Low-Bias Extraction of Domain-Specific Concepts

Der Fakultät für Mathematik und Informatik
der Universität Leipzig
eingereichte

DISSERTATION

zur Erlangung des akademischen Grades

Doctor Rerum Naturalium
(Dr. rer. nat.)

Im Fachgebiet
Informatik

vorgelegt

von **Dipl.-Inf. Axel-Cyrille Ngonga Ngomo**

geboren am 04. August 1983 in Bafoussam, Kamerun

Leipzig, den 08.12.2008

Acknowledgment

There are many people I am thankful to and I apologize for lacking the space necessary to mention you all by name.

First of all, I would like to thank Prof. Fähnrich for supervising this thesis, allowing me to explore this area of science and setting guidelines for the content of this work. I would still be writing on this work without his supervision. I am also endebted to colleagues of the Departments for Business Information Systems and Natural Language Processing at the University of Leipzig for discussions, hints and tips. In this regard, I am especially thankful to Frank Schumacher for his comments on practical issues concerning BorderFlow. I would also like to thank reviewers and fellow conference attendees for their constructive criticism of the approaches presented herein.

The quality of this manuscript was greatly enhanced by Mimy Gardner, Susan Little, Matthew Brown, Susanne Eckert, David Nnetu, Stefanie Böhm and Joseph Atoyebi. Thank you all for the careful proof reading and the many corrections. For their moral support in good and bad times, I would like to thank the whole Leipzig English Church in general and my home group in particular. Thank you for helping me to give my chaotic life a sense of priorities.

I am indebted to the German Research Foundation and to the German Ministry for Education and Research for stipends that allowed me to carry out my research without the burden of an extra job. I would like to thank especially Prof. Gerhard Brewka and Herrad Werner for their support in this respect.

Finally, I would like to thank my family. Thank you for being who you are and for taking me as I am. Especially, I would like to thank my parents for making my studies possible and giving me a solid basic education upon which the rest could be built.

Bibliographic Data

Title: Low-Bias Extraction of Domain-Specific Concepts
Author: Axel-Cyrille Ngonga Ngomo
Institution: Universität Leipzig, Fakultät für Mathematik und Informatik
221 pages, 53 figures, 31 tables, 3 appendices, 233 literature references

Abstract

The recent availability of domain-specific knowledge models in various forms has led to the development of information systems specialized on complex domains such as bio-medecine, tourism and chemistry. Domain-specific information systems rely on domain knowledge in forms such as terminologies, taxonomies and ontologies to represent, analyze, structure and retrieve information. While this integrated knowledge boosts the accuracy of domain-specific information systems, modeling domain-specific knowledge manually remains a challenging task. Therefore, considerable effort is being invested in developing techniques for the extraction of domain-specific knowledge from various resources in a semi-automatic fashion. Domain-specific text corpora are widely used for this purpose. Yet, most of the current approaches to the extraction of domain-specific knowledge in the form of terminologies or ontologies are limited in their portability to other domains and languages. The limitations result from the knowledge-rich paradigm followed by these approaches, i.e., from them demanding hand-crafted domain-specific and language-specific knowledge as input. Due to these constraints, domain-specific information systems exist currently for a limited number of domains for which domain-specific knowledge models are available. An approach to remedy the high human costs linked with the modeling of domain-specific knowledge is the use of low-bias, i.e., knowledge-poor and unsupervised approaches. They require little human effort but more computational power to achieve the same goals as their hand-crafted counterparts.

In this work, we propose the use of low-bias approaches for the extraction of domain-specific terminology and concepts from text. Especially, we study the low-bias extraction of concepts out of text using a combination of metrics for domain-

specific multi-word units and graph clustering techniques. The input for this approach consists exclusively of a domain-specific text corpus. We use a novel metric, the Smoothed Relative Expectation, to extract domain-specific multi-word units from the input data set. Subsequently, a novel binary clustering algorithm called SIGNUM is introduced and applied to the results of the metric. By these means, we compute a domain-specific lexicon. Finally, we use second-order collocations to extract the semantic features of the domain-specific terms contained in the domain-specific lexicon. These terms are then clustered to concepts using the third innovation of this work, the graph clustering algorithm BorderFlow. Our approach is unsupervised and makes no use of a-priori knowledge on language-specific patterns and the like. Therefore, it can be applied to virtually all domains and languages. We evaluate our approach on two domain-specific data sets from the bio-medical domain against domain-specific terminologies and standard clustering techniques.

This work is structured as follows: *Chapter 2* presents the state of the art in related research areas. First, we epitomize approaches to preprocessing text for the purpose of concept extraction, focusing on the extraction of domain-specific terminology. Thereafter, we give an overview of approaches to concept extraction from text. The approaches are differentiated according to the means through which they recognize concepts. We give emphasis to approaches based on natural language processing and on clustering. Then, we present prominent tools for ontology extraction. Thereafter, we epitomize graph theory, focusing on the terminology we use in the later parts of this work. Subsequently, we give an overview of data clustering, including both standard clustering techniques and graph clustering algorithms. Last, we present some considerations on evaluation and discuss the measures and statistical tools we chose to use in this work for the purpose of evaluation.

Chapter 3 and Chapter 4 are concerned with the low-bias preprocessing of text. In *Chapter 3*, we present and evaluate the Smoothed Relative Expectation (SRE) metric, a novel metric for the low-bias extraction of multi-word units. First, we introduce several characteristics of domain-specific terminology. These characteristics are then used to specify a model for domain-specific terminology. Subsequently, we utilize this model to specify the SRE metric. On the basis of two data sets of different size and cleanness, the accuracy of SRE is compared with six state-of-the-art metrics found in relevant literature. We conclude Chapter 3 by evaluating SRE against other multi-contextual metrics for the extraction of domain-specific terminology.

In *Chapter 4*, the findings of Chapter 3 are used to extract domain-specific lexica. First, the results obtained in Chapter 3 are used to generate word graphs of different topologies. Then, we present a graph algorithm for the extraction of domain-specific terminology. This algorithm, SIGNUM, relies on local information to achieve a binary clustering on word graphs. We present a formal specification of the basic SIGNUM algorithm. Then, we show how SIGNUM can be generalized so as to cluster hypergraphs. To evaluate our approach, we measure the precision and recall which SIGNUM achieves on simple graphs and on link graphs. Finally, we show how the results of SIGNUM can be used to compute high-degree multi-word units. The results of Chapter 4 are the basis for our approach to concept extraction.

In *Chapter 5*, we present the local graph clustering algorithm BorderFlow. This algorithm is designed to cluster large graphs by maximizing the intra-cluster similarity while minimizing the inter-cluster similarity. We first specify the algorithm formally. Subsequently, we prove that it can viably extract concepts by using it to cluster two categories of synthetic graphs. Then, we present a computationally less expensive heuristic for BorderFlow and use it to cluster large graphs extracted from the Wikipedia Category Graph. In a last step, we use BorderFlow for the purpose of concept extraction per se. The results obtained are evaluated quantitatively and qualitatively. The quantitative evaluation is carried out by computing the silhouette of the clusters generated by BorderFlow and comparing it with the silhouette of clusters extracted by kNN. The qualitative evaluation is carried out by measuring the purity of the clusters obtained.

The final chapter of this thesis, *Chapter 6*, summarizes our insights on the low-bias extraction of concepts. Furthermore, it presents possible extensions to our core algorithms. We also discuss possible applications of these algorithms to different areas of computer science. We conclude this work by presenting ideas for future research directions.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The recent availability of domain-specific knowledge models in various forms has led to the development of information systems specialized on complex domains (Mollá and Vicedo, 2007) such as bio-medecine (Doms and Schroeder, 2005), tourism (Benamara and Dizier, 2003) and chemistry (Barker et al., 2004). Domain-specific information systems (Jacob, 1999; Camon et al., 2003) rely on domain knowledge in forms such as terminologies, taxonomies and ontologies to represent, analyze, structure and retrieve information (Can and Baykal, 2007; Doms and Schroeder, 2005; Mollá and Vicedo, 2007). While this integrated knowledge boosts the accuracy of domain-specific information systems, modeling domain-specific knowledge manually remains a challenging task (Lin and Pantel, 2002; Maedche, 2002; Gómez-Pérez et al., 2004). Therefore, considerable effort is being invested in developing techniques for the extraction of domain-specific knowledge from various resources in a semi-automatic fashion (Hindle, 1990; Caraballo, 1999; Khan and Luo, 2002; Pantel, 2003; Zhou, 2007). Domain-specific text corpora are widely used for this purpose (Biemann, 2005; Heyer et al., 2006). Yet, most of the current approaches to the extraction of domain-specific knowledge in form of terminologies or ontologies are limited in their portability to other domains and languages. The limitations result from the knowledge-rich paradigm followed by these approaches, i.e., from them demanding hand-crafted domain-specific and language-specific knowledge as input (Omelayenko, 2001; Biemann, 2005; Zhou, 2007). Due to these constraints, domain-specific information systems exist currently for a limited number of domains for which domain-specific knowledge models are available (Minock, 2005; Mollá and Vicedo, 2007). An approach to remedy the high human costs linked with the modeling of domain-specific knowledge is the use of low-bias, i.e., knowledge-poor and unsupervised approaches (Biemann, 2007; Bordag, 2007). They require little human effort but more computational power to achieve the same goals as their hand-crafted counterparts. In this work, we propose the use of low-bias approaches for the extraction of domain-specific terminology and concepts from text.

1.1 Motivation

Domain-Specific Information Systems (DSIS) are characterized by their use of domain knowledge to represent, analyze, structure or retrieve information[1] (Can and Baykal, 2007; Doms and Schroeder, 2005; Mollá and Vicedo, 2007). DSIS range from domain-specific Information Retrieval (IR) systems (Henstock et al., 2001; Siadatyand et al., 2007) to Question Answering (QA) systems for restricted domains (Benamara and Dizier, 2003; Barker et al., 2004). The domain-specific knowledge integrated in DSIS defines "a common vocabulary for accessing information in a domain" (Mollá and Vicedo, 2007, p. 49). This knowledge is integrated in DSIS in different forms (Guarino, 1998) ranging from simple lists of domain-specific entities to formal ontologies. Most modern DSIS integrate knowledge as conceptual taxonomies (Henstock et al., 2001; Buitelaar et al., 2004) or as formal ontologies (Guarino, 1998; Benamara and Dizier, 2003).

The main bottleneck during the implementation of DSIS lies in the acquisition of domain-specific knowledge (Gómez-Pérez et al., 2004; Maedche and Staab, 2004; Mollá and Vicedo, 2007). During the last decades, large open-domain resources such as top-level ontologies (Suggested Merged Upper Ontology (Niles and Pease, 2001), General Formal Ontology (Degen et al., 2001)) and terminologies (WordNet (Miller, 1990), EuroWordNet (Vossen, 1998)) have been developed to model general knowledge. However, these resources are unsuitable for DSIS for three main reasons. First, open-domain resources tend to be too coarse-grained, i.e., they are unable to capture domain-specific knowledge in depth so as to model it accurately. For example, WordNet[2] does not contain the term *thrombocytopathy*, a term that designates an abnormality of the platelets in the bio-medical domain. Second, open-domain resources tend to be too fine-grained. Therefore, they contain polysemous terms that can reduce the accuracy of DSIS considerably. The term *vessel*, for example, bears the meaning of a "watercraft", an" object used as a container" and a "tube in which body fluid circulates" according to WordNet. Only the third meaning is commonly used in the bio-medical domain. Third, open-domain resources may contain incorrect interpretations of domain-specific terminology. For example, *acid* is either "any of various water-soluble compounds having a sour taste and capable of

[1] In this work, knowledge is used in the same sense as ontological resources in (Mollá and Vicedo, 2007), i.e., in the sense of all possible domain knowledge representations

[2] In this work, we use Version 3.0 of Wordnet. We accessed it on August 18^{th}, 2008.

turning litmus red and reacting with a base to form a salt" or a "street name for lysergic acid diethylamide" according to WordNet. Yet, it is a brand of house music in the musical domain (Lee et al., 2000) and a set of database properties in computer science (Gray, 1981).

A solution to this problem is the use of ontology learning techniques. However, current approaches to ontology learning are either knowledge-rich or rely on results of knowledge-rich approaches (Zhou, 2007). Modern ontology learning techniques encompass three main steps dubbed preprocessing, concept extraction and relation harvesting (Maedche and Staab, 2000; Zhang et al., 2001; Buitelaar et al., 2004). The first two are of interest for this work. The preprocessing step is the basis of the two other steps and is mainly concerned with the extraction of domain-specific terminology (Zhang et al., 2001; Turmo et al., 2006). This goal is usually achieved by using a combination of syntactic knowledge (Tlili-Guiassa, 2006; Singh et al., 2006), statistical techniques (Cutting et al., 1992; Leech et al., 1994) and differential analysis (Maedche and Staab, 2000; Buitelaar et al., 2004). The concept extraction step is usually based on deep parsing (Pantel, 2003), language-specific patterns (Hearst, 1992) or other high-level features (Biemann, 2005; Zhou, 2007).

The problems engendered by knowledge-rich approaches are manifold. First, the computation of syntactic categories relies heavily on knowledge about the language in which the corpus is written. Consequently, it is language-dependent and is not robust against mixed or noisy corpora. Second, techniques implementing knowledge-rich approaches cannot be ported to other domains or languages without being re-implemented or manually adapted to the new domains. The adaptation and re-implementation of these techniques involve the development of resources such as training sets (Tan et al., 2005), which demand an important amount of manual work and are therefore very time-consuming (Faure and Poibeau, 2000). Furthermore, differential analysis demands the use of well-balanced reference corpora, which are not always available and can be cost-intensive (Biemann, 2007). Last, modern tools for ontology learning are designed to be used by experts, restricting the set of possible users (Faure and Nédellec, 1999; Cucchiarelli et al., 2004).

1.2 Contributions

The main aim of our work is the development of a low-bias approach to the extraction of domain-specific knowledge. This thesis focuses on the extraction of domain-specific concepts with high purity and builds the foundation upon which techniques for relation harvesting and ontology extraction can be applied (Zhou, 2007). Hence, this work is closely related to ontology learning, which is defined by

Maedche (2002, p. 4) as the "integration of a multitude of disciplines in order to facilitate the construction of ontologies". Amongst all disciplines, particularly machine learning, natural language processing and statistical techniques have been used to learn ontologies (Faure and Poibeau, 2000). These approaches have been applied to several resources ranging from unstructured data such as natural language text to structured data such as database entries (Zhou, 2007). This thesis will be exclusively concerned with pre-segmented text as the source for concept extraction. In the whole of this work, we will define concepts as semantic classes (Lin and Pantel, 2002; Pantel, 2003; Mollá and Vicedo, 2007).

The contributions of this work to the low-bias extraction of domain-specific concepts for information systems are threefold (see Figure 1.1). First, we present a novel metric[3] for the low-bias extraction of domain-specific multi-word units (MWUs). As MWUs constitute a large subset of domain-specific terminology (Jiang and Tan, 2005), an accurate technique is needed to recognize n-grams which belong to the domain being investigated. To achieve this goal, we introduce a novel multi-contextual metric dubbed Smoothed Relative Expectation (SRE). SRE combines the distributional characteristics of domain-specific terminology over sentences and documents to compute a score for each n-gram in the corpus. Based on the results obtained by using SRE, we extract a graph representation of corpora.

The second contribution of our work is a graph-based approach to the extraction of the domain-specific lexica from text. For this purpose, we use the novel binary clustering algorithm SIGNUM. This algorithm uses the spreading activation principle to detect domain-specific terms. We use SIGNUM on several graph configurations and show how it builds upon the results of SRE to improve the precision of the terminology extracted.

The last step of this work consists of the extraction of domain-specific concepts per se. For this purpose, we developed the third contribution of this work, a general-purpose graph clustering algorithm called BorderFlow. BorderFlow was conceived with the aim of being suitable to cluster large graphs such as the similarity graphs that can result from terminology extraction. It maximizes the intra-cluster similarity and minimizes the inter-cluster similarity simultaneously by using a local search strategy. We use BorderFlow to extract semantic classes out of second-order-collocation graphs.

Based on evaluations against reference data sets, we first show that SRE yields better results than other metrics used for the extraction of MWUs. Subsequently, we show that the precision of SRE can be improved by using graph clustering. We show that the combination of SRE and SIGNUM can extract a relevant subset of the

[3]Throughout this work, we will use metric in the sense of measure and not in the mathematical sense of a distance function.

domain-specific terminology included in text corpora. Based on this automatically extracted terminology, we then show that BorderFlow can accurately detect domain-specific concepts. In a nutshell, we show that low-bias techniques can be used to extract background knowledge for domain-specific information systems with a high precision.

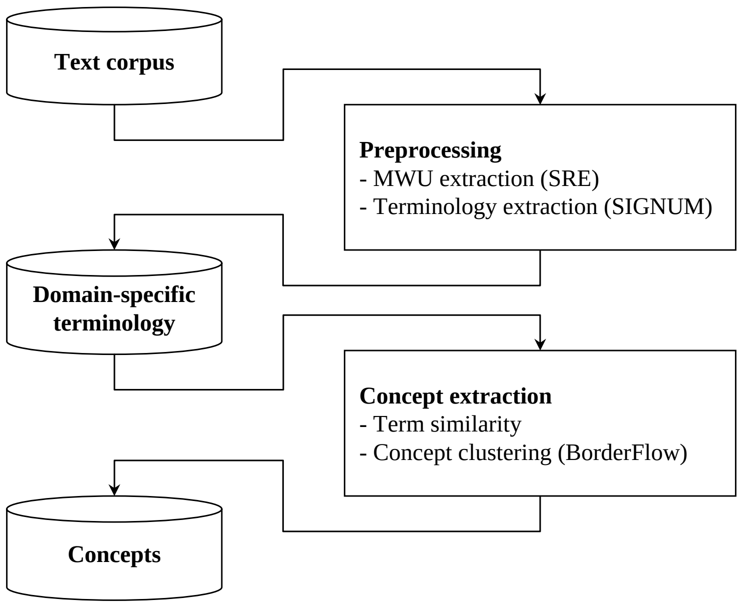


Figure 1.1: Contributions of this work

1.3 Chapter Overview

This work is structured as follows: *Chapter 2* presents the state of the art in related research areas. First, we epitomize approaches to preprocessing text for the purpose of concept extraction, focusing on the extraction of domain-specific terminology. Thereafter, we give an overview of approaches to concept extraction from text. The approaches are differentiated according to the means through which they recognize concepts. We give emphasis to approaches based on natural language processing and on clustering. Then, we present prominent tools for ontology extraction. Thereafter, we epitomize graph theory, focusing on the terminology we use in the later parts of this work. Subsequently, we give an overview of data clustering, including both standard clustering techniques and graph clustering algorithms. Last, we present some considerations on evaluation and discuss the measures and statistical tools we use in this work.

Chapter 3 and Chapter 4 are concerned with the low-bias preprocessing of text. In *Chapter 3*, we present and evaluate the Smoothed Relative Expectation (SRE) metric, a novel metric for the low-bias extraction of MWUs. First, we introduce several characteristics of domain-specific terminology. These characteristics are then used to specify a model for domain-specific terminology. Subsequently, we utilize this model to specify the SRE metric. On the basis of two data sets of different size and cleanness, the accuracy of SRE is compared with six state-of-the-art metrics found in relevant literature. We conclude Chapter 3 by evaluating SRE against other multi-contextual metrics for the extraction of domain-specific terminology.

In *Chapter 4*, the findings of Chapter 3 are used to extract domain-specific lexica. First, the results obtained in Chapter 3 are used to generate word graphs of different topologies. Then, we present a graph algorithm for the extraction of domain-specific terminology. This algorithm, SIGNUM, relies on local information to achieve a binary clustering on word graphs. We present a formal specification of the basic SIGNUM algorithm. Then, we show how SIGNUM can be generalized so as to cluster hypergraphs. To evaluate our approach, we measure the precision and recall which SIGNUM achieves on simple graphs and on link graphs generated by using the results of Chapter 3. Finally, we show how the results of SIGNUM can be used to compute high-degree MWUs. The results of Chapter 4 are the basis for our approach to concept extraction.

In *Chapter 5*, we present and evaluate the local graph clustering algorithm BorderFlow. This algorithm is designed to cluster large graphs by maximizing the intra-cluster similarity while minimizing the inter-cluster similarity. We first specify the algorithm formally. Subsequently, we prove that it can viably extract concepts by using it to cluster two categories of synthetic graphs. Then, we present a computationally less expensive heuristic for BorderFlow and use it to cluster large graphs extracted from the Wikipedia Category Graph. In a last step, we use BorderFlow for the purpose of concept extraction per se. The results obtained are evaluated quantitatively and qualitatively. The quantitative evaluation is carried out by measuring the silhouette (Rousseeuw, 1987) of the clusters generated by BorderFlow and comparing it with the silhouette of clusters extracted by kNN. The qualitative evaluation is carried out by measuring the purity of the clusters obtained.

The final chapter of this thesis, *Chapter 6*, summarizes our insights on the low-bias extraction of concepts. Furthermore, we present possible extensions to our core algorithms. We also discuss possible applications of these algorithms to different areas of computer science. We conclude this work by presenting ideas for future research directions.

Chapter 2

Background

In this chapter, we present background knowledge necessary to understand the work described in the subsequent chapters. The goal of this thesis lies in the extraction of domain-specific concepts out of text using graph clustering algorithms. Thus, this work is closely related to ontology learning and data clustering. Therefore, this section is structured as follows: first, we present an overview of the state of the art in ontology learning. Therein, we focus especially on preprocessing, concept extraction and tools for ontology learning. Then, we present some basics of graph theory, which we subsequently use to describe current approaches to data clustering in general and graph clustering in particular. The last section of this chapter presents considerations on the metrics and statistical tests we use to measure the quality and accuracy of results.

2.1 Preprocessing

Typical approaches to ontology learning consist of three main steps: preprocessing, concept extraction per se and relation harvesting (Maedche and Staab, 2000; Buitelaar et al., 2004). Preprocessing includes the steps from the transformation of the raw input data into a format suitable to the extraction of domain-specific terminology. The resulting terminology is subsequently used for the generation of concepts. These concepts are then labeled and finally put in relation to each other through a relation harvesting process. In this section, we elaborate on current preprocessing and concept extraction methods, as these two steps are the focus of this work. Current approaches to ontology learning use several preprocessing methods, of which the most common include token detection, conflation and categorization (Faure and Nédellec, 1999; Maedche and Staab, 2000; Turmo et al., 2006).

2.1.1 Token Detection

Token detection aims at the detection of terms from a given corpus, term being defined as a "meaningful constituent of a sentence" (Zhang et al., 2001, p. 2). This functionality is provided by two main categories of tools called segmenters and tokenizers. Both categories of tools have similar functionality: whilst segmenters extract words boundaries out of data streams especially in languages without blanks such as Chinese (Chen et al., 1997; Teahan et al., 2000), tokenizers deal with marking the boundaries of terms (Mikheev and Finch, 1997; Zhang et al., 2001).

State-of-the-art segmenters use several approaches that can be subdivided into two main categories: knowledge-driven and knowledge-free approaches. Knowledge-driven approaches use reference sources including dictionaries and taxonomies (Yao and Lua, 1998; Zhou and Liu, 2002) of the language at hand. Some approaches of this category prerequisite explicit a-priori knowledge about the frequency, the distribution or the semantics of words to analyze the input corpus (Cheng et al., 1999). A further category of knowledge-driven approaches, the so-called linguistic approaches, use grammar rules (Dai and Lee, 1994; Wu and Tseng, 1995) to compute word boundaries. Machine learning approaches use initial-state annotators and rule templates (Palmer, 1997; Hockenmaier and Brew, 1998) for the same purpose. Knowledge-free approaches use solely the information contained in the corpus to segment the input data. They focus on building models of character distribution based on probabilistic (Ponte and Croft, 1996; Dai et al., 1999) and information theoretical assumptions (Lua and Gan, 1994; Teahan et al., 2000). Segmentation will not be part of this work, since pre-segmented text is assumed as input (see Section 1.2).

The discovery of MWUs is the main step of the tokenization process (Zhang et al., 2001). Again, knowledge-free and knowledge-driven approaches have been developed to solve this task. Knowledge-driven approaches can be subdivided into two categories dubbed syntatic and hybrid approaches. Syntactic approaches use linguistic patterns to extract MWUs (Hearst, 1992). For example, LEXTER (Bourigault et al., 1996) uses an extensive list of predefined syntactic patterns to segment sentences in their components and identify potential nominal phrases and collocations. Furthermore, this tools utilizes a list of nouns that use given prepositions as complements to filter the initial segmentation results. By applying a learning approach on the corpus, LEXTER is then able to improve the results it achieves. A similar, yet semi-automatic strategy, is implemented in Termight (Dagan and Church, 1994). The algorithm implemented in the tool is based on the assumption that the frequency of syntactic patterns are correlated with their relevance. Hence, it performs MWU extraction by extracting the most frequent combination of syntactic patterns and applying a frequency analysis on the terms which match these patterns. The overall drawback of purely syntactic approaches is their restriction to a specific language due to the patterns they necessitate.

Hybrid approaches improve the flexibility of knowledge-driven approaches. They combine syntax and statistics for MWU extraction, either by first applying a numerical preprocessing to detect potential MWUs and pruning the resulting list by using linguistic patterns like XTRACT (Smadja, 1993) or by processing the input in the reserve order, first using syntactic patterns and subsequently filtering the results by using numerical models (Justeson and Katz, 1991). Still, they have the same restrictions as syntactic approaches, since they are also language-specific.

Most knowledge-free approaches use probabilistic metrics (e.g., the occurrence frequency (Giuliano, 1964), the symmetrical conditional probability (Ferreira da Silva and Pereira Lopes, 1999) or the dice formula (Dice, 1945)) to compute the significance of collocations (Schone and Jurafsky, 2001; Dias, 2002). Schone (2001) proposes an approach based on Latent Semantic Analysis to compute the semantic similarity of terms. He uses these similarity values to improve his scoring function during the MWU extraction. This technique shows some improvement, yet is computationally expensive. Another approach proposed later by Dias (2002) yields similar improvement and is computationally cheaper. Dias uses the distribution of patterns over positional word n-grams to detect MWUs. He defines a new metric called Mutual Expectation (ME). ME is suitable for detecting general-language MWUs, yet it presents weaknesses when it is used for detecting domain-specific MWUs because it does not model their specificity. An approach which takes into account the distribution of MWUs over the corpus is based on the term frequency-inverse document frequency (TF-IDF) metric (Salton and McGill, 1986) and implemented in the ontology extraction tool TextToOnto (Maedche and Staab, 2000) (see section 2.2.3). TF-IDF is based on the assumption, that highly frequent multi-word terms which appear in a small number documents are domain-specific. Yet, it is biased against high-frequent terms as it does not include the frequency of the constituents of terms in the computation. In Chapter 3 of this work, we present a metric that takes the characteristics of domain-specific MWUs into account and evaluate it against the current metrics.

Knowledge-free MWU extraction techniques mostly return ordered list of n-grams. The subsequent extraction of relevant n-grams out of this list will be called lexicon extraction (other names found in relevant literature include terminology extraction and vocabulary extraction (Manning and Schütze, 1999)). Knowledge-driven approaches for lexicon extraction depend on human input in the form of seed terms and language resources for the detection of domain-specific terms (Dias, 2002). Most approaches designed especially for lexicon extraction do not try to discover domain-specific languages. Rather, they expand existing lexica. For example, Hersh et al. (1996) use a set of ten manually selected seed words to extract new terminology on pain from medical reports. They achieve this goal by retrieving all words and word patterns that are syntactically similar to their input seeds. Moldovan et al. (2000) propose the manual validation of an automatically generated list of term candidates to complete the selection of n-grams. Bodenreider et al. (2002) combine pattern matching and dictionary search on phrases containing adjectival noun modifiers to discover new bio-medical terminology. Wermter and Hahn (2005) define a threshold for the frequency of multi-words manually. Knowledge-free approaches use mostly local information on the distribution of MWUs to extract relevant n-grams. Especially, Dias et al. (1999a) select n-grams generated using a purely statistical score by computing whether their cohesiveness is higher than that of the (n-1)-grams they contain and of the (n+1)-grams containing them. In Chapter 4, we propose the use of the paradigmatic cohesiveness of domain-specific words to generate domain-specific lexica.

2.1.2 Conflation

Conflation refers to the mapping of non-identical terms to a single one (Frakes, 1984). The second category of tools for preprocessing, the stemmers, achieve this goal by reducing the terms detected in the corpus to their morphological root. This process is called stemming. For example, the words *humanity* and *humanoid* have the same canonical form (i.e., the same stem) *human*. By these means, stemmers try to tackle the data sparseness that can occur when retrieving seldom word forms.

Several approaches to stemming have been proposed. The knowledge-driven approaches fall into two main categories: dictionary and linguistic approaches (Al-Sughaiyer and Al-Kharashi, 2004). Dictionary-based (also called lookup) approaches are the simpler of both approaches. They use a stem dictionary for the identification of possible word stems (Frakes and Baeza-Yates, 1992). Many linguistic approaches use a combination of manually defined rules and longest match for the detection of stem boundaries (Lovins, 1968; Porter, 1980). A minimalistic approach belonging to this category are S-stemmers for English (Frakes and Baeza-Yates, 1992), which delete the ending letter "s" to transform the plural forms into singular. Porter's stemmer (Porter, 1980), the most widely used stemmer for English, uses transformation rules on word endings for reduction. Other linguistic stemmers model morpheme translations by using final-state automata such as Hidden Markov Models (Melucci and Orio, 2003). Several stemmers for languages other than English (such as Arabic (Al-Sughaiyer and Al-Kharashi, 2004), French (Savoy, 1999), Slovenian (Popovic and Willett, 1992) etc.) have been developed over the past two decades.

Yet, they are language-specific and fail to cover certain morphological phenomena in other languages.

Knowledge-free approaches try to remedy the weaknesses of the knowledge-driven approaches by abstaining from using dictionaries or transformation rules. Stemmers based on n-grams use character sequences to compute the similarity of words (Adamson and Boreham, 1974; Kosinov, 2001). For example, Schone (2001) uses prefix trees to extract prefixes and suffixes by inserting words from left to right and from right to left. These approaches work well for languages that present mainly affix construction as morphological phenomena. Yet, languages such as Hebrew allow the alteration of all vowels in a noun depending on the context. Hence, they are can hardly be processed by this model. More recently, (Hammarström, 2006) proposed a poor man's approach to the recognition of same-stem words. Yet, this approach can only capture affixes. Therefore, it has the same restrictions as the other approaches with respect to complex morphological phenomena that occur in morphology-driven languages. A more complete overview of approaches to conflation can be found in (Bordag, 2007). Since current knowledge-free approaches to morphology are unable to handle all categories of morphological uses[1], we will not apply any morphological preprocessing to the corpora at hand. Nevertheless, morphological analysis could be used in the post-processing on the concept extraction results, given that a clustering of semantically related terms can allow an effective categorization of morphological phenomena in the language at hand. This will be the object of further research and not included in this work.

2.1.3 Categorization

Categorization designates the process of grouping objects based on similar properties (Wanas et al., 2006). In the context of pre-processing, categorization can be carried out on several levels. On the morphological level, lemmatizers aim at tagging the entries in a corpus with the corresponding dictionary-form, also called lemma (Perera and Witte, 2005). Approaches to lemmatizing combine lexicon-based (Lezius et al., 1998; Perera and Witte, 2005; Al-Shammari and Lin, 2008) and rule-based (Paulussen and Martin, 1992; Kettunen, 2006) techniques. Overall, all lemmatizers demand either some morphological analysis and/or background knowledge on the language to analyze. They will not be further considered, since they are not knowledge-free (Bordag, 2007).

Part-of-speech (POS) taggers process text at the syntactic level and have a function similar to that of lemmatizers. Yet, instead of marking each word with its non-inflected form, POS-taggers assign a code to each lexical unit in a text that

[1]See (Schone, 2001) for a list of morphological phenomena

indicates their POS, i.e., the syntactic category to which it belongs (noun, adverb, etc.). Several categories of approaches to POS-tagging can be differentiated. The most common include machine learning (Brill, 1995; Ratnaparkhi, 1996), stochastic (Cutting et al., 1992; Leech et al., 1994) and more recently morphology-based (Tlili-Guiassa, 2006; Singh et al., 2006) approaches. Standard sets of POS-tags such as the UPenn (University of Pennsylvania) TreeBank tag set (also called Penn TreeBank (Santorini, 1990)) have a magnitude of approximately fifty tags. Similarly to lemmatizers, POS-taggers demand background knowledge on the language to process and are thus knowledge-driven.

The discovery of tokens is a crucial preprocessing step for the extraction of terms. As stated in the premises of this work, a segmented text corpus is presupposed. We will thus focus on the extraction of terms in the form of MWUs as tokenizing step. Since both stemmers and lemmatizers are purely language-specific and demand manual input in the form of training data, they will not be considered further in this work. This is also valid for POS taggers, which necessitate explicit knowledge on the structure of the language to process, usually in the form of manually generated training data.

2.2 Concept Extraction

The main goal of concept extraction is the extraction of semantically similar terms out of a data corpus. Approaches to ontology extraction can be categorized by a variety of dimensions including units processed, data sources and knowledge support (Zhou, 2007). The overview of techniques for concept extraction presented in this section focuses on the knowledge support dimension. Accordingly, we differentiate between two main categories of approaches to concept extraction, namely knowledge-rich and low-bias approaches. Knowledge-rich approaches use knowledge about the structure of the data sources to process. Especially, text-based approaches include knowledge such as phrase structure, lemmas and POS to extract nouns or noun phrases as units to process (Biemann, 2005). Therefore, they are subject to the same limitations as the other knowledge-driven approaches presented in Section 2.1. The category of knowledge-rich approaches includes supervised machine learning techniques and clustering techniques based on knowledge-rich features (Omelayenko, 2001). Low-bias (also called knowledge-lean (Zhou, 2007)) approaches do not use a-priori knowledge on the language to process. Rather, they make use of statistical features to extract the features of the terms which compose a concept. Clustering techniques based on low-bias features are the main constituent of this category of approaches. Since this work focuses on unsupervised approaches, we will be mainly concerned with clustering techniques.

2.2.1 Approaches Based on Clustering

The basic idea behind approaches based on clustering is to generate a hierarchy of concepts based on information from a text corpus or other sources. The concepts in such hierarchies are mostly linked by the same relation (e.g., hyperonymy, hyponymy, part-of) (Biemann, 2005). Approaches based on clustering can be differentiated based on the type of features on which they are based. The two main categories of features that can be found in the literature are the linguistic and the statistical features (Zhou, 2007).

Approaches based on linguistic features (also called linguistic patterns) utilize known syntactic patterns such as Hearst patterns (Hearst, 1992) to extract semantic similarity[2]. One of the earliest works in this area was carried out by Hindle (1990). Hindle uses predicate-argument structures to extract similarity values for nouns and verbs. He first uses a deterministic parser to analyze the structure of the phrases in the corpus. Based on the resulting syntax analysis tree, the correlation of nouns and verbs is computed by using the mutual information metric. The resulting similarity matrix can be then used for further processing, especially for clustering. In Caraballo (1999), a labeled noun-hierarchy is built from text using bottom-up clustering. After the capture of appositives and noun phrases, Caraballo uses a stemmer to extract the distinct nouns appearing in the data corpus. The similarity between the co-occurrence vectors representing each of the previously extracted nouns is subsequently computed using the cosine metric. The single nouns are then merged to clusters agglomeratively. A label is assigned to each of the clusters based on the association between hypernyms extracted using Hearst-patterns and the noun tree. A similar approach is used by Cimiano and Staab (2005), with the slight difference that their technique integrates the hypernyms into the clustering. Another clustering approach is described by Pantel and Lin (2002). First, they use the MiniPar (Lin, 1998) to extract phrase structure out of text. Then, they utilize the verb-object and verb-subject relation to cluster terms into classes of semantically similar words. The clustering of the words is carried out by using the Clustering By Committee (CBC) algorithm (Pantel, 2003). One of its main features is its ability to disambiguate polysemantic terms, e.g., orange as a color and orange as a fruit. Approaches based on linguistic patterns are knowledge-driven, since they require knowledge on the structure of the language at hand. For this reason, they will not be considered further in this work.

[2] A survey on pattern extraction can be found in (Muslea, 1999).

Statistical features are based on two assumptions: first, Firth's assumption (Firth, 1957), which states that the semantics of words are specified by the words with which they collocate. Second, Harris's distributional hypothesis (Harris, 1968), which states that words which tend to appear in similar context are similar as well. Based on these hypotheses, several categories of features for the characterization of words have been developed, of which the most important include collocation-based and window-based features. Window-based and collocation-based features measure the similarity between the collocations of terms to determine the similarity of these terms. An early work on the use of collocation for measuring the degree of association of words is described by Church and Hanks (1989). A similar approach based on head modifiers and modifiers was implemented by Ruge (1992). For each term, the number of occurrences as head modifier/modifier of other terms is computed. The resulting vectorial descriptions are compared using the cosine metric. Schütze (1998) uses so-called word vectors to describe terms in a corpus. The word vector to each term consist of all its "close neighbors", i.e., of all the words which appear in the sentence or within a larger context (e.g., a document (Qiu and Frei, 1993)). To reduce the dimensionality of the resulting word space, Schütze (1998) uses Latent Semantic Analysis (LSA) (Deerwester et al., 1990). Then, he uses the cosine metric to measure the correlation between the term descriptions. Sanderson and Croft (1999) use collocations to derive a concept hierarchy from a set of documents. They define a subsumption relation by stating that a term t subsumes a term t', when t appear in every document in which t' appears. Using this subsumption relation, Sanderson and Croft (1999) computes a term hierarchy automatically. Khan and Luo (2002) propose a technique that generates concept hierarchies out of document hierarchies. The first step of this technique consists of selecting documents from the same domain. Then, a hierarchy of document clusters is generated by using the SOTA-Algorithm (Dopazo and Carazo, 1997). A keyword matching a Wordnet-concept is then assigned bottom-up to each cluster of the hierarchy. First, a concept representing the typical content of the documents of each leaf node is assigned to the node. In a second step, the labels of the interior nodes are assigned by using hypernyms of their children. This method does not allow the determination of the relations that exist between linked nodes. In Bisson et al. (2000), the Mo'K Workbench, a tool designed especially to support the development of conceptual clustering methods for ontology building is described. Approaches based on hybrid features combine statistical and linguistic features (Zhou, 2007).

2.2.2 Other Approaches

Most other approaches to concept extraction use natural language processing and machine learning techniques. Techniques based on natural language processing use knowledge-rich tools in the background to analyze sentence structure parsing and the extraction of noun phrases.

One of the first techniques in this area was developed by Hearst (1998). Hearst considers a concept as being a term with semantic relevance. Hearst's approach to harvest semantical relations between concepts consists of two steps. First, concepts related by a given relation are retrieved from an existing ontology. Then, word patterns that express this relationship are extracted from a text corpus. Based on these patterns, Heart's algorithm computes new relationships between existing concepts and retrieves new concepts related to existing concepts. This method demands a large amount of background knowledge.

Moldovan and Girju (2001) developed a method aiming at discovering domain-specific concepts and relationships to extend an existing ontology. The input for this method consists of a domain-specific text corpus, linguistic resources such as dictionaries and a set of seed concepts given in by the user. The first automatic step of this method computes a synset (Miller, 1990) out of each seed concept and its synonyms. Then, all nominal phrases in the corpus that contain one or more of the seed concepts or synonyms terms are retrieved. After a POS-tagging and parsing phase, new concepts are extracted and validated by the user. The discovery of new relation instances is carried out by finding lexico-syntactic patterns that involve the new concepts and are characteristic for a certain relation. The validation of the relations is carried out by the user.

Another technique based on NLP was developed by Missikof et al. (2002). The method consists of three steps. First, high-frequent and specific terms and term combinations are extracted from a given text corpus. Second, a sense disambiguation process is triggered. The goal of the disambiguation is to identify the semantic relations between the terms and term combinations extracted previously. Related terms are then merged to more complex concepts. Finally, the resulting taxonomy is integrated in a core ontology if one is available, or else in a pruned version of WordNet (Miller, 1990).

A method for the enrichment of ontologies based on semantic knowledge from the World Wide Web was developed by Faatz and Steinmetz (2002). After choosing the documents to use through an IR process, a set of candidate concepts similar to the concepts found in a core ontology is computed. For this purpose, a set of enrichment rules that do not interfere with the semantic distance information stored in the core ontology is used. The co-occurrences of the candidate concepts are added to the set of candidates and eventually presented to a domain expert, who decides on the concepts that are to be integrated into the existing ontology. A similar method based on quality labels is described by Hahn and Markò (2001).

Aussenac-Gilles et al. (2000) propose a generic method consisting in three main steps. First, a domain expert selects a corpus consisting of a set of technical documents. A selection criteria could be to select documents where terms from a domain-specific glossary occur. Second, adequate linguistic tools (like LEXTER for terminology extraction (Bourigault et al., 1996), Caméléon for relation extraction (Aussenac-Gilles and Seguela, 2000), etc.) are chosen. These are used to parse the text for domain terms, lexical terms and sets of synonyms. Finally, a semantic network is generated out of the data extracted from the text corpus in a third step.

Overall, current approaches to the extraction of domain-specific concepts are either knowledge-driven or are based on the results of knowledge-driven approaches such as lemmatizers and taggers. A further drawback of existing approaches to the extraction of domain-specific concepts lies in the fact that they do not take the possible polysemy of termini into consideration (Cicurel et al., 2006). Although polysemy is a limited phenomenon when dealing with domain-specific corpora (Bisson et al., 2000), the approach presented in this work will take this particular aspect of clustering into consideration. A more exhaustive enumeration of existing methods for concept extraction can be found in (Maedche and Staab, 2001; Omelayenko, 2001; Biemann, 2005; Zhou, 2007).

2.2.3 Tools for Ontology Extraction

Several tools have been developed for the extraction of ontologies. In this section, we epitomize the most popular of these tools, focusing especially on their use of external resources and the automation of the steps from the input data to the ontology.

ASIUM

ASIUM is one of the first ontology learning tools (Faure and Nédellec, 1999). It learns subcategorization frames and ontologies from text. To achieve this goal, it processes the input corpus in three steps named syntactic analysis, concept extraction and validation. The syntactic analysis is implemented by the tool SYLEX (Constant, 1995). SYLEX extracts instantiated subcategorization frames using manually defined patterns. Stop words and adjectives are automatically removed from the extracted frames. The output of SYLEX is then used by ASIUM, which selects all the head nouns occurring with the same verbs and prepositions to generate initial clusters. The concept extraction per se is based on the assumption that headwords occurring after similar prepositions and with similar verbs represent similar concepts. The conceptual clustering is realized by using a threshold-based bottom-up algorithm. This algorithm processes the clusters linearly and merges all clusters with a similarity below a manually set threshold sequentially. By these means, it generates a taxonomy. A post-processing step subsequently removes unuseful clusters. The validation step is carried out by the user, who is given the possibility to interactively correct the clustering and label the clusters. Figure 2.1 shows the architecture of ASIUM.

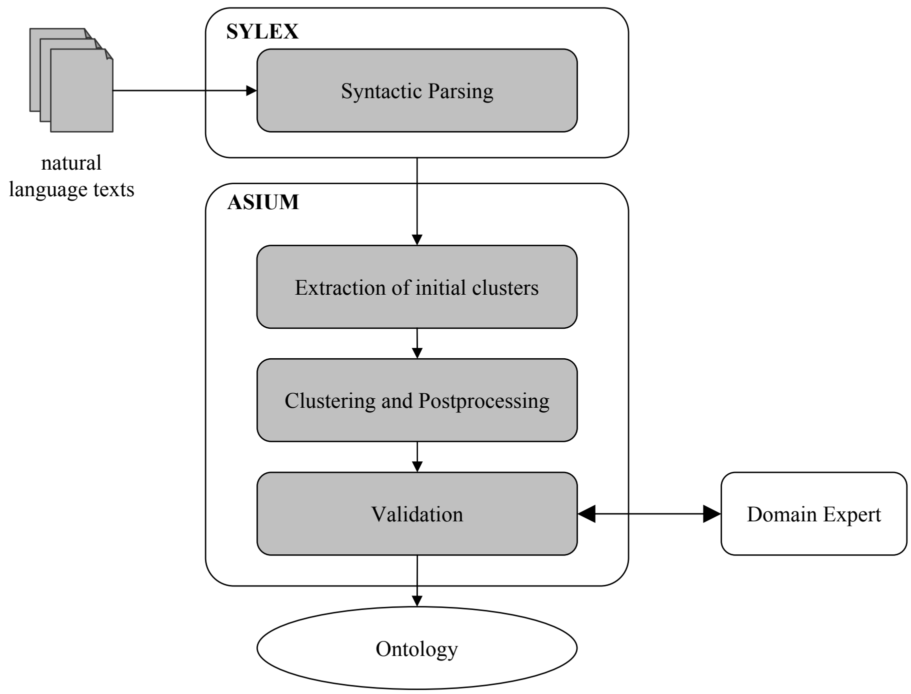


Figure 2.1: Architecture of ASIUM

The preprocessing step implemented in ASIUM is based on the functionality of SYLEX, which implements a knowledge-driven and language-dependent approach. Therefore, ASIUM is unable to extract domain-specific concepts in a low-bias fashion.

TextToOnto

TextToOnto is a tool for the semi-automatic extraction of ontologies out of text (Maedche and Staab, 2000). The extraction process proposed is subdivided in four main steps: preprocessing, concept extraction, relation harvesting and ontology pruning (see Figure 2.2).

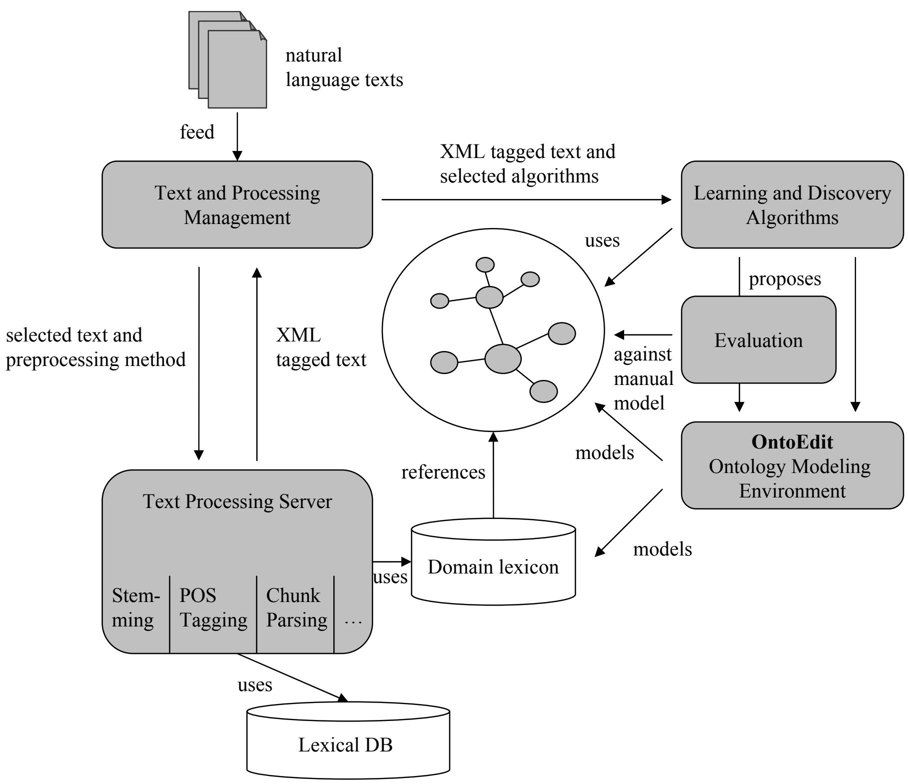


Figure 2.2: Architecture of TextToOnto

The preprocessing functionality of TextToOnto is implemented by two components of the tool. The first of these components is called the text and processing management component. It selects appropriate standard preprocessing tools implemented by the second preprocessing component, the text processing server, as well as learning and knowledge discovery techniques. The preprocessing of the input corpus is run on the server and can include POS-tagging, stemming, chunk parsing and term extraction. In addition, the preprocessing can make use of background resources such as domain-specific lexica. The preprocessed text is formatted in XML and forwarded to the learning and discovery component along with a selection of appropriate algorithms for concept acquisition and relation harvesting such as those described by Srikant and Agrawal (1995). The final results are then sent to an ontology engineering tool, which allows a knowledge engineer to manually alter the extracted ontology. In recent versions [3], most of the steps described above must be manually validated.

TextToOnto utilizes language-specific shallow parsing of text corpora for the extraction of terms. Furthermore, the tool relies on lexical and domain-specific databases. Therefore, it is unsuitable for ontology extraction from corpora written in languages for which reference data is not available. For example, version 1.0 can only analyze German, English and Italian. Furthermore, the concept extraction is based on syntactic patterns.

JATKE

JATKE provides a unified framework for ontology learning (see Figure 2.3), designed to be used by professional knowledge engineers. It allows the combination of plug-ins for ontology extraction, making it highly configurable. The plug-ins must suffice the input-output behavior defined by the internal, extensible JATKE ontology. They can thus implement any kind of approach, e.g., statistic, linguistic or hybrid. The design of the tool is based on three main principles (also called concepts (Endres, 2005)): containdness, integration and user interaction.

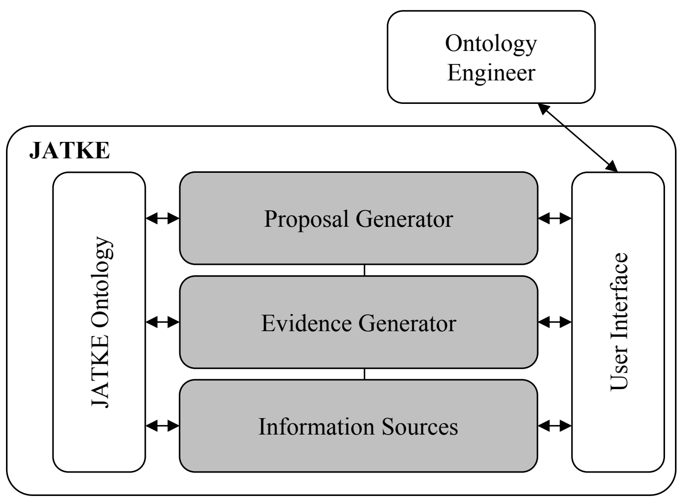


Figure 2.3: Architecture of JATKE

The core idea of the containdness principle is the regulation of the input and output behavior of JATKE plugins and the communication between them through an internal ontology. By these means, JATKE ensures that tools which were developed

[3] Referred here is version 1.0 from November 11^{th}, 2004.

independently can communicate with each other. Each of the events in JATKE is registered as an instance of the internal ontology, which itself is engrafted as a hidden tree in the ontology project.

Integration refers to the architecture of the system. The main aim during the development of JATKE was to create a system that allows the integration of arbitrary modules. The interpretability of all integrated modules is guaranteed by the integration of all information flowing during the ontology acquisition process in the JATKE ontology or its project-specific specializations. Therewith, each new module can always interpret the information produced by existing ones and vice versa.

Finally, the JATKE system is designed for heavy user interaction. Each of the results proposed by the modules at hand must be manually approved by the user. The parallel generation of proposals and their evaluation through the user ensures a fast integration of feedback in the proposal generation.

JATKE is designed for user-driven ontology extraction. Thus, it is unsuitable for the fully automatized extraction of concepts. Nevertheless, modules for low-bias concept extraction could be implemented and integrated in the tool.

OntoLT

OntoLT (Buitelaar et al., 2004) was developed as a text analysis tool for ontology extraction and extension. The extraction of ontologies with OntoLT is subdivided into three main steps (see Figure 2.4): linguistic annotation, definition of mappings and extraction.

The linguistic annotation is implemented by SCHUG, an integrated set of tools for the annotation of English and German (Declerck, 2002). SCHUG provides functionality for statistical POS tagging, morphological inflection and decomposition and pattern-based phrase structure analysis. The mapping rules are defined using precondition rules and operators. Precondition rules are expressed as constraints over the linguistically annotated corpus. The definition of precondition rules can be carried out either manually by a knowledge engineer or semi-automatically. The automatic generation of precondition rules is realized by using mapping rules for all linguistic annotations included in the results of a differential corpus analysis based on the χ^2-metric. When a term satisfies every constraint of an operator, this operator is activated and executes a predefined operation, e.g., adding candidates for slots, instances or classes to an ontology. The results of the operators have to be manually validated by a knowledge engineer before they are actually implemented by OntoLT. The extraction per se consists of the utilization of the predefined rules on the corpus at hand.

Like most other tools for ontology extraction, OntoLT implements a knowledge-

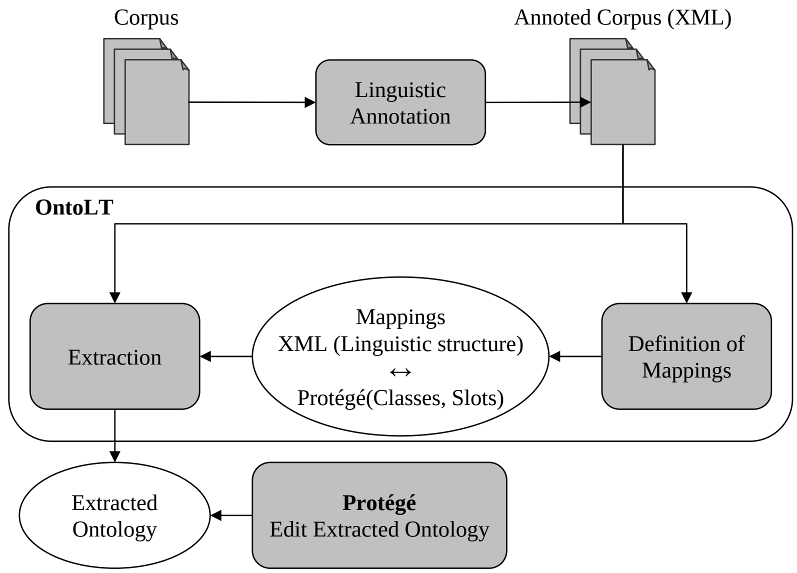


Figure 2.4: Architecture of OntoLT

driven approach, using reference corpora for differential analysis. The functionality of SCHUG is implemented using knowledge-driven techniques. Therefore, porting OntoLT to languages other that English and German would require a considerable amount of human effort.

OntoLearn

The idea behind OntoLearn is the reuse of general-purpose ontologies for domain-specific purposes (Cucchiarelli et al., 2004). It uses an existent ontology such as WordNet and transforms it into a domain-specific ontology by two means: the addition of domain-specific classes and pruning of irrelevant concepts. To achieve this goal, a domain-specific corpus is analyzed in three steps. First, the domain-specific terminology is extracted from the corpus at hand using a statistical comparative analysis based on contrastive corpora and glossaries. The second step aims at generating the compositional interpretation of extracted terms. For these means, a word sense disambiguation algorithm called SSI (Structural Semantic Interconnections, (Navigli and Velardi, 2004)) is utilized. Based on SSI's results, the relations between the domain-specific termini are extracted by using the reference ontology. The resulting terms and relations are organized in sub-trees and appended under relevant nodes in the ontology. SSI's results are also used to prune irrelevant senses of concepts from the ontology. Figure 2.5 shows the architecture of OntoLearn.

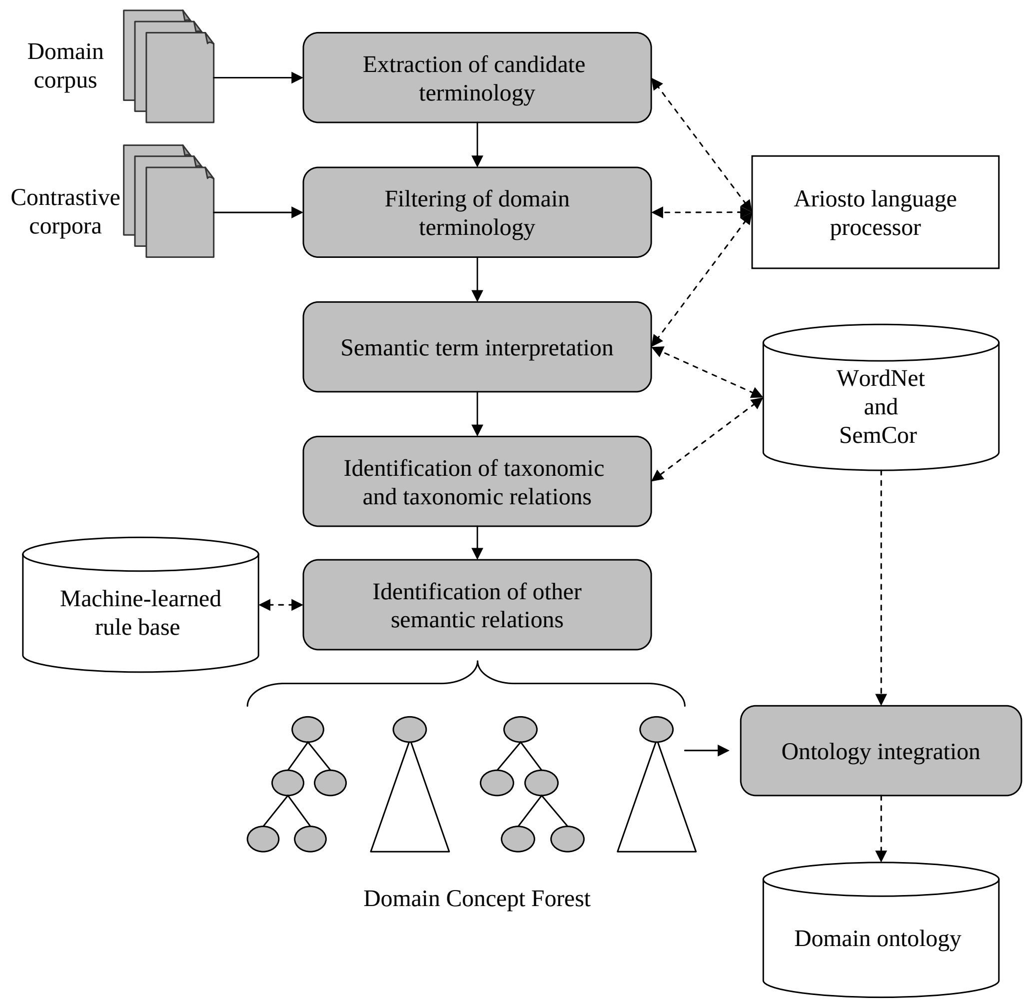


Figure 2.5: Architecture of OntoLearn

The approach implemented by OntoLearn depends heavily on external resources such as WordNet. The compositional interpretation of terms, which is the key step for the extraction of concepts, works solely for domain, which are not "highly technical (e.g., sports, tourism, etc.)" (Cucchiarelli et al., 2004, p. 1).

Overall, none of the tools presented uses exclusively low-bias techniques for concept extraction. ASIUM requires the results of SYLEX, which implements a knowledge-driven and language-dependent approach to syntactic analysis. The pre-processing components of TextToOnto use POS-tagging and stemming techniques, which are knowledge-driven. JATKE implements a framework for ontology learning that is based on current tools for ontology extraction. Thus, it does not implement any ontology extraction approach per se. Yet, it is designed to integrate approaches based on heavy user-interaction. OntoLT integrates the results of SCHUG, a set of tools that implement language-specific annotation. Furthermore, OntoLT uses differential analysis for terminology extraction, making it knowledge-driven. Finally, OntoLearn depends on external resources such as WordNet. It is therefore difficult to port for technical domains. Consequently, all the tools presented in this section are unsuitable for the low-bias extraction of concepts.

2.3 Graph Theory

Since graph algorithms will play an important role in this thesis, it is relevant to give a short overview of the terminology used in the domain of finite graphs, with which we will be dealing exclusively. The terminology used here is defined in accordance with (Diestel, 2005). Two set-theoretical concepts are necessary for the understanding of the considerations presented in this section. First, we define the set

$$[V]^k = \{V' \subseteq V : |V'| = k, k \in \mathbb{N}\} \tag{2.1}$$

as the set of subsets of a set V that have exactly k elements. Second, we define a *partition* of a set V as a set $p(V)$ of subsets $V_{i,i=1...n}$ such that

$$\forall V_i, V_j \in p(V),\ V_i \neq V_j \rightarrow V_i \cap V_j = \emptyset \tag{2.2}$$

and

$$\bigcup_{V_i \in p(V)} V_i = V. \tag{2.3}$$

Graphs are natural structures which appear in several domains such as sociology (social networks), telecommunication (computer networks) and biology (biological networks) (Dorow, 2006). In its simplest form, a graph G is a pair $G = (V, E)$ such that:

- V is the set of *vertices* (or *nodes*) of G. The set of vertices of a graph G will also be referred to as $V(G)$ independently from the symbol used in its signature. Thus, for the graph $X = \{Y, Z\}$, $V(X) = Y$. The *order* of a graph G (denoted by $|G|$) is the number $|V|$ of its vertices.

