Natural Language Processing meets Business

Algorithms for Mining Meaning from Corporate Texts

Ravi Ashwin Ittoo

Published by: University of Groningen
Groningen
The Netherlands

Printed by: Ipskamp Drukkers B.V.

ISBN: 978-90-367-5257-2 (printed version)
978-90-367-5258-9 (electronic version)

RIJKSUNIVERSITEIT GRONINGEN

Natural Language Processing Meets Business

Algorithms for Mining Meaning from Corporate Texts

Proefschrift

ter verkrijging van het doctoraat in de
Economie en Bedrijfskunde
aan de Rijksuniversiteit Groningen
op gezag van de
Rector Magnificus, dr. E. Sterken,
in het openbaar te verdedigen op
donderdag 5 januari 2012
om 16.15 uur

door

Ravi Ashwin Ittoo
geboren op 25 september 1979
te Moka, Mauritius

Promotor	:	Prof.dr.ir. J.C. Wortmann
Copromotores	:	Dr. G. Bouma Dr. L. Maruster
Beoordelingscommissie	:	Prof. dr. E. Metais Prof. dr. A. van den Bosch Prof. dr. T.H.A. Bijmolt

ISBN: 978-90-367-5257-2 (printed version)
978-90-367-5258-9 (electronic version)

ACKNOWLEDGEMENTS

Many people have contributed to making this thesis and my PhD a success. First and foremost, I want to express my thanks to my supervisor, Hans Wortmann, for giving me the opportunity to pursue this PhD and for his guidance throughout the last 3.5 years. It was really a pleasure to work with Hans, and I look forward to continue this collaboration in future. Secondly, I would like to thank Gosse Bouma, one of my co-promotors, who willingly agreed to collaborate with our group in order to provide the necessary expertise in Natural Language Processing. This project would not have been possible without his support. His very critical and sound remarks when reviewing the initial drafts have significantly improved the thesis' quality. Besides our academic interests, we also share a similar passion for cycling, and I always enjoyed the cycling-related discussions with Gosse, which helped break the monotony of academic work. I would also like to thank Laura Maruster, my other co-promotor, for her useful advice, especially during the initial part of my PhD.

I want to thank my current and ex colleagues in RuG, especially in the department of B&ICT, for providing a conducive environment for my research. They are our secretaries Iris Huizinga, Durkje van Lingen-Elzinga, and Irene Ravenhorst; Chee Wee Tan and Eric Lim for providing the Singapore touch to our department; Hans van Uitert for his help on all matters concerning courses; Marco Stuit, Javid Koochaki and their respective partners for our nice dinners and other useful discussions; Wu Shiliang for being my "comrade" during the week-ends in the office; Nick van Beest for our conversations, mostly around motoring and non-academic affairs; Ype van Wijk for ensuring that I have lunch on time; Nick Szirbik, for our gourmet discussions on fine-dining and Wilrik Mook for trying to force me to practice the Dutch language. In particular, I would like to express my thanks to Karel de Bakker for being one of the nicest office mates one could ever dream of. He taught me a lot about Dutch culture, and without his crucial advice, this book would never have materialized.

My research was done in collaboration with other academic and industrial partners, who have provided tremendous support during my PhD. They include Lu Yuan, Joel Ribeiro, Ton Weijters, and Aarnout Brombacher from the Eindhoven University of Technology; Guillaume Stollman, Frank Spronck and Cees Wolvekamp from Philips Healthcare, Leif Sorensen and Susanne Korsch from Bang&Olufsen; Kristiaan Smits from Philips Consumer Lifestyle; Ludwig Nooyens from Xerox; and Arnold van Putten from ASML. My project was carried out as part of the DataFusion project, sponsored by the Netherlands Ministry of Economic Affairs, Agriculture and Innovation, under the IOP-IPCR program. In this respect, I would like to thank Michiel de Boer and and Joop Postema from AgentSchapNL.

I would like also to thank Jur Raatjes and the entire team (there are too many names to be listed) for organizing the cycling sessions in the evenings. I always look forward to these sessions every week as they are a means for relaxing and for keeping in shape. Special thanks also to Renate, Mieke, Francois, Petra, Pablo, Carmen Jin, Aditya, Asoka and Shiva, and to all the other personal friends (there are too many names to be listed) who have made my stay here enjoyable.

Finally, I would like to express my thanks and gratitude to my parents for their support and guidance throughout my life; to S.K. Rao for putting me back on the right track; to Linda Zhang for her kindness in informing me on this PhD a few years ago; and to the *one without a second.*

Table of Contents

Part I - Introduction and Context 1
Chapter 1 - Introduction 5
1.1 Background 5
1.2 Motivation 6
1.2.1 Manual Text Analyses 7
1.2.2 Information Retrieval Technologies 7
1.2.3 Natural Language Processing (NLP) 9
1.2.4 NLP Algorithms: State-of-the-Art and Challenges 11
1.3 Problem Statement, Objective and Contribution 13
1.4 Meaningful Information 14
1.4.1 Terms 14
1.4.2 Semantic Relations between Terms 14
1.4.3 Relations between Documents/Document Clustering 15
1.5 Steps in the Quest for Meaningful Information 16
1.6 Specific Research Questions 17
1.7 Thesis Overview and Publications Included 18
Part II - Terms 21
Chapter 2 - Term Extraction 25
2.1 Introduction 26
2.2 Related Work 27
2.2.1 Terms 27
2.2.2 Simple and Complex Terms 28
2.2.3 Base-Terms and Term Formation 28
2.2.4 (Domain-Specific) Terms vs. (General) Words 28
2.2.5 Unithood 29
2.2.6 Termhood 30
2.2.7 Automatic Approaches for Term Extraction 30
2.2.8 Linguistic Techniques: Unithood 30
2.2.9 Statistical techniques: Unithood 31
2.2.10 Statistical Approach: Termhood 31
2.2.11 Hybrid Approach: Unithood and Termhood 32
2.3 Automatic Term Extraction Challenges 34
2.4 Contributions 36
2.5 ExtTerm Framework for Term Extraction 37
2.5.1 Document Pre-processing 37
2.5.2 Linguistic Filtering 39
2.5.3 Relevant Term Selection 40
2.5.4 Term Ranking 42
2.6 Experimental Evaluation 47
2.6.1 Corpora 47

2.6.2 Linguistic Filtering 48
2.6.3 Relevant Term Selection 49
2.6.4 Term Ranking 51
2.6.5 Evaluating ExtTerm's Output and Selecting Threshold 51
2.6.6 Benchmarking against Baseline 53
2.6.7 Influence of Term Frequency and Length 53
2.7 Conclusion 55
Part III - Semantic Relations 57
Chapter 3 –An Overview of Semantic Relations 61
3.1 Introduction 61
3.2 Representing Semantic Relations 61
3.3 Arity of Relations 62
3.4 Types of Semantic Relations 62
3.5 Part-Whole and Causal Relations Extraction 63
Chapter 4 - Part-Whole Relation Extraction 65
4.1 Introduction 66
4.2 Related Work 68
4.2.1 Part-Whole Relations 68
4.2.2 Motivation for Learning Domain-Specific Part-Whole Relations 68
4.2.3 Extracting Part-Whole Relations from Texts 69
4.2.4 Knowledge bases 70
4.3 Part-Whole Relation Extraction Challenges 70
4.4 Contributions 73
4.5 Framework for Domain-Specific Part-Whole Relation Extraction 74
4.5.1 Architecture Overview 74
4.5.2 Wikipedia as a Knowledge-Base 74
4.5.3 Pattern Induction and Formalization 76
4.5.4 Seed Selection 79
4.5.5 Pattern Selection 79
4.5.6 Instance Selection 80
4.5.7 Extracting Domain-Specific Part-Whole Relations 83
4.6 Experimental Evaluation 84
4.6.1 Corpora 84
4.6.2 Pattern Induction 84
4.6.3 Seed Selection 86
4.6.4 Pattern and Instance Selection 86
4.6.5 Extracting Domain-Specific Part-Whole Relations 87
4.6.6 Performance Measures 89
4.6.7 Comparison against Espresso 91
4.6.8 Reducing Semantic-Drift with Instance_Pair_Purity 92
4.7 Conclusion 93
Chapter 5 –Causal Relation Extraction 95
5.1 Introduction 96
5.2 Related Work 98

5.2.1 Cause-Effect Relations 98
5.2.2 Types of Causal Relations 98
5.2.3 Extracting Causal Relations from Texts 99
5.3 Causal Relation Extraction Challenges 100
5.4 Contributions 101
5.5 Framework for Domain-Specific Causal Relation Extraction 102
5.5.1 Pattern Acquisition 103
5.5.2 Causal Pattern Extraction 104
5.5.3 Causal Relation Extraction 105
5.6 Experimental Evaluations 106
5.6.1 Pattern Acquisition 106
5.6.2 Causal Pattern Extraction 106
5.6.3 Causal Relation Extraction 107
5.7 Conclusion 109
Chapter 6 –Effect of Seeds on Minimally-Supervised Algorithms 111
6.1 Introduction 112
6.2 Meronymic and Mereological Part-Whole Relations 113
6.3 Adopted Methodology for Investigating Effect of Seeds 114
6.3.1 Corpora 114
6.3.2 Information Extraction (IE) Algorithm 115
6.3.3 Seed Selection 115
6.4 Experiments and Evaluation 117
6.4.1 Precision of Extracted Results 117
6.4.2 Types of Extracted Relations 118
6.4.3 Distinct Patterns and Tuples 119
6.5 Conclusion 120
Part IV - Text clustering 121
Chapter 7 –Text Clustering 125
7.1 Introduction 126
7.2 Related Work 128
7.2.1 Clustering: Unsupervised Learning 128
7.2.2 Feature Identification 128
7.2.3 Vector Space Model, Similarity Measures and Centroid 130
7.2.4 Traditional Clustering Algorithms 131
7.2.5 Traditional Algorithms for Text Clustering 132
7.2.6 Frequent Wordsets for Text Clustering 133
7.3 Wordset-Based Clustering Challenges 135
7.4 Contributions 136
7.5 ClustText Framework for Text Clustering 138
7.5.1 Document Pre-processing 138
7.5.2 Feature Identification 139
7.5.3 Cluster Induction 141
7.5.4 Cluster Population 143
7.5.5 Cluster Taxonomy Formation 146

7.6 Experimental Evaluation 148
7.6.1 Feature Identification 148
7.6.2 Cluster Induction 150
7.6.3 Cluster Population 151
7.6.4 Evaluating ClustText's Accuracy and Selecting Threshold 151
7.6.5 ClustText vs. FIHC Baseline 156
7.6.6 Analyzing the Scalability 158
7.7 Conclusion 159
Part V - Conclusion 163
Chapter 8 – Summary and disussions 165
8.1 Background 165
8.2 Academic Objective 166
8.2.1 Research Question 1 (RQ1) - Term Extraction 166
8.2.2 Research Question 2 (RQ2) - Relation Extraction 167
8.2.3 Research Question 3 (RQ3) - Text Clustering 169
8.2.4 Further Discussions and Improvements 172
8.3 Industrial Objective 173
8.4 Future Directions 174
8.4.1 Effects of Seeds on Domain-Specific Part-Whole Relation Extraction 174
8.4.2 Causal Relation Extraction 174
8.4.3 Incorporating Semantic Information for Clustering 174
8.4.4 Multi-Lingual Information Extraction 174
8.4.5 Information Integration 175
8.4.6 Ontology Learning 175
8.4.7 Question-Answering Systems 175
Bibliography 177
Samenvatting 187

PART I - INTRODUCTION AND CONTEXT

PREAMBLE

Part I of this thesis lays the foundation of our work. It comprises of a single chapter, Chapter 1. This chapter introduces the research problem that will be addressed in the thesis, namely, the extraction of meaningful information from corporate texts.

We will provide an overview of the challenges posed by corporate (domain-specific) texts, and briefly describe the shortcomings of extant Natural Language Processing (NLP) techniques in extracting meaningful information from these types of texts. This lacuna in current NLP research and the inadequacies of current techniques serve as the motivation for our work. In this chapter, we will also specify our objectives in conducting this research and formulate the key research questions that we will address towards realizing the objectives.

CHAPTER 1 - INTRODUCTION

1.1 BACKGROUND

In the current era of digital information, electronic texts have become so pervasive that they have been completely assimilated in our daily lives. Our activities at work, for instance, are dictated by electronic-mail (e-mail) messages; we comment on the photos of our acquaintances in online social networks; we write reviews in customer forums to express our positive or negative experiences with products, and we consult these forums to gauge the opinions of other customers prior to our purchase.

This proliferation of electronic texts has also affected the corporate landscape. It has led to a drastic shift in the information space of companies, from one dominated by traditional sources of structured data, like data warehouses, to one in which unstructured texts are more prevalent. According to recent surveys [1, 2], unstructured texts constitute an overwhelming 80% of all corporate data, with the remaining 20% being accounted for by traditional structured data. These findings highlight two major implications for organizations. The first implication pertains to Business Intelligence (BI) analyses, which support (business) decision making. BI analyses are primarily designed to operate on structured data, for example, sales transactions. Thus, they have been confined to a relatively insignificant fraction (20%) of all corporate data. Unstructured texts, which are the largest asset (80%) of corporate information, have been overlooked. Consequently, extant BI analyses provide organizations with an incomplete and limited view of their operations, leading to less informed decision making. The second implication is that the huge repositories of corporate texts have opened up new business opportunities. Unstructured text data can be richer in information content than their structured counterparts, which are bounded by predefined data models such as database schemas. They precisely capture and describe the occurrences of events, such as product failures mentioned in customer complaints, as well as the situational contexts around these events, such as the factors leading to product malfunctions. Hence, buried within organizational documents are valuable information nuggets that are meaningful in a wide array of corporate activities. For example, customer complaint emails and repair notes of engineers contain pertinent information that can be exploited to enhance customer satisfaction, to improve product quality and brand image, and for new product development. These types of meaningful information represent actionable intelligence, which, if properly incorporated in BI analyses, provide organizations with a comprehensive view of their operations, and facilitate informed decision making for better corporate performance.

Companies have become increasingly aware of the value lying untapped in their sources of unstructured texts. However, the efficient management and exploitation of corporate texts is still a conundrum for most business organizations. The major obstacle lies in the identification and extraction of information that is meaningful for corporate usage from unstructured text data. This challenge is the main research problem that we investigate in this dissertation. The corresponding main research question that we address can be broadly formulated as *"how to extract meaningful information from large amounts of corporate documents"*.

The problem of identifying and extracting pertinent information items (nuggets) from natural language texts has been widely studied in the Natural Language Processing (NLP) community. Therefore, the numerous NLP techniques developed over the years represent a potential solution to our earlier mentioned problem.

However, in this thesis, we posit that corporate texts, especially those in the domain of Product Development-Customer Service (PD-CS), pose a set of unique challenges, which, to date, have been largely overlooked in current NLP research. As will be described and demonstrated experimentally in the later chapters, existing NLP techniques fail to adequately resolve these challenges, and they exhibit various shortcomings, which hinder their practical application in corporate domains. This argument, pertaining to the lacuna in

extant NLP research and to the inadequacies of current techniques for processing corporate documents, is at the crux of our work. It provides both the scientific and industrial underpinnings of our research, and serves as our motivation. Our core objective with this dissertation is therefore to fill the aforementioned gap in extant NLP research. We will achieve it by developing novel, state-of-the-art NLP techniques (algorithms, approaches), which successfully alleviate the shortcomings of existing ones, and overcome the intricacies involved in processing corporate texts. The techniques that we propose will enable organizations to unlock the valuable information that is hidden within their sources of unstructured texts. The uncovered information can then be used to support various types of corporate activities, as described before.

It is worth mentioning that the techniques we present, should in principle, be applicable to unstructured texts generated in other real-life disciplines, such as healthcare. It is reasonable to expect that these texts, such as prescriptions of surgeons and patient symptom descriptions, exhibit many similarities with the corporate documents that we treat in this thesis. However, given the scope of the project[1] within which our research was conducted, in this dissertation we will concentrate on corporate domains, and in particular, PD-CS.

1.2 MOTIVATION

To better illustrate our motivation and objective, and to position our contributions, we will briefly review some of the past and current approaches commonly employed for extracting meaningful information from (corporate) texts. We will focus on three approaches. They include:

1. Manual text analyses.
2. Information Retrieval (IR) techniques (e.g. search engines).
3. Natural Language Processing (NLP) techniques for Information Extraction (IE) and Text Clustering (TC). We focus on IE and TC since they are particularly relevant for corporate applications and are gradually finding their way into mainstream commercial off-the-shelf (COTS) BI suites. Our interest in these two applications will be explained in more details in Section 1.4.

Before proceeding any further, two points require some elaboration.

- Although we distinguish between IR and NLP, such a clear cut demarcation is not always reflected in practice. IR and NLP techniques are commonly employed in tandem. A typical example is for Question-Answering (QA) applications [3] . In QA, for instance, pertinent facts (e.g. events), extracted from documents by NLP algorithms for IE, can be used to supplement IR systems in order to improve their accuracy. It is also possible to couple IR with TC. For example, a TC algorithm can be used to group together similar search results returned by an IR system for a more effective presentation to the user. In addition, an IR system can be built on top of groups of similar documents, discovered by a TC algorithm. Such a cluster-based retrieval strategy improves the accuracy and the efficiency of the IR system [4]. In this dissertation, however, we refer to IR in its primal state, independent of NLP, where each document is simply encoded as a bag-of-words, with each word corresponding to an individual token. That is, no additional information, such as the syntactic dependencies between the words and pertinent facts extracted by NLP-IE algorithms, are taken into account. This enables us to clearly highlight the strengths of NLP approaches compared to the bag-of-word approaches.
- There are many approaches to IE and TC that are not grounded in NLP. These approaches treat the documents as bags-of-words, as described above. However, in this thesis, we focus on NLP algorithms for IE and TC, which are based on more sophisticated linguistic (and statistical) analyses, and enable a wider range of interesting applications for corporate usage.

[1] The DataFusion Project:www.iopdatafusion.org

We will examine the aforementioned three approaches according to two core characteristics, namely, their *analysis* of text data for locating meaningful information items, and their *organization (management)* of the identified information items to make them suitable for corporate use. In addition, the approaches will be presented chronologically; an ordering which is also reflected in their complexity. At this stage of the thesis, the reader can safely assume that *meaningful information* refers to pertinent items, such as events or instances[2] mentioned in a text collection, which are useful for corporate activities like BI. For example, meaningful information in engineers' repair notes, generated in the PD-CS domain, may denote product failures or customer complaints.

We will start by describing manual approaches and those based on Information Retrieval technologies in Sections 1.2.1 and 1.2.2 respectively. We show that they do not exhibit the desired characteristics for text *analysis* and information *organization* that warrant their corporate usage. Then, in Sections 1.2.3 and 1.2.4, we respectively describe NLP techniques for IE and TC, and illustrate their shortcomings in overcoming the challenges posed by corporate texts. As earlier stated, these shortcomings serve as the motivation underlying our work. They highlight the gap in current NLP research and stress the need for more innovative techniques, which we remedy in this dissertation. It must be noted that our intention in this section is neither to provide a comprehensive review of related NLP literature nor to describe the limitations of existing techniques at length. These will be discussed extensively in the later chapters.

1.2.1 MANUAL TEXT ANALYSES

Initial attempts for extracting and managing information from corporate texts were largely manual, and hence, extremely daunting. A Gartner report in 2003 estimated that knowledge workers, such as data analysts and managers, spent 40% of their time in *analyzing* documents to identify pertinent information from their contents, and in *organizing* them so that they are easily accessible and usable for corporate activities [1]. In that same year, the information deficit, which is the cost incurred in reworking information that is not found due to the improper analysis and organization of text data, was calculated to be at least US$6 million for an organization of 1000 employees [5]. Nearly a decade later, there are all reasons to believe that these figures should be revised upwards, given the prominence of unstructured text data. Another shortcoming of manual analyses is that they are subjective, often based on intuition or guess work [5], which compromises their reliability. These findings clearly illustrate that manual approaches are unsuitable for supporting corporate applications.

1.2.2 INFORMATION RETRIEVAL TECHNOLOGIES

The introduction of Enterprise Search Applications (ESA) allayed some of the shortcomings of manual approaches. ESA are built on the same Information Retrieval (IR) [6, 7] technology as Internet search engines like Yahoo! and Google. They have matured considerably over the years, with new capabilities such as proximity search and query expansion. IR technologies, such as ESA, are now a de-facto offering in most COTS BI suites, such as IBM Cognos [8].

IR systems *analyze* texts at the lexical (surface-word) level. To search for relevant information, users formulate queries, usually consisting of keywords, which express their information needs. Documents, which contain the keywords of the query string, are then returned as results. For instance, to search for product failures that are mentioned in a collection of engineers' repair notes, one may issue *"defective screen"* as query. All documents that mention the query string or its (inflectional) variants like *"defect screens"* will be returned as search results. This mechanism is based on two fundamental assumptions. First, it entails that users have some preconceived idea on what information

[2] We use instances in its general sense, and not according to its ontological sense, where it is defined as an object instantiated from a concept

they are looking for. Second and more importantly, it presupposes that they know how the relevant information items are expressed (written) in the documents. This is an essential prerequisite for formulating productive queries, which contain keywords that match the largest number of documents, and ensure the harvest of good quality results. We will see later that NLP techniques eschew the need for such assumptions.

The above illustration (searching for *"defective screen"*) highlights one of the characteristics of traditional IR systems, pertaining to their *analysis* of text data, which makes them unsuitable for many applications. Since most IR systems operate at the lexical level[3], their results are heavily dependent upon and constrained by the actual word forms occurring in the queries and in the documents. Documents are retrieved only if they contain the query string. All other relevant documents with similar or related information, which are also of interest to the user, will be unjustifiably omitted from the search results. This severely hampers the efficacy of IR systems for corporate tasks, which demand more comprehensive results, such as, "an *aggregated* list of *all* product failures", or more granular (detailed) results, such as, "*causes* of product failures". The first task of retrieving all product failures is impractical with current IR technologies. It requires that a user knows beforehand all the product failure instances mentioned in the documents, for example *"defective screen"* and *"leaking oil"*, which is highly unlikely. Then, individual queries must be issued for each instance, and the search results compiled into a single list. Formulating a query like *"list all product failures"* is futile. It only leads to the retrieval of documents that contain the words of the query string, such as "list" and "all". The second task of identifying causes of product failures poses even more challenges. Issuing the query *"causes of defective screen"* will lead to the retrieval of some relevant documents, such as *"blown fuse was the cause of defective screen"*, which contains the query words *"cause of"* and *"defective screen"*. However, these results do not fully satisfy the user's information needs. Other relevant documents, containing phrases that are lexically different but semantically similar to *"cause(s) of"*, such as *"responsible for"* and *"due to"*, will be incorrectly excluded from the search results. For instance, an IR application will fail to retrieve documents that mention *"low voltage was responsible for defective screen"* or *"defective screen was due to wrong antenna frequency"* in response to the above query. This difference between the surface forms of the query string and of the desired information in the documents is commonly known as the lexical gap or chasm [9, 10].

The next characteristic of IR systems that we discuss concerns their *organization* of search results. The result-set of an IR system is usually represented as a ranked-list of documents, sorted based on their relevancy (similarity) to the user-query. For instance, documents having more keywords in common with a query will be concentrated in the top section of the list. Such an organization of documents is convenient to users, enabling them to efficiently locate those documents that are of most interest to them. This organization is also easily amenable to basic functionalities, such as report generation. However, in most cases, directly consuming ranked lists of documents is difficult both for human users and for automatic applications. The ranked lists first have to be manually or automatically scoured in order to locate the relevant information items from their contents, which is a time-consuming and error-prone process. Only then can the identified items be extracted and used to supplement activities like BI.

The above discussion shows that traditional IR systems provide only a partial solution to the problem of extracting meaningful information from texts to support corporate activities. They do not fully satisfy the desired characteristics for advanced text *analysis* and information *organization* that are crucial for many corporate activities.

---

[3] As already mentioned, recent developments in IR, such as Question-Answering, also take into account deep linguistic information, such as syntactic information. Other studies have relied on techniques like Latent Semantic Indexing (LSI) to incorporate semantic features in IR systems. But we refer to the traditional bag-of-words approach to IR.

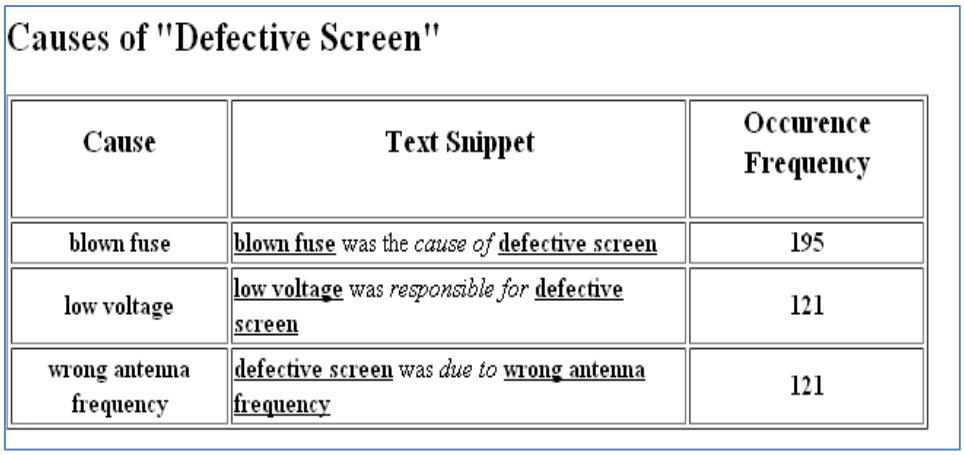
1.2.3 NATURAL LANGUAGE PROCESSING (NLP)

Natural Language Processing (NLP) techniques do not operate solely at the lexical level. Instead, they enable sophisticated *analyses* of natural language texts by also taking into account deeper linguistic features. Typical features include those derived from syntactic information, such as grammatical dependencies, and those based on semantic information, such as synonymy. Therefore, NLP techniques represent a potential solution for our problem of analyzing corporate texts and extracting meaningful information from their contents.

Two interesting applications of NLP in corporate activities are for Information Extraction (IE) and Text Clustering (TC), as will be briefly described below.

Information Extraction (IE)
Information Extraction (IE) algorithms automatically extract from a text collection all the meaningful information items that are semantically similar or related to a user query.

For example, a user searching for the most common causes of a product failure may issue the query *"most frequent causes of defective screen"* to an IE algorithm. Then, unlike IR, the IE algorithm will not merely return a list of documents in response to the user-query. Instead, it will identify and extract from the documents all the relevant information items, which are related to the user query, regardless of their lexical affinities with the query. That is, the search results will not be constrained by the query string *"most frequent causes of defective screen"*, as shown in Figure 1.1.



Causes of "Defective Screen"

Cause	Text Snippet	Occurence Frequency
blown fuse	blown fuse was the *cause of* defective screen	195
low voltage	low voltage was *responsible for* defective screen	121
wrong antenna frequency	defective screen was *due to* wrong antenna frequency	121

Fig. 1.1: Sample output of Information Extraction application.

Consequently, IE algorithms have higher coverage than conventional IR techniques, which only retrieve those documents containing the query string. In addition, IE algorithms do not require the formulation of individual queries to capture the distinct ways in which the relevant information items are expressed in the documents. Instead, a single query is sufficient to identify all the relevant (semantically similar/related) items, regardless of their different lexical manifestations. This mechanism also alleviates the difficulty of IR techniques, whereby specifying a productive query entails that the user anticipates exactly how the desired information is manifested in the documents. We will elaborate on the other benefits of IE techniques when discussing how they *organize* their information.

Text Clustering (TC)
Text Clustering (TC) algorithms automatically discover cohesive sub-groups of similar documents from a text collection. Each sub-group is known as a cluster.

For example, a user interested in finding the similarities between the repair actions performed by engineers can apply a TC algorithm to a collection of engineers' repair notes. The algorithm will then discover sub-groups of documents (i.e. engineers' notes) that describe similar repair actions. As an illustration, the TC algorithm may group together in a cluster the documents *"control board was at fault and replacement performed"* and *"replaced control board and tested"*. The rationale underlying such a clustering is that these documents both deal with the common topic of "replacement of control boards", and hence, can be considered as related. Similarly, the documents *"no display on monitor: adjusted the system"* and *"made adjustment to monitor to improve image quality"*, both dealing with the topic of "adjusting monitor (display)", will be clustered together. An illustration of clusters of documents that mention similar repair actions is depicted in Figure 1.2. Once these clusters have been discovered, users can then decide which one is most relevant to them or for a particular application. Obtaining such results is difficult with traditional IR, since as already mentioned, the user has to know beforehand the different repair actions that are described in the text collection and how they are realized in the documents. Then, individual queries must be issued for each repair action, and the similarity between the retrieved documents established in order to generate the desired clusters. We will elaborate on the other benefits of TC techniques when discussing how they *organize* their information.

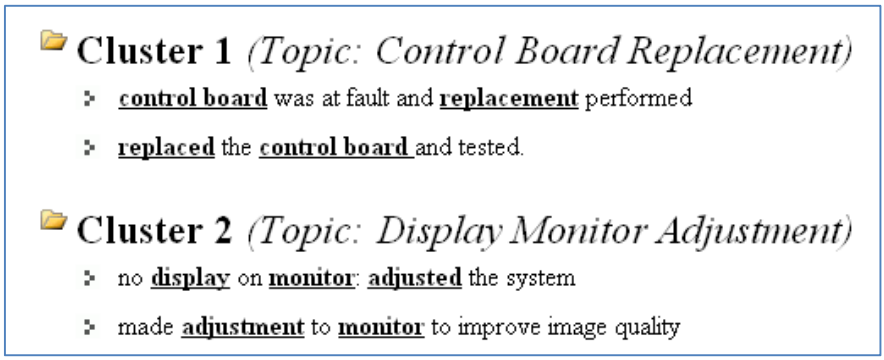


Fig. 1.2: Sample output of Text Clustering application.

Next, we examine how IE and TC *organize* their results. The outputs of IE and TC techniques can be organized and encoded into a wide variety of data structures and formats. Thus, they are more manageable and can be conveniently adapted to suit the requirements of applications like BI. This is unlike the relatively "rigid" ranked lists of documents output by IR systems, which, as discussed above, may prove unwieldy for corporate usage.

As shown in Figure 1.1, compared to IR, IE algorithms alleviate the need for users to manually and painstakingly browse through large document lists for identifying the contents of interest to them. Instead, IE algorithms automatically extract the relevant information items from the documents and present only these items to the users. In addition, the information extracted by IE algorithms can be straightforwardly canonicalized into structured formats, such as database or ontology triples. For example, the information item "defective screen was *due to* wrong antenna frequency", returned by an IE algorithm as shown in Figure 1.1, can be transformed into the triple <"due to", "defective screen", "wrong antenna frequency">, which consists of the pattern "due to" and of the event/instance pair "defective screen – wrong antenna frequency". These structured results can be stored in traditional BI repositories like data warehouses or they can be used for more sophisticated tasks like ontology learning. Thus, the structured format of these triples makes them directly amenable for consumption by different types of automatic applications.

In a similar fashion, the clusters of similar documents, output by TC algorithms, also exhibit a number of desirable features that make them attractive for corporate usage. For example, document clusters can be easily conceptualized into corresponding database tables, facilitating their exploitation in BI analyses. More importantly, the clusters can be labeled with their respective topics, as depicted in Figure 1.2, and arranged hierarchically into topic trees to enable *user browsing*. For instance, document clusters dealing with more specialized topics can be inserted as the child-nodes of other more general clusters. Users can then determine which of the clusters are of interest to them by traversing the topic tree and inspecting the clusters' labels. Subsequently, they can zoom into the relevant documents contained in the selected clusters or dive into deeper levels of the hierarchy to investigate more specialized topics. User browsing offers a multitude of advantages over traditional keyword searching. For example, browsing is more appropriate when users have no clear idea of what information they are looking for, or when they are unable to anticipate how the desired information is expressed in the documents. Also, the paths and other nodes (clusters) that are visited during browsing provide significant contextual information, which is not available with mere keyword searching [11]. The merits/demerits of user browsing versus keyword searching have been widely debated in the field of Human Computer Interaction. These issues are beyond the scope of our dissertation, and we refer the reader to the works of Olston and Chin [11] and of Katz and Byrnes [12] for a more comprehensive study.

The above discussion suggests that NLP algorithms exhibit the desired characteristics of advanced text *analysis* and information *organization* that corporate applications demand. They are thus promising solutions to our problem of extracting (and organizing) pertinent information from corporate texts. However, as will be next discussed, extant NLP algorithms, despite their sophistication, are unable to accurately process documents generated in corporate domains.

1.2.4 NLP Algorithms: State-of-the-Art and Challenges

Several NLP algorithms (e.g. for IE and TC) are mentioned in academic literature. They have been predominantly applied to documents in the general and in the scientific domains. This interest is usually attributed to a multitude of factors in these two domains that facilitate the computational treatment of texts.

One of the main factors is the availability of well-written texts. General and scientific documents, such as newspaper articles and bio-medical publications, are grammatically sound and well-written. They provide reliable linguistic evidence (features), which can be readily acquired from their contents using standard NLP tools, such as the Stanford syntactic parser [13] and the Specialist Minimal Commitment parser [14] for parsing newspaper and bio-medical articles. NLP algorithms leverage upon the acquired evidence, which enables them to accurately detect meaningful information from the documents. For example, NLP techniques for IE often rely on syntactic dependencies to identify associated/related events (e.g. causes and their effects) in texts.

Another important factor facilitating the application of NLP algorithms in the general and scientific domains is the availability of large document collections [15, 16]. These documents provide ample redundancy, whereby the same information is repeated multiple times in different places. Consequently, the algorithms are able to leverage upon the significant statistical evidence that can be computed from the documents' contents, and use this evidence to complement their linguistic features. As a result, their accuracy improves. For example, in a large text collection, the important information items will tend to appear recurrently. These items can be straightforwardly identified based on their high occurrence frequency. Also, the general and scientific domains provide a plethora of (electronic) knowledge resources, such as ontologies, which further facilitate the accurate detection of important information items from documents [17]. An example of a knowledge resource that describes the events, entities (e.g. "hiv", "aids") and their relations (e.g. "hiv causes aids") in the scientific domain is the Gene Ontology [18]. A similar resource for the general domain is the WordNet [19] lexico-semantic dictionary.

It can be deduced from the previous discussion that the majority of conventional NLP algorithms achieve high performance on large collections of well-written documents. As described above, these documents offer reliable statistical and linguistic evidence, which facilitates the accurate detection of meaningful information from their contents. The accuracy of these algorithms can be further improved by leveraging upon additional information from knowledge resources like ontologies. However, these desiderata, of large collections of well-written texts and the availability of knowledge resources, are often not satisfied in real-world applications. In particular, corporate domains, like PD-CS, are in stark contrast with the traditional disciplines that have been investigated in current NLP research. They introduce a new class of distinct challenges that extant NLP techniques fail to address satisfactorily.

A major challenge in corporate domains is the idiosyncratic language that typifies (corporate) documents, such as engineers' repair notes, customer complaints and meeting minutes [20, 21]. Corporate texts are usually intended for a specialized readership, consisting of domain experts. These documents are expressed in a domain-specific, often technical language, which differs considerably from the ordinary language of general documents like newspaper articles. In addition, these documents are created spontaneously, for instance, during product maintenance, and less care is taken towards ensuring their grammatical soundness [20, 21]. They are therefore rife with various types of linguistic inconsistencies (ungrammaticalities), and are difficult to analyze for the acquisition of valid linguistic evidence. Subsequently, NLP algorithms, which heavily rely upon these features, are unable to accurately extract meaningful information items from these corporate documents.

Another challenge posed by corporate texts is the data sparsity issue [22, 23]. These sparse documents do not provide sufficient statistical evidence, which could have been exploited to compensate for the lack of reliable linguistic features. The absence of statistical information further impedes the detection of pertinent information items from the documents. For example, most NLP algorithms are not accurate for detecting important information items (e.g. key terms) that are mentioned sparsely (once or twice) in a text collection. This limitation can have serious implications in practical applications. For instance, these rare items may correspond to sporadic occurrences of life-threatening product failures. If left undetected, these failures may recur in the future, with drastic consequences on the customer satisfaction and brand image of a company. Also, in most corporate domains, knowledge resources, like domain-specific ontologies, for supporting NLP algorithms are scarce. Available resources are either outdated, for example, they do not reflect recent additions to product lines, or are not amenable to automatic analysis, for example, they reside on legacy systems and cannot be easily accessed.

The preceding discussion suggests that current NLP algorithms (e.g. for IE and TC) do not adequately overcome the challenges involved in processing corporate texts. We will elaborate on these challenges and empirically demonstrate them in the later chapters of this thesis, namely Chapters 2, 4, 5 and 7. Because of these difficulties, the extraction of meaningful information from corporate documents has remained a largely unsolved problem. Many business organizations are still struggling to capitalize on the valuable contents lying untapped in their sources of unstructured texts. Such a state-of-affairs highlights the urgent need for innovative NLP algorithms that successfully resolve the intricacies posed by texts generated in corporate domains. This need is made even more pressing as unstructured texts have been predicted to become the predominant source of data warehouse feeds in the near future [2].

1.3 PROBLEM STATEMENT, OBJECTIVE AND CONTRIBUTION

In this thesis, we will investigate the aforementioned problem of extracting meaningful corporate information from (corporate) documents. The main research question (RQ) that we address can be articulated as

RQ: "How to automatically extract meaningful information from sparse, domain-specific and informally-written corporate texts?"

Addressing this research question has both academic/scientific and industrial/corporate ramifications. Consequently, our objective with this research will be two fold, encompassing both an academic and an industrial aspect. Our academic objective is to fill the lacuna in extant NLP research, which was described in the previous section and will be elaborated in the later chapters. We will realize this objective by developing novel, state-of-the-art NLP techniques (algorithms, approaches) for Information Extraction (IE) and Text Clustering (TC), which successfully alleviate the limitations of existing ones and overcome the challenges in discovering *meaningful information* items from corporate texts. The techniques that we develop will therefore correspond to our core academic contributions.

Our industrial objective is to enable business organizations to better exploit their otherwise underutilized textual assets. This will be achieved by providing selected industrial partners[4] with our techniques and corresponding prototypes, which they can use to uncover the valuable information hidden within their massive amounts of unstructured texts. In this way, our industrial contribution extends to the emerging field of business text analytics. This latter contribution is particularly promising and timely given that text analytics software are "poised to become an integral part of the organizational IT portfolio in a near future" [24]. It should be noted that given its scientific nature, this dissertation covers mostly the academic aspect of our work. The implementation aspects, such as the technicalities concerning the development of our techniques into prototypes and their deployment for use by our industrial partners are beyond the scope of our research. These issues will not be covered.

Before proceeding any further, we will specify what we mean by *meaningful information* in Section 1.4. Such a specification enables us to decompose our main RQ into a number of tasks. Each task is a step to be executed in order to reach our objective of extracting meaningful information from corporate texts. These different tasks (steps), in turn, entail a set of more specific research questions, which we will address by developing suitable NLP techniques, as will be described in the later chapters. Together, the solutions to each of the specific questions constitute the answer to our main RQ. Viewed in this light, the different tasks define the individual steps of our (overall) adopted methodology, which we will revisit in section 1.5.

[4] http://is.ieis.tue.nl/research/processmining/datafusion/doku.php?id=partners

1.4 MEANINGFUL INFORMATION

Meaningful information is ambiguous, and lends itself to manifold interpretations. What constitutes meaningful information for someone may be completely meaningless for somebody else.

Given the backdrop of this thesis, as described in the earlier sections, we define meaningful from a linguistic (NLP) perspective and from an industrial (corporate) perspective. The linguistics perspective is concerned with symbols, such as words, which have a *special function in a language*. Compared to other ordinary symbols, they are information bearing units, which represent important events, instances and their relations in texts. The industrial perspective is concerned with the usefulness of these symbols. They must be relevant for *serving the purpose of corporate activities*, such as BI. Our definition along these two dimensions closely resembles that of the Merriam-Webster dictionary [25], according to which, "to be meaningful" is "1) to have a *special function in a language system*" and 2) "to *serve a purpose"*.

However, if we aim at the automatic extraction of meaningful information from texts, we need a more formal definition that can be computationally implemented. To illustrate our computational model of "meaningful information", we take as example the text below, which narrates the repair actions of an engineer, and we consider the common PD-CS activity of investigating product failures.

> Arrived at hospital yesterday following call of customer Jane. Defective *proximity sensor switch* found to be the cause of blinking *monitor.*
> *Proximity sensor switch* located in *magnetic resonance circuit:* may require dismantling

1.4.1 TERMS

Some of the words (or word sequences) in the above snippet appear to be more meaningful than others for investigating product failures. They are shown italicized. In linguistics, these words are known as terms. Terms, as opposed to general words, have a *special function in a language*. They are created to denote specific domain concepts. For instance, unlike the general word *"yesterday"*, the term "*(defective) proximity sensor switch*" designates a particular piece of hardware in the PD-CS domain. From an industrial perspective, terms are *useful* in a wide range of corporate activities. For example, the terms in engineers' repair notes may represent malfunctioning products that fail recurrently. Once these terms have been identified, remedial measures can be initiated to improve product quality. Therefore, we regard terms as the most basic component in our definition of meaningful information. Our objective of extracting meaningful corporate information will thus entail the (automatic) extraction of domain-specific terms from (corporate) documents.

