),:);
    vc=biVSMmtx(nx,:);
    matrix2(nx,1)=pdist2(vc,(va-vc),'cosine');
    matrix2(nx,2)=pdist2(vc,(vb-vc),'cosine');
    matrix2(nx,3)=pdist2(vc,(((va+vb)-vc)-vc),'cosine');
    matrix2(nx,4)=pdist2(vc,((va-vc).*(vb-vc)),'cosine');
end;

matrix3 = zeros(length(bigram_word1),4);
```

```
32
33 for nx=1:length(bigram_word1)
34     va=uniVSMR1(nx,:);
35     vb=uniVSMR2(nx,:);
36     vc=biVSMmtx(nx,:);
37     matrix3(nx,1)=pdist2(vc,va,'cosine');
38     matrix3(nx,2)=pdist2(vc,vb,'cosine');
39     vd=va+vb;
40     ve=va.*vb;
41     matrix3(nx,3)=pdist2(vc,vd,'cosine');
42     matrix3(nx,4)=pdist2(vc,ve,'cosine');
43 end;
```