- $E \subseteq [V]^2$ is the set of *edges*. For each edge $e = \{u, v\}$, the vertices u and v will be called its *ends* and e will be said to *join* u and v. The set of edges of a graph G will also be referred to as $E(G)$ analogously to the set of vertices. For the sake of unambiguity, it shall always be assumed that $V \cap E = \emptyset$.

The *empty graph* is a graph with $|V| = 0$. A *finite graph* is a graph such as $|G| < |\mathbb{N}|$.

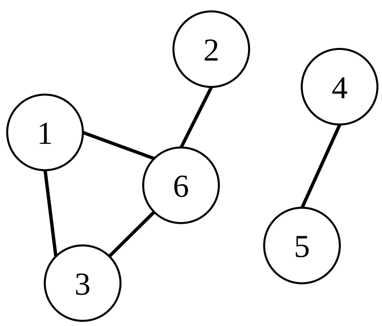
Graphs can be visualized by picturing their vertices as circles or rectangles containing their names and their edges as curves joining these geometric figures (see Figure 2.6). From an algebraic point of view, graphs can be characterized through their adjacency matrix, which will be denoted by $A(G)$. An entry a_{ij} $(i, j \in V)$ of $A(G)$ is 1, when i and j are joined. In any other case, $a_{ij} = 0$. Formally:

$$A(G) = (a_{ij})_{i,j \in V} \; with \; a_{ij} = \begin{cases} 1 & \text{if } \{i, j\} \in E \\ 0 & \text{else.} \end{cases} \tag{2.4}$$

The adjacency matrix of undirected graphs is always symmetric, i.e.,

$$\forall i, j \in V, a_{ij} = a_{ji}. \tag{2.5}$$

Figure 2.6 depicts an undirected graph and its adjacency matrix. The graph displayed consists of the set of nodes $V = \{1, 2, 3, 4, 5, 6\}$ and the set of edges $E = \{\{1, 3\}, \{1, 6\}, \{2, 6\}, \{3, 6\}, \{4, 5\}\}$.



	1	2	3	4	5	6
1	0	0	1	0	0	1
2	0	0	0	0	0	1
3	1	0	0	0	0	1
4	0	0	0	0	1	0
5	0	0	0	1	0	0
6	1	1	1	0	0	0

Figure 2.6: An undirected graph and its adjacency matrix

Since graphs are not always symmetric, directed graphs were introduced. In directed graphs, the vertices are not sets of nodes but ordered pairs, depicting the origin and the destination of edges. Thus, $E \subseteq V \times V$. For an edge $e = (u, v)$ (also noted uv), u is called the *initial vertex* of e, while v is its *terminal vertex*.

Two vertices u, v ∈ V are adjacent iff[4]

$$\exists e \in E : e = vu \vee e = uv. \tag{2.6}$$

[4] if and only if

The graph $G' = \{V', E'\}$ is a *subgraph* of G (denoted by $G' \subseteq G$) iff

$$E' \subseteq E \wedge V' \subseteq V. \tag{2.7}$$

An *induced subgraph* G' of G is graph such as

$$\forall \{u, v\} \subseteq V(G'), (u, v) \in E \leftrightarrow (u, v) \in E'. \tag{2.8}$$

$V(G')$ is said to *span* G' in G. The *spanning subgraph* G' of G will be denoted by $G' := G[V']$. Given a set of edges V'

$$G - V' = G[V \backslash V']. \tag{2.9}$$

For a set of edges $E' \subseteq E$,

$$G - E' = (V, E \backslash E') \tag{2.10}$$

and

$$G + E' = (V, E \cup E'). \tag{2.11}$$

$G' \in G$ is *node-maximal* with a given property if it fulfills that property but any G'' such that $V(G') \subset V(G'')$ does not fulfill that property. Similarly, $G' \in G$ is *edge-maximal* with a given property if it fulfills that property but any G'' with $E(G') \subset E(G'')$ does not fulfill that property.

A *path* in G is a non-empty subgraph $P = (\{v_1, ..., v_n\}, \{(v_i, v_{i+1})_{i \in \{1 \dots n-1\}}\})$ of G. The vertices v_2 ... v_{n-1} are called *inner vertices*, v_1 and v_n are the *beginning* and *end* of P respectively. $|E(P)|$ is the *length* of a path P. A path of length k will be denoted by P^k. G is *connected* when each of its nodes can be reached from any other node, i.e., when for all nodes $u, v \in V(G)$ a path $P \in G$ exists such that u is the beginning of P and v its end. Graphs, which do not fulfill this criterion are called *disconnected*.

A *component* of a graph is a node-maximal and edge-maximal connected subgraph of this graph. The example in Figure 2.6 is composed of the two components $G[\{1, 2, 3, 6\}]$ and $G[\{4, 5\}]$. The union of all disjunct components of a graph is the graph itself. Components are never empty, since they are connected. Thus, the empty graph has no components. A graph is called *complete* (or a *clique*) when all its nodes are pairwise connected, i.e.,

$$\forall u, v \in V \ \exists e \in E : e = (u, v). \tag{2.12}$$

Figure 2.7 depicts undirected complete graphs with three, four and five nodes.

The *degree* (or *valency*) $d(v)$ of a node $v \in V(G)$ is the number of vertices with which it is connected:

$$d(u) = |\{v \in V(G) : \exists e \in E : e = (u, v)\}|, u \in V(G). \tag{2.13}$$

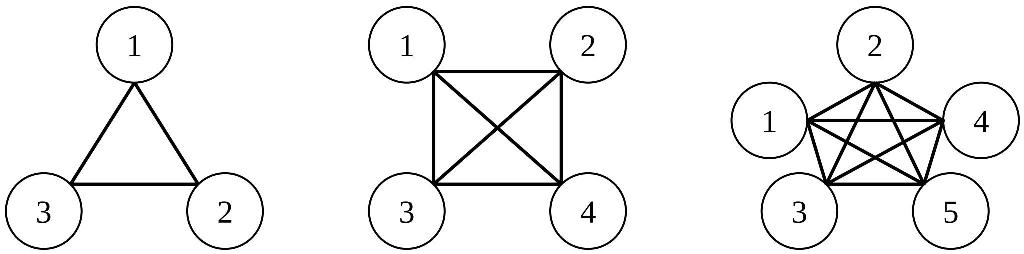


Figure 2.7: Complete graphs with 3, 4 and 5 nodes (from left to right)

The *minimum degree* of a graph G is $\delta(G) = \min_{v \in V(G)} d(v)$. Analogously, the maximal degree of G is $\Delta(G) = \max_{v \in V(G)} d(v)$. A graph is called *k-regular* if all its vertices have the same degree $k \in \mathbb{N}$. A graph is called *r-partite* if a partition of size r of its vertex set exists, such that each edge has its ends in different partitions. 2-partite graphs are called *bipartite*.

In order to be able to describe more complex phenomena, weighted graphs were introduced. A *weighted graph* is a triplet $G = (V, E, \omega)$ where $V = V(G)$ and $E = E(G)$ are as defined previously, and

$$\omega : E \rightarrow \mathbb{R} \tag{2.14}$$

is a function which assigns real weights to the edges of the graph. Weighted graphs are widely used in NLP because they allow the accurate description of a broad range of phenomena such as probabilistic transitions and co-occurrence graphs (Manning and Schütze, 1999), which can not be described with non-weighted graphs. Weighted graphs can be represented using a weight matrix W(G) defined as follows:

$$W(G) = (a_{ij})_{i,j \in V} \textit{ with } a_{ij} = \begin{cases} \omega(ij) & \text{if } (i,j) \in E, \\ 0 & \text{else.} \end{cases} \tag{2.15}$$

Two further categories of graphs called multigraphs and hypergraphs were introduced to generalize the idea of graphs. *Multigraphs* generalize the idea of a graph by allowing more than one edge between two nodes. A multigraph is thus an ordered set $M = (V, E)$ that consists of a set of nodes V and multiset of edges $E = (E', \mu)$ where E' is the underlying set of elements of E and the function

$$\mu : E' \rightarrow \mathbb{N}^{+} \tag{2.16}$$

is the multiplicity function of E.

Hypergraphs extend the idea of graphs by allowing edges between more than two nodes. Formally, a hypergraph is an ordered set $H = (V, E)$ where V is the set of nodes and $E \subseteq \wp(V)$.

For a more thorough analysis of graph theory, the reader is referred to (Diestel, 2005).

2.4 Data Clustering

Clustering is generally defined as unsupervised classification (Jain et al., 1999). The goal of clustering algorithms is to determine subsets of data that are somehow similar. Generally, a clustering algorithm consists of five steps, namely pattern representation, definition of a similarity (resp. distance) measure, clustering per se, data abstraction (if necessary) and output evaluation (Jain and Dubes, 1988; Jain et al., 1999; Duda et al., 2001). In this section, we present a overview of pattern representation and distance metrics. Clustering algorithms and output evaluation are presented in subsequent sections.

2.4.1 Pattern Representation

Patterns are usually represented as feature vectors. These vectors consist of quantitative or qualitative values depending on the feature to describe. Quantitative values include both numerical (e.g., 1, 2, 3) and binary (true or false) feature values. Qualitative (also called categorical) values are either nominal (e.g., blue, white, red, ... for the feature color) or ordinal values (e.g., boxing weight divisions such as lightweight, super lightweight, welterweight, super welterweight, junior middleweight, ...). A mixture of different value types is possible. For example, a feature vector describing a word in a corpus by its frequency (numerical), its syntactical class (nominal) and whether or not it can be found in a corpus (boolean). A pattern can be represented as a vector ω in a feature vector space Γ of dimension n. A data set of m patterns is then a $n \times m$-Matrix.

Features do not hold the same amount of information. For the sake of space and time complexity reduction, the subset of features yielding the highest amount of information is usually used for clustering. The process of selecting this subset of features is called feature selection. Another pre-processing step, called feature extraction, consists of transforming the feature vectors to obtain more salient features. The feature selection and extraction processes can have a crucial impact on the clustering quality (Duda et al., 2001).

2.4.2 Similarity Measures

Similarity (resp. dissimilarity) measures are used to express how close (resp. different) feature vectors or clusters are. The similarity (resp. dissimilarity) of two patterns $\omega = (\omega_1, ..., \omega_n)$ and $\omega' = (\omega'_1, ..., \omega'_n)$ can be defined in various ways depending on the features contained in their feature vectors.

Numerical Features

Dissimilarity measures are most commonly used when processing numerical features. The distance between fully numerical vectors is usually measured by using instances of the Minkowsky-distance d_{Min}:

$$d_{Min}(\omega, \omega') = \sqrt[p]{\sum_{i=0}^{n} |\omega_i - \omega'_i|^p}. \tag{2.17}$$

The most prominent instances of d_{Min} metric include the euclidean distance d_{Euc} and the Manhattan distance d_{Man} (Jain et al., 1999):

$$d_{Euc}(\omega, \omega') = \sqrt{\sum_{i=0}^{n} (\omega_i - \omega'_i)^2}; \tag{2.18}$$

$$d_{Man}(\omega, \omega') = \sum_{i=0}^{n} |\omega_i - \omega'_i|. \tag{2.19}$$

One of the well known problems that occur when using a Minkowsky metric is that large-scale features often dominate the others. A solution to this problem consists of using the squared Mahalanobis distance d_{Mah} (Mahalanobis, 1936):

$$d_{Mah}(\omega, \omega') = (\omega - \omega')\Sigma^{-1}(\omega - \omega')^T, \tag{2.20}$$

where Σ is the sample covariance matrix or the correlation matrix of the pattern generation process. Other weighing schemes include probabilistic techniques such as nonlinear accuracy weighing (Cha et al., 2005) and machine learning approaches based on Case Based Reasoning (CBR) models (Stahl, 2005).

One prominent distance measure used especially in IR (Salton et al., 1975; Wilkinson and Hingston, 1991) and NLP (Manning and Schütze, 1999; Fleischman and Hovy, 2003) is the cosine metric d_{Cos}:

$$d_{Cos}(\omega, \omega') = \frac{\sqrt{\sum\limits_{i=0}^{n} (\omega_i - \omega'_i)^2}}{\sqrt{\sum\limits_{i=0}^{n} {\omega_i}^2} \times \sqrt{\sum\limits_{i=0}^{n} {\omega'_i}^2}}. \tag{2.21}$$

Additionally, specialized metrics which take the neighborhood of the input data into account have been developed. An example of such a metric is the mutual neighborhood distance MND (Krishna and Krishna, 1978). It defines the neighborhood of two patterns ω and ω' based on the neighborhood number NN of the two patterns:

$$NN(\omega, \omega') = n \text{ iff } \omega' \text{ is the n}^{th} \text{ nearest neighbor of } \omega. \tag{2.22}$$

Based on this definition, Krishna and Krishna (1978) define $MND(\omega, \omega')$ as follows:

$$MND(\omega, \omega') = NN(\omega, \omega') + NN(\omega', \omega). \tag{2.23}$$

Binary Features

In most cases, the similarity between binary vectors is computed by using measures based on set theory. These measures include the Dice and the Jaccard coefficients (Pantel, 2003; Theodoridis and Koutroumbas, 2006). Let $s(\omega)$ be the set defined as follows:

$$s(\omega) = \{i \in \mathbb{N} : \omega_i = 1\}, \tag{2.24}$$

where 1 stands for $true$ and 0 for $false$.

The Dice coefficient d_{Dic} (Dice, 1945) normalizes the size of the intersection of the sets $s(\Omega)$ and $s(\Omega')$ by using the total size of the input patterns:

$$d_{Dic}(\omega, \omega') = \frac{2|s(\omega) \cap s(\omega')|}{|s(\omega)| + |s(\omega')|}. \tag{2.25}$$

On the other hand, the Jaccard coefficient (Jaccard, 1901) normalizes the intersection of $s(\Omega)$ and $s(\Omega')$ by using the number of elements contained in the union of both sets:

$$d_{Jac}(\omega, \omega') = \frac{|s(\omega) \cap s(\omega')|}{|s(\omega) \cup s(\omega')|}. \tag{2.26}$$

Other measures consider both positive and negative matches. For example, the Hamming distance d_{Ham} (Hamming, 1950) counts the number of entries in which the input patterns differ:

$$d_{Ham}(\omega, \omega') = \sum_{i=1}^{n} |\omega_i - \omega'_i|. \tag{2.27}$$

A good overview of binary distance measures can be found in (Cha et al., 2005).

Categorical Features

Categorical features fall under two categories: nominal and ordinal. *Nominal features* are categorical features without a notion of order (e.g., word categories). They can be transformed into binary values. Therefore, distances between features vectors of such type can be measured using the coefficients described above. *Ordinal features* (i.e., academical degrees) can be assigned to numerical values. Thus, distances between vectors containing such values can be computed using measures for numerical values.

For a more detailed review on metrics used for clustering, the reader is referred to (Jain et al., 1999; Theodoridis and Koutroumbas, 2006; Deza and Deza, 2006).

2.5 Clustering Algorithms

Clustering algorithms can be differentiate according to their category and specific properties associated with their category. In general, clustering algorithms fall under three main categories (Pantel, 2003) dubbed partitional, hierarchical and hybrid algorithms. Partitional algorithms produce a single partitioning of the data by optimizing a given criterion. Hierarchical algorithms generate a partitioning of the data by merging or splitting clusters according to a given distance measure. Hybrid algorithms combine aspects of partitional and hierarchical algorithms. Furthermore, clustering algorithms can be distinguished by five main properties (Jain et al., 1999):

- *Divisive vs. agglomerative*: These attributes apply to hierarchical and some hybrid algorithms. Divisive algorithms regard the initial data set as an initial cluster. They generate clusters by iteratively splitting the set of clusters according to a given criterion (e.g., maximal inner distance). The split of clusters is carried out until each cluster consists of exactly one data point or a stopping condition is met. Agglomerative clustering algorithms process the input data in the exact opposite way. At the beginning, every data point is regarded as a cluster. According to a given merging strategy (e.g., maximal similarity), the clusters are merged until all points are in one cluster or a certain stopping condition is met.

- *Hard vs. soft*: An algorithm is called hard when it assigns each pattern to exactly one cluster. Soft algorithms can assign data points to more than one cluster. In this case, a membership function is then assigned to each data point. This function states to which degree the data point belongs to a cluster.

- *Incremental vs. non-incremental*: Most clustering algorithms were not designed to work with large data sets. Incremental algorithms are usually optimized versions of classical algorithms. They require less scans of the pattern set or reduce the number of patterns examined during the execution. Therefore, they can process larger input data sets.

- *Deterministic vs. stochastic*: Clustering techniques use stochastic approaches mainly in two computation steps: during the initialization step, in which the seeds for the clustering are determined and during the search step, in which optimized stochastic approaches are used to optimize the time complexity of the clustering process.

- *Monothetic vs. polythetic*: This pair of attributes is related to the number of features (i.e., of dimensions) considered simultaneously during the clustering process. Monothetic algorithms consider features sequentially. Most algorithms are polythetic, i.e., they cluster data by using all features simultaneously.

In the following, we epitomize clustering algorithms according to categories.

2.5.1 Partitional Algorithms

Partitional algorithms cluster data by generating a partition of the input (Berkhin, 2002). In most cases, these algorithms are initialized with the desired number of clusters K. Finding the best combination of K clusters is computationally expensive. Therefore, typical partitional algorithms are initialized randomly with a given number of seeds. They are subsequently ran a certain number of times. The best run is then given out as result.

The most prominent partitional algorithm, K-means, was developed by McQueen (1967). The algorithm works as follows:

1. Randomly compute K cluster centers μ_i with $1 \leq i \leq K$.

2. Classify the samples by assigning them to the nearest cluster center μ_i.

3. Recompute the cluster centers.

4. Repeat step 2 and 3 until a convergence criterion is met.

In most implementations of K-means, the convergence criterion is either no reassignment of the input patterns or a minimal decrease of the alteration of the total distance J between patterns and the centroid of their respective cluster:

$$J = \sum_{i=0}^{m} \sum_{j=0}^{K} \delta_{i,j} ||\omega_i - \mu_j||^2 \tag{2.28}$$

with

$$\delta_{i,j} = \begin{cases} 1 \text{ if } \forall j' \neq j ||\omega_i - \mu_j|| < ||\omega_i - \mu_{j'}|| \\ 0 \text{ else.} \end{cases} \tag{2.29}$$

The result of the standard K-means is a hard clustering of the input data (Duda et al., 2001). K-means can be modified to achieve a soft clustering (Bezdek, 1981). The $\delta_{i,j}$ values for soft clustering are computed as follows:

$$\delta_{i,j} = \frac{1}{\sum_{r=1}^{K} \left(\frac{||\omega_i - \mu_j||}{||\omega_i - \mu_r||} \right)^{\frac{2}{b-1}}} \tag{2.30}$$

where $b > 1$ is a free parameter to adjust the blending of different clusters. The degree of membership of a sample ω_j to the cluster with centroid μ_i is then given by

$$\frac{\left(\frac{1}{||\omega_j - \mu_i||} \right)^{\frac{2}{b-1}}}{\sum_{r=1}^{K} \left(\frac{1}{||\omega_j - \mu_r||} \right)^{\frac{2}{b-1}}}. \tag{2.31}$$

Variations of K-means include K-medoids (Kaufmann and Rousseeuw, 1987), which addresses the issue of better describing clusters by using one of the cluster elements to represent it and bisecting K-means (Steinbach et al., 2000), a divisive version of K-means. A good overview of further clustering algorithms can be found in (Jain et al., 1999; Berkhin, 2002).

2.5.2 Hierarchical Algorithms

Hierarchical clustering can be carried out in either an agglomerative or in a divisive fashion. The standard algorithm for agglomerative clustering is the AGglomerative NESting (AGNES) algorithm (Kaufmann and Rousseeuw, 2001). Let m be the number of patterns to cluster. AGNES can be depicted as follows:

1. Begin with m non-empty clusters containing different elements.
2. Merge the two most similar clusters.

3. Repeat step 2 until there is only one cluster left or a stopping condition is met.

The different implementations of AGNES vary in the measure they use to compute the distance between two clusters. The most common distances used for this purpose are the following:

- *Single-link clustering* computes the distance between two clusters ζ_1 and ζ_2 as the distance between the two nearest elements of the clusters:

$$d(\zeta_1, \zeta_2) = \min_{\omega_{1,i} \in \zeta_1,\ \omega_{2,j} \in \zeta_2} d(\omega_{1,i}, \omega_{2,j}). \tag{2.32}$$

  This distance measure nurtures the creation of elongated clusters.

- *Complete-link clustering* computes the distance between the most dissimilar elements of the clusters:

$$d(\zeta_1, \zeta_2) = \max_{\omega_{1,i} \in \zeta_1,\ \omega_{2,j} \in \zeta_2} d(\omega_{1,i}, \omega_{2,j}). \tag{2.33}$$

  Clustering with this measure leads to compact clusters.

- *Average-link clustering* computes the average distance between the elements of both clusters:

$$d(\zeta_1, \zeta_2) = \frac{1}{|\zeta_1||\zeta_2|} \sum_{i=1}^{|\zeta_1|} \sum_{j=1}^{|\zeta_2|} d(\omega_{1,i}, \omega_{2,j}). \tag{2.34}$$

  The clusters computed using this measure are similar to those computed using complete-link clustering (Han and Kamber, 2001) but are less sensible to outliers.

The standard divisive hierarchical algorithm is the DIvisive ANAlysis Clustering (DIANA) (Kaufmann and Rousseeuw, 2001). Divisive algorithms are not as popular as agglomerative because of their higher complexity (there are $2^{m-1} - 1$ splits of the original cluster into 2 clusters). Nevertheless, some heuristics can reduce their complexity to AGNES'. The basic DIANA algorithm looks as follows:

1. Begin with a single cluster containing all m elements.

2. Select the largest cluster ζ.

3. Find the element $\omega \in \zeta$ with the highest average dissimilarity to the other elements of ζ; ω is the first element of the splinter.

4. Find the element $\omega' \in \zeta$ that has the highest similarity to the splinter group; if the average similarity of ω' to the splinter is higher than its similarity to the remainder of the original cluster then add ω' to the splinter and go to step 4.

5. Repeat step 2 to 4 until all clusters contain a single element or a stopping condition is fulfilled.

DIANA can be extended to compute the distance between the potential elements ζ' of the splinter and the remainder of the cluster by using the single-link, complete-link or average-link distances as described in Eq. (2.32), (2.33) and (2.34).

Several variations of AGNES and DIANA can be found in literature (Jain et al., 1999; Berkhin, 2002; Theodoridis and Koutroumbas, 2006).

2.5.3 Hybrid Algorithms

Hybrid algorithms are multi-phase algorithms that combine partitional and hierarchical clustering techniques. Most of them were developed to address some of the drawbacks or to make advantage of known partitional or hierarchical clustering methods. For example, one of the earliest hybrid clustering algorithm combines the advantages of K-means and single-link clustering (Wong, 1982). In a first step, the algorithm generates a K-partition of the input data set by using K-means. Then, it computes the single-link distance matrix of neighboring clusters. Finally, the algorithm uses the K clusters as input for single-link clustering. The result of the clustering is a tree of dense clusters.

Another hybrid algorithm is *Buckshot* (Cutting et al., 1992). It was developed to address the initialization drawback of K-means. It applies average-link AGNES to a random set of $\sqrt{m}$ elements (m being the number of input patterns) to generate K clusters. Subsequently, BUCKSHOT uses the centroids of the resulting clusters as seeds for K-means. The output is a partition of the data set. For a low number of clusters the complexity of Buckshot is almost O(n) (Pantel, 2003).

BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies (Zhang et al., 1996)) addresses the problem of memory overload caused by certain clustering algorithms. It uses Cluster Feature (CF) trees as a compact representation of the clusters. Each CF of a cluster ζ is a triple that contains the number $|\zeta|$ of cluster elements, the linear sum $\sum\limits_{\omega\in\zeta}\omega$ of the cluster elements and the square sum $\sum\limits_{\omega\in\zeta}(\omega\cdot\omega)$ of the cluster elements. This information is sufficient to compute the centroid and the compactness of a cluster efficiently. Additionally, it enables to compute the distance between clusters. After having processed and stored the input data in a CF-tree, BIRCH can use any global clustering algorithm to carry out the clustering per se. (Pantel, 2003) gives a good overview of hybrid clustering algorithms.

2.5.4 Other Clustering Algorithms

There are several other families of clustering algorithms. Algorithms based on neural networks, especially self-organizing maps (SOM, also called Kohonen-maps) (Kohonen, 1989) use learning techniques to place patterns in the right cluster. Density-based algorithms like DBSCAN (Ester et al., 1996) discover high-density regions separated by low-density areas. Model-based clustering techniques (e.g., the Expectation Maximization algorithm (Dempster et al., 1977)) assume that the sample can be expressed by a mixture of distributions and use these characteristics of the data set to compute an appropriate clustering. The most relevant family of clustering algorithms for this work consists of the graph-based algorithms. Therefore, we epitomize this family of approaches in the following section.

2.6 Graph Clustering

Numerous variants of the definition of clusters in graphs (also called communities (Girvan and Newman, 2002; Flake et al., 2004)) are used in the literature (Edachery et al., 1999). Nevertheless, it is generally agreed upon that a subset of vertices forms a good cluster if the induced subgraph is dense but has a limited number of connections to the rest of the graph (Kannan et al., 2000; Kleinberg and Lawrence, 2001). Several categories of algorithms for graph clustering have been proposed throughout literature. These algorithms can be subdivided into two main groups (Schaeffer, 2007), namely global and local clustering.

2.6.1 Global Clustering

Global clustering techniques try to minimize or maximize a criterion on the whole graph (Newman, 2004). Similar to standard clustering algorithms, two main subclasses of global clustering algorithms exist: divisive and agglomerative clustering algorithms.

Divisive clustering algorithms partition the graph into clusters iteratively. The most common category of divisive graph clustering algorithms are the minimum cut algorithms (Elias et al., 1956; Schaeffer, 2007; Lang and Andersen, 2007). The fitness measures they utilize are based on the cut value defined between a set $S \subseteq V$ and its relative complement $V \backslash S$ in the set V of vertices of a graph $G = (V, E, \omega)$:

$$c(S, V \backslash S) = \sum_{v \in S, u \in V \backslash S} \omega(vu). \qquad (2.35)$$

The most common measure based on the cut value $c(S, V\backslash S)$ is the conductance $\Phi(S, V\backslash S)$:

$$\Phi(S, V\backslash S) = \frac{c(S, V\backslash S)}{min\{deg(S), deg(V\backslash S)\}}, \tag{2.36}$$

where

$$deg(S) = \sum_{v \in S} d(v) \tag{2.37}$$

is the sum of the degrees of the elements of S.

Common minimal cut algorithms use the structure of the input graph or transform the input graph to cluster it. For example, Condon and Karp (1999) use the underlying planted partition l-model underlying their input graph to cluster it into l groups of same size. Flake et al. (2004) present another divisive graph clustering method. Their clustering algorithm is based on inserting an artificial node, called sink, into the graph to cluster. The sink is then connected to all nodes in the graph. Subsequently, maximum flows between all nodes of the network and the sink are computed. The resulting flows are then used to compute a minimal cut of the graph. Other divisive clustering algorithms use the conductance or variations of the conductance of cuts to improve the quality of the clusters computed (Kannan et al., 2000). Finding a cut which minimizes the conductance is NP-hard. Thus, most algorithms based on conductance use heuristics of different kinds to approximate the best cut. For this purpose, they make use of subsets or topological characteristics of graph classes (Matula and Shahrokhi, 1990; Johnson et al., 1993). Other categories of global divisive clustering algorithms include techniques based on spectral analysis (Stoica and Moses, 1997; Gkantsidis et al., 2003) and on Markov chains (van Dongen, 2000).

Agglomerative approaches try to detect clusters by merging vertices in a bottom-up fashion. The choice of the vertices to merge is usually based on either topological or semantic similarity measures (Franti et al., 2006; Choo et al., 2007; Du et al., 2007). The basic approach to agglomerative graph clustering is known as the pairwise nearest neighbors method and consists of two steps. First, each vertex is put in a cluster. Then, the most similar clusters are iteratively merged to larger clusters until a stopping condition is met (e.g., a given number of clusters, (Schaeffer, 2007)). Using this technique demands the definition of a similarity measure for clusters (Franti et al., 2006). The simplest similarity measure on graphs is based on vertex similarity known as the neighborhood overlap measure as

$$\frac{\Gamma(u) \cap \Gamma(v)}{\Gamma(u) \cup \Gamma(v)}, \tag{2.38}$$

where u and v are nodes of the input graph and $\Gamma(u)$ (resp. $\Gamma(v)$) is the set of neighbors of u (resp. v).

Some of the more elaborated approaches to agglomerative graph clustering are tailored to cluster certain graph classes such as bi-partite graphs (Joseph et al., 2003) or sparse graphs (Harel and Koren, 2001; Clauset et al., 2004). The other agglomerative approaches try to maximize quality indexes on the clusters. For example, Clauset et al. (2004) maximize the modularity of clusters in general graphs and show that their method performs in quasi-linear time on sparse graphs. Donetti and Muñoz (2004) exploit spectral properties of the graph Laplacian matrix and combine it with hierarchical clustering.

The main drawback of global graph clustering algorithms lies in their space and time complexity, which results from them requiring the whole graph to generate an accurate clustering. Thus, they are unable to deal with large graphs such as the web graph (Schaeffer, 2007). Another drawback of global clustering algorithm is the determination of the termination point. A wide range of approaches have been proposed to tackle this problem including stopping conditions such as size constraints (Condon and Karp, 1999; Flake et al., 2004) and cluster density (Hartuv and Shamir, 2000).

2.6.2 Local Clustering

Local clustering algorithms address the complexity drawbacks of global clustering algorithms by using solely local information to generate an appropriate clustering of the input graph. The input nodes for local algorithms are called seeds. Two main categories of approaches implement local clustering, namely approaches based on local search and approaches based on fitness functions (Schaeffer, 2007). Local search methods apply probabilistic decision-making to retrieve nearly-optimal solution to the clustering problem. Johnson et al. (1989) and later Schaeffer (2005) propose local search approaches based on simulated annealing (Kirkpatrick et al., 1983). More recently, Booth et al. (2007) used another stochastic search method based on the Metropolis-Hastings algorithm (Hastings, 1970). Local search methods also include the use of heuristics such as those studied by Monien and Diekmann (1997) and Hoos and Stützle (1999). Other local search-based approaches are based on techniques such as hill-climbing (do Nascimento and Eades, 2001) and tabu search (Glover and Laguna, 1997).

Current algorithms for local graph clustering based on fitness functions optimize a wide range of criteria. Simple fitness functions on clusters ζ include the average internal degree of $u \in \zeta$

$$\frac{1}{|\zeta|}\sum_{u\in\zeta} deg_{int}(v,\zeta), \tag{2.39}$$

where

$$deg_{int}(v,\zeta) = |\{vu \in E \mid u \in \zeta\}| \tag{2.40}$$

and the introversion

$$\frac{1}{|\zeta|}\sum_{v\in\zeta} \frac{deg_{int}(v,\zeta)}{d(v)}. \tag{2.41}$$

Other algorithms utilize variations of more complex fitness functions such as PageRank vector (Andersen et al., 2006) and the Cheeger ratio (Orponen and Schaeffer, 2005; Chung, 2007)

$$\frac{|\{uv \in E | u \in \zeta, v \in V\backslash\zeta\}|}{\min\left\{\sum_{u\in\zeta} d(u), \sum_{v\in V\backslash\zeta} d(v)\right\}}. \tag{2.42}$$

Local algorithms are usually faster than global algorithms. Furthermore, some of them can be used in an online fashion (Schaeffer, 2007).

2.7 Evaluation

This section gives a short overview of the measures and tests used in this work for the evaluation and comparison of techniques.

2.7.1 Evaluation Measures

Several evaluation measures have been defined to ensure the comparability of results achieved by different systems or in different settings. The underlying model for our evaluation can be formulated as follows: let the universe U be the set of entities that can be retrieved by a system S. Furthermore, let Rel be the set of relevant entities for a given task and Ret the set of entities retrieved by S. The precision p of S gives the ratio between the number of relevant entities retrieved by S and the total number of entities that S retrieved. Thus,

$$p = \frac{|Rel \cap Ret|}{|Ret|}. \tag{2.43}$$

The recall r measures the ratio between the number of relevant entities retrieved by S and the total number of relevant entities:

$$r = \frac{|Rel \cap Ret|}{|Rel|}. \tag{2.44}$$

Depending on the task at hand, the precision or recall might be of greater importance. In our special case, we will be more interested in precision because we aim at extracting a set of concepts with a purity as high as possible.

2.7.2 Statistical Testing

Parts of the evaluations carried out in this work compare the results of different techniques on the same task. The precision and recall give us detailed information on the performance of different systems in different settings. However, it is often relevant to compare systems on a more global level. A series of statistical tests were proposed in literature to achieve this goal. The most common test in this respect is Student's t-test (Gosset, 1908). Yet, this test cannot be used in our evaluations because it assumes a normal distribution of the measurements. Such a distribution is not always given in our experiments.

A popular parameter-free test is the Wilcoxon Signed Rank test (Wilcoxon, 1945) (henceforth also Wilcoxon Rank test). It is used to compare the results of two sets X and Y of measurements. The only assumption underlying this test is that the differences $z_i = y_i - x_i$ with $x_i \in X$ and $y_i \in Y$ come from a continuous population. Although we deal with rational values in this thesis, the large size of the populations used in our evaluation allow the use of this test. To ensure the comparability of our results, we will also provide the p-values computed by a t-test. However, we shall rely on the results of the Wilcoxon Signed Rank test for comparing approaches.

2.7.3 Cluster Evaluation

The necessity of evaluating cluster validity has led to the development of numerous metrics (also called indexes). In general, the quality of a cluster ζ of size n can be measured according to two main criteria: the intra-cluster and the inter-cluster similarity (Boutin and Hascoet, 2004). Metrics based on the intra-cluster similarity take only local information on the cluster to measure its validity. For example, the compactness index (Botafogo et al., 1992) given by

$$\frac{Max - \sum\limits_{i=1}^{n-1} \sum\limits_{j=i}^{n} d(\omega_i, \omega_j)}{Max - Min}, \tag{2.45}$$

where

$$\begin{aligned} Max &= \max_{i \neq j} d(v_i, v_j), \\ Min &= \min_{i \neq j} d(v_i, v_j) \text{ and} \\ d(\omega_i, \omega_j) &\text{ is a distance measure} \end{aligned} \tag{2.46}$$

measures the variation of the intra-cluster connectivity from the maximal connectivity. Most other metrics use a combination of intra- and inter-cluster similarity to measure the quality of clusters. For example, the silhouette index (Rousseeuw, 1987) is defined by

$$\frac{1}{|\zeta|} \sum_{\omega_i \in \zeta} \frac{a_i - b_i}{\max(a_i, b_i)}, \tag{2.47}$$

where a_i is the distance from ω_i to the closest cluster to which it does not belong and b_i is the average distance from ω_i to the elements of the cluster to which it belongs. The silhouette index combines the intra-cluster dissimilarity of a cluster with the dissimilarity of the elements of the same cluster to external nodes to compute the quality of a cluster. The global quality of a clustering according to the silhouette index is defined herein as the average silhouette index of the clusters generated by this clustering.

The most common global indexes that evaluates the whole clustering is Dunn's index (Dunn, 1974), which is defined as

$$\min_{1 \leq i \leq K} \left\{ \min_{1 \leq j \leq K, i \neq j} \left\{ \frac{\delta(\zeta_i, \zeta_j)}{\max_{1 \leq k \leq K} \Delta(\zeta_k)} \right\} \right\}, \tag{2.48}$$

where

- K is the number of clusters,
- $\delta(\zeta_i, \zeta_j) = \min_{\omega \in \zeta_i, \omega' \in \zeta_j} d(\omega, \omega')$ is the minimal link distance between the clusters ζ_i and ζ_j, and
- $\Delta(\zeta) = \max_{\omega, \omega' \in \zeta} d(\omega, \omega')$ is the diameter of ζ.

Dunn's index measures the ratio between the minimal dissimilarity between clusters and the diameter of the largest cluster.

In this work, we will use the silhouette index to measure the validity of our clusters because it allows a fine-grained comparative evaluation of the quality of clusterings generated by different algorithms. For more complete surveys on graph clustering and validation, the reader is referred to Newman (2004); Boutin and Hascoet (2004); Tan et al. (2005); Schaeffer (2007).

Chapter 3

Discovery of Domain-Specific Multi-Word Units

The goal of this chapter is to present and evaluate the Smoothed Relative Expectation (SRE), a novel metric designed for the low-bias extraction of domain-specific MWUs. This chapter is structured as follows: in the first section, we present criteria that characterize domain-specific MWUs. In the section thereafter, we map these criteria to measures that allow us to compute the degree to which these criteria are fulfilled by a given n-gram. Then, we specify the formula for SRE as the product of the prior measures. Finally, we compare SRE with other metrics for MWU extraction. For this purpose, we evaluate all metrics against prominent gold standards of varying completeness on two data sets of different size. Some of the results presented in this chapter were published in (Ngonga Ngomo, 2008a,b).

3.1 Characterization of Domain-Specific Multi-Word Units

Discovering multi-word units is a preprocessing task that can be integrated in almost all NLP applications. According to Choueka (1988), a *multi-word unit* is a connected collocation, "whose exact and unambiguous meaning or connotation cannot be derived from the meaning or connotation of its components". Choueka's definition mainly implies that the meaning of a MWU is not a function of the meaning of its components. This characteristic is known in literature as *semantic non-compositionality* and considered to be one of the main criterion for differentiating general-language MWUs from other collocations (Manning and Schütze, 1999; Schone and Jurafsky, 2001). However, semantic non-compositionality only holds partly for domain-specific MWUs. Individuals in a given domain utilize expressions that are typical for their domain to convey a certain meaning. According to Manning and Schütze (1999), albeit the meaning of certain domain-specific expressions can be derived from their constituents, they are still to be considered as domain-specific MWUs, as they convey exactly the meaning of a domain-specific concept. Therefore, the semantic non-compositionality criterion is not sufficient for detecting domain-specific MWUs.

Another characteristic of domain-specific MWUs, called *non-substitutability*, is also pointed out in the literature (Manning and Schütze, 1999; Schone and Jurafsky, 2001). Non-substitutability holds for a given n-gram when its components cannot be replaced by semantically similar components without altering the meaning of the n-gram or making it meaningless. For example, the word *compact* in the expression *compact disk* cannot be replaced by *dense* or any other similar term without altering the meaning of the expression. Non-substitutability can be used for the extraction of domain-specific MWUs because a high percentage of domain-specific vocabularies consists of such fixed expressions (Jiang and Tan, 2005).

The third criterion for the extraction of MWUs is their *non-modifiability*. This criterion holds for MWUs because their structure cannot be altered into a grammatically equivalent structure without changing their meaning. For example, the common expression *black sheep* cannot be transformed into *sheep that is black* without changing the idiomatic meaning of the expression. Thus, an approach to MWU extraction must take into account the position of terms in expression, making sequence-based approaches (such as that presented herein) best suited for the extraction of MWUs.

Compared with general language MWUs, domain-specific MWUs bear a higher *specificity*. Therefore, domain-specific MWUs must display a smaller scattering over documents according to the considerations of Bookstein and Swanson (1974) and Robertson and Jones (1976). This particular characteristic is usually not taken into consideration by metrics for MWU extraction. The modeling of the specificity of domain-specific MWUs is hence the main difference between SRE and other metrics. The three criteria semantic non-substitutability, non-modifiability and specificity will be the basic assumptions underlying SRE.

3.2 Smoothed Relative Expectation

The Smoothed Relative Expectation (SRE) metric was developed especially for discovering domain-specific MWUs according to the criteria described in Section 3.1. To achieve this goal, SRE uses the distribution of MWUs over documents to smooth their relative expectancy score. Our metric is independent from manually set thresholds for function words because it inherently detects and ranks down patterns that are too frequent. SRE also ranks down patterns which are not frequent enough to be supposed correct. In the following, we use each of the three criteria proposed in Section 3.1 to specify a section of SRE. Subsequently, we explicate the complete SRE formula. Finally, we compare SRE with state-of-the-art metrics commonly used for the extraction of MWUs.

3.2.1 Non-Substitutability and Non-Modifiability

Let $c_1, \ldots, c_n \in C$ be words from the set C of words contained in a corpus. Furthermore, let $w = c_1 ... c_n$ be a connected collocation occurring in a given corpus. The assumptions of non-substitutability implies that none of the c_i can be substituted with a $c'_i \in C$ having approximately the same semantics as c_i without considerably altering the semantics of the term. Let $w' := c_1 ... c_{i-1} c'_i c_{i+1} ... c_n$, $1 \leq i \leq n$, be the sequence of terms that results from the substitution of c_i by c'_i ($c_i \neq c'_i$) in w. w will be considered likely to be a MWU if it occurs often in comparison with other patterns w', i.e., if it has a high relative expectaction.

Non-modifiability implies that w can be considered to be a multi-word unit when the probability of non-MWU patterns similar to w occurring in the same corpus is lower than the probability of occurrence of w. Defining the similarity of patterns has been deeply investigated in the domain of pattern and string matching algorithms. A good overview of techniques for this purpose can be found in (Charras and Lecroq, 2004). In this work, we will use the Hamming distance (Hamming, 1950) $ham(w, w')$ between the patterns w and w' to measure their similarity because it has a linear complexity. The Hamming distance is defined as follows:

$$ham(w, w') = \sum_{i=1}^{n} dif(c_i, c'_i) \tag{3.1}$$

with

$$dif(c_i, c'_i) = \begin{cases} 1 & \text{if } c_i \neq c'_i; \\ 0 & \text{else.} \end{cases} \tag{3.2}$$

The combination of the considerations on non-substitutability and non-modifiability presented above leads to the assumption that the expectation of a MWU should be greater than that of similar non-MWU patterns. Over all Hamming distances between 1 and $n-1$, the expectation $E_n(w)$ of w relatively to similar patterns can be computed as follows:

$$E_n(w) = p(w) \prod_{i=1}^{n-1} \frac{n^i f(w)}{\sum_{w'} f(w') : ham(w, w') = i}, \tag{3.3}$$

where

- $f(w)$ is the number of occurrences of w in the corpus and
- p is the probability of a random collocation to be w.

Henceforth, we will use the first approximation $E_1(w)$ of the expectation of w, which is computed relatively to the patterns such that $ham(w, w') = 1$:

$$E_1(w) = p(c_1...c_n) \frac{n f(c_1...c_n)}{\sum_{i=1}^{n} f(c_1...c_i * c_{i+2}...c_n)}, \tag{3.4}$$

where $*$ is the wild card symbol.

3.2.2 Specificity

Another characteristic of domain-specific MWUs are their higher specificity and, thus, their lower scattering over the corpus. This scattering of domain-specific MWUs is not considered in most metrics proposed for MWU extraction, leading to a bias toward highly frequent patterns. Using solely the expectation E_1 (see Equation (3.4)) for the extraction of domain-specific MWUs would also be biased toward counting highly frequent n-grams as being better MWU candidates. In order to eliminate this bias, we introduce a smoothing factor. According to the assumption of specificity, the smoothing factor must use the scattering of MWU candidates over the corpus to improve their score. Hence, the smoothing factor aims at reducing both the score of very frequent patterns that contain less information and the score of very infrequent patterns, on which the information available is too sparse. Several smoothing factors can be considered to achieve this goal. In this work, we are mainly concerned with showing that the use of a model for specificity does improve the extraction of domain-specific MWUs. Therefore, we will use a simple model for specificity and weight the relative expectation E_n along the binomial distribution, as it fulfills both conditions.

The binomial distribution is given as follows:

$$P(k; n, p) = \frac{n!}{k!(n-k)!} p^k (1-p)^{n-k}, \tag{3.5}$$

where

- n is the number of documents and
- p is the probability of a n-gram occurring in a document, i.e., $p = 1/N$, N being the mean size of a document.

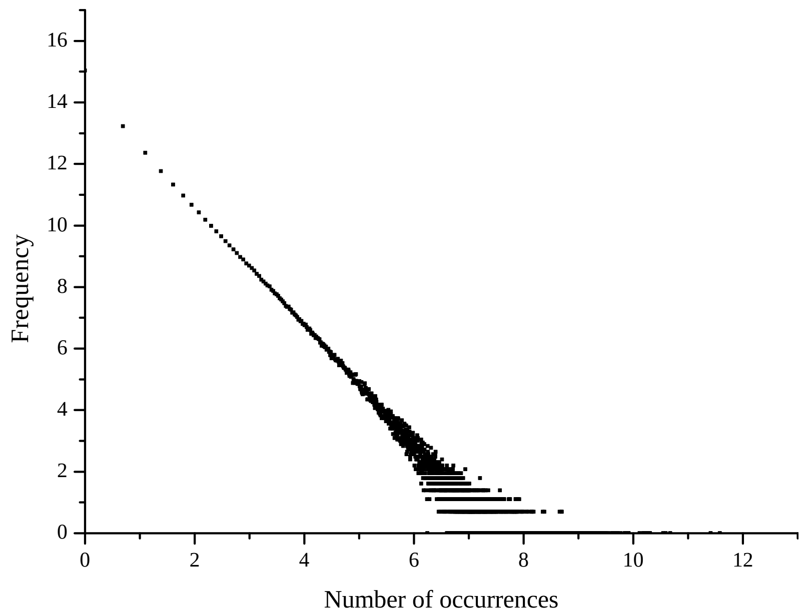


Figure 3.1: Scatter graph of bigram distribution in the OSHU-TREC corpus (log-log scale)

Given the large value of the mean for the corpora considered in this section and for the sake of computational complexity, we will approximate the binomial distribution with the Gaussian distribution, which looks as follows:

$$p(k; \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(k-\mu)^2}{2\sigma^2}}, \tag{3.6}$$

where

- μ is the mean of n-gram occurrences in documents and
- σ^2 is its variance.

This smoothing component boosts the weight of the n-grams having a frequency around the mean μ and scales down the weight of the n-grams with frequencies lying far from μ, which results in a reduction of the bias toward very frequent patterns.

For the TREC corpus for example, the mean of the distribution of bigrams (also written bi-grams) over documents is 5.67, while the standard deviation of the same distribution is 137.27 (see Figure 3.1), leading to the smoothing factor displayed in Figure 3.2.

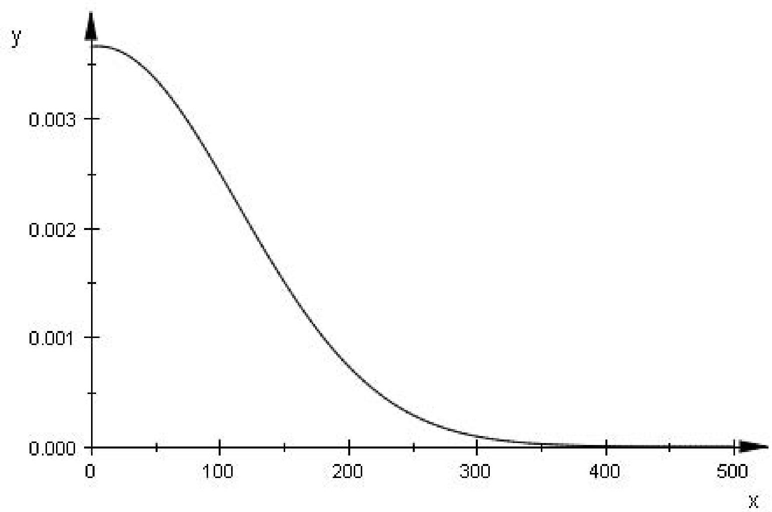


Figure 3.2: Normal distribution with $\sigma = 137.27$ and $\mu = 5.67$

3.2.3 Resulting Metric

Based on the previous considerations, we define the Smoothed Relative Expectation (SRE) of a given n-gram as follows:

$$SRE(w) = p(w) \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(d(w)-\mu)^2}{2\sigma^2}} \frac{nf(w)}{\sum_{i=1}^{n} f(c_1 \ldots c_i * c_{i+2} \ldots c_n)}, \tag{3.7}$$

where

- $d(w)$ is the number of documents in which w occurs,
- μ and σ^2 are the mean and the variance of the distribution of n-grams in documents respectively,
- $p(w)$ is the probability of occurrence of w in the whole corpus,
- $f(w)$ is the frequency of occurrence of w in the whole corpus and

- $c_1...c_i * c_{i+2}...c_n$ are patterns such that $ham(w, c_1...c_i * c_{i+2}...c_n) = 1$.

The normalized score SRE$_{norm}$ (i.e., with a range between 0 and 1 inclusive) resulting from SRE is given by

$$SRE_{norm}(w) = p(w)e^{-\frac{(d(w)-\mu)^2}{2\sigma^2}} \frac{nf(w)}{\sum_{i=1}^{n} c_1...c_i * c_{i+2}...c_n}. \tag{3.8}$$

As SRE$_{norm}$ produces the same results as SRE in terms of the ranking of MWUs according to their scores, SRE can be implemented as such.

3.3 Implementation Details

In our implementation of SRE, we used prefix trees (also called TRIEs) (Fredkin, 1960) to store token sequences. Prefix trees were developed for the efficient storage and retrieval of indexes in the IR context. Minor modifications of the basic concept of prefix trees allow the efficient storage and counting of n-grams. A prefix tree of depth n was used to store sequences of length n. A counter for the frequencies was assigned to each node, the counter of a node at the depth d storing the frequency of the pattern comprised between the root of the prefix tree and the depth d. Similarly, a counter for the occurrence in documents was integrated in the nodes of the prefix-tree. An excerpt of the modified prefix tree extracted while processing the TREC corpus for bigrams is displayed in Figure 3.3.

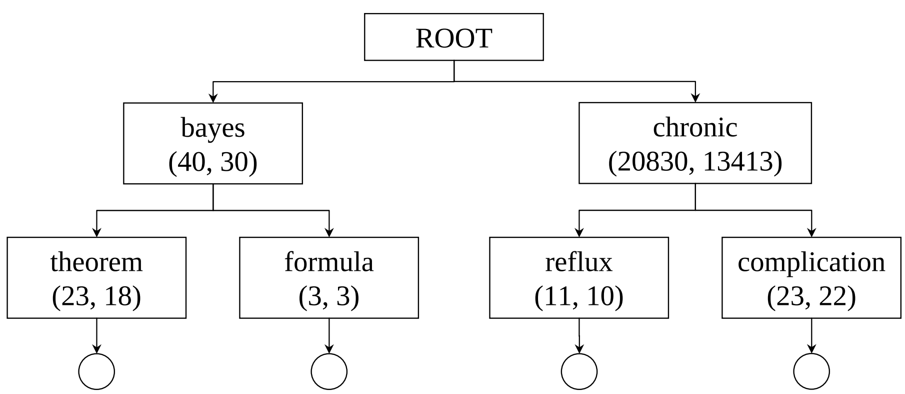


Figure 3.3: Excerpt from the prefix tree for bigrams of TREC. *The circles represent null nodes, while the entries of the number pair in the nodes stand for the frequency and document count.*

The main advantage of prefix trees lies in the fact that each insertion of patterns of length n enables the counting of sub-patterns $c_i...c_j$ with $1 \leq i < j \leq n$ where

$c_1...c_N$ with $N >> n$ is the stream to analyze. When inserting a pattern $c_1...c_n$, all its sub-patterns $c_1...c_j$ with $j < n$ are counted. Text being processed linearly, the next processing step consists of processing the pattern $c_2...c_{n+1}$, which results in counting all the patterns $c_2...c_j$ with $j < n$. After n processing steps, all patterns $c_i...c_j$ with $1 \leq i < j \leq n$ are inserted in the tree. An example of the sequential insertion of the words from the word stream $(c_1c_2c_3c_1c_4c_5)$ in a prefix tree of depth 3 is shown in Figure 3.4. This characteristic of the modified prefix trees used in our implementation speeds up the pattern matching necessary to compute the SRE of patterns of size greater than 2.

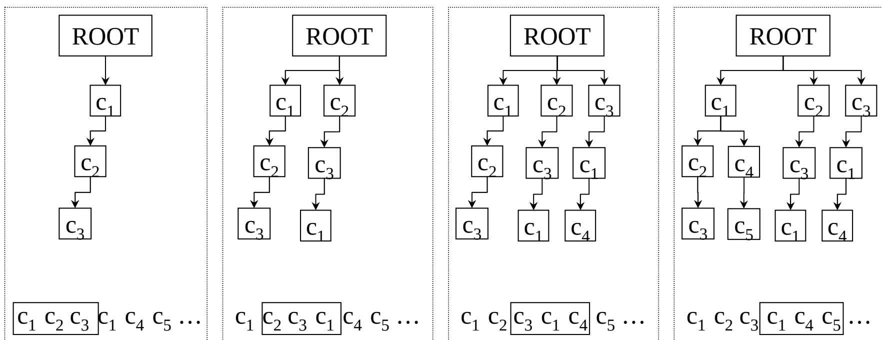


Figure 3.4: Insertion of a word sequence in a prefix tree of depth 3. *The null nodes, frequency and document counters were omitted for reasons of clarity.*

In addition, the modified prefix tree used in our implementation also allows the rapid retrieval of the stored entries. In particular, the complexity of this operation can be reduced to $O(n)$, when using a perfect minimal hash (Botelho et al., 2007; Botelho and Ziviani, 2007) to store the nodes' indexes.

3.4 Experiments and Results

We carried out the evaluation process presented in this section as depicted in Figure 3.5. We used two data sets and three gold standards from the bio-medical domain. The data sets were chosen to reflect average data corpora available in domain-specific information systems. The first data set was a manually preprocessed data set that was designed to be used in the context of information retrieval. The second data set was extracted directly from a set of documents and was accordingly noisy. Several gold standards are used for experiments on bio-medical corpora. We used the three most common gold standards for two reasons. First, we wanted to ensure the comparability of our results with those obtained by other researchers on the same domain. Second, these gold standards present different levels of restrictiveness. This difference allowed us to simulate the behavior of the metrics in domains of varying breadth.

In the following, we first discuss our experimental setup and present specifics of the data sets and gold standard used for our evaluation. Then, we present a generalization of the metrics used in our experiments. Subsequently, we evaluate our metric by comparing it with other metrics in two sets of experiments. In the first series of evaluations, we compare our metric with six other metrics commonly used in literature. In the second series, we compare our metric with other multi-contextual metrics.

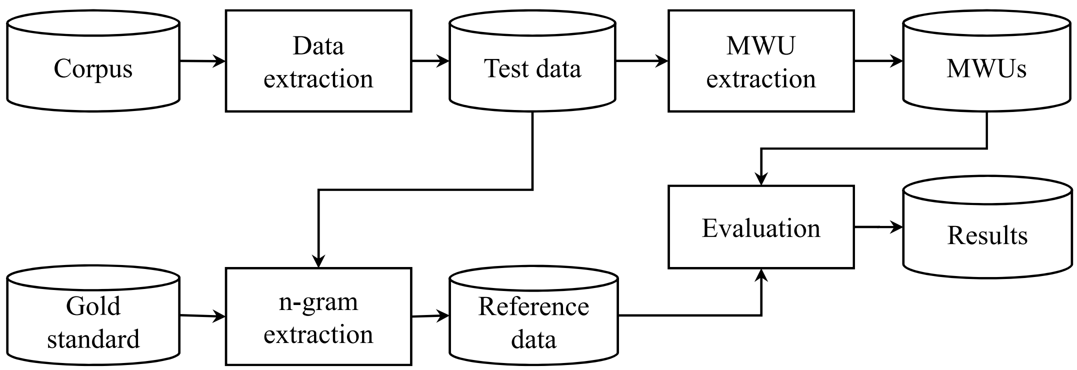


Figure 3.5: Evaluation process for MWU extraction metrics

3.4.1 Experimental Setup

Data Sets

The data sets underlying the results presented in this chapter are the *TREC* corpus for filtering (Robertson and Hull, 2001) and a subset of the articles published by BioMed Central (BMC[1]). Henceforth, we will call the second corpus *BMC*. The TREC corpus is a test collection composed of abstracts of publications from the bio-medical domain. The entries in this corpus include the abstracts of publications (marked with a `.W` in each entry) and further metadata such as the subject, type of publication, etc. (see Appendix A). The data extraction process consisted exclusively of retrieving all the text entries (i.e., those marked with `.W` in the TREC corpus) and the deletion of punctuation. 233,445 abstracts (244 MB) contained 38,790,593 running word forms, which were retrieved and used for the evaluation

[1] http://www.biomedcentral.com

presented in this section. 355,616 unique word forms were extracted from the corpus with a mean frequency of 109.08. 6,096,183 unique bigrams were found, their mean frequency being 6.36. The mean occurrence of bigrams in documents was 5.67, with a standard deviation of 137.27.

The *BMC corpus* consists of full text publications extracted from the BMC Open Access library. The original documents were in XML. We extracted the text entries from the XML data using a SAX[2] Parser. The preprocessing consisted of retrieving the content of the text nodes and the deletion of punctuation. The corpus was not post-processed manually. Therefore, it contained a large amount of impurities that were not captured by the XML-parser. The main idea behind the use of this corpus was to test our method on real life data. 13,943 full text documents (507 MB) contained 70,464,269 running word forms. 2,720,845 unique word forms were extracted from the corpus with a mean frequency of approximately 25.90. 13,929,186 unique bigrams were found, their mean frequency being approx. 5.04. The mean occurrence of bigrams in documents was 3.49 with a standard deviation of 52.05.

Gold Standards

We used data extracted from the MESH (*ME*dical *S*ubject *H*eadings), SNOMED-CT (*S*ystematized *NO*men-clature of *MED*icine-*C*linical *T*erms) and UMLS (*U*nified *M*edical *L*anguage *S*ystem) terminologies as gold standards (Ananiadou and Mcnaught, 2005). The first gold standard (MESH) consists of the set of all MESH descriptors[3]. MESH terms were chosen because they are used in several journals and conferences to tag and classify medical publications. In particular, they were used to tag the TREC corpus.

The second gold standard consists of terms extracted from the SNOMED-CT terminology[4]. SNOMED-CT is a standardized health care terminology, which includes terms describing diseases, clinical findings, therapies, etc. It contains more than 357,000 concepts organized as a Directed Acyclic Graph (DAG).

Both the MESH and the SNOMED-CT standards are very restrictive because they capture only certain aspects of the bio-medical domain. Therefore, we chose UMLS[5] as our third gold standard. UMLS aims at being the unification of the main terminologies used in medicine. The core of UMLS is the Metathesaurus, which contains approximately 1.4 million concepts and contains large subsets of 22 source vocabularies including MESH and SNOMED-CT. UMLS is, thus, more complete

[2]SAX stands for Simple Application Programming Interface for XML.

[3]Found at `http://www.nlm.nih.gov/mesh`. Version of July 16^{th}, 2007

[4]Found at `http://www.ihtsdo.org/snomed-ct/`. Version of July 31^{st}, 2007.

[5]Found at `http://www.nlm.nih.gov/research/umls`. Version of July 31^{st}, 2007.

and consequently more reliable than the two other gold standards. Terms randomly chosen from all three gold standards are shown in Table 3.1.

MESH terms	**SNOMED-CT terms**	**UMLS terms**
2-phospho-d-glycerate hydrolase	epidermolysis bullosa	toxic myocarditis
alcohol dehydrogenase	mycosis fungoides	sodium metabisulphite
alpha-aminoadipic acid	spina bifida	splenogonadal fusion
anterodorsal nucleus	lichen planus	allergic stomatitis
cronkhite-canada syndrome	isosorbide dinitrate	closed dislocation
gambierdiscus toxicus	carbonic anhydrase	cutaneous vein
madurella mycetomatis	bacillus calmette-guerin	lipid-reducing agents
morone americana	anorexia nervosa	haemorrhagic hypotension
tyrosyl-trna synthetase	sclerosing cholangitis	sensory cell

Table 3.1: Exemplary MESH, SNOMED-CT and UMLS bigrams found in the TREC corpus

Metrics

In our evaluation, we compared SRE with six other metrics used for MWU extraction (Schone and Jurafsky, 2001; Dias, 2002): the Dice formula (DICE) (Dice, 1945), the frequency of patterns (FR) (Giuliano, 1964), the Pointwise Mutual Information (PMI) (Church and Hanks, 1989), the Symmetric Conditional Probability (SCP) (Ferreira da Silva and Pereira Lopes, 1999), the Mutual Expectation (ME) (Dias, 2002) and the TF-IDF norm (TFIDF) (Maedche and Staab, 2000). Three of these metrics (i.e., DICE, PMI and SCP) are not suitable for measuring the degree of association of more than 2 terms. For this reason, they could not be used in their original form when processing n-grams with $n > 2$. In our implementation, we used generalized versions of the three metrics according to the formulae derived in the following section.