1.4.2 SEMANTIC RELATIONS BETWEEN TERMS

Terms in isolation are not very informative and meaningful. The term "*(defective)proximity sensor switch*", in our earlier illustration, only tells us that it is a malfunctioning product that was operated upon by an engineer. Corporate activities demand richer and more comprehensive information, such as the factors responsible for product failures, the events triggered by malfunctioning product components, or the location of these components in a product.

To acquire additional information about terms, we have to inspect their contexts. The phrase *"(found to be the) cause of"*, for instance, occurs in the context of the term "*(defective) proximity sensor switch*", and relates it to "(*blinking) monitor*". This phrase tells us that the *"defective proximity sensor switch"* was responsible for the *"blinking monitor"*. In this case, it establishes a cause-effect relation between these two terms. Similarly, the phrase *"located in"* establishes a part-whole relation between the two terms *"proximity sensor switch"* and *"magnetic resonance circuit"*, indicating that the former is a

part of the latter. These types of binary relations between pairs of terms are known as semantic relations [26].

Semantic relations enable us to learn more about terms and associated events and instances. Hence, they are of greater relevance for corporate activities. Cause-effect relations, for example, are useful for determining, and subsequently fixing, the causes of product failures. Hence, we also consider semantic relations as another component in our definition of meaningful information. Our objective of extracting meaningful corporate information will thus also entail the extraction of semantic relations from documents.

At this juncture, three points are worth mentioning. First, since we consider terms and semantic relations as lexical manifestations of meaningful information, our work is grounded in *lexical semantics*. Second, in NLP, the tasks of term extraction and of relation extraction are commonly referred to as Information Extraction (IE). Third, our representation of context differs slightly from that of traditional *distributional semantics*, which defines the context as the words preceding and/or succeeding a term. Our "context" is defined as the phrase connecting a pair of terms.

1.4.3 Relations between Documents/Document Clustering

Relations can also exist between units larger than terms. The text fragment below, for example, bears many affinities with our previous snippet.

> Erratic table rotation was due to damaged *proximity sensor switch*; table had to be dismantled.

Both documents mention *"dismantling"* as a repair action, and *"proximity sensor switch"* as the cause of an event. These similarities establish a relation between the documents. Unlike relations between terms, which can be expressed by their connective phrases, we represent the relations between documents by grouping them into clusters. A cluster is therefore a collection of related documents, which for example, deal with similar topics [27, 28].

Document clustering is central in a wide range of corporate activities. For example, clusters of documents that describe similar repair actions of engineers are useful for investigating whether safety guidelines have been followed, or for determining problems that require similar repairs. Therefore, our objective of extracting meaningful corporate information will also involve the extraction (discovery) of document clusters.

1.5 STEPS IN THE QUEST FOR MEANINGFUL INFORMATION

In the preceding sections, we decomposed our main RQ into three major components and corresponding tasks, viz. term extraction, relation extraction, and text (document) clustering. In fact, such a decomposition was intentional. These tasks are defined and presented in such a way that they form a logical/natural sequence of actions towards our goal of discovering meaningful information from corporate texts. In this way, they determine the overall methodology that we adopt in this thesis, as depicted in Figure 1.3. Each task, shown in Figure 1.3, involves more detailed operations, such as part-of-speech tagging, syntactic parsing and measuring the association strength between words. These operations are not depicted for conciseness. They will be discussed in the corresponding chapters for each of the tasks. Also, as shown in the illustration, our 3 main tasks form a coherent framework (application pipeline) consisting of distinct, yet inter-dependent modules, which business organizations can adopt (implement) in their endeavors to better exploit their sources of unstructured texts.

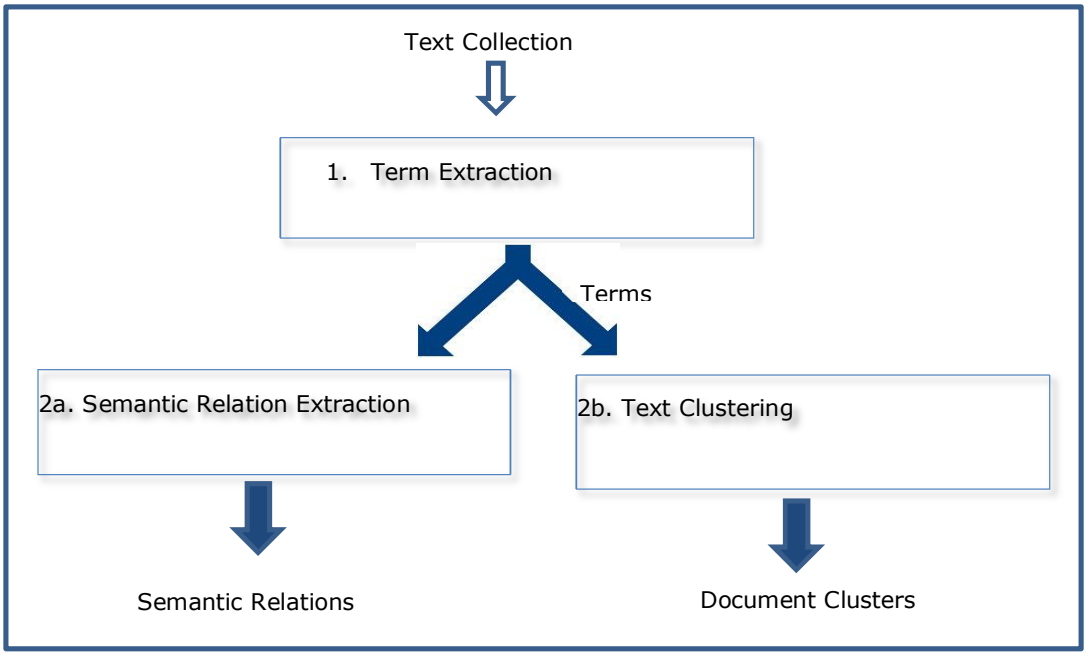


Fig. 1.3: Application pipeline depicting our adopted methodology in the quest for meaning.

In Figure 1.3, the numbering accompanying the different tasks corresponds to the order in which they are executed. The labels of the filled arrows indicate the outputs, i.e. results, of each step. Given a collection of corporate texts, such as engineers' repair notes, we start with Term Extraction, which is considered as the most fundamental task in NLP applications. The identified terms are used as the basis for the more sophisticated activities of Relation Extraction and Text Clustering. As can be seen from our framework, Relation Extraction and Text Clustering are two independent activities, in the sense that the output of one does not constitute the input of the other. Not shown in the above pipeline is the basic, but no less important, task of document pre-processing (data cleaning). This phase transforms the input documents into a format more amenable to the automatic analyses performed during term extraction, relation extraction and text clustering. We will describe our pre-processing task when discussing Term Extraction in Chapter 2.

To develop and implement our NLP algorithms for the aforementioned tasks, we will adopt mostly an *empiricist* approach, also known as *statistical NLP*. The basic principle of such an approach is to determine the most typical patterns (e.g. terms) that characterize the *use* of language in a text collection or corpus. The primary apparatus to identify these patterns is their occurrence frequency [29, 30]. For example, one can count the number of times a

particular term or word category, such as nouns or adjectives, occurs in a corpus. This information can then be used to identify terms, detect semantic relations and discover document clusters. Our primary motivation in adopting an empiricist approach is that we are interested in analyzing the language that *has been used* in corporate texts in order to find out what has been said on events like product failures or repair actions. We consider an empiricist approach to be more suitable for such an endeavor since it studies language as *it has been used*. It is also the predominant approach employed in text analyses [31]. For more details on the empiricist approach we refer the reader to the work of Manning and Schutze [31]. Details on other approaches in linguistic, such as the rationalist approach, can be found in the work of Chomsky [32], and the recently published article (April 2011) of Dunn et al. in Nature [33]. A detailed description and comparison of these approaches is out of the scope of this dissertation.

To validate our proposed techniques, we will evaluate their performance on real-life corporate documents that are provided by our industrial partners. This will be achieved by using the classical evaluation metrics traditionally employed in NLP and related fields like IR and Data Mining. These metrics include the *precision*, *recall* and *F1* scores for estimating the performance of our Term Extraction (Chapter 2) and Relation Extraction (Chapters 4, 5, 6) techniques. Precision gives the fraction of correctly extracted terms/relations, while recall is the fraction of correct terms/relations extracted. The F1 score is defined as the (weighted) harmonic mean between precision and recall, and penalizes large divergence between them. Thus, it can be thought of as the balance between precision and recall. These metrics of precision, recall and F1 scores can also be used for evaluating the performance of our Text Clustering technique. In this case, the precision of a cluster, with respect to a predefined class, is the fraction of documents in the cluster that belongs to the class (e.g. they deal with similar topics). The recall of the cluster with respect to a predefined class is the fraction of documents of that class that can be found in the cluster. Another widely employed metric for evaluating cluster quality is the *purity*. It is calculated as the largest fraction of documents in a cluster that belongs to the same class. However, in our clustering experiments (to be reported in Chapter 7), we will show that these metrics are unsuitable for our application. Hence, for evaluating the quality of our clusters, we will define a new metric, which is quite similar to the purity.

As will be elaborated in the later chapters, the evaluation results reveal that our techniques outperform other state-of-the-art NLP algorithms for term and relation extraction and text clustering. This indicates that they successfully overcome the challenges posed by the extraction of meaningful information from corporate documents. Thus, they represent a viable solution that can be used by business organizations to unlock and leverage upon the value lying untapped in their textual assets. We will next define the research questions that we address in this thesis.

1.6 SPECIFIC RESEARCH QUESTIONS

As already mentioned, our three different tasks of Term Extraction, Relation Extraction and Text Clustering each entail a more specific research question, which we formulate below. As can be seen, a central theme in our questions is the sparsity, informal language constructs and domain-specificity that characterize our documents.

The first question, pertaining to Term Extraction, is:
RQ1: How to accurately identify and extract terms from domain-specific, sparse and informally-written corporate texts?

The second question, pertaining to Relation Extraction, is:
RQ2: How to accurately detect occurrences of semantic relations from domain-specific, sparse and informally-written corporate texts?

The third question, pertaining Text Clustering, is:
RQ3: How to discover cohesive and homogeneous clusters of similar documents from a collection of domain-specific, sparse and informally-written corporate texts?

1.7 THESIS OVERVIEW AND PUBLICATIONS INCLUDED

This thesis is organized as follows:

- Part I – Introduction and Context
  - Chapter 1 (this chapter)

- Part II – Terms
  - Chapter 2: presents our proposed approach for identifying terms from domain-specific texts. In this chapter, we will answer our first question, RQ1.

- Publications for Part II
  - This chapter is an extended version of the manuscript "Ittoo, A; Maruster, L; Wortmann, J.C; Bouma, G.: Textractor: A Framework for Extracting Relevant Domain Concepts from Irregular Corporate Textual Datasets". It was published as a book chapter in *Lectures Notes in Business Information Systems, vol. 47, pp.71-82.*
  - It is also a pre-print of a manuscript to be submitted to the journal *Computer Speech and Language*.

- Part III – Semantic Relations
  - Chapter 3: provides an overview of semantic relations, and motivates our interest in part-whole and causal relations.

  - Chapter 4: presents our proposed approach for extracting part-whole relations from domain-specific texts.

  - Chapter 5: presents our proposed approach for extracting causal relations from domain-specific texts.

    In Chapters 4 and 5, we will answer our second question, RQ2.

  - Chapter 6: is a detailed study investigating the behavior of our proposed technique for relation extraction.

- Publications for Part III
  - Chapter 4 corresponds to the journal article "Ittoo A; Bouma G.: Minimally-Supervised Extraction of Domain-Specific Part-Whole Relations using Wikipedia as Knowledge-Base". It has been conditionally accepted for publication in the journal *Data and Knowledge Engineering, Special Issue NLDB2010*.
  - A preliminary version was also published as the book chapter "Ittoo, A; Bouma G; Maruster L; Wortmann J.C.: Extracting Meronymy Relationships from Domain-Specific, Textual Corporate Databases", in *Lecture Notes in Computer Science, vol. 6177, pp. 48-59.*

  - Chapter 5 has been published as the book chapter "Ittoo A; Bouma, G.: Extracting Explicit and Implicit Causal Relations from Sparse, Domain-Specific Texts", in *Lecture Notes in Computer Science, vol. 6716, pp.52-63*

  - Chapter 6 was published as "Ittoo A; Bouma G.: On learning subtypes of the part-whole relation: do not mix your seeds", in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 1328--1336.*
  - An earlier version appeared as the book chapter "Ittoo, A.; Bouma, G.: Semantic Selectional Restrictions for Disambiguating Meronymy Relations", in *Computational Linguistics in the Netherlands, pp. 83-98, LOT Occasional Series.*

- Part IV – Text Clustering
  - Chapter 7: describes our proposed technique for clustering documents from domain-specific corpora. In this chapter, we will answer our third question, RQ3.

- Publications for Part IV
  - Chapter 7 is an extended version of the publication "Ittoo, A.; Maruster, L. Ensemble Similarity Measures for Clustering Terms", *in Proceedings of World Congress in Computer Science and Information Engineering, vol. 4, pp. 315-319, IEEE Computer Society.*
  - It is also a pre-print of a manuscript to be submitted to the journal *"Expert Systems with Applications".*

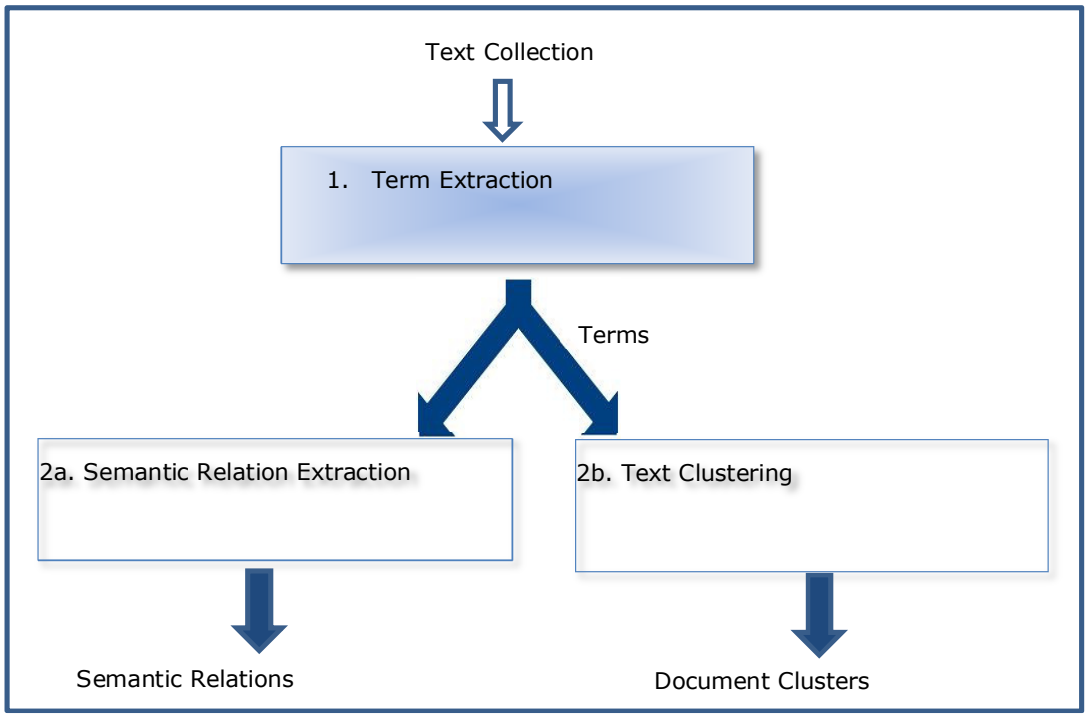
- Part V - Conclusion
  - Chapter 8: summarizes the results of this thesis in light of the research questions and academic/industrial objectives. It also highlights possible improvements and avenues for further research.

PART II - TERMS

PREAMBLE

In Part II of the thesis, we will address the first research question *RQ1: How to accurately identify and extract terms from domain-specific, sparse and informally-written corporate texts?*

As mentioned earlier, this corresponds to the first task in our application pipeline, shown below, which we need to execute for realizing our overall aim of extracting meaningful information from corporate texts. Our proposed solution in response to RQ1 will be presented in Chapter 2.



CHAPTER 2 - TERM EXTRACTION

CHAPTER SUMMARY

In this chapter, we will develop and present a novel term extraction algorithm, ExtTerm, in order to answer our first research question. Although several term extraction techniques have been proposed, they exhibit several shortcomings when used for domain-specific texts, hindering their practical applications. A major difficulty, due to the issue of data sparsity, is the detection of low frequency terms. Also, most existing techniques are not suitable for detecting terms containing more than 2 words, which are prominently used in specialized domains like PD-CS. Furthermore, especially when dealing with informally-written documents, many irrelevant or ungrammatical expressions may be incorrectly selected as terms on account of their high frequencies in these types of documents.

ExtTerm successfully addresses these intricacies. To detect rare terms, ExtTerm contrasts their statistical distributions across the domain-specific corpus and a collection of general texts. Expressions that are found to be more relevant to the domain-specific corpus are selected as terms, regardless of their absolute frequencies. We will also present a technique that replicates the mechanism underlying the formation terms, which enables us to detect terms of arbitrary lengths, including those with more than 2 words. This technique is also useful for discriminating between valid multi-word terms and invalid word sequences.

As will be shown in our experiments, ExtTerm outperforms a state-of-the-art baseline, and extracts more accurate terms from a real-life corpus of domain-specific texts. We will also demonstrate that ExtTerm is equally precise in extracting terms regardless of their lengths (i.e. including very long terms) and of their frequencies (i.e. including terms of low frequencies).

The work described in this chapter is an extended version of the earlier published book chapter:

- Ittoo, A; Maruster, L; Wortmann, J.C; Bouma, G.: Textractor: A Framework for Extracting Relevant Domain Concepts from Irregular Corporate Textual Datasets", in *Lectures Notes in Business Information Systems, Vol. 47, pp.71-82, Springer.*

It is also a pre-print of a manuscript to be submitted to the journal *Computer Speech and Language*

In this chapter, we have expanded the earlier published work by performing additional experiments in the Experimental Evaluation section, and by providing more background information in the Related Work section.

2.1 INTRODUCTION

The most fundamental step to realize our goal of extracting meaningful information from corporate documents is the identification of domain-specific terms from their contents. Terms are information bearing linguistic units, which are crucial in a number of corporate activities. For example, terms extracted from engineers' repair notes may correspond to malfunctioning products, which are the source of customer dissatisfaction and complaints. Once these terms have been identified, appropriate remedial measures, geared at improving product quality and enhancing customer satisfaction, can be implemented. Terms are also useful for activities like populating dictionaries [34] and learning ontologies [35, 36]. In the later chapters of this thesis, namely Chapters 4, 5 and 7, we will use terms in order to detect semantic relations and to discover groups of similar documents, which are even more meaningful information items.

Many NLP techniques for term extraction exist [37, 38]. To accurately identify terms, most of these techniques leverage upon sound statistical and linguistic evidence that can be acquired from large amounts of well-written texts, such as collections of newspaper articles in the general domain or bio-medical documents in the scientific domain. They also rely on additional sources of domain-knowledge, such as dictionaries like WordNet [19] for the general domain and ontologies like the Gene Ontology [18] for the scientific domain, to complement their statistical and linguistic analyses. This improves their performance in accurately detecting terms from texts.

However, current techniques have not adequately addressed the challenges in extracting terms from documents generated in corporate disciplines like PD-CS. These documents exhibit certain peculiarities that severely compromise the performance of extant term extraction techniques, hindering their widespread application in real-life, corporate domains. The challenges include:

1. Data sparsity. Documents generated in corporate domains are sparse [22, 39, 40, 41, 42]. They do not provide sufficient redundancy, which hinders the acquisition of reliable statistical evidence for detecting terms. For example, a significant portion of the terms in our corpus had low frequencies (e.g. occurring less than 5 times). Detecting these terms based solely on their (low) frequencies is difficult.

2. Incoherent language. Corporate texts are written in an informal style, and do not provide reliable linguistic evidence, which is useful to detect terms [20, 21]. For example, detecting the term *"regulating switch"* from the ambiguous construct "device is regulating switch" is difficult since "regulating" is often misinterpreted as a progressive verb, and thus, not considered as a part of the term *"regulating switch"*.

3. Multi-word terms. Current term extraction algorithms are inadequate to identify long, complex terms [38, 43, 44], such as *"radiation protector shield arm"*, which are rife in corporate domains like PD-CS. This difficulty is compounded by the absence of reliable statistical and linguistic evidence.

4. Frequently used irrelevant or invalid constructs[5]. Irrelevant words like *"meeting room"* and invalid constructs like *"customer helpdesk collimator shutter"* tend to occur more frequently than valid terms like *"radiation protector shield arm"* in informally-written documents. Precisely discriminating the valid terms from the other non-terminological expressions is challenging [41].

5. Lack of domain-specific knowledge resources. Additional sources of domain knowledge to support term extraction algorithms are unavailable in most

[5] An invalid expression is incoherent (e.g. "customer helpdesk collimator shutter"), while an irrelevant expression may be coherent but not meaningful in the given domain (e.g. "meeting room").

specialized domains [45, 46] (with the exception of the scientific domain of bio-medicine).

Challenges 1 and 5 can be considered as longstanding issues in term extraction research, while Challenges 2, 3 and 4 are peculiar to the texts generated in specialized domains like PD-CS.

To overcome these challenges and answer our first research question, we develop and present our ExtTerm framework for term extraction. ExtTerm presents a number of innovative aspects that enables it to successfully alleviate the limitations of existing algorithms, and to accurately extract terms from domain-specific, sparse and informally-written documents

Our core contributions with ExtTerm, in response to the aforementioned challenges, are novel techniques based on sophisticated linguistic and statistical analyses for:

1. Overcoming the issue of data sparsity and accurately detecting terms in the absence of significant statistical evidence (Contribution 1 in response to Challenge 1).

2. Robustly identifying terms from informal texts, which do not provide reliable linguistic evidence (Contribution 2).

3. Precisely identifying complex terms of arbitrary length (Contribution 3).

4. Accurately discriminating valid terms from invalid word sequences (Contribution 4).

5. Accurately detecting terms in an unsupervised fashion, eschewing the need for additional sources of domain knowledge (Contribution 5).

We will elaborate on the aforementioned challenges and on our contributions in Sections 2.3 and 2.4.

To evaluate our ExtTerm framework, we estimated its performance in extracting terms from real-life corporate documents provided by our industrial partners. We also compared ExtTerm's performance against a state-of-the-art term extraction system, which we used as baseline. Our experimental results reveal that ExtTerm significantly outperformed the baseline, and accurately extracted terms from our corpus. These results suggest that ExtTerm successfully overcomes the challenges posed by term extraction from domain-specific, sparse and informally-written texts. Thus, ExtTerm indeed provides a solution to our first research question.

This chapter in organize as follows. In Section 2.2, we present some related work on terminology and term extraction. Section 2.3 describes the limitations of existing techniques for extracting terms from domain-specific documents. We present our contributions in response to these challenges in Section 2.4. Our proposed ExtTerm framework is presented in Section 2.5, and its performance is evaluated in Section 2.6, before concluding in Section 2.7.

2.2 RELATED WORK

2.2.1 TERMS

A term is formally defined by the ISO-704 standard [47] as a "designation consisting of one or more words, representing a *concept* in a *special language*".

A *concept* is a mental representation (perception, abstraction or conceptualization) of objects in a specialized domain or field [47].

A *special language* refers to the functional language employed in specialized disciplines to communicate domain-specific knowledge. In the corporate domain of PD-CS, for example, information on products and customers is conveyed in a language that is distinctive of that discipline. Similarly, in the scientific domain of bio-medicine, information on diseases and

symptoms is communicated in a specific bio-medical language. Special languages are derived from universal (general) languages [48]. Zellig Harris, in his work on the mathematical theory of language [49, 50], considers special languages as a subset of general languages, and refers to them as sub-languages. Since sub-languages are used for restricted communications within specialized domains, they exhibit various peculiarities that distinguish them from their general counterparts. For instance, the terse and often ungrammatical language employed by engineers for narrating their repair actions differs considerably from the standard language used in general documents like newspaper articles.

Thus, terms are also commonly defined as lexical manifestations used in a sub-language for denoting domain-specific concepts. For example, *"sensor button"* designates a particular type of product component (concept) in the PD-CS domain, and hence, is a domain-specific term.

2.2.2 SIMPLE AND COMPLEX TERMS

Terms can be classified as simple or complex depending on the number of words they contain. Simple terms are those expressed using a single word, and hence, they are also known as single-word terms. An example of a simple term in the PD-CS domain is t_1 = *"sensor"*. Complex terms consist of two or more words, and they are also commonly known as multi-word terms. Examples of complex PD-CS terms include t_2 = *"sensor button"*, t_3=*"rotation sensor button"*, t_4= *"plate rotation sensor button"*, t_5=*"floor plate rotation sensor button"*, t_6=*"rear floor plate rotation sensor button"*, and t_7=*"rotation sensor button for rear floor plate"*. The length (number of words) of these complex terms is respectively 2, 3, 4, 5, 6, and 7. We will refer to a term with n words as an n-word term. For instance, t_3 is a 3-word term.

2.2.3 BASE-TERMS AND TERM FORMATION

To investigate the prototypical syntactic structure of terms, Daille et al. [51], analyzed a large amount of human-generated terminological data. They found out that the basic structure of English terms could be represented by patterns of nouns (N) and adjectives (A) of the form [N] (e.g. *"sensor"*), [N N] (e.g. *"sensor button"*) or [A N] (e.g. *"rear shutter"*). They referred to these fundamental units of one or two words as *base-terms*.

Their study also revealed that base-terms can be composed into longer terms by the applications of lexical operations such as overcomposition by juxtaposition or insertion of prepositional modifiers. For example, the term t_3=*"rotation sensor button"*, defined by the syntactic structure [N N N], is formed from the base-term t_2 = *"sensor button"*, with structure [N N], by a juxtaposition operation, which prepends the noun *"rotation"* to t_2. This yields *"rotation [sensor button]*$_{base\text{-}term}$*"*, with the structure [N N N]. Similarly, longer complex terms, such as t_4, t_5, t_6 and t_7, described previously, can be created from base-terms, such as t_2, by the recursive applications of lexico-syntactic operations. For example, the term t_6=*"rear floor plate rotation sensor button"* is formed by successively juxtaposing the base-term t_2 = *"sensor button"* ([N N]) with the nouns ([N]) *"rotation"*, *"plate"* and *"floor"* , and with the adjective ([A]) *"rear"* . This yields =*"rear floor plate rotation [sensor button]* $_{base\text{-}term}$*"*, whose structure is defined by the pattern [A N N N N N]. In addition, prepositions can also be "inserted" to connect shorter terms into longer ones. This is illustrated with the term t_7=*"rotation sensor button for rear floor plate"*. It is created by juxtaposing the two terms *"rotation sensor button"* ([N N N]) and *"rear floor plate"* ([A N N]), and inserting the preposition ([P]) *"of"* to connect them, yielding *"rotation sensor button for rear floor plate"*, with syntactic structure [N N N P A N N].

We will exploit the recursive mechanism underlying term formation, presented in this section, in our technique for identifying complex terms of arbitrary lengths and for discriminating them from invalid word sequences. This will be presented in Section 2.5.4.

2.2.4 (DOMAIN-SPECIFIC) TERMS VS. (GENERAL) WORDS

There exists an important, albeit subtle, difference between (domain-specific) terms and (general) words. At the surface level, terms and words are both lexical expressions, and can be considered identical to each other. For example, the PD-CS term *"sensor button"* and the collocation of general words *"meeting room"* are both lexically manifested as

character strings, inclusive of a whitespace. Furthermore, as shown in past studies in terminology, terms tend to adopt all the word formation rules in a language [52, 53]. As an illustration, the PD-CS term *"sensor button"* and the sequence of general words *"meeting room"* are both formed by the juxtaposition operation. The term *"sensor button"* is composed by juxtaposing the base-term *"button"* with *"sensor",* while *"meeting room"* is created by juxtaposing the words *"meeting"* and *"room"*

To depict the main difference between terms and words, we have to consider the meaning they convey in a particular domain. As we discussed above, *"sensor button"*, just like *"meeting room"*, are both sequences of words. However, the distinction between them is that *"sensor button"* is ascribed an idiosyncratic meaning in the PD-CS domain. It is used to designate a specific concept, corresponding to a type of product component. Conversely, *"meeting room"* is a general collocation, devoid of any specific meaning in PD-CS. Hence, it is not treated as a domain-specific PD-CS term. Thus, terms and words have completely different semantics in a domain.

Therefore, to achieve our goal of extracting meaningful information from corporate texts, it is imperative that we properly distinguish domain-specific terms from general words (sequences). However, accurately recognizing terms from words is a major challenge in terminology [41, 53]. The main impediment lies in the surface level affinities that exist between them. This difficulty is compounded in corporate domains, like PD-CS, and in other specialized disciplines, by the absence of additional sources of knowledge. These resources, such as ontologies, provide useful semantic information that could have facilitated term identification. Hence, we will have to rely on other features or properties to detect terms from our documents.

Two fundamental properties, which are useful for determining whether an expression[6] qualifies as a term candidate[7], are the *unithood* and *termhood* [54, 55].

2.2.5 UNITHOOD

Unithood determines whether an expression is well-formed and operates as a coherent atomic unit [55]. There are thus are two aspects to unithood.

The first aspect concerns the syntactic structure. An expression qualifies as a term candidate only if it adheres to certain syntactic criteria (i.e. it is well-formed). For example, expressions with nouns and adjectives, such as the base-terms *"sensor"* and *"rear shutter"*, are considered to be well-formed as terms. In most cases, the syntactic structure adopted by terms is that of noun phrases.

The second aspect of unithood indicates whether the expression behaves as an atomic and coherent linguistic unit. For example, *"sensor button"* is an atomic (and coherent) unit. Its constituent words, *"sensor"* and *"button"*, are strongly collocated (correlated) since they co-occur together more often than spuriously. Such an expression is likely to qualify as a term candidate. Conversely, *"customer amplification"* is not an atomic unit since the words *"customer"* and *"amplification"* rarely co-occur together. Such an expression, with weakly associated words, is unlikely to correspond to a term candidate. It should be noted, however, that not all atomic/coherent expressions qualify as terms. For example, idioms like *"kick the bucket"* and fixed phrases like *"ad hoc"* are sequences of highly correlated words (collocations). They have high unithood. But they do not denote any concept in a domain and do not satisfy the syntactic structure to be sanctioned as terms.

Based on its definition, the unithood property has to be evaluated only for complex expressions, consisting of more than one word. Simple expressions are usually assumed to

[6] An expression is simply a word or a word sequence.
[7] Following previous studies in terminology, we use term candidate instead of term. It denotes an expression that is likely (i.e. candidate) to be selected as a term. The final decision as to whether a candidate is an actual term is usually made by domain experts.

have perfect unithood [54]. They consist of a single word, and thus, always operate as atomic units.

2.2.6 TERMHOOD

Termhood [54] determines whether an expression is representative of a domain. Expressions that are deemed to be representative are said to be relevant or domain-specific. These expressions are likely to qualify as term candidates. For example, *"sensor button"* represents a concept that is closely related to the PD-CS domain. Hence, it qualifies as a term candidate. Conversely, *"meeting room"* designates a concept that is atypical to PD-CS. Hence, it is unlikely that *"meeting room"* denotes a term candidate in the PD-CS domain.

Termhood is a unique property of terms. This is unlike the unithood property, which, as described earlier, also characterizes a wide range of expressions. An idiom like *"jump the bandwagon"*, for instance, satisfies the criteria of unithood by virtue of its strongly correlated words. However, it is a generic expression, which does not assume any specific meaning and is irrelevant in most domains. Thus, it fails to meet the termhood requirement for qualifying as a term candidate. Termhood therefore seems to play a more important role than unithood for accurately detecting terms.

In the next section, we will describe how the unithood and termhood properties can be computationally implemented in order to enable the automatic identification of terms from texts.

2.2.7 AUTOMATIC APPROACHES FOR TERM EXTRACTION

Several techniques have been proposed for detecting and extracting terms based on the properties of unithood and termhood. In this dissertation, we are only concerned with techniques that evaluate these properties by relying on evidence acquired from the computational analyses of corpora. This follows from the empiricist groundings of our work, as mentioned in Chapter 1.

The majority of term extraction (TE) techniques evaluate the unithood and termhood by relying either on linguistic or on statistical evidence. Linguistic techniques determine the unithood of an expression and establish whether it is well-formed by examining syntactic evidence (features). We will describe them in Section 2.2.8.

Statistical techniques are used for estimating both the unithood and termhood.
Since simple expressions are assumed to possess perfect unithood, statistical techniques for unithood estimation are usually defined for complex expressions, in particular, those containing two words. These techniques estimate the unithood of a (complex) expression based on the co-occurrence frequency of the individual words that make up the expression. Statistical techniques for unithood estimation are discussed in Section 2.2.9.

To evaluate the termhood of an expression, statistical techniques rely either on its occurrence frequency in a domain-specific corpus or on its relative frequency across a domain-specific corpus and another corpus from a different discipline. These statistical termhood estimation techniques are presented in Sections 2.2.10.

Recently, a new class of hybrid techniques that combine both linguistic and statistical evidence have been proposed. We describe them in section in 2.2.11.

2.2.8 LINGUISTIC TECHNIQUES: UNITHOOD

Linguistic techniques identify term candidates based on the first criterion of unithood. An expression is admitted as candidate only if its underlying structure (i.e. its formation pattern) corresponds to a well-formed syntactic unit. The formation pattern is often represented using syntactic features.

The most popular syntactic feature used by linguistic techniques for TE is the Part-of-Speech (POS). The POS of a word (token) corresponds to its "type" ("word-class"), such as

nouns, verbs (in various tenses), adjectives and prepositions. For example, the POS of *"sensor"* is a noun (N), that of *"rear"* is an adjective (A), while *"of"* is a preposition (P).

Techniques that exploit the POS information operate upon the premise that since terms are designators of domain-specific concepts, they can only be manifested by certain types of words (POS). As observed by Wright and Budin [56] , some POS are more probable and convenient than others to function as terms. For example, none of the terms t_1–t_7, depicted earlier in Section 2.2.2, contain adverbs or particles, while they all consist of at least one noun, optionally modified by an adjective or a preposition.

Most linguistic techniques are therefore implemented as filters, which accept as term candidates only those words (or sequences) that belong to a restricted set of POS. A well-known linguistic filter is that of Dagan and Church [57]. This filter admits only nouns or noun sequences as terms. It is often referred to as a *close-filter* by virtue of its selective admission procedure. Another widely employed filter is that of Justeson and Katz [58]. It has a more relaxed selection policy, and admits any sequence of nouns with optional adjectives and/or prepositions as candidates. It is often referred to as an *open-filter*.

2.2.9 STATISTICAL TECHNIQUES: UNITHOOD

Several statistical techniques have been proposed for evaluating the second aspect of unithood, which determines whether an expression is an atomic (and coherent) linguistic unit. It is worth mentioning again that these techniques have been defined for complex expressions.

They determine the unithood of an expression by estimating the strength of the syntagmatic associations between its individual words. They operate upon the premise that sequences of n-words (n>1), which co-occur more often than spuriously, have a particular function or meaning in a corpus, and hence, behave as atomic linguistic units. These expressions will be awarded relatively high unithood scores, which are derived from the co-occurrence frequencies of their constituent words. Expressions whose scores satisfy a user-defined or experimentally-set threshold can then selected as term candidates. Thus, the higher the unithood score, the likelier it is for an expression to qualify as a candidate.

Lexical Association Measures (LAMs) are a set of statistical and information-theoretic techniques that compute the unithood of an expression by measuring the collocation strength (glue) between the individual words. A large number of LAMs can be found in NLP literature. They include standard measures, such as log-likelihood [8] and mutual information [4]. A thorough review of LAMs can be found in the work Pecina and Schlesinger [36]. They described a total of 82 LAMs, and compared their performance in extracting general collocations from texts.

LAMs have been originally designed for word pairs. Thus, they can only evaluate the unithood of expressions that contain two words. In their study, Petrovic et al. [59] extended several traditional LAMs, such as log-likelihood and mutual information, for estimating the unithood of expressions with at most 4 words. Schone and Jurafksy [60] also proposed a technique for computing the unithood of multi-word expressions containing n words. Given an expression with the words $w_1w_2…w_{n-1}w_n$, they split it into two smaller sub-expressions A= $w_{1…}w_i$ and B= $w_{i+1…}$ w_n, where i is chosen such that its value maximizes the probability of finding A and B, i.e. P(A)P(B). According to Schone and Jurafsky this strategy gives the expected probability that the 2 most probable sub-expressions are concatenated into the larger candidate. As will be explained in Section 2.5.4, this technique may prove to be inaccurate for our informally-written corpus. It tends to underestimate the unithood of valid terms (e.g. *"xray tube window cover"*), while overestimating that of invalid word sequences (e.g. *"customer helpdesk collimator shutter"*). Our proposed technique for unithood computation (Section 2.5.4), which also operates upon the premise of decomposing larger expressions into smaller sub-expressions, overcomes these limitations.

2.2.10 STATISTICAL APPROACH: TERMHOOD

Statistical techniques for evaluating the termhood property of terms also exist. These techniques compute a termhood score, which indicates the degree to which an expression is closely related (relevant) to a domain (corpus). An expression is considered to be

relevant, and is selected as a term candidate if its score is larger than a user-defined or experimentally-set threshold. Two main types of approaches are usually adopted for calculating the termhood [54], as will be next described.

Approach 1
Techniques following the first approach are based upon the observation that the recurrent use of an expression in a text collection is indicative of its relevancy. These techniques formulate the termhood of expressions as a function of their occurrence frequency in a domain-specific corpus. Typical examples of such techniques include the "term frequency – inverse document frequency" (tf-idf) [61] and the C-value [37]. Tf-idf considers an expression to be relevant if that expression effectively discriminates different documents in a corpus.

C-value calculates the termhood of an expression by taking into account three parameters. The first parameter is the expression's occurrence frequency in a domain-specific corpus. This parameter is based on the earlier mentioned observation that frequent expressions are likelier than infrequent ones to denote relevant expressions. The second parameter takes into account the frequency with which the expression is nested in a longer string. The rationale for this parameter is that nested expressions, which never or rarely occur on their own, are highly unlikely to be relevant. The final parameter is the length of the expression. This parameter is based on the assumption that the occurrence of a longer expression is considered to be a more important event than the occurrence of a shorter expression if both of them have the same frequency.

Approach 2
Techniques adopting the second approach are known as corpus-comparison techniques. They compute the termhood based on the statistical contrast between the distributions of expressions in a reference corpus (of standard/general language) and in a domain-specific document collection. The fundamental principle underlying these techniques is that terms, as designators of domain-specific concepts, are likelier to be found in a corresponding domain-specific corpus than in the reference corpus. In most corpus-comparison techniques, such as [62, 63, 64], the statistical contrast is calculated by comparing the frequencies with which the expressions occur in a domain-specific corpus and in a general text collection. For instance, an expression, which appears only in the domain-specific corpus but not in the general corpus, is likely to be relevant to the domain. Conversely, an expression that appears frequently in the general corpus is likely to be irrelevant.

Besides the relative frequencies, other metrics have been employed to characterize the statistical contrast between the distributions of expressions in a domain-specific and a general corpus. These include standard (statistical) measures, such as log-likelihood [65], χ^2 , t-test and mutual information [66]. For example, Rayson and Garside [65] relied on the log-likelihood measure to evaluate the contrast between a domain-specific corpus of air traffic control reports and the British National Corpus (BNC) [67] of general texts. Expressions that were more "probable" to occur in the domain-specific corpus were then extracted as relevant term candidates. A complete review of statistical measures employed in corpus-comparison techniques is given in the work of Kilgarriff [66].

Two important points concerning corpus-comparison techniques must be noted. First, there are no restrictions on the number of corpora that can be compared. Most existing studies employ only two, namely, a domain-specific and a general corpus. Second, and more importantly, the corpora being compared must exhibit comparable and contrastive characteristics. In particular, as indicated by Rayson and Garside [65] and Chung [63], they should differ along the dimensions of representativeness and size. This condition is a prerequisite for maximizing the statistical contrast between the corpora being compared, which facilitates the subsequent identification of domain-specific terms. We will elaborate on these dimensions when presenting our term extraction technique in section 2.5.3.

2.2.11 HYBRID APPROACH: UNITHOOD AND TERMHOOD

The linguistic and statistical techniques described in the previous sections for unithood and termhood computations are rarely used in isolation. A more common practice is to combine them in a cascading configuration.

The first step in such a configuration is an initial linguistic filtering stage, which identifies syntactically well-formed expressions that are likely to correspond to terms. The identified expressions are then weighed by the application of statistical techniques for determining their unithood or termhood scores. Since they rely on both linguistic and statistical information, these configurations are known as hybrid TE techniques.

Several hybrid TE algorithms are mentioned in literature [55]. We will briefly describe some of the most popular ones. The algorithms that we present were deliberately chosen to ensure that they combine the different techniques discussed in the preceding sections. Specifically, we will present hybrid algorithms that combine linguistic filtering with LAMs for unithood estimation, and linguistic filtering with C-value and corpus-comparison for termhood computation.