3.4.2 Generalization of Binary Measures

In the following, extensions of the DICE, the PMI and the SCP metrics to n-ary interdependencies are proposed. The approach followed in each case is based on the behavior of the functions in the extreme cases of perfect association and perfect independence. The functions f and p on the set of terms are the frequency and probability function respectively.

Dice Metric

When applied to the score of bigrams the Dice formula is as follows (see (Dias, 2002, p. 137))

$$dice(w) = \frac{2f(w)}{f(c_1) + f(c_2)}, \tag{3.9}$$

with $w = c_1c_2$ being a bigram. When extended to an n-gram $w = c_1...c_n$, this formula can be extended to

$$dice(w) = \frac{nf(w)}{\sum_{i=1}^{n} f(c_i)}, \tag{3.10}$$

ranging between 0 (perfect independence) and 1 (perfect association) like the original formula.

Pointwise Mutual Information

The Pointwise Mutual Information PMI(X,Y) measures the degree to which the occurrence of a word c_1 depends on the occurrence of a word c_2 and is given by

$$PMI(w) = log_2 \frac{p(w)}{p(c_1)p(c_2)}, \tag{3.11}$$

where $w = c_1c_2$. In case of independence of c_1 and c_2, $p(w) = 0$ and therefore $PMI(w) = -\infty$. In case of perfect correlation, $p(w) = p(c_1) = p(c_2)$ and thus $PMI(w) = -log_2(p(c_1))$. The extended version of PMI used in this work measures to which degree the single events are interdependent and keeps the boundaries set in case of independence and perfect association.

$$PMI(c_1...c_n) = \frac{1}{n-1} log_2 \frac{p(w)}{\prod_{i=1}^{n} p(c_i)}, \tag{3.12}$$

where $w = c_1...c_n$. In case of independence p(w) = 0 and thus $PMI(w) = -\infty$. In case of perfect correlation $PMI(w) = \frac{1}{n-1} log_2(1/p(c_1)^{n-1}) = -log_2(p(c_1))$. Normalizing a function using a positive constant does not alter its monotony. Since the results of MWU extraction is an ordered list of weights, the PMI scores were normalized by multiplying them with (n-1), leading to the final formula:

$$PMI(w) = log_2 \frac{p(w)}{\prod_{i=1}^{n} p(c_i)}. \tag{3.13}$$

Symmetric Conditional Probability

When applied to bigrams, the Symmetric Conditional Probability $SCP(w)$ is as follows (see (Dias, 2002, p. 137)):

$$SCP(w) = \frac{f(w)^2}{f(c_1)f(c_2)}, \tag{3.14}$$

where $w = c_1c_2$. In case of independence of c_1 and c_2, p(w) = 0 and therefore $SCP(w) = 0$. In the other case, i.e., perfect correlation, $f(w) = f(c_1) = f(c_2)$ and thus $SCP(w) = 1$. Thus, SCP can be easily transformed to apply to n-grams with $n > 2$ in the following manner:

$$SCP(w) = \frac{f(w)^n}{\prod_{i=1}^{n} f(c_i)}, \tag{3.15}$$

where $w = c_1...c_n$. The extension yields the same behavior when in both extreme cases.

The resulting metrics are summarized in Table 3.2, where f represents the frequency function, p the probability, n the length of the n-grams, D is the number of the documents in the corpus and $d(w)$ is the number of documents in which the n-gram w occurs.

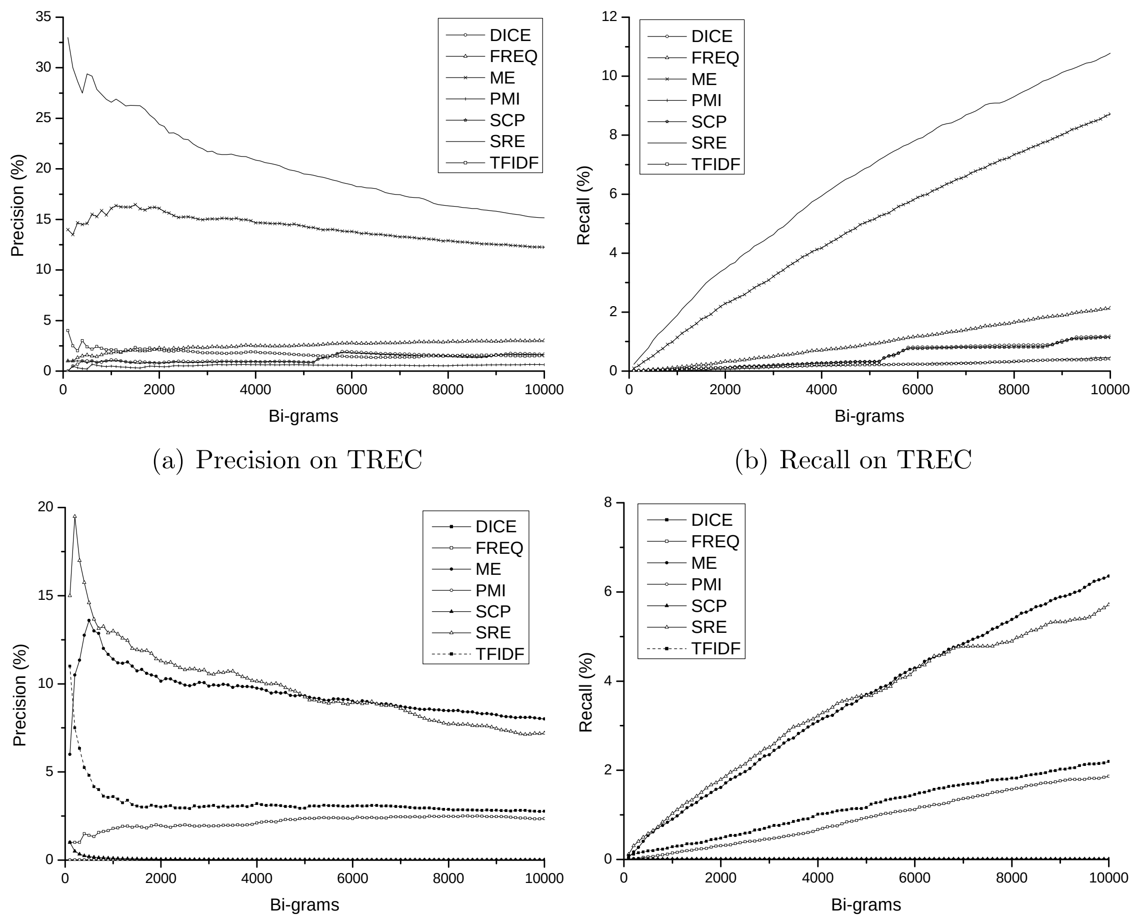
Metric	Formula
Dice formula (DICE)	$n\frac{f(w)}{\sum_{i=1}^{n} f(c_i)}$
Frequency (FR)	$f(w)$
Pointwise Mutual Information (PMI)	$log_2\frac{p(w)}{\prod_{i=1}^{n} p(c_i)}$
Symmetric Conditional Probability (SCP)	$\frac{f(w)^n}{\prod_{i=1}^{n} f(c_i)}$
Mutual Expectation (ME)	$p(w)\frac{nf(w)}{\sum_{i=1}^{n} f(c_1...c_{i-1}*c_{i+1}...c_n)}$
TF-IDF Norm (TFIDF)	$f(w)log\left(\frac{D}{d(w)}\right)$

Table 3.2: Metrics for MWU extraction

3.4.3 Evaluation on Bigrams

In this section, we present the results of our evaluation on the TREC and BMC corpora. The output of each metric was an ordered list of n-grams, of which η between 100 and 10,000 were considered in each evaluation step. The full results of the evaluation can be found in Appendix B. The figures display the precision and recall in percent and were computed using the full result tables. We used the Wilcoxon Rank test and the t-test to compute the statistical significance of the results we obtained. For the sake of clarity, we first present the results obtained using each reference terminology separately and subsequently discuss and compare them.

Using MESH as the Gold Standard



(a) Precision on TREC

(b) Recall on TREC

(c) Precision on BMC

(d) Recall on BMC

Figure 3.6: Precision and recall on TREC using MESH as the gold standard

The initial set of 36,984 bigrams was reduced to the subset of terms that could actually be found in our data sets. Thus, the first gold standard consisted of 14,055 bigrams (approximately 38%) for TREC and 12,602 bigrams (approximately 34.07%)

for BMC.

The precision achieved by the seven metrics considered in this evaluation is shown in the upper section of Table 3.3. Figure 3.6(a) displays the precision achieved in the same setting in a graphical form. The baseline for the precision on TREC was 0.23%. SRE significantly outperformed the other metrics, the greatest difference in comparison with Dias' ME being of 19% absolute and 135.71% relative in the best case (n = 100, see Appendix B), the mean difference being of 6.25% absolute and 44.17% relative. A t-test and a Wilcoxon Rank test with a confidence level of 99% revealed that the precision of SRE was significantly better than that of ME ($p < 10^{-6}$).

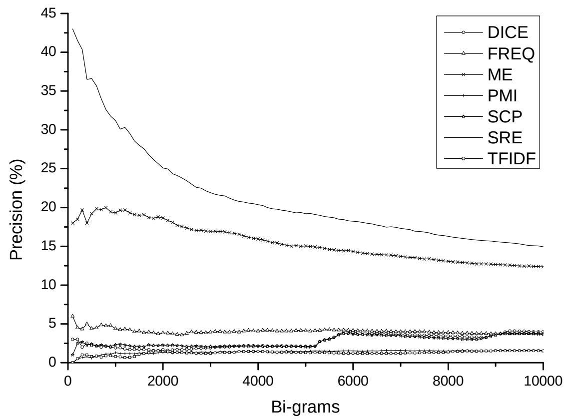
The recall achieved by the metrics is displayed in Figure 3.6(b) and the lower section of Table 3.3. Again, SRE outperforms the other metrics. The maximum improvement in comparison with ME was 135.71% (n = 100, see Appendix B) and the mean improvement was 33,89% relative. A t-test and a Wilcoxon Rank test with a confidence level of 99% revealed that the recall of SRE was significantly better than that of ME ($p = 3 \times 10^{-5}$).

On the BMC corpus, SRE outperformed all other metrics on the first 5,000 bigrams (see Figure 3.6(c)) but was then outperformed by ME. A t-test and a Wilcoxon Rank text (confidence level 95%) revealed that both populations (i.e., ME and SRE) did not significantly differ from each other. The same statistical results were computed for the recall (see Figure 3.6(d)). The performance of SRE on the last 5,000 bigrams was mostly due to the restrictiveness of MESH, as the evaluation using the other gold standards showed.

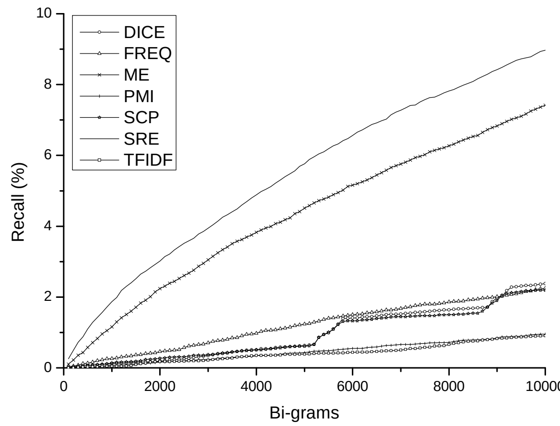
Using SNOMED-CT as the Gold Standard

We extracted 72,218 bigrams out of the set of SNOMED-CT concepts, of which 16,661 (approx. 23.07%) could be found in the TREC corpus and 13,800 (approx. 19,11%) could be found in the BMC corpus.

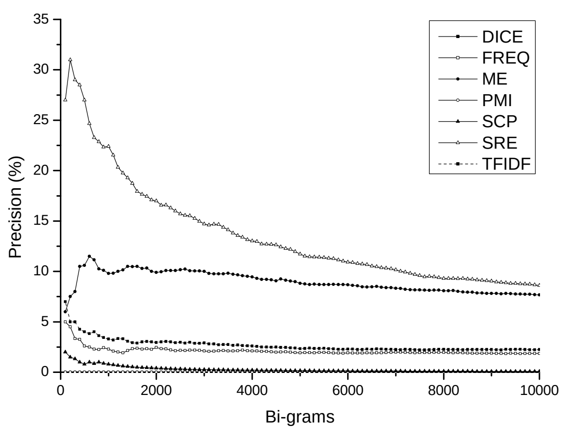
An excerpt of the precision and recall achieved by the seven metrics on the TREC and BMC corpora on list sizes between 100 and 10,000 is displayed in the upper section of Table 3.4. The complete results are shown graphically in Figure 3.7(a). The baseline for the precision was 0.27%. Again, SRE significantly outperformed the other metrics. The greatest difference in comparison with Dias' ME was 25% absolute and 138.89% relative in the best case (n = 100, see Appendix B). The mean difference between SRE and ME was 5.76% absolute and 37.24% relative. A t-test with a confidence level of 99% revealed that the precision achieved by SRE was significantly better than that of ME ($p < 10^{-6}$). A Wilcoxon Rank test with the same confidence level yielded the same results.



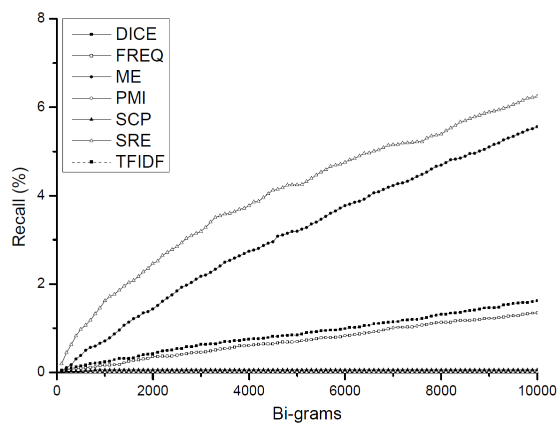
(a) Precision on TREC



(b) Recall on TREC



(c) Precision on BMC



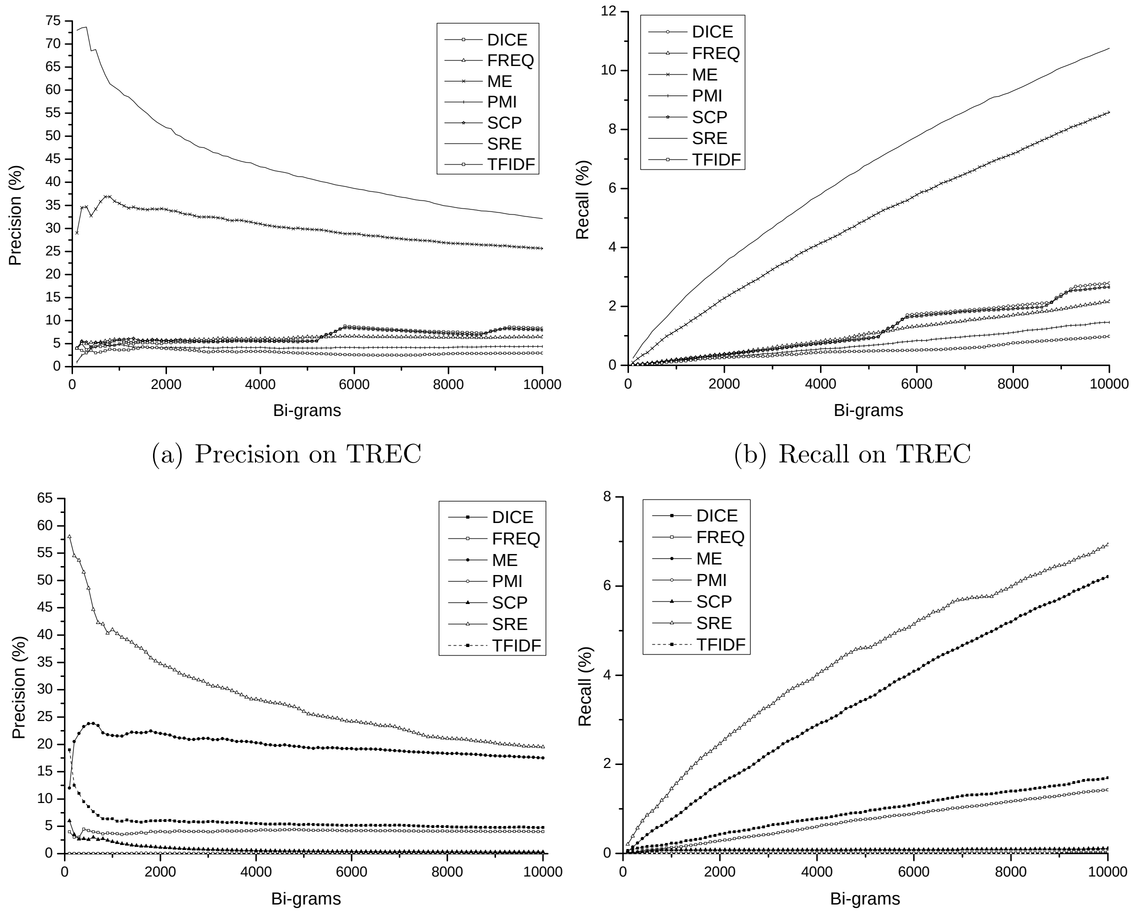
(d) Recall on BMC

Figure 3.7: Precision and recall using SNOMED-CT as the gold standard

In terms of recall (see Figure 3.6(b)), SRE also outperformed the other metrics (see lower section of Table 3.4). The maximum improvement in comparison with ME was 138.89% relative (n = 100) and 1.63% absolute (n = 9400, see Appendix B). The mean difference was 27.56% relative. A t-test with a confidence level of 99% reveals that the recall of SRE was significantly better than that of ME ($p = 2.5 \times 10^{-4}$). The statistical signifance of the results was confirmed by a Wilcoxon Rank test with the same confidence level.

On the BMC corpus, SRE significantly outperformed all other metrics (t-test and Wilcoxon Rank test, confidence level 99%) both in precision (see Figure 3.7(c)) and recall (see Figure 3.7(d)). With respect to precision, the greatest difference in comparison with ME is 21% absolute (n = 100) and 350% relative (n = 100). The mean difference was 4.64%. In terms of recall, the greatest difference between ME and SRE was 3.33% absolute (n = 3300) and 400% relative (n = 100).

Using UMLS as the Gold Standard



(a) Precision on TREC

(b) Recall on TREC

(c) Precision on BMC

(d) Recall on BMC

Figure 3.8: Precision and recall using UMLS as the gold standard

To generate the UMLS reference data, we extracted 827,159 entries representing 591,213 concepts from the table of inflections (LGAGR). 171,635 entries were bi-grams, of which 29,887 (approx. 17.41%) could be found in the TREC corpus and 28,204 (approx. 16.43%) in the BMC corpus. The results of the evaluation using UMLS confirm the superior performance of SRE over the other metrics. The precision achieved by the seven metrics on the TREC corpus on list sizes between 100 and 10,000 with respect to UMLS is shown in Table 3.5 and displayed in a graphical form in Figure 3.8(a). The baseline for the precision was 0.49%. The greatest difference in comparison with Dias' ME being 44% absolute and 151.72% relative in the best case (n = 100, see Appendix B). The mean difference was 13.65% absolute and 45.34% relative. A t-test and a Wilcoxon Rank test with a confidence level of 99% reveal that the precision achieved by SRE is significantly better than that of ME ($p < 10^{-6}$).

On the BMC corpus, SRE also significantly outperformed ME (t-test and Wilcoxon Rank test, significance level of 99%, $p < 10^{-6}$). With respect to precision, the greatest difference in comparison with ME was 383.33% relative and 46% absolute (n = 100, see Figure 3.8(c) and Table 3.5). In terms of recall, the greatest difference was 2.19% absolute (26.54% relative, n = 9500).

η	DICE		FREQ		ME		PMI		SCP		SRE		TFIDF	
	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC
100	0	0	1.00	1.00	14.00	6.00	0	0	1.00	1.00	**33.00**	**15.00**	4.00	11.00
1,000	1.10	0	1.80	1.80	16.10	11.40	0.50	0	1.00	0.10	**26.60**	**13.00**	2.10	3.60
2,000	0.80	0	2.30	1.95	16.10	10.15	0.45	0	0.80	0.05	**24.40**	**11.30**	2.10	3.00
3,000	0.97	0	2.33	1.93	15.03	9.87	0.60	0	0.90	0.03	**21.70**	**10.57**	1.80	3.20
4,000	0.93	0	2.50	2.10	14.68	9.75	0.63	0	0.95	0.03	**20.88**	**10.15**	1.88	2.94
5,000	0.88	0	2.58	2.36	14.34	**9.32**	0.62	0	0.90	0.02	**19.50**	9.26	1.58	3.07
6,000	1.90	0	2.78	2.35	13.80	**9.02**	0.57	0	1.78	0.02	**18.43**	8.93	1.42	3.03
7,000	1.67	0	2.80	2.46	13.27	**8.71**	0.57	0	1.60	0.01	**17.44**	8.60	1.37	2.88
8,000	1.54	0	2.91	2.48	12.90	**8.48**	0.54	0	1.42	0.01	**16.35**	7.70	1.53	2.83
9,000	1.61	0	2.94	2.47	12.52	**8.24**	0.61	0	1.52	0.01	**15.81**	7.47	1.54	2.83
10,000	1.66	0	3.00	2.35	12.25	**8.01**	0.64	0	1.60	0.01	**15.15**	7.20	1.52	2.77
100	0	0	0.01	0	0.10	0.05	0	0	0.01	0.01	**0.23**	**0.12**	0.01	0.09
1,000	0.08	0	0.13	0.14	1.15	0.90	0.04	0	0.07	0.01	**1.89**	**1.03**	0.06	0.29
2,000	0.11	0	0.33	0.31	2.29	1.61	0.06	0	0.11	0.01	**3.47**	**1.79**	0.11	0.48
3,000	0.21	0	0.50	0.46	3.21	2.35	0.13	0	0.19	0.01	**4.63**	**2.52**	0.15	0.73
4,000	0.26	0	0.71	0.67	4.18	3.09	0.18	0	0.27	0.01	**5.94**	**3.22**	0.20	1.02
5,000	0.31	0	0.92	0.94	5.10	**3.70**	0.22	0	0.32	0.01	**6.94**	3.67	0.21	1.17
6,000	0.81	0	1.19	1.12	5.89	**4.29**	0.24	0	0.76	0.01	**7.87**	4.25	0.23	1.46
7,000	0.83	0	1.39	1.36	6.61	**4.84**	0.28	0	0.80	0.01	**8.69**	4.78	0.26	1.68
8,000	0.88	0	1.66	1.57	7.34	**5.38**	0.31	0	0.81	0.01	**9.30**	4.90	0.33	1.83
9,000	1.03	0	1.89	1.76	8.02	**5.89**	0.39	0	0.97	0.01	**10.12**	5.33	0.38	2.02
10,000	1.18	0	2.13	1.86	8.72	**6.36**	0.46	0	1.14	0.01	**10.78**	5.71	0.41	2.20

Table 3.3: Precision and recall using MESH as the gold standard. *The upper section of the table shows the precision and the lower part the recall. In each pair of columns labeled with a metric, the left column shows the precision (resp. recall) obtained on TREC, while the right column shows the precision (resp. recall) on BMC. Values in bold font mark the best results.*

η	DICE		FREQ		ME		PMI		SCP		SRE		TFIDF	
	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC
100	3.00	0	6.00	5.00	18.00	6.00	0	0	1.00	2.00	**43.00**	**27.00**	0	7.00
1,000	1.90	0	4.40	2.30	19.30	9.80	1.30	0	2.30	0.80	**31.20**	**22.40**	0.80	3.30
2,000	1.65	0	3.85	2.45	18.65	9.90	1.35	0	2.25	0.40	**25.10**	**17.00**	1.45	2.95
3,000	1.90	0	3.93	2.10	16.93	10.00	1.33	0	2.06	0.27	**21.90**	**14.70**	1.20	2.90
4,000	2.13	0	4.08	2.10	15.95	9.45	1.43	0	2.13	0.20	**20.35**	**13.03**	1.45	2.34
5,000	2.10	0	4.12	1.92	15.04	8.82	1.44	0	2.02	0.16	**19.20**	**11.72**	1.28	2.28
6,000	3.92	0	4.17	1.91	14.33	8.68	1.53	0	3.68	0.13	**18.23**	**10.93**	1.22	2.30
7,000	3.61	0	3.99	1.99	13.70	8.33	1.57	0	3.44	0.11	**17.31**	**10.16**	1.20	2.26
8,000	3.41	0	3.89	1.96	13.06	8.09	1.50	0	3.13	0.10	**16.29**	**9.30**	1.38	2.28
9,000	3.67	0	3.72	1.88	12.64	7.82	1.57	0	3.51	0.09	**15.59**	**9.03**	1.51	2.24
10,000	3.96	0	3.74	1.86	12.36	7.67	1.59	0	3.65	0.08	**14.95**	**8.62**	1.51	2.24
100	0.02	0	0.04	0.04	0.11	0.04	0	0	0.01	0.01	**0.26**	**0.20**	0	0.05
1,000	0.11	0	0.26	0.17	1.16	0.71	0.08	0	0.14	0.06	**1.87**	**1.62**	0.05	0.24
2,000	0.20	0	0.46	0.36	2.24	1.43	0.16	0	0.27	0.06	**3.01**	**2.46**	0.17	0.43
3,000	0.34	0	0.71	0.46	3.05	2.17	0.24	0	0.37	0.06	**3.94**	**3.20**	0.22	0.63
4,000	0.51	0	0.98	0.61	3.83	2.74	0.34	0	0.51	0.06	**4.89**	**3.77**	0.35	0.75
5,000	0.63	0	1.24	0.70	4.51	3.20	0.43	0	0.60	0.06	**5.76**	**4.25**	0.38	0.85
6,000	1.41	0	1.50	0.83	5.16	3.78	0.55	0	1.33	0.06	**6.57**	**4.75**	0.44	0.99
7,000	1.52	0	1.67	1.01	5.76	4.22	0.66	0	1.45	0.06	**7.27**	**5.15**	0.50	1.14
8,000	1.64	0	1.86	1.14	6.27	4.68	0.72	0	1.50	0.06	**7.82**	**5.39**	0.66	1.32
9,000	1.98	0	2.01	1.22	6.83	5.10	0.85	0	1.90	0.06	**8.42**	**5.89**	0.82	1.46
10,000	2.38	0	2.24	1.35	7.42	5.56	0.95	0	2.19	0.06	**8.97**	**6.25**	0.90	1.62

Table 3.4: Precision and recall using SNOMED-CT as the gold standard. *The upper section of the table shows the precision and the lower part the recall. In each pair of columns labeled with a metric, the left column shows the precision (resp. recall) obtained on TREC, while the right column shows the precision (resp. recall) on BMC. Values in bold font mark the best results.*

η	DICE		FREQ		ME		PMI		SCP		SRE		TFIDF	
	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC
100	4.00	0	4.00	4.00	29.00	12.00	1.00	0	4.00	6.00	**73.00**	**58.00**	4.00	19.00
1,000	4.90	0	5.80	3.70	35.40	21.60	4.80	0	5.90	2.20	**59.90**	**41.00**	3.60	6.40
2,000	5.20	0	5.65	4.05	34.10	22.00	4.20	0.05	5.75	1.15	**51.85**	**34.75**	3.90	6.05
3,000	5.40	0	5.97	4.00	32.43	21.07	4.03	0.03	5.43	0.77	**46.47**	**31.00**	3.23	5.83
4,000	5.45	0.03	6.03	4.23	31.00	20.30	4.20	0.03	5.55	0.58	**43.35**	**28.28**	3.30	5.50
5,000	5.60	0.02	6.48	4.32	29.84	19.46	4.02	0.02	5.42	0.48	**40.88**	**26.02**	2.88	5.30
6,000	8.65	0.02	6.62	4.18	28.83	19.22	4.18	0.02	8.28	0.40	**38.68**	**24.18**	2.55	5.17
7,000	7.93	0.03	6.44	4.17	27.74	18.31	4.17	0.01	7.76	0.37	**36.76**	**22.97**	2.47	5.20
8,000	7.50	0.04	6.36	4.11	26.83	18.80	4.16	0.01	7.16	0.38	**34.80**	**21.10**	2.83	4.93
9,000	7.97	0.03	6.34	4.06	26.31	17.89	4.37	0.08	7.74	0.31	**33.53**	**20.26**	2.89	4.79
10,000	8.33	0.04	6.46	4.02	25.65	17.51	4.37	0.10	7.93	0.31	**32.15**	**19.53**	2.94	4.79
100	0.01	0	0.01	0.01	0.10	0.04	0	0	0.01	0.02	**0.24**	**0.21**	0.01	0.07
1,000	0.16	0	0.19	0.13	1.18	0.77	0.16	0	0.20	0.08	**2.00**	**1.45**	0.21	0.23
2,000	0.35	0	0.38	0.29	2.28	1.56	0.28	0	0.38	0.08	**3.47**	**2.46**	0.26	0.43
3,000	0.54	0	0.60	0.43	3.26	2.24	0.40	0	0.55	0.08	**4.66**	**3.30**	0.32	0.62
4,000	0.73	0	0.81	0.60	4.15	2.88	0.56	0	0.74	0.08	**5.80**	**4.01**	0.44	0.78
5,000	0.94	0	1.08	0.77	4.99	3.45	0.67	0	0.91	0.09	**6.84**	**4.61**	0.48	0.94
6,000	1.74	0	1.33	0.89	5.79	4.09	0.84	0	1.66	0.09	**7.77**	**5.14**	0.51	1.10
7,000	1.86	0.01	1.51	1.04	6.50	4.67	0.98	0	1.82	0.09	**8.61**	**5.70**	0.58	1.29
8,000	2.01	0.01	1.70	1.17	7.18	5.19	1.11	0	1.92	0.10	**9.32**	**5.98**	0.76	1.40
9,000	2.40	0.01	1.91	1.29	7.92	5.71	1.31	0.02	2.33	0.10	**10.10**	**6.46**	0.87	1.53
10,000	2.79	0.01	2.16	1.43	8.58	6.21	1.46	0.04	2.65	0.11	**10.75**	**6.92**	0.98	1.70

Table 3.5: Precision and recall using UMLS as the gold standard. *The upper section of the table shows the precision and the lower part the recall. In each pair of columns labeled with a metric, the left column shows the precision resp. recall obtained on TREC, while the right column shows the precision and the recall on BMC. Values in bold font mark the best results.*

3.4.4 Further Evaluations

Most of the metrics considered in the context of terminology extraction are mono-contextual. As SRE proved to outperform mono-contextual metrics, we compared the combination of four of these metrics in two different fashions, i.e., division and multiplication, over two contexts, i.e., documents and sentences. For a given sequence $w = c_1c_2...c_n$ of any length n greater than 0, let

- $d(w)$ be the number of documents within which w occurs;
- $f(w)$ be the number of sentences, in which w occurs;
- $p(w)$ be the probability that a sequence of length n is w;
- w' be a sequence with same length as w such that the Hamming distance between w and w' is 1.
- $p_d(w) = d(w)/\sum_{w'} d(w')$

We considered the following mono-context metrics using solely sentences as context:

$$tf(w) = f(w), \tag{3.16}$$

$$tme(w) = p(w)\frac{nf(w)}{\sum_{w'} f(w')}, \tag{3.17}$$

$$ts(w) = \frac{f(w)^n}{\prod_{i=1}^{n} f(c_i)}. \tag{3.18}$$

Analogously, the mono-context metrics using solely documents as context were as follows:

$$df(w) = d(w), \tag{3.19}$$

$$idf(w) = 1/d(w), \tag{3.20}$$

$$dme(w) = p_d(w)\frac{nd(w)}{\sum_{w'} d(w')}, \tag{3.21}$$

$$idme(w) = 1/dme(w), \tag{3.22}$$

$$ds(w) = \frac{d(w)^n}{\prod_{i=1}^{n} d(c_i)}, \tag{3.23}$$

$$ids(w) = 1/ds(w). \tag{3.24}$$

The multi-contextual metrics were composed by multiplying each of the scores achieved by a given sequence w using the first category of mono-context metric with the scores achieved using the second category. For the sake of clarity, we named the resulting metrics using the concatenation of the names of the metrics from which they issued as names (e.g., $tfidf(w) = tf(w) \cdot idf(w)$). We obtained 18 different multi-context metrics and 9 different mono-context metrics. The 28^{th} metric considered was SRE. We evaluated the precision of the metrics using the three gold standards MESH (see Figures 3.9 and 3.12), SNOMED-CT (see Figures 3.10 and 3.13) and UMLS (see Figures 3.11 and 3.14) on the TREC corpus. The metrics were divided into two groups: the first group contained the metrics obtained by multiplying the results obtained using mono-contextual metrics as well as the mono-contextual metrics df, dme, ds, tf, ts and tme (see Figures 3.9, 3.10 and 3.11). The second group was made up of the metrics obtained by dividing the results obtained using mono-contextual metrics as well as the mono-contextual metrics idf, $idme$, ids, tf, ts and tme (see Figures 3.12, 3.13 and 3.14).

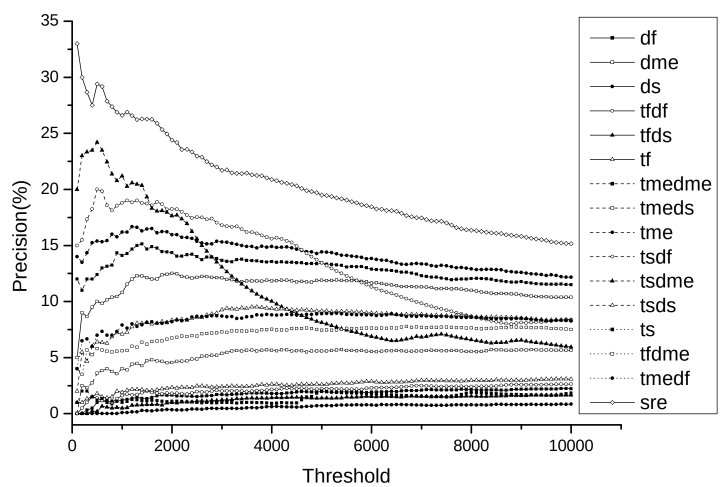


Figure 3.9: Precision of the first group of metrics using MESH

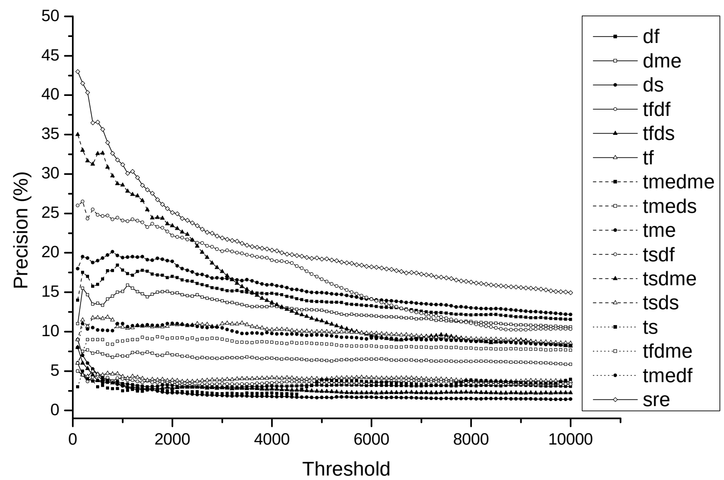


Figure 3.10: Precision of the first group of metrics using SNOMED-CT

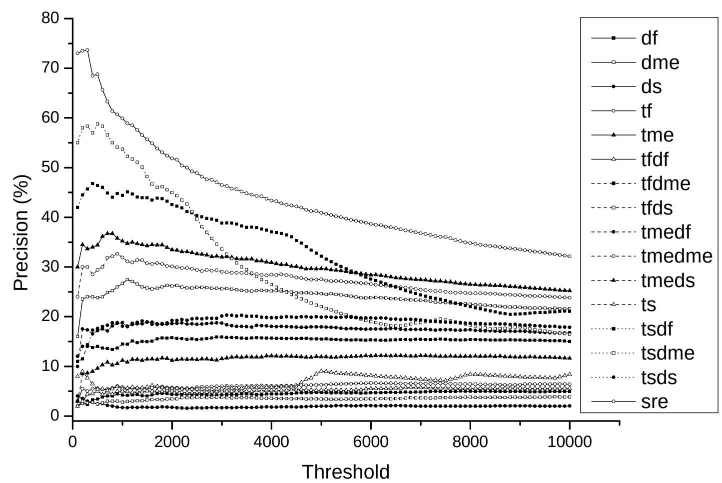


Figure 3.11: Precision of the first group of metrics using UMLS

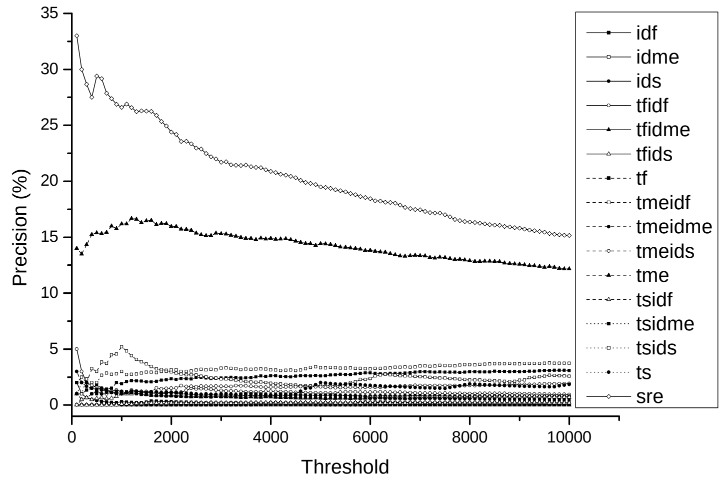


Figure 3.12: Precision of the second group of metrics using MESH

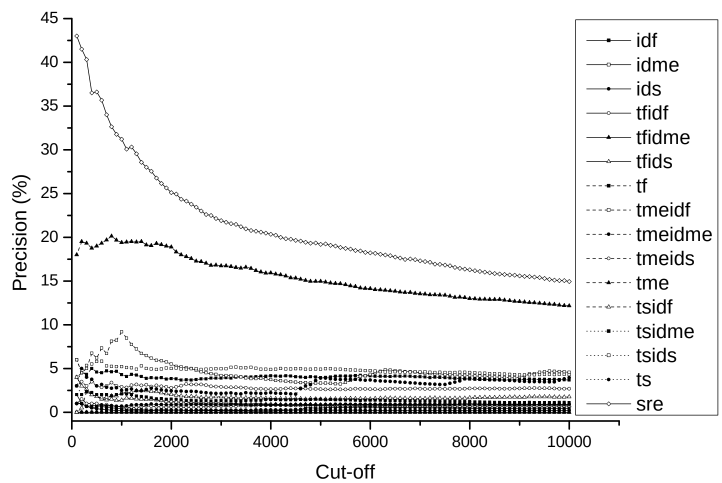


Figure 3.13: Precision of the second group of metrics using SNOMED-CT

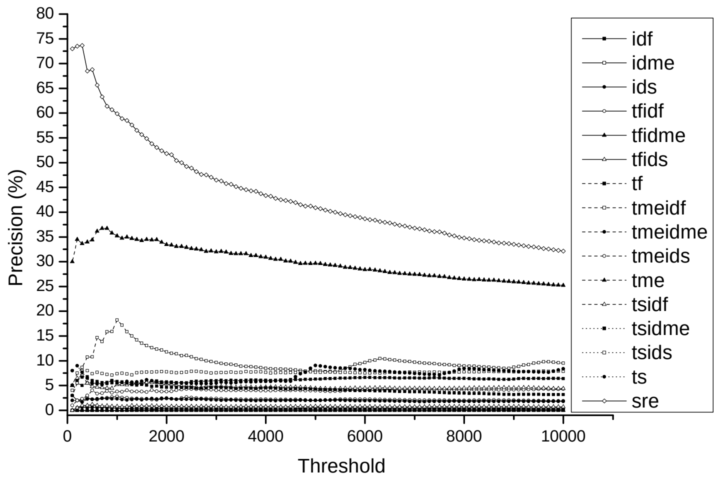


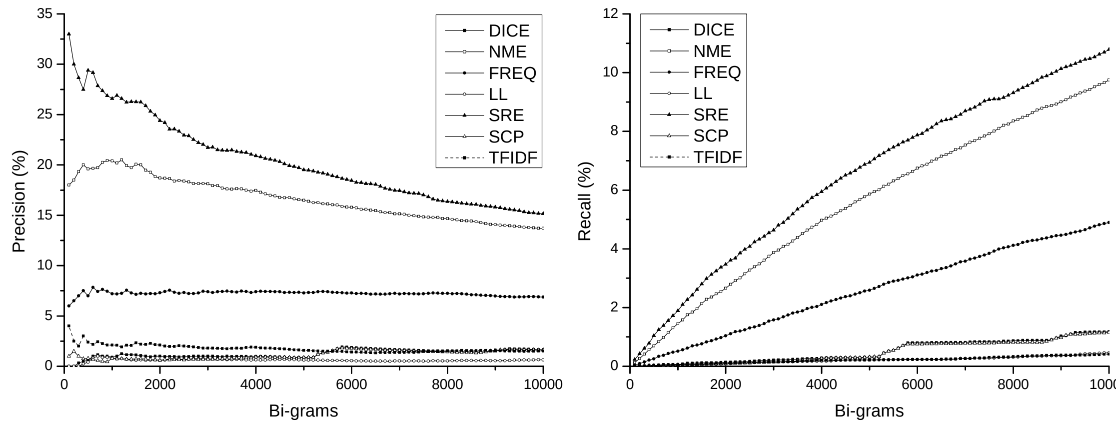
Figure 3.14: Precision of the second group of metrics using UMLS

In both groups, SRE significantly outperformed all other metrics (t-test and Wilcoxon Rank test, $p < 10^{-6}$). In the first group *tsdme* was the second best metric on the first 2,000 bigrams and was then outperformed by *tsdf*. *tsdme* outperforming *ts* and *dme* supports the idea that the combination of metrics over several context can improved the results obtained using the mono-contextual metrics on their own. The same holds for *tsdf*, *ts* and *df*. The results obtained on the second group display the importance of an appropriate model for domain-specificity. In the particular case of our evaluation, the use of the inverses worsened the precision of certain metrics, especially *tme*.

3.4.5 Discussion

The evaluation of SRE against other metrics on two corpora of different sizes and three gold standards of varying granularity provides some insights in three major areas: the appropriateness of several metrics for the extraction of domain-specific MWUs, the effect of corpus cleanness and corpus size on the behavior of the metrics and the effect of the gold standard's size on the precision and recall of metrics.

The appropriateness of SRE for the extraction of domain-specific MWUs is demonstrated by our evaluation. SRE outperforms the other metrics in our evaluation. In most cases, SRE is significantly superior to the other metrics with a confidence level of 99%. The difference between SRE and the other metrics may conceivably be explained by several factors. First, all metrics apart from SRE de-



(a) Precision with filtering (t = 10) (b) Recall with filtering (t = 10)

Figure 3.15: Precision and recall using filtering and MESH

pend solely on the relative distribution of words, not taking information on the global distribution and thus the higher specificity of domain-specific MWUs into consideration. Second, most of the other metrics are biased towards high frequencies. Therefore, they can not detect rare MWUs, which make up a high percentage of domain-specific MWUs as suggested by Zipf's law (Thanopoulos et al., 2002). The smoothing component of SRE lessens the effect of the patterns' frequency on their score. Schone (2001) filters MWUs containing very frequent words to improve his results. Filtering the bigrams containing the 10 most frequent terms from the results of all metrics partly improves their precision and recall. Especially, the mean of ME is improved by 2,50% in precision and 0.72% in recall when using MESH as the gold standard (see Figure 3.15(a) and 3.15(b)). A remarkable effect of the frequency filtering is the improvement of FREQ by 180,01% relative (4,64% absolute) in precision. Yet, the choice of the threshold for the acquisition of the best possible results remains a difficult task. Setting a threshold is made unnecessary by SRE, which yields constant results when applying the different thresholds, as can be seen in Figure 3.15(a) and 3.15(b).

Most metrics display a poorer precision on the noisy data set BMC, although it is larger. Interestingly, only the TFIDF norm seems not to be affected by the more noisy corpus and presents a gain in precision and recall when being used to process a larger corpus. Generally, the results (i.e., the precision and recall) obtained on the less noisy corpus TREC suggest that the use of manually edited corpora should be preferred to that of larger yet more noisy corpora for practical applications. Furthermore, they suggest that SRE is best suited for corpora containing documents of small size (i.e., documents of the size of abstracts).

The precision and recall obtained using different gold standards differ widely. When using the very restrictive gold standard MESH, some of the bigrams recognized by SRE as belonging to the domain of biomedicine are counted as false positives. The evaluation using more complete terminologies such as SNOMED-CT and especially UMLS show that SRE clearly outperforms all other metrics in the MWU extraction task. An exhaustive table containing all the evaluation data can be found in Appendix B.

The results of most MWU extraction are a list of n-grams with weighings, which yet do not reveal which terms actually belong to the domain-specific vocabulary of the corpus at hand. Extracting the domain-specific lexicon of the corpus will be the aim of the next chapter.

Chapter 4

Extraction of Domain-Specific Lexica

The aim of this chapter is the extraction of the domain-specific terminology based on our method for the extraction of domain-specific MWUs. Lexicon extraction (also called terminology extraction) is an essential step in many fields of NLP, especially when processing domain-specific corpora (Thelen and Riloff, 2002; Widdows and Dorow, 2002). A lexicon is defined as "the vocabulary of a [...] branch of knowledge" (Pearsall, 2001, p. 1061), i.e., of a domain. Most of the current algorithms for terminology extraction are knowledge-driven and use approaches such as differential analysis combined with statistical measures for the extraction of domain-specific vocabularies (Maedche and Staab, 2000; Drouin, 2004; Witschel, 2004). These methods work well when large and well-balanced reference corpora exist in the language to process. Yet, such datasets are available only for a few of the more than 6,000 languages currently in use on the planet. The need is, thus, for low-bias approaches to terminology extraction. In this chapter, we propose the use of a binary graph clustering algorithm for the purpose of lexicon extraction. The input graphs are generated using techniques for the extraction of MWUs.

This chapter is structured as follows: first, we use the results of the Chapter 3 to extract several graph categories, of which each represents the vocabulary of the corpus from which the underlying results were computed. In a second step, we analyze the topological characteristics of the graphs extracted in the prior step. Subsequently, we present a novel, general-purpose, binary graph clustering algorithm for terminology extraction called SIGNUM. We use SIGNUM for the extraction of domain-specific lexica. The resulting lexica are evaluated using the same reference data as in Chapter 3. Finally, three time-efficient approaches to the extraction of high-degree MWUs based on the results of SIGNUM are presented and compared.

The results of this chapter are used as vocabulary for the concept extraction approach presented in Chapter 5. Parts of the results presented in this chapter were published in (Ngonga Ngomo, 2008b).

4.1 Graph Representation for n-Grams

From a graph-theoretical point of view, the results of SRE can be interpreted as the sequential representation of a weighted directed graph. The labels of the nodes of this graph are the words contained in the vocabulary of the corpus. The weight of the edge between from a node with label u to a node with label v is a function of the scores obtained by the n-grams containing u and v. Several graph topologies arise when taking a closer look at the different possibilities for generating graphs out of n-gram results. In this chapter, we will focus especially on the use of

- *simple graphs*, i.e., graphs that can be directly computed out of the results of the MWU extraction process and for which $E(G) \subseteq [V(G)]^2$ or $E(G) \subseteq V(G) \times V(G)$ holds and on
- *link graphs* (Schütze, 1998; Dorow, 2006), which we will generate out of simple graphs.

Simple graphs allow the immediate extraction of domain-specific terminology because their nodes are labeled with words. On the other hand, link graphs enable the extraction of word meanings in context and, thus, the detection of their contextual belonging to the lexicon. As these graphs differ in their complexity, the following section will present and characterize both graph topologies.

4.1.1 Simple Graphs

A simple graph G on a set of words W is a graph such that each of its nodes can be labeled with exactly one word from W[1]. A simple graph G may be either directed or undirected, weighted or unweighted. Weighted directed graphs arise naturally from n-gram score list for two reasons: first, n-grams are ordered n-tuples. Therefore, they allow to define the direction of edges between nodes. Second, the scores achieved by n-grams can be conceived as edge weights. Let

- $G = (V, E, \omega)$ be a weighted directed graph,

[1] For the sake of brevity, we will use nodes and node labels interchangeably henceforth, as the mapping between node and node label is bijective.

- W the set of distinct words contained in the n-grams,
- L the set of n-grams $c_1 \dots c_n$ ($c_i \in W, 1 \leq i \leq n$) occurring in the corpus to analyze,
- $s : L \rightarrow \mathbb{R}$ a function which assigns a score to each n-gram and
- $L_{c_1 \dots c_m}$ be the subset of L that contains all n-grams of which $c_1 \dots c_m$ is a subsequence ($c_i \in W, 1 \leq i \leq m, m \leq n$):

$$L_{c_1 \dots c_m} := \left\{ l_1 \dots l_n \in L : \exists i \in \{0, \dots, n-1\} : c_1 \dots c_m = l_{i+1} \dots l_{i+m} \right\}. \quad (4.1)$$

We define the graph G by the following equations:

$$V = W, \quad (4.2)$$

$$\forall uv \in E \ \omega(uv) = \sum_{l \in L_{uv}} s(l). \quad (4.3)$$

The weight of each edge uv in the graph is cumulative, i.e., it is the sum of the scores of all n-grams of which uv is a subsequence. In the special case of bigram graphs, the weight of each edge uv is a function of the score of the associated bigram uv. Figure 4.1 shows an example of such a graph centered around the word *ion*.

When using the construction proposed above, n-grams are represented as paths in the graph G. For values of n above 2, the pairwise linking of all words in significant n-grams serves the purpose of lexicon extraction better. The relation displayed by the edges of the graph is then the co-occurrence of words in n-grams. The new weighing function ω' then looks as follows:

$$\forall u, v \in V \ \omega'(uv) = \omega'(vu) = \sum_{l \in L_u \cap L_v} s(l) \quad (4.4)$$

The graphs obtained using Equation (4.4) are undirected due to the symmetry of the co-occurrence relation. The relation between Equation (4.4), which generates undirected graphs and Equation (4.3), which generates directed graphs, is as follows:

$$\forall u, v \in V \ \omega'(uv) = \omega(uv) + \omega(vu) - \sum_{l \in L_{uv} \cap L_{vu}} s(l). \quad (4.5)$$

Applied to bigram graphs, where $\forall u, v \in V, u \neq v \rightarrow L_{uv} \cap L_{vu} = \emptyset$, ω' is then

$$\forall u, v \in V \ \omega'(\{u, v\}) = \omega(uv) + \omega(vu), \quad (4.6)$$

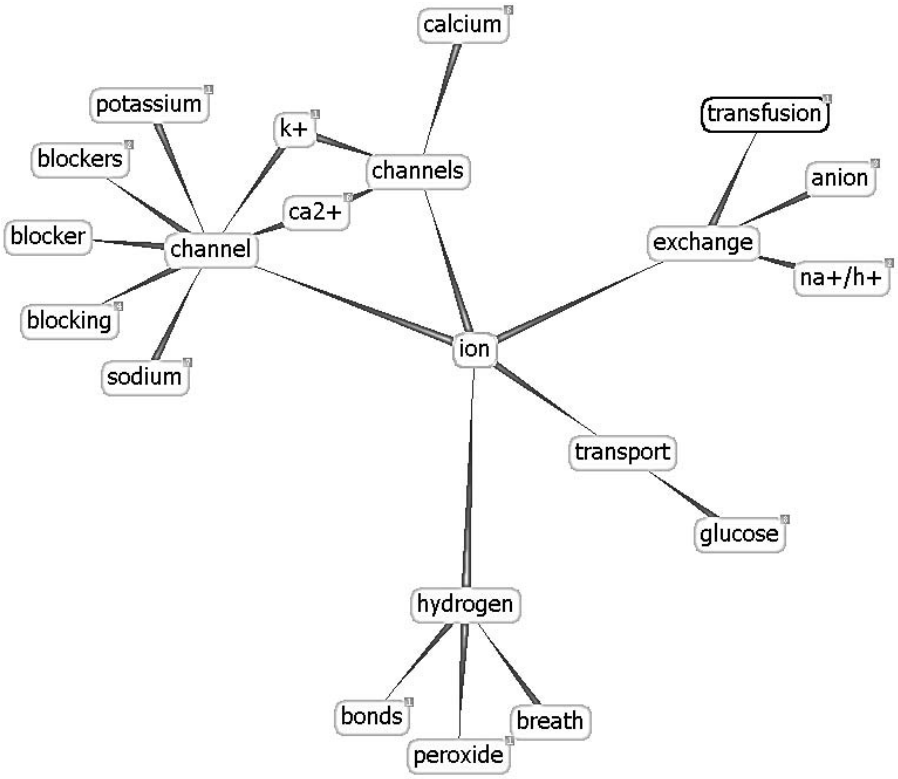


Figure 4.1: Bigram graph for ion. *The numbers at the upper right corners of nodes state the number of non-displayed neighbors. The graph was generated using the first 10,000 entries of the output of SRE on the TREC corpus. The length of the edges is inversely proportional to their weight. The score function is set to* $-1/log_{10}(SRE)$.

allowing a direct transformation from directed to undirected graphs.

Independently from their being directed or not, simple graphs display the same highly disconnected topology (Dorow, 2006). Some topological characteristics of simple graphs generated out of bigrams are shown in Table 4.1. In general, simple graphs consist of a large main component and smaller satellite components (as the average node/component shows). It is important to notice that the n-grams with the best scores tend to be included in the largest component of the graph.

4.1.2 Link Graphs

A main issue in NLP is lexical ambiguity, which designates the property of terms to bear more than different meanings depending on the context in which they oc-

Bigrams	Nodes	Edges	Components	Avg. N/C	Avg. E/C	Max. N/C	Max. E/C
5,000	6,593	4,990	1,812	3.64	2.75	1,647	1,807
10,000	11,106	9,969	2,606	4.26	3.83	4,854	6,282
20,000	23,733	19,939	7,415	3.20	2.69	7,136	10,688
50,000	47,905	49,685	14,579	3.29	3.41	13,895	30,204
100,000	79,658	98,893	21,454	3.71	4.61	25,315	65,811

Table 4.1: Topology of undirected bi-gram graphs generated out the TREC corpus. *Avg. N/C stands for average number of nodes per component, Avg. E/C for average number of edges per component, Max. N/C for maximal number of nodes per component and Max. E/C for maximal number of edges per component.*

cur (Manning and Schütze, 1999). In the context of lexicon discovery, ambiguity can be conceived as the property of words to belong to a domain-specific lexicon solely in combination with other words (polysemy). For example, while *acid* depicts a substance with a pH value less than 7 in its most common sense, it also depicts a variant of house music when combined with the word *house* (i.e., building the domain-specific MWU *acid house*). Thus, while neither *acid* nor *house* would improbably be added into a lexicon of music, *acid house* would.

A graph-based approach to the extraction of polysemes lies in the use of link graphs (Schütze, 1998; Dorow, 2006). Link graphs are generated from simple graphs in two steps. First, the edges of the simple graph are collapsed to nodes. The second step consists of adding an edge uv between the pairs of nodes (u, v) of the linkgraph, whose elements u and v represents edges that shared common nodes in the original simple graph. An example of a simple graph and the resulting link graph is shown in Figure 4.2.

While simple graphs provide a lexicon of domain-specific words, link graphs allow the discovery of domain-specific word combinations. This is partially achieved by the context-dependent disambiguation that link graphs accomplish inherently. An example of such a disambiguation is shown in Figure 4.3, in which the two meanings of *mercury* are split due to the different contexts in which they appear.

Schütze (1998) and Dorow (2006) use undirected graphs for the detection of word senses. Yet, as the graphs considered in this section can be directed, an extension of the approach described by both authors to directed graphs is needed. Let $G = (V, E, \Omega)$ be a directed weighted graph. We define the link graph $L(G)$ of the graph G is a graph such that:

$$V(L(G)) = E(G) \tag{4.7}$$

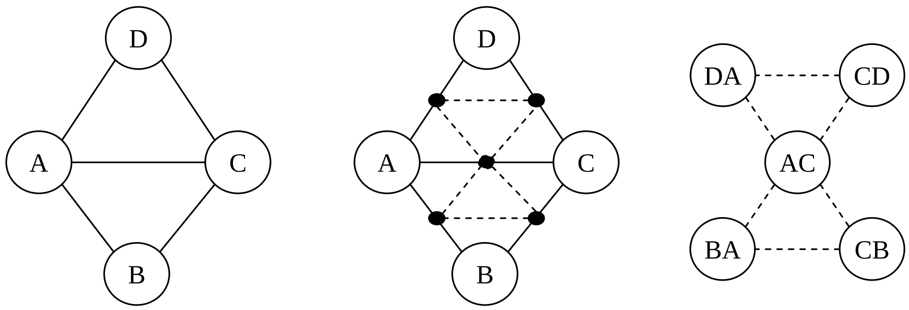
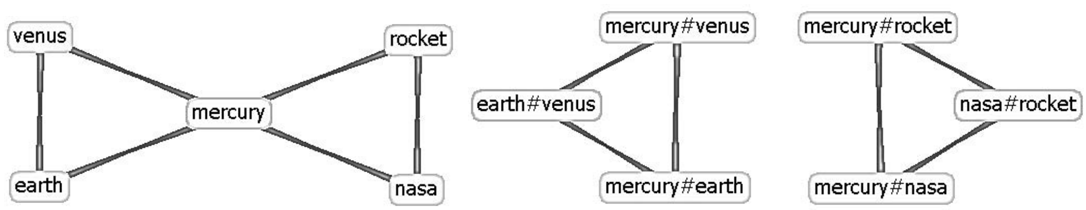


Figure 4.2: A simple graph and the resulting link graph. *The original simple graph (leftmost side of the picture) is neither directed nor weighted. The resulting link graph is displayed on the rightmost side. The image in the middle shows the nodes of the link graph (black dots) in the original graph.*



(a) Initial graph

(b) Resulting Link graph

Figure 4.3: Disambiguation of "mercury". *The two meanings of mercury (planet and rocket) contained in the original graph are not connected in the resulting link graph.*

and

$$E(L(G)) = \{xy | x \in E(G) \wedge y \in E(G) \wedge (\exists u, v, w \in V(G) : \\ x = uv \wedge y = vw \wedge wu \in E(G))\}. \tag{4.8}$$

Instead of using undirected triangles, our extension uses directed triangles exclusively. Hence, it allows to generate directed triangles out of directed graphs. The downside of our definition is its restrictiveness. It accepts exclusively cycles of length 3. Therefore, it extracts only 2 components out of the graph generated using the best 10,000 bigrams computed using SRE on the TREC corpus (see Table 4.2) for example. The following weaker approach accepts all triangles (i.e., even undirected) at the cost of losing the property of direction in the link graph:

$$E(L(G)) = \{\{x, y\} | \exists u, v, w \in V(G) : \\ x \in \{uv, vu\} \wedge y \in \{vw, wv\} \wedge (\{wu, uw\} \cap E(G) \neq \emptyset)\}. \tag{4.9}$$

Independently from its being directed or not, we define the weighing function Ω on the link graph L(G) as:

$$\forall x, y \in E(L(G)) : \Omega(xy) = \omega(x)\omega(y) \tag{4.10}$$

The function Ω preserves the weighing and symmetry properties of G.