Hybrid 1: Linguistic Filtering and LAM
Vivaldi and Rodriguez [41, 68]proposed the YATE hybrid TE system. The linguistic filter of YATE selected noun sequences with optional prepositions as term candidates. The unithood scores of the accepted candidates were then computed using different statistical techniques, including mutual information (MI) [69], cube mutual information (MI3) [70], and log-likelihood [71]. YATE was evaluated by estimating its precision and recall in extracting terms from a bio-medical corpus. Its maximum precision of 0.50 and recall of 0.15 was achieved when the MI3 measure was used. In another set of experiments, Vivaldi and Rodriguez [41, 68] used the EuroWordNet dictionary (EWN) [72] to determine whether a word belonged to the bio-medical domain, i.e. to evaluate the termhood. They considered a word as relevant if it was a hyponym[8] of the concept "physiological state" in EWN. The maximum precision achieved when relying on EWN was 0.96, while the maximum recall was of 0.30. These results show that the performance of term extraction systems can be improved considerably by relying on external knowledge resources. A similar hybrid TE technique was developed by Xu et al. [73]. Their linguistic filter admitted nominal phrases as term candidates. They also relied upon additional information provided by an external knowledge resource, namely the GermaNet dictionary [74]. It was used to identify the synonyms of terms that had been extracted from a corpus. The unithood scores of the selected candidates were then calculated using statistical measures, such as MI and log-likelihood. The maximum precision of this technique in extracting terms from newspaper articles was estimated as 0.61. This performance was achieved when log-likelihood was used to compute the unithood scores. It is worth noting that the approach of relying upon external knowledge resources is often known as exogenous term identification [46, 45].

Hybrid 2: Linguistic Filtering and C-value
Frantzi et al. [37] developed a hybrid TE technique, particularly suitable for detecting complex terms, such as *"adenoid cystic basal cell carcinoma"*, and nested terms, such as *"cystic basal cell"* . Nested terms are those that appear in longer strings, and not on their own. The linguistic filter employed in their study accepted sequences of nouns with optional adjectives and/or prepositions as candidates. Then, a frequency cut-off was applied to discard all candidates that occurred less than three times. The termhood scores of the selected candidates were next calculated using the C-value algorithm, discussed in Section 2.2.10. The performance of this approach was evaluated over a medical corpus consisting of eye pathology reports. The (maximum) precision and recall reported were respectively 0.38 and 0.98.

This technique is considered as a state-of-the-art TE system [75, 76, 77], and has been extensively employed as part of other NLP and IR applications [78, 79]. We will use it as a baseline to compare the performance of our proposed TE approach in the Experimental Evaluations, discussed in Section 2.6.

Hybrid 3: Linguistic Filtering and Corpus Comparison
Another example of a hybrid TE application is the TermoStat system of Drouin [80]. TermoStat selects nouns with optional adjectives as term candidates. To calculate the

[8] Hyponymy refers to the "is-a" relationship between a sub-concept (sub-class) and a concept (class), such as between "car" and "vehicle".

termhood scores of the identified candidates, it adopts a corpus-comparison approach. TermoStat was applied to extract terms from a domain-specific corpus dealing with telecommunications and related topics (e.g. optics). The general corpus used for comparison comprised of articles from the "The Gazette" newspaper. TermoStat achieved a reasonably high precision of 0.81 in extracting simple terms. However, its precision in detecting complex terms was much lower, at only 0.65.

2.3 AUTOMATIC TERM EXTRACTION CHALLENGES

It can be seen from the previous section that the majority of TE techniques developed to date have primarily been applied on documents from the general and scientific domains, such as newspaper and bio-medical articles. As discussed in Chapter 1, the extraction of terms from documents in these domains is facilitated by several factors, namely, the large collections of well-written texts, which provide reliable linguistic and statistical evidence, and the availability of resources, such as ontologies, which are valuable repositories of domain knowledge. We also saw in Chapter 1 that these characteristics, which enable the accurate identification of terms, are rarely replicated in corporate domains like PD-CS. As a result, term extraction from corporate texts introduces several important challenges, which existing TE algorithms do not adequately address. In this section, we will elaborate on these challenges (introduced in Section 2.1), and discuss the shortcomings exhibited by current techniques. We will then elaborate on our contributions to overcome these challenges in Section 2.4.

Challenge 1: Data sparsity
Domain-specific texts are sparse, and do not provide sufficiency redundancy for the acquisition of reliable statistical evidence. The absence of significant statistical evidence hinders the detection of terms from these documents. For example, in sparse texts, important domain concepts are often represented by rare terms, which occur with low frequencies [22, 39, 40, 41, 42, 81]. Due to their low frequencies, these rare, but no less important terms, are often awarded low (unithood and termhood) scores by most existing TE algorithms, and are incorrectly rejected.

This phenomenon, known as *silence* [41], is especially detrimental to the recall (coverage) of TE techniques. The issue of silence is further exacerbated by the conventional TE practice of discarding expressions that occur five times or less in a corpus [59, 81].

Challenge 2: Incoherent language
As mentioned in Chapter 1, domain-specific texts abound with various types of linguistic incoherencies since they are often informally written, and less care is taken to ensure their legibility and grammatical soundness. These incoherencies are visible both at the surface (word) level and at the deeper (syntactic) level. Surface level incoherencies are manifested as typographical errors, unofficial abbreviations, or inconsistent use of symbols like hyphens. Several techniques have been proposed to automatically detect and correct their occurrences [82, 83, 84, 85].

Linguistic incoherencies at the syntactic level have more serious implications. An example of such an incoherency, which is common in terse documents like engineers' repair reports, is the omission of the determiner "the" [20]. As an illustration, consider the sentence *"device is cooling fan"*, in which the author has omitted a *"the"* between *"is"* and *"cooling"*, as in "*device is the cooling fan*". While such an omission appears to be innocuous at first sight, a closer inspection reveals that it leads to ambiguity. For example, *"device is cooling fan"*, lends itself to two possible interpretations. The first and correct interpretation is that *"device"* and *"cooling fan"* refer to the same entity, that is, the *"device"* is the *"cooling fan"*. In this reading, *"is"* denotes the simple present form of the verb *"to be"*, and *"cooling"* is a gerund noun, which modifies *"fan"* to create the complex term (noun compound) *"cooling fan"*. The second and erroneous interpretation is that the *"device"* is actually *"cooling"* a *"fan"*. In this reading, *"cooling"* is incorrectly considered to be a (progressive) verb, with *"is"* as its auxiliary, *"device"* as its syntactic subject and *"fan"* as its direct object. These ambiguous and ungrammatical constructs hinder the accurate analyses of informally written, domain-specific documents for the acquisition of reliable linguistic evidence. The

absence of such evidence compromises the accuracy of TE algorithms in extracting terms from the documents' contents. For example, most linguistic filters will only identify *"fan"* as a term, instead of the actual term *"cooling fan"* since *"cooling"* will be regarded as a progressive verb, and not part of the term.

Another example of a syntactic level incoherency is the omission of punctuation symbols, in particular, periods (".”). Such omissions give rise to unintelligible constructs, such as the sequence *"customer helpdesk collimator shutter"*. These constructs are incoherent, and do not operate as atomic linguistic units. For instance, *"customer helpdesk collimator shutter"* is made up of two units, viz. *"customer helpdesk"* and *"collimator shutter"*, which are not properly demarcated by a period (".”), as in *"…customer helpdesk . Collimator shutter …."*. However, most linguistic filters will accept these invalid sequences as term candidates based on their syntactic structure. For example, the syntactic structures of the invalid sequence *"customer helpdesk collimator shutter"* and of the PD-CS term *"radiation protector shield arm"* are indistinguishable from each other. Both are noun phrases.

Challenge 3: Multi-word terms
Past TE studies have revealed that complex terms account for around 85% of all terms in domain-specific corpora [86]. Most of the existing LAMs can be used only to detect 2-word terms, such as *"sensor button"*. The LAM extensions of Petrovic et al. [59], discussed in Section 2.2.9 for collocation extraction, can also be applied for identifying terms containing at most four words, like *"radiation protector shield arm"*. However, these techniques fail to accurately detect longer terms, such as *"xray image frequency convertor control board"*, with six words, which are commonly encountered in specialized domains like PD-CS.

Challenge 4: Frequently used irrelevant or invalid constructs
Some irrelevant words like *meeting room"* or incoherent constructs like *"customer helpdesk collimator shutter"* tend to occur more frequently than actual terms like *"frequency oscillator"* in informally-written documents. Consequently, they will be awarded relatively higher unithood/termhood scores, and incorrectly selected as terms. In addition, these irrelevant/invalid sequences adopt the same syntactic structure as valid terms, which compounds the difficulty in detecting and discarding them. For example, the general expression *"meeting room"*, the incoherent construct *"customer helpdesk collimator shutter"* and the term *"frequency oscillator"* are all noun phrases.

This phenomenon, whereby irrelevant (and incoherent) expressions are more recurrently used than valid terms, and are subsequently incorrectly selected as terms, is known as *noise* [41]. It degrades the precision of TE techniques.

Challenge 5: Lack of domain-specific knowledge resources
The availability of domain-specific knowledge resources greatly facilitates term extraction efforts. For example, Hoste et al. [46], selected as terms any sequences of words in a medical corpus that corresponded to an entry in the Medical Subject Heading (MeSH) thesaurus [87] and the ziekenhuis.nl dictionary [88]. Similarly, Aubin and Hamon [45] relied on the Gene Ontology [18] and the MeSH thesaurus [87] to identify terms from a medical text collection. Other knowledge resources for supporting term extraction include the EuroWordNet (EWN) and the GermaNet, as discussed in Section 2.2.11.

However, with the exception of bio-medicine, specialized sources of domain-knowledge are scarce in most disciplines. This scarcity is notably conspicuous in corporate (industrial) environments, where much of the operational knowledge is not properly documented, and may not be reliable to support critical activities such as Business Intelligence (BI). Manually creating and maintaining these resources is also not viable and is impeded by the Knowledge Acquisition (KA) bottleneck. Consequently, TE algorithms in corporate domains are unable to supplement their linguistic and statistical analyses with additional, valuable knowledge from external resources like ontologies. As a result, their performance deteriorates.

2.4 CONTRIBUTIONS

The preceding section highlighted the lacunae in extant term extraction research, and illustrated the limitations that plague current TE algorithms. To fill this gap, and to address our first research question, we developed and implemented the ExtTerm framework for term extraction.

Compared to other TE algorithms, ExtTerm presents a number of novel aspects, which enable it to successfully overcome the intricacies of term extraction from domain-specific, sparse and informally-written documents.

Our main innovations and contributions with ExtTerm, in response to the aforementioned challenges, are:

1. A statistical technique for termhood estimation, which accurately detects rare terms, including those occurring with very low frequencies, for example, only once or twice in a corpus. This technique plays an important role in alleviating the issue of silence (Contribution 1 in response Challenge 1).

2. A linguistic filter that robustly identifies valid terms, even if reliable syntactic features are not available. This filter alleviates the difficulties involved in detecting terms from informally-written documents (Contribution 2).

3. A statistical technique that computes the unithood of an expression by leveraging upon the mechanism of term formation, discussed in Section 2.2.3. Our proposed technique accurately detects complex terms of arbitrary lengths, overcoming the limitations of current TE approaches, which are mostly suitable for terms with 2 words. (Contribution 3).

4. In addition, the technique that we present precisely discriminates between valid terms and invalid expressions, alleviating the effect of noise (Contribution 4).

5. Our overall term extraction framework is unsupervised, and identifies terms solely based on linguistics and statistical analyses (derived from the corpus). It eschews the need for domain-specific knowledge resources, such as terminological databases of PD-CS terms, which are expensive to produce and not always available (Contribution 5).

Our ExtTerm framework will be presented in the next section.

2.5 EXTTERM FRAMEWORK FOR TERM EXTRACTION

The overall architecture of the ExtTerm framework is shown in Figure 2.1. Blank arrows represent inputs and outputs, and are not considered to be part of ExtTerm per se. Filled arrows depict the processing performed by the various phases of ExtTerm.

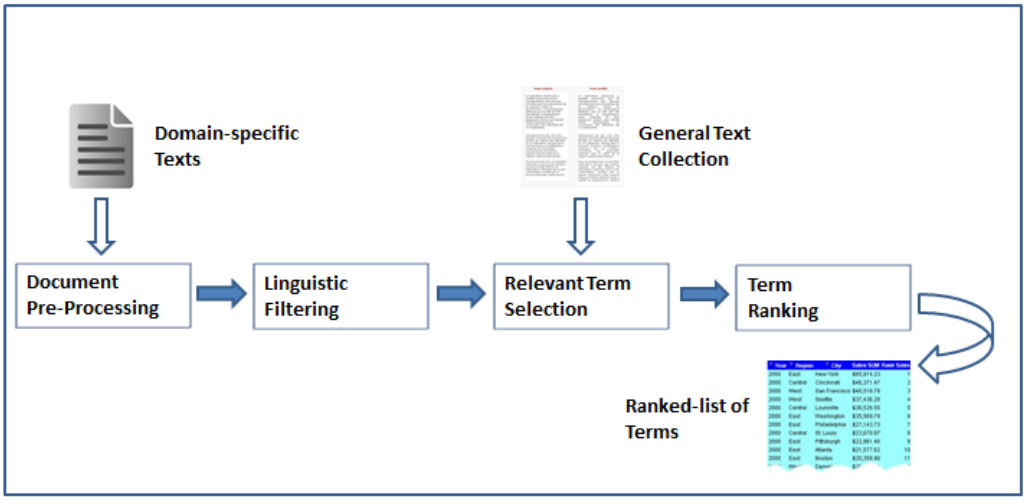


Fig. 2.1: ExtTerm Framework for Term Extraction.

ExtTerm is a hybrid TE system, and hence, it identifies terms by relying on both linguistic and statistical evidence. It takes as input a collection of sparse, domain-specific and informally-written documents. It starts with a Pre-Processing phase (Section 2.5.1), which transforms the input documents into an amenable format for term extraction. Then, the Linguistic Filtering stage (Section 2.5.2) identifies term candidates from the pre-processed documents. Next, the Relevant Term Selection stage (Section 2.5.3) detects those candidates that are closely related (i.e. relevant) to our domain. The Term Ranking phase (Section 2.5.4) then identifies those candidates that operate as atomic and coherent linguistic units. Finally, ExtTerm generates as output a ranked-list of terms, which can be examined by domain experts or consumed by automatic BI applications.

In the next sections, we will describe the different phases of ExtTerm, using as illustrations, real-life examples from an informally-written and sparse corpus in the corporate domain of PD-CS.

2.5.1 DOCUMENT PRE-PROCESSING

The Document Pre-processing phase converts the input documents into a more amenable format that facilitates their automatic analyses by the subsequent stages of ExtTerm. It involves the two basic operations of Data Cleaning and Linguistic Pre-processing. Before describing these operations, two points need to be mentioned.

1. Although document pre-processing is presented as part of the ExtTerm framework, it is a standard activity performed in NLP efforts with an empiricist grounding. Our documents could have been pre-processed independently of ExtTerm. Our rationale for positioning document pre-processing within the ExtTerm framework is that term extraction is the first task (step) to be executed in our quest for meaningful information from the documents' contents.
2. Since ExtTerm is a precursor to our tasks of Relation Extraction and Text Clustering, the same set of pre-processed documents is used as input to these tasks. Therefore, we will omit the document pre-processing phase when describing our approaches for Relation Extraction and Text Clustering, discussed in the later chapters of the thesis.

Data Cleaning
During Data Cleaning, we discard all extraneous contents, which can impede the accurate discovery of meaningful information (terms, semantic relations, document clusters) from our text collection. Our cleaning mechanisms are implemented as regular-expression wrappers, which detect and discard entities that are unlikely to be content-bearing. Examples include undesirable symbols (e.g. "^"), meta-data (e.g. email headers), and other "noisy" strings that domain experts considered redundant (e.g. "external remarks"). As part of our cleaning procedure, we also normalize the use of punctuations by keeping only one instance of punctuation symbols (e.g. "?") if they appeared in a sequence (e.g. "???").

Sample cleaning activities performed by ExtTerm, original text snippets on which they are applied, and the transformed cleaned texts are listed in Table 2.1. Noisy entities that are dropped are shown italicized.

Type of cleaning	Original text snippet	Cleaned text
Discarding irrelevant identifiers, symbols, and strings	*…080107 1650: <EOL> External Remarks* -> Reference monitor is flickering …	…reference monitor is flickering…
Dropping enclosing brackets	…they have had the intermittent problems *(*after patch upgrade*)*…	…they have had the intermittent problems after patch upgrade…

Table 2.1: Sample cleaning activities of ExtTerm.

Linguistic Pre-Processing
After cleaning, the texts are linguistically pre-processed to derive linguistic features from their contents. In this section, we will only be concerned with Part-of-Speech (POS) features, which are used by ExtTerm's Linguistic Filtering phase (Section 2.5.2). The acquisition of deeper level syntactic features, such as dependency trees, is deferred until our discussion on Relation Extraction in Chapters 4 and 5.

The POS of a token (word) is usually inferred from the tokens (their lexical forms) and from their contexts (neighboring words). POS information is relatively easy to generate using readily available NLP tools such as POS-taggers, which automatically annotate (tag) each token in a sentence with its corresponding POS.

Table 2.2 shows two cleaned text segments whose tokens have been annotated with their corresponding POS using the Stanford POS-tagger [89]. Tokens and POS-tags are delimited with the pipe ("|") character. We use "N" as POS-tag for all classes of nouns (e.g. plural, singular, proper), "P" for prepositions, and "A" for adjectives. All other POS-tags are as originally defined in the Penn Treebank [90]. For example, "VBG" refers to a verb in the progressive tense.

Before	After linguistic processing
reference monitor is flickering	reference|N monitor|N is|VBZ flickering|VBG
they have had the intermittent problems after patch upgrade.	they|PRP have|VBP had|VBD the|DT intermittent|A problems|N after|P patch|N upgrade|N

Table 2.2: Words and their POS

In spite of its robustness, POS-tagging is not error-free. Standard POS-taggers are trained on large collections of well-written texts. Their performance degrades when they are applied to domain-specific texts, which are rife with informal language constructs. In particular, ambiguous constructs, in which "is/are" immediately precedes a gerund noun, represent a common source of POS-tagging errors. For example, most POS-taggers fail to determine that the gerund *"cooling"* in *"device is cooling fan"* corresponds to a noun. Instead, they almost always incorrectly tag *"cooling"* as a progressive verb, having as syntactic subject and object the nouns *"device"* and *"fan"* respectively. This happens because such subject-verb-object triples closely resemble the standard language models

on which the POS-taggers are trained. Another illustration of a similar POS-tagging error, in which a gerund (*"regulating"*) is incorrectly identified as a progressive verb, is depicted in Figure 2.2.

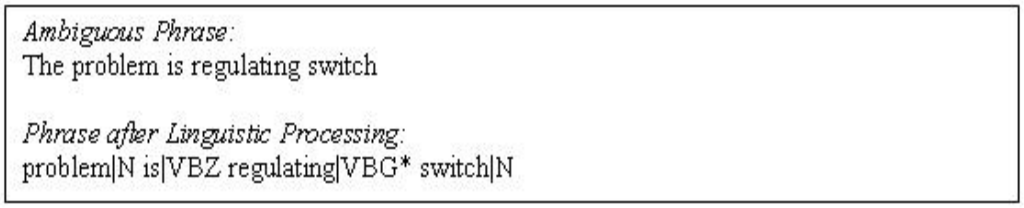


Fig. 2.2: Incorrect POS assignment.

Incorrect assignments of POS-tags affect the accuracy of linguistic TE techniques, which rely on the POS information to identify terms (Challenge 2). Possible solutions to prevent such errors are to manually correct the ambiguous phrases or to re-train standard NLP tools on our domain-specific texts. However, these alternatives are tedious and time-consuming. Furthermore, re-training standard tools may be hindered by the unpredictable language models that characterize informally-written documents. Therefore, we do not pursue any of these alternatives. Instead, we overcome the aforementioned difficulties in identifying terms from informally-written texts by implementing additional mechanisms in our TE framework, as will be described in the following sections.

2.5.2 LINGUISTIC FILTERING

The pre-processed documents are first fed to the Linguistic Filtering phase, which identifies term candidates from the documents' contents. Only those expressions whose syntactic structure is licensed by a well-formed POS pattern will be admitted as candidates. Thus, Linguistic Filtering considers only the first aspect of unithood. The second aspect of unithood will be evaluated in the Term Ranking phase in Section 2.5.4.

Conventional linguistic filters, such as those of Dagan and Church [57], Justeson and Katz [58] and Frantzi et al. [37], are susceptible to POS-tagging errors. For example, in Figure 2.2, they are unable to extract the entire expression *"regulating switch"* as a term candidate. Instead, they can only identify the fragment *"switch".*

Dealing with POS-Tagging Errors
As a brute-force solution to overcome this difficulty, ExtTerm defines a linguistic filter that accepts as term candidates any sequences of nouns (N), preceded by an optional progressive verb (VBG) and an adjective (A). This filter, which we call Filter_1, is depicted in the regular-expression shown in equation (2.1). The symbols "?" and "+" are regular-expression cardinality operators, respectively indicating that their operands are optional or that they occur at least once.

$$Filter\_1 = \sim A?\ VBG?\ N+ \qquad (2.1)$$

Filter_2: Complex Terms
Traditional filters are also too restrictive to detect complex candidates, especially those comprising of any number of nouns, interspersed with adjectives and/or prepositions. An example of such a candidate is *"rear battery cabinet coaxial cable"*, whose POS pattern can be defined as A N N A N, where A represents an adjective and N is a noun.

To overcome the difficulties in detecting such terms, we rely on the notion of base-terms, which are the primary units of the term formation process. As discussed in Section 2.2.3, the syntactic structure adopted by base-terms is defined by the patterns N, A N or N N. In regular-expression notation, these three patterns can be compressed into A? N+.

Since longer terms are created by juxtaposing base-terms with other adjectives and/or nouns, and by inserting prepositional modifiers, they can be represented by the pattern in equation (2.2). The regular-expression cardinality operator "*" indicates that its operand can occur 0 or more times.

$$Filter\_2 = \sim A?\ N *\ P?\ A?\ N + \quad (2.2)$$

Combining the regular-expressions shown in equations (2.1) and (2.2), and factoring their common elements yields the POS pattern employed by ExtTerm's linguistic filter (Contribution 2). It is depicted in equation (2.3).

$$Candidate\_Term = \sim A?\ VBG?\ N * P?\ A?\ N + \quad (2.3)$$

Our proposed filter maximizes the balance between precision and recall. Precision is optimized by being strict about the order and cardinality of the filter elements (i.e. POS-tags). Recall is favored by allowing progressive verbs to be part of terms.

The above discussion shows that when identifying terms from domain-specific texts, there is often a need to redefine existing linguistic filters so as to take into account the peculiarities of the (sub-)language in which the texts are expressed. It can be argued that, to some extent, our filter in equation (2.3) is idiosyncratic to our domain/corpus. However, since the term formation mechanism, upon which it is designed, is a generic process, our filter can be easily modified to detect terms in documents of other domains. For example, by dropping the "VBG" element, we obtain a filter which is a more elaborate version of that of Justeson and Katz [58]. Such a filter, should, in principle, be able to detect candidate terms from texts in other domains, such as newspaper or bio-medical articles in the general and scientific domains.

As output, Linguistic Filtering produces a set of term candidates, which are sanctioned by the POS pattern of equation (2.3). These candidates are then fed to the Relevant Term Selection phase, which we discuss next.

2.5.3 RELEVANT TERM SELECTION

The Relevant Term Selection (RTS) phase estimates the termhood scores of candidates. This is achieved using a corpus-comparison approach. The main strength of corpus-comparison is that it is independent from the candidates' *absolute* frequency. Therefore, it plays an important role in mitigating the phenomena of silence and noise (Challenges 1 and 4). For example, consider a rare, relevant candidate, which occurs only once (i.e. frequency =1) in the domain-specific corpus, but not (i.e. frequency =0) in the general corpus for comparison. Such a candidate will be awarded an extremely high termhood score, which is given by its *relative* frequency (i.e. frequency ratio) across these two corpora. The candidate can then be easily detected and selected based on its high termhood score, thereby addressing the issue of silence (Contribution 1). Conversely, consider a frequent, irrelevant candidate, which is recurrently used both in the domain-specific corpus and in the general corpus. Such a candidate will be awarded a much lower termhood score. It can then be easily detected and rejected based on its low termhood, thereby addressing the issue of noise (Contribution 4).

Corpus-Comparison Algorithm

Our corpus-comparison procedure for calculating the termhood of candidates is listed in the pseudo-code below. The variable *cand* represents a candidate selected from the Linguistic Filtering phase. Its frequency in the domain-specific corpus, *DS*, is given by f_{DS}. The variable f_{NC} denotes the candidate's corresponding frequency in the general corpus used for comparison. Following Rayson and Garside, [65] we refer to the latter corpus as a normative corpus, *NC*.

Procedure termhood_score(cand)
1. relevant_candidates = {} //set of identified relevant expressions
2. $p_{DS} = f_{DS}/|DS|$
3. $p_{NC} = f_{NC}/|NC|$
4. $termH = p_{DS}/ p_{NC}$
5. if termH >= t then
6. add (t, termH) to relevant_candidates

We start by initializing an empty set (line 1) to harvest the candidates deemed specific to the specialized corpus. For the given candidate, *cand*, we estimate its probability of occurring in the domain-specific corpus (line 2), and in the normative corpus (line 3).We then compute its termhood score, *termH*, as the ratio of its probabilities across these two corpora (line 4). In this way, candidates that are likelier to appear in the domain-specific corpus will be allocated higher termhood scores. A candidate is then considered to be relevant if its score satisfies a termhood threshold *t* (line 6). These relevant candidates will be fed to the Term Ranking phase, discussed in Section 2.5.4. Choosing an appropriate threshold is dictated by performance considerations. Its discussion is deferred until Sections 2.5.4 and 2.6.4.

A more systematic approach to capture the contrast between the distribution of candidates across our domain-specific and normative corpora is to use a statistical test, such as log-likelihood. The null hypothesis in this case is that there are no differences between the (observed) frequencies of a candidate across the corpora. The alternative hypothesis is that the candidate's distribution is different in the domain-specific and normative corpora. When using the log-likelihood test, for instance, candidates that exhibit the most significant (contrasting) frequencies (distributions) across the two corpora will be awarded higher log-likelihood (termhood) scores. Conversely, those that occur with approximately the same frequencies across the domain-specific and normative corpora will be awarded much lower scores. Relevant candidates, which are likelier to be found in the domain-specific corpus, can then be identified based on their higher scores (i.e. we reject the null hypothesis at a given significance level). Such an approach was adopted by Rayson and Garside [65], as described in Section 2.2.10. However, in our study, we did not investigate the use of statistical tests since reasonable results were obtained using our straightforward procedure based on the occurrence probability ratio, listed above.

Desiderata of Normative Corpus: Representativeness and Size

The accuracy of corpus-comparison techniques in calculating the termhood depends on how contrasting the domain-specific corpus and the normative corpus are. A high degree of contrast makes it easier to identify which candidates are relevant to the domain-specific corpus (based on their large termhood scores).

According to previous studies [63, 65], the domain-specific and normative corpora must exhibit contrasting characteristics along the primary dimensions of representativeness and size. This entails that the normative corpus we choose in our framework must satisfy two basic desiderata:

1. First, the normative corpus should be representative of a language. A representative corpus contains different types of texts that cover different domains, and that characterize the variations of a language, e.g. formal, informal and technical. Such a broad coverage corpus (of general language) will contrast more acutely with the contents (sub-language) of our domain-specific documents. As a result, the relevant terms will be more easily discernible.
2. The second criterion is that the normative corpus should be much larger that our sparse, domain-specific texts. A larger normative corpus is likelier to contain more occurrences of general expressions, which are not closely related to our domain. These irrelevant expressions can then be discarded from the domain-specific texts.

Wikipedia as a Normative Corpus

Typical corpora of standard (English) language, such as the British National Corpus (BNC) [67] and Wikipedia [91], satisfy the aforementioned desiderata. In our framework, we rely on Wikipedia as normative corpus[9]. This decision was motivated by practical considerations. We already had at our disposal a linguistically pre-processed (POS-tagged and syntactically parsed) version of the Wikipedia corpus, which could be straightforwardly incorporated in our framework. In addition, as will be discussed in the later chapters, Wikipedia was also more suitable than BNC to support our subsequent (and more sophisticated) task of

[9] Strictly speaking, BNC, which is a collection of general texts, is more representative of the English language than Wikipedia, which is an encyclopedic resource.

relation extraction. Therefore, using Wikipedia for the current task of term extraction allowed us to be consistent in our choice of resource.

Sample Output
The output of the RTS phase is a set of (simple and complex) candidates, which are relevant to our domain-specific corpus. Table 2.3 shows sample candidates, their occurrence frequency in the domain-specific corpus, f_{DS}, and their termhood, as calculated by the *termhood_score* procedure. All scores are normalized to lie in the range 0-500. Candidate appearing only in the domain-specific corpus and not in Wikipedia are awarded a maximum score of 500.

Candidate	f_{DS}	Termhood
filament control board replacement kit	2	500
c-arm backplane rotation sensor	108	500
customer helpdesk collimator shutter	94	500
radiation shield	58	479.06
Baton Rouge	245	52.28

Table 2.3: Candidates and Termhood Scores from Relevant Term Selection.

From the results depicted in Table 2.3, it can be seen that our technique successfully addresses the issue of silence (Challenge 1), and accurately detects even the low frequency terms (Contribution 1). For example, despite occurring only twice in the domain-specific corpus, the relevant candidate *"filament control board replacement kit"* is awarded the maximum termhood score of 500 by ExtTerm, indicative of its domain-specificity.

The results also show that ExtTerm overcomes of the issue of noise (Challenge 4), and precisely discriminates between valid and invalid terms. For example, despite its high frequency, the candidate *"Baton Rouge"* is awarded the lowest termhood score, indicative of its irrelevancy to our PD-CS domain. In Section 2.6.7, we will perform additional experiments to demonstrate that ExtTerm is equally successful in extracting rare terms as well as more frequent ones.

The candidate *"customer helpdesk collimator shutter"* deserves special mention. This candidate occurs only in our domain-specific texts, and not in Wikipedia. Consequently, it is awarded the maximum termhood score on account of its specificity to our domain. However, as discussed earlier, it is an incoherent construct, which is unlikely to correspond to a term. These types of candidates, which are devoid of any meaning, must be properly recognized and rejected. At this stage of ExtTerm, however, we are only interested in finding all relevant candidates that are specific to our corpus based on their termhood. We will reject the incoherent candidates, and select only those that are coherent and behave as atomic linguistic units in the Term Ranking phase.

2.5.4 TERM RANKING

In the previous RTS stage, we computed the candidates' termhood scores, and the most relevant ones, whose scores satisfied a threshold *t*, are fed to the Term Ranking (TR) phase.

At this point in our term extraction procedure, the unithood has only been partially evaluated. During Linguistic Filtering, we selected candidates according to their syntactic structure. In the TR phase, we will evaluate the other aspect of unithood, which establishes whether the relevant candidates from RTS behave as atomic (and coherent) linguistic units. A candidate that is relevant (i.e. high termhood score from RTS) and functions as an atomic unit is likely to denote a term.

In the remainder of this section, we will refer to a candidate containing n-words as an n-candidate. For instance, *"radiation protector shield arm"*, which is made up of four words, is a 4- candidate.

Unithood for 2-candidates
The unithood scores of 2-candidates, such as *"tube cover"*, can be straightforwardly calculated using traditional LAMs. In ExtTerm, we use the cube mutual information (MI3) measure proposed by Daille [70]. Given a 2-candidate, *cand=x y* (e.g. *cand ="tube cover"*), MI3 estimates its unithood as

$$unithood\_score(cand) = \log \frac{\left(\frac{f(x,y)}{N}\right)^3}{\frac{f(x)}{N} \times \frac{f(y)}{N}} \qquad (2.4)$$

In this equation, *f(x,y)* is the co-occurrence frequency of the pair *"x y"*, while *f(x)* is the (individual) frequency of the word *x*.

Compared to the basic mutual information (MI) measure of Church and Hanks [69], MI3 takes the cube of the joint probability of the events, as shown in the numerator of equation (2.4). This strategy overcomes the shortcoming of the basic MI which tends to overemphasize rare events [70, 92]. Our choice for MI3 was motivated by our preliminary experiments and by other research efforts such as [41], which demonstrated that MI3 outperformed other LAMs like log-likelihood for term extraction.

Unithood for n-candidates, n>2
LAMs have been designed to operate upon word pairs. Hence, they are unsuitable for evaluating the unithood scores of longer candidates, containing more than 2 words, such as *"radiation protector shield arm"*. To overcome this difficulty, we propose a novel technique for accurately estimating the unithood of n-candidates, with n>2.

Our technique is hinged upon a mechanism that attempts to mimic the term formation process, described in 2.2.3. The central idea lies in decomposing an n-candidate into smaller sub-expressions, containing at least 2 and at most (n-1) words. We then formulate the unithood score of the n-candidate as a function of the unithood of its sub-expressions.

To illustrate the notion of sub-expressions, we consider two 4-candidates, namely, a valid term *cand="xray tube window cover"* and an invalid one, *candX ="customer helpdesk collimator shutter"*. Their sub-expressions of at least 2 and at most 3 (i.e. n-1) words are listed in Table 2.4.

n-gram (n = 4)	**Sub-expressions of size 2...(n-1)**	
cand = "xray tube window cover"	Size 2	Size 3
	"xray tube"	"xray tube window"
	"xray window"	"tube window cover"
	"xray cover"	"xray window cover"
	"tube window"	"xray tube cover"
	"tube cover"	
	"window cover"	
candX = "customer helpdesk collimator shutter"	Size 2	Size 3
	"customer helpdesk"	"customer helpdesk collimator"
	"customer collimator"	"helpdesk collimator shutter"
	"customer shutter"	"customer helpdesk shutter"
	"helpdesk collimator"	"customer collimator shutter"
	"helpdesk shutter"	
	"collimator shutter"	

Table 2.4: Sub-expressions generated from 4-candidates.

Two observations can be made from the sub-expressions presented in the above table.

1. (Observation 1) All the sub-expressions of the term *cand*, such as *"xray tube"* and *"xray tube window"*, are atomic units. These sub-expressions are likely to correspond to terms themselves. Conversely, with the exception of "*collimator shutter"* and *"customer helpdesk"*, the other sub-expressions generated from the invalid candidate *candX,* such as *"helpdesk collimator shutter"*, are incoherent. They do not behave as atomic units, and are unlikely to denote terms.
2. (Observation 2) As a direct consequence of the way we split an n-candidate, each of its sub-expression that contains m words will be nested in a longer expression with (m+1) words. The latter will then appear as part of other longer sub-expressions with (m+2) words. In this way, it is possible to reconstruct an n-candidate from its sub-expressions.

Based on these above observations, our technique for computing the unithood score of any n-candidate, n>2, is listed in the pseudo-code below.

```
Procedure unithood_score( )
1.    n = 2
2.    while (n < max_length) //maximum number of words found in candidates
3.       setCand_n = {all candidates containing n words}
4.       if (n ==2) //2-candidates
5.          for each 2-candidate cand in setCand_2
6.             score = MI3(cand)
7.             add(cand, score) scoreHash_2
8.       else //longer candidates, n > 2
9.          for each cand in setCand_n
10.            subExprSet = generate sub-expressions(cand)
11.            for each subExpr with m=2…(n-1) words in subExprSet
12.               score += retrieve score(subExpr) from scoreHash_m
13.            score /= n
14.            add(cand, score) to scoreHash_n
15.   n++
```

We start by computing the unithood scores of 2-candidaes (e.g. *"tube cover"*) using MI3 (lines 1-8). The 2-candidates and their scores are indexed in a look-up (e.g. hash) table so that they can be efficiently retrieved later.

In each subsequent iteration (lines 8-15), we process candidates containing an additional word. For example 3-candidates after 2-candidates, 4-candidatates after 3-candidates, and in general, n-candidates after (n-1)-candidates. This strategy is based upon our observation (Observation 2) that an n-candidate (n>2) can be reconstructed from its sub-expressions.

Therefore, given an n-candidate, we generate its sub-expressions, each of which contains at least 2 and at most (n-1) words (line 10). We consider all possible word permutations in order to detect sub-expressions even if their word-ordering varies. For example, the words *"tube"* and *"cover"* in the sub-expression *"tube cover"* are not adjacent when the sub-expression is nested in the 4-candidate *"xray tube window cover"*. This mechanism enables us to effectively deal with word order variations.

After decomposing the n-candidate, we inspect the look-up tables of the previous iterations to retrieve and accumulate the unithood scores of its sub-expressions (line 12). That is, we formulate the unithood of the n-candidate as a function of the scores of its sub-expressions. In this way, candidates that represent valid terms will be allocated higher scores. This follows from our observation (Observation 1) that the sub-expressions of a valid term are likely to represent terms themselves, and thus, they will have high unithood scores. On the other hand, invalid candidates will be assigned much lower unithood scores since their sub-expressions are likely to be incoherent word sequences, with low (even 0) unithood scores. Besides facilitating the detection of complex term candidates with more than two words, this strategy also amplifies the difference between the unithood scores

awarded to valid candidates and to invalid ones. This enables us to overcome the difficulties involved in discriminating between valid candidate terms and invalid word sequences.

After accumulating the unithood scores of the sub-expressions, we normalize this aggregated value by the candidate's length (line 13). This normalization is done to penalize longer candidates since

- They can be decomposed into a larger number of sub-expressions, and will be awarded higher scores than shorter ones.
- The probability that a candidate designates a term decreases as its length increases [93].

Finally, we index the n-candidate and its unithood score in a look-up table (line 14). It can then be retrieved for calculating the scores of longer candidates, e.g. with (n+1) words, in future iterations.

As already mentioned in Section 2.2.9, another approach that computes the unithood of an n-candidate based on its sub-expressions is that of Schone and Jurafsky [60]. To better illustrate the differences with our algorithm, we consider the previous examples *cand*="*xray tube window cover"* and *candX* ="*customer helpdesk collimator shutter"*. From *cand*, the approach of Schone and Jurafsky [60] identifies two sub-expressions, namely A="*xray tube"* and B= *"window cover"*, which are the 2 most probable sub-expressions in *cand*. The unithood of *cand* is then calculated as the product of the probabilities of A and B, i.e. P(A)P(B) = P("*xray tube"*)P("*window cover"*). As can be seen, compared to our proposed algorithm, it does not take into account the other sub-expressions formed by different word permutations. For example, it will not consider the sub-expressions "*xray cover"* and "*tube window"*, which are also valid terms. Hence, the approach of Schone and Jurafsky [60] may underestimate the unithood score of valid term candidates, which hinders their detection and extraction. Our algorithm, on the other hand, accumulates the unithood score of all the sub-expressions, which ensures that valid terms are awarded much higher scores than invalid ones.

From *candX* ="*customer helpdesk collimator shutter"*, the approach of Schone and Jurafsky [60] identifies two sub-expressions, namely C="*customer helpdesk"* and D= *"collimator shutter"*, which are the 2 most probable sub-expressions in *candX*. The unithood of *candX* is then given as P(C)P(D). However, since P(C) and P(D) will be large (they are the most 2 probable sub-expressions), the invalid candidate *candX* will be incorrectly awarded a high unithood score. Thus, this approach may overestimate the unithood of the invalid candidates, which are rife in our corpus. With our algorithm, on the other hand, the sub-expressions of invalid candidates will contribute only marginally to the candidates' unithood. This is because the majority of these sub-expressions, which are formed by considering all word permutations in the candidates, will also be invalid and have low unithood. As a result, the invalid candidates will be awarded equally low unithood scores (and rejected subsequently).

Sample Output

The output of the TR phase of ExtTerm is set of relevant candidates, which behave as atomic and coherent linguistic units. These candidates can be sorted according to their unithood scores, and presented in a ranked-list to facilitate their inspection by domain experts or consumption by automatic applications. Candidates that appear in the top section of the list are likely to denote valid (relevant and coherent) terms.

Sample candidates extracted by the TR phase are presented in Table 2.5. Similar to the termhood scores, we also normalize the unithood scores to lie in the range 0-500. In addition, since simple candidates, consisting of 1 word, are assumed to have perfect unithood, they are awarded the maximum score (500).

Candidate	Unithood	Length
fluoroscopy	500	1
filament control board replacement kit	234.03	5
fluid injector indication lamp control knob assembly	203.4	7
xray image frequency convertor control board	170.01	6
frequency control relay	162.54	3
radiation protector shield arm	152.01	4
collimator shutter	61.44	2
customer helpdesk collimator shutter*	15.36*	4

Table 2.5: Candidates extracted by Term Ranking.

Note: * denotes invalid candidates, which are unlikely to represent terms.

As illustrated in the above table, ExtTerm successfully addresses the difficulties in detecting multi-word terms (Challenges 3). It accurately extracts complex terms of arbitrary lengths, including those with more than 2 words (Contribution 3). Examples of such terms include *"filament control board replacement kit"* and *"fluid injector indication lamp control knob assembly"*, respectively containing 5 and 7 words. In Section 2.6.7, we will describe additional experiments to illustrate that ExtTerm is equally precise in detecting terms of different lengths.

These results also show that ExtTerm overcome the issue "noise" (Challenge 4). For example, it awards much higher scores to valid terms like *"filament control board replacement kit"* than to invalid ones like *"customer helpdesk collimator shutter"*, which enables us to precisely discriminate between them (Contribution 4). In addition, ExtTerm is completely unsupervised (Contribution 5) and eschews the need for knowledge resources, such as ontologies, which may not always be available in corporate domains to support TE efforts (Challenge 5).