A potential hindrance to the use of link graphs is their worst-case complexity. Let $g := |G|$. If G is undirected, i.e., if the nodes uv and vu of $L(G)$ are equivalent,

$$|L(G)| \leq \frac{g(g-1)}{2} \in O(g^2). \tag{4.11}$$

Else

$$|L(G)| \leq g(g-1) \in O(g^2). \tag{4.12}$$

In both cases, the growth of the graph is quadratic in the worst case. Hence, the link graph of a graph containing approx. 100,000 nodes can contain up to $(10^5)^2 = 10^{10}$ nodes. The same worst case growth holds for the number of edges:

$$|E(L(G))| \leq \frac{|L(G)(|L(G)| - 1)}{2} \in O(g^4) = O(|E(G)|^2). \tag{4.13}$$

	Directed link graph		Undirected link graph	
Bigrams	TREC	BMC	TREC	BMC
5,000	0	9	196	36
10,000	6	21	662	112
20,000	40	39	1,896	286
50,000	367	99	11,098	896
100,000	4,221	153	43,438	902

Table 4.2: Number of nodes in directed and undirected link graphs. *The left column under each graph configuration displays the number of nodes obtained when processing the TREC corpus. The right column displays the same for the BMC corpus.*

In practical applications, the size of link graphs varies considerably depending on whether they are directed or not. Table 4.2 shows the number of terms included in the link graphs generated using Equation (4.8) (directed link graph) and Equation (4.9) (undirected link graph) on bigrams. This topology is similar to that reported by other groups (see e.g., (Dorow, 2006)). A comparison of the worst case size of the link graphs and their actual size hints toward polysemes as discovered by link graphs being seldom in the top n-grams of the corpus at hand.

4.2 SIGNUM

SIGNUM is a local graph clustering algorithm that makes use of the topological characteristics of small-world graphs for the extraction of domain-specific lexica based on graphs extracted out of n-grams. The idea behind SIGNUM is based on two characteristics of domain-specific terms. First, terms from the same domain tend to occur in the same paradigmatic context (Manning and Schütze, 1999). Thus, the predecessors and successors of domain-specific words can be assumed as potentially belonging to the same lexicon. Second, co-occurrence graphs display small-world characteristics (Ferrer-i-Cancho and Sole, 2001; Steyvers and Tenenbaum, 2005). This property of co-occurrence graphs makes them particularly suitable for graph clustering algorithms. Especially, the small mean path length between nodes allows the use of algorithms that necessitate exclusively local information for clustering because the transfer of local information from one node to all other nodes of the graph occurs considerably faster than in purely random graphs (Milgram, 1967). The main advantage of clustering approaches that use local information lies at hand: they are computationally cheap and can thus deal with very large graphs, such as those usually generated in the context of NLP.

4.2.1 Formal Specification

SIGNUM is designed to achieve a binary clustering of graphs. The basic idea behind SIGNUM originates from the spreading activation principle, which is being used in several areas such as neural networks (Picton, 2000) and information retrieval (Baeza-Yates and Ribeiro-Neto, 1999): the simultaneous propagation of information across edges. In the case of SIGNUM, this information consists of the classification of the predecessors of each node in one of the two classes dubbed $+$ (positive signum) and $-$ (negative signum). Each propagation step consists of simultaneously assigning the predominant class of its predecessors to each node. The processing of a graph using SIGNUM consists of three phases: the *initialization phase*, during which each node is assigned an initial class; the *propagation phase*, during which the classes are propagated along the edges and the *termination phase*, which stops the propagation when a termination condition is satisfied. The following specification of SIGNUM is carried out on directed weighted graphs because they encompass all other categories of simple graphs. Undirected graphs can be implemented as directed graphs G such that the following holds:

$$uv \in E(G) \rightarrow vu \in E(G). \tag{4.14}$$

Unweighted graphs can be considered as directed graphs with a constant weight function ω, i.e.:

$$\forall uv \in E, \omega(uv) = 1. \tag{4.15}$$

Phase I: Initialization

The goal of the initialization phase is the definition of the initial class of each node of the input graph. Directed weighted graphs are triplets $G = (V, E, \omega)$ with $E \subseteq V \times V$ and $\omega : E \rightarrow \mathbb{R}$. Let

$$\sigma : V \rightarrow \{+, -\} \tag{4.16}$$

be a function, which assign vertices a positive or negative signum. The goal of the initialization phase is the complete definition of the initial values of σ, i.e., the definition of the value $\sigma(v)$ initially returns for each v node in V. Depending on the field in which SIGNUM is used, the initial values of σ might differ. In the special case of terminology extraction, the information available about the edges is suitable for determining the initial values of σ. Thus, let

$$\sigma_e : E \rightarrow \{+, -\} \tag{4.17}$$

be a function which assigns a positive or negative signum to edges. The weight of the edge uv between two nodes u and v allows to approximate the degree to which the terms u and v belong to the domain of interest. Let σ_e be fully known. Furthermore, let

$$\Sigma^+(v) = \{u : uv \in E \wedge \sigma_e(uv) = +\} \tag{4.18}$$

and

$$\Sigma^-(v) = \{u : uv \in E \wedge \sigma_e(uv) = -\}. \tag{4.19}$$

The initial values of σ are then be given by:

$$\sigma(v) = \begin{cases} + & \text{if } \sum_{u \in \Sigma^+(v)} \omega(uv) \ > \sum_{v \in \Sigma^+(v)} \omega(uv); \\ - & \text{else.} \end{cases} \tag{4.20}$$

This initialization prioritizes one class (in this case the $-$ class). In the case of lexicon extraction, this implies that a term is considered as initially not belonging to the domain-specific lexicon when the evidence for its belonging to the lexicon equals the evidence for the opposite.

Phase II: Propagation

Each node is assigned the class resulting from the weighted vote of its predecessors. The class $-$ is assigned in case of a tie. Let σ_{new} be the signum assignment after a propagation step and σ_{old} the signum assignment before that step. Formally,

$$\sigma_{new}(v) = \begin{cases} + & \text{if } \sum_{\sigma_{old}(u)=+} \omega(uv) > \sum_{\sigma_{old}(u)=-} \omega(uv) \\ - & \text{else.} \end{cases} \tag{4.21}$$

Each edge is used exactly once during a propagation phase, making each propagation step linear in the number of edges. Furthermore, the re-assignment of the classes to the node occurs simultaneously, allowing SIGNUM to be easily implemented in a parallel architecture.

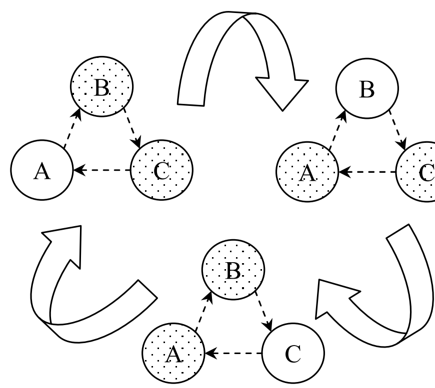


Figure 4.4: Example of non termination of SIGNUM. *Every edge has a weight of 1. The nodes without relief are assigned to +, else to −.*

Phase III: Termination

In the best case, SIGNUM terminates when the function σ remains constant. Yet, several graph configurations exist in which the propagation approach does not terminate. Figure 4.4 shows an example of such a configuration. Such examples appear rarely in real life data, due to the fact that co-occurrence graphs extracted from real world data are usually large and scale-free. Nevertheless, the need to ensure the termination of SIGNUM arises. Several means can be used to achieve this goal. The simplest mean consists of setting of an upper boundary $step_{max}$ for the maximal number of propagation steps. Another possibility resides in setting an upper boundary ϵ for the residual energy of the graph between two iteration steps. This solution has been successfully used in implementations of the Markov CLustering

(MCL) algorithm (van Dongen, 2000). The residual energy is computed as the max norm $||M - M'||_{max}$ of the difference of the weight matrices M before and M' after each iteration step. MCL terminates once this difference is less than or equal to a threshold ϵ.

4.2.2 Generalization to Hypergraphs

The approach followed by SIGNUM can be easily extended to hypergraphs. Let $H = (V, E, \omega)$ be a weighted hypergraph, where V is the set of vertices of H, the set of hyperlinks E is a subset of the power set $\wp(V)$ of V and ω is the weighing function (see Figure 4.5 for an example). A high-degree n-gram score list (with $n > 2$) can be interpreted as the specification of a n-uniform hypergraph H where

- the set of words W contained in the list is set to be $V(H)$,
- $E \subseteq \{M \in \wp(V) : |M| = n\} = [V]^n$ and
- $\omega(e)$ is a function of the scores of all n-grams which contain all elements of e.

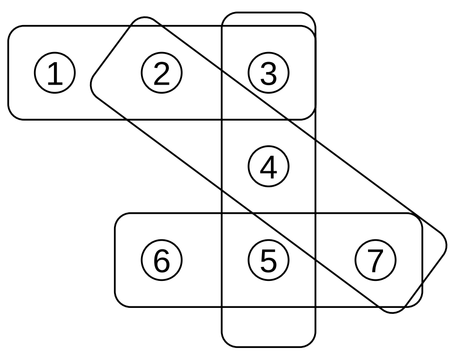


Figure 4.5: A 3-uniform hypergraph. *Nodes are presented by circles and edges by rounded rectangles. The set of nodes V =* {*1, 2, 3, 4, 5, 6, 7*}. *The set of edges E =* {{*1, 2, 3*}, {*3, 4, 5*}, {*6, 5, 7*}, {*2, 4, 7*}}.

The generalization of SIGNUM on hypergraphs follows the same three steps:

Phase I: Initialization

Let

$$\sigma : V \rightarrow \{+, -\} \tag{4.22}$$

be a function, which assign vertices a positive or negative signum. The initialization of SIGNUM based on the edge signum function σ_e can be carried out by redefining $\Sigma^+(v)$ and $\Sigma^-(v)$ as follows:

$$\Sigma^+(v) = \{e \in E : v \in e \wedge \sigma_e(e) = +\} \tag{4.23}$$

and

$$\Sigma^-(v) = \{e \in E : uv \in e \wedge \sigma_e(e) = -\}. \tag{4.24}$$

The initial values of σ are then be given by:

$$\sigma(v) = \begin{cases} + & \text{if } \sum_{e \in \Sigma^+(v)} \omega(e) \;\; > \sum_{e \in \Sigma^+(v)} \omega(e); \\ - & \text{else.} \end{cases} \tag{4.25}$$

Phase II: Propagation

The hypergraphs generated from n-gram score lists are not directed. Thus, each node v is assigned the signum of the majority of its neighbors. Let $u, v \in V$ with $u \neq v$. The propagation in hypergraphs follows the following equation:

$$\sigma(v) = \begin{cases} + & \text{if } \sum_{e \in E : v \in e} \sum_{u \in e \wedge \sigma(u) = +} \omega(e) > \sum_{e \in E : v \in e} \sum_{u \in e \wedge \sigma(u) = -} \omega(e) \\ - & \text{else.} \end{cases} \tag{4.26}$$

Phase III: Termination

The termination of runs on hypergraphs can be implemented similarly to that on simple graphs, i.e., when the function σ remains constant or when a manually defined termination condition (i.e., reaching a threshold for the number of iterations or for the residual energy) is satisfied.

4.3 Implementation Details

The current implementation of SIGNUM is based on sparse matrices as implemented in the COLT library (Hoschek, 2004). In a first step, each of the terms in the lexicon is indexed. The rows of the matrix M of dimension $m \times m$ are stored as a hash table mapping the index i of the corresponding node u to the set of entries $M(i, 1) \ldots M(i, m)$. This set is represented as a hash table mapping the index j of the nodes v to the weight $\omega(uv)$, i.e., the entry $M(i, j)$. Thus, the access to a value stored in M can be carried out in constant time. Only weight values differing from 0 are stored. We chose an implementation using matrices as implemented in COLT because it can be easily extended to hypergraphs, as they can be represented as high-dimensional matrices.

The initial signum function σ is computed out of the initial signum σ_e of the edges in three steps. First, the weights contained in the input score list are summed to a value β. In a second step, the signum of the edges σ_e is computed by cumulating the scores contained in the input score list sorted in a descending order. The cumulation is carried out until the threshold $\beta/2$ is reached. All n-grams whose weight were cumulated up to this point are assigned a positive signum. The remnant is assigned a negative signum. The assignment of the signum σ per se is the third step. It is implemented by multiplying the weight of each of the edges in the graph by the signum assigned to the corresponding n-gram. The signum of the nodes v with index j is then computed by cumulating the values $M(i, j)$, $1 \leq i \leq m$. In particular, the value 0 is considered to have a negative signum.

The propagation step consists of iteratively assigning a positive (resp. negative) weight to all edges whose source is a node with a positive (resp. negative) signum. After each iteration step, the resulting matrix M is compared with the matrix M' resulting from the previous iteration. If all entries $M(i, j)$ and $M'(i, j)$ bear the same signs, i.e., if the signum of all nodes is constant, all nodes with a positive signum are given out. Else, the fulfillment of the other termination condition is assessed. If it is met, all nodes with a positive signum are given out. To determine the thresholds for SIGNUM, we experimented with several upper boundaries for the number of iterations between 10 and 200. The results obtained with a maximal number of iteration above 50 did not show any significant alteration of precision or recall. The maximal number of iterations $step_{max}$ was therefore set to 50 in our experiments.

4.4 Experiments and Results

The experiments presented in this section were carried out based on the results obtained in section 3.4.

4.4.1 Experimental Setup

The scores computed using SRE bear small values for large corpora, ranging between 0 and 10^{-4} in the special case at hand. The weight $\omega(w_1 w_2)$ of the edge between two words w_1 and w_2 was thus set to

$$\omega(w_1 w_2) = \frac{-1}{log_{10}(SRE(w_1 w_2))}. \tag{4.27}$$

The function $-1/log_{10}$ is monotonically growing on the interval [0, 1[. Thus, it preserves the order in the n-gram list.

SIGNUM was evaluated on both the TREC and the BMC corpora. For this purpose, we used the n best scoring bigrams according to SRE (with n taking values between 5,000 and 100,000). The result of the clustering was the list of terms labeling nodes that were assigned a positive signum. We evaluated the results of SIGNUM on both simple graphs and link graphs. The baseline consisted of the results obtained using SRE. Our experiments on simple graphs were carried out using four graph configurations:

1. *Weighted directed graphs*: This graph configuration was generated using the information on direction (i.e., the sequence of occurrence of words in a n-gram) and SRE scores provided by the input score lists.

2. *Unweighted directed graphs*: In this configuration, all non-null weights contained in the first configuration were set to 1.

3. *Undirected weighted graphs*: The direction information contained in the first configuration was not considered in this configuration. The weights of the edges uv and vu were cumulated according to the transformation specified in Equation (4.5).

4. *Undirected unweighted graphs*: This graph configuration was generated out of undirected weighted graphs by replacing all non-null weights by 1.

Analogously, we generated four link graphs configurations out of the simple graphs according to Equation (4.8). We used three gold standards (MESH, SNOMED-CT and UMLS) to evaluate the results of SIGNUM.

4.4.2 Results

Results on Simple Graphs

On the TREC corpus, a considerable difference between the results of SRE and SIGNUM could be observed when the graph was generated out of 20,000 bigrams. However, the gain in precision then decreased with the size of the graph. This decrease in precision can be explained by the fact that larger graphs include more functions words, which tend to co-occur with terms from both classes and thus augment the total weight of the intra-cluster edges, leading to more errors as the class labels are transferred over the edges. This is especially clear, when the results obtained on the 100,000 bigram graphs are considered. In the case of the BMC corpus, the difference between the precision of SIGNUM and that achieved by SRE remained under that achieved on TREC on small graphs. On large graphs (especially on the 100,000 bigram graph) the difference in precision on BMC was greater than the difference on TREC.

In terms of recall, weighted undirected graphs proved to be the best configuration for lexicon extraction with SIGNUM on both corpora. The recall obtained by using SIGNUM depended more on the input graphs being directed or not than on their weighing (see Figure 4.7 and Table 4.4). Comparatively, SIGNUM achieved a better recall on the BMC corpus. Interestingly, the recall achieved on the largest graph (100,000 bigram) was almost equal (more than 97%) to the baseline.

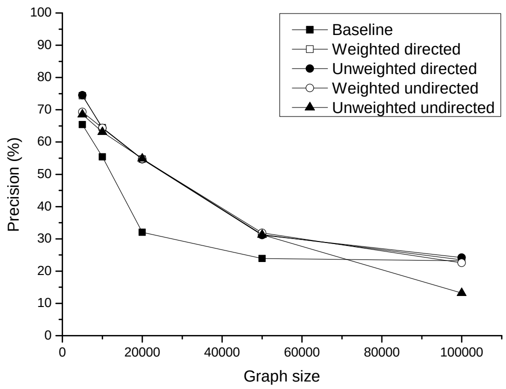
SIGNUM outperformed SRE in precision (see Figure 4.6 and Table 4.3). As expected, it achieved a lower recall than the original graph. SRE and consequently SIGNUM achieved a higher precision on the TREC data set as shown by a comparison of Figure 4.6(a) and 4.6(b), 4.6(c) and 4.6(d) and 4.6(e) and 4.6(f).

	Baseline		Weighted directed		Unweighted directed		Weighted undirected		Unweighted undirected	
N-grams	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC
5,000	65.41	35.53	74.36	39.13	**74.52**	39.09	69.30	**40.87**	68.57	39.47
10,000	55.38	28.43	**64.44**	34.91	64.29	34.67	64.33	**36.63**	63.11	35.44
20,000	32.05	18.58	54.71	27.07	54.63	26.89	54.78	**28.37**	**54.89**	27.72
50,000	23.91	10.22	31.34	17.25	31.18	17.13	**31.85**	**17.73**	31.26	17.39
100,000	23.23	5.04	23.52	10.19	**24.17**	101.9	22.54	10.38	13.21	**10.39**
5,000	69.19	39.81	79.46	44.45	**79.62**	44.31	72.09	**45.97**	71.93	44.57
10,000	60.32	32.24	**67.99**	39.04	67.79	38.84	67.73	**40.73**	66.90	39.74
20,000	36.07	21.38	59.32	30.71	59.17	30.56	**59.51**	**32.35**	59.28	31.76
50,000	27.50	11.86	35.44	20.03	35.26	19.86	**36.23**	**20.52**	34.85	20.14
100,000	25.75	5.87	26.91	11.77	**27.62**	11.76	25.64	12.05	16.21	**12.06**
5,000	87.06	58.69	90.56	62.69	**90.70**	62.55	89.07	**65.66**	88.82	64.48
10,000	80.24	49.27	87.31	57.18	**87.37**	56.94	87.16	**60.21**	86.37	59.18
20,000	52.99	34.34	79.89	46.05	79.77	45.86	**80.15**	**49.49**	79.69	48.54
50,000	46.89	20.11	**54.50**	31.71	54.29	31.55	**55.94**	**33.66**	52.62	33.08
100,000	44.35	10.09	**46.76**	19.09	46.40	19.09	45.17	20.43	39.44	**20.44**

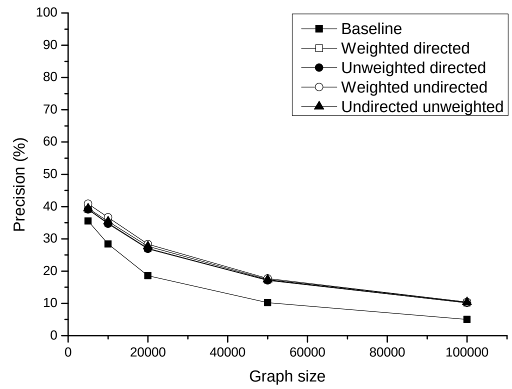
Table 4.3: Comparison of the precision of SRE and SIGNUM on simple graphs. *The upper, middle and lower section of the table show the precision obtained using MESH, SNOMED and UMLS respectively. The left column of each block under a graph configuration displays the precision obtained on the TREC corpus, while the right column displays the precision obtained on the BMC corpus.*

	Baseline		Weighted directed		Unweighted directed		Weighted undirected		Unweighted undirected	
N-grams	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC
5,000	8.62	11.07	1.17	3.21	1.17	3.19	**5.48**	**6.94**	4.45	5.90
10,000	12,27	16.73	7.55	5.38	7.30	5.31	**8.61**	**11.86**	6.81	10.03
20,000	15.21	21.65	11.63	8.31	11.54	8.21	**13.26**	**17.37**	11.99	15.29
50,000	22.95	28.45	15.91	13.34	15.69	13.20	**18.74**	**25.58**	15.38	23.80
100,000	30.76	28.73	21.06	16.83	18.41	16.83	**21.93**	**28.43**	13.90	**28.43**
5,000	6.54	10.04	0.90	2.95	0.89	2.93	**4.09**	**6.32**	3.35	5.39
10,000	9.61	15.36	5.72	4.88	5.52	4.81	**6.50**	**10.66**	5.17	9.10
20,000	12.38	20.15	9.04	7.63	8.96	7.55	**10.33**	**16.03**	9.28	14.18
50,000	18.89	26.73	12.90	12.53	12.71	12.39	**15.28**	**23.95**	12.29	22.30
100,000	25.74	27.13	17.29	15.72	15.08	15.72	**17.89**	**26.70**	11.71	**26.70**
5,000	1.59	4.27	0.20	1.20	0.20	1.19	**0.97**	**2.61**	0.80	2.25
10,000	2.46	6.77	1.41	2.06	1.37	2.04	**1.61**	**4.55**	1.29	3.91
20,000	3.49	9.34	2.35	3.30	2.33	3.27	**2.68**	**7.08**	2.41	6.26
50,000	6.21	13.07	3.82	5.73	3.77	5.68	**4.55**	**11.34**	3.58	10.57
100,000	9.76	13.45	5.78	7.36	4.88	7.37	**6.07**	**13.07**	4.12	13.06

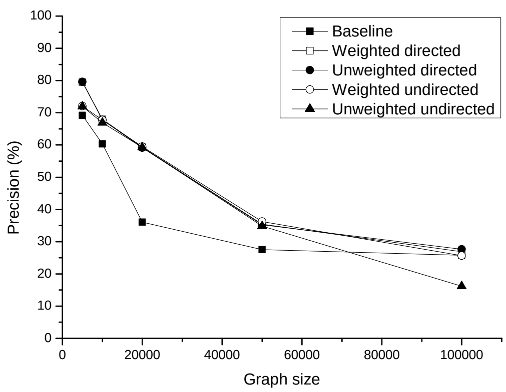
Table 4.4: Comparison of the recall of SRE and SIGNUM on simple graphs. *The upper, middle and lower section of the table show the recall obtained using MESH, SNOMED and UMLS respectively. The left column of each block under a graph configuration displays the recall obtained on the TREC corpus, while the right column displays the recall obtained on the BMC corpus.*



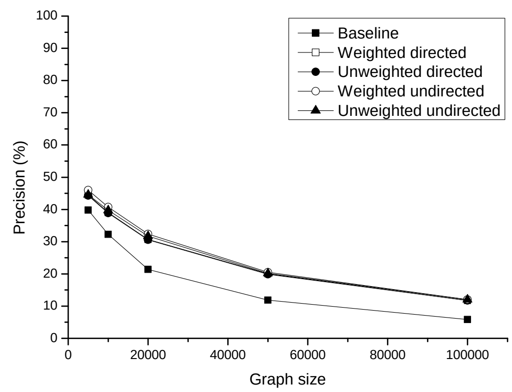
(a) Precision on TREC using MESH



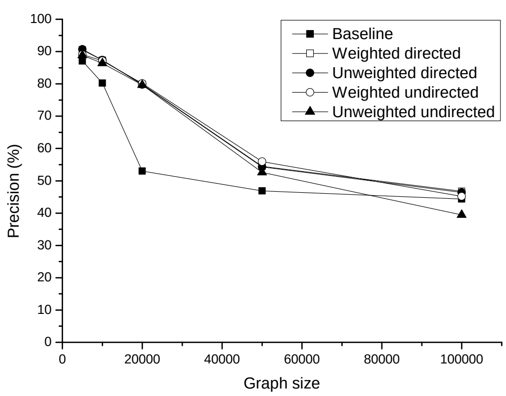
(b) Precision on BMC using MESH



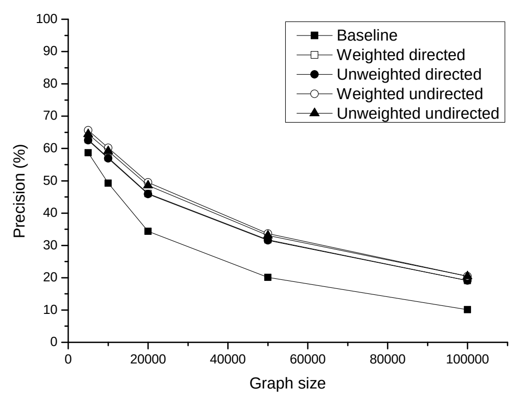
(c) Precision on TREC using SNOMED



(d) Precision on BMC using SNOMED

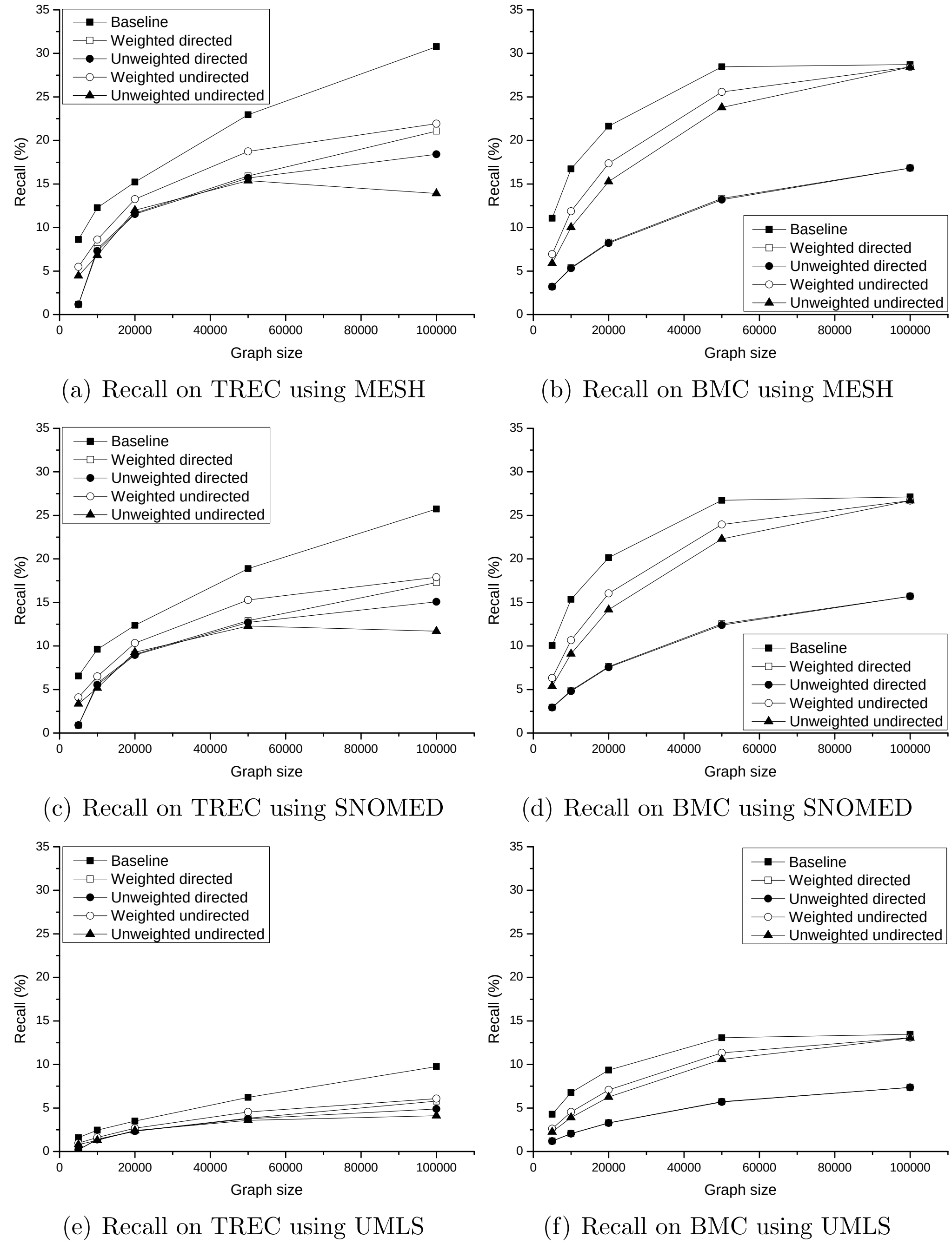


(e) Precision on TREC using UMLS



(f) Precision on BMC using UMLS

Figure 4.6: Precision achieved by SIGNUM on the TREC and BMC corpora. *The baseline was computed by measuring the precision obtained on the input graphs for SIGNUM.*



(a) Recall on TREC using MESH

(b) Recall on BMC using MESH

(c) Recall on TREC using SNOMED

(d) Recall on BMC using SNOMED

(e) Recall on TREC using UMLS

(f) Recall on BMC using UMLS

Figure 4.7: Recall achieved by SIGNUM on the TREC and BMC corpora. *The baseline was computed by measuring the recall obtained on the input graphs for SIGNUM.*

Results on Link Graphs

In order to evaluate SIGNUM on link graphs, we used each of the four possible simple graph configurations (i.e., directed weighted, directed unweighted, undirected weighted and undirected unweighted). Consequently, four categories of link graphs were considered. As expected, directed link graphs produced graphs of small size (see Table 4.2) and thus lead to low recall values. In particular, directed link graphs did not retrieve any relevant word combination on the BMC corpus.

	Weighted directed		Unweighted directed		Weighted undirected		Unweighted undirected	
N-grams	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC
5,000	–	0	–	0	**17.89**	0	**17.89**	0
10,000	**50.00**	0	**50.00**	0	17.87	**3.23**	17.08	0
20,000	7.41	0	7.41	0	15.50	6.18	**15.56**	**6.32**
50,000	3.23	0	3.30	0	9.65	6.90	**9.83**	**6.96**
100,000	3.03	0	3.62	0	5.46	**6.43**	**5.93**	**6.43**
5,000	–	0	–	0	**16.84**	0	**16.84**	0
10,000	**50.00**	0	**50.00**	0	14.01	4.84	13.61	**5.17**
20,000	3.70	0	3.70	0	10.84	5.62	**10.86**	**5.75**
50,000	2.15	0	2.20	0	7.83	4.23	**7.90**	**4.27**
100,000	2.33	0	2.51	0	4.71	3.76	**5.15**	**3.77**
5,000	–	0	–	0	**38.95**	0	**38.95**	0
10,000	0	0	0	0	**27.05**	1.61	25.99	**1.72**
20,000	3.70	0	3.70	0	21.89	**5.62**	**22.07**	5.17
50,000	4.30	0	4.40	0	15.74	8.15	**15.89**	**8.22**
100,000	5.29	0	5.54	0	9.55	**6.87**	**10.34**	**6.87**

Table 4.5: Precision of SIGNUM on link graphs. *The upper, middle and lower section of the table show the precision obtained using MESH, SNOMED and UMLS respectively. The left column of each block under a graph configuration displays the precision obtained on the TREC corpus, while the right column displays the precision obtained on the BMC corpus. The symbol – stands for link graphs of size 0.*

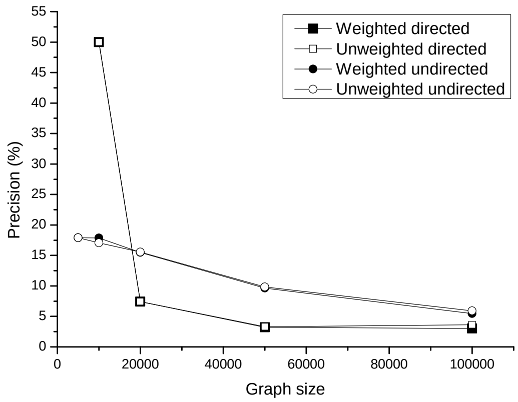
The precision obtained using both directed and undirected link graphs was inferior to the precision obtained using simple graphs, as shown by Figure 4.8 and Table 4.5, except on the 10,000 bigram graph extracted from the TREC corpus. This precision value was yet coupled with a recall of under 0.01%.

Overall, undirected weighted graphs outperformed the other configurations in our experiments. They achieved a higher recall (on the UMLS vocabulary (see Figure 4.9 and Table 4.6), which might appear counterintuitive, as UMLS is larger than the

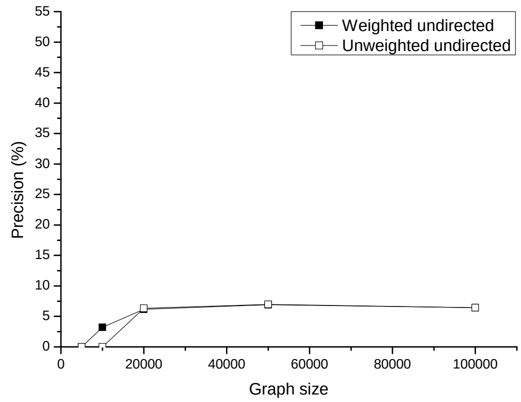
	Weighted directed		Unweighted directed		Weighted undirected		Unweighted undirected	
N-grams	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC
5,000	–	0	–	0	**0.046**	0	**0.046**	0
10,000	0.003	0	0.003	0	**0.200**	**0.016**	0.187	0
20,000	0.005	0	0.005	0	**0.603**	**0.087**	0.600	**0.087**
50,000	0.016	0	0.016	0	**1.641**	**0.349**	1.606	**0.349**
100,000	0.141	0	0.132	0	**4.664**	**0.460**	4.507	**0.460**
5,000	–	0	–	0	**0.096**	0	**0.096**	0
10,000	0.006	0	0.006	0	**0.348**	**0.022**	0.330	**0.022**
20,000	0.006	0	0.006	0	**0.936**	**0.072**	0.930	**0.072**
50,000	0.024	0	0.024	0	**2.953**	**0.196**	2.863	**0.196**
100,000	0.240	0	0.240	0	**8.943**	**0.246**	8.703	**0.246**
5,000	–	0	–	0	**0.124**	0	**0.124**	0
10,000	0	0	0	0	**0.375**	**0.004**	0.351	**0.004**
20,000	0.003	0	0.003	0	**1.054**	**0.035**	**1.054**	0.032
50,000	0.026	0	0.027	0	**3.312**	**0.184**	3.212	**0.184**
100,000	0.304	0	0.251	0	**10.108**	**0.220**	9.730	**0.220**

Table 4.6: Recall of SIGNUM on link graphs. *The upper, middle and lower section of the table show the recall obtained using MESH, SNOMED and UMLS respectively. The left column of each block under a graph configuration displays the recall obtained on the TREC corpus, while the right column displays the recall obtained on the BMC corpus. Due to the small recall values obtained, the recall values presented are shown up to 3 numbers after th comma. The symbol – stands for link graphs of size 0.*

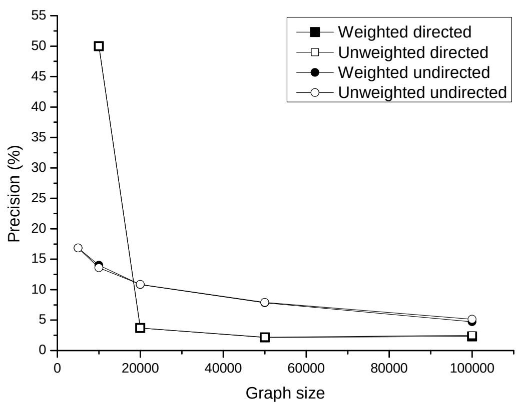
other two gold standards. The difference in recall seem to imply the terminology detected by link graphs belongs marginally to a domain-specific vocabulary. Further experiments in this area being out of the scope of this work and will be performed in a later stage. The recall and precision obtained when using link graphs being inferior to that obtained using simple, we will use exclusively simple graphs for the purpose of concept extraction.



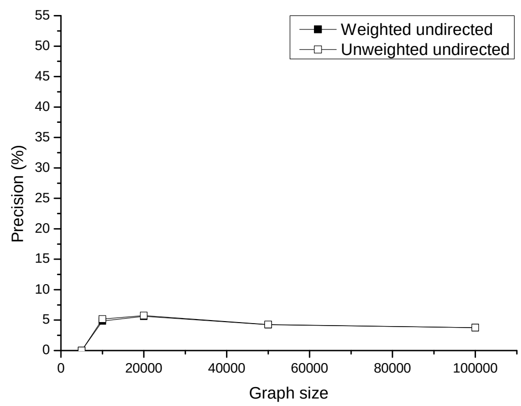
(a) Precision on TREC using MESH



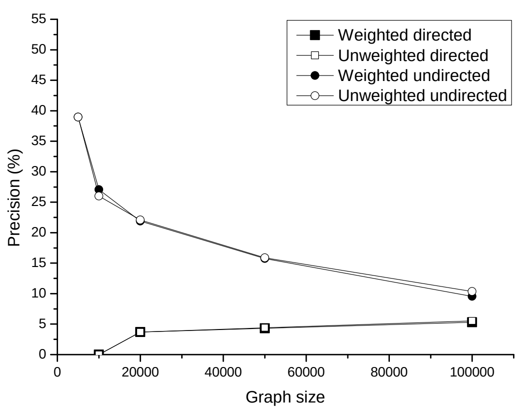
(b) Precision on BMC using MESH



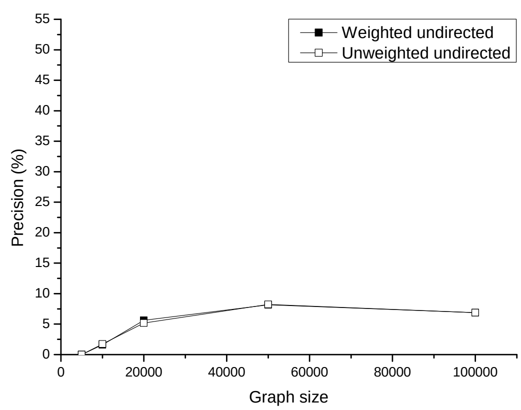
(c) Precision on TREC using SNOMED



(d) Precision on BMC using SNOMED

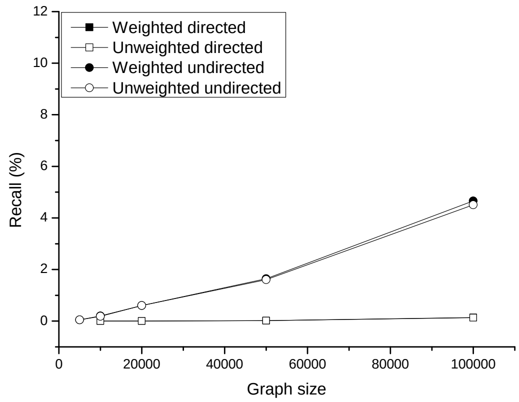


(e) Precision on TREC using UMLS

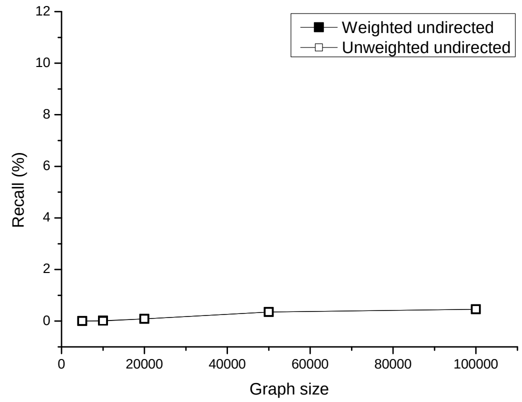


(f) Precision on BMC using UMLS

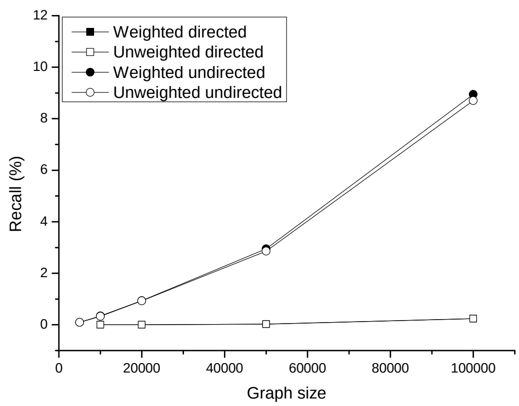
Figure 4.8: Precision achieved by SIGNUM link graphs issue from the TREC and BMC corpora. *The configurations omitted achieved a precision of 0.*



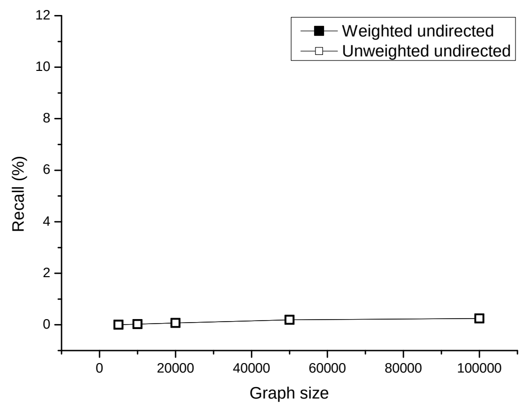
(a) Recall on TREC using MESH



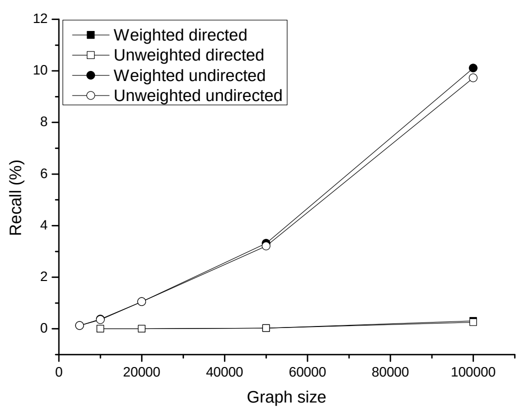
(b) Recall on BMC using MESH



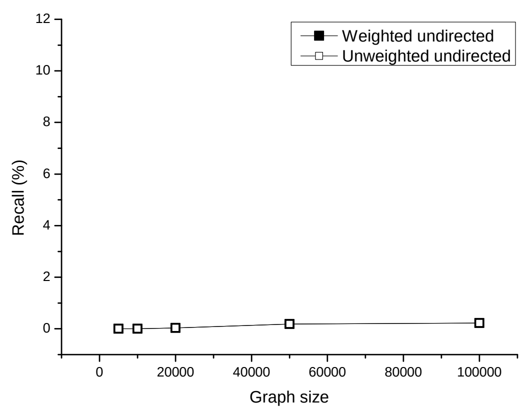
(c) Recall on TREC using SNOMED



(d) Recall on BMC using SNOMED



(e) Recall on TREC using UMLS



(f) Recall on BMC using UMLS

Figure 4.9: Recall achieved by SIGNUM link graphs issue from the TREC and BMC corpora. *The configurations omitted achieved a recall of 0.*

4.4.3 Discussion

Overall, the underlying graph configuration significantly altered the results obtained using SIGNUM. Weighted graphs generally achieved a slightly higher precision than their unweighted counterparts on both corpora. The difference in precision and recall between weighted and unweighted graphs was yet marginal. The performance of SIGNUM differed significantly depending on the underlying graphs being directed or undirected. This difference in precision and recall allows the assumption that the topology of the graph plays a more significant role than its weighing with respect to the precision and recall achieved by SIGNUM. This can conceivably be explained by the small-world characteristics of n-gram graphs. Due to the high clustering factor of these graphs, the class information can be spread along the whole graph in a small number of iterations. Independently from the edge weighing, a similar stable classification is achieved. This characteristic also explains the small number of iterations needed to reach constant recall and precision values. In our experiments, SIGNUM always outperformed SRE (and thus the other metrics presented in Chapter 3) in precision when using undirected graphs. While the best configuration varied depending on the graph size on the TREC corpus, it was mainly the undirected weighted graph configuration on the BMC corpus.

Given sufficient large corpora and input graphs, SIGNUM is able to reliably detect domain-specific terms. In our experiments, SIGNUM achieved approximately 97.17% of the recall of the 100,000 bigram graph extracted out of and outperformed its precision by more than 102.5% relative (see Tables 4.4 and 4.4). Furthermore, SIGNUM converged faster on graphs computed out of the larger BMC corpus than on graphs of the same size generated out of the TREC corpus. Especially, it terminated after 3 iteration steps on the undirected weighted graph extracted out of the best 100,000 bigrams of the BMC corpus. In terms of precision, SIGNUM performed better on the smaller, manually processed TREC corpus. However, the results obtained on BMC were superior in terms of recall. This can be explained by the fact that the number of intra-cluster edges is less in graphs extracted from the TREC corpus, leading to a higher precision but also higher number of false negatives. On the other hand, the higher recall on the bigger corpus could be due to the more representative distribution of words in the corpus that allowed for more inter-cluster edges and thus for the detection of larger sets of true (but also false) positives.

The SIGNUM idea can be extended in several ways. The following variation on the propagation step might lead to a faster convergence:

$$\sigma_{new}(v) = \begin{cases} + & \text{if } \sum_{\sigma_{old}(u)=+} \omega(uv) > \sum_{\sigma_{old}(u)=-} \omega(uv), \\ - & \text{if } \sum_{\sigma_{old}(u)=+} \omega(uv) < \sum_{\sigma_{old}(u)=-} \omega(uv), \\ \sigma_{old}(v) & \text{else.} \end{cases} \tag{4.28}$$

A faster convergence may also be achieved by adding a weight decay parameter, leading to alterations of the matrices only when higher levels of evidence than in previous iteration steps are given (Gupta and Lam, 1998). Furthermore, SIGNUM can be extended to cluster graphs with an unknown number of classes, for example to detect semantic classes. In this case, the initialization needs to be modified by assigning the same unique class label to each clique or almost-clique of the graph. An algorithm implementing such a clique detection is discussed in (Ngonga Ngomo, 2006). SIGNUM can be easily modified to provide a classification based on known positive and negative examples. The initialization of the algorithm would then consist of two steps: in a first step, the nodes assigned to the examples would be initialized with their respective classes (i.e., + for the positive examples to be positive and analogously − for the negative ones). Then, in a second step, the rest of the graph would be initialized as discussed in the preceding sections. During the propagation phase, SIGNUM would not alter the class of terms with known signum. In the case of terminology extraction, the use of known positive and negative examples could be used for lexicon expansion.

4.5 Extraction of High-Degree n-Grams

The metric SRE presented in Chapter 3 is general enough to be used for the computation of n-grams of arbitrary size. Yet, this computation can be very time expensive for high-degree n-grams (i.e., n-grams with $n > 2$), as all n-grams need to be extracted from the data set at hand in order to compute their score. When considering a corpus containing 10^5 word forms, the extraction of 3-gram using metrics can lead to the computation of up to $(10^5)^3 = 10^{15}$ trigrams. Several authors (Smadja, 1993; Thanopoulos et al., 2003) have proposed the use of agglomerative approaches to practically resolve this complexity problem.

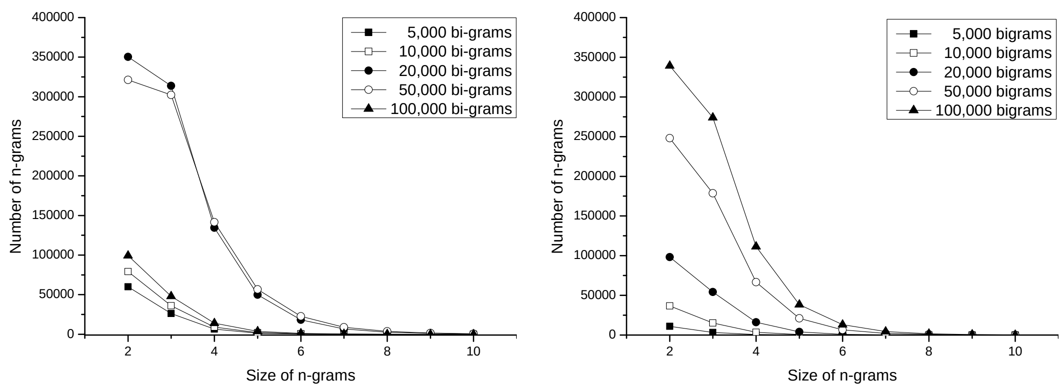
In this section, we use three approaches to extract high-degree n-grams. We present two linear baseline approaches to the extraction of high-degree n-grams based on previously extracted bigrams, namely the *lexicon-based* and the *overlap-based* approach. They demand exactly one pass on the data set to extract n-grams of any length. The drawback of these two approaches is their low precision. The third approach presented is an *agglomerative approach* based on SRE. It is more precise than the other approaches, yet can only extract up to 2^{k+1}-grams after k passes on the data set. The approaches are evaluated based on the lexica extracted by SIGNUM using weighted undirected graphs.

4.5.1 Lexicon-Based Approach

A computationally cheap technique for the extraction of high-degree n-grams consists of extracting all sequences consisting exclusively of terms characterized by SIGNUM as being domain-specific. Given the set W of terms w_i extracted by SIGNUM, each sequence

$$w_1 ... w_m : \forall i \in \{1 \ldots m\}, w_i \in W \tag{4.29}$$

which is found in the text corpus is then considered to be a MWU. The lexicon-based extraction presents the advantage of necessitating exactly one pass over the whole corpus to simultaneously extract MWUs of all lengths. Therefore, it is suitable to process very large text corpora.



(a) Distribution on the TREC corpus (b) Distribution on the BMC corpus

Figure 4.10: Distribution of n-grams extracted using the lexicon-based approach

The distribution of n-grams retrieved using this technique on the TREC and BMC corpus is displayed in Figure 4.10. The exact values are shown in Table 4.7. The lexicon-based approach easily detects domain-specific terms and their specializations. For example, it detects terms such as *arteriovenous malformations* and *intramedullary arteriovenous malformations*. However, this approach faces the problem of over-generation: it considers every random sequence of domain-specific terms as being a domain-specific term. Thus, it cannot differentiate sequences of domain-specific terms and MWUs from a single MWU. Therefore, it generates a relatively large number of long sequences and erroneously considers them to be domain-specific MWUs, leading to a poor precision.

Length	5,000 bigrams		10,000 bigrams		20,000 bigrams		50,000 bigrams		100,000 bigrams	
	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC
2	59,985	10,980	79,176	36,795	350,280	98,076	321,359	248,031	99,418	339,260
3	26,070	3,289	36,098	15,147	313,785	54,300	302,188	178,584	48,023	273,924
4	6,464	648	9,143	3,483	134,499	16,213	141,470	66,590	13,982	111,615
5	1,364	84	2,026	674	49,932	3,926	56,759	21,305	3,654	38,310
6	393	21	580	170	18,220	1,035	22,631	6,554	1,177	13,009
7	86	2	138	47	6,670	303	8,836	2,191	330	4,510
8	30	4	52	24	2,600	141	3,723	825	134	1,728
9	13	0	15	9	1,088	37	1,593	312	49	649
10	5	0	6	5	500	33	690	154	33	312

Table 4.7: Distribution of n-grams extracted using the lexicon-based approach on the TREC corpus

Length	5,000 bigrams		10,000 bigrams		20,000 bigrams		50,000 bigrams		100,000 bigrams	
	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC	TREC	BMC
2	2,807	2,775	5,671	5,698	13,143	11,244	36,896	27,052	54,730	49,217
3	45	59	189	160	823	361	4,936	1,016	6,677	1,666
4	5	16	26	38	122	112	892	348	1,375	682
5	1	5	3	17	19	43	117	134	248	295
6	0	4	0	13	3	29	22	68	58	165
7	0	2	0	4	0	9	7	32	15	107
8	0	3	0	4	0	8	0	24	5	85
9	0	1	0	2	0	3	0	12	1	38
10	0	0	0	0	0	5	0	14	0	49

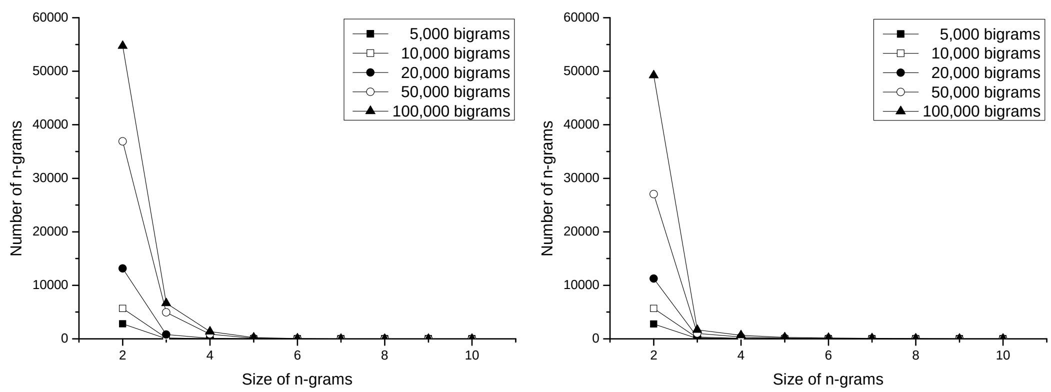
Table 4.8: Distribution of n-grams extracted using the overlap-based approach on the TREC corpus

4.5.2 Overlap-Based Approach

An approach to remedy the over-generation drawback of the lexicon-based approach lies in using sequences consisting of overlapping bigrams. Given the set W of bigrams $w_i w_{i+1}$ extracted by SIGNUM, each sequence

$$w_1 ... w_n \text{ with } \forall i \in \{1...n-1\}, w_i w_{i+1} \in W \tag{4.30}$$

is then be considered a domain-specific n-gram.



(a) Distribution on the TREC corpus (b) Distribution on the BMC corpus

Figure 4.11: Distribution of n-grams extracted using the overlap-based approach

The overlap-based approach is more restrictive than the lexicon-based approach and thus generates comparatively less long sequences (see Table 4.8 and Figure 4.11). Therefore, it can potentially lead to a higher precision. However, every sequence that satisfies Equation (4.30) also satisfies Equation (4.29). Hence, the recall of the overlap-based approach also remains inferior to that of the lexicon-based approach.

4.5.3 Agglomerative Approach

The lexicon-based and overlap-based approaches detect sequences composed of domain-specific terms or bigrams. Yet, they do not approximate the statistical relevance of these sequences. Therefore, they can not differentiate between domain-specific high-degree n-grams and sequences of domain-specific n-grams. The agglomerative approach (Smadja, 1993; Thanopoulos et al., 2003) detects high-degree n-grams in a bootstrapping fashion. Given the domain-specific terms extracted by SIGNUM from an initial graph, we first extract those bigrams from the initial graph which consist exclusively of domain-specific terms. Then, we replace every occurrence of these sequences in the corpus by a single token. Subsequently, we re-apply

SRE to the tokenized corpus. The resulting score list contains n-grams of length 2, 3 and 4. By using this agglomerative approach iteratively, n-grams of any given length can be extracted.

The agglomerative approach bears several advantages. First, it has a lower time and space complexity than the brute-force approach to extracting high-degree n-grams. Second, its precision can be improved by applying SIGNUM. Last, it can be combined with the LocalMax approach described in (Ferreira da Silva and Pereira Lopes, 1999; Dias et al., 1999b) to compute the best maximal length of n-grams.

4.5.4 Comparison

We evaluated the three approaches presented in this section on 3-grams and 4-grams. As input data, we used the results of SIGNUM for the lexicon-based and the overlap-based approach. The results of SIGNUM on the 20,000 bigram graph were used to evaluate the agglomerative approach because they displayed the highest absolute difference from the baseline in precision. The results obtained using the three approaches are therefore only really comparable on the 20,000 bigram graph. The lexicon-based approach can be considered to be a baseline approach yielding the highest recall possible.

Length	Lexicon		Overlap		Agglomerative	
	TREC	BMC	TREC	BMC	TREC	BMC
3-grams	1.19	1.81	5.39	0.55	6.30	2.98
4-grams	0.35	0.75	2.35	0	1.01	1.25
3-grams	8.80	2.80	0.11	0.01	2.27	0.62
4-grams	5.58	1.38	0.11	0	2.74	0.97

Table 4.9: Precision and recall on 3-grams and 4-grams. *The precision is displayed in the upper section of the table, the recall in the lower section.*

Measuring the precision and recall of high-degree n-grams proves to be a difficult task, since they are less frequently included in reference terminologies, due to the fact that they are often specializations of other termini. For example, the term *continuous ambulatory peritoneal dialysis* is considered to be a specialization of *peritoneal dialysis*. Furthermore, terminology from related domain such as *cox proportional hazards model* correctly occur in the list of retrieved high-degree n-grams but are counted as false positives, since they do not appear in the gold standards at hand. Therefore, the precisions computed using the gold standards do not reflect the absolute precisions achieved by the techniques evaluated in this section. Nevertheless, they can be used as a mean to compare the techniques. To limit the distortion of the precision and recall values for high-degree n-grams, we used the most complete gold standard (i.e., UMLS) for the evaluation. The precision and recall obtained on the TREC and BMC corpora are shown in Table 4.9

The agglomerative approach outperforms the overlap-based approach in both precision and recall (except on 4-grams on TREC). Compared with the lexicon-based approach, it always displays a lower recall but also a considerably higher precision.

Chapter 5

Concept Extraction

The extraction of concepts is the final step of this work. The computation of concepts demands a clustering algorithm that can efficiently deal with large graphs. In this section, we propose a novel clustering algorithm named BorderFlow. Similarly to other clustering algorithms (Jain et al., 1999), our algorithm is based on maximizing a criterion to extract clusters of high quality. It maximizes the intra-cluster similarity and inter-cluster dissimilarity simultaneously. This chapter is structured as follows: in the first section, we epitomize the idea behind our algorithm. Thereafter, we specify BorderFlow formally. Then, we evaluate BorderFlow's performance on synthetic graphs. Subsequently, we show that BorderFlow can efficiently cluster large scale-free graphs by using it to cluster graphs generated out of the Wikipedia Category Graph (WCG). Finally, we use our algorithm to cluster the domain-specific terminology extracted in the preceding chapter. We evaluate our clustering results quantitatively and qualitatively. The quantitative evaluation of the clusters is carried out against kNN (Tan et al., 2005) using the silhouette index (Rousseeuw, 1987). The qualitative evaluation of the clusters is carried out against the MESH taxonomy. We conclude the chapter by a discussion of our findings.

5.1 BorderFlow

BorderFlow is a general-purpose graph clustering algorithm. It uses solely local information for clustering and achieves a soft clustering of the input graph. The definition of cluster underlying BorderFlow was proposed by Flake et al. (2000). They state that a cluster is a collection of nodes that have more links between them than links to the outside. When considering a graph as the description of a flow system (van Dongen, 2000), Flake et al.'s definition of a cluster implies that a cluster X can be understood as a set of nodes such that the flow within X is maximal while the flow from X to the outside is minimal. The idea behind BorderFlow is to maximize the flow from the border of each cluster to its inner nodes (i.e., the nodes within the cluster) while minimizing the flow from the cluster to the nodes outside of the cluster. In the following, we will specify BorderFlow for weighted directed graphs, as they encompass all other forms of non-complex graphs.

5.1.1 Formal Specification

Let G = (V, E, ω) be a weighted directed graph with a set of vertices V, a set of edges E and a weighing function ω, which assigns a positive weight to each edge $e \in E$. In the following, we will assume that non-existing edges are edges e such that $\omega(e) = 0$. Before we describe BorderFlow, we need to define functions on sets of nodes. Let $X \subseteq V$ be a set of nodes. We define the set $i(X)$ of inner nodes of X as:

$$i(X) = \{x \in X | \forall y \in V : \omega(xy) > 0 \rightarrow y \in X\}. \tag{5.1}$$

The set $b(X)$ of border nodes of X is then

$$b(X) = \{x \in X | \exists y \in V \backslash X : \omega(xy) > 0\}. \tag{5.2}$$

The set $n(X)$ of direct neighbors of X is defined as

$$n(X) = \{y \in V \backslash X \ | \exists x \in X : \omega(xy) > 0\}. \tag{5.3}$$

In the example of a cluster depicted in Figure 5.1, $X = \{3, 4, 5, 6\}$, the set of border nodes of X is $\{3, 5\}$, $\{6, 4\}$ its set of inner nodes and $\{1, 2\}$ its set of direct neighbors.

Let Ω be the function that assigns the total weight of the edges from a subset of V to another one to these subsets (i.e., the flow between the first and the second subset). Formally:

$$\begin{aligned} \Omega : 2^V \times 2^V \rightarrow \mathbb{R} \\ \Omega(X, Y) = \textstyle\sum_{x \in X, y \in Y} \omega(xy). \end{aligned} \tag{5.4}$$

We define the border flow ratio $F(X)$ of $X \subseteq V$ as follows:

$$F(X) = \frac{\Omega\big(b(X), X\big)}{\Omega\big(b(X), V \backslash X\big)} = \frac{\Omega\big(b(X), X\big)}{\Omega\big(b(X), n(X)\big)}. \tag{5.5}$$

Based on the definition of a cluster by Flake et al. (2000), we define a cluster X as a node-maximal subset of V that maximizes the ratio $F(X)$[1], i.e.:

[1] For the sake of brevity, we shall utilize the notation $X + c$ to denote the addition of a single element c to a set X. Furthermore singletons will be denoted by the element they contain, i.e., $\{v\} \equiv v$.

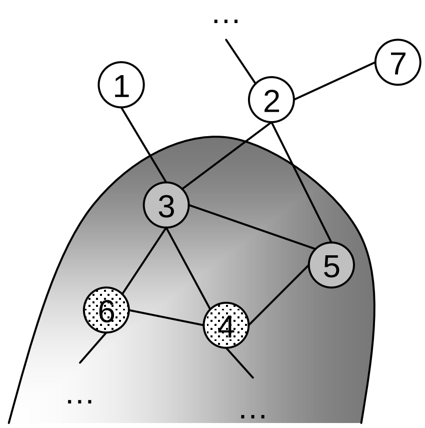


Figure 5.1: An exemplary cluster. *The nodes with relief are inner nodes, the grey nodes are border nodes and the white are outer nodes. The graph is undirected.*

$$\forall X' \subseteq V,\ \forall v \notin X : X' = X + v \rightarrow F(X') < F(X). \tag{5.6}$$

The idea behind BorderFlow is to select elements from the border $n(X)$ of a cluster X iteratively and insert them in X until the border flow ratio $F(X)$ is maximized, i.e., until Equation (5.6) is satisfied. The selection of the nodes to insert in each iteration is carried out in two steps. In a first step, the set $C(X)$ of candidates $u \in V \backslash X$ which maximize $F(X+u)$ is computed is as follows:

$$C(X) := \underset{u \in n(X)}{\arg\max}\ F(X+u). \tag{5.7}$$

By carrying out this first selection step, we ensure that each candidate node u which produces a maximal flow to the inside of the cluster X and a minimal flow to the outside of X is selected. The flow from a node $u \in C(X)$ can be divided into three distinct flows:

- the flow $\Omega(u, X)$ to the inside of the cluster,
- the flow $\Omega(u, n(X))$ to the neighbors of the cluster and
- the flow $\Omega(u, V\backslash(X \cup n(X)))$ to the rest of the graph.

Prospective cluster members are elements of $n(X)$. To ensure that the inner flow within the cluster is maximized in the future, a second selection step is necessary. During this second selection step, BorderFlow picks the candidates $u \in C(X)$ which maximize the flow $\Omega(u, n(X))$. The final set of candidates $C_f(X)$ is then

$$C_f(X) := \arg\max_{u \in C(X)} \; \Omega(u, n(X)). \tag{5.8}$$

All elements of $C_f(X)$ are then inserted in X if the condition

$$F(X \cup C_f(X)) \geq F(X) \tag{5.9}$$

is satisfied. Based on these two selection steps, BorderFlow can be implemented as described in Algorithm 1. Note that $C_f(X) = C(X)$ always holds when $|C(X)| = 1$. Therefore, the second selection step is only necessary when more than one element $u \in n(X)$ maximizes $F(X + u)$.

```
Data: Graph to cluster
Result: Fuzzy clustering
for each v ∈ V do
    X := {v};
    while |n(X)| > 0 do
        //computationally most expensive routine;
        C(X) := argmax_{u∈n(X)} F(X + u).;
        if (|C(X)| == 1 ∧ F(X ∪ C(X)) ≥ F(X)) then
            X := X ∪ C(X);
        else
            C_f(X) := argmax_{u∈C(X)} Ω(u, n(X));
            if (F(X ∪ C_f(X)) ≥ F(X)) then
                X := X ∪ C_f(X);
            else
                break;
            end
        end
    end
    store X;
end
merge all identical X;
return;
```

Algorithm 1: Naive implementation of BorderFlow

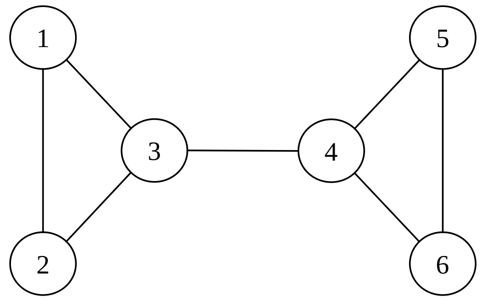


Figure 5.2: A simple graph containing two clusters

5.1.2 Exemplary Run

Let the input graph be as displayed in Figure 5.2 with constant weight function 1. It contains the two 3-cliques {1, 2, 3} and {4, 5, 6}. As it is symmetrical, we shall focus on the clustering when using 1 and 3 as seeds and use the symmetry of the graph to conclude on the final clustering generated by BorderFlow.

X={1}

- $X = \{1\} \rightarrow n(X) = \{2, 3\}$.

$$F(X+2) = \frac{\omega(12)+\omega(21)}{\omega(13)+\omega(23)} = 1.$$

$$F(X+3) = \frac{\omega(13)+\omega(31)}{\omega(12)+\omega(32)+\omega(34)} = 2/3.$$

  Thus $C(X) = 2$, which implies that $C_f(X) = 2$. Hence, $X := X + 2$. $F(X)$ is now 1.

- $X = \{1, 2\} \rightarrow n(X) = \{3\}$.

$$F(X+3) = \frac{\omega(31)+\omega(32)}{\omega(34)} = 2.$$

  Thus $C(X) = C_f(X) = \{3\}$. Hence, $X := X + 3$. $F(X)$ is now 2.

- $X = \{1, 2, 3\} \rightarrow n(X) = \{4\}$.

$$F(X+4) = \frac{\omega(43)}{\omega(45)+\omega(46)} = 1/2 < F(X).$$

The clustering thus stops with $X = \{1, 2, 3\}$. Due to the symmetry of the graph, initializing X with 2 leads to the same result. For the same reason, initializing X with 5 and 6 leads to $X = \{4, 5, 6\}$.

X={3}

- $X = \{3\} \rightarrow n(X) = \{1, 2, 4\}$.

$$F(X + 1) = \frac{\omega(13) + \omega(31)}{\omega(12) + \omega(32) + \omega(34)} = 2/3.$$

  Similarly, $F(X + 2) = 2/3$.

$$F(X + 4) = \frac{\omega(34) + \omega(43)}{\omega(31) + \omega(32) + \omega(45) + \omega(46)} = 1/2.$$

  Thus, $C(X) = \{1, 2\}$.

$$\Omega(1, n(X)) = \Omega(2, n(X)) = 1.$$

  Therefore, $C_f(X) = \{1, 2\}$. Hence, $X := X \cup \{1, 2\}$. $F(X)$ is now 2.

- $X = \{1, 2, 3\} \rightarrow n(X) = \{4\}$.

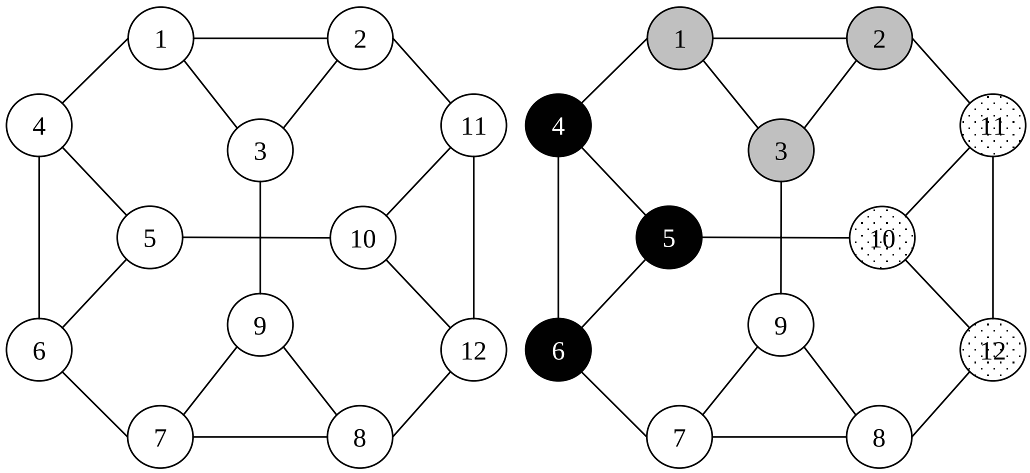
$$F(X + 4) = \frac{\omega(43)}{\omega(45) + \omega(46)} = 1/2 < F(X).$$

  The clustering ends here because $F(X + 4)$ is less than $F(X)$.

Due to the symmetry of the graph, we can conclude that the final clustering is {1, 2, 3}, {4, 5, 6} as expected.

5.2 Verification on Synthetic Graphs

In this section, we verify the correctness of the clustering achieved of BorderFlow by evaluating it on two synthetic graphs with known best clustering. The edge weights are all considered to be 1 if not stated otherwise. Furthermore, undirected edges are considered as representing two directed edges.



(a) A topped tetrahedron

(b) Natural clustering of a topped tetrahedron

Figure 5.3: A topped tetrahedron and its natural clustering

5.2.1 Clustering the Topped Tetrahedron

The topped tetrahedron (van Dongen, 2000) displayed in Figure 5.3(a) possesses a symmetrical structure and is thus difficult to cluster. Due to this symmetry, the nodes of a topped tetrahedron can be mapped to two topological equivalence classes $C_1 = \{1, 2, 4, 6, 7, 8, 11, 12\}$ and $C_2 = \{3, 5, 9, 10\}$. Hence, it is sufficient to explain the clustering of the graph based on nodes 1 and 3 to show that BorderFlow generates the adequate clustering shown in Figure 5.3(b).

$\mathbf{X = \{1\}}$

- $X = \{1\} \rightarrow n(X) = \{2,3,4\}$

$$\left.\begin{array}{ll} F(X+2) & = 1/2 \\ F(X+3) & = 1/2 \\ F(X+4) & = 1/2 \end{array}\right\} \rightarrow C(X) = n(X),$$

$$\left.\begin{array}{ll} \Omega(n(X),2) & = 1 \\ \Omega(n(X),3) & = 1 \\ \Omega(n(X),4) & = 0 \end{array}\right\} \rightarrow C_f(X) = \{2,3\}.$$

Hence, 2 and 3 are added to X, which is now $X = \{1,2,3\}$ with $F(X) = 2$.

- $X = \{1, 2, 3\} \rightarrow n(X) = \{4, 9, 11\}$

$$\left.\begin{aligned} F(X+4) &= 5/4 \\ F(X+9) &= 5/4 \\ F(X+11) &= 5/4 \end{aligned}\right\} \rightarrow \text{No further node is added to X, as } F(X) = 2.$$

X = {3}

- $X = \{3\} \rightarrow n(X) = \{1, 2, 9\}$

$$\left.\begin{aligned} F(X+1) &= 1/2 \\ F(X+2) &= 1/2 \\ F(X+9) &= 1/2 \end{aligned}\right\} \rightarrow C(X) = n(X),$$

$$\left.\begin{aligned} \Omega(n(X), 1) &= 1 \\ \Omega(n(X), 2) &= 1 \\ \Omega(n(X), 9) &= 0 \end{aligned}\right\} \rightarrow C_f(X) = \{1, 2\}.$$

  Thus, 2 and 3 are added to X, which is now $X = \{1, 2, 3\}$ with $F(X) = 2$.

- $X = \{1, 2, 3\} \rightarrow n(X) = \{4, 9, 11\}$

$$\left.\begin{aligned} F(X+4) &= 5/4 \\ F(X+9) &= 5/4 \\ F(X+11) &= 5/4 \end{aligned}\right\} \rightarrow \text{No further node is added to X, as } F(X) = 2.$$

Due to the symmetry of the topped tetrahedron, the clustering generated by BorderFlow is thus the partition $\{\{1, 2, 3\}, \{4, 5, 6\}, \{7, 8, 9\}, \{10, 11, 12\}\}$ as expected.

5.2.2 Clustering (m, k)-Partite-Cliques

We define a (m, k)-partite-clique ($m > k$, $k \geq 2$) as an undirected graph $G = (V, E)$ consisting of mk nodes and $mk(m + k - 2)/2$ edges such that its vertices can be partitioned into two categories of edge-disjoint cliques, namely k cliques containing exactly m nodes each and into m cliques containing exactly k nodes. Figure 5.4 shows an example of such a graph. Let $\zeta_m(v) \subset V$ be the clique of size m which contains v and $\zeta_k(v) \subset V$ be the clique of size k that contains v. Note that each node v has exactly $m - 1$ neighbors from $\zeta_m(v)$ and $k - 1$ neighbors from $\zeta_k(v)$, since all ζ_m and ζ_k are edge-disjoint. Thus, each node in a (m, k)-partite-clique has exactly $m + k - 2$ neighbors.

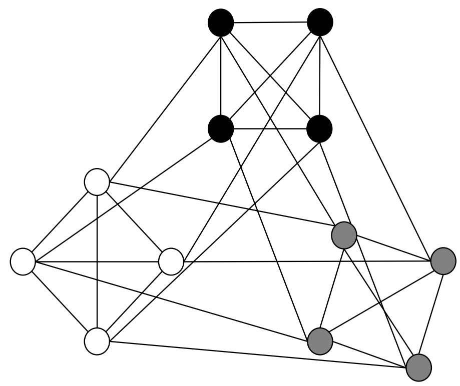


Figure 5.4: A (4,3)-partite-clique

The best clustering of a (m, k)-partite-clique consists of assigning each node v to the clique $\zeta_m(v)$, i.e., of partitioning V into the k sets $C_1 \dots C_k$ such that

$$\forall v \in V, \exists v_i \in \zeta_k(v) : C_i = \zeta_m(v_i). \tag{5.10}$$

To prove that BorderFlow generates the best possible clustering of (m,k)-partite-cliques, we first need to show that it assigns each node v to $\zeta_m(v)$. Then, we need to show that BorderFlow terminates after that step, i.e., that it does not add any other node to $\zeta_m(v)$. We show that $X = \{v\} \rightarrow C(X) = n(X)$ by proving the following lemma:

Lemma 5.2.1. $\forall v \in V, X = \{v\} \rightarrow (\forall v', v'' \in n(X)\ F(X + v') = F(X + v''))$

Proof. Each node v of $V(G)$ has exactly $k-1$ neighbors from $\zeta_k(v)$ and $m-1$ neighbors from $\zeta_m(v)$. Hence,

$$\forall X \subseteq V, |X| = 1 \rightarrow |n(X)| = m + k - 2. \tag{5.11}$$

Now let us add a node $u \in n(v)$ to X. The edge linking u to v and v to u is now an internal edge. All other edges remain unchanged. Two possibilities occur (see Figure 5.5):

Case 1: $u \in \zeta_m(v)$

In this case, the neighbors of $\{v, u\}$ are

1. the other $m-2$ elements of $\zeta_m(v)$ and
2. the elements of $\zeta_k(v)$ and $\zeta_k(u)$.

Thus,

$$\Omega(\{v, u\}, n(\{v, u\})) = 2(m-2) + 2(k-1) = 2(m+k-3). \tag{5.12}$$

Case 2: $u \notin \zeta_m(v)$

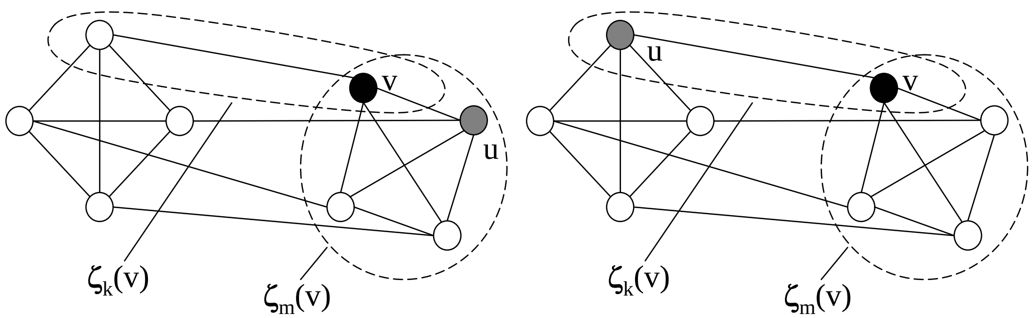
In this case, v and u are elements of the same k-clique $\zeta_k(v)$. Hence, the neighbors of $\{v, u\}$ are

1. the other $k-2$ elements of $\zeta_k(v)$ and

2. the elements of $\zeta_m(v)$ and $\zeta_m(u)$.

Hence,

$$\Omega(\{v,u\}, n(\{v,u\})) = 2(m-1) + 2(k-2) = 2(m+k-3). \tag{5.13}$$



(a) Case 1 (b) Case 2

Figure 5.5: Computation of the best candidates for addition in a cluster. *The input graph is a (4,2)-clique. The seed node v is the black node. In the first case, a node v' (grey node) from $\zeta_m(v)$ is considered for addition, whilst in the second case, v' is from $\zeta_k(v)$.*

In both cases, $F(\{v, u\})$ is

$$\frac{2}{2(k+m-3)} = \frac{1}{k+m-3} \tag{5.14}$$

and thus the same for all u. Thus $C(X) = n(X)$. □

Now that we have computed $C(X)$, we show that $C_f(X) = \zeta_m(v)$ by proving the following lemma:

Lemma 5.2.2. $\forall v' \in \zeta_m(v)\ \forall v'' \in \zeta_k(v)\ \Omega(X + v', n(X)) > \Omega(X + v'', n(X))$

Proof. We know that $u \in C(X) \rightarrow (u \in \zeta_m(v) \lor u \in \zeta_k(v))$. We can determine $\Omega(u, n(X))$ in both cases.

Case 1: $u \in \zeta_m(v)$

In this case, u is connected to all other $m-2$ elements of $\zeta_m(v)$. Thus,

$$\Omega(u, n(X)) = m - 2. \tag{5.15}$$

Case 2: $u \in \zeta_k(v)$

In this case, u is connected to all other $k-2$ elements of $\zeta_k(v)$. Thus,

$$\Omega(u, n(X)) = k - 2. \tag{5.16}$$

Since $m > k$, we can conclude that $\forall v' \in \zeta_m(v),\ \forall v'' \in \zeta_k(v) :\ \Omega(X+v', n(X)) > \Omega(X + v'', n(X))$. All elements of $\zeta_m(v)\backslash v$ are added to X. Hence, $X = \zeta_m(v)$ with

$$F(X) = \frac{m(m-1)}{m(k-1)} = \frac{m-1}{k-1}. \tag{5.17}$$

□

The last step of this proof consists of showing that no further node $u \in n(\zeta_m(v))$ can be added to $\zeta_m(v)$ without degrading the value of $F(X)$. To achieve this goal, we prove the following lemma:

Lemma 5.2.3. $\forall u \in n(\zeta_m(v))\ F(\zeta_m(v) + u) < F(\zeta_m(v))$

Proof. Two cases must be differentiated.

Case 1: $k = 2$

Adding u to $\zeta_m(v)$ causes $b(\zeta_m(v))$ to consist of the $m-1$ elements of $\zeta_m(v)$ and u (see Figure 5.6 for an example). Thus,

$$\Omega(b(\zeta_m(v) + u), \zeta_m(v) + u) = (m-1)^2 + 1. \tag{5.18}$$

The flow to the neighbors of $\zeta_m(v) + u$ consists of the flow from $m-1$ elements of the cluster to their neighbors in their respective ζ_2 and the flow from u to its neighbors in $\zeta_m(v)$. Thus,

$$\Omega(b(\zeta_m(v) + u), n(\zeta_m(v) + u)) = (m-1)(k-1) + (m-1) = 2(m-1). \tag{5.19}$$

Consequently, the border flow ratio of $\zeta_m(v) + u$ is given by

$$F(\zeta_m(v) + u) = \frac{(m-1)^2 + 1}{2(m-1)}. \tag{5.20}$$

The difference $\Delta F = F(\zeta_m(v) + u) - F(\zeta_m(v))$ is thus given by

$$\Delta F = \frac{(m-1)^2+1}{2(m-1)} - \frac{m-1}{2-1} = -\frac{m(m-2)}{2(m-1)}. \tag{5.21}$$

Note that $k = 2 \rightarrow m > 2$. Consequently,

$$\Delta F < 0. \tag{5.22}$$

No further node is added. Borderflow achieves the correct clustering.

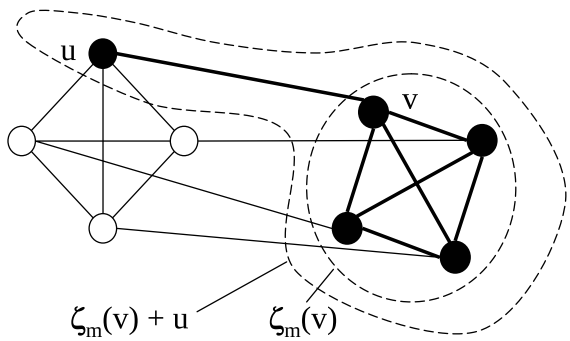


Figure 5.6: Addition of a node to a 4-clique in a (4,2)-clique. *The filled nodes and thick edges are the constituents of the* $\zeta_m(v) + u$*.*

Case 2: $k > 2$

Adding u to $\zeta_m(v)$ causes $b(\zeta_m(v))$ to consist of all m elements of $\zeta_m(v)$ and u (see Figure 5.7 for an example). The flow from $b(\zeta_m(v) + u)$ to $\zeta_m(v) + u$ is

$$\Omega(b(\zeta_m(v) + u), \zeta_m(v) + u) = m(m-1) + 2. \tag{5.23}$$

The flow to the neighbors of $\zeta_m(v)+u$ consists of the flow from the m elements of the cluster to their neighbors in their respective ζ_k and the flow from u to its neighbors in $\zeta_m(u)$. As two elements of $\zeta_k(u)$ are in the cluster, the flow to the neighbors is given by

$$\Omega(b(\zeta_m(v)+u), n(\zeta_m(v)+u)) = (m-1)(k-1)+2(k-2)+(m-1) = mk+k-4. \tag{5.24}$$

Thus, the border flow ratio of $\zeta_m(v) + u$ is given by

$$F(\zeta_m(v) + u) = \frac{m(m-1)+2}{mk+k-4} = \frac{m^2-m+2}{mk+k-4}. \tag{5.25}$$

The difference $\Delta F = F(\zeta_m(v) + u) - F(\zeta_m(v))$ is thus

$$\Delta F = \frac{m^2 - m + 2}{mk + k - 4} - \frac{m-1}{k-1} = -\frac{(m-3)(k+m-2)}{(k-1)(mk+k-4)}. \tag{5.26}$$

$k > 2$ implies that $m > 3$ and thus

$$\Delta F < 0. \tag{5.27}$$

No other node is added to $\zeta_m(v)$. BorderFlow achieves the best possible clustering.

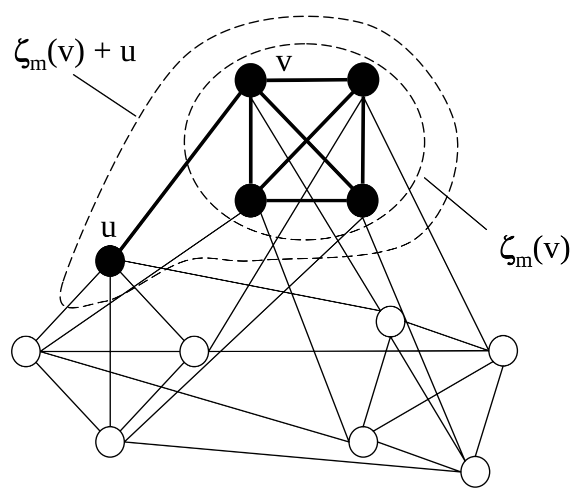


Figure 5.7: Addition of a node to a 4-clique in a (4,3)-clique. *The filled nodes and thick edges are the constituents of the $\zeta_m(v) + u$.*

□

BorderFlow converges fast for all graphs of this type, as it selects all nodes from the same clique in the first step and no other in the subsequent one.

5.3 A Heuristic for Maximizing the Border Flow Ratio

The implementation proposed above demands the simulation of the inclusion of each node in $n(X)$ in the cluster X before choosing the best ones. Such an implementation can be time-consuming as nodes in terminology graphs can have a high number of neighbors. The need is for a computationally less expensive criterion for selecting a nearly optimal node to optimize $F(X)$. In this section, we present a heuristic that enables BorderFlow to run more efficiently.

An efficient method for maximizing $F(X)$ is to iteratively maximize its alteration when a node $v \in n(X)$ is added to X. We define the difference

$$\Delta F(X,v) = F(X+v) - F(X). \tag{5.28}$$

Let $d(X,v)$ be the set of elements of the border of X that will not belong to the border of $X+v$:

$$d(X,v) = \{x \in b(X) | x \in i(X+v)\}. \tag{5.29}$$

Two possibilities can occur when adding a node v to the cluster X:

Case 1: $v \notin b(X+v)$

In the example depicted in Figure 5.1, this case would occur if the node 1 was added to the cluster. In this case $b(X+v) = b(X) \backslash d(X,v)$. Thus:

$$\Delta F(X,v) = \frac{\Omega(b(X),X) + \Omega(b(X),v) - \Omega(d(X,v),X+v)}{\Omega(b(X+v),n(X+v))} - \frac{\Omega(b(X),X)}{\Omega(b(X),n(X))}. \tag{5.30}$$

Let us assume that X is large enough. This assumption implies that the flow from the cluster boundary to the rest of the graph is altered insignificantly when adding a node to the cluster. Under this condition, the following two approximations hold:

$$\begin{aligned} \Omega(b(X),n(X)) \approx{} & \Omega(b(X+v),n(X+v)), \\ \Omega(b(X),v) - \Omega(d(X,v),X+v) \approx{} & \Omega(b(X),v). \end{aligned} \tag{5.31}$$

Consequently, the following approximation holds:

$$\Delta F(X,v) \approx \frac{\Omega(b(X),v)}{\Omega(b(X+v),n(X+v))}. \tag{5.32}$$

Case 2: $v \in b(X+v)$

This would occur if the node 2 was added to the cluster depicted in Figure 5.1. In this case $b(X+v) = \{v\} \cup b(X) \backslash d(X,v)$. Thus

$$\Delta F(X,v) = \frac{\Omega(b(X),X) + \Omega(b(X),v) - \Omega(d(X,v),X+v) + \Omega(v,X)}{\Omega(b(X+v),n(X+v))} - \frac{\Omega(b(X),X)}{\Omega(b(X),n(X))} \tag{5.33}$$

Using the assumptions stated in Equation (5.31), $\Omega(d(X,v),X)$ can be neglected and $\Omega(b(X),X+v) \approx \Omega(b(X),X)$. Note that

$$v \in n(X) \rightarrow \Omega(v,X) = \Omega(v,b(X)). \tag{5.34}$$

Thus,

$$\Delta F(X,v) \approx \frac{\Omega(v,b(X)) + \Omega(b(X),v)}{\Omega(b(X+v),n(X+v))}. \tag{5.35}$$

For symmetric graphs, $\Omega(A,B) = \Omega(B,A)$. In this case,

$$\Delta F(X,v) \approx 2\frac{\Omega(v,b(X))}{\Omega(b(X+v),n(X+v))}. \tag{5.36}$$

Overall, the approximation of the optimal node is found by maximizing the numerator and minimizing the denominator. For the latter, this is equivalent to minimizing $\Omega(v,V\backslash X)$. The ratio $f(X,v)$ to maximize can thus be approximated by

$$f(X,v) = \begin{cases} \frac{\Omega(b(X),v)}{\Omega(v,V\backslash X)} & \text{if } v \notin b(X+v), \\ \frac{\Omega(b(X),v)+\Omega(v,b(X))}{\Omega(v,V\backslash X)} & \text{else.} \end{cases} \tag{5.37}$$

Note that no differentiation is needed for $f(X,v)$ when the input graph is symmetrical, since the two approximations for $\Delta F(X,v)$ differ only by a constant. Hence,

$$f(X,v) = \frac{\Omega(b(X),v)}{\Omega(v,V\backslash X)} \text{ for symmetrical graphs.} \tag{5.38}$$

Now, BorderFlow can be implemented in a two-step greedy fashion by ordering all nodes $v \in n(X)$ according to $1/f(X,v)$ (to avoid dividing by 0) and choosing the node v that minimizes $1/f(X,v)$. Using this heuristic, BorderFlow is easy to implement and fast to run. The resulting main routine is shown in Algorithm 2.

5.4 Evaluation on Large Scale-Free Graphs

The goal of the evaluation of the heuristic was to determine how well it performs on large, real-world scale-free graphs. For this purpose, we used the Wikipedia[2] Category Graph[3] (WCG) as raw input data and generated three similarity graphs out of it. Subsequently, we used these graphs for clustering.

[2] http://www.wikipedia.org

[3] Version of July 31^{st}, 2007

```
Data: Graph to cluster
Result: Fuzzy clustering
for each v ∈ V do
    X := {v};
    while |n(X)| > 0 do
        C(X) := argmin_{u∈n(X)} 1/f(X, u);
        if (|C(X)| == 1 && F(X ∪ C(X)) ≥ F(X)) then
            X := X ∪ C(X);
        else
            C_f(X) := argmax_{u∈C(X)} Ω(u, n(X));
            if (F(X ∪ C_f(X)) ≥ F(X)) then
                X := X ∪ C_f(X);
            else
                break;
            end
        end
    end
    store X;
end
merge all identical X;
return;
```

Algorithm 2: Current implementation of BorderFlow

5.4.1 Experimental Setup

The WCG is a freely available and large scale-free graph (Zesch and Gurevych, 2007) containing 244,545 categories (i.e., nodes). In this series of experiments, we aimed at discovering similar categories. Wikipedia categories are interrelated by the *sub-category* relation, which is equivalent to the specialization, i.e., the *is-a* relation. As categories can be used to tag articles, we defined a further relation called *shared-article*, which holds for two categories when they have been used to tag the same article. We also considered the inverse relation to *sub-category*, i.e., *parent-of*, for the purpose of our evaluation. We used the Jaccard metric (Tan et al., 2005) to measure the similarity $\sigma_r(X, X')$ of the categories X and X' according to each of the relations previously defined:

$$\sigma_r(X, X') = \frac{2|R(X, r) \cap R(X', r)|}{|R(X, r) \cup R(X', r)|} \tag{5.39}$$

with

$$R(X, r) = \{y : r(x, y)\}. \tag{5.40}$$

The result of each similarity computation was a weighted category similarity graph G_r=(V, E, σ_r). The average connectivity was approximately 295 for *parent-of*, 8 for *sub-category* and 60 for *shared-article*. We measured the quality of the clustering by applying the following variation of the silhouette index $\sigma(X)$ (Rousseeuw, 1987) to each cluster X:

$$\sigma(X) = \frac{1}{|X|} \sum_{v \in X} \frac{a(v, X) - b(v, V \backslash X)}{max\{a(v, X), b(v, V \backslash X)\}}, \tag{5.41}$$

where

$$a(v, X) = \frac{\sum_{v' \in n(v) \cap X} \omega(v, v')}{|n(v) \cap X|} \tag{5.42}$$

and

$$b(v, V \backslash X) = \max_{v' \in V \backslash X} \omega(v, v'). \tag{5.43}$$

A value of $\sigma(X)$ around 1 hints toward a good clustering, whilst a value of -1 hints toward an unsuitable clustering.

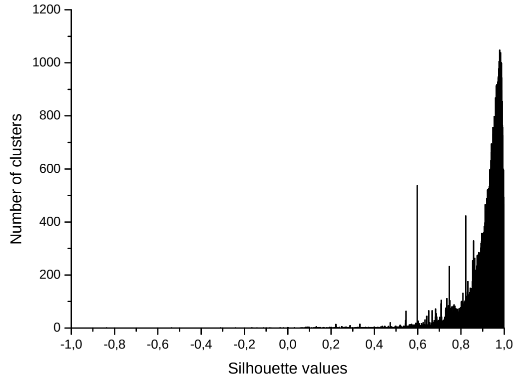
5.4.2 Results and Discussion

Some topological characteristics of the graphs we used for this clustering experiment are shown in Table 5.1. A high percentage of the categories did not have any descendant. Therefore, clustering over *sub-category* covered solely 31.63% of the categories in the WCG. The other two relations covered approximately the same percentage of categories (82.21% for *shared-article* and 82.07% for *parent-of*).

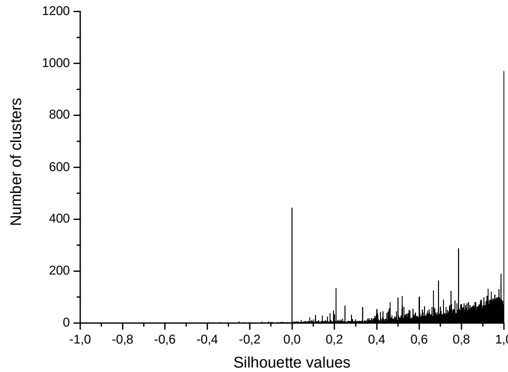
Relation	Categories	Clusters	Avg. N/C	Avg. C/N	% categories	$\mu \pm \sigma$
shared-article	201,049	93,331	3.59	7.74	82.21	0.92 ± 0.09
son-of	77,292	28,586	2.29	6.20	31.61	0.20 ± 0.19
parent-of	200,688	90,418	8.63	19.15	82.07	0.74 ± 0.24

Table 5.1: Results of the clustering obtained on the WCG using BorderFlow. *Avg. N/C stands for the average number of nodes per cluster. Avg. C/N stands for the average number of clusters per nodes. μ is the average silhouette value of the clusters computed using each of the relations. σ is the standard deviation of the same silhouette value.*

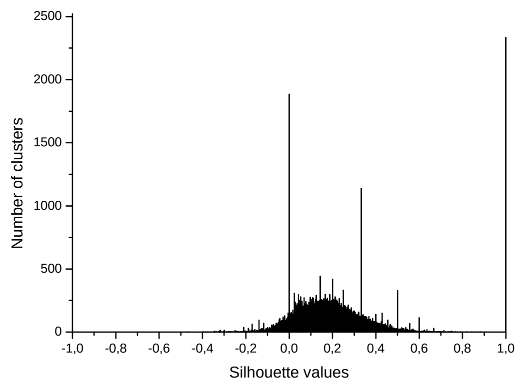
Figure 5.8 displays the results we obtained by using the three relations considered. The best clustering was achieved when using the *shared-article* relation (see



(a) Using shared-article



(b) Using son-of

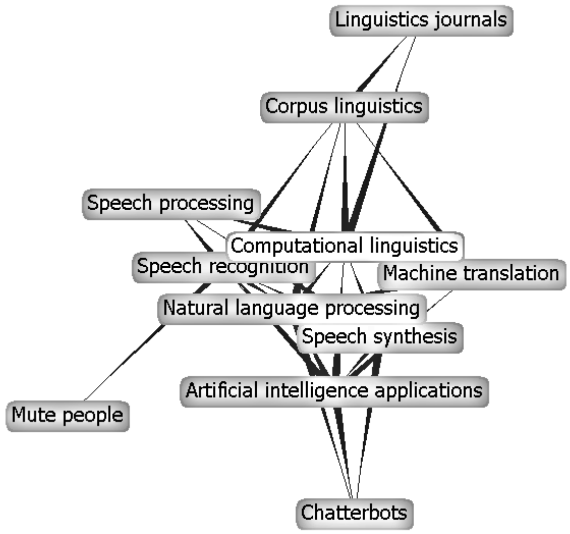


(c) Using parent-of

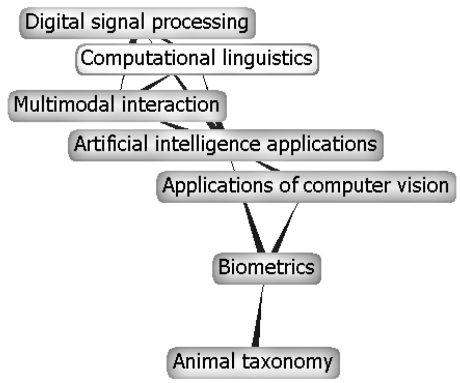
Figure 5.8: Distribution of silhouette values. *Clustering using shared-article leads to the best results.*

Figure 5.8(a)): we obtained the highest mean silhouette (0.92) with the smallest standard deviation (0.09). An analysis of the silhouette values resulting from using the *sub-category* relation revealed that the mean of the silhouette lied around 0.74, yet with a standard deviation of 0.24 (see Figure 5.8(b)). Clustering using *parent-of* yielded the worst results, with a mean at 0.20. The standard deviation was approximately 0.19.

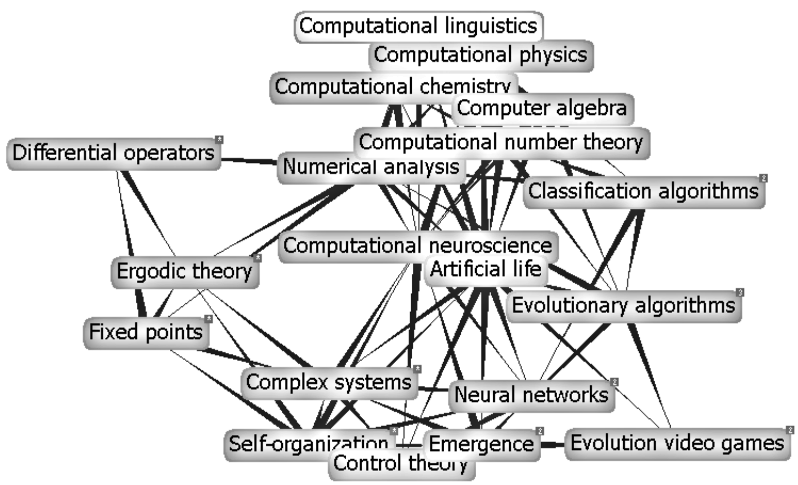
The distribution of silhouette values we obtained when clustering the similarity graph based on the *parent-of* relation were caused by the high connectivity of this graph. Its connectivity resulted into large clusters, leading to a higher flow to the outside of each cluster and thus to small silhouette values. Clustering by using *sub-category* yielded better silhouette results because the connectivity of the graph generated using this relation was reduced. Yet, the reduced connectivity also led to the smallest average cluster size. The results resulting from clustering by *shared-*



(a) Using shared-article



(b) Using sub-category



(c) Using parent-of

Figure 5.9: Examples of clusters containing “Computational Linguistics”

article can be linked to the so-called intelligence of crowds. The entries we utilized to generate the similarity data were collected from manually corrected Wikipedia pages, leading to a more reliable data set. Table 5.1 shows the mean and deviation values of the clusterings we obtained by using the different relations.

The results achieved on the Wikipedia graphs show that BorderFlow revealed that our heuristic can be used to efficiently cluster large graphs. Interestingly, our evaluation also showed that BorderFlow can disambiguate the meanings of polysemantic categories (see Figure 5.9 for an example). This particular property will be investigated in depth in future work.

5.5 Experiments and Results

In this section, we present experiments carried out by using BorderFlow for the task of concept extraction. We carried out two types of evaluation, namely a quantitative and a qualitative evaluation. In the quantitative evaluation, we compared the clustering achieved by BorderFlow with that achieved by kNN (Tan et al., 2005) on word similarity graphs . In the qualitative evaluation, we compared the content of the clusters with the MESH taxonomy.

5.5.1 Experimental Setup

Most techniques for semantic clustering have been optimized for high-level features such as verb-subject relations (see, e.g., (Pantel and Lin, 2002) and (Khan and Luo, 2002)). Yet, computing such features requires knowledge about the grammar of the language processed. In our experiments, we used purely statistical and thus language-independent features for semantic clustering. Instead of high-level features, we used features based on second-order co-occurrences (Heyer et al., 2001; Biemann et al., 2004). The idea behind second-order co-occurrences is that similar terms tend to have similar first-order co-occurrences. Hence, the set of most significant first-order co-occurrences of a term can be used to measure its similarity with other terms. The example shown in Figure 5.10 illustrates the idea. The term *leukocyte* is similar to *neutrophil* and they share a subset of their most significant first-order co-occurrences.

The most significant co-occurrences of the terms included in the lexicon were extracted from the results presented in Section 4.5. In a first step, we extracted function words by retrieving the f terms with the lowest information content according to Shannon's law (Shannon, 1948). Function words were not considered as being significant co-occurrences. Then, the s best scoring co-occurrences of each term that were not function words were extracted and stored as binary feature vectors. The similarity of the feature vectors v_1 and v_2 of two terms t_1 and t_2 was then computed using the cosine metric:

$$cos(v_1, v_2) = \frac{v_1.v_2}{||v_1||.||v_2||}. \tag{5.44}$$

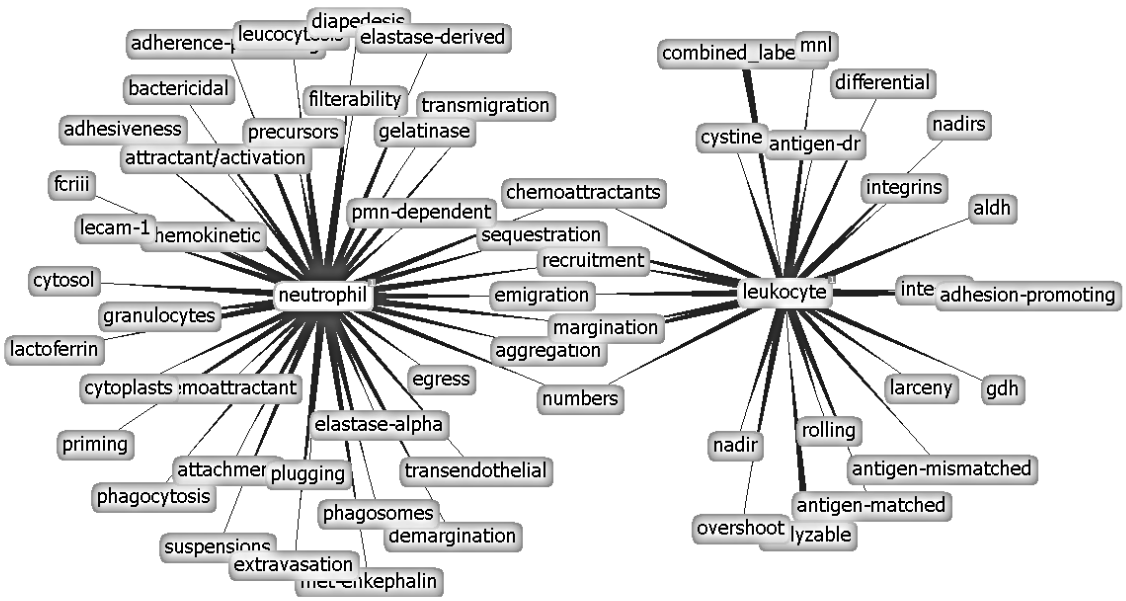


Figure 5.10: Excerpt of the most significant co-occurrences of leukocyte and neutrophil. *The two terms share a subset of their most significant first co-occurrences including chemoattractants, sequestration and aggregation.*

The resulting similarity graph was finally clustered using BorderFlow. Figure 5.11 shows an excerpt of the similarity generated out of the TREC data.

We carried out our experiments on the TREC and the BMC corpus. On the TREC corpus, similarity values below 0.01 were not considered. On the BMC corpus, we used a threshold of 0.05 because it was more noisy. Only words with a frequency above 25 were considered for clustering. We did not use potentially polysemic terms (i.e., hubs) as seeds. Thus, we used only terms that had a connectivity less or equal to the average connectivity for clustering. Note that polysemes not being used as seeds does not imply that polysemes were excluded from the clustering.

5.5.2 Quantitative Evaluation

The goal of the quantitative evaluation was to determine the accuracy of the clustering achieved by BorderFlow. We compared the average silhouettes of the clusters computed by BorderFlow with those computed by kNN on the same graphs. To ensure that all clusters had the same maximal size k, we use the following greedy approach for each seed: first, we initiated the cluster X with the seed. Then, we sorted all $v \in n(X)$ according to their flow to the inside of the cluster $\Omega(v, X)$ in

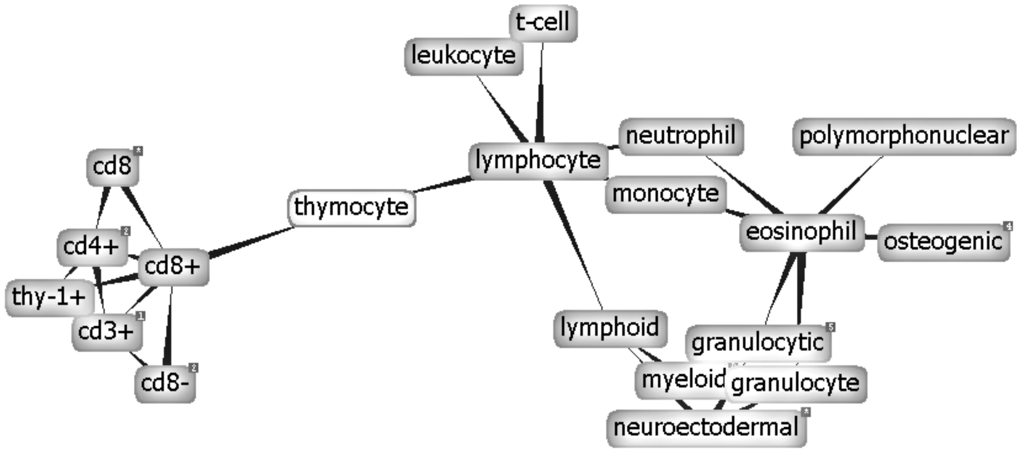


Figure 5.11: Excerpt of the similarity graph computed using the TREC data with $f = 100$ and $s = 400$.

the descending order. Thereafter, we sequentially added all v until the size of the cluster reached k. If it did not reached k after adding all neighbors, the procedure was iterated with $X = X \cup n(X)$ until the size k was reached or no more neighbors were found.

One of the drawbacks of kNN lies in the need for specifying the right value for k. In our experiments, we used the average size of the clusters computed using BorderFlow as value for k. This value was 7 when clustering the TREC data. On the BMC corpus, the experiments with $f = 100$ led to $k = 7$, whilst the experiments with $f = 250$ led to $k = 9$. We used exactly the same set of seeds for both algorithms.

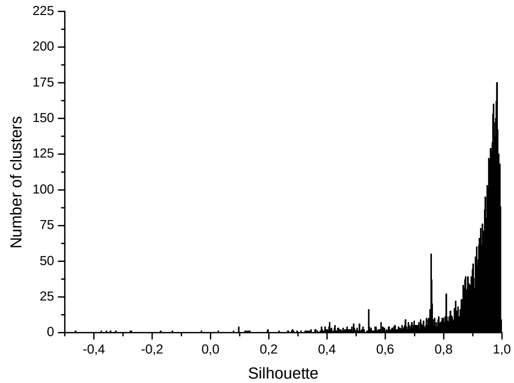
We measured the accuracy of the clustering in two ways. First, we used the average silhouette value of the clusters. Second, we computed the number of erroneous clusters, i.e., the number of clusters with negative silhouette values. The results of the evaluation are shown in Table 5.2. On both data sets, BorderFlow significantly outperformed kNN in all settings.

On the TREC corpus, both algorithms generated clusters with high silhouette values. BorderFlow outperformed kNN by 0.23 in the best case ($f = 100$, $s = 100$). The greatest difference between the standard deviations, 0.11, was observed when $f = 100$ and $s = 200$. In average, BorderFlow outperformed kNN by 0.17 with respect to the mean silhouette value and by 0.08 with respect to the standard deviation. In the worst case, kNN generated 73 erroneous clusters, while BorderFlow generated 10. The distribution of the silhouette values across the clusters on the TREC corpus for all six combinations of f and s are shown in Figure 5.12 for BorderFlow and Figure 5.13 for kNN.

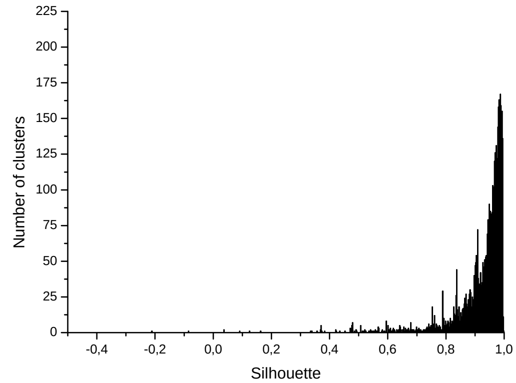
		$\mu \pm \sigma$				Erroneous clusters			
f	s	TREC		BMC		TREC		BMC	
		kNN	BF	kNN	BF	kNN	BF	kNN	BF
100	100	0.68±0.22	**0.91**±0.13	0.37±0.28	**0.83**±0.13	73	**10**	214	**1**
100	200	0.69±0.22	**0.91**±0.11	0.38±0.27	**0.82**±0.12	68	**1**	184	**1**
100	400	0.70±0.20	**0.92**±0.11	0.41±0.26	**0.83**±0.12	49	**1**	142	**1**
250	100	0.81±0.17	**0.93**±0.09	0.23±0.31	**0.80**±0.14	10	**2**	553	**0**
250	200	0.84±0.13	**0.94**±0.08	0.23±0.31	**0.80**±0.14	5	**2**	575	**0**
250	400	0.84±0.12	**0.94**±0.08	0.24±0.32	**0.80**±0.14	2	**1**	583	**0**

Table 5.2: Comparison of the distribution of the silhouette index over clusters extracted from the TREC and BMC corpora. *f is the threshold for function words, s the number of co-occurrences considered during the extraction of the feature vectors, μ the mean of silhouette values over the clusters and σ the standard deviation of the distribution of silhouette values. Erroneous clusters are cluster with negative silhouette silhouettes. Bold fonts mark the best results in each experimental setting.*

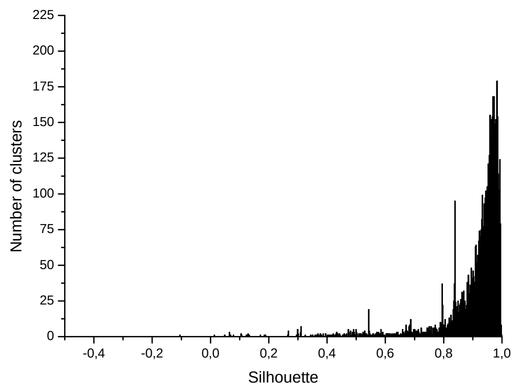
The superiority of BorderFlow over kNN was better demonstrated on the noisy BMC corpus. Both algorithms generate a clustering with lower silhouette values than on TREC. In the best case, BorderFlow outperformed kNN by 0.57 with respect to the mean silhouette value ($f = 250$, $s = 200$ and $s = 400$). The greatest difference between the standard deviations, 0.18, was observed when $f = 250$ and $s = 400$. In average, BorderFlow outperformed kNN by 0.5 with respect to the mean silhouette value and by 0.16 with respect to the standard deviation. Whilst BorderFlow was able to compute a correct clustering of the data set, generating maximally 1 erroneous cluster, using kNN led to large sets of up to 583 erroneous clusters ($f = 100$, $s = 400$). Figures 5.14 and 5.15 show the distribution of the silhouette values across the clusters on the BMC corpus for all six combinations of f and s.



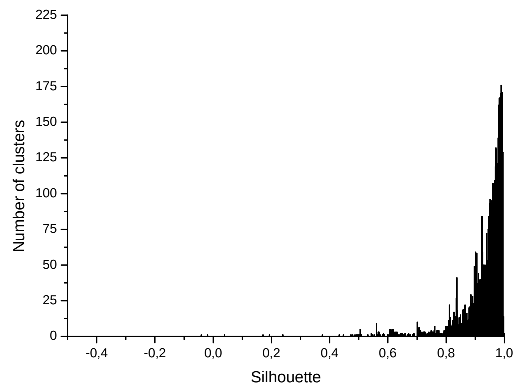
(a) f=100, s=100



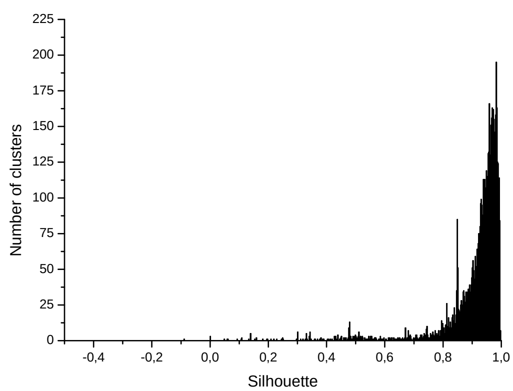
(b) f=250, s=100



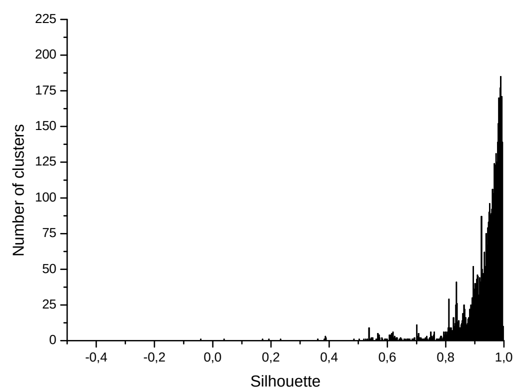
(c) f=100, s=200



(d) f=250, s=200

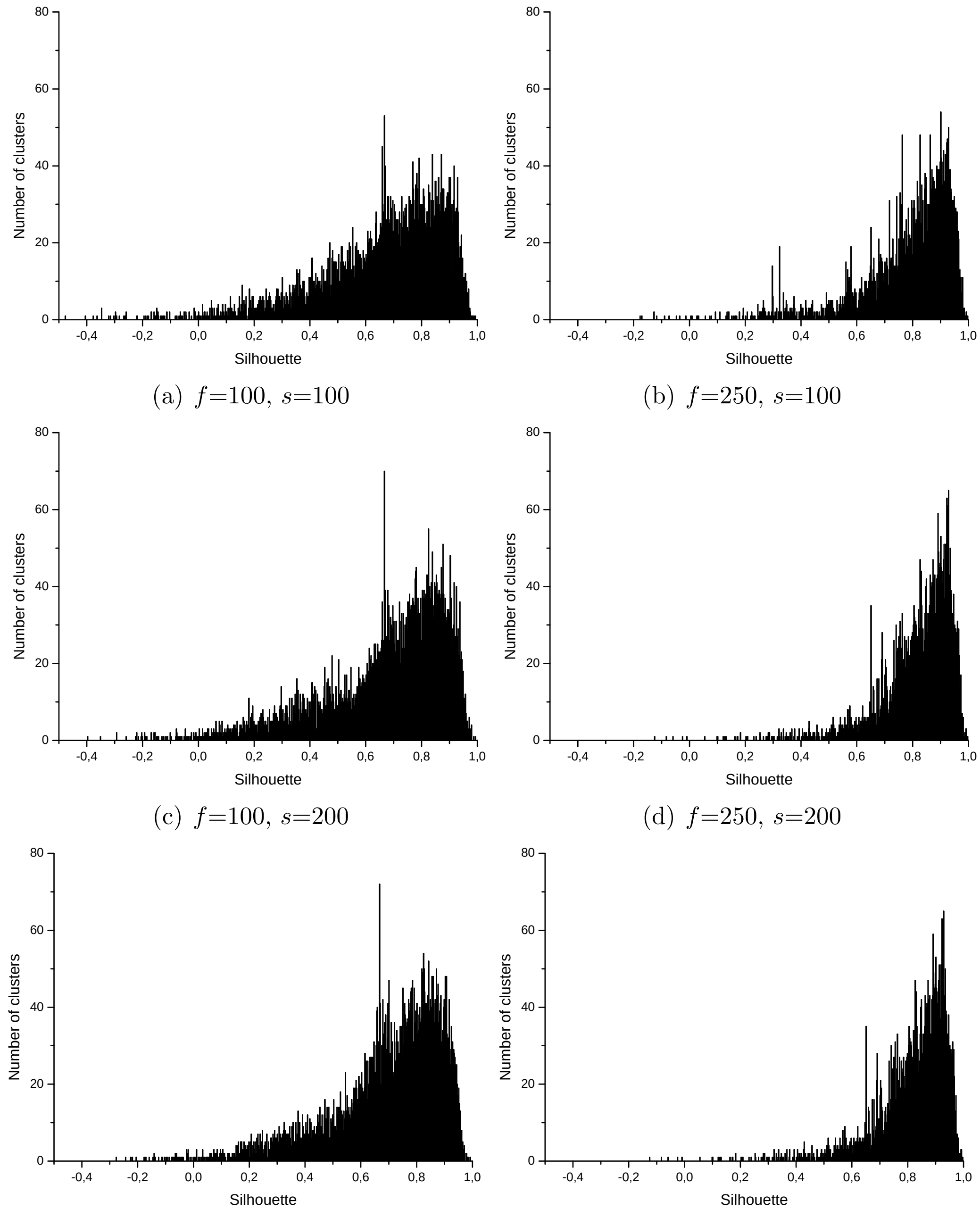


(e) f=100, s=400



(f) f=250, s=400

Figure 5.12: Distribution of the average silhouette values obtained by using BorderFlow on the TREC data set. *f is the threshold for function words. s is the number of co-occurrences considered during the extraction of the feature vector.*



(a) f=100, s=100

(b) f=250, s=100

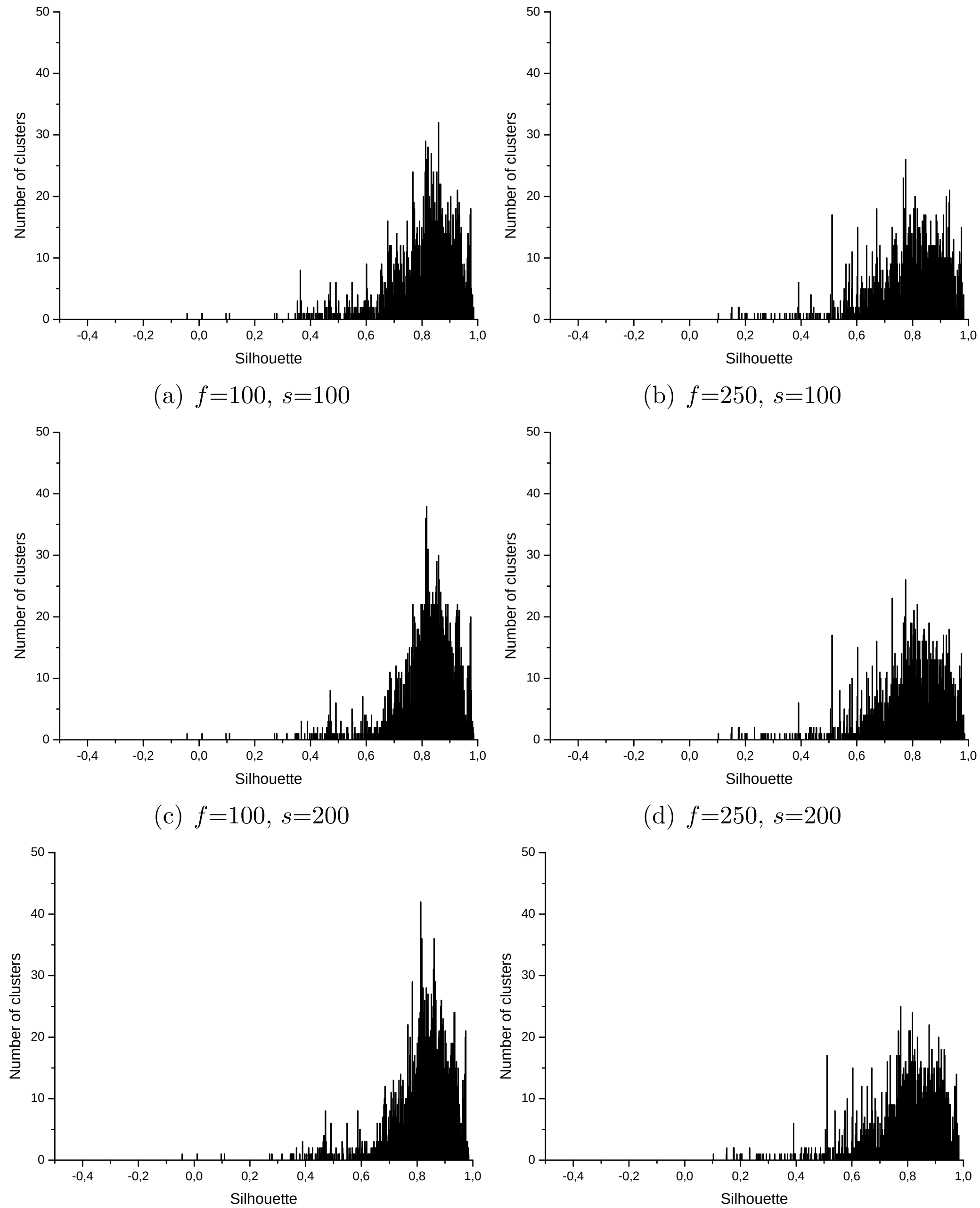
(c) f=100, s=200

(d) f=250, s=200

(e) f=100, s=400

(f) f=250, s=400

Figure 5.13: Distribution of the average silhouette values obtained by using kNN on the TREC data set. *s is the number of co-occurrences considered during the extraction of the feature vector.*



(a) f=100, s=100

(b) f=250, s=100

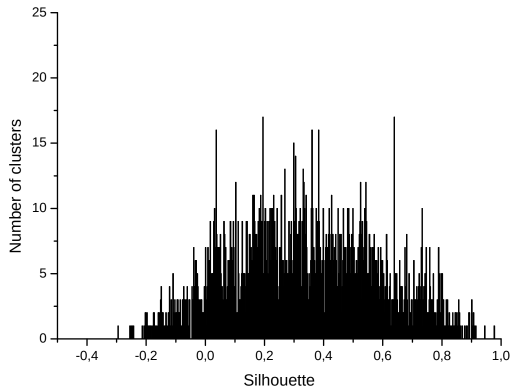
(c) f=100, s=200

(d) f=250, s=200

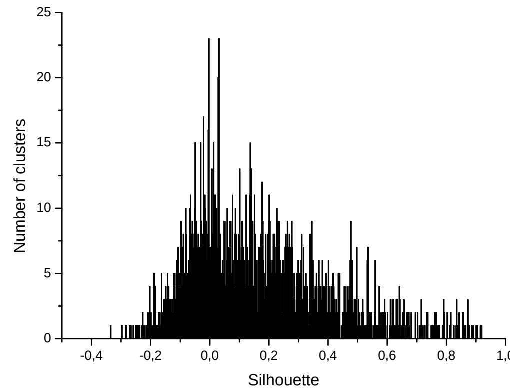
(e) f=100, s=400

(f) f=250, s=400

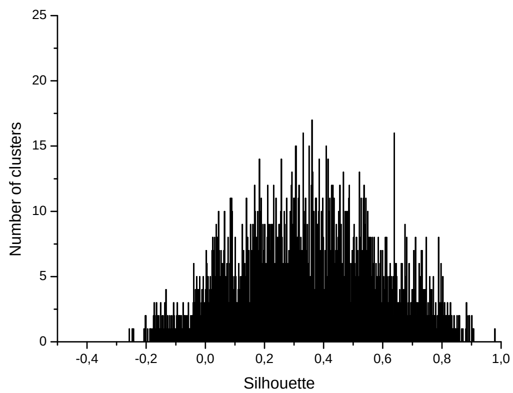
Figure 5.14: Distribution of the average silhouette values obtained by using BorderFlow on the BMC data set. *s is the number of co-occurrences considered during the extraction of the feature vector.*



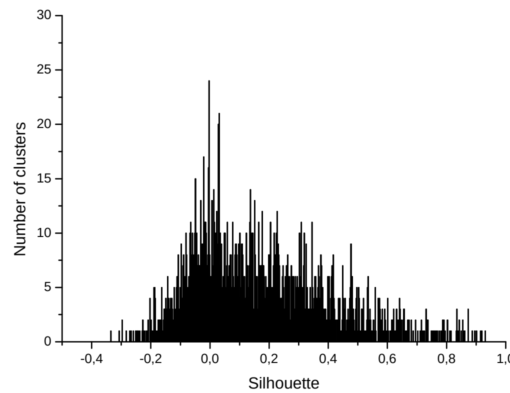
(a) f=100, s=100



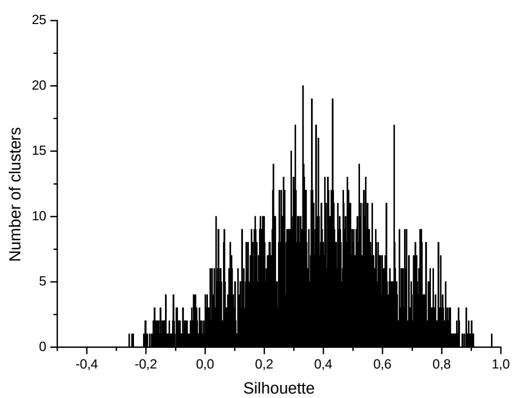
(b) f=250, s=100



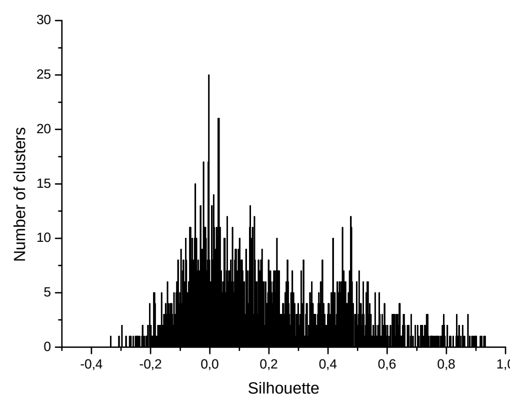
(c) f=100, s=200



(d) f=250, s=200



(e) f=100, s=400



(f) f=250, s=400

Figure 5.15: Distribution of the average silhouette values obtained by using kNN on the BMC data set. *s is the number of co-occurrences considered during the extraction of the feature vector.*

5.5.3 Qualitative Evaluation

The goal of the qualitative evaluation was to determine the quality of the content of our clusters. We focused on elucidating whether the elements of the clusters were labels of semantically related categories. To achieve this goal, we compared the content of the clusters computed by BorderFlow with the MESH taxonomy (Ananiadou and Mcnaught, 2005). It possesses manually designed levels of granularity as displayed in Figure 5.16. Therefore, it allows to evaluate cluster purity at different levels. We did not evaluate our results against SNOMED-CT because it does not allow the evaluation of cluster purity in the same way due to its ontological structure (Ananiadou and Mcnaught, 2005). Furthermore, we did not use UMLS because it presents cycles in its taxonomical structure (Mougin and Bodenreider, 2005).