The terms extracted by ExtTerm can then be further inspected by domain experts, and used to support various types of BI activities. In our quest for meaningful information from corporate texts, we will use these terms as the basis for the more sophisticated tasks of Relation Extraction and Text Clustering, to be presented in the later chapters.

2.6 EXPERIMENTAL EVALUATION

In this section, we evaluate the performance of our proposed ExtTerm framework. We start in Section 2.6.1 by briefly describing the corpora employed in our experiments. Then, in Section 2.6.2-2.6.4 we discuss the results of the various phases of our ExtTerm framework. The performance of ExtTerm is evaluated in Section 2.6.5, and is compared against a baseline in 2.6.6. As baseline, we used the term extraction approach of Frantzi et al. [37], which we presented in Section 2.2.11. This approach employs a linguistic filter to detect term candidates, and then computes the termhood of the selected candidates using the C-value algorithm. For convenience, we will refer to this approach as C-value. We selected C-value as baseline since it is considered to be a state-of-the-art TE technique [75, 76, 77]. It has been shown to achieve reasonable accuracy in term extraction from different domains [78, 79]. In addition, it is one of the most commonly used baselines to gauge the performance of TE efforts in extracting both single and multi-word terms, such as in the work of Zhang et al. [77].

2.6.1 CORPORA

Domain-Specific Corpus
The corpus that we targeted for the extraction of domain-specific terms is a collection of 32,545 documents generated in the business/corporate domain of Product Development-Customer Service (PD-CS). The documents describe customer complaints and the ensuing repair actions performed by service engineers on high-end, electronic equipment. The corpus was compiled over a period of five years, spanning from 2005 to 2009, and the documents were collected from customer-call centers and engineers' repair notes. The texts were expressed in English.

Wikipedia Normative Corpus
As normative corpus in our corpus-comparison technique for termhood computation (Relevant Term Selection phase of ExtTerm), we relied on the English Wikipedia collection (August 2007 dump, ~ 50 GB uncompressed) [91].

Some basic statistics on our domain-specific and normative corpora are presented in Table 2.6.

	Number of words	**Number of documents**
Domain-specific corpus	1,952,739	32,545
Wikipedia normative corpus	500 million (approx.)	5,100,000 (approx.)

Table 2.6: Corpora Statistics.

Type-Token Ratio for Sparsity
To illustrate the sparse characteristics of our domain-specific corpus, we computed its type-token (TTR) ratio, i.e. the ratio between the number of distinct word forms and the total number of words. The TTR was estimated as 0.07. Then, we determined the TTR of a subset of the Wikipedia collection, which had the same size (i.e. same number of words) as our domain-specific corpus. The corresponding TTR was calculated as 0.05. These TTR values indicate that words in our domain-specific corpus are repeated less frequently than in the Wikipedia collection. This suggests that the domain-specific corpus offers less redundancy and statistical evidence (to facilitate term extraction) than a general collection like Wikipedia. We will further illustrate the issue of data sparsity in our later experiments, and show that a state-of-the-art term extraction algorithm (i.e. our baseline) is inaccurate in detecting low frequency terms (e.g. occurring less than 5 times) in our corpus.

2.6.2 LINGUISTIC FILTERING

We evaluated ExtTerm's linguistic filter, $filter_{ExtTerm}$, presented earlier in equation (2.3), by comparing it with the filter shown in equation (2.5). It is similar to the filter employed by Frantzi et al. [37] in the baseline, and it also accepts single-words as candidate terms.

$$Filter_{Baseline} = ((A|N) * | ((A|N) * (NP)?)(A|N) *)N) \quad (2.5)$$

Our experiment results revealed that the filter with the broadest coverage was $filter_{ExtTerm}$. It admitted 85,342 distinct candidates as terms, which was around twice as many as the 48,984 that were admitted by $filter_{Baseline.}$ The higher coverage of $filter_{ExtTerm}$ guaranteed that it captured a larger number of candidates early in the TE process, which is beneficial to its overall performance. Conversely, $filter_{Baseline}$ prematurely rejected a number of valid candidates. It will be seen in Section 2.6.6 that these prematurely discarded candidates are responsible for the baseline's lower recall compared to ExtTerm.

Next, we compared the degree of (non-)overlap between the candidate sets output by the filters. For each filter, we determined the candidates that it uniquely identified, i.e. those candidates that it extracted but that were missed by its competitor. $Filter_{ExtTerm}$ was able to detect all the candidates that $filter_{Baseline}$ identified from our corpus. However, many of the candidates detected by $filter_{ExtTerm}$ were unique, and were missed by $filter_{Baseline}$.

To investigate the filters' behavior, we inspected their respective outputs, examples of which are summarized in Table 2.7. A value "Y" indicates that a filter (column) successfully extracted a candidate (row). An "N" is used to indicate unsuccessful extraction. In some cases, $filter_{Baseline}$ was only able to detect fragments instead of the entire candidates. These instances are marked with a "*".

	$Filter_{ExtTerm}$	**$Filter_{Baseline}$**
Complex Candidates		
Coaxial cable for rear battery cabinet	Y	N (coaxial cable for rear battery)*
Flat screen monitor wireless adaptor	Y	N (flat screen monitor)*
POS Errors		
limiting amplifier	Y	N (circuit)*
regulating switch	Y	N (switch)*
Common Candidates		
bodyguard sensor button	Y	Y
frontal lobe	Y	Y
Invalid Candidates		
tear box helpdesk cable sleeve	Y	Y
customer helpdesk collimator shutter	Y	Y

Table 2.7: Comparing candidates output by linguistic filters.

Four major observations can be made from the above results.

Complex Candidates

First, the unique candidates captured by $filter_{ExtTerm}$ (i.e. those missed by $filter_{Baseline}$) were complex, consisting of at least three words, for .e.g. *"coaxial cable for rear battery cabinet"*. On the other hand, $filter_{Baseline}$ could only capture fragments, e.g. *"coaxial cable for rear battery"*, instead of the entire complex candidates.

Candidates with POS-tagging Errors

The second observation is that $filter_{ExtTerm}$ detected even those candidates in which gerunds (nouns) had been incorrectly marked as progressive verbs due to POS-tagging errors. For example, it accurately extracted the candidate *"regulating switch"* despite *"regulating"* being incorrectly POS-tagged as a progressive verb. $Filter_{Baseline}$, on the other hand, could

only detect *"switch"*. This result illustrates that ExtTerm's linguistic filter accurately detected term candidates from our informally-written texts, even in the absence of reliable linguistic evidence.

Common Candidates
We noted that $filter_{ExtTerm}$ could extract all the candidates of $filter_{Baseline}$. This was possible since both filters were fundamentally similar to each other, allowing nouns, adjectives and/or prepositions as the basic constituents of terms.

Invalid Candidates
Some candidates identified by both filters, such as *"customer helpdesk collimator shutter"*, were invalid and incoherent word sequences, devoid of any meaningful interpretation. They were extracted since they were noun sequences, and satisfied the admission criteria of both $filter_{ExtTerm}$ and $filter_{Baseline}$. In the absence of statistical evidence in this phase, we were unable to conjecture about the validity of such expressions. They are dealt with in our Term Ranking experiments (Section 2.6.4).

We did not estimate the performance scores, such as the precision and recall, of the two filters since their outputs were not ranked in any order, which made it difficult for us to evaluate the top-n "best" candidates that they admitted. The large sets of unordered ("not ranked") candidates produced by the filters were too time-consuming to be evaluated. In Sections 2.6.5 and 2.6.6, we will determine the performance of both ExtTerm and the C-value baseline based on the ranked lists of candidates that they ultimately extract.

The candidates identified by $filter_{Baseline}$ were submitted to the baseline's statistical component, which calculated their termhood using the C-value algorithm, as described in [37]. We will deal with the baseline again when we compare its performance against ExtTerm in Section 2.6.6. The candidates from $filter_{ExtTerm}$ were them fed to ExtTerm's Relevant Term Selection phase for termhood computations.

2.6.3 RELEVANT TERM SELECTION

The Relevant Term Selection (RTS) phase estimated the termhood scores of the candidates based on their relative probabilities of occurring in the domain-specific texts and in the Wikipedia normative corpus. Candidates whose scores satisfied a threshold *t* were considered as relevant, and submitted to the Term Ranking stage (Section 2.6.4) of ExtTerm.

Sample candidates extracted by the RTS phase (1st column), their occurrence probabilities in our domain-specific and normative corpora (2nd and 3rd columns), and their termhood scores (4th column) are depicted in Table 2.8. Those candidates that appeared only in the domain-specific texts, but not in the normative corpus were awarded the maximum score of 500.

Candidate	**P_{DS}**	**P_{NC}**	**Termhood = P_{DS}/ P_{NC}**
Relevant and Likely			
angioplasty contrast adjustment buzzer button	0.0017	0	500
c-arm backplane rotation sensor	0.0010	0	500
Relevant but Sparse			
radiation protector shield arm	0.0000015	0	500
coaxial cable for rear battery cabinet	0.0000010	0	500
Relevant but Invalid			
tear box helpdesk cable sleeve	0.0016	0	500
customer helpdesk collimator shutter	0.0011	0	500
Likelier in Specialized Corpus			
sensor button	0.00035	0.0000011	308.17
camera shutter	0.00020	0.0000016	123.89
Irrelevant but Likely			
airport	0.00062	0.000012	50.85
day	0.00044	0.000014	31.68

Table 2.8: Candidates output from Relevant Term Selection phase.

The above results can be broadly classified into five categories. Candidates in the first category ("Relevant and Likely") occurred frequently in our domain-specific texts. For example, the candidate *"c-arm backplane rotation sensor"* was found a total of 2100 times. These candidates did not occur in the normative corpus. Hence, they were assigned the maximum termhood score (of 500), indicative of their relevancy to our domain.

Candidates in the second category ("Relevant but Sparse") occurred sparsely in the domain-specific texts. For example, *"radiation protector shield arm"* had a frequency of 3, while that of *"coaxial cable for rear battery cabinet"* was even lower; it appeared only 2 times in the domain-specific corpus. These infrequent candidates cannot be accurately detected by conventional TE algorithms, and they are responsible for the issue of silence (Challenge 1). However, based on the results presented in the above table, it can be seen that ExtTerm successfully overcomes this issue. Despite their low frequencies, these candidates were awarded high (maximum) termhood scores by ExtTerm. These high scores are indicative of the candidates' domain-specificity, and facilitate their detection, alleviating the issue of silence (Contribution 1).

Candidates in the third category ("Relevant but Invalid"), e.g. *"customer helpdesk collimator shutter"*, were also assigned maximum termhood scores since they occurred only in the domain-specific corpus. However, they were incoherent word sequences, and will be dealt with during the Term Ranking phase.

Candidates in the fourth category ("Likelier in Specialized Corpus") were likelier to occur in the domain-specific texts than in the normative corpus. For example, the frequency of *"sensor button"*, in the domain-specific corpus and in the normative corpus was respectively of 674 and 560. These candidates were awarded relatively high termhood scores, indicative of their relevancy.

Candidates in the last category ("Irrelevant but Likely") occurred frequently in the domain-specific texts. For example, *"airport"* was mentioned 1202 times. Thus, it was even more frequent than valid terms like *"sensor button"* and "*radiation protector shield arm*", which respectively occurred 674 and 3 times in the domain-specific texts. However, despite their relatively high frequencies, candidates in this category, such as *"airport",* were irrelevant to our domain. They are responsible for the issue of noise since they are incorrectly selected as terms by conventional TE algorithms on account of their high frequencies (Challenge 4). However, as can be seen from the above results, ExtTerm effectively addresses this issue. Despite their high frequencies, the candidates were awarded much lower termhood scores by ExtTerm. These scores are indicative of the candidate's irrelevancy, and facilitate their detection and rejection, alleviating the issue of noise (Contribution 4).

Selecting a Termhood Threshold

As mentioned earlier, only those relevant candidates, whose termhood scores satisfy a threshold *t*, will be selected and input to the subsequent TR stage of ExtTerm. Determining an appropriate threshold value is dictated by the intended application. Applications that favor precision over recall employ relatively larger threshold values. Thus, only a small set of relevant candidates, with high termhood scores, will be selected. On the other hand, applications favoring recall employ smaller thresholds. In this way, a larger number of candidates will satisfy the threshold, and will be selected.

In our framework, we are interested in finding the optimal balance between precision and recall in order to extract the largest (i.e. highest recall) set of valid (i.e. highest precision) terms. To this aim, we varied the threshold *t* across six different values of termhood scores: 10, 100, 200, 300, 400, and 500. These values were chosen to reflect the candidates' distribution. Around 90% of candidates had scores in the range 10-500, while the remaining 10% had infinitesimal scores and could be safely discarded. The threshold increments of 90, 100, 100, 100, and 100 were determined so that each interval consisted of approximately the same number of candidates. For each of these (six) threshold values *t*, we harvested (six) different sets of candidates from the RTS phase based on their termhood scores (i.e. the candidates' scores satisfied *t*). For each candidate set, we measured the precision and recall of the Term Ranking phase (Section 2.6.5). Then, we chose as threshold that value of *t* which optimized the balance between precision and

recall. As will be described in the next section, we found out that this balance was maximized at t=200.

2.6.4 TERM RANKING

The Term Ranking (TR) phase computed the unithood scores of the candidates, which were selected from the previous RTS stage for each of the six different threshold values, i.e. t= 10, 100, 200, 300, 400, and 500.

As output, TR generated a ranked-list of candidates, sorted according to their unithood scores. An example of such a ranked-list output is shown in Table 2.9. As discussed before, simple candidates, consisting of single words, were awarded the maximum unithood score of 500.

Candidate	**Unithood score**
footswitch	500.00
filament control board replacement kit	234.03
c-arm backplane rotation sensor	230.63
fluid injector indication lamp control knob assembly	203.40
coaxial cable for rear battery cabinet	200.70
tube window cover	189.72
xray image frequency convertor control board	170.01
radiation protector shield arm	152.01
wireless repeater keyboard front cover	102.12
angioplasty contrast adjustment buzzer button	98.35
cable control box	92.34
circuit controller	89.12
…	…
customer helpdesk collimator shutter	15.36
tear box helpdesk cable sleeve	10.78

Table 2.9: Sample Ranked-List from Term Ranking Phase.

From the above results, we can see that ExtTerm successfully addresses the difficulty of extracting terms containing more than 2 words (Challenge 3), and accurately identifies terms of arbitrary lengths, including those with 3 or more words (Contribution 3). For example, ExtTerm assigned a relatively high unithood score to *"fluid injector indication lamp control knob assembly"*, with 7 words, to indicate that it is an atomic and coherent unit, which is likely to denote a valid term. Our results also show that ExtTerm alleviates the issue of noise (Challenge 4), and precisely distinguishes valid terms from invalid ones (Contribution 4). This is reflected in the much lower scores it assigned to invalid candidates like *"customer helpdesk collimator shutter"*, indicating that they are unlikely to denote terms. On the other hand, valid candidates, such as *"c-arm backplane rotation sensor"*, were assigned much higher scores. Furthermore, our term extraction approach is unsupervised (Contribution 5), and does not require additional information from other resources, such as domain-specific ontologies, which are scarce in most specialized/corporate disciplines like PD-CS (Challenge 5).

2.6.5 EVALUATING EXTTERM'S OUTPUT AND SELECTING THRESHOLD

For each of the six threshold values t, we evaluated the corresponding set of candidates extracted by ExtTerm, as depicted in Table 2.9. Since evaluating entire ranked lists is tedious and time-consuming, past TE studies have focused on the top-n candidates. In addition, it was shown by Evert and Krenn [94] that the evaluation results over a random sample of size n were comparable to those over the entire population set.

In our experiments, we set n = 1000, i.e. we evaluated the top-1000 candidates in each of the six ranked lists extracted (giving a total of 6000). To ensure accuracy, our evaluations were performed by two human judges (annotators), who were well-versed in the domain. They were independently asked to tag the candidates that denoted valid terms as correct. Otherwise, the candidates were marked as incorrect. The two annotators were allowed to inspect the documents to facilitate their decisions. Following Zhang et al. [77], we adopted a strict evaluation mode, in which a candidate was deemed correct only if both annotators

agreed on that decision. These candidates were counted as *true_positive*, while those deemed incorrect by both of annotators were *false_positive.*

The precision scores, P, of the candidates extracted at the different threshold values *t* were then estimated using equation (2.6). They are reported in the 2nd column of Table 2.10. To mitigate the effect of coincidental agreements between the two annotators, we computed the level of inter-annotator agreement using the kappa coefficient [95]. Our calculated kappa values, which hovered around 0.68-0.72, were indicative of a strong level of inter-annotator agreement since kappa values of 0.7 are considered desirable. (For conciseness and space considerations, we do not report on the detailed kappa statistics in our results).

Estimating the recall is more difficult as it entails prior knowledge of the correct terms contained in the corpus. To calculate the recall, we relied on a manually created gold-standard. It consisted of 1000 known terms, identified by both annotators, from a subset of our domain-specific corpus. No additional constraints, pertaining to the length (number of words) or frequency, were imposed on the terms at this stage. We will perform separate experiments in Sections 2.6.7 to investigate the effect of term length and frequency on the performance.

This sub-corpus was then analyzed by the various phases of ExtTerm, as described in the previous sections. Candidates extracted by ExtTerm that were also found in the gold-standard were *true_positive*. Terms in the gold-standard that ExtTerm failed to extract were *false_negative*. The recall scores, R, for the different threshold values *t*, were then computed using equation (2.7). They are reported in the 3rd column of Table 2.10.

$$Precision = \frac{true\_positive}{true\_positive + false\_positive} \tag{2.6}$$

$$Recall = \frac{true\_positive}{true\_positive + false\_negative} \tag{2.7}$$

To obtain a single performance value, we determined the F-score [96]. Since we are interested in balancing precision (P) and recall (R), we set the weighing factor, β, of the F-score to 1. The F-score is then referred to as the F1-score, and is computed using equation (2.8). These scores are presented in the 4th column of Table 2.10.

$$F1 = \frac{2 \times P \times R}{P + R} \tag{2.8}$$

Threshold *t*	**Precision**	**Recall**	**F1**
10	0.79	0.96	0.86
100	0.84	0.90	0.87
200	0.87	0.89	0.88
300	0.89	0.84	0.87
400	0.87	0.81	0.84
500	0.87	0.80	0.84

Table 2.10: ExtTerm's performance scores at different threshold values.

The highest F1 score of 0.88, which reflected the optimal balance between precision and recall, was obtained at *t*=200. It indicated that the candidate set harvested from the RTS phase (Section 2.6.3) at *t*=200 contained the largest number (i.e. high recall) of valid (i.e. high precision) terms. The selected candidates were then submitted to the TR phase (Section 2.6.4), which computed their unithood as described previously. This ensures that

ExtTerm extracted the largest set of valid (relevant and coherent) terms. Therefore in our experiments, we used t=200.

2.6.6 BENCHMARKING AGAINST BASELINE

To benchmark the performance of ExtTerm, we compared its outputs against the C-value baseline. We manually inspected the top-1000 candidates generated by the baseline, and estimated the corresponding precision, recall and F1 scores. These values are presented in Table 2.11. In this table, we also repeat the corresponding performance scores of ExtTerm for ease of comparison.

	Precision	**Recall**	**F1**
Baseline	0.71	0.84	0.77
ExtTerm	0.87	0.90	0.88

Table 2.11: Performance of ExtTerm vs. Baseline.

As can be seen from the above table, ExtTerm achieved higher F1, precision and recall scores than the baseline. These results indicate that ExtTerm extracted a larger set of valid terms from our domain-specific, sparse and informally-written documents. Next, we inspected the candidates extracted by the baseline to investigate the cause of its lower performance compared to ExtTerm.

Lower Precision of Baseline

The lower precision of the baseline was attributable to the issue of noise. It extracted frequently occurring candidates, a significant portion of which were either irrelevant expressions, such as *"meeting room"* or incoherent word sequences, such as *"customer helpdesk collimator shutter"*. As we saw before, ExtTerm employs a corpus-comparison approach for identifying relevant and irrelevant candidates based on their termhood. It ensures that irrelevant candidates are awarded low termhood, and subsequently discarded, regardless of whether they occur frequently. Also, ExtTerm formulates the unithood of a candidate as function of its sub-expressions. In this way, invalid candidates, which are generally composed of other invalid sub-expressions, are awarded lower unithood scores and discarded.

Lower Recall of Baseline

The lower recall of the baseline was due to the issue of silence. It failed to detect many of the valid terms, which occurred sparsely (e.g. less than 5 times) in the entire domain-specific corpus (Challenge 1). ExtTerm, on the other hand, overcomes this difficulty by estimating the termhood of the candidates, and selecting only those that are closely relevant to the domain, regardless of their (absolute) frequencies.

Another factor responsible for the lower recall of the baseline is that its linguistic filter failed to detect some complex terms, such as *"rear battery cabinet coaxial cable"*, and those that contained POS-tagging errors, e.g. *"regulating switch"*.

2.6.7 INFLUENCE OF TERM FREQUENCY AND LENGTH

The frequency and length of terms are two of the most influential factors that impact the performance of TE algorithms [37, 41, 93, 77]. Our previous results have shown that ExtTerm successfully extracted terms regardless of their length and frequency. For example, we could detect extremely long terms, containing 5 words or more, as well as infrequent terms, which occurred only once or twice in a corpus. In this section, we performed additional experiments to investigate the effect of the term length and frequency on the performance.

Effect on Term Frequency on Precision

Our experimental procedure was similar to that of Piao et al. [93]. We divided the top-1000 candidates output by ExtTerm (at t=200) and by the baseline into 10 frequency bands (buckets). They are listed in the first column of Table 2.12. The bands were

deliberately made more granular (i.e. fine-grained) at lower frequencies for two reasons. First, we were interested in the performance in detecting low frequency terms, especially those occurring less than five times, which most TE algorithms discard [37, 59, 81]. Second, low frequency expressions constituted the bulk of terms in our specialized corpus.

Candidates in each frequency band were analyzed by our two human annotators, who marked them as *true_positive* if they were correct terms, and *false_positive* otherwise. Then, we estimated the precision per frequency band according to equation (2.6). The precision scores for ExtTerm and the C-value baseline are reported in the 2nd and 3rd columns of Table 2.12.

Frequency (f)	ExtTerm Precision	Baseline Precision
1	0.85	0.65
2	0.86	0.65
3	0.86	0.66
4	0.87	0.67
5-6	0.86	0.69
8-10	0.87	0.72
11-30	0.88	0.74
31-60	0.88	0.75
61-100	0.88	0.76
>=100	0.88.	0.76

Table 2.12: Effect of term frequency on performance.

The results reveal that the baseline's precision fluctuated widely across the different frequency bands. As expected, it performed reasonably well in detecting frequent terms, for e.g. those with frequencies of more than 30 ($f > 30$). However, its performance dropped drastically for rare terms, as reflected by its lower precision in extracting terms occurring less than five times ($f < 5$). On the other hand, the precision of ExtTerm remained stable across the different frequency ranges, indicating that it was equally precise in extracting rare terms as well as those occurring more frequently.

Effect on Term Length on Precision

To investigate the effect of the terms' length on the performance, we divided the top-1000 candidates output by ExtTerm and by the baseline into five different groups according to their lengths. They are shown in the first column of Table 2.13. Single-word terms were excluded. We then measured the precision scores of ExtTerm and the baseline for terms in each length-group. These scores are presented in the 2nd and 3rd column of Table 2.13.

Length	ExtTerm Precision	Baseline Precision
2	0.87	0.86
3	0.87	0.86
4	0.88	0.86
5	0.87	0.70
>= 6	0.88	0.65

Table 2.13: Effect of term length on performance.

Both ExtTerm and the baseline achieved reasonable performance for terms with two and three words. However, the baseline's performance degraded for longer terms, and worsened as the term length increased. This was due to the difficulties in distinguishing valid (complex) terms from invalid word sequences. The performance of ExtTerm, on the other hand, remained constant across the different term lengths, suggesting that it was equally precise in extracting longer complex terms (e.g. containing 5 or 6 words) as well as shorter ones (e.g. containing 2 or 3 words).

2.7 CONCLUSION

In this chapter, we were concerned with our first research question, *RQ1: How to accurately identify and extract terms from domain-specific, sparse and informally-written corporate texts?*

We addressed this question by developing the ExtTerm framework for term extraction. The term extraction strategy adopted by ExtTerm is based on a hybrid approach, combining both linguistic and statistical information.

Similar to most other term extraction techniques, we acquired linguistic information from syntactic features, namely, Part-of-Speech (POS). However, our domain-specific texts were informally-written, and were rife with POS-tagging errors. One common source of errors involved gerunds (nouns) that were incorrectly POS-tagged as progressive verbs. We found out that conventional linguistic filters, commonly employed by previous term extraction studies, were susceptible to these types of POS-tagging errors. In addition, many of our domain-specific terms had complex syntactic structures. We overcame these difficulties by designing a new filter, which was hinged upon the notion of base-terms and on the mechanism governing the synthesis of these base-terms into longer, more complex terms. We also incorporated an additional component in our filter so that it could detect gerunds that were incorrectly POS-tagged as progressive verbs. It was therefore less susceptible (more robust) than conventional filters, and could also detect the complex domain-specific terms with greater accuracy.

Another challenge that we faced with our sparse domain-specific corpus was that of extracting low frequency terms. We addressed this challenge by adopting a corpus-comparison approach. The main strength of such an approach was that it was independent from the terms' absolute frequency. Instead, it considered the terms' relative frequencies (i.e. frequency ratio) across the domain-specific corpus and a general corpus, which in our case was Wikipedia. In essence, we computed the termhood, which is the most fundamental property of terms. With this approach, term candidates that were more likely in the domain-specific corpus than in the general corpus were awarded higher termhood scores and could qualify as valid terms, even if they occurred with extremely low frequency in the domain-specific corpus. Another benefit of the corpus-comparison approach was that it enabled us to discard irrelevant expressions that occurred frequently in the domain-specific texts. Since these expressions were likely to appear in the general corpus, they were awarded lower termhood scores, and rejected.

A large number of terms in our domain-specific texts were realized as multi-word expressions, often consisting of more than 3 words. Standard Lexical Association Measures (LAMs), designed for expressions that contain 2 words, were therefore inadequate to detect these domain-specific multi-word terms. To accurately detect these terms, we relied on the mechanism governing the formation of multi-word terms from base-terms. Multi-word terms were decomposed into corresponding sub-expressions. Then, we recursively estimated the unithood scores of these sub-expressions. We started with those of length 2 since their unithood scores could be easily computed using traditional LAMs, which in our case was the cube mutual information. We formulated the unithood of a multi-word term as a function of the unithood of its sub-expressions. Since the sub-expressions of valid terms were likelier to be terms themselves, our strategy ensured that valid (multi-word) terms were awarded much higher unithood scores than invalid ones, which facilitated their identification.

We evaluated our ExtTerm framework on a collection of real-life, domain-specific texts provided by our industrial partners. We also compared it against the C-value algorithm, a state-of-the-art term extraction technique. From our experimental results, we saw that ExtTerm outperformed the C-value baseline. Specifically, C-value's filter could only extract fragments of terms, while that of ExtTerm was able to detect the entire terms. Also, ExtTerm's filter detected terms, even if they contained POS-tagging errors and had complex syntactic structures. These terms were missed by C-value, and rejected by its linguistic filter at a premature stage. We also found out that ExtTerm was able to detect

rare terms, even those occurring with extremely low frequencies, for example, only twice, in the domain-specific corpus. Conversely, the C-value baseline was often unable to detect these rare terms on account of their low frequencies. Concerning the extraction of multi-word terms, ExtTerm successfully identified terms with 3 or more words, and could precisely discriminate between valid multi-word terms and incoherent word sequences that were rife in our domain-specific corpus. C-value was also relatively successful in detecting multi-word terms. However, it also extracted a large number of incoherent words sequences, which were invalid terms. In our additional experiments, we observed that ExtTerm's performance was relatively stable when extracting terms of different frequencies and lengths. These experiments further confirmed that traditional algorithms like C-value have difficulties in detecting low frequency terms and in discriminating between valid multi-word terms and invalid word sequences.

The main findings/conclusions of this chapter can be summarized as follows:

1. A large number of terms in domain-specific corpora are realized as multi-word expressions, and cannot be accurately detected using standard LAMs.
2. Domain-specific terms tend to have relatively low frequencies. Most existing algorithms fail to identify these terms due to the lack of significant statistical evidence.

These above 2 findings, pertaining to the frequent occurrence of rare terms and to the prominence of multi-word terms in domain-specific corpora, have also been reported in several prior studies in terminology, which we described earlier.

In addition, we can also note that:

3. The corpus-comparison approach is an efficient technique for detecting low frequency terms based on their termhood. The relative ease with which it can be implemented makes it even more attractive for use in term extraction endeavors.
4. The unithood of a multi-word term can be formulated as function of the unithood of its sub-expressions. This makes it easier to compute the unithood of terms that contain more than 2 words.
5. Termhood can be considered as a more fundamental property than unithood for identifying terms in texts. A general or irrelevant expression may have a high unithood even if it is not a valid term, in which case it will have a low termhood.

Further discussions on our term extraction framework with respected to RQ1 will be provided in Chapter 8.

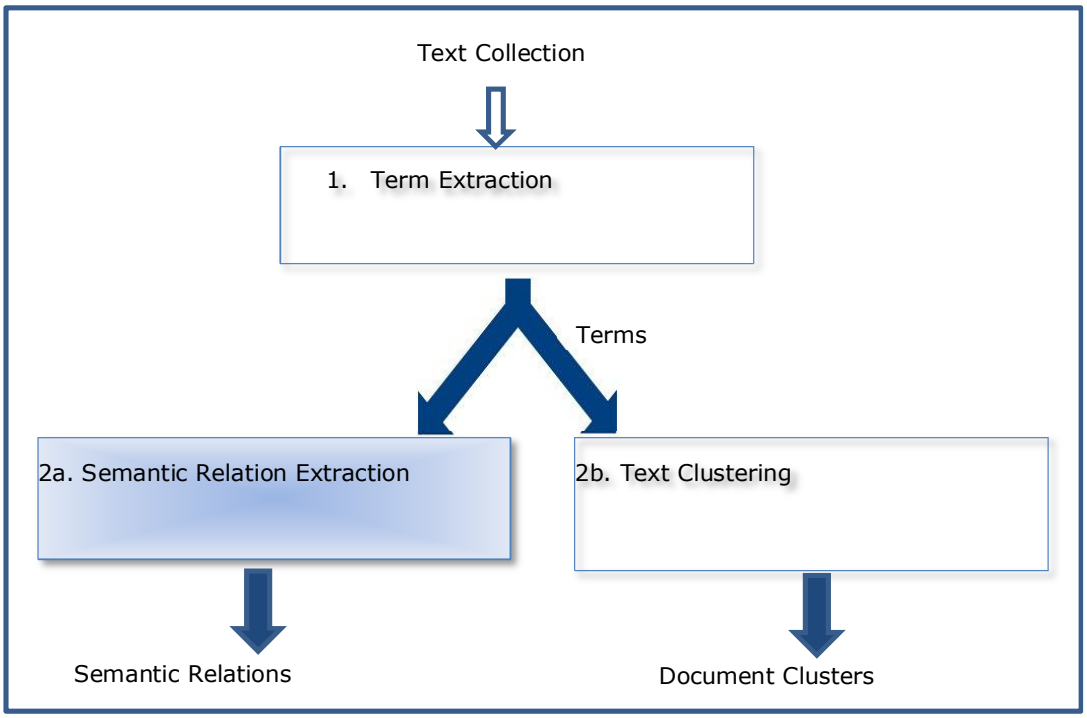
Concerning the business applications, ExtTerm can be used by data analysts to identify frequently malfunctioning products from repair notes of engineers. Future development effort can then be devoted towards improving the quality of these products. It is also a useful application for marketers to detect (probe) the moods or opinions of customers from online forums, and to react accordingly, for example, by targeted advertising.

PART III - SEMANTIC RELATIONS

PREAMBLE

In Part III of the thesis, we will address the second research question *RQ2: How to accurately detect occurrences of semantic relations from domain-specific, sparse and informally-written corporate texts?*

As mentioned earlier, this corresponds to the task 2a in our application pipeline, shown below, which we need to execute for realizing our overall aim of extracting meaningful information from corporate texts.



We have split up our work in addressing the second research question, RQ2, into 4 different chapters to make it more manageable and comprehensible.

In Chapter 3, we present an overview of semantic relations in general. We will also motivate our interest in two types of relations, namely part-whole and causality, which are investigated in the subsequent chapters.

The gist of our proposed solutions in response to RQ2 is contained in Chapters 4 and 5, which respectively deal with the extraction of part-whole and causal relations from domain-specific texts.

Chapter 6 reports on additional experiments to investigate a peculiar behavior of our proposed solutions, which was observed when mining part-whole and causal relations.

It can be seen that all these 4 chapters deal with the common theme of extracting semantic relations from domain-specific texts. Hence, we could have, and in fact, did try to bundle them together. However, it resulted in a single, bloated chapter, which not only was difficult to read, but made it even more challenging for us to formulate our aims and contributions and to present our results. Subsequently, we decided to adopt the structure as shown above. Therefore, the reader can expect some overlapping sections across the different chapters. Wherever possible, repetitive materials have been omitted or presented in a very concise way.

CHAPTER 3 –AN OVERVIEW OF SEMANTIC RELATIONS

3.1 INTRODUCTION

Concepts, lexically manifested as terms, are considered as the basic building blocks of knowledge [26]. For example, as we saw in Chapter 2, a term like "radiation protection shield arm" represents a concept (in this case, a product) in the domain of Product Development – Customer Service (PD-CS). However, terms by themselves convey a limited amount of information. For example, the occurrence of the term "(blinking) monitor" in a collection of engineers' repair notes only indicates that this particular product experienced a malfunction. Corporate activities demand richer and more comprehensive information, such as the factors responsible for product failures, the events triggered by malfunctioning product components or the location of defective components in a product.

This kind of information can be obtained from semantic relations, which are meaningful associations that exist between terms in texts [26]. In the PD-CS domain, for instance, the terms "(defective) proximity sensor switch" and "(blinking) monitor" in "defective proximity sensor switch found to be *the cause of* blinking monitor" participate in a cause-effect (or causal) relation. In this example, the relation is established by the pattern "(found to be the) *cause of*". The relation provides us with additional information on the cause (i.e. "defective proximity sensor switch") of a product failure (i.e. "blinking monitor").

From the above illustration, we can see that a term's relationship with other terms (or words) is part of its meaning. For instance, the meaning of "(defective) proximity sensor switch" when it is related to "(blinking) monitor", as shown above, is that of a causal agent, responsible for a product failure. This observation is in fact deeply-rooted in structural linguistics. According to Ferdinand de Saussure, for example, *"La langue est un système dont tous les termes sont solidaires et où la valeur de l'un ne résulte que de la présence simultanée de l'autre"*, i.e. "language is a system of interdependent terms in which the value of each term results solely from the simultaneous presence of the others" [97]. Lyons [98, 99] also asserted that, in general, "words" (including terms) cannot be simply defined independently of others.

3.2 REPRESENTING SEMANTIC RELATIONS

Semantic relations can also be used to represent the associations between ideas that exist in the semantic memory (mind) of human beings. Chaffin and Herrmann [100], for instance, noted that "relations have long been viewed as basic to thought, language, comprehension and memory". Thus, they play an important role in defining how we represent knowledge psychologically.

However, in this thesis, we are concerned with semantic relations that are lexically manifested in texts between terms, and not with their mental encoding. Our relations of interest are often known as lexical-semantic relations. We will refer to them as semantic relations for short.

A semantic relation can be represented as a triple, consisting of a pair of terms, which participate in the relation, and of a lexical/syntactic (lexico-syntactic) pattern, which establishes the relation between the terms. That is, for a given relation *R*, it can be

formalized as <*pattern, term1, term2*>, where *term1* and *term2* denote a pair of instances/concepts. Each relation can then be viewed as consisting of 2 slots (*term1* and *term2*), which can be filled by some arguments (terms). For example, in a part-whole relation, the 2 slots can be occupied by the terms "engine" (part) and "car" (whole), as in the "car consists of (an) engine", corresponding to the triple <"consists of", "engine"", "car>".

3.3 ARITY OF RELATIONS

In our work, we will be concerned only with binary relations, which connect exactly two terms (i.e. a term pair), as exemplified in the illustrations provided this far. However, it should also be noted that relations with lower or higher degrees of arity also exist.

Sowa [101] proposed the notion of unary (monadic) relations, which involves verb tenses and modalities (e.g. possibility, necessity). For example, verbs in the past tense can be thought of as instantiating a PAST relation to indicate that certain events occurred (were true) in the past.

Other relations may also involve more than 2 arguments. The "buy" relation [26] is an example of a 4-ary relation, involving 4 arguments, viz. the product being purchased, the amount paid, the seller and the buyer[10]. It has also been proven that relations with arity of more than 2 can always be decomposed into binary relations [26].

Semantic relations need not always connect terms, but can also exist between larger units of texts. A well-known example of such a relation is that of (textual) entailment, which exists between sentence pairs. At an even higher level are discourse relations, which exist between entire segments of texts. However, in this thesis, we will only deal with semantic relations between term pairs.

3.4 TYPES OF SEMANTIC RELATIONS

Semantic relations can exist between any pairs of terms. This gives rise to an extremely large number of potential relations in a given domain.

One of the first attempts to enumerate different types of semantic relations was by Aristotle and his work entitled Metaphysics [102]. In this study, various relations, such as hyponymy (relates sub-concepts to their concepts), part-whole and causality, as well as their attributes were described. Another such study is that of Evens et al. [103], which described semantic relations commonly employed in anthropology, linguistics, psychology and computer science. A more recent survey can be found in the work of Auger and Barriere [22]

In this thesis, we will investigate two types of semantic relations, namely, part-whole and causality. Our decision to concentrate on these 2 relations was motivated by 3 main reasons.

1. Part-whole and causal relations convey valuable operational knowledge that business organizations in the PD-CS domain can exploit to support their activities for improving product quality and customer satisfaction. Examples to illustrate the significance of part-whole and causal relations will be provided in Chapters 4 and 5.
2. Part-whole and causal relations are considered as universal relations. Previous research [104, 105] in anthropology and psychology have shown that universal relations (which also include antonymy) are easily recognized and used with equal frequency by diverse groups of people, regardless of their cultural background.

[10] These arguments are often expressed as semantic roles, such as patients and agents.

3. The part-whole and causal relations have almost always been included by the numerous surveys, mentioned previously, which have attempted to enumerate the different relation types.

3.5 PART-WHOLE AND CAUSAL RELATIONS EXTRACTION

We will now provide a brief overview of some existing techniques that have been developed for automatically extracting part-whole and causal relations from texts. This section lays the foundations for Chapters 4, 5 and 6.

Most of the techniques for mining part-whole and causal relations proposed to date can be broadly classified into two categories depending on their adopted approach. The first group consists of supervised techniques, such as the works of Girju et al. [106, 107] and of Beamer et al. [108] for extracting part-whole and causal relations respectively. Supervised techniques are trained on annotated data, labeled with examples of the target relation (e.g. part-whole or causality). After training, they use their acquired knowledge to identify new occurrences of the target relation. Other supervised techniques also rely on hand-crafted patterns that express the target relation, such as the pattern *"cause of"* for causality. These patterns and the term (instance/event) pairs that they connect in a corpus are then extracted as relation triples. An example of such a technique is that of Khoo et al. [109] for mining causal relations.

The second category of relation extraction techniques consists of minimally-supervised algorithms, such as the Espresso algorithm of Pantel and Pennacchiotti [110]. Espresso has been employed for mining various types of semantic relations, including those between parts and wholes. The main strength of minimally-supervised algorithms is that they alleviate the need for labeled training data. Thus, they represent a more attractive alternative to their supervised counterparts, especially in domains where annotated training data are not available and too expensive to create. These algorithms are initialized with instance/event pairs (e.g. "engine-car" or "hiv-aids"), called seeds, which instantiate the target relation (e.g. part-whole or causality). Then, they iteratively extract the surface-patterns (e.g. "*consist of*" or "*cause of*") that connect the pairs in a corpus. The most reliable patterns are bootstrapped and used to acquire other pairs (e.g. "grape-wine" or "smoking-cancer") Then, a recursive process of harvesting new patterns from pairs and vice-versa is triggered, until a suitable number of relations (pattern and/or pairs) have been collected. However, minimally-supervised algorithms, despite their attractiveness, also exhibit various shortcomings. For example, their performance is severely affected by the issue of data sparsity. Also, the surface-strings that these algorithms use to represent their extracted patterns are susceptible to morphological variations, resulting in a large number of redundant patterns (e.g. *"consists of"* and *"consisted of"* for part-whole relations). In addition, the relations ultimately extracted by minimally-supervised techniques may be completely divergent from the target relation of interest. This phenomenon is commonly known as *semantic-drift*, and is detrimental to the algorithms' performance. Semantic-drift is particularly severe when dealing with part-whole and causal relations since these relations are realized by a wide variety of ambiguous patterns, which do not always connect valid part-whole and cause-effect pairs.

We will elaborate on part-whole and causal relations, the challenges that their automatic extraction poses to extant algorithms and our proposed solutions to address the challenges in Chapters 4 and 5. These solutions will constitute our answer to the second research question, RQ2.