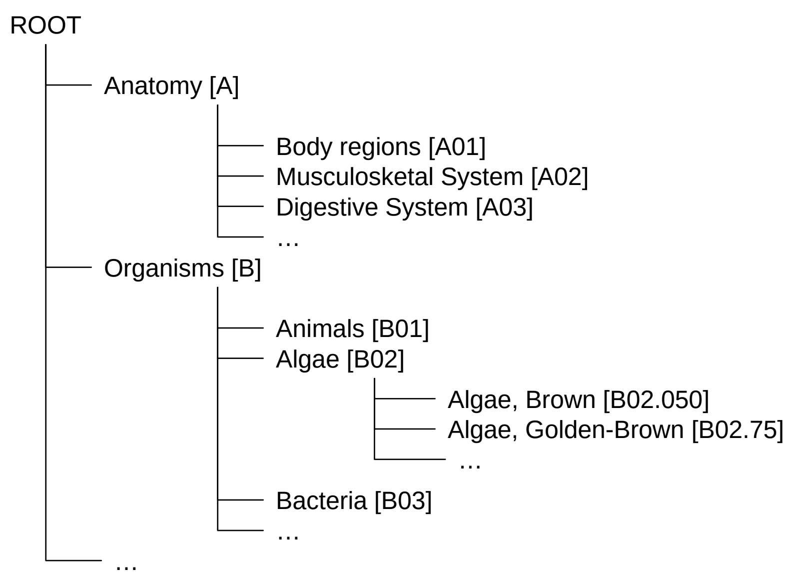


Figure 5.16: Excerpt of the MESH taxonomy.

We evaluated the purity of our clusters by measuring the following value:

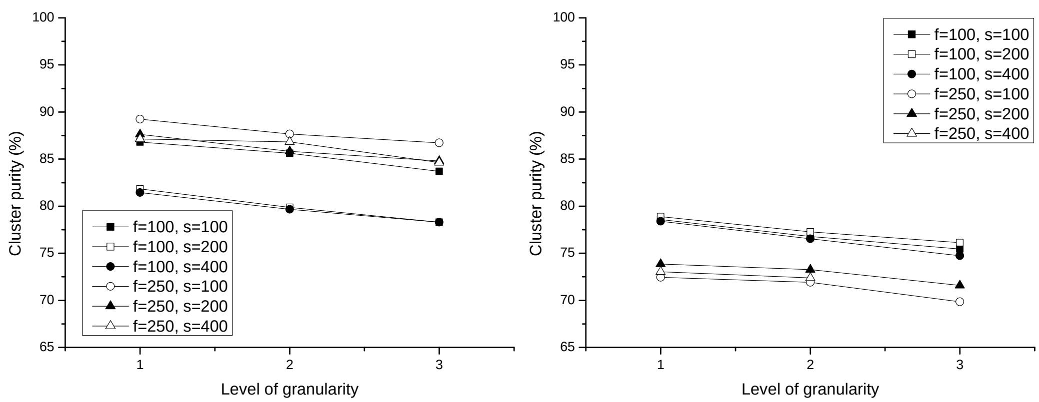
$$\varphi(X) = \max_{C} \left(\frac{|X \cap M|}{|X \cap C^{*}|} \right), \tag{5.45}$$

where X is a cluster computed by BorderFlow, M is the set of all mesh category labels, C is a MESH category and C^{*} is the set of labels of C and all its subcategories. For our evaluation, we considered only clusters that contained at least one term that could be found in MESH.

The results of the qualitative evaluation are shown in Table 5.3 and in Figure 5.17. The best cluster purity, 89.23%, was obtained when clustering the vocabulary extracted from the TREC data with $f = 250$ and $s = 100$. In average, we obtained a lower cluster purity when clustering the BMC data. The best cluster purity using BMC was 78.88% ($f = 100$, $s = 200$). On both data sets, the difference in cluster quality at the different levels was low, showing that BorderFlow was able to detect fine-grained cluster with respect to the MESH taxonomy. Example of clusters computed with $f = 250$ and $s = 400$ using the TREC corpus are shown in Table 5.4.

Level	f=100 s=100	f=100 s=200	f=100 s=400	f=250 s=100	f=250 s=200	f=250 s=400
1	86.81	81.84	81.45	89.23	87.62	87.13
2	85.61	79.88	79.66	87.67	85.82	86.83
3	83.70	78.55	78.29	86.72	84.81	84.63
1	78.58	78.88	78.40	72.44	73.85	73.03
2	76.79	77.28	76.54	71.91	73.27	72.39
3	75.46	76.13	74.74	69.84	71.58	70.41

Table 5.3: Cluster purity obtained using BorderFlow on TREC and BMC data. *The upper section of the table displays the results obtained using the TREC corpus. The lower section of the table displays the same results on the BMC corpus. All results are in %.*



(a) Clustering using TREC data

(b) Clustering using BMC data

Figure 5.17: Cluster purity obtained using BorderFlow on TREC and BMC data.

Cluster members	Seeds	Hypernym
b_fragilis, c_albicans, candida_albicans, l_pneumophila	c_albicans	Etiologic agents
acyclovir, methotrexate_mtx, mtx, methotrexate	methotrexate	Drugs for curing endo-bronchial papillomatosis
embryo, embryos, mouse_embryos, oocytes	embryo, embryos, mouse_embryos, oocytes	Egg cells
leukocytes, macrophages, neutrophils, platelets, pmns	platelets	Blood cells
intramuscular_injections, intravenous_infusions, intravenous_injections, *developmental_stages*, bolus_doses	intravenous_injections	General anesthesia
agonist, agonists, receptor_agonist, receptor_agonists, receptor_antagonists, receptor_blockade, receptor_gene, receptor_number, receptors	receptor_number	Receptors
albuterol, carbamazepine, deferoxamine, diuretic, diuretics, fenoldopam, hmg, inh, mpa, nedocromil_sodium, osa, phenytoin, pht	albuterol	Drugs
antisocial, depressive, paranoid, personality_disorder	paranoid	Personality disorders
atropine, atropine_sulfate, cocaine, *epinephrine*, morphine, *nitroglycerin*, scopolamine, verapamil	atropine_sulfate	Alkaloids
flap, flaps, free_flap, muscle_flap, musculocutaneous_flap	flap, free_flap	Flaps
complete_absence, complete_resolution, dose-dependent_inhibition, dramatic_improvement, little_change, marked_elevation, marked_improvement, marrow_involvement, symptomatic_improvement, wide_variations	marrow_involvement	Diagnostic findings
leukocyte, monocyte, neutrophil, polymorphonuclear_leukocyte	polymorphonuclear_leukocyte	White blood cells

Table 5.4: Examples of clusters extracted from the TREC corpus. *The relation between the elements of the clusters is displayed in the rightmost column. Cluster members in italics are erroneous.*

5.5.4 Discussion

Overall, we obtained a better clustering on the TREC data than on the BMC data. From a quantitative point of view, the average silhouette values μ on TREC were higher with lower standard deviations σ. The difference in silhouette can be conceivably explained by the higher amount of noise contained in the BMC corpus. On the TREC corpus, a higher size of the feature vectors led to a higher value μ of the average silhouette of the clusters. The same relation could be observed between the number f of function words omitted and the value of μ. The standard deviation σ was inversely proportional to the size of the feature vectors and the number of function words. The number of erroneous clusters (i.e., clusters with average silhouette value less than 0) was inversely proportional to the size of the feature vectors. This can be explained by the higher amount of information available, which led to a better approximation of the semantic similarity of the terms and, thus, to less clustering mistakes. In the worst case (f=100, s=100), 99.85% of the clusters had positive silhouettes.

From a qualitative point of view, BorderFlow computed clusters with a high purity based on low-level features extracted on a terminology extracted using low-bias techniques. As expected, the average cluster purity was higher for clusters computed using the TREC data set. The results of the qualitative evaluation support the basic assumption underlyging this work, i.e., that it is indeed possible to extract high-quality background knowledge from text automatically given a sufficient amount of input data and suitable algorithms for analyzing and clustering this data.

BorderFlow can be extended in several ways. First, a stronger definition of concept could be adopted by demanding that the border flow of each cluster to its inside should be higher than that to the outside, i.e.,

$$\Omega(b(X), X) > \Omega(b(X), n(X)) \rightarrow F(X) > 1. \tag{5.46}$$

Yet, this stronger definition might be too restrictive for certain graphs, especially graphs of high density. Furthermore, BorderFlow can be extended to hierarchical clustering. The resulting clusters can be namely seen as nodes of a higher-level weighted graph $\Gamma = (\Psi, \Sigma, \chi)$ with Ψ being the set of all generated clusters, Σ being the set of edges between clusters and χ being the flow between clusters. Given two clusters X and X', the weight of the edge XX' would then be

$$\chi(X, X') = \begin{cases} 0 & \text{if } \Omega(X, V) = 0; \\ \frac{\Omega(X, X')}{\Omega(X, V)} & \text{else.} \end{cases} \tag{5.47}$$

Using this simple equation, a hierarchical, bottom-up and fuzzy clustering of the graph G can be generated in a bootstrapping fashion. Finally, BorderFlow can be extended to produce a crisp clustering of the graph, as demanded by certain domains of application. A hardening of BorderFlow's clustering can be carried out by assigning each node u to the cluster X which maximizes its membership $\mu(u, X)$:

$$\mu(u, X) = \frac{\Omega(u, X)}{\Omega(u, V)}. \tag{5.48}$$

The sum of the memberships of a node u over all clusters can be higher than one, when some of these clusters overlap. Yet, the membership $\mu(u, X)$ to a given cluster X is bounded between 0 and 1.

Chapter 6

Conclusion and Future Work

The aim of this thesis was to present and evaluate an approach to the low-bias extraction of domain-specific concepts from unrestricted text. We have focused on extracting concepts of high purity. Therefore, we have been mainly interested in approaches with a high precision. Overall, we have shown that low-bias approaches can be used to extract high-quality concepts out of text. In our experiments, our approach reached an average cluster purity close to 90%. To obtain these results, we subdivided our work into three main sections: discovery of domain-specific multi-word units (MWUs), extraction of domain-specific lexica and extraction of concepts.

6.1 Extraction of Multi-Word Units

The first section of our work presents a novel measure for the extraction of domain-specific MWUs called Smoothed Relative Expectation metric (SRE). The measure was applied to two data sets of different size, granularity and cleanness. We compared our results with those obtained by six other common metrics against three different gold standards. Subsequently, we compared SRE with other multi-contextual metrics. In both experimental settings, SRE significantly outperformed all other metrics in both precision and recall. Consequently, the soundness of our assumptions on domain-specific MWUs was proven. We have also demonstrated that the inclusion of a model for specificity can significantly improve the low-bias detection of domain-specific MWUs.

An aim of future research will be to elaborate on the specificity idea that underlies SRE. We will implement and compare other possible models. Moreover, we will develop a technique for finding the best cut-off for the subsequent terminology extraction automatically. A careful study of the topology generated by the scores of single words and multi-words and of the correlation between the gradients and the domain specificity, as proposed by Ferreira da Silva and Pereira Lopes (1999), might produce valuable results. The criterion of non-substitutability could play a greater role in an extended version of SRE. By using the results of the concept extraction technique presented in this work, it should be possible to improve the measurement of the similarity of patterns and thus the total scoring function. Further extensions of SRE will include the usage of a higher complexity of the expectancy E_n and the analysis of non-connected collocations (Dias, 2002). Future analyses will compare the performance of the current implementation with that of implementations based on other data structures such as suffix arrays (Morrison, 1968; Sedgewick, 1988).

6.2 Extraction of Domain-Specific Lexica

Chapter 4 is concerned with the extraction of domain-specific lexica from the results of SRE. To achieve this goal, we proposed a graph-based algorithm called SIGNUM. This algorithm uses the spreading activation principle to compute a binary clustering of word graphs. We presented a basic version of SIGNUM for simple graphs and an extended version of the same algorithm for hypergraphs. In a final step, the results of SIGNUM were compared with those of SRE. Our evaluations support that SIGNUM can significantly boost the precision of SRE. Our results also show that SIGNUM does not significantly alter the recall of SRE on large graphs extracted from large corpora.

Extensions of SIGNUM could be used in many other research areas. In future work, we will evaluate the performance of SIGNUM and its extensions on more complex graph categories such as hypergraphs and multigraphs. Additionally, SIGNUM will be applied to several other NLP tasks, including lexicon expansion and ontology population. Our algorithm could be used for classification tasks, provided that it is supplied with training data in the form of initial graph configurations. A combination of SIGNUM and CLIque-based Clustering (Ngonga Ngomo, 2006) could be utilized for general-purpose clustering on arbitrary graphs.

6.3 Concept Extraction

The last section of this work focuses on the extraction of concepts. We presented a novel graph clustering algorithm for arbitrary graphs called BorderFlow. The idea behind BorderFlow is to regard graphs as flow systems and to cluster them by maximizing the ratio between the inner and the outer flow of each cluster. Our algorithm was evaluated in three settings with different goals. First, in evaluating BorderFlow on synthetic graphs with known best partition, we have proven that the algorithm achieves the desirable clustering. Second, in evaluating it on three similarity graphs extracted from the Wikipedia Category Graph, we have shown that our algorithm can cluster large scale-free graphs. The final evaluation of our algorithm was carried out on word similarity graphs computed by using second-order co-occurrences. In the quantitative section of the final evaluation, we compared the results obtained by using BorderFlow with those achieved by kNN. BorderFlow significantly outperformed kNN with respect to the silhouette index in all settings. We also carried out a qualitative evaluation of BorderFlow by comparing the purity of the clusters it computed with MESH. In our experiments, BorderFlow extracted clusters of high average purity. The overall conclusion of this work is consequently that we can extract high-quality concepts from text without having a-priori knowledge on language or domain. Our experiments have shown that our technique can be used both on clean and noisy data sets.

In the future, we will integrate BorderFlow in NLP applications and recommender tools. BorderFlow is suitable for clustering large graphs that display a high degree of symmetry because of the algorithm's fast convergence and local search approach. Therefore, BorderFlow can be integrated in NLP tools such as relation extraction tools for bio-medicine, whose functionality is based on large interaction graphs (Qian et al., 2001). BorderFlow can also be integrated in a large range of applications that demand clusters to be computed at runtime. These applications include tag, keyword and document recommenders based on similarity graphs (Basile et al., 2007). Other possible domains of application include data classification, information retrieval based on browsing and ontology population.

6.4 Future work

The global aim of our future work will be to integrate our results in the back-end and front-end of domain-specific information systems. The direct continuation of the work presented herein will lie in the areas of knowledge discovery and semantic tools (see Figure 6.1).

6.4.1 Knowledge Discovery

The logical step following the extraction of concepts is the extraction of relations between these concepts. To the best of our knowledge, methods for the low-bias extraction of ontological relations have not yet been proposed. This lack is certainly due to the domain-specificity of such relations, which demands the development of knowledge-rich extraction techniques. Our research in this area will be concerned with the extraction of domain-specific relations based on a combination of low-bias

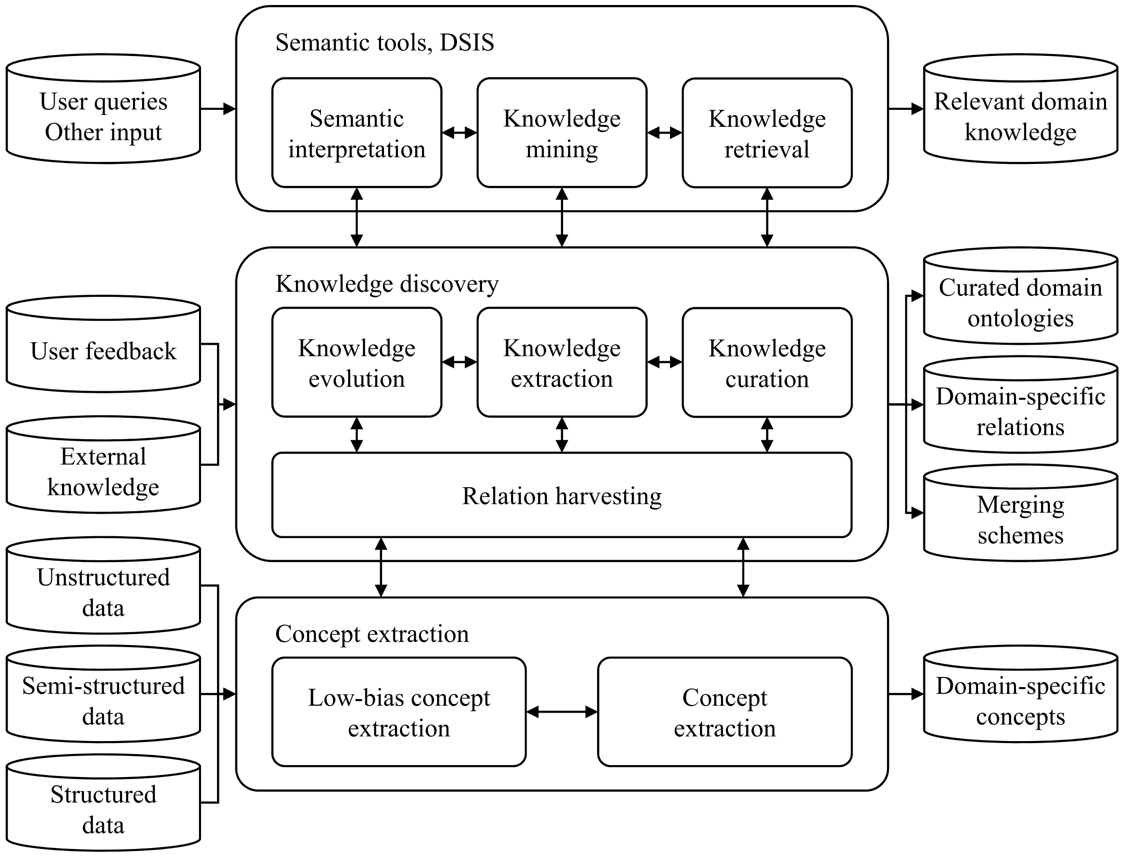


Figure 6.1: Future Work

techniques for concept extraction, statistical testing for significance and clustering. Hoehndorf et al. (2008) presented first results in this area.

A drawback of low-bias concept extraction is the non-formal representation of concepts. Formal approaches to the description of language allow such a representation (Barwise and Perry, 1983). Therefore, our future research will also aim at the integration of low-bias concept extraction and relation harvesting techniques in a formal framework. Subsequently, we will develop techniques that combine user-generated feedback and reasoners to curate, populate and evolve formal ontologies. The incorporation of our approach to concept extraction and of relation harvesting techniques into a formal framework promises to lower the bias of domain-specific ontology extraction significantly. In addition, the integral formalization of domain-specific knowledge should allow the creation of merging schemes for domain-specific knowledge bases. Merged knowledge bases would be particularly useful in domains where knowledge bases represent solely facets of the same formalized domain-specific knowledge (e.g., bio-medicine). The results of approaches to knowledge discovery build the basis upon which domain-specific semantic tools operate.

6.4.2 Semantic Tools

Semantic tools provide dedicated functionality to manipulate and utilize formalized knowledge. They encompass the functionality required to implement methods for the ontological interpretation of input data, knowledge mining and knowledge retrieval.

The ontological interpretation of input data requires the existence of accurate domain ontologies. Based on this formal knowledge, input data such as text and data mining results can be integrated in existing instance knowledge. The generation and the analysis of an ontological interpretation are not straight forward, as it is necessary to deal both with inconsistent and incomplete knowledge. Classical logics will prove to be insufficient for this task. Therefore, we will use a combination of non-monotonic logics and non-classical inferences such as abduction and induction. First considerations in this area were discussed in (Hoehndorf et al., 2008).

Knowledge mining tools use ontological background knowledge and mechanisms of ontological interpretation on large data sets to filter, extract and aggregate a formal representation of the relevant domain-specific knowledge contained in these data sets. Knowledge mining tools go beyond current approaches to data mining. They cannot only recognize patterns contained in large data sets but are also able to derive possible explanations for such patterns. To implement this functionality, we will also use a combination of non-monotonic logics and non-classical reasoning.

Knowledge retrieval is the ontological counterpart to information and data retrieval. One of the main drawbacks of current approaches to information retrieval is the information overflow with which users are confronted. Knowledge retrieval tools promise to find the relevant knowledge out of large data sets and to present it in a structured form, so as to provide the user with exactly the information he needs. This functionality is currently being developed for domains where the amount of knowledge available grows rapidly (e.g., bio-medicine, physics and chemistry) and where domain-specific ontologies are available. Overall, our future research will aim at creating user-friendly semantic applications based on knowledge discovery from heterogeneous sources.

Appendix A

Example from the OHSU-TREC-9 corpus

The test sub-corpus of OHSU-TREC-9 consists of three files, of which the corpus file `ohsumed.88-91` was selected for extracting our test data. The file itself consists of a concatenation of entries, of which each contains the following tags:

- .I: ID
- .U: ID
- .S: Subject
- .M: MeSH terms
- .P: Type of publication
- .W:
- .A: Author

An exemplary entry of the OHSU-TREC-9 corpus is displays below:

.I 54711
.U
88000001
.S
Alcohol Alcohol 8801; 22(2):103-12
.M

Acetaldehyde/*ME; Buffers; Catalysis; HEPES/PD; Nuclear Magnetic Resonance; Phosphates/*PD; Protein Binding; Ribonuclease, Pancreatic/AI/*ME; Support, U.S. Gov't, Non-P.H.S.; Support, U.S. Gov't, P.H.S..
.T
The binding of acetaldehyde to the active site of ribonuclease: alterations in catalytic activity and effects of phosphate.
.P
JOURNAL ARTICLE.
.W
Ribonuclease A was reacted with [1-13C,1,2-14C]acetaldehyde and sodium cyanoborohydride in the presence or absence of 0.2 M phosphate. After several hours of incubation at 4 degrees C (pH 7.4) stable acetaldehyde-RNase adducts were formed, and the extent of their formation was similar regardless of the presence of phosphate. Although the total amount of covalent binding was comparable in the absence or presence of phosphate, this active site ligand prevented the inhibition of enzymatic activity seen in its absence. This protective action of phosphate diminished with progressive ethylation of RNase, indicating that the reversible association of phosphate with the active site lysyl residue was overcome by the irreversible process of reductive ethylation. Modified RNase was analysed using 13C proton decoupled NMR spectroscopy. Peaks arising from the covalent binding of enriched acetaldehyde to free amino groups in the absence of phosphate were as follows: NH2-terminal alpha amino group, 47.3 ppm; bulk ethylation at epsilon amino groups of nonessential lysyl residues, 43.0 ppm; and the epsilon amino group of lysine-41 at the active site, 47.4 ppm. In the spectrum of RNase ethylated in the presence of phosphate, the peak at 47.4 ppm was absent. When RNase was selectively premethylated in the presence of phosphate, to block all but the active site lysyl residues and then ethylated in its absence, the signal at 43.0 ppm was greatly diminished, and that arising from the active site lysyl residue at 47.4 ppm was enhanced. These results indicate that phosphate specifically protected the active site lysine from reaction with acetaldehyde, and that modification of this lysine by acetaldehyde adduct formation resulted in inhibition of catalytic activity.
.A
Mauch TJ; Tuma DJ; Sorrell MF.

Appendix B

Recall and precision tables of metrics for multi-word extraction

The following tables show the complete results of the fine-grained evaluation of the metrics displayed in Table 3.2.

n	DICE	ME	FREQ	PMI	SRE	SCP	TFIDF
100	0	14	1	0	33	1	4
200	0.5	13.5	1	0.5	30	1	2.5
300	0.66667	14.66667	1.33333	0.33333	28.66667	1	2
400	1	14.5	1.5	0.25	27.5	1	3
500	1	14.6	1.6	0.2	29.4	0.8	2.4
600	1	15.5	1.5	0.66667	29.16667	1	2.16667
700	0.85714	15.28571	1.42857	0.57143	27.85714	0.85714	2.42857
800	0.875	15.875	1.625	0.5	27.375	1	2.25
900	1	15.44444	1.77778	0.44444	26.88889	1	2.11111
1000	1.1	16.1	1.8	0.5	26.6	1	2.1
1100	1.09091	16.36364	1.90909	0.45455	26.90909	1	2.09091
1200	1	16.25	1.83333	0.41667	26.58333	1	1.91667
1300	0.92308	16.23077	2	0.38462	26.23077	0.92308	2.07692
1400	0.85714	16.21429	2.07143	0.35714	26.28571	0.85714	2.07143
1500	0.93333	16.46667	2.06667	0.33333	26.26667	0.8	2.33333
1600	0.9375	16.0625	2	0.3125	26.25	0.75	2.1875
1700	0.88235	15.94118	2	0.41176	25.88235	0.82353	2.17647
1800	0.83333	16.16667	2.05556	0.5	25.33333	0.83333	2.27778

Continued on next page

Table B.1 – continued from previous page

n	DICE	ME	FREQ	PMI	SRE	SCP	TFIDF
1900	0.78947	16.21053	2.15789	0.47368	24.94737	0.84211	2.15789
2000	0.8	16.1	2.3	0.45	24.4	0.8	2.1
2100	0.85714	15.7619	2.2381	0.42857	24.19048	0.80952	2
2200	0.90909	15.63636	2.13636	0.45455	23.54545	0.81818	1.95455
2300	0.91304	15.3913	2.26087	0.52174	23.56522	0.91304	1.95652
2400	0.95833	15.20833	2.20833	0.54167	23.33333	0.875	2.04167
2500	0.96	15.24	2.28	0.52	22.96	0.84	2
2600	0.92308	15.26923	2.34615	0.53846	22.88462	0.84615	1.96154
2700	0.92593	15.18519	2.37037	0.51852	22.48148	0.85185	1.92593
2800	0.92857	15.03571	2.35714	0.57143	22.17857	0.85714	1.85714
2900	0.93103	14.96552	2.31034	0.55172	22	0.86207	1.7931
3000	0.96667	15.03333	2.33333	0.6	21.7	0.9	1.8
3100	0.96774	15.06452	2.41935	0.6129	21.74194	0.87097	1.80645
3200	0.96875	15.03125	2.40625	0.65625	21.46875	0.875	1.78125
3300	0.9697	15.12121	2.36364	0.63636	21.42424	0.93939	1.75758
3400	0.97059	15.08824	2.38235	0.61765	21.41176	0.94118	1.73529
3500	0.97143	15.02857	2.42857	0.65714	21.45714	0.91429	1.77143
3600	0.94444	15.11111	2.41667	0.63889	21.30556	0.94444	1.80556
3700	0.94595	14.97297	2.51351	0.64865	21.24324	0.91892	1.78378
3800	0.94737	14.97368	2.55263	0.65789	21.21053	0.92105	1.86842
3900	0.94872	14.87179	2.53846	0.64103	21.02564	0.94872	1.89744
4000	0.925	14.675	2.5	0.625	20.875	0.95	1.875
4100	0.90244	14.65854	2.5122	0.63415	20.78049	0.92683	1.82927
4200	0.92857	14.64286	2.5	0.61905	20.61905	0.92857	1.80952
4300	0.90698	14.60465	2.46512	0.62791	20.53488	0.95349	1.7907
4400	0.93182	14.59091	2.47727	0.61364	20.43182	0.93182	1.75
4500	0.95556	14.6	2.46667	0.62222	20.31111	0.91111	1.73333
4600	0.93478	14.52174	2.5	0.63043	20.08696	0.91304	1.71739
4700	0.93617	14.46809	2.46809	0.61702	19.89362	0.93617	1.68085
4800	0.91667	14.54167	2.52083	0.625	19.8125	0.91667	1.64583
4900	0.89796	14.42857	2.53061	0.63265	19.69388	0.89796	1.61224
5000	0.88	14.34	2.58	0.62	19.5	0.9	1.58
5100	0.86275	14.21569	2.54902	0.60784	19.45098	0.88235	1.56863
5200	0.86538	14.19231	2.57692	0.59615	19.36538	0.88462	1.53846
5300	1.16981	14.07547	2.60377	0.58491	19.24528	1.18868	1.50943

Continued on next page

Table B.1 – continued from previous page

n	DICE	ME	FREQ	PMI	SRE	SCP	TFIDF
5400	1.33333	13.96296	2.62963	0.59259	19.16667	1.33333	1.5
5500	1.32727	14	2.67273	0.58182	19.05455	1.41818	1.49091
5600	1.53571	14.01786	2.71429	0.58929	18.91071	1.51786	1.48214
5700	1.7193	13.92982	2.7193	0.57895	18.80702	1.73684	1.47368
5800	1.93103	13.84483	2.75862	0.58621	18.63793	1.82759	1.46552
5900	1.91525	13.81356	2.76271	0.57627	18.54237	1.81356	1.44068
6000	1.9	13.8	2.78333	0.56667	18.43333	1.78333	1.41667
6100	1.86885	13.72131	2.7377	0.55738	18.2459	1.7541	1.40984
6200	1.83871	13.59677	2.74194	0.54839	18.22581	1.74194	1.3871
6300	1.80952	13.63492	2.74603	0.5873	18.12698	1.73016	1.36508
6400	1.78125	13.54688	2.73438	0.57813	18.10938	1.70313	1.34375
6500	1.75385	13.52308	2.75385	0.56923	18.04615	1.69231	1.33846
6600	1.72727	13.51515	2.78788	0.57576	17.86364	1.66667	1.36364
6700	1.71642	13.43284	2.79104	0.56716	17.65672	1.64179	1.35821
6800	1.70588	13.42647	2.77941	0.55882	17.55882	1.64706	1.36765
6900	1.69565	13.33333	2.78261	0.56522	17.46377	1.62319	1.37681
7000	1.67143	13.27143	2.8	0.57143	17.44286	1.6	1.37143
7100	1.67606	13.28169	2.83099	0.56338	17.30986	1.57746	1.3662
7200	1.65278	13.23611	2.81944	0.55556	17.20833	1.55556	1.38889
7300	1.63014	13.19178	2.86301	0.54795	17.16438	1.53425	1.36986
7400	1.60811	13.10811	2.90541	0.54054	17.14865	1.52703	1.41892
7500	1.61333	13.13333	2.88	0.53333	17.01333	1.50667	1.48
7600	1.59211	13.05263	2.88158	0.55263	16.81579	1.48684	1.5
7700	1.58442	13.03896	2.8961	0.54545	16.5974	1.46753	1.49351
7800	1.5641	12.94872	2.88462	0.53846	16.47436	1.44872	1.5
7900	1.55696	12.87342	2.87342	0.53165	16.40506	1.43038	1.48101
8000	1.5375	12.9	2.9125	0.5375	16.35	1.425	1.525
8100	1.51852	12.83951	2.93827	0.55556	16.30864	1.40741	1.54321
8200	1.5122	12.78049	2.91463	0.57317	16.2561	1.39024	1.53659
8300	1.49398	12.77108	2.92771	0.57831	16.18072	1.37349	1.56627
8400	1.47619	12.7381	2.94048	0.58333	16.13095	1.35714	1.55952
8500	1.45882	12.67059	2.96471	0.6	16.08235	1.34118	1.55294
8600	1.44186	12.66279	2.96512	0.59302	16.06977	1.32558	1.55814
8700	1.43678	12.58621	2.96552	0.5977	15.96552	1.36782	1.57471
8800	1.44318	12.57955	2.96591	0.61364	15.90909	1.42045	1.56818

Continued on next page

Table B.1 – continued from previous page

n	DICE	ME	FREQ	PMI	SRE	SCP	TFIDF
8900	1.51685	12.55056	2.95506	0.60674	15.85393	1.51685	1.5618
9000	1.61111	12.52222	2.94444	0.61111	15.81111	1.52222	1.54444
9100	1.6044	12.50549	2.94505	0.62637	15.73626	1.63736	1.53846
9200	1.67391	12.51087	2.98913	0.61957	15.6413	1.66304	1.52174
9300	1.74194	12.44086	3	0.6129	15.56989	1.65591	1.50538
9400	1.7234	12.42553	3	0.62766	15.51064	1.6383	1.5
9500	1.71579	12.37895	3	0.62105	15.45263	1.64211	1.48421
9600	1.70833	12.35417	2.98958	0.64583	15.33333	1.63542	1.47917
9700	1.69072	12.29897	2.98969	0.65979	15.25773	1.62887	1.48454
9800	1.67347	12.26531	3	0.65306	15.22449	1.63265	1.4898
9900	1.66667	12.27273	3.0101	0.64646	15.17172	1.61616	1.51515
10000	1.66	12.25	3	0.64	15.15	1.6	1.52

Table B.1: Precision of MWU extraction metrics on TREC against MESH

n	DICE	ME	FREQ	PMI	SRE	SCP	TFIDF
100	0	0.09961	0.00711	0	0.23479	0.00711	0.01082
200	0.00711	0.1921	0.01423	0.00711	0.42689	0.01423	0.01352
300	0.01423	0.31306	0.02846	0.00711	0.61188	0.02134	0.01622
400	0.02846	0.41266	0.04269	0.00711	0.78264	0.02846	0.03245
500	0.03557	0.51939	0.05692	0.00711	1.04589	0.02846	0.03245
600	0.04269	0.66169	0.06403	0.02846	1.24511	0.04269	0.03515
700	0.04269	0.76129	0.07115	0.02846	1.38741	0.04269	0.04597
800	0.0498	0.90359	0.09249	0.02846	1.55816	0.05692	0.04867
900	0.06403	0.98897	0.11384	0.02846	1.72181	0.06403	0.05137
1000	0.07826	1.1455	0.12807	0.03557	1.89256	0.07115	0.05678
1100	0.08538	1.28068	0.14941	0.03557	2.10601	0.07826	0.06219
1200	0.08538	1.38741	0.15653	0.03557	2.26965	0.08538	0.06219
1300	0.08538	1.50125	0.18499	0.03557	2.42618	0.08538	0.073
1400	0.08538	1.61508	0.20633	0.03557	2.61829	0.08538	0.07841
1500	0.09961	1.75738	0.22056	0.03557	2.80327	0.08538	0.09464
1600	0.10672	1.82853	0.22768	0.03557	2.98826	0.08538	0.09464
1700	0.10672	1.92814	0.24191	0.0498	3.13056	0.09961	0.10004

Continued on next page

Table B.2 – continued from previous page

n	DICE	ME	FREQ	PMI	SRE	SCP	TFIDF
1800	0.10672	2.07044	0.26325	0.06403	3.2444	0.10672	0.11086
1900	0.10672	2.19139	0.29171	0.06403	3.37247	0.11384	0.11086
2000	0.11384	2.291	0.32729	0.06403	3.47207	0.11384	0.11356
2100	0.12807	2.35503	0.3344	0.06403	3.61437	0.12095	0.11356
2200	0.1423	2.44753	0.3344	0.07115	3.68552	0.12807	0.11627
2300	0.14941	2.51868	0.36998	0.08538	3.85628	0.14941	0.12167
2400	0.16364	2.59694	0.37709	0.09249	3.98435	0.14941	0.13249
2500	0.17076	2.71078	0.40555	0.09249	4.08396	0.14941	0.13519
2600	0.17076	2.82462	0.43401	0.09961	4.23337	0.15653	0.1379
2700	0.17787	2.91711	0.45535	0.09961	4.31875	0.16364	0.1406
2800	0.18499	2.99538	0.46958	0.11384	4.41836	0.17076	0.1406
2900	0.1921	3.08787	0.4767	0.11384	4.53931	0.17787	0.1406
3000	0.20633	3.20882	0.49804	0.12807	4.6318	0.1921	0.14601
3100	0.21345	3.32266	0.53362	0.13518	4.79545	0.1921	0.15142
3200	0.22056	3.42227	0.54785	0.14941	4.88794	0.19922	0.15412
3300	0.22768	3.55034	0.55496	0.14941	5.03024	0.22056	0.15682
3400	0.23479	3.64995	0.57631	0.14941	5.17965	0.22768	0.15953
3500	0.24191	3.74244	0.60477	0.16364	5.34329	0.22768	0.16764
3600	0.24191	3.87051	0.619	0.16364	5.45713	0.24191	0.17575
3700	0.24902	3.94166	0.66169	0.17076	5.59232	0.24191	0.17846
3800	0.25614	4.04838	0.69015	0.17787	5.73461	0.24902	0.19197
3900	0.26325	4.12665	0.70438	0.17787	5.83422	0.26325	0.20009
4000	0.26325	4.17645	0.71149	0.17787	5.94095	0.27037	0.20279
4100	0.26325	4.27606	0.73284	0.18499	6.0619	0.27037	0.20279
4200	0.27748	4.37567	0.74707	0.18499	6.16151	0.27748	0.20549
4300	0.27748	4.46816	0.75418	0.1921	6.28246	0.29171	0.2082
4400	0.29171	4.56777	0.77552	0.1921	6.3963	0.29171	0.2082
4500	0.30594	4.67449	0.78975	0.19922	6.50302	0.29171	0.2109
4600	0.30594	4.75276	0.81821	0.20633	6.57417	0.29883	0.21361
4700	0.31306	4.83814	0.82533	0.20633	6.65244	0.31306	0.21361
4800	0.31306	4.9662	0.8609	0.21345	6.76628	0.31306	0.21361
4900	0.31306	5.03024	0.88225	0.22056	6.86588	0.31306	0.21361
5000	0.31306	5.10139	0.91782	0.22056	6.93703	0.32017	0.21361
5100	0.31306	5.15831	0.92494	0.22056	7.05799	0.32017	0.21631
5200	0.32017	5.2508	0.9534	0.22056	7.16471	0.32729	0.21631

Continued on next page

Table B.2 – continued from previous page

n	DICE	ME	FREQ	PMI	SRE	SCP	TFIDF
5300	0.44112	5.30772	0.98186	0.22056	7.2572	0.44824	0.21631
5400	0.51227	5.36464	1.01032	0.22768	7.36393	0.51227	0.21901
5500	0.51939	5.47848	1.04589	0.22768	7.45642	0.55496	0.22172
5600	0.61188	5.5852	1.08147	0.23479	7.53469	0.60477	0.22442
5700	0.69726	5.64924	1.10281	0.23479	7.62718	0.70438	0.22713
5800	0.79687	5.71327	1.13838	0.24191	7.69121	0.75418	0.22983
5900	0.80398	5.79865	1.15973	0.24191	7.78371	0.76129	0.22983
6000	0.8111	5.89114	1.18819	0.24191	7.86909	0.76129	0.22983
6100	0.8111	5.95518	1.18819	0.24191	7.91889	0.76129	0.23253
6200	0.8111	5.99787	1.20953	0.24191	8.03984	0.76841	0.23253
6300	0.8111	6.1117	1.23088	0.26325	8.12522	0.77552	0.23253
6400	0.8111	6.16862	1.24511	0.26325	8.24618	0.77552	0.23253
6500	0.8111	6.254	1.27357	0.26325	8.34578	0.78264	0.23524
6600	0.8111	6.3465	1.30914	0.27037	8.38847	0.78264	0.24335
6700	0.81821	6.40342	1.33049	0.27037	8.41693	0.78264	0.24605
6800	0.82533	6.49591	1.34472	0.27037	8.4952	0.79687	0.25146
6900	0.83244	6.54571	1.36606	0.27748	8.57346	0.79687	0.25687
7000	0.83244	6.60975	1.39452	0.2846	8.6873	0.79687	0.25957
7100	0.84667	6.70936	1.4301	0.2846	8.74422	0.79687	0.26228
7200	0.84667	6.78051	1.44433	0.2846	8.81537	0.79687	0.27039
7300	0.84667	6.85165	1.48702	0.2846	8.91498	0.79687	0.27039
7400	0.84667	6.90146	1.5297	0.2846	9.02882	0.80398	0.28391
7500	0.8609	7.00818	1.53682	0.2846	9.07862	0.80398	0.30013
7600	0.8609	7.05799	1.55816	0.29883	9.09285	0.80398	0.30824
7700	0.86802	7.14337	1.58662	0.29883	9.09285	0.80398	0.31095
7800	0.86802	7.18605	1.60085	0.29883	9.14265	0.80398	0.31635
7900	0.87513	7.23586	1.61508	0.29883	9.22092	0.80398	0.31635
8000	0.87513	7.34258	1.65777	0.30594	9.3063	0.8111	0.32987
8100	0.87513	7.3995	1.69335	0.32017	9.39879	0.8111	0.33798
8200	0.88225	7.45642	1.70046	0.3344	9.48417	0.8111	0.34069
8300	0.88225	7.5418	1.72892	0.34152	9.55532	0.8111	0.3515
8400	0.88225	7.61295	1.75738	0.34863	9.6407	0.8111	0.35421
8500	0.88225	7.66275	1.79296	0.36286	9.72608	0.8111	0.35691
8600	0.88225	7.74813	1.8143	0.36286	9.8328	0.8111	0.36232
8700	0.88936	7.79082	1.83565	0.36998	9.8826	0.84667	0.37043

Continued on next page

Table B.2 – continued from previous page

n	DICE	ME	FREQ	PMI	SRE	SCP	TFIDF
8800	0.90359	7.8762	1.85699	0.3842	9.96087	0.88936	0.37313
8900	0.96051	7.94735	1.87122	0.3842	10.03913	0.96051	0.37584
9000	1.03166	8.0185	1.88545	0.39132	10.12451	0.97474	0.37584
9100	1.03878	8.09676	1.90679	0.40555	10.18855	1.06012	0.37854
9200	1.0957	8.18926	1.9566	0.40555	10.23835	1.08858	0.37854
9300	1.15261	8.23195	1.98506	0.40555	10.30238	1.0957	0.37854
9400	1.15261	8.31021	2.0064	0.41978	10.37353	1.0957	0.38125
9500	1.15973	8.36713	2.02775	0.41978	10.44468	1.10993	0.38125
9600	1.16684	8.43828	2.04198	0.44112	10.47314	1.11704	0.38395
9700	1.16684	8.48808	2.06332	0.45535	10.53006	1.12416	0.38936
9800	1.16684	8.55212	2.09178	0.45535	10.61544	1.13838	0.39477
9900	1.17396	8.64461	2.12024	0.45535	10.68659	1.13838	0.40558
10000	1.18107	8.71576	2.13447	0.45535	10.77908	1.13838	0.41099
10000	1.66	12.25	3	0.64	15.15	1.6	1.52

Table B.2: Recall of MWU extraction metrics on TREC against MESH

n	SRE	ME	DICE	PMI	FREQ	SCP	TFIDF
100	43	18	3	0	6	1	0
200	41.5	18.5	3	0.5	4.5	2.5	0.5
300	40.33333	19.66667	2	0.66667	4.33333	2.66667	1
400	36.5	18	2.5	0.75	5	2.25	1
500	36.6	19.2	2.2	0.6	4.4	2.4	0.8
600	35.66667	19.83333	2.16667	0.83333	4.5	2.16667	0.83333
700	34	19.71429	2	1	4.85714	2.28571	0.71429
800	32.625	20	2.125	1.125	4.75	2.125	0.875
900	31.77778	19.44444	2	1.11111	4.77778	2.11111	0.88889
1000	31.2	19.3	1.9	1.3	4.4	2.3	0.8
1100	30.09091	19.63636	1.90909	1.18182	4.27273	2.36364	0.72727
1200	30.33333	19.66667	1.75	1.16667	4.33333	2.25	0.66667
1300	29.53846	19.30769	1.69231	1.15385	4.23077	2.15385	0.69231
1400	28.57143	19.07143	1.71429	1.14286	4	2.07143	0.78571
1500	28	19	1.73333	1.26667	4.06667	2.06667	1.06667

Continued on next page

Table B.3 – continued from previous page

n	SRE	ME	DICE	PMI	FREQ	SCP	TFIDF
1600	27.5625	19.0625	1.6875	1.1875	3.875	2.0625	1.1875
1700	26.76471	18.70588	1.64706	1.17647	3.94118	2.29412	1.29412
1800	26.16667	18.61111	1.55556	1.22222	3.83333	2.22222	1.5
1900	25.63158	18.78947	1.52632	1.26316	3.73684	2.21053	1.52632
2000	25.1	18.65	1.65	1.35	3.85	2.25	1.45
2100	24.95238	18.33333	1.57143	1.33333	3.80952	2.2381	1.38095
2200	24.36364	18.09091	1.63636	1.27273	3.72727	2.27273	1.31818
2300	24.13043	17.69565	1.69565	1.34783	3.65217	2.21739	1.30435
2400	23.79167	17.54167	1.66667	1.29167	3.58333	2.16667	1.29167
2500	23.44	17.36	1.72	1.32	3.8	2.08	1.24
2600	23	17.15385	1.73077	1.34615	3.96154	2.11538	1.26923
2700	22.59259	17.03704	1.81481	1.2963	3.92593	2.14815	1.22222
2800	22.5	17.07143	1.85714	1.39286	3.92857	2.10714	1.17857
2900	22.13793	16.96552	1.89655	1.34483	3.86207	2.03448	1.17241
3000	21.9	16.93333	1.9	1.33333	3.93333	2.06667	1.2
3100	21.70968	16.93548	1.96774	1.29032	4.03226	2.03226	1.25806
3200	21.5625	16.90625	2.03125	1.4375	4.03125	2.09375	1.28125
3300	21.48485	16.84848	2.0303	1.39394	3.93939	2.12121	1.30303
3400	21.20588	16.73529	2.02941	1.35294	3.94118	2.11765	1.32353
3500	20.97143	16.68571	2.11429	1.42857	4.02857	2.14286	1.31429
3600	20.77778	16.55556	2.13889	1.41667	3.94444	2.13889	1.41667
3700	20.7027	16.32432	2.16216	1.40541	4.08108	2.16216	1.43243
3800	20.57895	16.18421	2.18421	1.42105	4.15789	2.13158	1.44737
3900	20.48718	16.02564	2.17949	1.38462	4.10256	2.10256	1.4359
4000	20.35	15.95	2.125	1.425	4.075	2.125	1.45
4100	20.2439	15.82927	2.19512	1.41463	4.17073	2.07317	1.41463
4200	20	15.66667	2.14286	1.38095	4.19048	2.07143	1.38095
4300	19.83721	15.46512	2.11628	1.37209	4.11628	2.09302	1.37209
4400	19.77273	15.43182	2.15909	1.36364	4.06818	2.09091	1.34091
4500	19.64444	15.24444	2.15556	1.4	4.08889	2.06667	1.33333
4600	19.56522	15.15217	2.13043	1.47826	4.06522	2.08696	1.36957
4700	19.42553	15.02128	2.12766	1.46809	4.08511	2.10638	1.34043
4800	19.3125	15.10417	2.10417	1.45833	4.14583	2.08333	1.3125
4900	19.34694	15	2.10204	1.42857	4.14286	2.06122	1.30612
5000	19.2	15.04	2.1	1.44	4.12	2.02	1.28

Continued on next page

Table B.3 – continued from previous page

n	SRE	ME	DICE	PMI	FREQ	SCP	TFIDF
5100	19.23529	14.96078	2.09804	1.47059	4.07843	2.01961	1.27451
5200	19.09615	14.92308	2.11538	1.51923	4.13462	2.11538	1.28846
5300	19	14.84906	2.69811	1.49057	4.15094	2.69811	1.30189
5400	18.83333	14.72222	2.90741	1.5	4.22222	2.90741	1.27778
5500	18.74545	14.6	3	1.47273	4.25455	3.05455	1.25455
5600	18.66071	14.55357	3.26786	1.48214	4.21429	3.23214	1.23214
5700	18.49123	14.47368	3.59649	1.50877	4.21053	3.57895	1.22807
5800	18.43103	14.41379	4	1.51724	4.2069	3.75862	1.2069
5900	18.28814	14.47458	3.94915	1.52542	4.18644	3.74576	1.22034
6000	18.23333	14.33333	3.91667	1.53333	4.16667	3.68333	1.21667
6100	18.18033	14.21311	3.85246	1.5082	4.13115	3.62295	1.21311
6200	18.06452	14.1129	3.85484	1.48387	4.09677	3.62903	1.19355
6300	17.96825	14.03175	3.8254	1.50794	4.11111	3.5873	1.1746
6400	17.89063	13.98437	3.76563	1.51563	4.07813	3.5625	1.1875
6500	17.73846	13.96923	3.75385	1.50769	4.06154	3.56923	1.18462
6600	17.62121	13.90909	3.72727	1.54545	4.06061	3.54545	1.19697
6700	17.46269	13.8806	3.73134	1.56716	4.07463	3.52239	1.19403
6800	17.51471	13.86765	3.70588	1.55882	4.01471	3.51471	1.19118
6900	17.44928	13.78261	3.65217	1.56522	4	3.49275	1.18841
7000	17.31429	13.7	3.61429	1.57143	3.98571	3.44286	1.2
7100	17.22535	13.6338	3.60563	1.5493	4	3.39437	1.22535
7200	17.15278	13.58333	3.56944	1.52778	3.98611	3.36111	1.26389
7300	16.94521	13.54795	3.57534	1.53425	4.0137	3.32877	1.24658
7400	16.90541	13.44595	3.54054	1.54054	4	3.31081	1.25676
7500	16.82667	13.37333	3.52	1.53333	4	3.26667	1.28
7600	16.73684	13.38158	3.5	1.55263	3.94737	3.22368	1.27632
7700	16.54545	13.28571	3.49351	1.53247	3.8961	3.18182	1.32468
7800	16.44872	13.20513	3.46154	1.52564	3.88462	3.19231	1.32051
7900	16.37975	13.11392	3.4557	1.50633	3.87342	3.1519	1.31646
8000	16.2875	13.0625	3.4125	1.5	3.8875	3.125	1.375
8100	16.17284	13	3.40741	1.53086	3.8642	3.08642	1.41975
8200	16.09756	12.96341	3.39024	1.56098	3.82927	3.07317	1.43902
8300	16.0241	12.91566	3.3494	1.56627	3.79518	3.03614	1.46988
8400	15.94048	12.86905	3.32143	1.54762	3.80952	3.03571	1.4881
8500	15.85882	12.81176	3.29412	1.54118	3.77647	3.02353	1.48235

Continued on next page

Table B.3 – continued from previous page

n	SRE	ME	DICE	PMI	FREQ	SCP	TFIDF
8600	15.81395	12.72093	3.27907	1.53488	3.76744	2.98837	1.47674
8700	15.75862	12.73563	3.25287	1.52874	3.77011	3.06897	1.49425
8800	15.70455	12.73864	3.23864	1.52273	3.75	3.25	1.5
8900	15.66292	12.70787	3.50562	1.52809	3.73034	3.42697	1.50562
9000	15.58889	12.64444	3.66667	1.56667	3.72222	3.51111	1.52222
9100	15.53846	12.62637	3.7033	1.6044	3.71429	3.75824	1.52747
9200	15.5	12.6087	3.94565	1.58696	3.71739	3.80435	1.52174
9300	15.44086	12.56989	4.08602	1.5914	3.72043	3.7957	1.52688
9400	15.39362	12.51064	4.06383	1.57447	3.71277	3.7766	1.53191
9500	15.30526	12.46316	4.05263	1.56842	3.71579	3.77895	1.52632
9600	15.19792	12.44792	4.04167	1.58333	3.73958	3.78125	1.52083
9700	15.09278	12.45361	4	1.60825	3.73196	3.74227	1.52577
9800	15.06122	12.41837	3.97959	1.60204	3.7551	3.71429	1.52041
9900	15.0404	12.39394	3.9697	1.60606	3.76768	3.68687	1.51515
10000	14.95	12.36	3.96	1.59	3.74	3.65	1.51

Table B.3: Precision of MWU extraction metrics on TREC against SNOMED-CT

n	SRE	ME	DICE	PMI	FREQ	SCP	TFIDF
100	0.25809	0.10804	0.01801	0	0.03601	0.006	0
200	0.49817	0.22208	0.03601	0.006	0.05402	0.03001	0.006
300	0.72625	0.35412	0.03601	0.012	0.07803	0.04802	0.01801
400	0.8763	0.43215	0.06002	0.01801	0.12004	0.05402	0.02401
500	1.09837	0.5762	0.06602	0.01801	0.13204	0.07202	0.02401
600	1.28444	0.71424	0.07803	0.03001	0.16206	0.07803	0.03001
700	1.42849	0.82828	0.08403	0.04201	0.20407	0.09603	0.03001
800	1.56653	0.96033	0.10203	0.05402	0.22808	0.10203	0.04201
900	1.71658	1.05036	0.10804	0.06002	0.25809	0.11404	0.04802
1000	1.87264	1.15839	0.11404	0.07803	0.26409	0.13805	0.04802
1100	1.98668	1.29644	0.12604	0.07803	0.2821	0.15605	0.04802
1200	2.18474	1.41648	0.12604	0.08403	0.31211	0.16206	0.04802
1300	2.30478	1.50651	0.13204	0.09003	0.33011	0.16806	0.05402
1400	2.40082	1.60254	0.14405	0.09603	0.33611	0.17406	0.06602

Continued on next page

Table B.4 – continued from previous page

n	SRE	ME	DICE	PMI	FREQ	SCP	TFIDF
1500	2.52086	1.71058	0.15605	0.11404	0.36612	0.18606	0.09603
1600	2.6469	1.83062	0.16206	0.11404	0.37213	0.19807	0.11404
1700	2.73093	1.90865	0.16806	0.12004	0.40214	0.23408	0.13204
1800	2.82696	2.01068	0.16806	0.13204	0.41414	0.24008	0.16206
1900	2.92299	2.14273	0.17406	0.14405	0.42614	0.25209	0.17406
2000	3.01302	2.23876	0.19807	0.16206	0.46216	0.27009	0.17406
2100	3.14507	2.31079	0.19807	0.16806	0.48016	0.2821	0.17406
2200	3.21709	2.38881	0.21607	0.16806	0.49217	0.3001	0.17406
2300	3.33113	2.44283	0.23408	0.18606	0.50417	0.3061	0.18006
2400	3.42717	2.52686	0.24008	0.18606	0.51618	0.31211	0.18606
2500	3.5172	2.60489	0.25809	0.19807	0.57019	0.31211	0.18606
2600	3.58922	2.67691	0.27009	0.21007	0.61821	0.33011	0.19807
2700	3.66124	2.76094	0.2941	0.21007	0.63622	0.34812	0.19807
2800	3.78129	2.86898	0.31211	0.23408	0.66022	0.35412	0.19807
2900	3.85331	2.953	0.33011	0.23408	0.67223	0.35412	0.20407
3000	3.94334	3.04904	0.34212	0.24008	0.70824	0.37213	0.21607
3100	4.03937	3.15107	0.36612	0.24008	0.75026	0.37813	0.23408
3200	4.14141	3.2471	0.39013	0.27609	0.77426	0.40214	0.24608
3300	4.25545	3.33713	0.40214	0.27609	0.78027	0.42014	0.25809
3400	4.32747	3.41516	0.41414	0.27609	0.80427	0.43215	0.27009
3500	4.4055	3.50519	0.44415	0.3001	0.84629	0.45015	0.27609
3600	4.48953	3.57722	0.46216	0.3061	0.85229	0.46216	0.3061
3700	4.59756	3.62523	0.48016	0.31211	0.90631	0.48016	0.31811
3800	4.6936	3.69126	0.49817	0.32411	0.94832	0.48617	0.33011
3900	4.79563	3.75128	0.51017	0.32411	0.96033	0.49217	0.33611
4000	4.88566	3.8293	0.51017	0.34212	0.97833	0.51017	0.34812
4100	4.98169	3.89532	0.54018	0.34812	1.02635	0.51017	0.34812
4200	5.04171	3.94934	0.54018	0.34812	1.05636	0.52218	0.34812
4300	5.11974	3.99136	0.54619	0.35412	1.06236	0.54018	0.35412
4400	5.22178	4.07539	0.57019	0.36012	1.07437	0.55219	0.35412
4500	5.3058	4.1174	0.5822	0.37813	1.10438	0.55819	0.36012
4600	5.40184	4.18342	0.5882	0.40814	1.12238	0.5762	0.37813
4700	5.47986	4.23744	0.6002	0.41414	1.15239	0.5942	0.37813
4800	5.56389	4.35148	0.60621	0.42014	1.19441	0.6002	0.37813
4900	5.68993	4.4115	0.61821	0.42014	1.21841	0.60621	0.38413

Continued on next page

Table B.4 – continued from previous page

n	SRE	ME	DICE	PMI	FREQ	SCP	TFIDF
5000	5.76196	4.51353	0.63021	0.43215	1.23642	0.60621	0.38413
5100	5.888	4.57956	0.64222	0.45015	1.24842	0.61821	0.39013
5200	5.96003	4.65758	0.66022	0.47416	1.29044	0.66022	0.40214
5300	6.04405	4.72361	0.85829	0.47416	1.32045	0.85829	0.41414
5400	6.10408	4.77162	0.94232	0.48617	1.36847	0.94232	0.41414
5500	6.1881	4.81964	0.99034	0.48617	1.40448	1.00834	0.41414
5600	6.27213	4.89166	1.09837	0.49817	1.41648	1.08637	0.41414
5700	6.32615	4.95168	1.23042	0.51618	1.44049	1.22442	0.42014
5800	6.41618	5.01771	1.39247	0.52818	1.4645	1.30844	0.42014
5900	6.4762	5.12574	1.39848	0.54018	1.4825	1.32645	0.43215
6000	6.56623	5.16175	1.41048	0.55219	1.50051	1.32645	0.43815
6100	6.65626	5.20377	1.41048	0.55219	1.51251	1.32645	0.44415
6200	6.72229	5.25179	1.43449	0.55219	1.52452	1.35046	0.44415
6300	6.79431	5.3058	1.44649	0.57019	1.55453	1.35646	0.44415
6400	6.87234	5.37183	1.44649	0.5822	1.56653	1.36847	0.45616
6500	6.92035	5.44985	1.4645	0.5882	1.58454	1.39247	0.46216
6600	6.98037	5.50987	1.4765	0.61221	1.60855	1.40448	0.47416
6700	7.02239	5.5819	1.50051	0.63021	1.63856	1.41648	0.48016
6800	7.14843	5.65992	1.51251	0.63622	1.63856	1.43449	0.48617
6900	7.22646	5.70794	1.51251	0.64822	1.65656	1.44649	0.49217
7000	7.27447	5.75596	1.51852	0.66022	1.67457	1.44649	0.50417
7100	7.3405	5.80998	1.53652	0.66022	1.70458	1.44649	0.52218
7200	7.41252	5.87	1.54252	0.66022	1.72259	1.45249	0.54619
7300	7.42452	5.93602	1.56653	0.67223	1.7586	1.4585	0.54619
7400	7.50855	5.97203	1.57253	0.68423	1.7766	1.4705	0.55819
7500	7.57458	6.02005	1.58454	0.69023	1.80061	1.4705	0.5762
7600	7.6346	6.10408	1.59654	0.70824	1.80061	1.4705	0.5822
7700	7.6466	6.14009	1.61455	0.70824	1.80061	1.4705	0.61221
7800	7.70062	6.1821	1.62055	0.71424	1.81862	1.49451	0.61821
7900	7.76664	6.21811	1.63856	0.71424	1.83662	1.49451	0.62421
8000	7.82066	6.27213	1.63856	0.72024	1.86663	1.50051	0.66022
8100	7.86267	6.32015	1.65656	0.74425	1.87864	1.50051	0.69023
8200	7.92269	6.38017	1.66857	0.76826	1.88464	1.51251	0.70824
8300	7.98271	6.43419	1.66857	0.78027	1.89064	1.51251	0.73225
8400	8.03673	6.48821	1.67457	0.78027	1.92065	1.53052	0.75026

Continued on next page

Table B.4 – continued from previous page

n	SRE	ME	DICE	PMI	FREQ	SCP	TFIDF
8500	8.09075	6.53622	1.68057	0.78627	1.92666	1.54252	0.75626
8600	8.16278	6.56623	1.69258	0.79227	1.94466	1.54252	0.76226
8700	8.2288	6.65026	1.69858	0.79827	1.96867	1.60254	0.78027
8800	8.29482	6.72829	1.71058	0.80427	1.98067	1.71658	0.79227
8900	8.36684	6.78831	1.87264	0.81628	1.99268	1.83062	0.80427
9000	8.42086	6.83032	1.98067	0.84629	2.01068	1.89664	0.82228
9100	8.48689	6.89634	2.02269	0.8763	2.02869	2.0527	0.83428
9200	8.55891	6.96237	2.17874	0.8763	2.0527	2.10071	0.84029
9300	8.61893	7.01639	2.28078	0.8883	2.07671	2.11872	0.85229
9400	8.68495	7.0584	2.29278	0.8883	2.09471	2.13072	0.86429
9500	8.72697	7.10642	2.31079	0.8943	2.11872	2.15473	0.8703
9600	8.75698	7.17244	2.32879	0.91231	2.15473	2.17874	0.8763
9700	8.78699	7.25047	2.32879	0.93632	2.17274	2.17874	0.8883
9800	8.85901	7.30448	2.3408	0.94232	2.20875	2.18474	0.8943
9900	8.93704	7.3645	2.3588	0.95432	2.23876	2.19074	0.90031
10000	8.97305	7.41852	2.37681	0.95432	2.24476	2.19074	0.90631

Table B.4: Recall of MWU extraction metrics on TREC against SNOMED-CT

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
100	4	4	29	1	4	73	4
200	5	5.5	34.5	2.5	5.5	73.5	3.5
300	3.66667	5	34.66667	3	5.33333	73.66667	3
400	4.25	5.25	32.75	4	4.25	68.5	3.5
500	4.4	5.2	34.2	4.2	5.2	68.8	3
600	4.33333	5.33333	35.83333	5	5.16667	65.66667	3.16667
700	4.42857	5.57143	36.85714	4.71429	5.14286	63.28571	3.42857
800	4.625	5.75	36.875	4.5	5.125	61.375	3.75
900	4.66667	5.88889	35.88889	4.66667	5.55556	60.66667	3.66667
1000	4.9	5.8	35.4	4.8	5.9	59.9	3.6
1100	5.18182	5.54545	34.90909	4.63636	5.90909	58.90909	3.63636
1200	5	5.5	34.41667	4.41667	6	58.5	3.58333
1300	4.92308	5.46154	34.61538	4.30769	6.15385	57.61538	3.84615

Continued on next page

Table B.5 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
1400	5.21429	5.5	34.28571	4.21429	5.78571	56.5	3.92857
1500	5.13333	5.66667	34.2	4.33333	5.66667	55.66667	4.26667
1600	5.1875	5.6875	34.0625	4.125	5.625	54.875	4.1875
1700	5.17647	5.76471	34.23529	4.05882	5.88235	53.82353	4.11765
1800	4.94444	5.77778	34.16667	4.11111	5.77778	53.05556	4.11111
1900	5	5.68421	34.26316	4.10526	5.63158	52.42105	4
2000	5.2	5.65	34.1	4.2	5.75	51.85	3.9
2100	5.19048	5.61905	33.80952	4.19048	5.66667	51.61905	3.85714
2200	5.31818	5.59091	33.77273	4.18182	5.81818	50.40909	3.72727
2300	5.30435	5.56522	33.3913	4.17391	5.86957	50	3.69565
2400	5.29167	5.58333	33.08333	4.16667	5.75	49.25	3.66667
2500	5.28	5.8	33.04	4.12	5.56	48.88	3.56
2600	5.38462	5.92308	32.73077	4.07692	5.5	48.19231	3.5
2700	5.44444	5.88889	32.48148	4	5.44444	47.62963	3.37037
2800	5.42857	5.85714	32.5	4.21429	5.35714	47.53571	3.25
2900	5.48276	5.82759	32.48276	4.10345	5.34483	47.03448	3.17241
3000	5.4	5.96667	32.43333	4.03333	5.43333	46.46667	3.23333
3100	5.3871	6.09677	32.32258	4.03226	5.35484	46.29032	3.29032
3200	5.53125	6.09375	32.15625	4.15625	5.46875	45.78125	3.28125
3300	5.45455	6.0303	31.81818	4.15152	5.54545	45.63636	3.27273
3400	5.5	5.97059	31.67647	4.08824	5.52941	45.14706	3.23529
3500	5.6	6.08571	31.77143	4.2	5.57143	44.82857	3.14286
3600	5.58333	5.97222	31.75	4.19444	5.55556	44.55556	3.25
3700	5.54054	5.94595	31.51351	4.13514	5.51351	44.32432	3.2973
3800	5.52632	6.07895	31.36842	4.15789	5.55263	44.21053	3.28947
3900	5.51282	6.02564	31.10256	4.10256	5.53846	43.74359	3.28205
4000	5.45	6.025	31	4.2	5.55	43.35	3.3
4100	5.58537	6.12195	30.73171	4.12195	5.46341	43.21951	3.26829
4200	5.52381	6.09524	30.59524	4.04762	5.47619	42.80952	3.2381
4300	5.46512	6.02326	30.4186	4.02326	5.44186	42.51163	3.18605
4400	5.59091	5.97727	30.27273	3.95455	5.47727	42.34091	3.11364
4500	5.64444	6.04444	30.26667	4	5.44444	42.17778	3.06667
4600	5.69565	6.15217	30.1087	4.04348	5.45652	41.93478	3.06522
4700	5.68085	6.23404	29.95745	4.06383	5.40426	41.53191	3
4800	5.625	6.35417	30.0625	4.04167	5.35417	41.25	2.9375

Continued on next page

Table B.5 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
4900	5.63265	6.36735	29.87755	4.02041	5.44898	41.22449	2.91837
5000	5.6	6.48	29.84	4.02	5.42	40.88	2.88
5100	5.56863	6.39216	29.78431	4.01961	5.47059	40.68627	2.84314
5200	5.55769	6.42308	29.71154	4.03846	5.55769	40.40385	2.82692
5300	6.43396	6.41509	29.67925	4.03774	6.43396	40.16981	2.81132
5400	6.96296	6.46296	29.48148	4.09259	6.88889	39.98148	2.75926
5500	7.14545	6.49091	29.30909	4.05455	7.18182	39.72727	2.70909
5600	7.66071	6.57143	29.16071	4.14286	7.57143	39.46429	2.66071
5700	8.15789	6.57895	29.01754	4.15789	8.15789	39.2807	2.64912
5800	8.7931	6.63793	28.84483	4.15517	8.39655	39.10345	2.60345
5900	8.69492	6.59322	28.83051	4.15254	8.35593	38.89831	2.59322
6000	8.65	6.61667	28.83333	4.18333	8.28333	38.68333	2.55
6100	8.54098	6.57377	28.78689	4.11475	8.18033	38.4918	2.55738
6200	8.51613	6.51613	28.54839	4.06452	8.19355	38.37097	2.54839
6300	8.42857	6.50794	28.46032	4.14286	8.12698	38.09524	2.50794
6400	8.32813	6.5	28.375	4.15625	8.0625	37.95313	2.48438
6500	8.24615	6.47692	28.36923	4.15385	7.98462	37.81538	2.46154
6600	8.16667	6.5	28.19697	4.15152	7.92424	37.60606	2.48485
6700	8.13433	6.49254	28.02985	4.16418	7.83582	37.35821	2.47761
6800	8.08824	6.47059	27.95588	4.16176	7.85294	37.20588	2.48529
6900	7.98551	6.43478	27.84058	4.14493	7.84058	36.97101	2.47826
7000	7.92857	6.44286	27.74286	4.17143	7.75714	36.75714	2.47143
7100	7.90141	6.49296	27.61972	4.16901	7.67606	36.64789	2.49296
7200	7.83333	6.43056	27.51389	4.16667	7.61111	36.44444	2.51389
7300	7.79452	6.43836	27.53425	4.13699	7.54795	36.21918	2.50685
7400	7.72973	6.45946	27.37838	4.13514	7.52703	36.06757	2.5
7500	7.68	6.42667	27.36	4.10667	7.44	36	2.61333
7600	7.64474	6.38158	27.28947	4.14474	7.35526	35.78947	2.61842
7700	7.62338	6.37662	27.19481	4.14286	7.2987	35.42857	2.67532
7800	7.57692	6.35897	27.03846	4.11538	7.26923	35.19231	2.69231
7900	7.58228	6.35443	26.92405	4.12658	7.18987	34.93671	2.75949
8000	7.5	6.3625	26.825	4.1625	7.1625	34.8	2.825
8100	7.49383	6.38272	26.74074	4.19753	7.11111	34.62963	2.82716
8200	7.47561	6.34146	26.73171	4.2439	7.06098	34.45122	2.84146
8300	7.42169	6.3012	26.66265	4.3012	7	34.31325	2.85542

Continued on next page

Table B.5 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
8400	7.40476	6.33333	26.63095	4.2619	6.96429	34.21429	2.83333
8500	7.35294	6.34118	26.56471	4.25882	6.90588	34.10588	2.85882
8600	7.31395	6.32558	26.51163	4.26744	6.83721	33.97674	2.84884
8700	7.26437	6.32184	26.44828	4.27586	6.97701	33.81609	2.86207
8800	7.27273	6.31818	26.39773	4.31818	7.25	33.73864	2.85227
8900	7.69663	6.32584	26.38202	4.32584	7.58427	33.67416	2.86517
9000	7.96667	6.34444	26.31111	4.36667	7.74444	33.53333	2.88889
9100	8.04396	6.36264	26.23077	4.3956	8.15385	33.38462	2.91209
9200	8.38043	6.38043	26.25	4.3913	8.19565	33.21739	2.8913
9300	8.5914	6.39785	26.12903	4.37634	8.1828	33.06452	2.89247
9400	8.54255	6.42553	26.05319	4.34043	8.12766	32.98936	2.90426
9500	8.49474	6.45263	25.95789	4.32632	8.09474	32.83158	2.89474
9600	8.5	6.46875	25.9375	4.38542	8.05208	32.66667	2.88542
9700	8.4433	6.4433	25.83505	4.40206	8.03093	32.54639	2.91753
9800	8.37755	6.45918	25.76531	4.39796	8.0102	32.40816	2.91837
9900	8.36364	6.49495	25.72727	4.39394	7.9899	32.27273	2.91919
10000	8.33	6.46	25.65	4.37	7.93	32.15	2.94

Table B.5: Precision of MWU extraction metrics on TREC corpus UMLS

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
100	0.01338	0.01338	0.09703	0.00335	0.01338	0.24425	0.01338
200	0.03346	0.03681	0.23087	0.01673	0.03681	0.49185	0.02342
300	0.03681	0.05019	0.34798	0.03011	0.05353	0.73945	0.03011
400	0.05688	0.07026	0.43832	0.05353	0.05688	0.91679	0.04684
500	0.07361	0.08699	0.57216	0.07026	0.08699	1.151	0.05019
600	0.08699	0.10707	0.71938	0.10038	0.10372	1.3183	0.06357
700	0.10372	0.13049	0.86325	0.11042	0.12045	1.48225	0.0803
800	0.1238	0.15391	0.98705	0.12045	0.13718	1.64285	0.10038
900	0.14053	0.17733	1.08074	0.14053	0.1673	1.82688	0.11042
1000	0.16395	0.19406	1.18446	0.1606	0.19741	2.00422	0.12045
1100	0.19072	0.2041	1.28484	0.17064	0.21749	2.16817	0.13384
1200	0.20076	0.22083	1.38187	0.17733	0.24091	2.34885	0.14388

Continued on next page

Table B.6 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
1300	0.21414	0.23756	1.50567	0.18737	0.26767	2.50611	0.1673
1400	0.24425	0.25764	1.60605	0.19741	0.27102	2.64664	0.18403
1500	0.25764	0.2844	1.71647	0.21749	0.2844	2.79386	0.21414
1600	0.27771	0.30448	1.82354	0.22083	0.30113	2.93773	0.22418
1700	0.29444	0.3279	1.94733	0.23087	0.33459	3.06153	0.23422
1800	0.29779	0.34798	2.05775	0.2476	0.34798	3.19537	0.2476
1900	0.31786	0.36136	2.1782	0.26098	0.35802	3.33255	0.25429
2000	0.34798	0.37809	2.28193	0.28106	0.38478	3.46974	0.26098
2100	0.36471	0.39482	2.37561	0.29444	0.39817	3.627	0.27102
2200	0.39147	0.41155	2.48603	0.30783	0.42828	3.71064	0.27437
2300	0.4082	0.42828	2.56968	0.32121	0.4517	3.84783	0.2844
2400	0.42493	0.44836	2.65667	0.33459	0.46174	3.9549	0.29444
2500	0.44166	0.48516	2.76374	0.34463	0.46509	4.08873	0.29779
2600	0.46843	0.51527	2.84739	0.35467	0.47847	4.19246	0.30448
2700	0.49185	0.532	2.93439	0.36136	0.49185	4.30287	0.30448
2800	0.50858	0.54873	3.0448	0.39482	0.50189	4.45344	0.30448
2900	0.532	0.56546	3.15187	0.39817	0.51862	4.56386	0.30783
3000	0.54204	0.59892	3.2556	0.40486	0.54539	4.66424	0.32456
3100	0.55877	0.63238	3.35263	0.41824	0.55543	4.80142	0.34129
3200	0.59223	0.65246	3.44297	0.44501	0.58554	4.9018	0.35132
3300	0.60227	0.66584	3.51323	0.45839	0.61231	5.03898	0.36136
3400	0.62569	0.67923	3.60357	0.46509	0.62904	5.13601	0.36805
3500	0.6558	0.71268	3.72068	0.49185	0.65246	5.24977	0.36805
3600	0.67253	0.71938	3.82441	0.50524	0.66919	5.36688	0.39147
3700	0.68592	0.73611	3.90136	0.51193	0.68257	5.48734	0.4082
3800	0.70265	0.77291	3.98836	0.52866	0.70599	5.62117	0.41824
3900	0.71938	0.7863	4.05862	0.53535	0.72272	5.70817	0.42828
4000	0.72941	0.80637	4.14896	0.56212	0.7428	5.80185	0.44166
4100	0.76622	0.83983	4.21588	0.56546	0.74949	5.929	0.44836
4200	0.77626	0.85656	4.29953	0.56881	0.76957	6.01599	0.45505
4300	0.7863	0.8666	4.37648	0.57885	0.78295	6.11637	0.45839
4400	0.8231	0.87998	4.45679	0.58219	0.80637	6.23348	0.45839
4500	0.84987	0.91009	4.55717	0.60227	0.81975	6.35059	0.46174
4600	0.87664	0.9469	4.63412	0.62234	0.83983	6.45431	0.47178
4700	0.89337	0.98036	4.71108	0.63907	0.84987	6.53127	0.47178

Continued on next page

Table B.6 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
4800	0.9034	1.02051	4.82819	0.64911	0.85991	6.62495	0.47178
4900	0.92348	1.04393	4.89845	0.65915	0.89337	6.75879	0.47847
5000	0.93686	1.08408	4.99214	0.67253	0.90675	6.83909	0.48181
5100	0.95025	1.09078	5.08248	0.68592	0.93352	6.94282	0.48516
5200	0.96698	1.11754	5.16947	0.70265	0.96698	7.02981	0.49185
5300	1.14096	1.13762	5.26316	0.71603	1.14096	7.1235	0.49854
5400	1.25807	1.16773	5.32673	0.73945	1.24469	7.22388	0.49854
5500	1.31495	1.1945	5.39365	0.74614	1.32164	7.31087	0.49854
5600	1.43541	1.2313	5.46391	0.77626	1.41868	7.39452	0.49854
5700	1.55586	1.25473	5.53418	0.79299	1.55586	7.49155	0.50524
5800	1.70643	1.28819	5.59775	0.80637	1.62947	7.58858	0.50524
5900	1.71647	1.30157	5.69144	0.81975	1.64955	7.67892	0.51193
6000	1.73654	1.32834	5.78847	0.83983	1.66293	7.76592	0.51193
6100	1.74323	1.34172	5.87546	0.83983	1.66962	7.85626	0.52197
6200	1.76665	1.35176	5.92231	0.84318	1.69974	7.95998	0.52866
6300	1.77669	1.37183	5.99926	0.87329	1.71312	8.03025	0.52866
6400	1.78338	1.39191	6.07622	0.89002	1.7265	8.12728	0.532
6500	1.79342	1.40864	6.16991	0.9034	1.73654	8.22431	0.53535
6600	1.80346	1.43541	6.22679	0.91679	1.74992	8.30461	0.54873
6700	1.82354	1.45548	6.28367	0.93352	1.75662	8.37488	0.55543
6800	1.84026	1.47221	6.36063	0.9469	1.78673	8.46522	0.56546
6900	1.84361	1.4856	6.42754	0.95694	1.81015	8.53548	0.57216
7000	1.85699	1.50902	6.49781	0.97701	1.81684	8.60909	0.57885
7100	1.87707	1.54248	6.56138	0.9904	1.82354	8.70613	0.59223
7200	1.88711	1.54917	6.6283	1.00378	1.83357	8.77974	0.60561
7300	1.90384	1.57259	6.72533	1.01047	1.84361	8.84666	0.61231
7400	1.91388	1.59936	6.77887	1.02386	1.86369	8.9303	0.619
7500	1.92726	1.61274	6.86586	1.03055	1.86703	9.03403	0.6558
7600	1.94399	1.62278	6.93947	1.05397	1.87038	9.10095	0.66584
7700	1.96406	1.64285	7.00639	1.06735	1.88042	9.12771	0.68926
7800	1.97745	1.65958	7.05658	1.07405	1.89715	9.1846	0.70265
7900	2.00422	1.67966	7.11681	1.09078	1.90049	9.23478	0.72941
8000	2.00756	1.70308	7.18038	1.1142	1.91722	9.31509	0.75618
8100	2.03098	1.72985	7.2473	1.13762	1.92726	9.38535	0.76622
8200	2.05106	1.73989	7.33429	1.16439	1.9373	9.45227	0.7796

Continued on next page

Table B.6 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
8300	2.0611	1.74992	7.40456	1.1945	1.94399	9.52923	0.79299
8400	2.08117	1.78004	7.48486	1.19785	1.95737	9.61622	0.79633
8500	2.09121	1.80346	7.55512	1.21123	1.96406	9.69987	0.81306
8600	2.10459	1.82019	7.62873	1.22796	1.96741	9.77683	0.81975
8700	2.11463	1.84026	7.699	1.24469	2.03098	9.84374	0.83314
8800	2.1414	1.86034	7.77261	1.27146	2.13471	9.93409	0.83983
8900	2.29197	1.88376	7.85626	1.28819	2.25851	10.02777	0.85321
9000	2.39904	1.91053	7.92318	1.31495	2.33212	10.09804	0.86994
9100	2.44923	1.9373	7.98675	1.33837	2.48268	10.16495	0.88667
9200	2.57972	1.96406	8.08044	1.35176	2.52284	10.22518	0.89002
9300	2.6734	1.99083	8.13063	1.3618	2.54626	10.28875	0.90006
9400	2.68679	2.02095	8.1942	1.36514	2.5563	10.37575	0.91344
9500	2.70017	2.05106	8.25108	1.37518	2.57303	10.43598	0.92013
9600	2.73028	2.07783	8.33138	1.40864	2.58641	10.49286	0.92682
9700	2.74032	2.09121	8.38492	1.42871	2.60648	10.56312	0.9469
9800	2.74701	2.11798	8.44849	1.4421	2.62656	10.62669	0.95694
9900	2.77044	2.15144	8.5221	1.45548	2.64664	10.69027	0.96698
10000	2.78716	2.16147	8.58233	1.46217	2.65333	10.75719	0.98371

Table B.6: Recall of MWU extraction metrics on TREC against UMLS

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
100	0	1	6	0	1	15	11
200	0	1	10.5	0	0.5	19.5	7.5
300	0	1	11.33333	0	0.33333	17	6.33333
400	0	1.5	12.75	0	0.25	15.75	5.25
500	0	1.4	13.6	0	0.2	14.6	4.8
600	0	1.33333	13	0	0.16667	13.66667	4.16667
700	0	1.57143	12.85714	0	0.14286	13.14286	4
800	0	1.625	12	0	0.125	13.25	3.625
900	0	1.66667	11.66667	0	0.11111	12.88889	3.55556
1000	0	1.8	11.4	0	0.1	13	3.6
1100	0	1.81818	11.18182	0	0.09091	12.81818	3.45455

Continued on next page

Table B.7 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
1200	0	1.91667	11.16667	0	0.08333	12.58333	3.25
1300	0	1.92308	11.23077	0	0.07692	12.46154	3.38462
1400	0	1.85714	11	0	0.07143	12	3.14286
1500	0	1.93333	10.73333	0	0.06667	11.93333	3.06667
1600	0	1.875	10.8125	0	0.0625	11.875	3
1700	0	1.82353	10.58824	0	0.05882	11.88235	3
1800	0	1.94444	10.5	0	0.05556	11.72222	3.11111
1900	0	2	10.42105	0	0.05263	11.42105	3.05263
2000	0	1.95	10.15	0	0.05	11.3	3
2100	0	1.90476	10.28571	0	0.04762	11.19048	3.04762
2200	0	1.86364	10.27273	0	0.04545	11.22727	3.09091
2300	0	1.95652	10.13043	0	0.04348	11.04348	2.95652
2400	0	1.95833	10.04167	0	0.04167	10.91667	2.95833
2500	0	2	9.92	0	0.04	10.8	2.96
2600	0	1.96154	9.88462	0	0.03846	10.84615	2.92308
2700	0	1.92593	9.96296	0	0.03704	10.85185	3.07407
2800	0	1.92857	10.07143	0	0.03571	10.75	3
2900	0	1.96552	10.06897	0	0.03448	10.7931	3.03448
3000	0	1.93333	9.86667	0	0.03333	10.56667	3.06667
3100	0	1.93548	9.93548	0	0.03226	10.54839	3.09677
3200	0	1.9375	9.90625	0	0.03125	10.65625	3
3300	0	1.9697	9.9697	0	0.0303	10.63636	3.06061
3400	0	1.97059	9.94118	0	0.02941	10.70588	3
3500	0	1.97143	9.8	0	0.02857	10.71429	3.05714
3600	0	2	9.88889	0	0.02778	10.52778	3.05556
3700	0	1.97297	9.83784	0	0.02703	10.40541	3.08108
3800	0	2	9.84211	0	0.02632	10.31579	3.02632
3900	0	2.02564	9.82051	0	0.02564	10.17949	3.10256
4000	0	2.1	9.75	0	0.025	10.15	3.2
4100	0	2.14634	9.70732	0	0.02439	10.12195	3.14634
4200	0	2.19048	9.61905	0	0.02381	10	3.09524
4300	0	2.18605	9.46512	0	0.02326	10	3.11628
4400	0	2.15909	9.52273	0	0.02273	10.02273	3.11364
4500	0	2.24444	9.46667	0	0.02222	9.93333	3.08889
4600	0	2.30435	9.5	0	0.02174	9.78261	3.04348

Continued on next page

Table B.7 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
4700	0	2.29787	9.34043	0	0.02128	9.65957	3.04255
4800	0	2.29167	9.3125	0	0.02083	9.58333	3
4900	0	2.34694	9.34694	0	0.02041	9.44898	2.93878
5000	0	2.36	9.32	0	0.02	9.26	2.94
5100	0	2.37255	9.23529	0	0.01961	9.07843	3.05882
5200	0	2.36538	9.17308	0	0.01923	9.07692	3.07692
5300	0	2.39623	9.16981	0	0.01887	9	3.0566
5400	0	2.40741	9.07407	0	0.01852	8.96296	3.11111
5500	0	2.38182	9.05455	0	0.01818	8.89091	3.09091
5600	0	2.41071	9.14286	0	0.01786	8.98214	3.08929
5700	0	2.38596	9.14035	0	0.01754	9	3.07018
5800	0	2.37931	9.10345	0	0.01724	8.87931	3.05172
5900	0	2.37288	9.10169	0	0.01695	8.84746	3.05085
6000	0	2.35	9.01667	0	0.01667	8.93333	3.06667
6100	0	2.42623	8.93443	0	0.01639	8.95082	3.08197
6200	0	2.40323	9.03226	0	0.01613	8.87097	3.06452
6300	0	2.39683	8.98413	0	0.01587	8.90476	3.07937
6400	0	2.42188	8.9375	0	0.01563	8.98438	3.09375
6500	0	2.38462	8.86154	0	0.01538	8.86154	3.09231
6600	0	2.39394	8.86364	0	0.01515	8.78788	3.06061
6700	0	2.38806	8.86567	0	0.01493	8.76119	3.07463
6800	0	2.45588	8.82353	0	0.01471	8.77941	3.05882
6900	0	2.44928	8.75362	0	0.01449	8.71014	3.04348
7000	0	2.45714	8.71429	0	0.01429	8.6	3.02857
7100	0	2.4507	8.67606	0	0.01408	8.47887	3.01408
7200	0	2.44444	8.63889	0	0.01389	8.375	2.98611
7300	0	2.46575	8.60274	0	0.0137	8.26027	2.9589
7400	0	2.44595	8.55405	0	0.01351	8.14865	2.94595
7500	0	2.44	8.54667	0	0.01333	8.04	2.96
7600	0	2.48684	8.56579	0	0.01316	7.93421	2.96053
7700	0	2.48052	8.51948	0	0.01299	7.90909	2.92208
7800	0	2.46154	8.52564	0	0.01282	7.85897	2.91026
7900	0	2.49367	8.48101	0	0.01266	7.77215	2.87342
8000	0	2.475	8.475	0	0.0125	7.7125	2.875
8100	0	2.48148	8.46914	0	0.01235	7.74074	2.83951

Continued on next page

Table B.7 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
8200	0	2.46341	8.4878	0	0.0122	7.69512	2.85366
8300	0	2.50602	8.42169	0	0.01205	7.6988	2.85542
8400	0	2.5	8.40476	0	0.0119	7.70238	2.84524
8500	0	2.48235	8.4	0	0.01176	7.63529	2.83529
8600	0	2.5	8.33721	0	0.01163	7.61628	2.82558
8700	0	2.49425	8.29885	0	0.01149	7.62069	2.81609
8800	0	2.46591	8.30682	0	0.01136	7.61364	2.84091
8900	0	2.46067	8.2809	0	0.01124	7.55056	2.82022
9000	0	2.46667	8.24444	0	0.01111	7.46667	2.83333
9100	0	2.46154	8.17582	0	0.01099	7.38462	2.8022
9200	0	2.45652	8.1413	0	0.01087	7.34783	2.80435
9300	0	2.43011	8.11828	0	0.01075	7.30108	2.78495
9400	0	2.40426	8.07447	0	0.01064	7.23404	2.81915
9500	0	2.37895	8.09474	0	0.01053	7.16842	2.82105
9600	0	2.38542	8.09375	0	0.01042	7.13542	2.8125
9700	0	2.36082	8.10309	0	0.01031	7.14433	2.78351
9800	0	2.33673	8.06122	0	0.0102	7.18367	2.76531
9900	0	2.34343	8.0303	0	0.0101	7.18182	2.75758
10000	0	2.35	8.01	0	0.01	7.2	2.77

Table B.7: Precision of MWU extraction metrics on BMC against MESH

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
100	0	0.00794	0.04761	0	0.00794	0.11903	0.08729
200	0	0.01587	0.16664	0	0.00794	0.30947	0.11903
300	0	0.02381	0.2698	0	0.00794	0.4047	0.15077
400	0	0.04761	0.4047	0	0.00794	0.49992	0.16664
500	0	0.05555	0.5396	0	0.00794	0.57927	0.19045
600	0	0.06348	0.61895	0	0.00794	0.65069	0.19838
700	0	0.08729	0.71417	0	0.00794	0.73004	0.22219
800	0	0.10316	0.76178	0	0.00794	0.84114	0.23012
900	0	0.11903	0.8332	0	0.00794	0.92049	0.25393
1000	0	0.14283	0.90462	0	0.00794	1.03158	0.28567

Continued on next page

Table B.8 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
1100	0	0.1587	0.97604	0	0.00794	1.11887	0.30154
1200	0	0.18251	1.06332	0	0.00794	1.19822	0.30947
1300	0	0.19838	1.15855	0	0.00794	1.28551	0.34915
1400	0	0.20632	1.22203	0	0.00794	1.33312	0.34915
1500	0	0.23012	1.27757	0	0.00794	1.42041	0.36502
1600	0	0.23806	1.3728	0	0.00794	1.5077	0.38089
1700	0	0.24599	1.42834	0	0.00794	1.60292	0.4047
1800	0	0.27773	1.49976	0	0.00794	1.67434	0.44437
1900	0	0.30154	1.57118	0	0.00794	1.72195	0.46024
2000	0	0.30947	1.61086	0	0.00794	1.79337	0.47611
2100	0	0.31741	1.71401	0	0.00794	1.86478	0.50786
2200	0	0.32535	1.79337	0	0.00794	1.96001	0.5396
2300	0	0.35709	1.84891	0	0.00794	2.01555	0.5396
2400	0	0.37296	1.91239	0	0.00794	2.07904	0.5634
2500	0	0.39676	1.96794	0	0.00794	2.14252	0.58721
2600	0	0.4047	2.03936	0	0.00794	2.23774	0.60308
2700	0	0.41263	2.13458	0	0.00794	2.32503	0.65863
2800	0	0.4285	2.23774	0	0.00794	2.38851	0.66656
2900	0	0.45231	2.31709	0	0.00794	2.48373	0.6983
3000	0	0.46024	2.34883	0	0.00794	2.51547	0.73004
3100	0	0.47611	2.44406	0	0.00794	2.59483	0.76178
3200	0	0.49199	2.51547	0	0.00794	2.70592	0.76178
3300	0	0.51579	2.6107	0	0.00794	2.78527	0.80146
3400	0	0.53166	2.68211	0	0.00794	2.88843	0.8094
3500	0	0.54753	2.72179	0	0.00794	2.97572	0.84907
3600	0	0.57134	2.82495	0	0.00794	3.00746	0.87288
3700	0	0.57927	2.88843	0	0.00794	3.05507	0.90462
3800	0	0.60308	2.96778	0	0.00794	3.11062	0.91255
3900	0	0.62688	3.0392	0	0.00794	3.15029	0.96017
4000	0	0.66656	3.09475	0	0.00794	3.22171	1.01571
4100	0	0.6983	3.15823	0	0.00794	3.29313	1.02365
4200	0	0.73004	3.20584	0	0.00794	3.3328	1.03158
4300	0	0.74591	3.22965	0	0.00794	3.41216	1.06332
4400	0	0.75385	3.32487	0	0.00794	3.49944	1.08713
4500	0	0.80146	3.38042	0	0.00794	3.54706	1.103

Continued on next page

Table B.8 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
4600	0	0.84114	3.4677	0	0.00794	3.57086	1.11093
4700	0	0.85701	3.48357	0	0.00794	3.6026	1.13474
4800	0	0.87288	3.54706	0	0.00794	3.65021	1.14268
4900	0	0.91255	3.63434	0	0.00794	3.67402	1.14268
5000	0	0.93636	3.69783	0	0.00794	3.67402	1.16648
5100	0	0.96017	3.7375	0	0.00794	3.67402	1.2379
5200	0	0.97604	3.78511	0	0.00794	3.74544	1.26964
5300	0	1.00778	3.85653	0	0.00794	3.78511	1.28551
5400	0	1.03158	3.88827	0	0.00794	3.84066	1.33312
5500	0	1.03952	3.95175	0	0.00794	3.88034	1.34899
5600	0	1.07126	4.06285	0	0.00794	3.99143	1.3728
5700	0	1.07919	4.13426	0	0.00794	4.07078	1.38867
5800	0	1.09506	4.18981	0	0.00794	4.08665	1.40454
5900	0	1.11093	4.26123	0	0.00794	4.1422	1.42834
6000	0	1.11887	4.29297	0	0.00794	4.25329	1.46009
6100	0	1.17442	4.32471	0	0.00794	4.33265	1.49183
6200	0	1.18235	4.44374	0	0.00794	4.36439	1.5077
6300	0	1.19822	4.49135	0	0.00794	4.45167	1.53944
6400	0	1.22996	4.53896	0	0.00794	4.56277	1.57118
6500	0	1.22996	4.5707	0	0.00794	4.5707	1.59498
6600	0	1.25377	4.64212	0	0.00794	4.60244	1.60292
6700	0	1.26964	4.71354	0	0.00794	4.65799	1.63466
6800	0	1.32519	4.76115	0	0.00794	4.73734	1.65053
6900	0	1.34106	4.79289	0	0.00794	4.76908	1.6664
7000	0	1.36486	4.8405	0	0.00794	4.77702	1.68227
7100	0	1.38073	4.88811	0	0.00794	4.77702	1.69814
7200	0	1.3966	4.93572	0	0.00794	4.78495	1.70608
7300	0	1.42834	4.98334	0	0.00794	4.78495	1.71401
7400	0	1.43628	5.02301	0	0.00794	4.78495	1.72988
7500	0	1.45215	5.08649	0	0.00794	4.78495	1.76163
7600	0	1.49976	5.16585	0	0.00794	4.78495	1.78543
7700	0	1.51563	5.20552	0	0.00794	4.83257	1.78543
7800	0	1.52357	5.27694	0	0.00794	4.86431	1.8013
7900	0	1.56324	5.31662	0	0.00794	4.87224	1.8013
8000	0	1.57118	5.3801	0	0.00794	4.89605	1.82511

Continued on next page

Table B.8 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
8100	0	1.59498	5.44358	0	0.00794	4.9754	1.82511
8200	0	1.60292	5.52293	0	0.00794	5.00714	1.85685
8300	0	1.65053	5.54674	0	0.00794	5.07062	1.88065
8400	0	1.6664	5.60229	0	0.00794	5.13411	1.89652
8500	0	1.67434	5.66577	0	0.00794	5.14998	1.91239
8600	0	1.70608	5.68957	0	0.00794	5.19759	1.92827
8700	0	1.72195	5.72925	0	0.00794	5.26107	1.94414
8800	0	1.72195	5.80067	0	0.00794	5.31662	1.98381
8900	0	1.73782	5.84828	0	0.00794	5.33249	1.99175
9000	0	1.76163	5.88795	0	0.00794	5.33249	2.02349
9100	0	1.7775	5.90382	0	0.00794	5.33249	2.02349
9200	0	1.79337	5.9435	0	0.00794	5.36423	2.04729
9300	0	1.79337	5.99111	0	0.00794	5.38803	2.05523
9400	0	1.79337	6.02285	0	0.00794	5.39597	2.10284
9500	0	1.79337	6.10221	0	0.00794	5.4039	2.12665
9600	0	1.81717	6.16569	0	0.00794	5.43565	2.14252
9700	0	1.81717	6.23711	0	0.00794	5.49913	2.14252
9800	0	1.81717	6.26885	0	0.00794	5.58641	2.15045
9900	0	1.84098	6.30852	0	0.00794	5.64196	2.16632
10000	0	1.86478	6.35613	0	0.00794	5.71338	2.19806

Table B.8: Recall of MWU extraction metrics on BMC against MESH

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
100	0	5	6	0	2	27	7
200	0	4,5	7,5	0	1,5	31	5
300	0	3,33333	8	0	1,33333	29	5
400	0	3,25	10,5	0	1	28,5	4,25
500	0	2,6	10,6	0	0,8	27	4
600	0	2,5	11,5	0	1	24,66667	3,83333
700	0	2,28571	11,14286	0	0,85714	23,28571	4
800	0	2,25	10,25	0	1	22,875	3,625
900	0	2.44444	10.11111	0	0.88889	22.33333	3.44444

Continued on next page

Table B.9 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
1000	0	2.3	9.8	0	0.8	22.4	3.3
1100	0	2.09091	9.81818	0	0.72727	21.54545	3.18182
1200	0	2	10	0	0.66667	20.33333	3.33333
1300	0	1.92308	10.15385	0	0.61538	19.76923	3.30769
1400	0	2.14286	10.5	0	0.57143	19.28571	3.07143
1500	0	2.33333	10.46667	0	0.53333	18.73333	2.93333
1600	0	2.375	10.5	0	0.5	17.9375	2.875
1700	0	2.29412	10.29412	0	0.47059	17.64706	3
1800	0	2.33333	10.33333	0	0.44444	17.44444	3.05556
1900	0	2.26316	10	0	0.42105	17.10526	3
2000	0	2.45	9.9	0	0.4	17	2.95
2100	0	2.33333	9.95238	0	0.38095	16.57143	3
2200	0	2.31818	10.09091	0	0.36364	16.59091	3.04545
2300	0	2.21739	10.08696	0	0.34783	16.30435	3
2400	0	2.125	10.08333	0	0.33333	16	2.91667
2500	0	2.16	10.16	0	0.32	15.72	2.96
2600	0	2.15385	10.23077	0	0.30769	15.57692	2.88462
2700	0	2.18519	10.07407	0	0.2963	15.51852	2.96296
2800	0	2.17857	10.03571	0	0.28571	15.25	2.85714
2900	0	2.17241	10.03448	0	0.27586	14.96552	2.86207
3000	0	2.1	10	0	0.26667	14.7	2.9
3100	0	2.06452	9.80645	0	0.25806	14.6129	2.80645
3200	0	2.09375	9.75	0	0.25	14.6875	2.8125
3300	0	2.12121	9.75758	0	0.24242	14.66667	2.72727
3400	0	2.14706	9.76471	0	0.23529	14.38235	2.73529
3500	0	2.11429	9.82857	0	0.22857	14.14286	2.74286
3600	0	2.11111	9.72222	0	0.22222	13.80556	2.66667
3700	0	2.13514	9.64865	0	0.21622	13.56757	2.7027
3800	0	2.18421	9.57895	0	0.21053	13.39474	2.63158
3900	0	2.12821	9.51282	0	0.20513	13.15385	2.61538
4000	0	2.1	9.45	0	0.2	13.025	2.6
4100	0	2.09756	9.31707	0	0.19512	12.97561	2.56098
4200	0	2.07143	9.21429	0	0.19048	12.71429	2.5
4300	0	2.06977	9.2093	0	0.18605	12.69767	2.48837
4400	0	2.02273	9.15909	0	0.18182	12.68182	2.47727

Continued on next page

Table B.9 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
4500	0	1.97778	9.06667	0	0.17778	12.64444	2.48889
4600	0	2	9.23913	0	0.17391	12.43478	2.45652
4700	0	2.02128	9.12766	0	0.17021	12.2766	2.44681
4800	0	1.97917	9.04167	0	0.16667	12.1875	2.41667
4900	0	1.93878	8.97959	0	0.16327	11.95918	2.38776
5000	0	1.92	8.82	0	0.16	11.72	2.34
5100	0	1.94118	8.76471	0	0.15686	11.5098	2.35294
5200	0	1.94231	8.69231	0	0.15385	11.46154	2.38462
5300	0	1.92453	8.73585	0	0.15094	11.43396	2.35849
5400	0	1.96296	8.68519	0	0.14815	11.40741	2.35185
5500	0	1.96364	8.69091	0	0.14545	11.38182	2.36364
5600	0	1.94643	8.69643	0	0.14286	11.30357	2.33929
5700	0	1.91228	8.7193	0	0.14035	11.2807	2.31579
5800	0	1.89655	8.68966	0	0.13793	11.15517	2.27586
5900	0	1.88136	8.69492	0	0.13559	11.01695	2.27119
6000	0	1.91667	8.68333	0	0.13333	10.93333	2.28333
6100	0	1.90164	8.60656	0	0.13115	10.88525	2.29508
6200	0	1.90323	8.56452	0	0.12903	10.77419	2.25806
6300	0	1.90476	8.47619	0	0.12698	10.73016	2.25397
6400	0	1.92188	8.45313	0	0.125	10.6875	2.28125
6500	0	1.90769	8.47692	0	0.12308	10.53846	2.26154
6600	0	1.92424	8.5	0	0.12121	10.4697	2.30303
6700	0	1.92537	8.41791	0	0.1194	10.37313	2.28358
6800	0	1.94118	8.39706	0	0.11765	10.32353	2.27941
6900	0	1.97101	8.37681	0	0.11594	10.27536	2.26087
7000	0	1.98571	8.32857	0	0.11429	10.15714	2.25714
7100	0	1.98592	8.29577	0	0.11268	10.02817	2.23944
7200	0	1.95833	8.22222	0	0.11111	9.94444	2.26389
7300	0	1.94521	8.17808	0	0.10959	9.82192	2.24658
7400	0	1.91892	8.16216	0	0.10811	9.7027	2.22973
7500	0	1.93333	8.16	0	0.10667	9.6	2.2
7600	0	1.94737	8.13158	0	0.10526	9.47368	2.21053
7700	0	1.94805	8.11688	0	0.1039	9.50649	2.22078
7800	0	1.96154	8.14103	0	0.10256	9.46154	2.25641
7900	0	1.96203	8.13924	0	0.10127	9.37975	2.26582

Continued on next page

Table B.9 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
8000	0	1.9625	8.0875	0	0.1	9.3	2.275
8100	0	1.93827	8.08642	0	0.09877	9.30864	2.24691
8200	0	1.91463	8.09756	0	0.09756	9.30488	2.2561
8300	0	1.93976	8.0241	0	0.09639	9.28916	2.24096
8400	0	1.92857	7.96429	0	0.09524	9.30952	2.22619
8500	0	1.90588	7.94118	0	0.09412	9.24706	2.24706
8600	0	1.88372	7.94186	0	0.09302	9.22093	2.24419
8700	0	1.88506	7.86207	0	0.09195	9.17241	2.24138
8800	0	1.875	7.85227	0	0.09091	9.125	2.25
8900	0	1.88764	7.82022	0	0.08989	9.08989	2.25843
9000	0	1.87778	7.82222	0	0.08889	9.03333	2.24444
9100	0	1.85714	7.82418	0	0.08791	8.96703	2.21978
9200	0	1.8587	7.78261	0	0.08696	8.91304	2.21739
9300	0	1.84946	7.80645	0	0.08602	8.87097	2.26882
9400	0	1.87234	7.78723	0	0.08511	8.81915	2.25532
9500	0	1.86316	7.75789	0	0.08421	8.81053	2.26316
9600	0	1.84375	7.75	0	0.08333	8.77083	2.26042
9700	0	1.86598	7.73196	0	0.08247	8.76289	2.24742
9800	0	1.86735	7.72449	0	0.08163	8.73469	2.22449
9900	0	1.86869	7.68687	0	0.08081	8.66667	2.23232
10000	0	1.86	7.67	0	0.08	8.62	2.24

Table B.9: Precision of MWU extraction metrics on BMC against SNOMED-CT

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
100	0	0.03623	0.04348	0	0.01449	0.19565	0.05072
200	0	0.06522	0.1087	0	0.02174	0.44928	0.07246
300	0	0.07246	0.17391	0	0.02899	0.63043	0.1087
400	0	0.0942	0.30435	0	0.02899	0.82609	0.12319
500	0	0.0942	0.38406	0	0.02899	0.97826	0.14493
600	0	0.1087	0.5	0	0.04348	1.07246	0.16667
700	0	0.11594	0.56522	0	0.04348	1.18116	0.2029
800	0	0.13043	0.5942	0	0.05797	1.32609	0.21014

Continued on next page

Table B.10 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
900	0	0.15942	0.65942	0	0.05797	1.45652	0.22464
1000	0	0.16667	0.71014	0	0.05797	1.62319	0.23913
1100	0	0.16667	0.78261	0	0.05797	1.71739	0.25362
1200	0	0.17391	0.86957	0	0.05797	1.76812	0.28986
1300	0	0.18116	0.95652	0	0.05797	1.86232	0.31159
1400	0	0.21739	1.06522	0	0.05797	1.95652	0.31159
1500	0	0.25362	1.13768	0	0.05797	2.03623	0.31884
1600	0	0.27536	1.21739	0	0.05797	2.07971	0.33333
1700	0	0.28261	1.26812	0	0.05797	2.17391	0.36957
1800	0	0.30435	1.34783	0	0.05797	2.27536	0.39855
1900	0	0.31159	1.37681	0	0.05797	2.35507	0.41304
2000	0	0.35507	1.43478	0	0.05797	2.46377	0.42754
2100	0	0.35507	1.51449	0	0.05797	2.52174	0.45652
2200	0	0.36957	1.6087	0	0.05797	2.64493	0.48551
2300	0	0.36957	1.68116	0	0.05797	2.71739	0.5
2400	0	0.36957	1.75362	0	0.05797	2.78261	0.50725
2500	0	0.3913	1.84058	0	0.05797	2.84783	0.53623
2600	0	0.4058	1.92754	0	0.05797	2.93478	0.54348
2700	0	0.42754	1.97101	0	0.05797	3.03623	0.57971
2800	0	0.44203	2.03623	0	0.05797	3.0942	0.57971
2900	0	0.45652	2.1087	0	0.05797	3.14493	0.60145
3000	0	0.45652	2.17391	0	0.05797	3.19565	0.63043
3100	0	0.46377	2.2029	0	0.05797	3.28261	0.63043
3200	0	0.48551	2.26087	0	0.05797	3.4058	0.65217
3300	0	0.50725	2.33333	0	0.05797	3.50725	0.65217
3400	0	0.52899	2.4058	0	0.05797	3.54348	0.67391
3500	0	0.53623	2.49275	0	0.05797	3.58696	0.69565
3600	0	0.55072	2.53623	0	0.05797	3.60145	0.69565
3700	0	0.57246	2.58696	0	0.05797	3.63768	0.72464
3800	0	0.60145	2.63768	0	0.05797	3.68841	0.72464
3900	0	0.60145	2.68841	0	0.05797	3.71739	0.73913
4000	0	0.6087	2.73913	0	0.05797	3.77536	0.75362
4100	0	0.62319	2.76812	0	0.05797	3.85507	0.76087
4200	0	0.63043	2.80435	0	0.05797	3.86957	0.76087
4300	0	0.64493	2.86957	0	0.05797	3.95652	0.77536

Continued on next page

Table B.10 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
4400	0	0.64493	2.92029	0	0.05797	4.04348	0.78986
4500	0	0.64493	2.95652	0	0.05797	4.12319	0.81159
4600	0	0.66667	3.07971	0	0.05797	4.14493	0.81884
4700	0	0.68841	3.1087	0	0.05797	4.18116	0.83333
4800	0	0.68841	3.14493	0	0.05797	4.23913	0.84058
4900	0	0.68841	3.18841	0	0.05797	4.24638	0.84783
5000	0	0.69565	3.19565	0	0.05797	4.24638	0.84783
5100	0	0.71739	3.23913	0	0.05797	4.25362	0.86957
5200	0	0.73188	3.27536	0	0.05797	4.31884	0.89855
5300	0	0.73913	3.35507	0	0.05797	4.3913	0.9058
5400	0	0.76812	3.39855	0	0.05797	4.46377	0.92029
5500	0	0.78261	3.46377	0	0.05797	4.53623	0.94203
5600	0	0.78986	3.52899	0	0.05797	4.58696	0.94928
5700	0	0.78986	3.60145	0	0.05797	4.65942	0.95652
5800	0	0.7971	3.65217	0	0.05797	4.68841	0.95652
5900	0	0.80435	3.71739	0	0.05797	4.71014	0.97101
6000	0	0.83333	3.77536	0	0.05797	4.75362	0.99275
6100	0	0.84058	3.80435	0	0.05797	4.81159	1.01449
6200	0	0.85507	3.84783	0	0.05797	4.84058	1.01449
6300	0	0.86957	3.86957	0	0.05797	4.89855	1.02899
6400	0	0.8913	3.92029	0	0.05797	4.95652	1.05797
6500	0	0.89855	3.99275	0	0.05797	4.96377	1.06522
6600	0	0.92029	4.06522	0	0.05797	5.00725	1.10145
6700	0	0.93478	4.08696	0	0.05797	5.03623	1.1087
6800	0	0.95652	4.13768	0	0.05797	5.08696	1.12319
6900	0	0.98551	4.18841	0	0.05797	5.13768	1.13043
7000	0	1.00725	4.22464	0	0.05797	5.15217	1.14493
7100	0	1.02174	4.26812	0	0.05797	5.15942	1.15217
7200	0	1.02174	4.28986	0	0.05797	5.18841	1.18116
7300	0	1.02899	4.32609	0	0.05797	5.19565	1.18841
7400	0	1.02899	4.37681	0	0.05797	5.2029	1.19565
7500	0	1.05072	4.43478	0	0.05797	5.21739	1.19565
7600	0	1.07246	4.47826	0	0.05797	5.21739	1.21739
7700	0	1.08696	4.52899	0	0.05797	5.30435	1.23913
7800	0	1.1087	4.60145	0	0.05797	5.34783	1.27536

Continued on next page

Table B.10 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
7900	0	1.12319	4.65942	0	0.05797	5.36957	1.2971
8000	0	1.13768	4.68841	0	0.05797	5.3913	1.31884
8100	0	1.13768	4.74638	0	0.05797	5.46377	1.31884
8200	0	1.13768	4.81159	0	0.05797	5.52899	1.34058
8300	0	1.16667	4.82609	0	0.05797	5.58696	1.34783
8400	0	1.17391	4.84783	0	0.05797	5.66667	1.35507
8500	0	1.17391	4.8913	0	0.05797	5.69565	1.38406
8600	0	1.17391	4.94928	0	0.05797	5.74638	1.39855
8700	0	1.18841	4.95652	0	0.05797	5.78261	1.41304
8800	0	1.19565	5.00725	0	0.05797	5.81884	1.43478
8900	0	1.21739	5.04348	0	0.05797	5.86232	1.45652
9000	0	1.22464	5.10145	0	0.05797	5.8913	1.46377
9100	0	1.22464	5.15942	0	0.05797	5.91304	1.46377
9200	0	1.23913	5.18841	0	0.05797	5.94203	1.47826
9300	0	1.24638	5.26087	0	0.05797	5.97826	1.52899
9400	0	1.27536	5.30435	0	0.05797	6.00725	1.53623
9500	0	1.28261	5.34058	0	0.05797	6.06522	1.55797
9600	0	1.28261	5.3913	0	0.05797	6.10145	1.57246
9700	0	1.31159	5.43478	0	0.05797	6.15942	1.57971
9800	0	1.32609	5.48551	0	0.05797	6.2029	1.57971
9900	0	1.34058	5.51449	0	0.05797	6.21739	1.60145
10000	0	1.34783	5.55797	0	0.05797	6.24638	1.62319

Table B.10: Recall of MWU extraction metrics on BMC against SNOMED-CT

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
100	0	4	12	0	6	58	19
200	0	3	20.5	0	3.5	54.5	12.5
300	0	3	22	0	2.66667	53.66667	11
400	0	4.5	23.25	0	2.75	51.5	9.5
500	0	4.2	23.8	0	2.6	48.6	8.6
600	0	4	23.83333	0	3	44.66667	7.66667
700	0	3.85714	23.42857	0	2.57143	42.28571	7

Continued on next page

Table B.11 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
800	0	3.625	22.125	0	2.75	42	6.375
900	0	3.77778	21.77778	0	2.44444	40.33333	6.33333
1000	0	3.7	21.6	0	2.2	41	6.4
1100	0	3.63636	21.54545	0	2	40.27273	5.90909
1200	0	3.5	21.5	0	1.83333	39.58333	5.91667
1300	0	3.61538	21.92308	0	1.69231	39.15385	6.15385
1400	0	3.64286	22.21429	0	1.57143	38.71429	5.92857
1500	0	3.73333	22.13333	0.06667	1.46667	38	5.86667
1600	0	3.875	22.125	0.0625	1.375	37.5625	5.75
1700	0	3.70588	22.17647	0.05882	1.35294	36.88235	5.88235
1800	0	4.05556	22.44444	0.05556	1.27778	35.83333	6
1900	0	3.94737	22.15789	0.05263	1.21053	35.26316	6
2000	0	4.05	22	0.05	1.15	34.75	6.05
2100	0	4	21.85714	0.04762	1.09524	34.38095	6.04762
2200	0	3.95455	21.68182	0.04545	1.04545	34.09091	6.09091
2300	0	4.08696	21.34783	0.04348	1	33.6087	5.95652
2400	0	4.125	21.16667	0.04167	0.95833	33.04167	5.83333
2500	0	4.08	21.12	0.04	0.92	32.68	5.88
2600	0	4.07692	20.88462	0.03846	0.88462	32.42308	5.73077
2700	0	4.03704	20.88889	0.03704	0.85185	32.07407	5.85185
2800	0	4.07143	21	0.03571	0.82143	31.78571	5.78571
2900	0	4.06897	21.06897	0.03448	0.7931	31.58621	5.75862
3000	0	4	21.06667	0.03333	0.76667	31	5.83333
3100	0	4	20.87097	0.03226	0.74194	30.64516	5.87097
3200	0	4.09375	20.875	0.03125	0.71875	30.625	5.78125
3300	0	4.15152	21.0303	0.0303	0.69697	30.33333	5.72727
3400	0	4.14706	20.88235	0.02941	0.67647	30.20588	5.67647
3500	0	4.08571	20.71429	0.02857	0.65714	29.82857	5.74286
3600	0	4.13889	20.52778	0.02778	0.63889	29.47222	5.69444
3700	0	4.13514	20.54054	0.02703	0.62162	29.05405	5.64865
3800	0	4.18421	20.52632	0.02632	0.60526	28.65789	5.57895
3900	0	4.17949	20.4359	0.02564	0.58974	28.30769	5.53846
4000	0.025	4.225	20.3	0.025	0.575	28.275	5.5
4100	0.02439	4.34146	20.19512	0.02439	0.56098	28.09756	5.46341
4200	0.02381	4.33333	19.95238	0.02381	0.54762	27.85714	5.38095

Continued on next page

Table B.11 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
4300	0.02326	4.30233	19.90698	0.02326	0.53488	27.69767	5.39535
4400	0.02273	4.25	19.79545	0.02273	0.52273	27.56818	5.38636
4500	0.02222	4.33333	19.77778	0.02222	0.51111	27.48889	5.42222
4600	0.02174	4.36957	19.91304	0.02174	0.5	27.32609	5.3913
4700	0.02128	4.38298	19.76596	0.02128	0.48936	27.06383	5.42553
4800	0.02083	4.39583	19.64583	0.02083	0.5	26.89583	5.35417
4900	0.02041	4.36735	19.61224	0.02041	0.4898	26.5102	5.26531
5000	0.02	4.32	19.46	0.02	0.48	26.02	5.3
5100	0.01961	4.29412	19.39216	0.01961	0.47059	25.56863	5.35294
5200	0.01923	4.25	19.26923	0.01923	0.46154	25.44231	5.30769
5300	0.01887	4.26415	19.41509	0.01887	0.45283	25.26415	5.26415
5400	0.01852	4.2963	19.31481	0.01852	0.44444	25.14815	5.2963
5500	0.01818	4.27273	19.36364	0.01818	0.43636	25	5.25455
5600	0.01786	4.28571	19.375	0.01786	0.42857	24.85714	5.23214
5700	0.01754	4.22807	19.33333	0.01754	0.42105	24.75439	5.19298
5800	0.01724	4.2069	19.24138	0.01724	0.41379	24.46552	5.15517
5900	0.01695	4.18644	19.22034	0.01695	0.40678	24.27119	5.15254
6000	0.01667	4.18333	19.21667	0.01667	0.4	24.18333	5.16667
6100	0.01639	4.21311	19.11475	0.01639	0.39344	24.18033	5.18033
6200	0.01613	4.16129	19.19355	0.01613	0.3871	24.01613	5.16129
6300	0.01587	4.20635	19.14286	0.01587	0.38095	23.88889	5.15873
6400	0.01563	4.21875	19.14063	0.01563	0.375	23.875	5.20313
6500	0.01538	4.18462	19.10769	0.01538	0.36923	23.6	5.18462
6600	0.01515	4.18182	19.06061	0.01515	0.36364	23.45455	5.16667
6700	0.01493	4.19403	18.97015	0.01493	0.35821	23.40299	5.19403
6800	0.01471	4.20588	18.91176	0.01471	0.35294	23.41176	5.17647
6900	0.01449	4.17391	18.82609	0.01449	0.34783	23.24638	5.2029
7000	0.02857	4.17143	18.8	0.01429	0.37143	22.97143	5.2
7100	0.02817	4.15493	18.73239	0.01408	0.38028	22.69014	5.1831
7200	0.02778	4.13889	18.66667	0.01389	0.375	22.48611	5.13889
7300	0.0274	4.12329	18.61644	0.0137	0.36986	22.19178	5.08219
7400	0.02703	4.10811	18.59459	0.01351	0.36486	21.94595	5.05405
7500	0.02667	4.09333	18.54667	0.01333	0.36	21.68	5
7600	0.02632	4.13158	18.48684	0.01316	0.35526	21.39474	4.96053
7700	0.02597	4.1039	18.44156	0.01299	0.35065	21.37662	4.94805