CHAPTER 4- PART-WHOLE RELATION EXTRACTION

CHAPTER SUMMARY

In this chapter, we will address our second research question. We will present a novel approach for learning part-whole relations from domain-specific texts. The crux of our approach lies in using Wikipedia as a source of additional information, which enables us to overcome the issue of data sparsity posed by domain-specific documents. We start by acquiring a set of patterns (e.g. *"consist of"*) that reliably express part-whole relations from Wikipedia. This is achieved by a minimally-supervised algorithm, which, as mentioned before, alleviates the need for annotated training data that may not always be available. Also, our algorithm overcomes the issue semantic-drift, which enables it to harvest the most precise patterns and pairs, leading to performance gains. In addition, we encode the patterns in the extracted relations using lexical and syntactic information. This encoding addresses the limitations of surface-strings, employed by most minimally-supervised techniques to represent their patterns.

After acquiring the high quality part-whole patterns from Wikipedia, we then use them to extract (domain-specific) part-whole relation triples from our domain-specific corpus. To evaluate our approach, we estimate its performance on a real-life collection of domain-specific documents. As will be shown by our experimental results, our approach outperforms a state-of-the-art baseline, and extracts more accurate part-whole relations from the domain-specific documents. In addition, we will also demonstrate the performance gains by relying on Wikipedia as a knowledge-base and by effectively mitigating the issue of semantic-drift. Furthermore, our results show that despite its broad-coverage contents, Wikipedia can still be exploited to support domain-specific part-whole relations.

The work described in this chapter is an edited version of the manuscript:

- Ittoo, A; Bouma, G.: "Minimally-Supervised Extraction of Domain-Specific Part-Whole Relations using Wikipedia as Knowledge-Base". It has been conditionally accepted for publication in the journal *Data and Knowledge Engineering Special Issue NLDB 2010.*

It is an extended version of the earlier published book chapter:

- Ittoo, A; Bouma, G; Maruster, L; Wortmann J.C.: "Extracting Meronymy Relations from Domain-Specific Textual Corporate Databases", *in Lecture Notes in Computer Science, Vol. 6177, pp. 48-59, Springer.*

4.1 INTRODUCTION

The relations between parts and the wholes they comprise is of fundamental importance in many disciplines, such as linguistics, cognitive science, and conceptual modeling [111, 112, 113, 114]. In Natural Language Processing (NLP), Information Extraction (IE) algorithms have been developed for automatically extracting occurrences of the part-whole relations from texts [110, 115, 106, 107, 116].

As mentioned in Chapter 3, most existing techniques adopt either a supervised or a minimally-supervised approach to part-whole relation extraction. Supervised techniques [106, 107] are those that require prior training on data annotated with part-whole relations. Minimally-supervised techniques [110], on the other hand, eschew the need for annotated data. They are simply initialized with a handful of seeds, which are instance pairs that instantiate part-whole relations, such as "engine-car". Then, they recursively extract all the part-whole patterns (e.g. *"consist of"*) connecting the seeds, and use the harvested patterns to acquire new instance pairs (e.g. *grape-wine*) and vice-versa.

Techniques (supervised and minimally-supervised) for learning part-whole relations developed to date have mostly focused on broad-coverage (open-domain, general-purpose) text collections, such as the L.A. Times news collection [117]. The task of mining part-whole relations from domain-specific texts has been largely overlooked despite the importance of these relations in several specialized disciplines. In the business/corporate domain of Product Development-Customer Service (PD-CS), for example, the relations between parts and wholes are crucial for activities like product design [116] and quality assurance [114]. More importantly, the information expressed by part-whole relations is valuable to support a wide variety of corporate activities, such as Business Intelligence. For example, the relation between the part "(blown) fuse" and its whole "set-top box", established by the pattern "*found in*", as in "blown fuse *found in* set-top box", enables engineers to efficiently locate and diagnose causes of product failures.

However, the extraction of part-whole relations from domain-specific texts poses a number of challenges that current techniques do not adequately address. These challenges are described below as Challenges 1 to 5. Challenge 1 concerns supervised techniques, while Challenges 2-5 pertain to minimally-supervised techniques.

1. Lack of domain-specific knowledge resources. Supervised techniques rely heavily on domain-specific knowledge resources such as labeled training data, which are often unavailable in domains like PD-CS (Challenge 1).

2. Data sparsity. Unlike supervised techniques, minimally-supervised algorithms alleviate the need for labeled training data, but their performance (e.g. accuracy) is severely compromised by the issue of data sparsity that characterizes domain-specific texts [22, 39, 118] (Challenge 2).

3. Seed selection. Furthermore, selecting suitable seeds from domain-specific texts to initialize minimally-supervised techniques is difficult (Challenge 3).

4. Ambiguous patterns/semantic-drift. Especially in domain-specific texts, part-whole relations are often realized using ambiguous patterns (e.g. *"is in"*), which do not always relate parts to wholes. These inaccurate patterns are responsible for the issue of semantic-drift, whereby the relations extracted by a minimally-supervised algorithm are different from the target relation defined by its initializing seeds [119] (Challenge 4).

5. Surface-patterns. Also, most minimally-supervised techniques represent the patterns in the extracted relations using simple surface-strings, which yields a large number of similar and redundant patterns (Challenge 5).

It should be noted that the aforementioned challenges are general limitations of relation extraction algorithms, which are more acute when dealing with informally-written and

sparse texts and when mining relations that are manifested by ambiguous patterns like part-whole. Hence, our solutions to overcome these limitations, which will be discussed next, are relevant to the field of relation extraction at large.

To address the aforementioned challenges, we develop and present in this chapter an approach for extracting high quality domain-specific part-whole relations from sparse and informally-written texts, typically generated in specialized disciplines. The crux of our approach lies in applying a minimally-supervised algorithm to a large, broad-coverage corpus, which we use as knowledge base. From this knowledge-base, we acquire a set of patterns that reliably express part-whole relations. Then, we extract, from the domain-specific text collection, all triples consisting of the acquired patterns and the (domain-specific) instance pairs they connect. These triples constitute our domain-specific part-whole relations.

Our core contributions are as follows:

1. We present an algorithm that extracts part-whole relations with minimal supervision, thereby alleviating the need for large amount of annotated training data (Contribution 1 in response to Challenge 1).

2. To overcome the issue of data sparsity posed by domain-specific texts, our algorithm relies on additional information that it acquires from a knowledge-base, namely Wikipedia[11] (Contribution 2).

3. Our use of Wikipedia as a knowledge-base also makes it easier to select prototypical part-whole seeds to initialize our algorithm (Contribution 3)

4. We propose a technique to mitigate the negative impact of semantic-drift on the performance of minimally-supervised techniques (Contribution 4).

5. To overcome the limitations of conventional surface-strings, we represent the patterns in the extracted relations using lexical and syntactic (lexico-syntactic) information (Contribution 5).

We will elaborate on these challenges and on our contributions in Sections 4.3 and 4.4 respectively.

Our experiments were conducted on real-life corporate documents provided by our industrial partners. The same corpus had been used previously when describing our ExtTerm framework for term extraction in Chapter 2. The results revealed that our proposed approach accurately extracts part-whole relations from sparse, domain-specific texts with minimal supervision. In addition, we empirically demonstrate the gains in performance by relying upon Wikipedia as a knowledge-base, and by overcoming the effect of semantic-drift. We also show that our approach outperforms the state-of-the-art, minimally-supervised Espresso algorithm [110] in the extraction of domain-specific part-whole relations.

This chapter is organized as follows. In Section 4.2, we describe related work on part-whole relation extraction. In Section 4.3, we highlight the limitations of existing techniques for extracting part-whole relations from domain-specific corpora. We elaborate on our contributions in response to these challenges in Section 4.4. Section 4.5 describes our methodology in details, while our experimental evaluations are reported in Section 4.6. We conclude in Section 4.7.

[11] We refer to Wikipedia as a knowledge-base even though we do not rely on its (semi-)structured contents, but acquire information by from its unstructured textual contents.

4.2 RELATED WORK

In this section, we will briefly review previous efforts concerning the study of part-whole relations. We start by defining part-whole relations and their different subtypes in Section 4.2.1. Then, Section 4.2.2 motivates our interest in learning part-whole relations from domain-specific texts, particularly those generated in the corporate domain of PD-CS. Section 4.2.3 describes existing approaches for extracting part-whole relations from texts. In Section 4.2.4, we describe some knowledge bases that are useful to improve the performance of Information Extraction algorithms, and that can play an important role in overcoming the issue of data sparsity.

4.2.1 PART-WHOLE RELATIONS

A part-whole relation is manifested in text as a pattern, e.g. "*consists of*", which connects a pair of related (part-whole) instances, e.g. "engine-car", as in "car *consists of* engine". Part-whole relations can therefore be formulated as triples, consisting of related instance pairs and the patterns that sub-categorize them, for e.g. <pattern=consists of, part=engine, whole=car>.

Investigations on the part-whole relations have spanned over a plethora of disciplines, including philosophy, cognitive science and conceptual modeling [111, 112, 113, 114]. Each of these different communities introduced their own, sometimes conflicting, definitions of part-whole relations.

To classify the different types of part-whole relations, several taxonomies have been proposed by Winston et al. [111], Gerstl and Pribbenow [112] and Odell [113]. These taxonomies are derived from the linguistic usage of part-whole relations. Hence, they often fail to unambiguously classify the different types of part-whole relations [114].

Keet and Artale [114] developed a formal taxonomy to clarify the semantics of part-whole relations. Their taxonomy is based on well-known foundational ontology principles, and provides a more precise classification. In addition, it encompasses all the different types of part-whole relations mentioned in other taxonomies [111, 112, 113].

The principal distinction in the taxonomy of Keet and Artale [114] (Keet's taxonomy) is between transitive and intransitive part-whole relations. This taxonomy identifies 8 part-whole relation types; 4 transitive and 4 intransitive. Transitive (part-whole) relations are known as *mereological* relations, while intransitive ones are referred to as *meronymic* relations. Subsequent distinctions are made by enforcing semantic selectional restrictions, in the form of Dolce ontology [120] classes, on the instances that participate in the different relation types. For example, in Keet's taxonomy, the *sub-quantity-of* part-whole relation can only connect instance pairs that belong to the class Amount-of-Matter in the Dolce ontology, such as the pair "alcohol-wine". We will describe this taxonomy in more details in Chapter 6.

4.2.2 MOTIVATION FOR LEARNING DOMAIN-SPECIFIC PART-WHOLE RELATIONS

As will be described later, existing Information Extraction (IE) algorithms have predominantly focused on discovering part-whole relations from large, broad-coverage (general-purpose) corpora, such as the L.A. Times collection [117].

A domain in which part-whole relations play an important role is the business/corporate discipline of Product Development-Customer Service (PD-CS). In this domain, part-whole relations extracted from customer complaint texts or repair notes of service engineers

encode valuable operational knowledge that organizations can exploit for product quality improvement.

We use three sample texts, *S1*, *S2* and *S3*, to illustrate the relevance of part-whole relations and their potential applications in Business Intelligence.

Example 1
S1 = "leaking tube *found in* radiator".
In *S1,* the part-whole relation between "leaking tube" (part) and "radiator" (whole) is lexically realized by the pattern "*found in*". This relation is useful to service engineers as it facilitates the diagnosis of malfunctioning products, and enables product failures to be efficiently determined.

Example 2
S2 = "brightness option not *available on* menu console".
In *S2,* the part-whole relation between "brightness option" (part) and "menu console" (whole) is established by the pattern *"available on"*. This relation expresses a customer's dissatisfaction at the "brightness option" not being available as part of the "menu console". Such customer complaints are more accurate but harder to detect than those which are unequivocally expressed, as in "menu console does not work". Hence, part-whole relations, such as the one in *S2*, enable the detection of customer dissatisfaction causes that are implicitly expressed with subtle patterns, such as "*available on"*.

Example 3
S3 = "calibration was *performed as part of* the upgrade".
S3 relates the phase "calibration" (part) to its encompassing process "upgrade" (whole). The part-whole relation between the phase and the process is established by the pattern "*(performed) as part of*". Such relations provide pertinent information that allows management to determine whether service engineers are executing the appropriate repair or servicing actions.

4.2.3 Extracting Part-Whole Relations from Texts

Part-whole relations are a de-facto benchmark for evaluating the performance of general IE algorithms [108, 110]. Other IE efforts have specifically targeted the extraction of part-whole relations from texts [106, 107]. These approaches can be classified either as (fully) supervised or minimally-supervised.

Supervised Approaches
In the approach of Girju et al. [106, 107], patterns expressing part-whole relations between WordNet [19] concept pairs were manually extracted from sentences of the L.A. Times and the SemCor corpora [121], and used to generate a training corpus with positive and negative examples of part-whole relations. Classification rules induced from the training data achieved a precision of 0.81 and a recall of 0.76 in identifying part-whole relations in previously unseen texts.

Van Hage et al. [116] acquired 503 part-whole pairs from dedicated thesauri, e.g. Agrovoc [122], to learn 91 patterns that conveyed part-whole relations. They substituted the patterns' "part" arguments with known entities, and formulated web-search queries. The corresponding "whole" entities were then discovered with a precision of 0.74.

Minimally-Supervised Approaches
Minimally-supervised approaches alleviate the need for labeled training data. They are initialized with instance pairs, called seeds, which denote part-whole relations, e.g. "engine-car". They use the instance pairs to acquire patterns expressing part-whole relations, and in turn, employ the patterns to extract other instance pairs.
Berland and Charniak [115] relied on manually-crafted patterns and on initial seeds denoting "whole" instances, e.g. "building", to harvest the corresponding "part" instances, e.g. "room", from the North American News Corpus (NANC) of 1 million words. They achieved an accuracy of 0.70 over the top-20 results.

The Espresso algorithm of Pantel and Pennacchiotti [110] was initialized with a set of seeds, and selected surface patterns that connected these seeds. The patterns were

bootstrapped and used to infer other part-whole pairs, which were then recursively used to harvest new patterns. Espresso achieved precision of 0.80 in extracting part-whole relations from the Acquaint corpus (TREC-9) of 6 million words.

4.2.4 KNOWLEDGE BASES

Several studies [17, 123, 124] have demonstrated that the performance of relation extraction techniques (and in general, IE algorithms) can be substantially improved by leveraging upon domain or world knowledge. Such knowledge, symbolically encoded in knowledge bases, is a valuable source to complement the (statistical) evidence derived from the target corpus being analyzed. Knowledge bases can therefore play an important role in alleviating the issue of data sparsity, which as earlier mentioned, plagues minimally-supervised algorithms.

Most knowledge bases traditionally employed to support existing algorithms are broad-coverage, such as ontologies like Cyc [123, 124], lexico-semantic dictionaries like WordNet [19], annotated collections of general texts like the ACE [125] and SemCor [121]corpora, and encyclopedic resources like Wikipedia, which we describe below.

Wikipedia as a Knowledge-Base
Recently, the use of Wikipedia as a knowledge-base has received considerable attention from the IE community. This surge in interest can be attributed to the attractive features/characteristics of Wikipedia, particularly, its (semi-)structured contents, which is a rich source of readily available information. For example, Wikipedia's redirect and disambiguation pages have been employed for word sense disambiguation [126]. Wikipedia's system of categories has served as the basis for extracting hypernymy relations[12] [17]. Structured information, available from infoboxes, has been exploited for building large ontologies, such as YAGO [127] and DBPedia [128]. A detailed review of the applications of Wikipedia in Natural Language Processing and Artificial Intelligence can be found in [129].

4.3 PART-WHOLE RELATION EXTRACTION CHALLENGES

It can be seen from the previous sections that the majority of techniques for part-whole relation extraction developed to date have primarily been applied on documents from the general domain (newspaper texts). As discussed in Chapter 1, the extraction of relations from documents in this domain is facilitated by several factors, namely, the large collections of well-written texts, which provide reliable linguistic and statistical evidence, and the availability of resources, such as annotated data. However, these desirable factors are rarely replicated in corporate domains like PD-CS. As a result, the extraction of part-whole relations from domain-specific (corporate) texts introduces several important challenges, which existing techniques do not adequately address. In this section, we will elaborate on these challenges (introduced in Section 4.1), and discuss the shortcomings exhibited by current techniques. We will then describe our contributions to address these challenges in Section 4.4.

Challenge 1: Lack of domain-specific knowledge resources
In most specialized disciplines, domain-specific knowledge resources, such as ontologies or texts annotated with examples of part-whole relations for training supervised algorithms, are not available. Manually labeling the existing documents to create a training dataset is not viable, and is impeded by the knowledge acquisition (KA) bottleneck.

Minimally-supervised techniques alleviate the need for annotated training data. However, they also display various shortcomings, as we describe below in Challenges 2, 3, 4 and 5.

[12] Hypernymy is the inverse of hyponymy. It exists between a concept (class) and a sub-concept (class), such as between "vehicle" and "car".

Challenge 2: Data sparsity
Minimally-supervised algorithms achieve high performance on large document collections, which provide sufficient redundancy and evidence to support their analyses [130, 131]. However, domain-specific texts, generated in specialized disciplines, are of limited size and sparse. Data sparsity not only affects the precision of minimally-supervised algorithms by making their analyses less reliable [22, 39, 118], but is also detrimental to their recall since sparse texts are less likely to contain the wide variety of patterns that realize a relation [39].

One solution as indicated in Section 4.2.3 is to rely on additional information from knowledge bases. However, most readily available knowledge bases, like Wikipedia, deal with general topics. Their exploitation to support domain-specific tasks, if at all possible, is yet to be investigated.

Challenge 3: Seed selection
Selecting seeds to initialize a minimally-supervised algorithm for extracting part-whole relations from domain-specific texts is non-trivial. Seed selection requires proficiency in the domain terminology to ensure that the selected seeds are genuine part-whole pairs. Terminological variations, due to multiple stakeholders (e.g. management, customers, engineers) referring to a single instance (concept) via different terms, must also be resolved. These difficulties are in stark contrast to traditional, broad-coverage corpora, which facilitate seed selection by offering an abundance of archetypal part-whole pairs, such as "engine-car".

Challenge 4: Ambiguous patterns/semantic-drift
Part-whole relations are manifested in texts by a wide variety of patterns, some of which are unambiguous and always connect parts to wholes, such as the pattern *"consist of"*. Other patterns, however, are ambiguous [107, 132]. They do not always connect parts and wholes, and weakly express part-whole relations. An example of such a pattern is *"is in".* It establishes a part-whole relation when it connects "engine" and "car" as in "engine *is in* (the) car"; but not when it relates "love" and "air" as in "love *is in* (the) air". When these ambiguous patterns like *"is in"* are used by minimally-supervised techniques, they promote the extraction of the erroneous pairs that they connect, such as "love-air". These invalid pairs will in turn favor the extraction of other unreliable patterns, such as "*smell in*" as in "love (can be) *smelt in* (the) air". Subsequently, incorrect pairs and patterns will start to dominate the recursive learning procedure of minimally-supervised algorithms, impeding the detection of accurate part-whole relations. This phenomenon is known as semantic-drift, and drastically affects the performance of minimally-supervised techniques [119].

Challenge 5: Surface pattern representations
Traditional minimally-supervised techniques, such as Espresso [110], DIRPE [133] and Snowball [134] represent the patterns that connect instance pairs using simple surface strings. However, such surface pattern representations are susceptible to variations in word-ordering and morphology (e.g. inflected word forms due to verb tenses or singular/plural counts). For example, consider three sentences, *S4*, *S5, S6* that express a part-whole relation between the pair "door-car".

- *S4* = "a car *consists of* at least two doors".
- *S5* = "all cars *consist of* doors".
- S6 = "a car consists of a number of doors".

Traditional algorithms will extract three distinct surface patterns, namely, "*consists of at least two*", "*consist of*" and "*consists of a number of*", from these sentences. Such a formalization is clearly inefficient since in all the three sentences, the part-whole relation between "door" and "car" can be adequately expressed using the single pattern "*consist of*". Thus, surface-string representations hinder the extraction of the most general patterns that connect instance pairs. They tend to identify a large number of similar and redundant patterns, which have to be manually aggregated.

Another issue with surface-string representations is their inability to capture long-range dependencies between related instances and patterns, which do not appear in adjacent positions in surface texts. One example of a long-range dependency is the part-whole

relation between the instances "poem" and "stanza", established by the pattern "*consisting of*", in the sentence *S7*.

- *S7* = "a poem is a literary piece of work, usually *consisting of* one or more stanzas".

4.4 CONTRIBUTIONS

The preceding section highlighted the lacunae in extant techniques developed for mining part-whole relations. To fill this gap, and to address our second research question, we developed and implemented a novel framework for part-whole relation extraction. Compared to other techniques, our framework presents a number of novel aspects, which enable it to successfully overcome the intricacies posed by domain-specific, sparse and informally-written documents

Our main contributions in response to the aforementioned challenges are:

1. We present an algorithm for automatically extracting high quality part-whole relations from texts with minimal supervision. Our approach is based on the Espresso algorithm of Pantel and Pennacchiotti [110]. It eschews the need for large amounts of training data, manually annotated with examples of part-whole relations. Instead, it is able to accurately extract part-whole relations solely by relying on a handful of instance (term) pairs that it takes as input (Contribution 1 in response to Challenge 1).

2. To overcome the issue of data sparsity posed by the domain-specific texts, our algorithm first acquires a set of patterns that accurately express part-whole relations from a large corpus, which we use as a knowledge-base. In our approach, we employ the English Wikipedia collection as the knowledge-base. We then use the reliable patterns harvested from the knowledge-base to identify occurrences of part-whole relations from the domain-specific texts. This also shows that despite its broad-coverage, Wikipedia can still be exploited as a knowledge-base to support domain-specific relation extraction (Contribution 2).

3. The use of Wikipedia as a knowledge-base also addresses the difficulties involved in selecting suitable seeds from domain-specific texts. Wikipedia, being an encyclopedic resource, abounds in prototypical part-whole pairs that can be used as seeds to initialize minimally-supervised techniques (Contribution 3).

4. To mitigate the issue of semantic-drift, we propose a technique that prevents invalid part-whole pairs (e.g. "love-air"), which are connected by ambiguous part-whole patterns (e.g. *"is in"*; "love *is in* the air"), from triggering the issue of semantic-drift (Contribution 4).

5. We represent the patterns (in the relations extracted by our algorithm) using lexico-syntactic information. As described by Stevenson and Greenwood [135], these types of lexico-syntactic patterns, which abstract from surface-texts, overcome the limitations of traditional surface-string representations, and are beneficial to the performance of relation extraction tasks (Contribution 5).

Our proposed approach for extracting domain-specific part-whole relations is discussed in the next section.

4.5 Framework for Domain-Specific Part-Whole Relation Extraction

In this section, we elaborate on our proposed approach for extracting part-whole relations from domain-specific texts. We start with a general overview of our overall methodology and the framework in which it is realized before describing the various phases.

4.5.1 Architecture Overview

Our methodology for extracting relations contrasts significantly with previous techniques. Instead of extracting part-whole relations directly from the target (domain-specific, sparse) corpus, we acquire a set of reliable patterns that express part-whole relations from the much larger and broad-coverage Wikipedia corpus, which serves as a knowledge-base (Section 4.5.2). As mentioned before, such a strategy enables us to overcome the data sparsity issue of our domain-specific texts. However, part-whole relations are not explicitly available from Wikipedia's (semi-)structured contents. They have to be mined from its unstructured texts.

Our problem is now one of extracting patterns that express part-whole relations from Wikipedia. This is accomplished as follows. During the Pattern Induction and Formalization phase (Section 4.5.3), we convert each Wikipedia sentence into a corresponding relation triple, consisting of a pattern and a pair of instances. To determine which of the patterns (and pairs) are reliable indicators of part-whole relations, we employ a minimally-supervised algorithm. It is initialized with a set of seeds (Section 4.5.4), and iterates between two phases, namely Pattern Selection (Section 4.5.5) and Instance Selection (Section 4.5.6), until a suitable number of patterns is collected.

Since the part-whole patterns acquired from the much larger Wikipedia knowledge-base are bound to occur in the much smaller and sparse target corpus, we extract as domain-specific part-whole relations all occurrences of these patterns and the instances they connect in the latter target corpus (Section 4.5.7). The overall architecture of our framework is illustrated in Figure 4.1. Blank arrows represent inputs and outputs, while filled arrows depict the processing performed by the various phases of our framework.

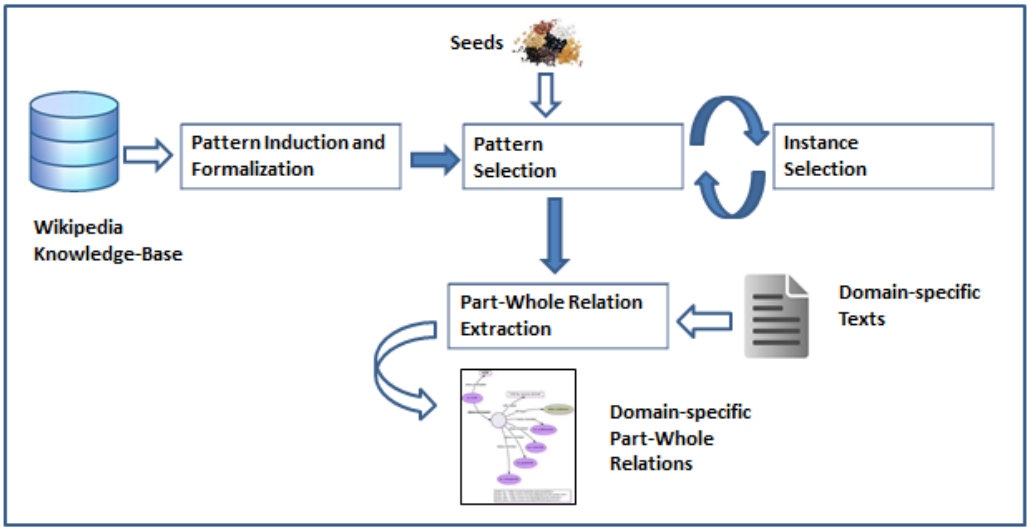


Fig. 4.1: Framework for domain-specific part-whole relation extraction.

4.5.2 Wikipedia as a Knowledge-Base

Broad-coverage knowledge resources present several characteristics that make them suitable for incorporation in Information Extraction (IE) frameworks, as we discussed in Section 4.2.3. In the absence of domain-specific resources, it is worth investigating

whether they can be exploited to support our task of (domain-specific) part-whole relation extraction.

Lexico-semantic dictionaries, e.g. WordNet [19] and corpora like ACE [125], are resources that provide explicit descriptions of part-whole relations, and therefore, they can be used as knowledge bases. These relations can also be easily extracted, for e.g. by traversing the WordNet graph or by relying on labels in the ACE documents. However, WordNet merely lists related instance pairs, without providing any information on the patterns (e.g. *"consist of"*) that relate them in part-whole relations. Also, due to its relatively small size, the ACE corpus may fail to capture the wide variety of linguistic patterns that express the relations between parts and wholes [132].

Other candidate resources, which can be leveraged upon, include the British National Corpus (BNC) [67] and Wikipedia [91].

Both Wikipedia and the BNC exhibit certain desirable features, namely:

- Readily Available: The readily availability of Wikipedia and BNC overcomes the knowledge acquisition (KA) bottleneck that would be involved had resources of similar scale been manually constructed from scratch.
- Representative: Both Wikipedia and the BNC consist of well-written and grammatically sound documents. Their contents are expressed in a conventional style, and depict the general characteristics of the English language. Thus, these two resources can be considered as reference corpora. They are likely to contain the typical expressions that are used in English to express part-whole relations.

However, compared to BNC, Wikipedia offers a number of additional desiderata, which make it particularly attractive as a knowledge-base to support our task of minimally-supervised part-whole relation extraction from domain-specific texts. These desiderata include:

- Broader coverage: Wikipedia is an encyclopedic resource, with its contents spanning diverse topics. As indicated in [136], it provides a more explicit description of relations between real-world entities than BNC. Wikipedia's broad coverage is important to our approach for two main reasons. First, it is likelier to contain the diverse patterns that express part-whole relations, which helps in overcoming the data sparsity issue of our domain-specific texts, and improves the recall (coverage) of our approach. Second, it abounds in typical part-whole pairs that can be used as seeds to initialize our minimally-supervised algorithm.
- Larger size: Related to the broader coverage of Wikipedia is its larger size. The Wikipedia corpus we employ in our proposed framework consists of around 500 million words, which compares favorably with the 100 million words of the BNC. It offers ample redundancy, whereby the same information (e.g. part-whole pairs or patterns) is repeated at multiples places. Consequently, significant statistical evidence can be derived from its contents. This evidence enables us to improve our precision in detecting part-whole relations. Thus, by virtue of its larger size, Wikipedia is a better candidate than BNC as a knowledge-base. Furthermore, as demonstrated in the study of Mihalcea and Csomai [137], the performance achieved in word sense disambiguation when Wikipedia was used as a knowledge-base was higher than the performance with BNC.
- Multi-lingual texts: Besides English, Wikipedia offers a plethora of well-written texts in many other languages, which broadens its application scope for multi-lingual IE. For example, simply switching the English Wikipedia collection in our framework with, say, the French or Dutch version, enables us to investigate manifestations of the part-whole relations in these languages, and possibly compare them with their corresponding occurrences in English. A detailed study on part-whole relations extracted from the English and Dutch Wikipedia can be found in our previous work [138]. It will be presented in Chapter 6.
- Accurate: Wikipedia's contents are regularly maintained and updated, ensuring their accuracy and trustworthiness [129].
- Maturity: Wikipedia has been employed in a host of NLP applications, as described in Section 4.2.3 and in the study of [129].

In our experiments, to be presented in Section 4.6, we will empirically compare Wikipedia and the BNC to further illustrate that Wikipedia is indeed more suitable as a knowledge-

base than BNC. A possible downside of relying on Wikipedia for learning part-whole relations is that these relations are not explicitly available in Wikipedia's (semi-)structured contents. They have to be extracted from the unstructured contents, as will be described in Sections 4.5.3 to 4.5.6.

4.5.3 PATTERN INDUCTION AND FORMALIZATION

In Pattern Induction and Formalization, we transform the sentences in our Wikipedia knowledge-base into corresponding relation triples, consisting of a pair of instances and the patterns that connect them in the sentences. In essence, this phase converts the unstructured texts of Wikipedia articles into a structured representation, more amenable for the automatic identification of relations.

Linguistic Pre-processing and Term Identification

We start by determining the Part-of-Speech (POS) of the tokens (e.g. words) contained in Wikipedia's sentences. As already discussed in Chapter 2, the POS of a token corresponds to its class, for e.g. adjective, noun or verb. This information is inferred using the Stanford POS-tagger [89], which automatically annotates (tags) each token in a sentence with its corresponding POS.

Figure 4.2 illustrates two sentences, *S8(a)* and *S9(a)*, whose tokens have been POS-tagged in *S8(b)* and *S9(b)*. Tokens are separated from POS-tags with the pipe ("|") delimiter. POS-tags are as defined in the Penn Treebank [90]. For e.g. ""JJ" denote adjectives, "NN" represents (common) nouns, and "DT" determiners.

```
S8
a)There is broad scientific consensus that HIV is the cause of AIDS .
b)There|EX is|VBZ broad|JJ scientific|JJ consensus|NN that|IN HIV|NNP is|VBZ the|DT
  cause|NN of|IN AIDS|NNP

S9
a)The poem consists of five stanzas written in terza rima
b)The|DT poem|NN consists|VBZ of|IN five|CD stanzas|NNS written|VBN in|IN terza|NN rima|NN
```

Fig. 4.2: Sample sentences with POS-tags.

Next, we identify terms from the contents of Wikipedia. Since Wikipedia is a broad-coverage (general-purpose) corpus, this is achieved not by our ExtTerm framework of Chapter 2, but by the Termight algorithm of Dagan and Church [57]. We chose Termight since it is one of the most commonly used techniques in NLP literature, and has been found to achieve reasonable performance in term extraction from different domain texts [37, 139]. To identify terms, Termight relies on the Part-of-Speech (POS) information, which we derived as shown in Figure 4.2. Then, it selects as terms any nouns or noun sequences, such as "stanza", "poem", "aids", and "hiv" of Figure 4.2. Each term identified is considered as an instance[13].

Pattern Formalization

After detecting occurrences of instances (terms) in the sentences of Wikipedia, we identify the linguistic patterns that connect them in order to generate our desired relation triples. The traditional surface pattern representations, employed in other minimally-supervised approaches like Espresso [110] and Snowball [134], exhibit a number of limitations, as discussed in Section 4.3. For example, they are susceptible to word order and morphological variations, leading to a large number of similar and redundant patterns. Also, they are unable to capture long range dependencies (Challenge 5).

[13] We do not distinguish between instances and concepts, and use them interchangeably. Hence, a term represents a concept/instance.

A solution to overcome the limitations of surface patterns is to rely on syntactic patterns, derived from the dependency trees of (syntactically) parsed sentences. In the dependency tree of a sentence, nodes correspond to the tokens (e.g. words). They are connected by edges (paths), depicting their syntactic associations, such as subject-verb-object. For example, the dependency tree obtained by syntactically parsing the sentence *S10* = "The church is build in a mostly Romanesque style and consists of three naves" is shown in Figure 4.3.

```
S10
The church is build in a mostly Romaneque style and consists of three naves

(ROOT
  (S
    (NP (DT The) (NN church))
    (VP
      (VP (VBZ is)
        (VP (VBN build)
          (PP (IN in)
            (NP (DT a) (RB mostly) (JJ Romanesque) (NN style)))))
      (CC and)
      (VP (VBZ consists)
        (PP (IN of)
          (NP (CD three) (NNS naves)))))))
```

Fig. 4.3: Dependency tree from sentence.

Different types of patterns can be derived from dependency trees, as was shown by Stevenson and Greenwood [135]. In our approach, we represent the pattern that connects a pair of related instances as the shortest path between the instances in the dependency tree. We refer to such a path as a lexico-syntactic pattern since it is composed of both lexical and syntactic features. To generate dependency trees from the Wikipedia sentences, we use the Stanford syntactic parser [13].

Examples of lexico-syntactic patterns, which we derive from the sentences *S8, S9* and *S10* are shown in Figure 4.4.

```
S8: There is broad scientific consensus that HIV is the cause of AIDS
Lexico-syntactic pattern: hiv+nsubj < cause > prep+of+pobj+AIDS

S9: The poem consists of five stanzas written in terza rima
Lexico-syntactic pattern: poem+nsubj < consist > prep+of+pobj+stanza

S10: The church is build in a mostly Romaneque style and consists of three naves
Lexico-syntactic pattern: church+nsubj < consist > prep+of+pobj+nave
```

Fig. 4.4: Lexico-syntactic patterns from sentences.

As its final output, the Pattern Induction and Formalization stage generates a set of lexico-syntactic patterns, the instance pairs they sub-categorize (connect), and statistics on the pair and pattern co-occurrence frequencies, harvested from our Wikipedia knowledge-base. Sample patterns, sub-categorized instance pairs and their co-occurrence frequencies are shown in the 4th (rightmost), 3rd, 2nd and 1st columns of Figure 4.5. They are respectively derived from sentences *S8*, *S9* and *S10*. N1 and N2 are generic placeholders that represent the actual instances. Each instance pair and pattern corresponds to a relation triple.

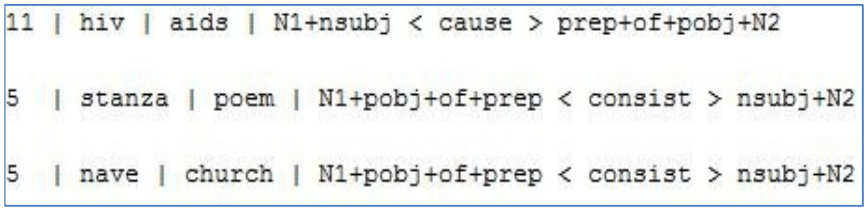


Fig. 4.5: Relations (patterns and pairs) from Wikipedia.

(For better readability, we will use the patterns' surface form, e.g. *"consist of"*, when referring to them in running-text. Lexico-syntactic patterns are shown in our illustrations)

Our lexico-syntactic patterns overcome the shortcomings of the traditional surface-strings (Contribution 5). As can be seen from the above illustrations, they abstract over surface text, and neutralize word order or morphological variations. For example, we derive the single, most general pattern, *"consist of",* to represent the relations between the pairs "stanza-poem" and "nave-church" in sentences *S9* and *S10*. The lexico-syntactic patterns also efficiently deal with long range dependencies. For example, we are able to accurately capture the relation between "nave" , "*consist of*" and "church" in *S10*, despite their not appearing in adjacent positions in surface texts.

At this juncture, two important points deserve further elaboration:

- First, since our Wikipedia knowledge-base is a broad-coverage resource, the triples extracted at this stage encode different semantic relations. For example, the triple "hiv *cause of* aids" with the pattern "*cause of*", derived from sentence *S8*, expresses a causal relation between the instance pair "hiv-aids". On the other hand, the triple "poem *consist of* stanza", with the pattern *"consist of"*, inferred from sentence *S9,* encodes a part-whole relation between the pair "stanza-poem".
- Second, some sentences may yield more than one relation triple, involving distinct instance pairs. An example of such a sentence is *S6* = "a car *consists of a* number *of* doors", presented in section 4.3. Two triples will be extracted from *S6*. They are namely, the triple "car *consist of* door", involving the part-whole instance pair "door-car", and the triple "car *consist of* number", involving the invalid part-whole pair "number-car".

To identify which of the instance pairs (e.g. "door-car", "number-car", "poem-stanza", "hiv-aids") and patterns (e.g. *"consist of", "cause of"*) express part-whole relations, we develop a minimally-supervised approach, based on the Espresso algorithm of Pantel and Pennacchiotti [110]. Compared to Espresso, the novel features in our technique are as follows:

- We extract lexico-syntactic patterns to address the drawbacks of traditional surface-strings, as already shown.
- We incorporate a mechanism to mitigate the negative impact of semantic-drift on the performance of minimally-supervised techniques.
- Instead of attempting to learn relations directly from the (sparse) domain-specific texts, we leverage upon a knowledge-base, namely Wikipedia, and use the information acquired from the knowledge-base to detect the domain-specific relations.

In essence, our technique determines whether an instance pair instantiates a part-whole relation according to the strength of its association with reliable part-whole patterns. For example, in a large corpus like Wikipedia, the pairs "door-car" (*S6*) and "poem-stanza" (*S9*) will be strongly associated (i.e. co-occur) with reliable part-whole patterns, such as *"contain"*, and *"part of"*. They will then be considered as valid part-whole pairs. Conversely, pairs like "number-car" (*S6*) and "hiv-aids" (*S8*) will not be strongly associated with reliable part-whole patterns. Consequently, they will not be selected as valid part-whole pairs. In a similar fashion, to determine whether a pattern expresses a part-whole relation, our technique estimates its association strength with valid part-whole instance pairs. For example, the pattern *"consist of"* (*S9, S10*) is likely to be strongly associated with part-whole pairs like "engine-car" and "grape-wine". It will therefore be considered as a reliable indicator of a part-whole relation. Conversely, the pattern *"cause of"* is unlikely to be

associated with part-whole pairs, and hence, will not be selected as a reliable indicator of a part-whole relation.

We will next describe how we identify which of the harvested patterns (and pairs) express part-whole relations.

4.5.4 SEED SELECTION

Similar to Espresso [110] , our proposed algorithm also takes as input a set of randomly selected seeds. A seed is an instance pair in our Wikipedia knowledge-base that unambiguously instantiates the relation of interest, in this case, part-whole. As mentioned earlier, Wikipedia abounds in prototypical part-whole pairs, such as "engine-car", which can be conveniently used as seeds to initialize our algorithm. Finding such good quality seeds directly from our domain-specific texts is difficult (Challenge 3) due to the domain-specific terminology. Thus, our use of Wikipedia also makes our seed selection procedure much easier (Contribution 3).

These seeds are used in a bootstrapping and recursive mechanism to extract the desired part-whole relations, as will be next discussed.

4.5.5 PATTERN SELECTION

Pattern Selection is the first phase of our minimally-supervised algorithm. It outputs a set of lexico-syntactic patterns that reliably encode part-whole relations.

In its first iteration, our algorithm determines which of the previously induced lexico-syntactic patterns (Section 4.5.3) connect the seeds (Section 4.5.4) in the Wikipedia knowledge-base. These patterns presumably express a part-whole relation since the seeds correspond to part-whole instance pairs. The identified patterns are then input to the subsequent phase of our approach, namely, Instance Selection (to be discussed in Section 4.5.6), where they are used to acquire new instance pairs. In turn, these pairs are fed back to the second iteration of the Pattern Selection phase. This process of learning new patterns from pairs and vice-versa is repeated until a suitable number of patterns are collected.

To ensure the accuracy of the recursive learning procedure, only the most reliable patterns must be selected in each iteration. We define the reliability of a pattern as the degree to which it expresses a part-whole relation. Reliable patterns have a higher likelihood of connecting valid part-whole pairs. For example, the pattern *"consist of"* can be expected to have a high reliability as it always connects valid part-whole pairs. We estimate the reliability of patterns using the measure of Pantel and Pennacchiotti [110], defined in equation (4.1). This measure computes the reliability of a pattern, *p*, in expressing a part-whole relation as its average strength of association with part-whole instance pairs, *i*, weighted by the reliability of these pairs, *r(i)*. The reliability of the initializing seeds is set to 1, i.e. *r(i)* = 1.

$$r(p) = \frac{\sum_{i \in I} \frac{pmi(i,p)}{max_{pmi}} \times r(i)}{|I|} \qquad (4.1)$$

In the above equation, |I| denotes the number of instance pairs. The value of *pmi(i,p)* is the pointwise mutual information [69] between an instance pair, *i=x-y* (e.g. *i*="engine-car"), and a pattern *p* (e.g. *p*="*consist of*"). It is calculated as shown in equation (4.2)[14], where |*x,p,y*| is the frequency (probability) with which the pattern *p* sub-categorizes the pair *x-y* The "*" symbol is a wildcard representing any pattern or any instance pair.

[14] It is similar to the cube mutual information we used for term extraction (Section 2.5.4), except that its numerator is not raised to the power of three (i.e. not cubed)

$$pmi(i,p) = \log \frac{|x,p,y|}{|x,*,y| \times |*,p,*|} \quad (4.2)$$

A well-known issue with the pointwise mutual information (equation (4.2)) is that it tends to overestimate low frequency events. To correct for this overestimate, Pantel and Ravichandran [140] suggested a discounting factor. In our experiments, we observed that the application of the discounting factor did not improve our performance. Therefore, we simply applied a frequency cut-off of 10 to discard all instance pairs and patterns that occurred less than 10 times in Wikipedia, as will be described in Section 4.6.2.

The reliability measure of equation (4.1) assigns higher scores to unambiguous patterns, e.g. "*consist of*", since they tend to co-occur with a large number of valid part-whole pairs, which have high *r(i)* values. The high reliability scores of unambiguous patterns indicate that they accurately express part-whole relations.

Examples of reliable part-whole patterns, identified during Pattern Selection are shown in Figure 4.6.

```
N1+pobj+of+prep < consist > nsubj+N2
N1+dobj < contain > nsubj+N2
N1+dobj < include > nsubj+N2
N1+pobj+of+prep < comprise > nsubjpass+N2
N1+nsubjpass < find > prep+in+pobj+N2
```

Fig. 4.6: Part-Whole lexico-syntactic patterns from Wikipedia.

After computing the patterns' reliability, our minimally-supervised algorithm bootstraps the top-*k* most reliable ones, and inputs them to its Instance Selection phase. The value of *k* is determined experimentally in Section 4.6.4.

4.5.6 INSTANCE SELECTION

In the Instance Selection phase, we use the *k* most reliable patterns previously identified, and extract the instance pairs that they sub-categorize in the Wikipedia knowledge-base. These instances are likely to instantiate part-whole relations. They will be selected and fed back to the Pattern Selection phase for identifying other reliable patterns. Therefore, to extract the most accurate patterns, we must ensure that only the valid and most reliable part-whole instance pairs are selected.

Our instance reliability measure is made up of two components, namely pair-pattern association strength and instance_pair_purity. We next define these two components, and establish the need for the additional instance_pair_purity measure.

Pair-Pattern Association

The pair-pattern association measure is defined by Pantel and Pennacchiotti [110] as shown equation (4.3). Given an instance pair, *i*, it estimates its average strength of association with part-whole patterns, *p*, weighted by the reliability of these patterns, *r(p)*. In equation (4.3), |P| denotes the number of patterns, and the value of *pmi(i,p)* is as defined in equations (4.1) and (4.2).

$$r(i) = \frac{\sum_{p \in P} \frac{pmi(i,p)}{max_{pmi}} \times r(p)}{|P|} \quad (4.3)$$

Instance_Pair_Purity

In the ideal case, if all the patterns, p, are unambiguous indicators of part-whole relations, i.e. $r(p) = 1$, then the pair-pattern association strength (equation (4.3)) is an accurate estimate of the instance reliability. However, part-whole relations are expressed by a wide variety of unambiguous (e.g. "*consist of*", "*contain*") and ambiguous (e.g. "*is in*", "*make in*", "*find in"*) patterns [107]. Ambiguous patterns are problematic since they express different types of relations depending on the instance pairs they sub-categorize. Thus, the instance pairs connected by these patterns do not always participate in a part-whole relation.

These invalid part-whole pairs, which are connected by ambiguous part-whole patterns, are responsible for the effect of semantic-drift [119]. To illustrate the semantic-drift phenomenon and how we overcome it, we consider as example two sentences, *S11* and *S12*:

- *S11* = "lyrics *found in* song"
- *S12* = "building *is in* use"

Both sentences contain an ambiguous part-whole pattern, namely p_{11} = *"find in"* in *S11* and p_{12} ="*is in"* in *S12*. In *S11*, the pattern p_{11} establishes a valid part-whole relation between the instances "lyrics" and "song" in the pair i_{11}="lyrics-song". That is, i_{11}="lyrics-song" corresponds to a valid part-whole pair. Conversely, in *S12*, the pattern p_{12} does not establish a valid part-whole relation between "building" and "use" in the pair i_{12}="building-use". That is, i_{12}="building-use" does not correspond to a valid part-whole pair. Since these types of invalid part-whole pairs tend to co-occur with many other ambiguous part-whole patterns, they will be awarded high instance reliability scores (equation (4.3)), and fed back to the next phase of Pattern Selection (Section 4.5.5). Then, they will cause other invalid patterns to be selected. For example, the invalid pair i_{12}="building-use" may lead to the extraction of the pattern *"convert for"*, as in "building *converted for* use*"*, which does not express a part-whole relation. In turn, these invalid part-whole patterns will be fed back to the Instance Selection phase, and will promote the extraction of other invalid part-whole pairs. For example, the pattern *"convert for"* will lead to the selection of the invalid part-whole pair "ship-use" as in "ship *converted for* use*".*

This phenomenon, whereby the relations extracted by a minimally-supervised technique differ from the target relation instantiated by the initializing seeds is known as semantic-drift. Semantic-drift has been shown to be detrimental to the performance of minimally-supervised algorithms [119].

To mitigate the issue of semantic-drift (Challenge 4), we therefore have to determine whether an instance pair instantiates a valid part-whole relation. Only those valid part-whole pairs are then selected and fed back to promote the selection of reliable part-whole patterns (in Pattern Selection). In our approach, this is achieved by our instance_pair_purity measure (Contribution 4). It differs from the pair-pattern association strength, defined in equation (4.3), in that it does not merely take into account the association of instance pairs with patterns, some of which may be ambiguous. As described above, many of the pairs connected by ambiguous part-whole patterns may be erroneous and will be incorrectly awarded high reliability scores. Instead, our instance_pair_purity measure ensures that these invalid pairs are awarded much lower scores than valid ones. In this way, they will be rejected on account of their lower scores. This, in turn, prevents the phenomenon of semantic-drift from settling in, and from deteriorating the performance of our algorithm. Consequently, only the valid pairs, with the highest scores, will be selected and fed back, which promotes the extraction of other high quality part-whole patterns. We will now illustrate how we calculate the instance_pair_purity.

Consider an unambiguous, archetypal part-whole pattern, e.g. $p_{unambiguous}$ ="*consist of"*, which is found to connect our initializing seeds in the Wikipedia knowledge-base. In such a large corpus as Wikipedia, it is reasonable to expect that pairs like i_{11}="lyrics-song" will also be connected by archetypal part-whole patterns like $p_{unambiguous}$, as in "song *consists of* lyrics". Since these pairs and our seeds are connected by the same patterns, e.g. $p_{unambiguous}$, they instantiate the same relation (as our seeds), namely part-whole. This follows from the Latent Relation Hypothesis (LRH) [141]. Thus, the pairs will then be considered as valid part-whole pairs.

On the other hand, it is highly improbable for pairs like i_{12}="building-use" to be sub-categorized by the same archetypical part-whole patterns, such as $p_{unambiguous}$, which also connect our seeds. For example "building *consists of* use" or "use *consists of* building" are incoherent constructs, and will not occur in a standard corpus like Wikipedia. Since these pairs and our seeds are not connected by the same patterns, they do not instantiate valid part-whole relations. Thus, pairs like i_{12}="building-use" are not considered as valid part-whole pairs.

Generalizing from the above observations, we estimate the instance_pair_purity of an instance pair as its likelihood (probability) of being connected by unambiguous part-whole patterns, which also sub-categorize our initializing seeds. For example, if an instance pair *i* appears 100 times in Wikipedia, and 63 of its occurrences are sub-categorized by an unambiguous pattern, e.g. $p_{unambiguous}$, then, the "purity" of *i* is 0.63; i.e. *instance_pair_purity(i)*= 0.63.

We estimate the instance_pair_purity of a pair, *i*, according to the pseudo-code below. To ensure robustness and accuracy, we employ two unambiguous patterns, namely $p_{unambiguous1}$ = "*consist of*" and $p_{unambiguous2}$ ="*contain*". These two patterns are defined in lines 1 and 2 of the pseudo-code. They were deliberately chosen since they were found to connect our seeds in the Wikipedia knowledge-base. Furthermore, previous studies have also identified these patterns as explicit expressions of part-whole relations [106, 107, 138, 132]. Their preciseness can be attributed to their semantic frames, which always involve a *container* and its *containment* as semantic roles [142].

Then, in lines 3-5 of the pseudo-code, we determine the values *n*, *n1* and *n2*, which respectively indicate the total frequency of *i*, and its co-occurrence frequencies with $p_{unambiguous1}$ and $p_{unambiguous2}$. If *i* does not co-occur with any of these unambiguous patterns, it is unlikely that it instantiates a valid part-whole relation. In this case, we return a purity score of 0, as indicated in lines 6-7. Else, we return as purity score the value *(n1+n2)/n*, which is computed in lines 8-9.

```
Procedure instance_pair_purity(instance-pair i)
1.   p_unambiguous1 = "consist of"
2.   p_unambiguous2 ="contain"
3.   n = total frequency i
4.   n1 = co-occurrence frequency of i and p_unambiguous1
5.   n2 = co-occurrence frequency of i and  p_unambiguous2
6.   if (n1 and n2) == 0
7.        return 0
8.   else
9.      return (n1+n2)/n
10   end if
```

Instance Reliability

Finally, to compute the (overall) reliability of an instance pair, *i*, we combine the pair-pattern association and instance_pair_purity components, as shown in equation (4.4).

$$r(i) = \frac{\sum_{p \in P} \frac{pmi(i,p)}{max_{pmi}} \times r(p)}{|P|} + instance\_pair\_purity(i) \qquad (4.4)$$

In this way, invalid part-whole pairs introduced by ambiguous patterns will be awarded lower purity scores by the instance_pair_purity component. As a result, these pairs will have smaller reliability scores, hindering their selection and input to the Pattern Selection phase. They are thus prevented from extracting other incorrect patterns (and pairs) in subsequent iterations, and are inhibited from triggering the semantic-drift phenomenon. On the other hand, valid part-whole pairs will have higher purity values. They are assigned larger reliability scores, and our algorithm selects only the top-*m* most reliable pairs to be fed back to the Pattern Selection phase. These valid pairs promote the extraction of other

accurate part-whole patterns in later iterations. Consequently, the performance of our algorithm improves. Examples of part-whole instance pairs identified by the Instance Selection phase are depicted in Figure 4.7.

```
chapter (part) | book (whole)
actor | cast
aircraft | fleet
track | album
musician | band
grape | wine
```

Fig. 4.7: Part-Whole pairs from Wikipedia.

We then iterate between the Pattern Selection (Section 4.5.5) and Instance Selection phases until a suitable number, *t*, of patterns have been extracted. The value of the parameters *m* and *t* are defined experimentally in Section 4.6.4.

4.5.7 Extracting Domain-Specific Part-Whole Relations

We now use the *t* reliable part-whole patterns, detected from the Wikipedia knowledge-base, to extract part-whole relations from our domain-specific target corpus. It should be noted that the latter corpus has already been cleaned and pre-processed, and a ranked-list of terms has been extracted from its contents by our ExtTerm framework, which we presented in Chapter 2.

To identify domain-specific part-whole relations, we use our patterns (from Wikipedia), and extract the instance pairs that they connect in the target corpus. Such a strategy enables us to overcome the issue of data sparsity that would have been encountered if the part-whole relations (patterns, pairs) were extracted directly from the domain-specific corpus (Challenge 2). Our Wikipedia knowledge-base, being a large collection of well-written texts, provides us with high quality linguistic (e.g. syntactic dependencies) and statistical (e.g. significant pair-pattern counts) evidence, which we exploit to discover reliable patterns from its contents. By leveraging upon these patterns, we are then able to accurately identify part-whole relations from our domain-specific texts (Contribution 2).

Examples of part-whole relation triples extracted from our domain-specific corpus are shown in Figure 4.8. Each triple is made up of a pattern, and a part-whole instance pair.

```
<pattern=locate in, part=adaptor board, whole=processing unit>
<pattern=contain, part=current monitoring circuit, whole=video>
<pattern=find in, part=leaking tube, whole=radiator>
<pattern=part of, part=rotor whole=c-arm>
<pattern=in, part=fuse, whole=cabinet>
```

Fig. 4.8: Domain-specific Part-Whole relation triples from target corpus.

In our experiments, we will empirically demonstrate the gains in performance realizable when leveraging upon the general-purpose Wikipedia corpus as a knowledge-base to support the extraction of domain-specific part-whole relations.

4.6 EXPERIMENTAL EVALUATION

In this section, we evaluate the performance of our proposed framework for extracting part-whole relations from domain-specific corpora by leveraging on Wikipedia as a knowledge-base. In Section 4.6.1, we provide some basic statistics on our Wikipedia and domain-specific corpora. These statistics have been presented earlier in Chapter 2. They are included here again for completeness.

In Sections 4.6.2-4.6.5 we discuss the results of the various phases of our approach for part-whole relation extraction. Performance scores are evaluated in Section 4.6.6. We compare our proposed approach with a baseline in Section 4.6.7 to demonstrate the performance gains by leveraging upon Wikipedia as a knowledge-base. Finally, in Section 4.6.8, we show that our instance_pair_purity measure plays an important role in alleviating the issue of semantic drift, and contributes to our performance.

4.6.1 CORPORA

Knowledge-Base
In our framework, we employed the Wikipedia collection of English texts (August 2007 dump, ~ 50 GB uncompressed) [91] as knowledge-base. The motivations underlying this choice were discussed in Section 4.5.2, where we (qualitatively) compared Wikipedia and BNC. We indicated that Wikipedia exhibited a number of desirable characteristics, such as its broader coverage and larger size, which made it more attractive as a knowledge-base. These two characteristics will be (empirically) investigated in details in Section 4.6.3

We used our Wikipedia knowledge-base to acquire a set of lexico-syntactic patterns that reliably express part-whole relations. The patterns were employed to extract domain-specific part-whole relations from a domain-specific (target) corpus, described below.

Target Corpus
The corpus that we targeted for the extraction of domain-specific part-whole relations is a collection of 32,545 documents generated in the business/corporate domain of Product Development-Customer Service (PD-CS). The documents describe customer complaints and the ensuing repair actions performed by service engineers on high-end, electronic equipment. The target corpus was compiled over a period of five years, spanning from 2005 to 2009, and the documents were collected from customer-call centers and engineers' repair notes. The texts were expressed in English.

Statistics on our two corpora are presented in Table 4.1.

	Number of words	**Number of documents**
Domain-specific corpus	1,952,739	32,545
Wikipedia	500 million (approx.)	5,100,000 (approx.)

Table 4.1: Statistics on Wikipedia and domain-specific corpora.

4.6.2 PATTERN INDUCTION

We linguistically processed the sentences of the Wikipedia knowledge-base as described in Section 4.5.3. An important aspect that must not be overlooked when processing large corpora is the prohibitive computational costs involved. Activities like syntactic parsing (to derive the lexico-syntactic patterns) are traditionally considered as computationally expensive. Pantel et al. [143], for example, reported that the full parsing of a 15 GB corpus can consume anywhere between 56 days to 10.2 years on a Pentium 4 – 2.5 GHz machine. Considering the size of our Wikipedia knowledge-base (~50 GB), parsing its contents would require approximately more than 56 days. This is clearly inefficient. To overcome the difficulty in linguistically processing our relatively large Wikipedia

knowledge-base, we took advantage of parallel computing techniques. Specifically, we split the linguistic processing tasks into a number of distinct jobs, and executed these jobs concurrently on a high performance computing cluster[15]. The entire operation completed[16] in about two days.

An alternative solution could have been to use a subset, say 10%, of our current Wikipedia corpus as knowledge-base. However, a smaller knowledge-base may fail to capture the wide variety of patterns that express the relations between parts and wholes [132]. As a result, the recall of our approach will be affected. In addition, such a smaller corpus may not provide sufficient redundancy and significant statistical evidence to overcome the data sparsity issue posed by our domain-specific texts. Consequently, our precision in extracting part-whole relations may deteriorate.

After linguistically processing the sentences of Wikipedia, we extracted as triples the instance pairs and the shortest lexico-syntactic paths (patterns) that connected them in their dependency trees. Statistics on the co-occurrence frequencies of pairs and patterns were also estimated. In our experiments, we applied a frequency cut-off of 10, and discarded all pairs and patterns that appeared less than 10 times in Wikipedia, as described in Section 4.5.5.

In total, we extracted 2,176,992 distinct lexico-syntactic patterns and 6,798,235 distinct instance pairs. Detailed statistics on the number of instance pairs and patterns after applying the frequency cutoff of 10 are listed in Table 4.2.

Instance pairs	328.0 million (approx.)
Distinct instance pairs	6,798,235
Patterns	238.0 million (approx.)
Distinct patterns	2,176,992

Table 4.2: Statistics on the number of pairs and patterns from Wikipedia.

An example of a lexico-syntactic pattern that we derived from a sentence is shown in Figure 4.9. In this illustration, *S13(a)* is a sentence from our Wikipedia knowledge-base, which describes various relations between the instances "song", "Alabama 3", "samples" and "speech". In *S13(b)*, the 4th (rightmost) column shows the lexico-syntactic pattern extracted from the sentence's dependency tree, the 2nd and 3rd columns depict the pair of related instances, and the 1st column gives the pair-pattern co-occurrence frequency in Wikipedia.

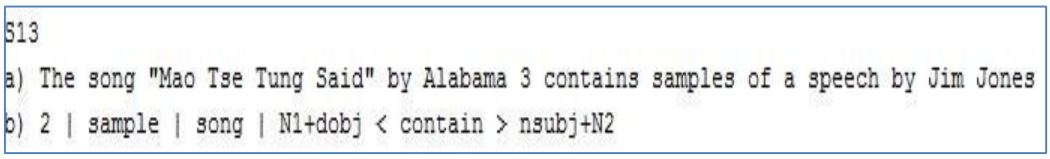


Fig. 4.9: Pair-pattern induced from Wikipedia.

It can be seen from Figure 4.9 that our lexico-syntactic patterns overcome the limitations of traditional surface-string representations (Challenge 5). For example, we were able to capture long-range dependencies between related instances, such as between "song" and "sample", which did not appear in adjacent positions in the surface text. In addition, as shown before, the lexico-syntactic patterns neutralized word order and morphological variations, enabling us to derive the most general patterns between the related instance pairs (Contribution 5).

[15] The High Performance Computing Cluster of the University of Groningen: http://www.rug.nl/cit/hpcv/faciliteiten/HPCCluster
[16] Our parsed Wikipedia version can be downloaded from http://www.let.rug.nl/gosse/Wikipedia/enwiki.html

To determine which of the patterns induced from Wikipedia expressed part-whole relations, we employed our minimally-supervised approach. It was initialized with a set of seeds, as will be next described.

4.6.3 SEED SELECTION

We handpicked 20 part-whole instance pairs from our Wikipedia knowledge-base to be used as seeds for initializing our minimally-supervised approach. Similar to previous studies, we did not enforce any selection criterion when choosing these initializing seeds. That is, they were randomly chosen. The only requirement that we imposed was that they should occur at least 50 times in our Wikipedia knowledge-base. This was done to ensure that we indeed selected the most prototypical and widely used part-whole pairs as seeds.

Some examples of seeds used in our experiments, and their occurrence frequency in our Wikipedia knowledge-base are presented respectively in the 1st and 2nd columns of Table 4.3. Also shown in the 3rd and 4th columns are the seeds' occurrence frequency in the British National Corpus (BNC), and their relative frequency (i.e. frequency ratio) across Wikipedia and the BNC. This enables us to highlight the statistical differences between Wikipedia and the BNC. Our aim is to empirically demonstrate that Wikipedia is a more promising alternative as a knowledge-base for overcoming the data sparsity issue of our domain-specific texts.

Seeds (part-whole pairs)	**Seeds Wikipedia Frequency**	**Seeds BNC Frequency**	**Frequency Ratio Wikipedia:BNC**
sugar-ingredient	166	1	166.0
guitarist-band	3660	69	53.04
village-area	7990	202	39.55
engine-car	3509	361	9.72
grave-church	155	29	5.34

Table 4.3: Sample seeds to initialize our minimally-supervised algorithm.

Given the size of our Wikipedia corpus of approximately of 500 million words, and the size of the BNC of approximately 100 million words, we would have expected a part-whole pair to be at most 5 times likelier in Wikipedia than in the BNC. However, as can be seen from the relative frequency values presented in Table 4.3, part-whole pairs are much more likely to be found in Wikipedia. For example, "engine-car" is around 10 times more frequent in Wikipedia, with a frequency of 3509, than in BNC, with a frequency of 361. These results indicate that the part-whole pairs, which we require as initializing seeds, are well represented in Wikipedia. Compared to the BNC, Wikipedia offers more significant statistical evidence, which enables us to effectively overcome the sparse characteristics of our domain-specific texts, and to improve our accuracy in extracting part-whole relations. These observations suggest that Wikipedia is more suitable as a knowledge-base to support our task of minimally-supervised domain-specific part-whole relation extraction.

4.6.4 PATTERN AND INSTANCE SELECTION

The selected seeds were used to initialize our minimally-supervised algorithm. In each of its iteration, our algorithm bootstrapped the top-k patterns that most reliably expressed a part-whole relation, and used them to learn new part-whole pairs. The top-m most reliable pairs were in turn bootstrapped to acquire new part-whole patterns. The recursive procedure of learning new patterns from pairs and vice-versa was repeated until t patterns had been harvested, as described in Sections 4.5.5 and 4.5.6.

In its first iteration, our algorithm extracted the 10 most reliable patterns that were found to connect the initializing seeds in Wikipedia. These 10 patterns were bootstrapped to induce 100 part-whole pairs. In each subsequent iteration, we then learnt 7 additional patterns, i.e. $k=|P| + 7$, and 20 additional instance pairs, i.e. $m=|I| + 20$, where $|P|$ and

$|I|$ are respectively the number of patterns and pairs harvested in a previous iteration. We halted the algorithm after the 12th iteration since the quality of the patterns collected in subsequent iterations (i.e. the performance) was almost constant. At this point, $t = 72$ distinct reliable part-whole patterns[17] had been harvested. We also observed that using larger increments for k and m, i.e. increasing k by more than 7 patterns and m by more than 20 instances per iteration, resulted in the extraction of fewer reliable patterns. Using smaller increments did not significantly affect the pattern quality, but more iterations were required to harvest the largest set of reliable patterns.

Table 4.4 shows the top-10 most reliable part-whole patterns that we extracted from the Wikipedia knowledge-base. Also depicted are the patterns' (linguistic) interpretations. We use "Part" and "Whole" to refer to part and whole instances, e.g. "engine" and "car", or "song" and "album".

Lexico-syntactic Pattern	Interpretation
N1+pobj+of+prep < consist > nsubj+N2	Whole *consists of* Part
N1+dobj < contain > nsubj+N2	Whole *contains* Part
N1+dobj < include > nsubj+N2	Whole *includes* Part
N1+pobj+of+prep < comprise > nsubjpass+N2	Whole comprised of Part
N1+nsubjpass < find > prep+in+pobj+N2	Part *found in* Whole
N1+nsubj < part > prep+of+pobj+N2	Part *part of* Whole
N1+nsubjpass < find > prep+on+pobj+N2	Part *found on* Whole
N1+nsubj < appear > prep+on+pobj+N2	Part *appears on* Whole
N1+nsubj < available > prep+on+pobj+N2	Part available on Whole
N1+pobj+from+prep < make > nsubjpass+N2	Whole *made from* Part

Table 4.4: Top-10 most reliable patterns harvested from the Wikipedia knowledge-base.

An interesting phenomenon that we observed during our experiments was that the initializing seeds had an unpredictable influence on the part-whole relations (patterns and instance pairs) that were extracted by our minimally-supervised approach. Also, some seeds appeared to be more fertile than others, leading to the discovery of more accurate part-whole patterns. At this point, however, we were only able to speculate about a possible correlation between the initializing seeds and the performance of our minimally-supervised approach. We did not perform any additional experiments to confirm or deny the existence of such a correlation. We will investigate this phenomenon in details in Chapter 6.

4.6.5 Extracting Domain-Specific Part-Whole Relations

To identify domain-specific part-whole relations, we used the patterns harvested from the Wikipedia knowledge-base, and extracted the instance pairs they were found to connect in our target (domain-specific) corpus. The latter corpus had already been cleaned and pre-processed, as discussed in Chapter 2, and a ranked-list of terms had been extracted from its contents by our ExtTerm term extraction framework.

Out of the 72 part-whole patterns from Wikipedia, 45 were found to connect domain-specific instances pairs in the target corpus. These pair-pattern combinations yielded a total of 13,592 domain-specific part-whole relation triples. Examples are shown in Table 4.5.

In this table, the 3rd column depicts the patterns that were found to connect domain-specific instances in the target corpus. The 2nd column indicates the percentage (fraction) of the 13,592 relation triples that contained these patterns. Part-whole relation triples, i.e. combinations of instance pairs and patterns, are shown in the 4th column, while the 5th column provides text snippets in which the relations were realized in the target corpus. The 1st column is an identifier for each pattern/relation group.

[17] A preliminary evaluation estimated the precision to be roughly around 0.80.

Grp #	Freq (%)	Pattern	Triple	Example
1	35	available on	<available on, brightness option , menu console>	Brightness control (not) *available on* menu console
		show in	<show in, background lines, picture>	Background lines *shown in* picture
		appear on	<appear on, grey pixels, image>	Grey pixels *appearing on* image
2	28	include	<include, calibration, corrective action >	Corrective action *includes* calibration
		perform in	<perform in, reboot, configuring>	Reboot *performed in* configuring
3	23	locate in	<locate in, adaptor board, pc>	Adaptor board l*ocated in* PC
		find in	<find in, blown fuse, settop box>	Blown fuse *found in* set-top box
			<find in, problem, past>*	Problem *found in* the past*
4	11	come from	<come from, bulb, x-ray tube>	Bulb *came from* x-ray tube
		reach	<reach, c-arm, table base>	C-arm unable to *reach* table base
		make with	<make with, iodine, contrast fluid>	Contrast fluid *made with* iodine
		make from	<make from, monitor, factory>*	Monitor *made from* factory*
5	3	release in	<release in, software upgrade, processor>	Software upgrade *released in* processor
		incorporate in	<incorporate in, security feature, software >	Security feature *incorporated in* software

Table 4.5: Domain-Specific Part-Whole relations extracted from target corpus.

*: denotes incorrectly detected part-whole relations.

Discussion of Results

The most frequent patterns, appearing in around 35% of all the extracted triples, were those that expressed part-whole relations between visual and intangible instances. Examples were "*appear in*" as in "grey pixels *appearing on* image", and "*available on*" as in "brightness control (not) *available on* menu console". Their high frequency can be attributed to the contents of the target corpus, which pertained to video/imaging equipment. The relations in which these patterns participated encode useful, but subtly expressed, customer complaints. They can be exploited by business/corporate organizations to investigate causes of customer dissatisfaction.

The next most frequent patterns, occurring in 28% of the triples, were those that related action sequences. Examples were "*include in*" as in "corrective action *includes* calibration" and "*perform in*" as in "reboot *performed in* configuring". The patterns were relatively frequent since our target corpus contained narrations of repair actions performed by engineers. The relations in which these patterns participated are useful for management to determine whether engineers are following the proper diagnosis and repair procedures when servicing faulty products.

Patterns that expressed the relations between "parts" and their "containers" were also relatively frequent, occurring in around 23% of all the extracted triples. Examples were "*locate in/on/at*" as in "adaptor board l*ocated in* PC", and "*find in/on/at*" as in "blown fuse *found in* set-top box". The relations containing these patterns facilitate the diagnosis procedures of service engineers, and enable them to efficiently detect causes of product failures. However, some patterns like "*find in*" were ambiguous, and led to the extraction of invalid part-whole relations in the target (domain-specific) corpus, such as "problem *found in* (the) past".

Rarer patterns were those that appeared in 11% of the extracted triples. Examples were "*come from"* as in "bulb *came from* x-ray tube", and "*make with*" as in "contrast fluid *made with* iodine". The relations instantiated by the patterns are also useful for investigating customer complaints, and detecting causes of product failures. However, some of the patterns, e.g. "*make from*", were ambiguous and extracted invalid part-whole relations in the target corpus, such as "monitor *made from* factory".

The least frequent patterns, occurring in around 3% of the relations extracted, were those representing the merging or union of parts and wholes. Examples are "*release in*" as in "software *released in* processor", and "*incorporate in*" as "security feature *incorporated in* software". The relations instantiated by these patterns provide relevant information on product upgrades or patches released in products, and are useful in versioning.

The remaining 27 (out of the 72) patterns harvested from the Wikipedia knowledge-base were not found in the target corpus since they were unlikely to be used in texts describing customer complaints and repair actions. Examples included "*member of*", "*collection of*" and "*published in*".

4.6.6 PERFORMANCE MEASURES

We evaluated the performance of our approach in extracting part-whole relations from the domain-specific target corpus by measuring its precision, recall and F1-scores.

Precision

Our evaluation methodology is based on that of Evert and Krenn [94]. They showed that the evaluation results obtained over a random sample of size *n* were comparable to those obtained over the entire result set. In our experiments, we set *n*=3000, i.e. we evaluated 3000 (distinct) triples from the total of 13,592 extracted from the target corpus. The triples were equally selected from each of the top-4 groups of Table 4.5. That is, from each group, 750 relations were chosen. We deliberately omitted relations that contained patterns in the last (5^{th}) group, e.g. "*incorporate in*". These patterns were extremely rare and very precise. They could positively bias our evaluation results.

Two human judges (annotators), who were well-versed in the domain, were independently asked to tag the 3000 triples as correct if they denoted valid part-whole relations or incorrect if they did not. The judges were allowed to inspect the documents to facilitate their decisions. Triples deemed correct (i.e. valid part-whole relations) by both judges were *true_positive*, while those considered incorrect by both of them were *false_positive*. Ambiguous cases, on which the judges disagreed (e.g. a triple marked correct by only one judge), were discarded. The number of *true_positive*, *false_positive* and ambiguous (undecided) cases were respectively 2332, 605 and 63. To mitigate the negative impact of coincidental agreements between the two judges on our evaluation results, we computed the inter-annotator agreement using the kappa coefficient [95]. Our kappa values were in the range of 0.69-0.74, which indicated a relatively high level of inter-annotator agreement since values of 0.7 are considered desirable.

The precision, P, of our approach was then estimated using equation (4.5) as 0.79.

$$Precision = \frac{true\_positive}{true\_positive + false\_positive} \qquad (4.5)$$

Recall

Estimating the recall is more difficult since it requires prior knowledge on the actual number of valid relation triples in the target corpus. To compute the recall, we manually inspected a subset of the target corpus, consisting of randomly selected documents, and identified 500 valid part-whole relations contained therein. These 500 known relations served as our gold-standard.

From the sub-corpus, our technique extracted 398 triples that were also found in the gold-standard. They were counted as *true_positive*. The remaining 102 (out of the 500) gold-standard triples that we failed to detect were counted as *false_negative*. Using equation (4.6), we estimated the recall, R, of our approach as 0.80.

$$Recall = \frac{true\_positive}{true\_positive + false\_negative} \tag{4.6}$$

An inspection of the results revealed that the 102 undetected relations were realized by unconventional part-whole patterns in the target corpus. These patterns had been rejected from our Wikipedia knowledge-base since they were not found to reliably express part-whole relations. Examples of such patterns included:

- Those that indicated the absence of a part from its whole, such as the pattern *"missing from"* in "battery (part) is *missing from* cabinet (whole)".
- Those that connected a part to its whole in a temporal or motional relation, such as the pattern *"while"* in "sensor (part) is activated *while* the backplane (whole) rotates".

F1 Measure

To obtain a single performance value, we determined the F1 score [96], which gives the weighted harmonic mean between precision (P) and recall (R), and penalizes high divergence between them. F1 was estimated as 0.79 using equation (4.7).

$$F1 = \frac{2 \times P \times R}{P + R} \tag{4.7}$$

The precision, recall and F1 scores, reported above, indicate that our minimally-supervised approach accurately extracted high quality part-whole relations from the domain-specific texts. It relied solely on a handful of randomly selected seeds, and alleviated the need for prior training on large amounts of annotated data (Contribution 1).

While our performance scores are promising, they do not reveal the contributions of the various novel aspects of our algorithm. In particular, we are interested in precisely (i.e. empirically) determining whether

- Our use of Wikipedia enabled us to effectively address the challenges posed by domain-specific texts (Contributions 2 and 3).
- Our instance_pair_purity measure effectively mitigated the effect of semantic-drift (Contribution 4).

These 2 issues will be investigated in our next experiments.

4.6.7 COMPARISON AGAINST ESPRESSO

To address these 2 aforementioned issues, pertaining to the role of Wikipedia in overcoming the challenges posed by our domain-specific corpus, and to the contribution of our instance_pair_purity measure for dealing with semantic-drift, we will compare the performance of our algorithm against that of a baseline.

As baseline, we will rely on the Espresso algorithm of Pantel and Pennacchiotti [110]. We considered Espresso to be a suitable baseline since it is a state-of-the-art minimally-supervised algorithm, and has been employed in many relation extraction endeavors [144]. Furthermore, in their study, Pantel and Pennacchiotti [110] showed that Espresso outperformed other existing minimally-supervised techniques for the extraction of part-whole relations from texts. In addition, they showed that Espresso could be applied for mining other types of semantic relations, such as hypernymy (between concepts and their sub-concepts, e.g. "vehicle-car"), and "successor-of" (e.g. between state presidents).

Also, Espresso presented several characteristics , which made it a suitable candidate against which to evaluate our proposed approach with regards to the 2 aforementioned issues. For example, compared to our algorithm, Espresso

- Does not supplement its analyses with additional knowledge acquired from other sources (e.g. Wikipedia) for boosting its performance. It simply attempts to learn the (part-whole) relations directly from the target corpus, which in our case, was a collection of domain-specific texts. As a consequence, it is also initialized with seeds taken directly from the target corpus.
- It does not incorporate any mechanism to mitigate the effect of semantic-drift.

We re-implemented the Espresso algorithm by modifying our proposed approach such that it bypassed the stage of learning reliable part-whole patterns from the Wikipedia knowledge-base. That is, it did not rely on any external sources of knowledge. We also did not include the instance_pair_purity component in the instance reliability measure. That is, the reliability of an instance pair was estimated using equation (4.3). In addition, we used simple surface-strings, and not our lexico-syntactic patterns, for representing the relations between instance pairs. Espresso was initialized with seeds taken directly from the target corpus. As seeds, we used the 20 most frequent part-whole pairs that were found in the relations extracted by our algorithm from the target corpus (Section 4.6.5). Espresso was then applied over the target (domain-specific) corpus to discover part-whole relations. In our experiments, we observed that the quality of the part-whole patterns harvested by Espresso from the target corpus started to deteriorate after only 5 iterations.

To investigate the cause of this rapid drop in performance, we searched for occurrences of the (domain-specific) seeds, used to initialize Espresso, in the target corpus. We noticed that the seeds participated in a host of other semantic relations, besides part-whole. For example, the seed "pixel-image" frequently instantiated both a part-whole relation, as in "pixels *found in* image", and a causation relation as in "(dead) pixels *caused* (blurred) image". This highlights the difficulties in selecting fertile seeds from domain-specific texts in order to initialize minimally-supervised techniques (Challenge 3). As expected, the incorrect patterns that connected these domain-specific seeds, e.g. *"cause"* and *"disable in"*, led to the extraction of other invalid part-whole pairs, giving rise to the semantic-drift phenomenon (Challenge 4).

In addition, we noticed that reliable patterns, which unequivocally express part-whole relations, such as *"consist of"* and *"contain"*, were awarded extremely low scores by Espresso. Further investigations revealed that this happened because they co-occurred sparsely (Challenge 2) with typical domain-specific part-whole pairs, such as the pairs "lead-radiation shield" and "iodine-contrast fluid", in the target corpus Also, the surface-strings used by Espresso to represent patterns led to the generation of a large number of redundant patterns, which were susceptible to word order and morphological variations (Challenge 5). These patterns further deteriorated the data sparsity issue. For example, *"consist of"* and *"consist of more than"* were treated as two different patterns. The individual reliability scores of these distinct patterns were relatively low, and they were not selected to harvest other part-whole pairs in the later iterations. More accurate results would have been obtained by representing both strings with their single, most general lexico-syntactic pattern, for e.g. *"consist of"*. This single pattern would then be awarded a

higher (combined) reliability score, making it more likely to be selected in subsequent iterations.

Due to the aforementioned difficulties, the performance of Espresso plummeted severely, as indicated by its precision, recall and F1 scores, which are listed in the first row of Table 4.6. The corresponding scores of our proposed framework are shown in the second row for ease of comparison.

	Precision	**Recall**	**F1**
Espresso	0.44	0.45	0.44
Proposed Framework	0.79	0.80	0.79

Table 4.6: Performance Comparison – Proposed Approach vs. Espresso.

From Table 4.6, it can be seen that our proposed approach outperformed the state-of-the-art Espresso algorithm in extracting high quality part-whole relations from domain-specific texts. The performance scores suggest that:

- Our strategy of leveraging upon a large collection of well-written texts, such as, Wikipedia, effectively enables us to overcome the issues of data sparsity and seed selection from domain-specific texts (Contributions 2,3). This also shows that Wikipedia, despite its broad-coverage contents, can still be exploited for supporting domain-specific tasks, like part-whole relation extraction.
- Our instance_pair_purity measure mitigates the negative impact of semantic-drift on the performance (Contribution 4). This will be discussed in details in Section 4.6.8.
- The lexico-syntactic patterns, employed by our algorithm to represent relations between instance pairs, address the shortcomings of traditional surface-strings (Contribution 5).

4.6.8 REDUCING SEMANTIC-DRIFT WITH INSTANCE_PAIR_PURITY

In this experiment, we will illustrate the significance of our instance_pair_purity measure (Section 4.5.6) and its contribution in mitigating the negative impact of semantic-drift on performance.

We re-implemented our minimally-supervised approach by removing the instance_pair_purity component from the instance reliability measure. That is, the reliability score of an instance pair was estimated using equation (4.3). This new implementation is referred to as Implementation_A. It was initialized with seeds, as mentioned in Section 4.6.3, and applied to the Wikipedia knowledge-base for harvesting part-whole patterns. We analyzed the results, and made the following three observations.

1. We observed that many invalid part-whole pairs, e.g. "building-use", were assigned higher instance reliability scores than valid ones, e.g. "grape-wine". Further investigation revealed that this occurred because the invalid pairs were associated with a larger number of ambiguous part-whole patterns. For example, the ambiguous part-whole patterns "*is in*", "*of*" and "*have",* were all found to connect the invalid part-whole pair "building-use", as in "building *is in* use", "use *of* building" and "building *has* use". As a result, these invalid part-whole pairs were awarded high instance reliability scores.
2. The invalid pairs were then bootstrapped since they had higher reliability scores. They led to the extraction of other incorrect patterns that did not encode part-whole relations. For example, the pair "building-use" caused the extraction of patterns like "*remain in*" ("building *remained in* use") and "*convert for*" ("building *converted for* use"). Then, other invalid part-whole pairs that were connected by these incorrect patterns were in turn harvested, for e.g. the pair "car-production", which was connected by the pattern "*remain in"* ("car *remained in* production"). Erroneous part-whole pairs and patterns started to dominate the iterations of Implementation_A, giving rise to the issue of semantic-drift.
3. We observed that due to semantic-drift, the quality of the extracted patterns (and pairs) deteriorated in earlier iterations. The best set of most reliable patterns extracted by Implementation_A was obtained in the 8^{th} iteration, which does not compare favorably with the 12^{th} iteration of our original approach. As expected,

fewer and less reliable part-whole patterns were thus harvested from the Wikipedia knowledge-base.

We then used the best set of patterns harvested by Implementation_A (in its 8th iteration) to extract domain-specific part-whole relation triples from our target corpus. The performance measures estimated by the precision, recall and F1 scores dropped respectively to 0.60, 0.62 and 0.61, compared to the 0.79, 0.80 and 0.79 achieved by our original approach in Section 4.6.6. These scores are shown in Table 4.7. For ease of comparison, we also depict the performance of the Espresso baseline, given in Section 4.6.7.

	Precision	**Recall**	**F1**
Espresso	0.44	0.45	0.44
Implementation_A	0.60	0.62	0.61
Proposed Framework	0.79	0.80	0.79

Table 4.7: Performance Comparison – Proposed Approach vs. Implementation_A vs. Espresso.