Continued on next page

Table B.11 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
7800	0.02564	4.11538	18.4359	0.01282	0.34615	21.30769	4.96154
7900	0.03797	4.12658	18.40506	0.01266	0.34177	21.16456	4.93671
8000	0.0375	4.1125	18.3125	0.0125	0.3375	21.1	4.925
8100	0.03704	4.11111	18.30864	0.01235	0.33333	21.06173	4.8642
8200	0.03659	4.08537	18.34146	0.02439	0.32927	21.0122	4.86585
8300	0.03614	4.12048	18.24096	0.0241	0.3253	20.95181	4.84337
8400	0.03571	4.11905	18.2381	0.02381	0.32143	20.88095	4.83333
8500	0.03529	4.08235	18.21176	0.02353	0.32941	20.74118	4.84706
8600	0.03488	4.06977	18.15116	0.02326	0.32558	20.61628	4.82558
8700	0.03448	4.09195	18.08046	0.02299	0.32184	20.56322	4.8046
8800	0.03409	4.06818	18.02273	0.02273	0.31818	20.46591	4.82955
8900	0.03371	4.04494	17.92135	0.05618	0.31461	20.39326	4.79775
9000	0.03333	4.05556	17.88889	0.07778	0.31111	20.25556	4.78889
9100	0.03297	4.05495	17.84615	0.07692	0.30769	20.08791	4.76923
9200	0.03261	4.07609	17.81522	0.1087	0.30435	20.02174	4.77174
9300	0.03226	4.07527	17.83871	0.10753	0.30108	19.96774	4.8172
9400	0.03191	4.06383	17.75532	0.10638	0.29787	19.90426	4.82979
9500	0.04211	4.08421	17.73684	0.10526	0.29474	19.82105	4.85263
9600	0.04167	4.0625	17.67708	0.10417	0.3125	19.6875	4.85417
9700	0.04124	4.06186	17.68041	0.10309	0.30928	19.64948	4.80412
9800	0.04082	4.04082	17.61224	0.10204	0.30612	19.63265	4.77551
9900	0.0404	4.0303	17.56566	0.10101	0.31313	19.59596	4.76768
10000	0.04	4.02	17.51	0.1	0.31	19.53	4.79

Table B.11: Precision of MWU extraction metrics on BMC against UMLS

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
100	0	0.01418	0.04255	0	0.02127	0.20564	0.06737
200	0	0.02127	0.14537	0	0.02482	0.38647	0.08864
300	0	0.03191	0.23401	0	0.02836	0.57084	0.117
400	0	0.06382	0.32974	0	0.039	0.73039	0.13473
500	0	0.07446	0.42193	0	0.04609	0.86158	0.15246
600	0	0.08509	0.50702	0	0.06382	0.95022	0.1631

Continued on next page

Table B.12 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
700	0	0.09573	0.58148	0	0.06382	1.0495	0.17373
800	0	0.10282	0.62757	0	0.078	1.19132	0.18083
900	0	0.12055	0.69494	0	0.078	1.28705	0.2021
1000	0	0.13119	0.76585	0	0.078	1.45369	0.22692
1100	0	0.14182	0.84031	0	0.078	1.5707	0.23046
1200	0	0.14892	0.91476	0	0.078	1.68416	0.25174
1300	0	0.16664	1.01049	0	0.078	1.80471	0.28365
1400	0	0.18083	1.10268	0	0.078	1.92171	0.29428
1500	0	0.19855	1.17714	0.00355	0.078	2.02099	0.31201
1600	0	0.21983	1.25514	0.00355	0.078	2.1309	0.32619
1700	0	0.22337	1.33669	0.00355	0.08155	2.22309	0.35456
1800	0	0.25883	1.43242	0.00355	0.08155	2.28691	0.38292
1900	0	0.26592	1.4927	0.00355	0.08155	2.37555	0.4042
2000	0	0.28719	1.56006	0.00355	0.08155	2.46419	0.42902
2100	0	0.29783	1.62743	0.00355	0.08155	2.55992	0.45029
2200	0	0.30847	1.69125	0.00355	0.08155	2.6592	0.47511
2300	0	0.33329	1.74089	0.00355	0.08155	2.74075	0.48575
2400	0	0.35101	1.80116	0.00355	0.08155	2.81166	0.49638
2500	0	0.36165	1.87207	0.00355	0.08155	2.89675	0.5212
2600	0	0.37583	1.92526	0.00355	0.08155	2.98894	0.52829
2700	0	0.38647	1.99972	0.00355	0.08155	3.07049	0.5602
2800	0	0.4042	2.08481	0.00355	0.08155	3.15558	0.57439
2900	0	0.41838	2.16636	0.00355	0.08155	3.24777	0.59211
3000	0	0.42547	2.24082	0.00355	0.08155	3.2974	0.62048
3100	0	0.43965	2.294	0.00355	0.08155	3.36832	0.6453
3200	0	0.46447	2.36846	0.00355	0.08155	3.47468	0.65594
3300	0	0.48575	2.46064	0.00355	0.08155	3.54914	0.67012
3400	0	0.49993	2.51737	0.00355	0.08155	3.64133	0.6843
3500	0	0.50702	2.57056	0.00355	0.08155	3.7016	0.71266
3600	0	0.52829	2.6202	0.00355	0.08155	3.76188	0.72685
3700	0	0.54248	2.69465	0.00355	0.08155	3.81152	0.74103
3800	0	0.56375	2.76557	0.00355	0.08155	3.86115	0.75167
3900	0	0.57793	2.82584	0.00355	0.08155	3.91434	0.76585
4000	0.00355	0.59921	2.87902	0.00355	0.08155	4.01007	0.78003
4100	0.00355	0.63112	2.93575	0.00355	0.08155	4.08453	0.79421

Continued on next page

Table B.12 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
4200	0.00355	0.6453	2.97121	0.00355	0.08155	4.14835	0.8013
4300	0.00355	0.65594	3.03503	0.00355	0.08155	4.22281	0.82258
4400	0.00355	0.66303	3.08821	0.00355	0.08155	4.30081	0.84031
4500	0.00355	0.69139	3.15558	0.00355	0.08155	4.3859	0.86513
4600	0.00355	0.71266	3.24777	0.00355	0.08155	4.45681	0.87931
4700	0.00355	0.73039	3.29386	0.00355	0.08155	4.51	0.90413
4800	0.00355	0.74812	3.3435	0.00355	0.08509	4.57736	0.91122
4900	0.00355	0.75876	3.40732	0.00355	0.08509	4.60573	0.91476
5000	0.00355	0.76585	3.44987	0.00355	0.08509	4.61282	0.93958
5100	0.00355	0.77649	3.50659	0.00355	0.08509	4.62346	0.96795
5200	0.00355	0.78358	3.55269	0.00355	0.08509	4.69082	0.97858
5300	0.00355	0.8013	3.64842	0.00355	0.08509	4.74755	0.98922
5400	0.00355	0.82258	3.69806	0.00355	0.08509	4.81492	1.01404
5500	0.00355	0.83322	3.77606	0.00355	0.08509	4.8752	1.02468
5600	0.00355	0.85094	3.84697	0.00355	0.08509	4.93547	1.03886
5700	0.00355	0.85449	3.90725	0.00355	0.08509	5.00284	1.0495
5800	0.00355	0.86513	3.95689	0.00355	0.08509	5.0312	1.06013
5900	0.00355	0.87576	4.02071	0.00355	0.08509	5.07729	1.07786
6000	0.00355	0.88994	4.08807	0.00355	0.08509	5.14466	1.09913
6100	0.00355	0.91122	4.13417	0.00355	0.08509	5.22975	1.12041
6200	0.00355	0.91476	4.21926	0.00355	0.08509	5.27939	1.13459
6300	0.00355	0.93958	4.27599	0.00355	0.08509	5.33612	1.15232
6400	0.00355	0.95731	4.34336	0.00355	0.08509	5.41767	1.18068
6500	0.00355	0.9644	4.40363	0.00355	0.08509	5.43894	1.19487
6600	0.00355	0.97858	4.46036	0.00355	0.08509	5.48858	1.20905
6700	0.00355	0.99631	4.50645	0.00355	0.08509	5.5595	1.23387
6800	0.00355	1.01404	4.55964	0.00355	0.08509	5.64459	1.24805
6900	0.00355	1.02113	4.60573	0.00355	0.08509	5.68714	1.27287
7000	0.00709	1.03531	4.666	0.00355	0.09219	5.70132	1.2906
7100	0.00709	1.04595	4.71564	0.00355	0.09573	5.71196	1.30478
7200	0.00709	1.05659	4.76528	0.00355	0.09573	5.74032	1.31187
7300	0.00709	1.06722	4.81847	0.00355	0.09573	5.74387	1.31542
7400	0.00709	1.07786	4.87874	0.00355	0.09573	5.75805	1.32605
7500	0.00709	1.0885	4.93192	0.00355	0.09573	5.76514	1.3296
7600	0.00709	1.11332	4.98156	0.00355	0.09573	5.76514	1.33669

Continued on next page

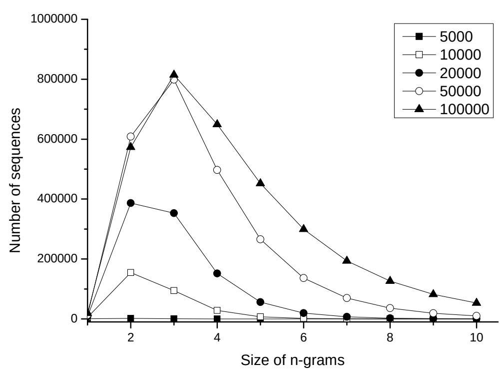
Table B.12 – continued from previous page

n	DICE	FREQ	ME	PMI	SCP	SRE	TFIDF
7700	0.00709	1.12041	5.03475	0.00355	0.09573	5.83605	1.35087
7800	0.00709	1.13814	5.09857	0.00355	0.09573	5.89278	1.37215
7900	0.01064	1.15586	5.1553	0.00355	0.09573	5.92824	1.38278
8000	0.01064	1.1665	5.1943	0.00355	0.09573	5.98497	1.39696
8100	0.01064	1.18068	5.25812	0.00355	0.09573	6.04879	1.39696
8200	0.01064	1.18777	5.33258	0.00709	0.09573	6.10906	1.41469
8300	0.01064	1.21259	5.36803	0.00709	0.09573	6.16579	1.42533
8400	0.01064	1.22678	5.43185	0.00709	0.09573	6.21898	1.43951
8500	0.01064	1.23032	5.48858	0.00709	0.09928	6.25089	1.46079
8600	0.01064	1.24096	5.53468	0.00709	0.09928	6.28634	1.47142
8700	0.01064	1.26223	5.57722	0.00709	0.09928	6.34307	1.48206
8800	0.01064	1.26932	5.62332	0.00709	0.09928	6.38562	1.50688
8900	0.01064	1.27641	5.65523	0.01773	0.09928	6.43526	1.51397
9000	0.01064	1.29414	5.70841	0.02482	0.09928	6.46362	1.52815
9100	0.01064	1.30833	5.75805	0.02482	0.09928	6.48135	1.53879
9200	0.01064	1.3296	5.81123	0.03546	0.09928	6.53099	1.55652
9300	0.01064	1.34378	5.88214	0.03546	0.09928	6.58417	1.58843
9400	0.01064	1.35442	5.9176	0.03546	0.09928	6.63381	1.6097
9500	0.01418	1.37569	5.97433	0.03546	0.09928	6.67636	1.63452
9600	0.01418	1.38278	6.01688	0.03546	0.10637	6.70118	1.65225
9700	0.01418	1.39696	6.0807	0.03546	0.10637	6.75791	1.65225
9800	0.01418	1.40406	6.1197	0.03546	0.10637	6.82173	1.65934
9900	0.01418	1.41469	6.16579	0.03546	0.10991	6.87846	1.67352
10000	0.01418	1.42533	6.20834	0.03546	0.10991	6.92455	1.69834

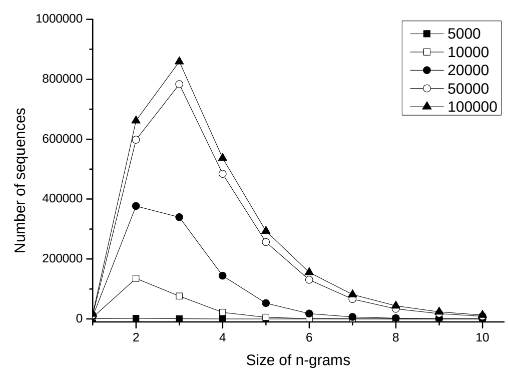
Table B.12: Recall of MWU extraction metrics on BMC against UMLS

Appendix C

Lexicon-based extraction using directed graphs.



(a) Distribution of sequences using weighted SIGNUM results



(b) Distribution of sequences using unweighted SIGNUM results

Figure C.1: Distribution of sequences using SIGNUM results on TREC Corpus

Size	5,000		10,000		20,000		50,000		100,000	
	U	W	U	W	U	W	U	W	U	W
1	770	769	5082	5252	8456	8510	12667	12852	16526	18674
2	1563	1667	135459	154620	376675	386534	597834	608397	661917	573242
3	338	360	76425	94233	339277	352885	783129	798733	859206	814403
4	71	82	21718	28470	144243	151823	484107	496780	536549	649453
5	11	9	5311	7332	52293	55820	256471	265663	293704	452483
6	4	4	1344	1976	17993	19456	130535	136419	155090	299258
7	0	0	368	550	6342	6921	66202	69829	81248	193700
8	0	0	105	149	2373	2611	33869	36005	43800	126142
9	0	0	39	59	968	1074	17762	19034	23692	82297
10	0	0	13	18	425	462	9161	9835	12741	53283

Table C.1: Distribution of words sequences extracted using sequence extraction and the results of SIGNUM on the TREC-9 corpus. *The columns marked with U display the results obtained using unweighted graph of the given size. The columns marked with W display the results obtained using weighted graphs.*

Bibliography

Adamson, G. and J. Boreham (1974). The use of an association measure based on character structure to identify semantically related words and document titles. *Information Storage and Retrieval 10*, 253–260.

Al-Shammari, E. and J. Lin (2008). A novel arabic lemmatization algorithm. In *Proceedings of the 2nd workshop on Analytics for noisy unstructured text data*, New York, NY, USA, pp. 113–118. ACM.

Al-Sughaiyer, I. and I. Al-Kharashi (2004). Arabic morphological analysis techniques: A comprehensive survey. *Journal of the American Society for Information Science and Technology 55*(3), 189–213.

Ananiadou, S. and J. Mcnaught (2005). *Text Mining for Biology and Biomedecine.* Norwood, MA, USA.

Andersen, R., F. Chung, and K. Lang (2006). Local graph partitioning using pagerank vectors. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, Washington, DC, USA, pp. 475–486. IEEE Computer Society.

Aussenac-Gilles, N., B. Biebow, and S. Szulman (2000). Revisiting ontology design: A methodology based on corpus analysis. In *Proceedings of the EKAW '00*, London, UK, pp. 172–188. Springer.

Aussenac-Gilles, N. and P. Seguela (2000). Les relations sémantiques: du linguistique au formel. *Cahiers de grammaire 25*, 175–198.

Baeza-Yates, R. and B. Ribeiro-Neto (1999). *Modern Information Retrieval.* ACM Press / Addison-Wesley.

Barker, K., V. Chaudhri, S. Chaw, P. Clark, J. Fan, D. Israel, S. Mishra, B. Porter, P. Romero, D. Tecuci, and P. Yeh (2004). A question-answering system for AP chemistry: Assessing KR&R technologies. In *Proceedings of the Ninth International Conference on Principles of Knowledge Representation and Reasoning*, pp. 488–497.

Barwise, J. and J. Perry (1983). *Situations and Attitudes*. Cambridge, MA: MIT Press.

Basile, P., D. Gendarmi, F. Lanubile, and G. Semeraro (2007). Recommending smart tags in a social bookmarking system. In *Proceeding of SemNet 2007*, pp. 22–29.

Benamara, F. and P. Dizier (2003). Webcoop: a cooperative question-answering system on the web. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics*, Morristown, NJ, USA, pp. 63–66. Association for Computational Linguistics.

Berkhin, P. (2002). Survey Of Clustering Data Mining Techniques. Technical report, Accrue Software.

Bezdek, J. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms*. Norwell, MA, USA: Kluwer Academic Publishers.

Biemann, C. (2005). Ontology learning from text: A survey of methods. *LDV Forum 20*(2), 75–93.

Biemann, C. (2007). *Unsupervised and Knowledge-Free Natural Language Processing in the Structure Discovery Paradigm*. Ph. D. thesis, University of Leipzig, Leipzig, Germany.

Biemann, C., S. Bordag, G. Heyer, U. Quasthoff, and C. Wolff (2004). Language-independent methods for compiling monolingual lexical data. In *Proceedings of CicLING 2004*, Seoul, Korea, pp. 215–228. Springer Verlag.

Bisson, G., C. Nedellec, and D. Caamero (2000). Designing clustering methods for ontology building - the Mo'K workbench. In *ECAI Workshop on Ontology Learning*, Volume 31 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Bodenreider, O., T. C. Rindflesch, and A. Burgun (2002). Unsupervised, corpus-based method for extending a biomedical terminology. In *Proceedings of the ACL '02 workshop on Natural language processing in the biomedical domain*, Morristown, NJ, USA, pp. 53–60. Association for Computational Linguistics.

Bookstein, A. and A. Swanson (1974). Probabilistic models for automatic indexing. *Journal of the American Society for Information Science 25*(5), 312–318.

Booth, J., G. Casella, and J. Hobert (2007). Clustering using objective functions and stochastic search. *Journal Of The Royal Statistical Society Series B 70*(1), 119–139.

Bordag, S. (2007). *Elements of Knowledge-free and Unsupervised Lexical Acquisition*. Ph. D. thesis, University of Leipzig, Leipzig, Germany.

Botafogo, R., E. Rivlin, and B. Shneiderman (1992). Structural analysis of hypertexts: identifying hierarchies and useful metrics. *ACM Transactions on Information Systems 10*(2), 142–180.

Botelho, F., R. Pagh, and N. Ziviani (2007). Simple and space-efficient minimal perfect hash functions. In *10th Workshop on Algorithms and Data Structures*, pp. 139–150.

Botelho, F. and N. Ziviani (2007). External perfect hashing for very large key sets. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM)*, pp. 653–662.

Bourigault, D., I. Gonzalez-Mullier, and C. Gros (1996). Lexter, a natural language tool for terminology extraction. In *Proceedings of the 7th EURALEX*, Götheborg, Sweden, pp. 771–779.

Boutin, F. and M. Hascoet (2004). Cluster validity indices for graph partitioning. In *Proceedings of the Eighth International Conference on Information Visualisation*, Washington, DC, USA, pp. 376–381. IEEE Computer Society.

Brill, E. (1995). Unsupervised learning of disambiguation rules for part of speech tagging. In *Proceedings of the Third Workshop on Very Large Corpora*, Somerset, New Jersey, pp. 1–13. Association for Computational Linguistics.

Buitelaar, P., D. Olejnik, and M. Sintek (2004). A protégé plug-in for ontology extraction from text based on linguistic analysis. In *Proceedings of the 1st European Semantic Web Symposium*, pp. 31–44.

Camon, E., D. Barrell, V. Lee, E. Dimmer, and R. Apweiler (2003). The gene ontology annotation (GOA) database - an integrated resource of go annotations to the uniprot knowledgebase. *In Silico Biology 4*(0002).

Can, A. and N. Baykal (2007). MedicoPort: A medical search engine for all. *Computer methods and programs in biomedicine 86*(1), 73–86.

Caraballo, S. (1999). Automatic construction of a hypernym-labeled noun hierarchy from text. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, Morristown, NJ, USA, pp. 120–126. Association for Computational Linguistics.

Cha, S., S. Yoon, and C. Tappert (2005). On binary similarity measures for handwritten character recognition. In *Proceedings of the Eighth International Conference on Document Analysis and Recognition*, Washington, DC, USA, pp. 4–8. IEEE Computer Society.

Charras, C. and T. Lecroq (2004). *Handbook of Exact String Matching Algorithms.* King's College Publications.

Chen, A., J. He, L. Xu, F. Gey, and J. Meggs (1997). Chinese text retrieval without using a dictionary. In *Proceedings of SIGIR '97*, New York, NY, USA, pp. 42–49. ACM Press.

Cheng, K., G. Young, and K. Wong (1999). A study on word-based and integral-bit chinese text compression algorithms. *Journal of the American Society on Information Science 50*(3), 218–228.

Choo, J., R. Jiamthapthaksin, C. Chen, O. Celepcikay, C. Giusti, and C. Eick (2007). Mosaic: A proximity graph approach for agglomerative clustering. In *9th International Conference on Data Warehousing and Knowledge Discovery*, pp. 231–240.

Choueka, Y. (1988). Looking for needles in a haystack or locating interesting collocation expressions in large textual databases. In *Proceedings of the RIAO'88*, pp. 38–43.

Chung, F. (2007). The heat kernel as the pagerank of a graph. *Proceedings of the National Academy of Sciences 104*(50), 19735–19740.

Church, K. W. and P. Hanks (1989). Word association norms, mutual information, and lexicography. In *Proceedings of the 27th. Annual Meeting of the Association for Computational Linguistics*, Vancouver, B.C., pp. 76–83. Association for Computational Linguistics.

Cicurel, L., S. Bloehdorn, and P. Cimiano (2006). Clustering of polysemic words. In *Proceedings of the 30th Annual Conference of the German Classification Society*, pp. 595–602.

Cimiano, P. and S. Staab (2005). Learning concept hierarchies from text with a guided agglomerative clustering algorithm. In *Proceedings of the ICML 2005 Workshop on Learning and Extending Lexical Ontologies with Machine Learning Methods*, Bonn, Germany, pp. 6–15.

Clauset, A., M. Newman, and C. Moore (2004). Finding community structure in very large networks. *Physical Review E 70*(6 part 2), 066111.

Condon, A. and R. Karp (1999). Algorithms for graph partitioning on the planted partition model. In *Proceedings of the Third International Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, London, UK, pp. 221–232. Springer.

Constant, D. (1995). L'analyseur linguistique SYLEX. In *Cinquième école d'été du Centre National des Télécommunications*, Lannion, France, pp. 8.

Cucchiarelli, A., R. Navigli, F. Neri, and P. Velardi (2004). Automatic generation of glosses in the ontolearn system. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, pp. 1293–1296. European Language Resources Association.

Cutting, D., J. Kupiec, J. Pedersen, and P. Sibun (1992). A practical part-of-speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*, pp. 133–140.

Cutting, D., J. Pedersen, D. Karger, and J. Tukey (1992). Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of SIGIR '92*, Copenhagen, Denmark, pp. 318–329. ACM Press.

Dagan, I. and K. Church (1994). Termight: identifying and translating technical terminology. In *Proceedings of the 4th conference on Applied natural language processing*, San Francisco, CA, USA, pp. 34–40. Morgan Kaufmann Publishers Inc.

Dai, J. and H. Lee (1994). Parsing with tag information in a probabilistic generalized LR parser. In *Proceedings of the Internation Conference on Chinese Computing*, pp. 33–39. National University of Singapore.

Dai, Y., T. E. Loh, and C. S. G. Khoo (1999). A new statistical formula for chinese text segmentation incorporating contextual information. In *Proceedings of SIGIR '99*, New York, NY, USA, pp. 82–89. ACM Press.

Declerck, T. (2002). A set of tools for integrating linguistic and non-linguistic information. In *Proceedings of SAAKM (Workshop on the 15th European Conference on Artifical Intelligence)*, pp. 5.

Deerwester, S., S. Dumais, T. Landauer, G. Furnas, and R. Harshman (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science 41*(6), 391–407.

Degen, W., B. Heller, H. Herre, and B. Smith (2001). GOL: toward an axiomatized upper-level ontology. In *Proceedings of the FOIS '01*, New York, NY, USA, pp. 34–46. ACM Press.

Dempster, A., N. Laird, and D. Rubin (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society 39*(1), 1–38.

Deza, M. and E. Deza (2006). *Dictionary of Distances*. Elsevier Science.

Dias, G. (2002). *Extraction Automatique dAssociations Lexicales partir de Corpora*. Ph. D. thesis, New University of Lisbon (Portugal) and LIFO University of Orlans (France), Lisbon, Portugal.

Dias, G., S. Guilloré, and J. G. P. Lopes (1999a). Multilingual aspects of multiword lexical units. In *Workshop on Language Technologies in the Framework of the 32rd Annual Meeting of the Societas Linguistica Europaea*, Ljubljana, Slovenia, pp. 11–21.

Dias, G., S. Guilloré, and J. G. P. Lopes (1999b). Mutual expectation and LocalMax algorithm for multiword lexical unit extraction. In *Ninth Portuguese Conference on Artificial Intelligence*, Evora, Portugal.

Dice, L. (1945). Measures of the amount of ecological association between species. *Ecology 26*, 297–302.

Diestel, R. (2005). *Graph theory*. Heidelberg, New York: Springer Verlag.

do Nascimento, H. and P. Eades (2001). A system for graph clustering based on user hints. In *VIP '00: Selected papers from the Pan-Sydney workshop on Visualisation*, Darlinghurst, Australia, Australia, pp. 73–74. Australian Computer Society, Inc.

Doms, A. and M. Schroeder (2005). GoPubMed: exploring PubMed with the gene ontology. *Nucleic Acids Research 33*.

Donetti, L. and M. Muñoz (2004). Detecting network communities: a new systematic and efficient algorithm. *Journal of Statistical Mechanics: Theory and Experiment 2004*(10), P10012.

Dopazo, J. and J. Carazo (1997). Phylogenic reconstruction using a growing neural network that adopts the topology of a phylogenic tree. *Molecular Evolution 44*, 226–233.

Dorow, B. (2006). *A Graph Model for Words and their Meanings*. Ph. D. thesis, University of Stuttgart, Stuttgart, Germany.

Drouin, P. (2004). Detection of domain specific terminology using corpora comparison. In *Proceedings of the 4th International Conference of Language Resources and Evaluation LREC 2004*, pp. 79–82.

Du, H., M. Feldman, S. Li, and X. Jin (2007). An algorithm for detecting community structure of social networks based on prior knowledge and modularity. *Complexity 12*(3), 53–60.

Duda, R., P. Hart, and D. Stork (2001). *Pattern Classification* (2 ed.). Wiley-Interscience Publication.

Dunn, J. C. (1974). Well separated clusters and optimal fuzzy-partitions. *Journal of Cybernetics* (4), 4–10.

Edachery, J., A. Sen, and F. Brandenburg (1999). Graph clustering using distance-k cliques. In *GD '99: Proceedings of the 7th International Symposium on Graph Drawing*, London, UK, pp. 98–106. Springer.

Elias, P., A. Feinstein, and C. Shannon (1956). A note on the maximum flow through a network. *IRE Transactions on Information Theory IT-2*, 117 – 199.

Endres, B. (2005). Jatke: A platform for the integration of ontology learning approaches.

Ester, M., H.-P. Kriegel, J. Sander, and X. Xu (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, Portland, USA, pp. 226–231. AAAI.

Faatz, A. and R. Steinmetz (2002). Ontology enrichment with texts from the WWW. In *Proceedings of the ECML/PKDD-2002*, Helsinki, Finland, pp. 20–35. Springer.

Faure, D. and C. Nédellec (1999). Knowledge acquisition of predicate argument structures from technical texts using machine learning: The system ASIUM. In *Proceedings of the EKAW '99*, pp. 329–334.

Faure, D. and T. Poibeau (2000). First experiences of using semantic knowledge learned by ASIUM for information extraction task using intex. In *ECAI Workshop on Ontology Learning*, Volume 31. CEUR-WS.org.

Ferreira da Silva, J. and J. Pereira Lopes (1999). A local maxima method and a fair dispersion normalization for extracting multi-words units from corpora. In *Sixth Meeting on Mathematics of Language*, Orlando, USA, pp. 369–381.

Ferrer-i-Cancho, R. and R. Sole (2001). The small world of human language. *Proceedings of The Royal Society of London. Series B, Biological Sciences 268*(1482), 2261–2265.

Firth, J. (1957). A synopsis of linguistic theory 1930-1955. In F. Palmer (Ed.), *Selected Papers of J.R. Firth 1952-1959*, pp. 168–205.

Flake, G., S. Lawrence, and C. Giles (2000). Efficient identification of web communities. In *Proceedings of the 6th ACM SIGKDD*, Boston, MA, pp. 150–160.

Flake, G., R. Tarjan, and K. Tsioutsiouliklis (2004). Graph clustering and minimum cut trees. *Internet Mathematics 1*(4), 385–408.

Fleischman, M. and E. Hovy (2003). Recommendations without user preferences: A natural language processing approach. In *Proceedings of the 7th International Conference on Intelligent User Interfaces (IUI). Miami Beach, FL.*

Frakes, W. (1984). Term conflation for information retrieval. In *Proceedings of SIGIR '84*, Swinton, UK, pp. 383–389. British Computer Society.

Frakes, W. and R. Baeza-Yates (Eds.) (1992). *Information Retrieval: Data Structures & Algorithms*. Prentice-Hall.

Franti, P., O. Virmajoki, and V. Hautamaki (2006). Fast agglomerative clustering using a k-nearest neighbor graph. *IEEE Transactions on Pattern Analysis and Machine Intelligence 28*(11), 1875–1881.

Fredkin, E. (1960). Trie memory. *Communications of the ACM 3*(9), 490–499.

Girvan, M. and M. Newman (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences 99*, 7821–7828.

Giuliano, V. E. (1964). The interpretation of word associations. In *Proceedings of the Symposiums on Statistical Association Methods for Mechanical Documentation*, Number 269, Washington D.C. NBS.

Gkantsidis, C., M. Mihail, and E. Zegura (2003). Spectral analysis of internet topologies. In *IEEE Conference on Computer Communications*, pp. 364–374.

Glover, F. and M. Laguna (1997). *Tabu Search*. Kluwer.

Gómez-Pérez, A., M. Fernandez-Lopez, and O. Corcho (2004). *Ontological Engineering: With Examples from the Areas of Knowledge Management, E-Commerce and the Semantic Web (Advanced Information and Knowledge Processing)*. Springer.

Gosset, W. (1908). The probable error of a mean. *Biometrika 6*(1), 1–25.

Gray, J. (1981). The transaction concept: Virtues and limitations (invited paper). In *Proceedings of the 7th International Conference on Very Large Data Bases*, pp. 144–154. IEEE Computer Society.

Guarino, N. (1998). Formal ontology and information systems. In *Proceedings of the 1st International Conference on Formal Ontologies in Information Systems*, Trento, Italy, pp. 3–15. IOS Press.

Gupta, A. and S. Lam (1998). Weight decay backpropagation for noisy data. *Neural Networks 11*(6), 1127–1137.

Hahn, U. and K. G. Markò (2001). Joint knowledge capture for grammars and ontologies. In *Proceedings of the 1st international conference on Knowledge Capture*, New York, NY, USA, pp. 68–75. ACM Press.

Hammarström, H. (2006). Poor man's stemming: Unsupervised recognition of same-stem words. In *Proceedings of the Asian Information Retrieval Symposium*, pp. 323–337.

Hamming, R. (1950). Error-detecting and error-correcting codes. In *Bell System Technical Journal*, Volume 29(2), pp. 147–160.

Han, J. and M. Kamber (2001). *Data Mining - Concepts and Techniques* (First ed.). New York, USA: Elsevier Science & Technology Books.

Harel, D. and Y. Koren (2001). On clustering using random walks. In *Proceedings of the 21st Conference on Foundations of Software Technology and Theoretical Computer Science*, London, UK, pp. 18–41. Springer.

Harris, Z. (1968). *Mathematical structures of language.* New York: Interscience Publishers: John Wiley & Sons.

Hartuv, E. and R. Shamir (2000). A clustering algorithm based on graph connectivity. *Information Processing Letters 76*(4-6), 175–181.

Hastings, W. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika 57*(1), 1849–1850.

Hearst, M. (1998). Automated discovery of wordnet relations. In *Wordnet: An Electronic Lexical Database*, Cambridge, Massachusetts, USA, pp. 68–75. MIT Press.

Hearst, M. A. (1992). Automatic Acquisition of Hyponyms from Large Text Corpora. In *Proceedings of the Fourteenth Conference on Computational Linguistics*, Morristown, NJ, USA, pp. 539–545. Association for Computational Linguistics.

Henstock, P. V., D. J. Pack, Y.-S. Lee, and C. J. Weinstein (2001). Toward an improved concept-based information retrieval system. In *Proceedings of SIGIR '01*, New York, NY, USA, pp. 384–385. ACM.

Hersh, W., E. Campbell, D. Evans, and N. Brownlow (1996). Empirical, automated vocabulary discovery using large text corpora and advanced natural language processing tools. In *Proceedings of the 1996 AMIA Annual Fall Symposium*, Number 269, Philadelphia, PA, pp. 159–163. Hanley & Belfus.

Heyer, G., M. Luter, U. Quasthoff, T. Wittig, and C. Wolff (2001). Learning relations using collocations. In *Workshop on Ontology Learning*, Volume 38 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Heyer, G., U. Quasthoff, and T. Wittig (2006). *Text Mining: Wissensrohstoff Text. Konzepte, Algorithmen, Ergebnisse.* W3l.

Hindle, D. (1990). Noun classification from predicate-argument structures. In *Meeting of the Association for Computational Linguistics*, pp. 268–275.

Hockenmaier, J. and C. Brew (1998). Error-driven segmentation of chinese. *Communications of COLIPS 1*(1), 69–84.

Hoehndorf, R., A.-C. Ngonga Ngomo, and M. Dannemann (2008). Towards ontological interpretations for improved text mining. In *Proceedings of the third International Symposium on Semantic Mining in Biomedicine*, pp. 165–166. TUCS.

Hoehndorf, R., A.-C. Ngonga Ngomo, M. Dannemann, and J. Kelso (2008). From terms to categories: Testing the significance of co-occurrences between ontological categories. In *Proceedings of the 3rd International Symposium on Semantic Mining in Biomedicine*, pp. 53–60. TUCS.

Hoos, H. and T. Stützle (1999). Systematic vs. local search for sat. In *Proceedings of the 23rd Annual German Conference on Artificial Intelligence*, London, UK, pp. 289–293. Springer.

Hoschek, W. (2004). The COLT project. http://acs.lbl.gov/∼hoschek/colt/. Visited on September 25th, 2008.

Jaccard, P. (1901). Etude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin de la Societe Vaudoise de Sciences Naturelles* (37), 547–579.

Jacob, A. (1999). Development of object oriented frameworks for spatio-temporal information systems. In *Proceedings of the 21st international conference on Software engineering*, Los Alamitos, CA, USA, pp. 720–721. IEEE Computer Society Press.

Jain, A. and R. Dubes (1988). *Algorithms for clustering data*. Upper Saddle River, NJ, USA: Prentice-Hall.

Jain, A., M. Murty, and P. Flynn (1999). Data clustering: a review. *ACM Computing Surveys 31*(3), 264–323.

Jiang, X. and A. Tan (2005). Mining ontological knowledge from domain-specific text documents. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, Washington, DC, USA, pp. 665–668. IEEE Computer Society.

Johnson, D., C. Aragon, L. McGeoch, and C. Schevon (1989). Optimization by simulated annealing: an experimental evaluation. Part I, graph partitioning. *Operations Research 37*(6), 865–892.

Johnson, E., A. Mehrotra, and G. Nemhauser (1993). Min-cut clustering. *Mathetical Programmming 62*(1), 133–151.

Joseph, J., M. Carrasco, D. Fain, K. Lang, and L. Zhukov (2003). Clustering of bipartite advertiser-keyword graph. In *Workshop on Large Scale Clustering at IEEE ICDM 2003*. IEEE.

Justeson, J. and S. Katz (1991). Co-occurrences of antonymous adjectives and their contexts. *Computational Linguistics 17*(1), 1–19.

Kannan, R., S. Vempala, and A. Veta (2000). On clustering - good, bad and spectral. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, Washington, DC, USA, pp. 367. IEEE Computer Society.

Kaufmann, L. and P. Rousseeuw (1987). Clustering by means of medoids. In *Statistical Data Analysis based on the L_1-Norm*, Elsevier, Holland, pp. 405–416.

Kaufmann, L. and P. Rousseeuw (2001). *Finding Groups in Data: an Introdution to Cluster Analysis* (Second ed.). New York, USA: Wiley and Sons.

Kettunen, K. (2006). Developing an automatic linguistic truncation operator for best-match retrieval of finnish in inflected word form text database indexes. *Journal of Information Science 32*(5), 465–479.

Khan, L. and F. Luo (2002). Ontology construction for information selection. In *Proceedings of the ICTAI'02*, Washington DC, USA, pp. 122–127. IEEE Computer Society.

Kirkpatrick, S., C. Gelatt, and M. Vecchi (1983). Optimization by simulated annealing. *Science 220*(4598), 671–680.

Kleinberg, J. and S. Lawrence (2001). Network analysis: The structure of the Web. *Science 294*(5548), 1849–1850.

Kohonen, T. (1989). *Self-Organization and Associative Memory* (Third ed.). New York, USA: Springer.

Kosinov, S. (2001). Evaluation of n-grams conflation approach in text-based information retrieval. *Proceedings of the Conference on String Processing and Information Retrieval*, 136–142.

Krishna, K. and C. Krishna (1978). Disaggregative clustering using the concept of mutual nearest neighborhood. *IEEE Transactions on Systems, Man and Cybernetics* (8), 888–894.

Lang, K. and R. Andersen (2007). Finding dense and isolated submarkets in a sponsored search spending graph. In *Proceedings of the sixteenth ACM conference on Conference on Information and Knowledge Management*, New York, NY, USA, pp. 613–622. ACM.

Lee, I., M. Berk, T. Marcus, K. Reighley, S. Reynolds, M. Rubin, C. Sharp, R. Young, D. Toop, and P. Shapiro (2000). *Modulations: A History of Electronic Music: Throbbing Words on Sound*. D.A.P./Caipirinha.

Leech, G., R. Garside, and M. Bryant (1994). Claws4: The tagging of the british national corpus. In *COLING'94*, pp. 622–628.

Lezius, W., R. Rapp, and M. Wettler (1998). A freely available morphological analyzer, disambiguator and context sensitive lemmatizer for german. In *Proceedings of the 17th international conference on Computational linguistics*, Morristown, NJ, USA, pp. 743–748. Association for Computational Linguistics.

Lin, D. (1998). Dependency-based evaluation of MINIPAR. In *Proceedings Workshop on the Evaluation of Parsing Systems*, Granada.

Lin, D. and P. Pantel (2002). Concept discovery from text. In *Proceedings of the 19th international conference on Computational linguistics*, Morristown, NJ, USA, pp. 1–7. Association for Computational Linguistics.

Lovins, J. (1968). Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics 11*, 22–31.

Lua, K. and K. Gan (1994). An application of information theory in chinese word segmentation. *Computer Processing of Chinese and Oriental Languages 8*(1), 115–123.

Maedche, A. (2002). *Ontology Learning for the Semantic Web*. Kluwer International Series in Engineering an. Boston, MA, USA: Kluwer Academic Publishers.

Maedche, A. and S. Staab (2000). Semi-automatic engineering of ontologies from text. In *Proceedings of 12th International Conference on Software and Knowledge Engineering*, Chicago, IL, pp. 231–239.

Maedche, A. and S. Staab (2001). Ontology learning for the semantic web. *IEEE Intelligent Systems 16*(2), 72–79.

Maedche, A. and S. Staab (2004). Ontology learning. In S. Staab and R. Studer (Eds.), *Handbook on Ontologies*, International Handbooks on Information Systems, pp. 173–190. Springer.

Mahalanobis, P. (1936). On generalized distance in statistics. *Proceedings of the National Institute of Science (India) 12*, 49–55.

Manning, C. and H. Schütze (1999). *Foundations of Statistical Natural Language Processing* (First ed.). Cambridge, Massachussets: MIT Press.

Matula, D. W. and F. Shahrokhi (1990). Sparsest cuts and bottlenecks in graphs. *Discrete Applied Mathematics 27*(1-2), 113–123.

McQueen, J. (1967). Some methods of classification and analysis of multivariate observations. In *Proceedings of Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297.

Melucci, M. and N. Orio (2003). A novel method for stemmer generation based on hidden markov models. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, New York, NY, USA, pp. 131–138. ACM Press.

Mikheev, A. and S. Finch (1997). A workbench for finding structure in texts. In *Proceedings of the fifth conference on Applied natural language processing*, San Francisco, CA, USA, pp. 372–379. Morgan Kaufmann Publishers Inc.

Milgram, S. (1967). The small world problem. *Psychology Today*, 60–67.

Miller, G. (1990). Word-net: An on-line lexical database. *International Journal of Lexicography 3*(4), 235–244.

Minock, M. (2005). Where are the killer applications of restricted domain question answering? In *Proceedings of the IJCAI Workshop on Knowledge Reasoning in Question Answering*, pp. 4.

Missikof, M., R. Navigli, and P. Velardi (2002). Integrated approach to web ontology learning and engineering. *Computer 35*(11), 60–63.

Moldovan, D. and R. Girju (2001). An interactive tool for the rapid development of knowledge bases. *Internation Journal on Artificial Intelligence Tools 10*(1–2), 65–86.

Moldovan, D., R. Girju, and V. Rus (2000). Domain-specific knowledge acquisition from text. In *Proceedings of the sixth conference on Applied natural language processing*, San Francisco, CA, USA, pp. 268–275. Morgan Kaufmann Publishers Inc.

Mollá, D. and J. Vicedo (2007). Question answering in restricted domains: An overview. *Computational Linguistics 33*(1), 41–61.

Monien, B. and R. Diekmann (1997). A local graph partitioning heuristic meeting bisection bounds. In *Proceedings of the Eighth SIAM Conference on Parallel Processing for Scientific Computing*.

Morrison, D. (1968). Patriciapractical algorithm to retrieve information coded in alphanumeric. *Journal of the Association for Computing Machinery 15*(4), 514–534.

Mougin, F. and O. Bodenreider (2005). Approaches to eliminating cycles in the umls metathesaurus: naive vs. formal. In *Proceedings of the AMIA Annual Symposium*, pp. 550–554.

Muslea, I. (1999). Extraction patterns for information extraction tasks: A survey. In *Proceedings of the AAAI-99 Workshop on Machine Learning for Information Extraction*.

Navigli, R. and P. Velardi (2004). Learning domain ontologies from document warehouses and dedicated websites. *Computational Linguistics 30*(2), 151–179.

Newman, M. (2004). Detecting community structure in networks. *The European Physical Journal B - Condensed Matter 38*(2), 321–330.

Ngonga Ngomo, A.-C. (2006). CLIque-based clustering. In *Proceedings of Knowledge Sharing and Collaborative Engineering Conference*, St. Thomas, VI, USA, pp. 16–19.

Ngonga Ngomo, A.-C. (2008a). Knowledge-free discovery of multi-word units. In *Proceedings of the 23rd Annual ACM Symposium on Applied Computing*, pp. 1561–1565. ACM Press.

Ngonga Ngomo, A.-C. (2008b). SIGNUM: A graph algorithm for terminology extraction. In *Proceedings of CICLing' 2008*, pp. 85–95. Springer.

Niles, I. and A. Pease (2001). Towards a standard upper ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems*, New York, NY, USA, pp. 34–46. ACM Press.

Omelayenko, B. (2001). Learning of ontologies for the web: the analysis of existent approaches. In *Proceedings of the International Workshop on Web Dynamics*, pp. 268–275.

Orponen, P. and S. Schaeffer (2005). Local clustering of large graphs by approximate Fiedler vectors. In *Proceedings of the 4th International Workshop on Efficient and Experimental Algorithms*, Volume 3503 of *Lecture Notes in Computer Science*, pp. 524–533. Springer.

Palmer, D. (1997). A trainable rule-based algorithm for word segmentation. In *Proceedings of the 35th annual meeting on Association for Computational Linguistics*, Morristown, NJ, USA, pp. 321–328. Association for Computational Linguistics.

Pantel, P. (2003). *Clustering by Committee*. Ph. D. thesis, University of Alberta, Edmonton, Alberta, Canada.

Pantel, P. and D. Lin (2002). Discovering word senses from text. In *Proceedings of SIGKDD-02*, Edmonton, Canada, pp. 613–619. ACM Press.

Paulussen, H. and W. Martin (1992). Dilemma-2: a lemmatizer-tagger for medical abstracts. In *Proceedings of the third conference on Applied natural language processing*, Morristown, NJ, USA, pp. 141–146. Association for Computational Linguistics.

Pearsall, J. (Ed.) (2001). *The New Oxford Dictionary of English*. Oxford, UK: Oxford University Press.

Perera, P. and R. Witte (2005). A self-learning context-aware lemmatizer for german. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, Morristown, NJ, USA, pp. 636–643. Association for Computational Linguistics.

Picton, P. (2000). *Neural Networks*. Indianapolis, IN, USA: Macmillan Publishing Co., Inc.

Ponte, J. and W. Croft (1996). Useg: A retargetable word segmentationprocedure for information retrieval. Technical Report TR-98-01, University of Massachusetts Amherst, Amherst.

Popovic, M. and P. Willett (1992). The effectiveness of stemming for natural-language access to slovene textual data. *JASIS 43*(5), 384–390.

Porter, M. (1980). An algorithm for suffix stripping. *Program 14*(3), 130–137.

Qian, J., M. Dolled-Filhart, J. Lin, H. Yu, and M. Gerstein (2001). Beyond synexpression relationships: Local clustering of time-shifted and inverted gene expression profiles identifies new, biologically relevant interactions. *Journal of Molecular Biology 314*, 1053–1066.

Qiu, Y. and H.-P. Frei (1993). Concept-based query expansion. In *Proceedings of SIGIR'93*, Pittsburgh, US, pp. 160–169.

Ratnaparkhi, A. (1996). A maximum entropy model for part-of-speech tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 133–142. Somerset, New Jersey: Association for Computational Linguistics.

Robertson, S. E. and D. Hull (2001). The TREC 2001 filtering track report. In *Proceedings of the Text REtrieval Conference*.

Robertson, S. E. and K. S. Jones (1976). Relevance Weighting of Search Terms. *Journal of the American Society for Information Science 27*(3), 129–146.

Rousseeuw, P. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics 20*(1), 53–65.

Ruge, G. (1992). Experiments on linguistically-based term associations. *Information Processing and Management 28*(3), 317–332.

Salton, G. and M. McGill (1986). *Introduction to Modern Information Retrieval*. New York, NY, USA: McGraw-Hill, Inc.

Salton, G., A. Wong, and C. Yang (1975). A vector space model for automatic indexing. *Communications of the ACM 18*(11), 613–620.

Sanderson, M. and B. Croft (1999). Deriving concept hierarchies from text. In *Proceedings of SIGIR '99*, New York, NY, USA, pp. 206–213. ACM.

Santorini, B. (1990). Part-of-speech tagging guidelines for the penn treebank project. Technical Report MS-CIS-90-47, Department of Computer and Information Science, University of Pennsylvania.

Savoy, J. (1999). A stemming procedure and stopword list for general french corpora. *Journal of the American Society of Information Science 50*(10), 944–952.

Schaeffer, S. (2005). Stochastic local clustering for massive graphs. In T. Ho, D. Cheung, and H. Liu (Eds.), *Proceedings of the Ninth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD-05)*, Volume 3518 of *LNCS*, pp. 354–360. Springer.

Schaeffer, S. (2007). Graph clustering. *Computer Science Review 1*(1), 27–64.

Schone, P. (2001). *Toward Knowledge-Free Induction of Machine-Readable Dictionaries*. Ph. D. thesis, University of Colorado at Boulder, Boulder, USA.

Schone, P. and D. Jurafsky (2001). Is knowledge-free induction of multiword unit dictionary headwords a solved problem? In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pp. 100–108.

Schütze, H. (1998). Automatic word sense discrimination. *Computational Linguistics 24*(1), 97–123.

Sedgewick, R. (1988). *Algorithms* (2 ed.). Addison-Wesley.

Shannon, C. (1948). A mathematic theory of communication. *Bell System Technical Journal 27*, 379–423.

Siadatyand, M., J. Shu, and W. Knaus (2007). Relemed: Sentence-level search engine with relevance score for the medline database of biomedical articles. *BMC Medical Informatics and Decision Making 7*, 1–11.

Singh, S., K. Gupta, M. Shrivastava, and P. Bhattacharyya (2006). Morphological richness offsets resource demand- experiences in constructing a pos tagger for hindi. In *Proceedings of the COLING*, Morristown, NJ, USA, pp. 779–786. Association for Computational Linguistics.

Smadja, F. A. (1993). Retrieving collocations from text: Xtract. *Computational Linguistics 19*(1), 143–177.

Srikant, R. and R. Agrawal (1995). Mining generalized association rules. In *Proceedings of 21th International Conference on Very Large Data Bases*, pp. 407–419. Morgan Kaufmann.

Stahl, A. (2005). Learning similarity measures: A formal view based on a generalized cbr model. In *Proceedings of the 6th International Conference on Case-Based Reasoning*, Volume 3620 of *LNCS*, Chicago, IL, pp. 507–521. Springer.

Steinbach, M., G. Karypis, and V. Kumar (2000). A comparison of document clustering techniques. Technical Report 00-034, Department of Computer Science and Engineering, University of Minnesota.

Steyvers, M. and J. B. Tenenbaum (2005). The large-scale structure of semantic networks: Statistical analyses and a model of semantic growth. *Cognitive Science 29*(1), 41–78.

Stoica, P. and R. Moses (1997). *Introduction to Spectral Analysis*. Prentice Hall.

Tan, P., M. Steinbach, and V. Kumar (2005). *Introduction to Data Mining* (1 ed.). Addison Wesley.

Teahan, W., Y. Wen, R. McNab, and I. Witten (2000). A compression-based algorithm for chinese word segmentation. *Computational Linguistics 26*(3), 375–393.

Thanopoulos, A., N. Fakotakis, and G. Kokkinakis (2002). Comparative evaluation of collocation extraction metrics. In *Proceedings of the 3rd International Conference on Language Resource and Evaluation*, pp. 620–625.

Thanopoulos, A., N. Fakotakis, and G. Kokkinakis (2003). Text tokenization for knowledge-free automatic extraction of lexical similarities. In *Traitement Automatique de la Langue Naturelle (TALN)*, pp. 397–403.

Thelen, M. and E. Riloff (2002). A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *Proceedings of the Conference on Empirical methods in natural language processing*, Morristown, NJ, USA, pp. 214–221. Association for Computational Linguistics.

Theodoridis, S. and K. Koutroumbas (2006). *Pattern Recognition*. Academic Press.

Tlili-Guiassa, Y. (2006). Hybrid method for tagging arabic text. *Journal of Computer Science 2*(3), 245–248.

Turmo, J., A. Ageno, and N. Català (2006). Adaptive information extraction. *ACM Computing Survey 38*(2), 1–47.

van Dongen, S. (2000). *Graph Clustering by Flow Simulation*. Ph. D. thesis, University of Utrecht.

Vossen, P. (Ed.) (1998). *EuroWordNet: a multilingual database with lexical semantic networks*. Norwell, MA, USA: Kluwer Academic Publishers.

Wanas, N., D. Said, N. Hegazy, and N. Darwish (2006). A study of local and global thresholding techniques in text categorization. In *AusDM '06: Proceedings of the fifth Australasian conference on Data mining and analystics*, Darlinghurst, Australia, Australia, pp. 91–101. Australian Computer Society, Inc.

Wermter, J. and U. Hahn (2005). Paradigmatic modifiability statistics for the extraction of complex multi-word terms. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, Morristown, NJ, USA, pp. 843–850. Association for Computational Linguistics.

Widdows, D. and B. Dorow (2002). A graph model for unsupervised lexical acquisition. In *Proceedings of the 19th international conference on Computational linguistics*, Morristown, NJ, USA, pp. 1–7. Association for Computational Linguistics.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin 1*(6), 80–83.

Wilkinson, R. and P. Hingston (1991). Using the cosine measure in a neural network for document retrieval. In *Proceedings of SIGIR '91*, New York, NY, USA, pp. 202–210. ACM.

Witschel, H. (2004). *Terminologie-Extraktion: Mlichkeiten der Kombination statistischer und musterbasierter Verfahren.* Content and Communication: Terminology, Language Resources and Semantic Interoperability. Würzburg: Ergon Verlag.

Wong, M. (1982). A hybrid clustering method for identifying high-density clusters. *Journal of the American Statistical Association 77*(380), 841–847.

Wu, Z. and G. Tseng (1995). Acts: an automatic chinese text segmentation system for full text retrieval. *Journal of the American Society of Information Science 46*(2), 83–96.

Yao, Y. and K. Lua (1998). Splitting-merging model of chinese word tokenization and segmentation. *Natural Language Engineering 4*(4), 309–324.

Zesch, T. and I. Gurevych (2007). Analysis of the Wikipedia Category Graph for NLP Applications. In *Proceedings of the NAACL-HLT 2007 Workshop on TextGraphs*, pp. 1–8.

Zhang, T., R. Ramakrishnan, and M. Livny (1996). Birch: An efficient data clustering method for very large databases. In H. V. Jagadish and I. S. Mumick (Eds.), *Proceedings of SIGMOD-96*, Montreal, Canada, pp. 103–113.

Zhang, Y., R. Brown, R. Frederking, and A. Lavie (2001). Pre-processing of bilingual corpora for mandarin-english ebmt. In *Proceedings of the 8th Machine Translation Summit*.

Zhou, L. (2007). Ontology learning: state of the art and open issues. *Information Technology and Management 8*(3), 241–252.

Zhou, L. and Q. Liu (2002). A character-net based chinese text segmentation method. In *COLING-02 on SEMANET*, Morristown, NJ, USA, pp. 1–6. Association for Computational Linguistics.

Curriculum Vitae

Personal Data

Family name	Ngonga Ngomo
Given name	Axel-Cyrille
Date of birth	August 4^{th}, 1983
Place of birth	Bafoussam, Cameroon
Nationality	Cameroonian

Education

09/1987 – 07/1998	A-Levels & Baccalauréat
02/1999 – 03/1999	Studienkolleg Sachsen Certificate German as language for graduate studies (good)
01/1999 – 02/1999	Studienkolleg Sachsen Certificate German as Foreign Language (very good)
09/1999 – 04/2004	Diplom in Informatik (very good)
05/2004 – today	Doctoral studies

Work Experience

05/2004 – 12/2006	Graduate assistant at the University of Leipzig. Project: PreBuilt Information Spaces
01/2003 – 04/2004	Student assistant at the University of Leipzig. Project: PreBuilt Information Spaces

08/2002 – 12/2007	Part-time Webmaster at the Leipzig Graduate School of Management
02/2002 – 05/2002	Research intern at the Fraunhofer IAO. Research Project: Pi-AVIDA
09/2001 – 02/2002	Web-Programmer at the University of Leipzig. Project: "House of the five continents"

Awards and Prices

2008	Best student paper award at the CicLING' 2008
09/2001 – 02/2002	DAAD STIBET scholarship
2003	DAAD Award "Best Foreign student" from the University of Leipzig
06/2004 – 05/2007	Scholarship from the Graduate School for Knowledge Representation at the University of Leipzig
06/2007 – today	Scholarship from the German Ministry for Education and Research

Related Peer-Reviewed Publications

- Gebauer, M. and A.-C. Ngonga Ngomo (2008). SMORE - a semantic model repository. In *To appear in Proceedings of the 1st Workshop of Knowledge Reuse at the International Conference on Software Reuse*. Springer.

- Hoehndorf, R., A.-C. Ngonga Ngomo, and M. Dannemann (2008). Towards ontological interpretations for improved text mining. In *Proceedings of the Third International Symposium on Semantic Mining in Biomedicine*, pp. 165–166. Turku Centre for Computer Science (TUCS).

- Hoehndorf, R., A.-C. Ngonga Ngomo, M. Dannemann, and J. Kelso (2008). From terms to categories: Testing the significance of co-occurrences between ontological categories. In *Proceedings of the Third International Symposium on Semantic Mining in Biomedicine*, pp. 53–60. Turku Centre for Computer Science (TUCS).

- Ngonga Ngomo, A.-C. (2008a). Knowledge-free discovery of domain-specific multiword units. In *Proceedings of the ACM SAC' 2008*, pp. 1561–1565. ACM

Press.

- Ngonga Ngomo, A.-C. (2008b). SIGNUM: A graph algorithm for terminology extraction. In *Proceedings of CICLing' 2008*, pp. 85–95. Springer.

- Ngonga Ngomo, A.-C. (2008c). Towards an implicit and collaborative evolution of terminological ontologies. In *Building the knowledge society on the internet*, pp. 65–88. Hershey, PA, USA: IGI Global.

- Ngonga Ngomo, A.-C. (2007). Adaptive and context-sensitive information retrieval. *Series on Innovation and Knowledge Management 5*, 289–300.

- Ngonga Ngomo, A.-C. and F. Schumacher (2007). Involving the user in semantic search. In *Proceedings of the 12th Human Computer Interaction International Conference*, pp. 507–516.

- Ngonga Ngomo, A.-C. and H. Witschel (2007). A framework for adaptive information retrieval. In *Proceedings of the 1st International Conference on Theoretical Information Retrieval*, pp. 105–113. Alma Mater Series.

- Ngonga Ngomo, A.-C. (2006). CLIque-based clustering. In *Proceedings of Knowledge Sharing and Collaborative Engineering Conference*, St. Thomas, VI, USA, pp. 16–19.

- Ngonga Ngomo, A.-C. and F. Schumacher (2006). Implicit knowledge sharing. In *Proceedings of the 7th European Conference on Knowledge Management*, pp. 736–747.

- Ngonga Ngomo, A.-C. and K. Böhm (2005). Building adaptive knowledge spaces by combining machine learning algorithms with ontologies and text mining technologies. In *Proceedings of the GfKL' 2005*, pp. 266.

- Härtwig, J. and A.-C. Ngonga Ngomo (2004). Kontext-dynamische informationsversorgung durch prozesse und ontologien. In *In Proceedings of the Leipziger Beiträge zur Informatik*, pp. 38–46.

- Ngonga Ngomo, A.-C. and K.-P. Fähnrich (2003). Der informationsraum als metapher für die realisierung dynamischer informationsbedürfnisse im kontext rollenbasierter communities. In *Proceedings of the Leipziger Beiträge zur Informatik*, pp. 26–34.

Selbständigkeitserklärung

Hiermit erkläre ich, die vorliegende Dissertation selbständig und ohne unzulässige fremde Hilfe angefertigt zu haben. Ich habe keine anderen als die angeführten Quellen und Hilfsmittel benutzt und sämtliche Textstellen, die wörtlich oder sinngemäß aus veröffentlichten oder unveröffentlichten Schriften entnommen wurden, und alle Angaben, die auf mündlichen Auskünften beruhen, als solche kenntlich gemacht. Ebenfalls sind alle von anderen Personen bereitgestellten Materialien oder erbrachten Dienstleistungen als solche gekennzeichnet.

Leipzig, den 08. Dezember 2008

Axel-Cyrille Ngonga Ngomo