From Table 4.7, it can be seen that the performance of Implementation_A is lower than that of our proposed framework (with the instance_pair_purity measure), albeit higher than that of the Espresso baseline. These results indicate that our technique successfully overcomes the negative influence of semantic-drift on the performance (Contribution 4). From the above table, it can also be seen that our instance_pair_purity measure contributes to a 29.5% gain in performance (cf. F1 scores of Proposed Framework and Implementation_A). Our strategy of learning patterns from the Wikipedia knowledge base accounts for a 38.6% performance gain (cf. F1 scores of Implementation_A and Espresso)

4.7 CONCLUSION

In this chapter, we were concerned with our second research question, *RQ2: How to accurately detect occurrences of semantic relations from domain-specific, sparse and informally-written corporate texts?*

We focused on part-whole relations since they encode valuable information, which can be exploited to support various types of corporate activities, such as Business Intelligence. We addressed the second research question by developing an approach that alleviated the challenges in learning part-whole relations from domain-specific texts, and overcame the shortcomings of extant techniques.

Since domain-specific knowledge resources, like annotated data for training, are not readily available in specialized domains, we employed a minimally-supervised approach. It only required a handful of initial part-whole pairs, called seeds, for extracting the desired relations. To overcome the issue of data sparsity, which is detrimental to the performance of minimally-supervised algorithms, we relied on a knowledge-base as a source of additional information. As knowledge-base, we employed Wikipedia since it exhibited several desiderata that made it more attractive as a knowledge-base than other similar resources, such as the BNC. Our minimally-supervised algorithm was then applied on the Wikipedia knowledge-base for acquiring patterns that reliably express part-whole relations.

A major difficulty in learning part-whole relations is that they are realized by many ambiguous patterns. These patterns do not always express part-whole relations, and may also connect invalid part-whole pairs. These ambiguous patterns and invalid pairs are responsible for the issue of semantic-drift, whereby the relations ultimately extracted by a minimally-supervised algorithm are different from the target relation, as instantiated by the initializing seeds. To mitigate the negative influence of semantic-drift on the performance, we defined an instance_pair_purity measure. It assigned much lower scores to invalid part-whole pairs that were connected by ambiguous part-whole patterns. These invalid pairs were then rejected, and prevented from triggering the semantic-drift phenomenon. Conversely, valid part-whole pairs were awarded much higher scores. They

were then selected and used to harvest other reliable part-whole patterns, which was beneficial to our performance.

After harvesting the reliable part-whole patterns from the Wikipedia knowledge-base, we used them to extract part-whole relation triples from the domain-specific texts. Each triple consisted of a part-whole pattern and a pair of (domain-specific) instances (terms) in the domain-specific texts.

We evaluated our approach on a collection of real-life, domain-specific texts provided by our industrial partners. We also compared it against the state-of-the-art minimally-supervised Espresso algorithm. From our experimental results, we saw that our approach achieved higher performance than Espresso, and extracted more accurate part-whole relations. For example, the performance of Espresso was affected by the sparse characteristics of the domain-specific texts, and it failed to detect reliable part-whole patterns. On the other hand, our approach leveraged upon the large Wikipedia corpus as a knowledge-base, which facilitated the discovery of reliable part-whole patterns. Also, the surface-strings employed by Espresso for representing patterns led to extraction of many redundant patterns due to word order or morphological variations. This further compounded the issue of data sparsity. Conversely, we represented the patterns using lexico-syntactic dependencies, which abstracted over surface-texts. Such a representation was more accurate as it did not suffer from word order or morphological variations. In addition, our approach successfully addressed the issue of semantic-drift, which ensured the extraction of more accurate part-whole patterns and relations.

The main findings/conclusions of this chapter can be summarized as follows:

1. Minimally-supervised algorithms seem to be more appropriate for use in specialized domains, where knowledge resources may not be always available. As illustrated in our experiments, our approach achieved high accuracy in extracting domain-specific part-whole relations, even though it did not rely on domain-specific resources such as annotated data.
2. Using syntactic dependencies to represent the patterns provides several benefits compared to the more conventional surface-strings employed by minimally-supervised algorithms. For example, the use of syntactic dependencies prevented the extraction of a large number of redundant (similar) patterns. They also enabled us to capture long range associations between related instance pairs and patterns even if these patterns/pairs did not occur in adjacent positions in surface texts.
3. Wikipedia can be used as a knowledge-base to overcome the issue of data sparsity posed by domain-specific corpora.
4. Despite its broad-coverage nature, Wikipedia can still be exploited to support domain-specific tasks, such as the extraction of (domain-specific) part-whole relations.
5. Overcoming the issue of semantic-drift plays an important role in the discovery of accurate relations, and is thus beneficial to the performance.

Further discussions pertaining to our approach for mining part-whole relations with respect to RQ2 will be given in Chapter 8.

Concerning the business applications, our part-whole relation extraction technique can be employed as a decision support application to assist engineers in efficiently locating and diagnosing product failures. This results in higher service quality and more satisfied customers. It can also be used to detect pertinent, but subtly expressed customer complaints, as shown in Table 4.5.

CHAPTER 5 –CAUSAL RELATION EXTRACTION

CHAPTER SUMMARY

This chapter also addresses our second research question. It deals with the automatic extraction of causal relations from domain-specific texts. This will be achieved using a slightly modified version of the approach that we presented in Chapter 4 for mining part-whole relations. As before, we will rely on Wikipedia as a knowledge-base to overcome the data sparsity issue posed by our domain-specific texts. We will then acquire a set of patterns that reliably express causality from Wikipedia using our minimally-supervised approach. To disambiguate ambiguous patterns that are responsible for the issue of semantic-drift, we will also exploit the Web as an additional source of information. Previously, when learning part-whole relations, we relied on Wikipedia itself to disambiguate these ambiguous patterns. This is the only difference between our technique for mining causal relation and part-whole relations. After acquiring high quality patterns that express causality from Wikipedia, we use them to extract (domain-specific) causal relation triples from our domain-specific texts.

Our objective with this chapter is therefore two fold. The first objective is to demonstrate that our (previously presented) approach is generic, and can also be applied to mine different types of semantic relations, such as causality, besides part-whole. Our second objective, which is a direct consequence of the first one, is to show that the key characteristics of our approach, as discussed earlier in Chapter 4, also enable us to effectively address the challenges in learning causal relations from domain-specific texts. For example, our approach is minimally-supervised and alleviates the need for knowledge resources (e.g. annotated training data), whereas most existing techniques are supervised and rely extensively on these resources, which may not always be available. Also, we successfully extract both explicit causal relations, which are expressed by causal verbs like "*(to) cause*", as well as implicit causal relations, which are implicitly expressed by verbal and non-verbal patterns, such as the verb *"reduce"* and the adjectival phrase *"due to"*. The majority of techniques developed to date focus on explicit relations. As can be expected, implicit relations are harder to detect than their explicit counterparts.

We will also evaluate our approach by estimating its performance on a real-life corpus of sparse, informally-written and domain-specific texts provided by our industrial partners. As will be seen during our experimental evaluations, our approach achieves relatively high performance in extracting causal relations (both implicit and explicit) from these texts in a minimally-supervised fashion.

A note to the reader:
This chapter and the previous one share many similar elements since they both deal with the common topic of minimally-supervised semantic relation extraction. Therefore, we will adopt a relatively concise style to describe the contents that have been already presented before in order to avoid the repetition of overlapping materials. In particular, since our relation extraction technique was already presented in the previous chapter, it will not be discussed in detail again. However, we will highlight those aspects that are peculiar to causal relations and their extraction, which is the core of this chapter.

The work described in this chapter is an edited version of the earlier published book chapter:

- Ittoo, A.; Bouma, G.: "Extracting Explicit and Implicit Causal Relations from Sparse, Domain-Specific Texts", in *Lecture Notes in Computer Science, Vol. 6716, pp. 52-63, Springer*.

5.1 INTRODUCTION

Causal relations, between causes and effects, are a complex phenomenon, pervading all aspects of life. In Natural Language Processing (NLP), several techniques for mining explicit causal relations from texts exist. Explicit relations are those that are manifested by explicit causal patterns, predominantly assumed to be causal verbs, such as *"(to) cause"* and its (near) synonyms like *"(to) induce"* or *"(to) generate"* [145, 146, 147]. Explicit patterns establish a causal link between a distinct causal-agent (e.g. "rain") and a distinct effect (e.g. "floods"), as in

- *S1= "[rain$_{causal\text{-}agent}$] causes [floods$_{effect}$]"*.

Besides focusing on explicit causal relations, most of the existing algorithms are heavily supervised. They rely extensively on hand-crafted patterns or on annotated training data, which facilitate the detection of causality in texts. In addition, these algorithms have been primarily applied over large corpora of standard texts, such as newspaper articles [145, 146, 147, 148]. As a result, the task of mining causal relations from domain-specific texts has been largely overlooked despite the crucial role that these relations play in specialized disciplines. In our domain of Product Development-Customer Service (PD-CS), causal relations encode valuable operational knowledge that can be used to support various types of corporate activities. For example, in "broken cable *resulted in* voltage loss", the causal relation between the causal-agent "broken cable" and the effect "voltage loss", established by the pattern "*resulted in*", indicates that some malfunctioning product suffered from a "voltage loss", and that this was due to a "broken cable". Such information provides business organizations with a better understanding of product failures. It enables them to devise appropriate corrective/remedial measures for subsequently improving product quality. Similarly, the causal relation between "(new) frequency oscillator" and "(blurred) image", established by the pattern *"leads to"*, as in "new frequency oscillator always *leads to* blurred image", can be exploited to acquire further insights on causes of customer dissatisfaction.

However, the extraction of causal relations from domain-specific texts poses a number of challenges that existing techniques do not adequately address. These difficulties are listed below.

1. Implicit causal relations. Most existing techniques fail to extract the implicit causal relations that abound in domain-specific texts. Unlike their explicit counterparts, implicit causal relations are realized by verbal and non-verbal patterns that are not synonymous with the verb *"(to) cause"*. Examples include resultative verbs like *"reduce"* and adjectival phrases like *"due to"*. These types of implicitly expressed causality are harder to detect than explicit ones (Challenge 1). We will elaborate upon implicit causal relations in Section 5.3.

2. Lack of domain-specific knowledge resources. Domain-specific knowledge resources, such as hand-crafted patterns and annotated training data, to support existing algorithms are scarce in specialized domains (Challenge 2).

3. Data sparsity. Minimally-supervised algorithms alleviate the need for domain-specific knowledge resources, such as training data. However, their performance is severely compromised by the issue of data sparsity, which characterizes domain-specific corpora [22, 39, 118] (Challenge 3). Previously, we saw that Wikipedia could be exploited to support the extraction of domain-specific part-whole relations. It remains to be seen whether it can also be leveraged upon for facilitating the mining of causal relations.

4. Ambiguous patterns/semantic-drift. Causal relations are realized by a wide range of patterns, which are ambiguous and establish a causal relation only in restricted contexts. These patterns are responsible for the phenomenon of semantic-drift, which causes the performance of minimally-supervised algorithms to deteriorate (Challenge 4).

It is worth noting that challenges 1 and 4 are longstanding issues that have plagued the automatic extraction of high quality causal relations from texts. They are thus not peculiar

to our domain, although their occurrences are more pronounced in domain-specific texts. Furthermore, Challenges 2-4 were also observed in Chapter 4 when mining part-whole relations, while Challenge 1 is specific to causal relations.

In this chapter, we will rely on a slightly modified version the approach presented in the previous chapter for mining part-whole relations, and use it for the task of extracting causal relations. As already indicated, our aim in doing so is two fold. First, we demonstrate that the proposed approach is generic, and can be applied for mining different types of semantic relations, including part-whole and causality. Second, we show that the main strengths of our approach, which have been described before, enable us to overcome the challenges in learning causal relations. Specifically,

1. Our approach accurately extracts both explicit and implicit causal relations (Contribution 1 in response to Challenge 1).

2. It is a minimally-supervised technique, and thus, eschews the need for domain-specific knowledge resources, which are not always available (Contribution 2).

3. It overcomes the issue of data sparsity posed by domain-specific texts by relying on additional information that it acquires from Wikipedia, which serves as a knowledge-base (Contribution 3).

4. It also mitigates the negative impact of semantic-drift on the performance of minimally-supervised techniques (Contribution 4).

With the exception of the first point pertaining to the extraction of explicit/implicit causality, the others are not "core contributions" per se. They have already been introduced in the preceding chapter. But we will still refer to them as "contributions" for the sake of convenience.

These challenges and our corresponding contributions to overcome them will be described in details in Section 5.3 and 5.4.

To evaluate our approach, we estimated its performance in extracting causal relations from a corpus of real-life documents generated in the PD-CS domain. The same corpus had been used in Chapter 4 for learning part-whole relations. The experimental results indicated that our proposed approach accurately extracts both explicit and implicit causal relations from sparse, domain-specific texts with minimal supervision. In addition, we also show the gains in performance that are possible by overcoming the issue of semantic-drift.

This chapter is organized as follows. In Section 5.2, we provide an overview of causal relations and various efforts that have investigated their automatic extraction from texts. Then, Section 5.3 presents some of the difficulties involved in automatically learning causal relations, especially from domain-specific texts. In Section 5.4, we describe the key characteristics (i.e. "contributions") of our propose approach, which effectively address these difficulties. Finally, experimental evaluations to empirically gauge the performance of the proposed approach are presented in Section 5.5.

5.2 Related Work

5.2.1 Cause-Effect Relations

The knowledge between causes and their corresponding effects, encoded in causal relations, provides the basis for rational decision making and problem-solving [149]. Causal relations play a fundamental role in many disciplines, including philosophy, psychology and linguistics. Some studies have even suggested that causality is more important and useful than other semantic relations [150].

In philosophy, several studies have been dedicated towards understanding and defining the notion of causality. Hume [151], for example, defines causality as the associations in the brain ("mind") between two ideas (events) as a result of experiencing their regular co-occurrence. A typical illustration is between "rain" and "flood", which are two events that tend to be highly associated in our minds. In psychology, recent research has attempted to investigate the mechanism by which we determine the causes (and effects) of observed phenomena. According to these studies, the causal inferencing procedure of humans is based on empirical data, such as the frequency with which an observation is made. This is supplemented by world knowledge, and shaped by social beliefs and cultural factors [149]. In addition, numerous other studies have shown that the ability to identify and infer causal relations is crucial for the comprehension of narrative texts. It has also been demonstrated that events in narratives are easier to recall if they are connected by a large number of causal relations to the remainder of the text [149].

Causal relations and their automatic extraction from texts have also been investigated in Natural Language Processing (NLP), which is the focus of this chapter. Before presenting the NLP techniques for causal relation extraction, we will briefly describe the types of causal relations.

5.2.2 Types of Causal Relations

Our above discussion suggests the existence of different types of causal relations, for example, in philosophy, psychology and linguistics. Thus, causality is a non-primitive relation, which can be decomposed into different (sub)types. Several studies have attempted to characterize the different types of causal relations. One of the earliest efforts was that of Aristotle [149], who discriminated between 4 main types of causality. The first 2 types refer to the material and the form of an object, which cause the object to exist. The third type is related to any mechanism that causes the object to move, come to rest or change, while the last type refers to the reason that spurs the change.

A more recent description of the types of causal relations is that of Barriere [152], who distinguishes between Existential Causality (existence dependency) and Influential Causality (influence dependency). Existential causality pertains to the existence of an event. It can further be decomposed into 4 subtypes, namely *creation*, *prevention*, *destruction* and *maintenance*. Influential causality concerns the features of an event. It can also be decomposed into 4 more specialized types, namely, *modification*, *decrease*, *increase* and *preservation*.

However, these efforts merely provide an overview of the possible types of causal relations. They do not rigorously and unambiguously define these types in a formal taxonomy, like that of Keet and Artale [114] for part-whole relations. To date, such a formal typology of causal relations is still lacking [149], and this can be attributed to 2 main reasons:

1. Causality is a complex and multi-faceted phenomenon that is hard to formalize precisely. For example, it is difficult to accurately capture the subtle difference between causal verbs like *"cause"* and other transitive non-causal verbs like *"kick"*.
2. It is equally challenging to accurately specify the types of events that can participate in a causal relation. This is unlike the case for part-whole relations, where semantic selectional restrictions can be imposed to constrain the instances that are allowed to participate in the different (part-whole relation) types. It is therefore reasonable to expect that a larger number and more diverse events pairs participate in causal relations than in part-whole relations.

5.2.3 EXTRACTING CAUSAL RELATIONS FROM TEXTS

We will now briefly review some well-known and recent NLP techniques for mining causal relations from texts.

Khoo et al. [109] developed an approach for identifying causal relations from newspaper texts. They manually constructed linguistic patterns that express causality, such as *"[effect] is the result of [cause]"*. Each pattern consisted of a causality marker, such as *"result of"*, and two slots, respectively corresponding to the cause and the effect. To construct these causality-encoding patterns, they consulted various sources including the Longman Dictionary of Contemporary English and the Roget International Thesaurus. The full list of hand-crafted patterns employed is available from [153]. Their algorithm then extracted, as causal relations, all the text segments in a corpus that matched the patterns. They evaluated the performance in extracting causal relations from the Wall Street Journal (WSJ) articles, and reported a precision of 0.25 and a recall of 0.68. One of the reasons accounting for the low precision was the inability of the hand-crafted patterns to capture the peculiar structure of some sentences. As a result, these patterns extracted the wrong sentence fragments as causal relations.

A similar approach was adopted by Khoo et al. [154] for mining causal relations from bio-medical texts. In this study, 68 linguistic patterns involving explicit indicators of causality, such as causal verbs, were manually created by consulting 200 bio-medical articles of the Medline [155] collection. The patterns were then applied to a new collection of 30 Medline documents, and all the text fragments that matched the patterns were extracted as causal relations. They achieved a precision of 0.62 and a recall of 0.76.

Girju [145] developed a supervised technique for mining causal relations. In her work, 429 concept pairs that participated in a causal relation, such as "earthquake" and "tidal waves", were first selected from the WordNet lexico-semantic dictionary [19]. Explicit causal patterns, such as the verbs *"generate"* and *"cause"*, which connected the concept pairs in Web documents, were then collected (after a web-search). Next, the sentences that contained these verbs in the L.A. Times corpus were manually annotated as positive (or negative) examples of causality. A decision-tree classifier was then trained on the annotated corpus, and used to detect new occurrences of causal relations. It achieved a precision of 0.74 and a recall of 0.89. Another supervised approach for mining causal relations is that of Beamer et al. [108]. In this study, they trained a support vector machine (SVM) over the SemEval 2007 Task 4 corpus, which was labeled with examples of causal relations. Their technique achieved an accuracy of 0.78 in identifying cause-effect noun pairs. In the work of Berthard et al. [147], an SVM, trained over manually-annotated documents of the WSJ, achieved a precision of 0.24 and recall of 0.80 in detecting causal relations. A similar approach is that of Rink et al. [146], where they trained an SVM on sub-graphs generated from annotated WSJ sentences, and used it to identify new occurrences of causality with a precision of 0.26 and recall of 0.78.

5.3 CAUSAL RELATION EXTRACTION CHALLENGES

It can be seen from the preceding section that the majority of extant techniques for causal relation extraction have primarily been applied on documents in the general and scientific domains. As discussed before, the extraction of relations from documents in these domains is facilitated by several factors, namely, the large collections of well-written texts, which provide reliable linguistic and statistical evidence, and the availability of resources, such as annotated data. However, these desirable characteristics are rarely replicated in corporate domains like PD-CS. As a result, the extraction of causal relations from corporate texts introduces several important challenges, which existing techniques do not adequately address. In this section, we will elaborate on these challenges (introduced in Section 5.1), and discuss the shortcomings exhibited by current techniques.

Challenge 1: Implicit causal relations.
Domain-specific texts abound in causal relations that are implicitly expressed. These implicit relations are manifested by patterns that do not have any (explicit) causal valence, but they subtly bias the reader into associating certain events in the texts with causal-agents or effects [149]. Implicit causal relations/patterns are harder to detect than their explicit counterparts. In our work, we concentrate on 3 types of implicit relations.

- Relations of the first type, *T1*, are realized by resultative verbal patterns [156]. These include verbs such as "*increase*", "*reduce*", "*kill*", "*remain* (as)" or "*become"*. Relations of this type inherently specify (part of) the resulting situation, as in "the temperature *increased*".
- The second type, *T2*, involves patterns that make the causal-agents inseparable from the resulting situation [149]. Such patterns include verbs like *"mar (by)"* and "*plague(by)"*. For example, in "white spots *mar* the x-ray image*"*, the causal-agent "white spots" is an integral component of the result "marred x-ray image".
- The last type of implicit causal relations, *T3*, involves non-verbal patterns. They are mostly noun and adjectival phrases like *"rise in"* and *"due to"*, such as "there was a *rise in* the voltage level" and "replaced camera *due to* horizontal calibration problem".

Challenge 2: Lack of domain-specific resources.
Existing algorithms for mining causal relations are heavily supervised. They rely extensively on hand-crafted patterns or on manually-annotated data, labeled with examples of causal relations, to facilitate the detection of causality in texts. However, these resources are not available in many specialized domains. Their creation and maintenance is hindered by the Knowledge Acquisition bottleneck. Furthermore, hand-crafted patterns have limited coverage and may fail to capture all the different ways in which causality can be expressed in texts, as shown in the work of Khoo et al. [109].

Challenge 3: Data sparsity
Unlike supervised techniques, minimally-supervised algorithms alleviate the need for domain-specific knowledge resources, such as labeled training data. However, as described before, these algorithms achieve optimal performance on large document collections, which provide sufficient redundancy and evidence to support their (statistical) inferences [130, 131]. However, domain-specific texts, generated in specialized disciplines, are of limited size and sparse [22, 39, 118]. Data sparsity not only affects the precision of minimally-supervised algorithms by making their statistical inferences less reliable, but is also detrimental to their recall.

Previously, we saw that Wikipedia could be exploited to support the extraction of domain-specific part-whole relations. It remains to be seen whether it can still be leveraged upon for facilitating the mining of causal relations.

Challenge 4: Ambiguous pattern/semantic-drift
Causal relations are realized by a wide variety of patterns, a significant portion of which are ambiguous. These patterns express causality only in restricted contexts. The pattern "*lead to*", for example, is ambiguous. It establishes a causal relation when it connects "smoking" and "cancer" as in "smoking *leads to* cancer"; but not when it relates "roads"

and "Rome" as in "all roads *lead to* Rome"*.* Consequently, when these ambiguous patterns, like *"lead to"*, are used by a minimally-supervised algorithm, they will promote the extraction of invalid cause-effect pairs like "roads-Rome". In turn, the invalid pairs will favor the extraction of other unreliable patterns, like *"found in"*, as in "roads *found in* Rome". Subsequently, incorrect causal pairs/patterns will start to dominate the iterative learning procedure of the minimally-supervised technique, hindering the detection of causal relations. This phenomenon, as we saw previously, is known as semantic-drift [119], and compromises the performance (Challenge 4).

5.4 CONTRIBUTIONS

The preceding section highlighted the lacunae in extant techniques developed for mining causal relations. We fill this gap and answer our second research question with our relation extraction approach. The main strengths of our approach are as follows.

1. We extract explicit relations, which are (explicitly) expressed by causal verbs as well as implicit ones, which are (implicitly) expressed by verbal and non-verbal patterns that do not have any causal connotation. These include relations of the type T1, T2 and T3, as described in Section 5.3 (Contribution 1 in response to Challenge 1).

2. The proposed approach is minimally-supervised, and eschews the need for domain-specific knowledge resources, including hand-crafted patterns and annotated training data, which are expensive to produce and not always available. Instead, it is initialized with a handful of event/instance (term) pairs, called seeds, which instantiate a causal relation, such as "hiv-aids". It recursively uses these pairs to acquire high quality causal relations, which are expressed by reliable (implicit and explicit) causal patterns. (Contribution 2).

3. To overcome the issue of data sparsity posed by the domain-specific texts, we first acquire a set of patterns that accurately express causality from a large corpus, which we use as a knowledge-base. In our approach, we employ the English Wikipedia as the knowledge-base. We then use the reliable patterns from the knowledge-base to identify occurrences of causal relations from the domain-specific texts. This also shows that despite its broad-coverage, Wikipedia can still be exploited as a knowledge-base to support the extraction of domain-specific causal relations (Contribution 3).

4. To mitigate the issue of semantic-drift, we propose a technique for preventing invalid cause-effect pairs (e.g. "path-garden"), which are connected by ambiguous causal patterns (e.g. *"lead to"*; "path *leads to* garden*"*), from triggering the issue of semantic-drift. This enables the extraction of more accurate causal relations (Contribution 4).

Our proposed approach for extracting domain-specific causal relations is discussed in the next section.

5.5 Framework for Domain-Specific Causal Relation Extraction

Our overall framework for mining causal relations from domain-specific texts is depicted in Figure 5.1. Blank arrows represent inputs and outputs, while filled arrows depict the processing performed by the various phases of our framework.

We start by acquiring a set of reliable patterns that express causality from the much larger and broad-coverage Wikipedia corpus, which serves as a knowledge-base. Since causal relations are not available from Wikipedia's (semi-)structured contents, we have to acquire them from its unstructured texts. To this aim, in the Pattern Acquisition phase (Section 5.5.1), we convert each Wikipedia sentence into a corresponding relation triple, consisting of a pattern and a pair that it connects in the sentence. Next, in Causal Pattern Extraction (Section 5.5.2), we employ our minimally-supervised technique to determine which pattern are reliable indicators of causal relations. In this phase, we also rely on the Web as an additional source of evidence for disambiguating polysemous causal patterns. The high quality patterns harvested from the knowledge-base are then used during Causal Relation Extraction (Section 5.5.3) to discover causal relations from domain-specific (target) texts.

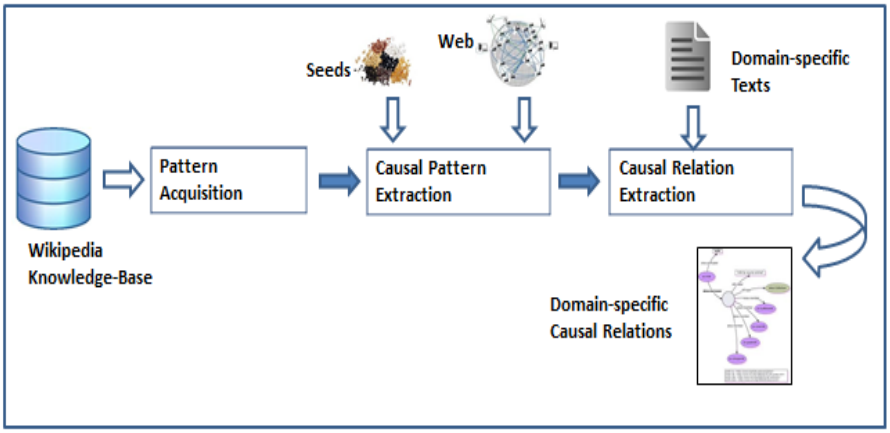


Fig. 5.1: Overall architecture of framework for extracting domain-specific causal relations.

5.5.1 PATTERN ACQUISITION

We syntactically parse Wikipedia's sentences with the Stanford parser [13], and represent the relation between each event pair[18] as the shortest path that connects the pair in the parse trees Such a path corresponds to a lexico-syntactic pattern.

As an illustration, we consider 3 sentences, *S1*, *S2*, and *S3*, from Wikipedia.

- *S1*="hurricanes *are the most severe climatic disturbance in this area and have been known to cause extensive* damage".
- *S2*=" hiv*, the virus which causes* aids*, is transmitted through contact between blood".*
- *S3*="*the* poem *consists of five* stanzas *written in terza rima".*

The corresponding lexico-syntactic patterns identified from each of these sentences, the pairs sub-categorized by the patterns, and the pair-pattern frequency are shown in the 4th, 3rd, 2nd, and 1st columns of Figure 5.2. ARG1 and ARG2 are generic placeholders, representing the pairs in the patterns.

14	hurricane	damage	ARG1+nsubj < cause > dobj+ARG2
11	hiv	aids	ARG1+nsubj < cause > dobj+ARG2
5	stanza	poem	ARG2+nsubj < consist > prep+of+pobj+ARG1

Fig. 5.2: Lexico-syntactic patterns, instances, and statistics from Wikipedia.

As can be seen from the above illustrations, a major strength of our lexico-syntactic patterns is that they abstract over surface text, and neutralize word order or morphological variations. Thus, they overcome the limitations of traditional surface-string representations. For example, we derive the single, most general pattern, *"cause"*, to represent the relations in *S1* and *S2*. Our patterns also capture long range dependencies, regardless of the distance between related pairs, and their positions in surface texts. For example, we are able to capture the relation between the pair "hurricane-damage" and the pattern *"cause"* in *S1*, despite their not appearing in adjacent positions in surface text.

Since Wikipedia is a broad-coverage corpus, the pair-pattern triples extracted at this stage encode different types of semantic relations. For example, the pattern *"cause"*, inferred from sentences *S1* and *S2*, establishes a causal relation between the pairs "hurricane-damage" and "hiv-aids". The last pattern, *"consist of"*, derived from *S3*, is not indicative of causality. Instead, it establishes a part-whole relation between "stanza" and "poem".

We will identify which patterns express causality during Causal Pattern Extraction, discussed in the next section.

[18] At this stage, we only consider nominal events, corresponding to noun phrases.

5.5.2 CAUSAL PATTERN EXTRACTION

We determine which of the patterns are causal using our minimally-supervised algorithm. Unlike supervised algorithms, it alleviates the need for annotated training data or hand-crafted patterns. Instead, to extract causal relations, it is only initialized with a handful of seeds, which are cause-effect pairs like "hiv-aids".

Our algorithm then starts by identifying patterns in Wikipedia that connect these pairs (seeds). The reliability, *r(p)*, of a pattern, *p*, is computed with equation (5.1) [110]. It measures the association strength between *p* and pairs, *e*, weighted by the pairs' reliability, *r(e)*. Initially, *r(e)*=1 for the seeds. In equation (5.1), *E* refers to the set of pairs, and *pmi(e,p)* is the pointwise mutual information [69] between a pattern *p* (e.g. "*cause*") and a pair *e*=*x*-*y* (e.g. "hiv-aids").

$$r(p) = \frac{\sum_{e \in E} \frac{pmi(e,p)}{max_{pmi}} \times r(e)}{|E|} \tag{5.1}$$

Then, we select the top-*k* most reliable patterns, and identify other pairs that they connect in Wikipedia. The pairs' reliability is estimated using equation (5.2). In the next iteration, we select the top-*m* most reliable pairs, and extract new patterns that connect them. Our recursive procedure of learning patterns from pairs and vice-versa is repeated until a suitable number of patterns *t* have been harvested. The values of *k, m* and *t* will be determined experimentally in Section 5.6.2.

$$r(e) = \frac{\sum_{p \in P} \frac{pmi(e,p)}{max_{pmi}} \times r(p)}{|P|} + purity(e) \tag{5.2}$$

The reliability, *r(e)*, of a pair, *e*, is defined by two components in equation (5.2). In the first component, which is analogous to the pattern reliability (equation (5.1)), |*P*| is a set of patterns. The second component determines the pair's *purity* in instantiating a causal relation. It is analogous to our instance_pair_purity presented in Chapter 4. This component plays an important role in overcoming the issue of semantic drift. It ensures that invalid cause-effect pairs, such as "path-garden", which are connected by ambiguous causal patterns, such as *"lead to"* ("path *leads to* garden"), are awarded much lower scores than valid ones. These invalid pairs will then be discarded based on their lower scores, and prevented from triggering the issue of semantic-drift.

To estimate the purity, we first define an unambiguous pattern, $p_{unambiguous}$, which always conveys causality. As $p_{unambiguous}$, we chose the pattern "*caused by*" since it explicitly and unambiguously expresses causality, and specifies the causal link between a distinct causal-agent and an effect. Also, *"caused by"* was found to co-occur with all our seeds.

In the previous chapter on part-whole relations, we computed the purity of a (part-whole) pair as its probability of being sub-categorized by patterns like $p_{unambiguous}$="*contains"* in Wikipedia. Such an approach was promising since, as we noted before, it is possible to constrain the types of instance pairs that participate in part-whole relations. This gives rise to a relatively "small" number of allowable pairs, which are likely to co-occur with patterns like $p_{unambiguous}$ in a corpus of the size of Wikipedia. However, causal relations can exist between a larger number of event pairs, as discussed in Section 5.2.2. Not all of these pairs will co-occur with causal patterns, e.g. $p_{unambiguous}$="*caused by"*, in a corpus like Wikipedia, despite its rather large size. For example, in our Wikipedia collection, the valid causal pair "attack-death" was sub-categorized only by an ambiguous causal pattern, viz. *"lead to"*. But it was not sub-categorized by unambiguous patterns like $p_{unambiguous}$="*caused by"*. Therefore, if we rely only on Wikipedia, such valid causal pairs will be incorrectly awarded a purity score of 0 (probability of being sub-categorized by $p_{unambiguous}$). Consequently, for the task of causal relation extraction, we computed the purity of an

event pair, extracted from Wikipedia, by consulting the Web as an additional source of information.

This is achieved by querying the Yahoo! search engine[19] with q= "y $p_{unambiguous}$ x" e.g. "flooding (y) *caused by* ($p_{unambiguous}$) rain (x)", and determining the fraction of the top-50 search results that contains phrase q in their summaries.

In this way, invalid pairs, for e.g. "trail-summit", identified by ambiguous causal patterns, for e.g., "*lead to*" ("trail *leads to* summit"), are assigned lower purity values. This is because incoherent queries, like q="summit *caused by* trail" or "trail *caused by* summit", are unlikely to return any Web-search results. The overall reliability scores of these invalid pairs will then be smaller, and they will be discarded. Otherwise, the invalid pairs will be selected in the next iteration, and incorrect patterns connecting them, for e.g. "*pass through*" ("*trail passes though summit*"), will be extracted.
Our purity measure also awards much higher scores to valid pairs, increasing their reliability. They are then selected in the next iteration, and will promote the extraction of other high quality causal patterns. In this way, our purity measure overcomes the issue of semantic-drift (Contribution 4). In addition, by selecting only the most reliable pairs (and discarding invalid ones), the purity measure enables our algorithm to harvest both the explicit and the implicit causal patterns that connect these pairs (Contribution 1).

Figure 5.3 shows 2 example patterns extracted by our minimally-supervised algorithm from 2 corresponding Wikipedia sentences:

- *S4 ="A non-zero current induces a magnetic field by Ampere's law".*
- *S5="The population of the state of Nebraska was increased by positive birthrates".*

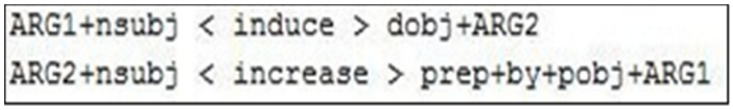


Fig. 5.3. Example of causal patterns learnt from Wikipedia.

In this illustration, ARG1 and ARG2 respectively denote the cause and effect events. The 1st (top-most) pattern explicitly expresses causality with the causal verb "*induce*". The 2nd pattern implicitly expresses causality with the resultative verb *"increase (by)".*

5.5.3 CAUSAL RELATION EXTRACTION

The reliable causal patterns harvested from the much larger Wikipedia corpus are bound to occur in the much smaller target corpus of domain-specific texts. Consequently, we use the Wikipedia patterns to extract domain-specific causal relations from the target corpus. Such a strategy enables us to overcome the issue of data sparsity that would have been encountered if the causal relations (patterns, pairs) were extracted directly from the domain-specific corpus. Our Wikipedia knowledge-base, being a large collection of well-written texts, provides us with high quality linguistic evidence (e.g. syntactic dependencies) as well as statistical evidence (e.g. significant pair-pattern counts, supplemented with additional information from the web), which we exploit to discover reliable causal patterns from its contents. By leveraging upon these patterns, we are then able to accurately identify causal relations from our domain-specific texts (Contribution 3).

A domain-specific causal relation is made up of a causal pattern and the events that it connects in the target corpus. We consider both nominal events, such as noun phrases like "loose connection", and verbal events, such as verb phrases like "replacing the cables". Figure 5.4 illustrates a domain-specific causal relation, extracted from the sentence *S6="replacing the cables generated intermittent x-rays"* in the target corpus.

[19] Using the Yahoo! Boss API, http://developer.yahoo.com/search/boss/boss_guide/.

<pattern = "generate" , cause="replace cable" , effect="intermittent x-ray">

Fig.5.4. Domain-specific causal relation triples.

5.6 EXPERIMENTAL EVALUATIONS

This section describes experiments to evaluate the performance of our approach in extracting causal relations from domain-specific texts.

5.6.1 PATTERN ACQUISITION

Using the Stanford parser [13], we syntactically parsed the sentences of the English Wikipedia collection [91] (August 2007 dump, around 500 million words). We identified 2,176,992 distinct lexico-syntactic patterns that connected 6,798,235 distinct event-pairs. Sample patterns, event pairs they sub-categorized and the pair-pattern co-occurrence frequencies were shown in Figure 5.2.

5.6.2 CAUSAL PATTERN EXTRACTION

To identify patterns that express causality, we used our minimally-supervised algorithm. As with other minimally-supervised algorithms, it was initialized with a set of randomly selected seeds. As seeds, we used 20 cause-effect pairs that unambiguously instantiated causal relations, like "bomb-explosion", and that occurred at least 50 times in Wikipedia.

The 1st iteration of our algorithm extracted the 10 most reliable patterns connecting the seeds. Then, 100 pairs that were also connected by these patterns were extracted. In subsequent iterations, we identified 5 additional patterns (using the previously extracted pairs) and 20 additional pairs (using the previously extracted patterns). The values of parameters k and m (Section 5.5.2) were therefore $k=|P|+5$ and $m=|E|+20$, where $|P|$ and $|E|$ are respectively the number of previously collected patterns and pairs.

The recursive process of learning new pairs from patterns, and vice-versa was repeated until we observed a drop in the quality of the harvested patterns (e.g. when non-causal patterns were extracted). The performance peaked in the 14th iteration, where we harvested $t=81$ causal patterns[20] from Wikipedia. Examples are in Table 5.1. Fifteen of the patterns were causal verbs that explicitly indicated causality, for e.g. "*induce*" (column T0). Resultative verbs, for example "*increase*", which implicitly expressed causality, accounted for 50 patterns (column T1). Fourteen non-verbal, implicit causal patterns were also found (column T3). They included nominalizations of resultative verbs, for example, "*increase in*" and adjectives, for example, "*responsible for*". Implicit causal patterns, which made the causal-agent inseparable from the result (effect), were least frequent. Only 2 such patterns, viz. "*mar (by)*" and "*plague (by)*", were detected (column T2).

These results indicate that our proposed approach extracted both implicit and explicit causal patterns with high accuracy (Contributions 1 ,4).

T0 (15/81=18.5%)	**T1 (50/81=61.7%)**	**T2 (2/81=2.5%)**	**T3 (14/81=17.3%)**
cause (by, of)	affect (with, by)	mar (by)	drop in
induce	decrease (by, to)	plague (by)	due to
result in	inflict (by, on)		rise in
spark	limit (by, to)		source of
trigger	prevent (by, to)		responsible for

Table 5.1: Explicit and implicit causal patterns extracted from Wikipedia.

[20] Precision was roughly estimated as 0.78.

During our experiments, we noticed that when different seeds were used to initialize our algorithm, different sets of causal relations (patterns and pairs) were extracted. Some seeds A similar observation was also made when learning part-whole relations in Chapter 4. We will defer further experiments for studying the effect of seeds on the performance of minimally-supervised techniques till Chapter 6.

5.6.3 CAUSAL RELATION EXTRACTION

The domain-specific (target) corpus from which we extracted causal relations contained 32,545 English documents (1,952,739 words). The documents described customer complaints and engineers' repair actions on high-end, electronic equipment.

Out of the 81 causal patterns from Wikipedia, 72 were found to connect nominal and verbal events in the target corpus, yielding a total of 19,550 domain-specific causal relations. Examples are presented in Table 5.2. The 3rd column shows the causal patterns that realized these relations. The 2nd column gives the percentage of the 19,550 relations that contained these patterns. Text snippets in which the causal relations were manifested in the domain-specific corpus are given in 4th column. The 1st column is an identifier for each pattern/relation group.

Id	Freq (%)	Causal Pattern	Linguistic realization
T1	55	destroy	"short-circuit in brake wiring *destroyed* the power supply"
		prevent	"message box *prevented* viewer from starting"
		exceed	"breaker voltage *exceeded* allowable limit"
		reduce	"the radiation *output* was reduced"
T0	23	cause (by)	"gray lines *caused by* magnetic influence"
		induce	"bad cable extension might have *induced* the motion problem"
T3	21	due to	"replacement of geometry connection cable *due to* wear and tear"
		drop in	"there was a slight *drop in* the voltage"
T2	1	mar	"cluttered options *mar*s console menu"

Table 5.2: Domain-specific causal relations from target corpus.

Discussion of Results

The most frequent patterns, participating in around 55% of the extracted relations, were resultative verbs like "*destroy*" and "*exceed*", which implicitly expressed causality. The high frequency of these patterns in our target corpus could be attributed to their common usage for describing product failures as in "short-circuit in brake wiring *destroyed* the power supply", and for reporting observations on product behavior as in "breaker voltage *exceeded* allowable limit". We noted that these relations had optional causal-agents. For example, a causal-agent was specified in "*[short-circuit in brake wiring$_{causal\text{-}agent}$] destroyed the power supply",* but not in "*breaker voltage exceeded allowable limit*". In our PD-CS domain, these relations provide useful information for product quality improvement.

The next most frequent patterns, appearing in around 23% of the extracted relations, were explicit causal verbs, including "*cause by*" and "*induce*". These relations always specified a distinct causal-agent and its effect, as in "*[gray lines$_{effect}$] caused by [magnetic influence$_{causal\text{-}agent}$]*". In the PD-CS domain, these relations can be exploited to facilitate the diagnosis procedure of engineers.

Around 21% of the relations were realized by non-verbal implicit causal patterns, such as noun and adjectival phrases like "*drop in*" and "*due to*" respectively. When they were realized by noun phrases, these relations described unexplained phenomena with unknown causes. Hence, their causal-agents were often unspecified, as in "*there was a slight drop in the voltage*". Conversely, when they were realized by adjectival phrases, their causal-agents were always specified, as in "*[replacement of geometry connection cable$_{effect}$] due to [wear and tear$_{causal\text{-}agent}$]*". In the PD-CS domain, these relations provide pertinent information on customer dissatisfaction and on repair actions of engineers.

The rarest patterns, appearing in around 1% of the relations, were those that implicitly expressed causality by making the causal-agent inseparable from the result, such as *mar*. For example, in "cluttered options *mars* console menu", *[cluttered options$_{causal\text{-}agent}$]* is an integral part of the *[marred console menu$_{effect}$].*

Nine (out of 81) patterns from Wikipedia were not found in the target corpus. They included explicit causal verbs, like "*spark*"; implicit resultative verbs, like "*end*" and "*outstrip*"; and implicit non-verbal expressions like "*growth*". These patterns were unlikely to occur in our corpus of customer complaints and of engineers' repair actions.

Performance Evaluation

To evaluate the performance of our approach, we randomly selected 3000 of the extracted causal relations, equally distributed across the groups T0, T1, and T3 of Table 5.2 (i.e. 1000 explicit relations with causal verbs, 1000 implicit relations involving resultative verbs, and 1000 implicit relations with non-verbal expressions). Relations in the group T2 were omitted as they were too sparse.

The evaluation set of 3000 relations was manually inspected by human judges, and 2295 were deemed to correctly express causality (*true_positive*). The remaining 705 relations were incorrect (*false_positive*). They were realized by causal patterns, for e.g., *"due to"*, which did not connect valid causal events, for e.g. "screen-today", in the target corpus, as in "screen *due to* arrive today". Using equation (5.3), we then estimated the precision (P) of our approach as 0.77. To determine the recall, a gold-standard of 500 valid causal relations was manually constructed from a subset of the target corpus. From this sub-corpus, our approach was able to detect 410 of the gold-standard relations (*true_positive*), but failed to discover remaining 90 (*false_negative*). The recall (R) was computed as 0.82 using equation (5.4).

$$precision = \frac{true\_positive}{true\_positive + false\_positive} \tag{5.3}$$

$$recall = \frac{true\_positive}{true\_positive + false\_negative} \tag{5.4}$$

The F1 score was then estimated as 0.79 by equation (5.5).

$$F1 = \frac{2 \times P \times R}{P + R} \tag{5.5}$$

Our performance scores of 0.77 (precision), 0.82 (recall) and 0.79 (F1) suggest that our proposed approach accurately extracted high quality explicit and implicit causal relations from the domain-specific texts. It relied solely on a handful of randomly selected seeds, and alleviated the need for hand-crafted patterns or for prior training on large amounts of annotated data. Our scores compare favorably with those of other supervised techniques that have been applied for mining explicit causal relations from large collections of well-written texts. For example, Berthard et al. [147] reported a precision of 0.26, a recall of 0.78 and a corresponding F1 score of 0.39 for their supervised technique, which used an SVM trained on Wall Street Journal articles that were manually annotated with examples of causality. Our approach, on the other hand, was minimally-supervised, overcoming the need for knowledge resources (Contribution 2). It successfully addressed the issue of data sparsity posed by our domain-specific corpus by leveraging upon Wikipedia as a knowledge-base (Contribution 3). In addition, we overcame the issue of semantic-drift, caused by ambiguous causal patterns (Contribution 4), and successfully detected both explicit and implicit causal relations (Contribution 1).

Significance of purity Measure

We conducted another set of experiments to illustrate the significance of our event pair *purity* measure (Section 5.5.2) in overcoming the issue of semantic-drift and extracting the most reliable causal patterns and relations. We re-implemented our minimally-supervised algorithm such that the reliability, *r(e)*, of a pair, *e*, was estimated with only the 1st component of equation (5.2). The pattern reliability measure in equation (5.1) was unchanged.

Our algorithm was initialized with seeds, and was run over the Wikipedia collection as before. We observed that invalid causal pairs, for example, "street-exit", which were connected by many ambiguous causal patterns, such as *"lead to"* and *"result in" ("street leads to exit", "street results in exit")*, were awarded higher reliability scores than valid ones. Subsequently, these invalid pairs were selected, and other incorrect patterns that connected them, such as *"at" ("street at exit")*, were in turn extracted, giving rise to the phenomenon of semantic-drift. The quality of the harvested patterns deteriorated in much earlier iterations, compared to our original implementation. Optimal performance was achieved in the 7th iteration, where 34 causal patterns were identified[21]. In addition, we found out that many of the implicit causal relations, which had been detected before, were now missing from the results.

We used the 34 causal patterns extracted by the new implementation from Wikipedia to extract domain-specific causal relations from the target corpus. The precision, recall and F1 scores were calculated by equations (5.3), (5.4) and (5.5) as 0.66, 0.68 and 0.67 respectively. These scores indicate a drop in the performance compared to our original algorithm, and suggest that the new implementation is less accurate in extracting causal patterns and relations. The results reveal that our purity measure for disambiguating ambiguous patterns contributes to the performance by overcoming the issue of semantic-drift. The purity measure also enables us to harvest the wide range of patterns participating in both explicit and implicit causal relations, as earlier mentioned.

5.7 CONCLUSION

In this chapter, as in the preceding one (Chapter 4), we were also concerned with the second research question, *RQ2: How to accurately detect occurrences of semantic relations from domain-specific, sparse and informally-written corporate texts?*

We focused on causal relations since they encode valuable information, which can be exploited to support various types of corporate activities, such as Business Intelligence. We addressed the second research question by adopting an approach, which is a slightly modified version of the technique previously presented in Chapter 4 for part-whole relation extraction. This approach involved the use of a minimally-supervised relation extraction algorithm. Such an algorithm enabled us to overcome the need for domain-specific knowledge resources like hand-crafted patterns and annotated training data, which are not always available in specialized domains. Since the sparsity that characterized our domain-specific texts is detrimental to the performance of minimally-supervised techniques, we also relied on Wikipedia as a knowledge-base. Our minimally-supervised algorithm was then applied on the Wikipedia knowledge-base for acquiring patterns that reliably express causal relations.

A major difficulty in learning causal relations is that they are realized by many ambiguous patterns. These patterns do not always express causality, and may also connect invalid causal pairs. These ambiguous patterns and invalid pairs are responsible for the issue of semantic-drift, whereby the relations ultimately extracted by a minimally-supervised algorithm are different from the target relation, as instantiated by the initializing seeds. To mitigate the negative influence of semantic-drift on the performance, we defined a purity measure. It assigned much lower scores to invalid causal pairs that were connected by

[21] Precision was roughly estimated as 0.65.

ambiguous causal patterns. These invalid pairs were then rejected, and prevented from triggering the semantic-drift phenomenon. Conversely, valid causal pairs were awarded much higher scores. They were then selected and used to harvest other reliable causal patterns, which was beneficial to our performance.

After harvesting the reliable causal patterns from the Wikipedia knowledge-base, we used them to extract causal relation triples from the domain-specific texts. Each triple consisted of a causal pattern and a (domain-specific) event (term) pair in the domain-specific texts.

We evaluated our approach on a collection of real-life texts provided by our industrial partners. Our experimental results revealed that our approach successfully addressed the challenges involved in learning causal relations from domain-specific documents. For example, our approach was minimally-supervised and did not require additional information from domain-specific knowledge resources, which are scarce in most corporate domains. It overcame the issues of data sparsity and of semantic-drift, which are detrimental to the performance of minimally-supervised techniques. Our approach also detected both implicit and explicit causal relations. Our performance scores compared favorably with other supervised techniques, such as that of Berthard et al. [147], which focused on learning explicit causal relations from large collections of well-written texts.

The main findings/conclusions of this chapter reinforced those of the previous chapter. They are summarized below:

1. Minimally-supervised algorithms seem to be more appropriate for use in specialized domains, where knowledge resources are scarce. As illustrated in our experiments, our approach achieved high accuracy in extracting domain-specific causal relations, even though it did not rely on domain-specific resources such as annotated data.
2. Wikipedia can be used as knowledge-base to overcome the issue of data sparsity posed by domain-specific corpora.
3. Despite its broad coverage nature, Wikipedia can still be exploited to support domain-specific tasks, such as the extraction of (domain-specific) causal relations.
4. Overcoming the issue of semantic-drift plays an important role in the discovery of accurate relations, and is thus beneficial to the performance.

In addition to the above, we can also conclude that:

5. Our relation extraction approach can be considered as generic since it has been successfully applied for mining different types of semantic relations, such as part-whole and causal relations.
6. Implicitly expressed causal relations are as important as, if not even more important than, their explicit counterparts. As illustrated in our experiments, around 80% of all causal patterns, both in a general corpus like Wikipedia and in our domain-specific texts, implicitly expressed causality. However, most techniques developed to date have overlooked the extraction of implicit causal relations, and have predominantly focused on detecting explicit relations.

Further discussions on our approach for mining causal relations with respect to RQ2 will be given in Chapter 8.

Concerning the business applications, our causal relation extraction technique can be used to detect causes of product failures. Once these failures are identified, they can be incorporated in the product development process to improve product quality.

CHAPTER 6–EFFECT OF SEEDS ON MINIMALLY-SUPERVISED ALGORITHMS

CHAPTER SUMMARY

In our two previous chapters, we have employed a minimally-supervised algorithm for mining part-whole and causal relations. Minimally-supervised algorithms are initialized with seeds, which are event/instance pairs depicting the relation of interest, such as "engine-car" for part-whole. Following a common practice adopted in previous research, the initializing seeds employed by our algorithms were randomly selected. However, a recurrent phenomenon that we observed when extracting both the part-whole and causal relations was that the seeds had an unpredictable influence on the relations (patterns and pairs) harvested by our algorithm. For example, we noted that some seeds were more fertile than others, and enabled the discovery of more accurate relations.

In this chapter, we will devote ourselves to study the effect of seeds on the performance of minimally-supervised algorithms. Our investigation will cover both the English and Dutch languages. As will be demonstrated during the experimental evaluations, the traditional practice of initializing minimally-supervised techniques with randomly selected seeds does not always ensure the discovery of the most accurate relations, and thus, is not beneficial to the performance. Our results suggest that future research efforts targeted at minimally-supervised relation extraction should adopt a more principled approach to seed selection.

The work described in this chapter is an abridged version of the manuscript:

- Ittoo, A., Bouma, G., "On Learning Subtypes of the Part-Whole Relation: Do Not Mix your Seeds", in Proc. Of the 48th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 1328-1336.

An earlier version was published as the book chapter

- Ittoo, A.; Bouma, G., Semantic Selectional Restrictions for Disambiguating Meronymy Relations", in Computational Linguistics in the Netherlands, pp. 83-98, LOT Occasional Series.

6.1 INTRODUCTION

As we saw in the preceding chapters, several minimally-supervised algorithms have been proposed for mining semantic relations from texts. The main strength of these algorithms is that they do not require large amounts of annotated data for training. Instead, they are initialized using a few term pairs, called seeds, which denote the relation of interest. For example, in Chapters 4 and 5, we initialized our minimally-supervised algorithm with the seeds "engine-car" and "hiv-aids" in order to learn part-whole and causal relations respectively. In these experiments, we noted that the initializing seeds had an unpredictable influence on the relations ultimately extracted by our algorithm. For example, we observed that some seeds led to the extraction of more accurate relations, and hence, to better performance, than others.

Most studies to date have overlooked the influence of the initializing seeds on the performance of minimally-supervised algorithms. Such an investigation is particularly relevant for non-primitive relations, such as part-whole. As we saw in Section 4.2.1, the taxonomy of Keet and Artale [114] distinguishes between 8 part-whole relation (sub)types. However, the majority of relation extraction efforts ignore the distinction between these different part-whole relation types. They assume the existence of a single part-whole relation that subsumes all the other types. For example, to initialize our algorithm for learning part-whole relations, we randomly selected part-whole instance pairs, such as "engine-car" and "guitarist-band", as seeds. We did not consider the types of part-whole relations instantiated by the selected seeds. Thus, we had a mixed set of seeds, each of which instantiated a different part-whole relation type. For example, according to the taxonomy of Keet and Artale [114], "engine-car" instantiated a part-whole relation of the type *structural part-of*, while "guitarist-band" corresponded to a *member-of* part-whole relation. Similarly, to initialize the Espresso algorithm for learning part-whole relations, Pantel and Pennacchiotti [110] defined a mixed seed set, overlooking the distinctions between the part-whole relations types instantiated by the seeds. For example, their seed "leader-panel" instantiated a *member-of* relation, while "oxygen-water" instantiated a *sub-quantity-of* relation.

Therefore, our aim in this chapter is to investigate the effect (or lack thereof) that different initializing seeds may have on the performance of minimally-supervised algorithms for Information Extraction (IE). To study this phenomenon, we will employ a stripped-down version of our minimally-supervised algorithm, presented in Chapters 4 and 5 for part-whole and causal relation extraction. This version does not rely on a knowledge-base and does not incorporate additional mechanisms for mitigating the issue of semantic-drift. It is therefore very similar to the basic Espresso algorithm of Pantel and Pennacchiotti [110]; the only difference being our use of lexico-syntactic patterns compared to the surface-strings of Espresso. Such a plain-vanilla implementation enables us to isolate the object of our investigation, and to better examine the effect of seeds on the relations extracted by our minimally-supervised IE algorithm. Given a set of initializing seeds, our algorithm simply extracts all the patterns that connect the seeds in a corpus. The patterns are then used to acquire new pairs from the corpus, and a recursive procedure of learning new patterns from the pairs and vice-versa is repeated until the desired number of relations (patterns/pairs) have been harvested.

Also, we will concentrate on experiments involving the extraction of part-whole relations. This is because the types of part-whole relations and the pairs (seeds) that are allowed to participate in each type have been rigorously defined in formal taxonomies, such as that of Keet and Artale [114]. Such an unambiguous description of the types and allowable pairs enables us to conduct a systematic and broader investigation. For example, we will be able to precisely analyze the effect of initializing our algorithm with seeds that instantiate only 1 particular part-whole type (e.g. *member-of*) and compare the results with the traditional practice of using a set of mixed seeds. This kind of analysis may not be possible for other semantic relations, including causality, since it is difficult to precisely decompose them into different types and to specify the allowable instance/event pairs (seeds) that can participate in each type, as we discussed in Section 5.2.2.

To make our investigation more manageable, we examine the effect of seeds on the performance of minimally-supervised algorithms along 3 dimensions. This gives rise to 3 corresponding main issues (questions) that we address in this chapter:

1. Is Information Extraction (IE) harder when we use distinct seed sets that instantiate the individual types of part-whole relations? That is, we determine whether the performance of our IE algorithm in learning the individual part-whole relation types increases (due to more coherency in the relations' linguistic realizations) or drops (due to fewer examples), compared to the traditional practice of considering a single part-whole relation.

2. Are the patterns and pairs discovered when using seeds that instantiate a particular part-whole type confined to that type only? That is, we investigate whether our IE algorithm discovers examples representative of the different types by targeting one particular part-whole relation type.

3. Are more unique examples discovered when our IE algorithm is initialized with distinct seed sets that instantiate the individual part-whole relation types? That is, we determine whether a wider variety of unique patterns and pairs are extracted when we target the different types of part-whole relations instead of considering a single part-whole relation that subsumes all the different types.

To address these issues, we bootstrapped our minimally-supervised algorithm, with different seed sets for the various types of part-whole relations, and analyzed the harvested pairs and patterns, as will be described in the remainder of this chapter. We will start with a brief overview of the formal taxonomy of Keet and Artale [114], which describes various types part-whole relations in Section 6.2. Our work relies heavily on this taxonomy. Section 6.3 describes our adopted methodology for addressing our 3 questions. Results of our experiments are given in Section 6.4 and we conclude in Section 6.5

6.2 MERONYMIC AND MEREOLOGICAL PART-WHOLE RELATIONS

Keet and Artale [114] developed a formal taxonomy, distinguishing between transitive and intransitive part-whole relations. Transitive (part-whole) relations are known as *mereological* relations, while intransitive ones are known as *meronymic* relations. Meronymic and mereological part-whole relations, as defined in the taxonomy of Keet and Artale [114], are shown in Table 6.1.

Meronymic		
Sub-type	**Description**	**Example**
member- of	between a physical object (or role) and an aggregation	player-team
constituted-of	between a physical object and an amount of matter	clay-statue
sub-quantity-of	between amounts of matter or units	oxygen-water or m-km
participates-in	between an entity and a process	enzyme-reaction
Mereological		
Sub-type	**Description**	**Example**
involved-in	between a phase and a process	chewing-eating
located- in	between an entity and its 2-dimensional region	city-region
contained-in	between an entity and its 3-dimensional region	tool- trunk
structural part-of	between integrals and their (functional) components	engine-car

Table 6.1: Meronymic and Mereological Part-Whole Relations.

This taxonomy further discriminates between part-whole relation types by enforcing semantic selectional restrictions, in the form of Dolce [120] ontology classes, on their instances. For example, the *member-of* meronymic relation can exist between instances

that belong to the class Physical Object (part) and Social Object (whole) in the Dolce ontology, such as the instance pair "player-team" in Table 6.1.

6.3 ADOPTED METHODOLOGY FOR INVESTIGATING EFFECT OF SEEDS

Our aim is to compare the relations extracted by a minimally-supervised IE algorithm when it is initialized with separate sets of seeds for each type of part-whole relation, and when it is initialized following the traditional practice of using a single (mixed) set of random seeds that belong to the different types.

To distinguish between types of part-whole relations, we will commit to the taxonomy of Keet and Artale (Keet's taxonomy) [114], which uses sound ontological formalisms to unambiguously discriminate the relation types. We adopt a 3-step approach to address the questions we presented in Section 6.1. Our steps are listed below.

1. Define prototypical seeds (part-whole pairs) as follows:
    a. (Separate) sets of seeds for each type of part-whole relation in Keet's taxonomy.
    b. A single set that mixes seeds denoting all the different part-whole relations types (i.e. the traditional practice adopted by current minimally-supervised techniques).

2. Extract part-whole relations from a corpus by initializing a minimally-supervised IE algorithm with the seed-sets (in Step 1a and 1b).

3. Evaluate the harvested relations to determine performance gain/loss, types of part-whole relations extracted, and distinct patterns and pairs discovered.

The corpora and IE algorithm we used, and the seed sets construction are described below. Results are will be presented in the subsequent section.

6.3.1 CORPORA

We used the English and Dutch Wikipedia texts since their broad-coverage and size ensures that they include sufficient lexical realizations of the different types of part-whole relations. Our motivation for relying on Wikipedia was discussed in details in Sections 4.5.2 and 4.6.3.

The size of the English corpus was approximately of 500M words, while that of the Dutch collection was of around 110 M words. We parsed the English and Dutch corpora respectively with the Stanford [13] and the Alpino [157] parsers, and formalized the relations between terms (instances) as dependency paths. A dependency path is the shortest path of lexico-syntactic elements, i.e. shortest lexico-syntactic pattern, connecting instances (proper and common nouns) in their parse-trees, as illustrated in Section 4.5.3. Stevenson and Greenwood [135] showed that such a formalization was beneficial to IE tasks. Our previous experiments, reported in Chapters 4 and 5, also illustrated the strengths of lexico-syntactic patterns compared to the more traditional surface-strings employed by most existing minimally-supervised algorithms. In our representation, we substitute instances in the lexico-syntactic patterns with generic placeholders, viz. PART and WHOLE. Below, we show two lexico-syntactic patterns (1-b) and (2-b), respectively derived from the English and Dutch Wikipedia sentences (1-a) and (2-a). They denote the relations between "sample-song" and "alkaloide-plant".

1.
    a. The song "Mao Tse Tung Said" by Alabama 3 contains samples of a speech by Jim Jones
    b. WHOLE+nsubj < contain > dobj+PART

2.

a. Alle delen van de planten bevatten alkaloiden en zijn daarmee giftig (*All parts of the plants contain alkaloids and therefore are poisonous*)
b. WHOLE+obj1+van+mod+deel+su < bevat > obj1+PART

In our experiments, we discarded all the instance pairs and patterns that occurred less than 10 times in the English corpus, and less than 5 times in the Dutch corpus. Statistics on the number of pairs and patterns preserved after applying the frequency cut-off are given in Table 6.2 (all figures are in millions).

	English	**Dutch**
Instance pairs	328.0	28.8
Distinct instance pairs	6.8	1.4
Patterns	238.0	54.0
Distinct patterns	2.2	0.9

Table 6.2: Statistics on Pairs and Patterns for English and Dutch Wikipedia.

6.3.2 INFORMATION EXTRACTION (IE) ALGORITHM

As indicated earlier, in our experiments, we will employ a basic version of our minimally-supervised IE algorithm, presented in Chapters 4 and 5 for part-whole and causal relation mining. This version is devoid of any knowledge-base accesses, and does not incorporate any measures to mitigate the issue of semantic-drift.

The algorithm is initialized with a set of seeds (instance pairs), and extracts the patterns connecting these seeds in a corpus. The reliability of a pattern, *p*, *r(p)*, given a set of instance pairs, I, is computed using equation (6.1) as its average strength of association with each pair, *i*, weighted by the each pair's reliability, *r(i)*.

$$r(p) = \frac{\sum_{i \in I} \frac{pmi(i,p)}{max_{pmi}} \times r(i)}{|I|} \tag{6.1}$$

In this equation, *pmi(i, p)* is the pointwise mutual information (PMI) score [69] between a pattern, *p* (e.g. *consist-of*), and a pair *i* (e.g. "engine-car"). The reliability of the initializing seeds is set to 1. The top-*k* most reliable patterns are selected to find new pairs. The reliability of each pair *i*, *r(i)* is computed according to equation (6.2), where *P* is the set of harvested patterns. The top-*m* most reliable pairs are then used to infer new patterns.

$$r(i) = \frac{\sum_{p \in P} \frac{pmi(i,p)}{max_{pmi}} \times r(p)}{|P|} \tag{6.2}$$

The recursive discovery of patterns from pairs and vice-versa is repeated until a suitable number of patterns and/or pairs have been extracted.

6.3.3 SEED SELECTION

Initially, we selected seeds from WordNet [19] (for English) and EuroWordNet [72] (for Dutch) to initialize our algorithm. However, we found that these pairs, such as "acinos-mother of thyme" or "radarscherm- radarapparatuur" ("radar screen-radar equipment"), hardly co-occurred with reasonable frequency in Wikipedia sentences, hindering pattern extraction. We therefore adopted the following strategy.

We searched our corpora for archetypal patterns, e.g. "*contain"*, which characterize all the different types of part-whole relations. The instance pairs sub-categorized by these patterns in the English texts were automatically typed to their appropriate Dolce ontology classes, corresponding to those employed by Keet and Artale [114] for constraining the

instances that can participate in different part-whole relations types. This was achieved using the Java-OWL API[22] and the Dolce[23] ontology version 0.72 in OWL.

The types of part-whole relations instantiated by the pairs could then be determined based on their Dolce ontology classes. Separate sets of 20 pairs, with each set corresponding to a specific relation type in Keet's taxonomy, were then created. For example, the English Wikipedia tuple *t1* ="actor-cast" was used as a seed for the *member-of* part-whole relation since both its elements ("actor" and "cast") were typed to the Social Object class of the Dolce ontology, and according to Keet's taxonomy, they instantiated a *member-of* relation. Seeds for extracting relations from the Dutch corpus were defined in a similar way, except that we manually determined their ontological classes based on the class glossary of Dolce.

Below, we only report on the *member-of* and *sub-quantity-of* meronymic relations, and on the *located-in*, *contained-in* and *structural part-of* mereological relations. We were unable to find sufficient seeds for the *constituted-of* meronymic relation (e.g. "clay-statue") in order to create a corresponding set of 20 seeds. Also, we did not experiment with the *participates-in* and *involved-in* relations since their lexical realizations in our corpora are sparse, and they contain at least one verbal argument, whereas we only targeted patterns connecting nominals. Sample seeds, their corpus frequency, and the part-whole relation type they instantiate from the English (EN) and Dutch (NL) corpora are illustrated in Table 6.3.

Seed set	**Relation type**	**Lang.**	**Part-Whole Pair**	**Freq**
Set_1	*Contained-in*	EN	grave-church	155
		NL	beeld-kerk (statue-church)	120
Set_2	*Located-in*	EN	city-region	3735
		NL	abdij-gemeente (abbey-community)	36
Set_3	*Member-of*	EN	actor-cast	432
		NL	club-voetbal_bond (club-soccer union)	178
Set_4	*Structural part-of*	EN	engine-car	3509
		NL	geheugen-computer (memory-computer)	14
Set_5	*Sub-quantity of*	EN	alcohol wine	260
		NL	alcohol bier (alcohol-beer)	28

Table 6.3: Sample seeds for learning part-whole relations.

Besides the five specialized sets of 20 prototypical seeds for the relations in Table 6.3, we also adopted the traditional approach of defining a *general* set of mixed seeds, which combines four instance pairs from each of the specialized sets.

[22] http://protege.stanford.edu/plugins/owl/api
[23] http://www. loa-cnr.it/DOLCE.html/

6.4 EXPERIMENTS AND EVALUATION

We initialized our IE algorithm with the seed sets to extract part-whole relations from our corpora.

In its first iteration, our algorithm extracted the 10 most reliable patterns that were found to connect the initializing seeds. These 10 patterns were bootstrapped to induce 100 part-whole pairs. In each subsequent iteration, we then learnt 1 additional patterns, i.e. $k=|P|+1$, and 100 additional instance pairs, i.e. $m=|I|+100$, where $|P|$ and $|I|$ are respectively the number of patterns and pairs harvested in a previous iteration. We evaluated our results after 5 iterations since the performance in later iterations was almost constant. The results are discussed next.

6.4.1 PRECISION OF EXTRACTED RESULTS

Two human judges manually evaluated the instance pairs extracted from the English and Dutch corpora per seed set. Pairs that unambiguously instantiated part-whole relations were considered *true_positive*. Those that did not were considered *false_positive*. Ambiguous pairs were discarded. The precision of the pairs discovered by the different seed-sets in the last (i.e. 5th) iteration of our algorithm are in Table 6.4.

	Meronymic Seed Sets		**Mereological Seed Sets**			**General (mixed) seeds**
	Member-of	*Sub-quantity-of*	*Contained-in*	*Structural part-of*	*Located-in*	
EN	0.67	0.74	0.70	0.82	0.75	0.80
NL	0.68	0.60	0.60	0.60	0.70	0.71

Table 6.4: Precision for seed-sets representing specific part-whole relation types and for the general set composed of mixed (all) types.

These results reveal that the precision of the harvested pairs varies depending on the part-whole relation type that the initializing seeds denote. Mereological seeds *(contained-in, structural part-of, and located-in* sets) outperformed their meronymic counterparts (*member-of*, *sub-quantify-of*) in extracting relations with higher precision from the English texts. This could be attributed to their formal ontological grounding, making them less ambiguous than the linguistically-motivated meronymic relations [114].The precision variations were less discernible for pairs extracted from the Dutch corpus, although the best precision was still achieved with the mereological *located-in* seeds. We also noticed that the precision of pairs extracted from both the English and Dutch corpora by the general set of mixed seeds was as high as the maximum precision obtained by the individual sets of specialized seeds, that is, 0.80 (general seeds) vs. 0.82 (*structural part-of* seeds) for English, and 0.71 (general seeds) vs. 0.70 (*located-in* seeds) for Dutch. Based on these findings, we address our first question, stated in Section 6.1, and conclude that:

1. The type of relation instantiated by the initializing seeds affects the performance of minimally-supervised IE algorithms, with mereological seeds being in general more fertile than their meronymic counterparts, and generating higher precision pairs.
2. The precision achieved when initializing IE algorithms with a general set, which mixes seeds of heterogeneous part-whole relation types, is comparable to the best results obtained with individual sets of specialized seeds denoting specific part-whole relations.

6.4.2 TYPES OF EXTRACTED RELATIONS

We found out that initializing our algorithm with seeds of a particular type always led to the discovery of pairs characterizing the other part-whole types in the English corpus. This can be explained by prototypical part-whole patterns, like *"include"*, which are generated regardless of the seeds' types. These patterns are highly correlated with, and hence, extract the pairs denoting other part-whole relation types.

An almost similar observation was made for the Dutch corpus, except that pairs instantiating the *member-of* relation could only be learnt using initial seeds of that particular type (i.e. *member-of*). Upon inspecting our results, it was found that this phenomenon was due to the specific patterns, such as *"treedt toe tot"* (*"become member of"*), which linguistically realize the *member-of* relations in the Dutch corpus. Thus, our IE algorithm fails to detect these specific patterns, and fails to subsequently discover part-whole pairs describing the *member-of* relation when it is initialized with seeds that instantiate relations other than *member-of*.

Our findings are illustrated in Tables 6.5 (English) and 6.6 (Dutch). Each cell shows an instance pair of a particular type (column) that was harvested from seeds of a given type (row). These results answer our second question, and we conclude that the patterns and pairs discovered from seeds of a particular type are not confined to that type.

	Extracted Tuples					
		Meronymic		**Mereological**		
Type of initial seeds (EN)		*Member-of*	*Sub-quantity-of*	*Contained-in*	*Structural part of*	*Located-in*
	Member-of	Ship-convoy	alcohol-wine	card-deck	proton-nucleus	lake-park
	Sub-quantify-of	Aircraft-fleet	moisture-soil	building-complex	engine-car	commune-canton
	Contained-in	Aircraft-fleet	alcohol-wine	relic-church	base-spacecraft	campus-city
	Structural part of	Brother-family	mineral-bone	library-building	inlay-fingerboard	hamlet-town
	Located-in	Performer-cast	alcohol-blood	artifact-museum	chassis-car	city-shore

Table 6.5: Sample tuples extracted per relation type for English.

	Extracted Tuples					
		Meronymic		**Mereological**		
Type of initial seeds (NL)		*Member-of*	*Sub-quantity-of*	*Contained-in*	*Structural part of*	*Located-in*
	Member-of	sporter-ploeg (athlete-team)	helium-atmosfeer (helium-atmosphere)	stalagmieten-grot (stalagnites-cave)	shirt-tenue (shirt-outfit)	boerderij-dorp (farm-village)
	Sub-quantify-of	—	vet-kaas (fat-cheese)	Pijp_orgel-kerk (pipe-organ-church)	kam-gitaar (bridge-guitar)	paleis-stad (palace-city)
	Contained-in	—	tannine-wijn (tannine-wine	kamer-toren (room-tower)	atoom-molecule (atom-molecule)	paleis-stad (palace-city)
	Structural part of	—	kinine-tonic (quinine-tonic)	beeld-kerk statue-church)	wervel-ruggengraat (vertebra-backbone)	paleis-stad (palace-city)
	Loacated-in	—	—	kunst_werk-kathedraal (work of art-cathedral)	poort-muur (gate-wall)	metro station-wijk(metrostation-quarter)

Table 6.6: Sample tuples extracted per relation type for Dutch.

6.4.3 DISTINCT PATTERNS AND TUPLES

We will now address our third question of whether a wider variety of patterns/pairs are extracted when our initializing seeds are of a particular type. This is achieved by comparing the outputs of our algorithm to determine whether the results obtained when initializing with the individual (specialized) seed sets were (dis)similar and/or distinct. Then, we will compare these results with those obtained by using a general set of mixed seeds.

Each result set (extracted by our algorithm) that we examined consisted of maximally 520 pairs (including 20 initializing seeds) and 15 lexico-syntactic patterns, obtained after five iterations. We observed that the pairs extracted from the English corpus using the *member-of* and *contained-in* seed sets exhibited a high degree of similarity, with 465 common pairs discovered by both sets. These identical pairs were also assigned the same ranks (reliability) in the results generated by the *member-of* and *contained-in* seeds, with a Spearman rank correlation of 0.82 between their respective outputs. This convergence was also reflected in the fact that the *member-of* and *contained-in* seeds generated around 80% of common pat- terns. These patterns were mostly prototypical ones, indicative of part-whole relations, such as WHOLE+nsubj < include > dobj+PART (*"include"*). These specialized seeds also generated distinct patterns, like *"joined as"* and *"released with"* for the *member-of* and *contained-in* seeds respectively.

The most distinct pairs and patterns were harvested with the *sub-quantity-of*, *structural part-of*, and *located-in* seeds. Negative Spearman correlation scores were obtained when comparing the results of these three sets among themselves, and with the results of the *member-of* and *contained-in* seeds, indicating insignificant similarity and overlap. Examining the patterns harvested by the *sub-quantity-of*, *structural part-of*, and *located-in* seeds revealed a high prominence of specialized patterns, which specifically characterize these relations. Examples of such patterns are *"made with"*, *"released with"* and *"found in"*, which lexically realize the *sub-quantity-of*, *structural part-of* and *located-in* relations respectively.

For the Dutch corpus, the seeds that generated the most similar pairs were those corresponding to the *sub-quantity-of*, *contained-in*, and *structural part-of* relations, with 490 common pairs discovered, and a Spearman rank correlation in the range of 0.89-0.93 between their respective outputs. As expected, these seeds also led to the discovery of a substantial number of common and prototypical part-whole patterns. Examples are *"bevat" ("contain")* and *"omvat" ("comprise")*. The most distinct results were harvested by the *located-in* and *member-of* seeds, with negative Spearman correlation scores between the output pairs, indicating hardly any overlap. We also found out that the patterns harvested by the *located-in* and *member-of* seeds characteristically pertained to these relations. Example of such patterns include *"ligt in" ("lie in")*, *"is gele- gen in" ("is located in")*, and *"treedt toe tot" ("become member of")*, respectively describing the *located-in* and *member-of* relations.

Thus, we observed that

1. Pairs harvested from both the English and Dutch corpora by seeds instantiating a particular type of part-whole relation highly correlated with pairs discovered by at least one other type of seeds (*member-of* and *contained-in* for English, and *sub-quantity- of*, *contained-in* and *structural part-of* for Dutch).
2. Some part-whole relations are manifested by a wide variety of specialized patterns (*sub-quantity- of*, *structural part-of*, and *located-in* for English, and *located-in* and *member-of* for Dutch).

Finally, instead of a single set that mixes seeds of different types, we created five such general sets by picking four different seeds from each of the specialized sets, and used them to initialize our algorithm. When examining the results of each of the five general sets, we found out that they were unstable, and always correlated with the output of a different specialized set.

Based on these findings, we believe that the traditional practice of initializing minimally-supervised IE algorithms with general sets that mix seeds denoting different part-whole relation types leads to inherently unstable results. As we have seen, the relations extracted by combining seeds of heterogeneous types almost always converge to one specific part-whole relation type, which cannot be conclusively predicted. Furthermore,

general seeds are unable to capture the specific patterns that lexically realize the individual types of part-whole relations, answering our third research question.

6.5 CONCLUSION

In this chapter, we have investigated the effect of seeds selected on the performance of minimally-supervised IE algorithms for extracting part-whole relations.

Our results illustrate that the outputs of IE algorithms are heavily influenced by the initializing seeds. Below, we summarize our main findings in response to the 3 main questions that we formulated at the beginning of this chapter.

1. In most cases, mereological seeds tend to yield more accurate results that meronymic ones. In addition, we showed that learning from specialized seeds-sets, denoting specific types of part-whole relations, results in precision that is as high as or higher than the precision achieved with a general set that mixes seeds of different types.

2. By comparing the outputs generated by different seed-sets, we observed that the pairs learnt with seeds denoting a specific part whole relation type are not confined to that particular type. In most case, we are still able to discover pairs across all the different types of part-whole relations, regardless of the type instantiated by the initializing seeds.

3. Given a set of mixed seeds, denoting heterogeneous relations, the harvested pairs may converge towards any of the relations instantiated by the seeds. Predicting the convergent relation is in usual cases impossible, and may depend on factors pertaining to corpus characteristics. This instability strongly suggests that seeds instantiating different types of relations should not be mixed. Furthermore, we have seen that a general set of mixed seeds may fail to capture several of the specialized patterns that realize the individual part-whole relations.

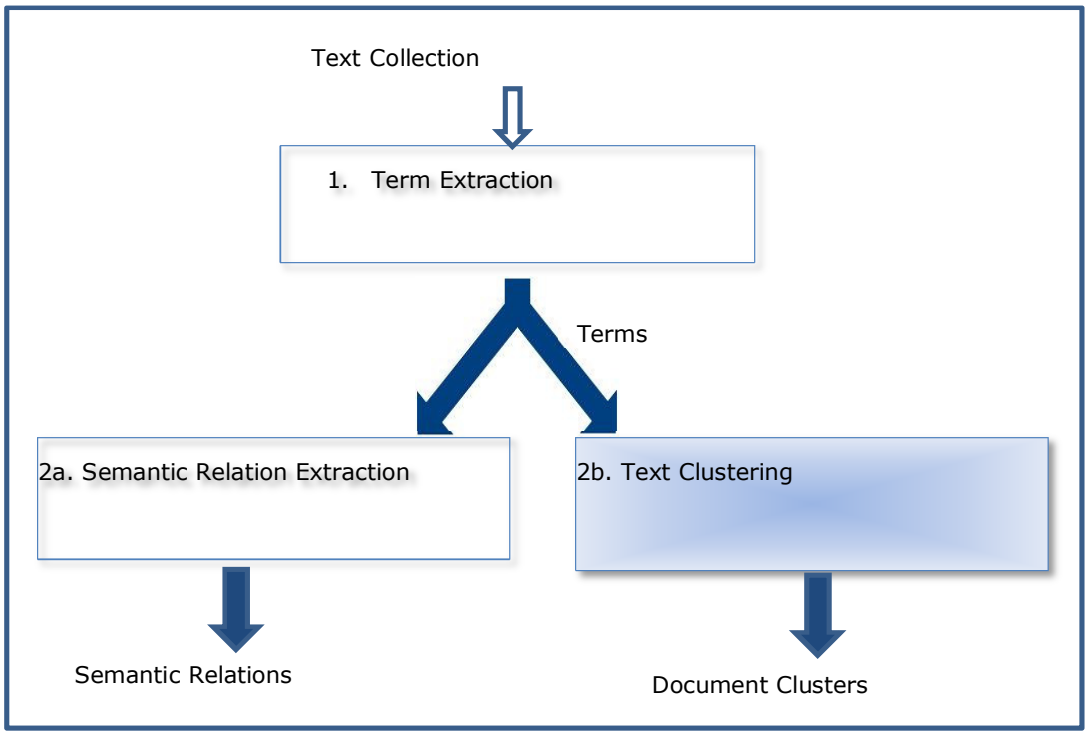
In Chapter 8, we will discuss further implications of these findings on our part-whole relation extraction approach, which was presented earlier in Chapter 4.

PART IV - TEXT CLUSTERING

PREAMBLE

In Part IV of the thesis, we will address the third research question *RQ3: How to discover cohesive and homogeneous clusters of similar documents from a collection of domain-specific, sparse and informally-written corporate texts?*

As mentioned earlier, this corresponds to the task 2b in our application pipeline, shown below, which we need to execute for realizing our overall aim of extracting meaningful information from corporate texts. Our proposed solution in response to RQ3 will be presented in Chapter 7.



CHAPTER 7 –TEXT CLUSTERING

CHAPTER SUMMARY

In this chapter, to address our third research question, we will present a novel text clustering algorithm, ClustText. The fundamental idea in ClustText is to discover sets of frequent features (e.g. words) that occur frequently across the documents of a corpus. These are known as wordsets. The main benefit with wordsets is that each of them corresponds to the description of a topic in the corpus. Thus, they serve as natural partitioning to cluster the documents. We treat each wordset as a label describing the topic of an (initially empty) cluster. In this way, we are able to address the issue of cluster labeling that plagues traditional clustering techniques like K-means.

The discovery of meaningful cluster labels (wordsets) entails the identification of the most informative features from our documents. As will be seen in our experiments, traditional feature selection methods are inadequate to identify features from our corpus of informally-written and sparse texts. This results in the formation of incoherent/invalid cluster labels, which are impractical for real-world applications. To overcome this challenge, we propose a novel technique for feature identification. Our technique is a hybrid one, incorporating both a wrapper and a filter method. We will see that such a hybrid configuration enables us to achieve the high accuracy levels of wrapper methods, without suffering from their high computational costs. After forming the labeled clusters, they are populated with the appropriate documents. Our cluster population strategy assigns a document to a cluster only if such an assignment maximizes the intra-cluster-similarity and minimizes the inter-cluster similarity. Such a strategy enables us to discover high quality clusters of similar documents, with minimal overlap. To evaluate our clustering algorithm, we will estimate its performance on a real-life corpus of domain-specific texts. Our experiments results reveal that our approach accurately discovers clusters with meaningful labels from our corpus of domain-specific texts, and outperforms a state-of-the-art text clustering algorithm.

- The work described in this chapter is an extended version of a pre-print to be submitted to the journal *"Expert Systems with Applications"*
- An earlier version appeared as "Ittoo, A.; Maruster, L. Ensemble Similarity Measures for Clustering Terms", in Proceedings of World Congress in Computer Science and Information Engineering, vol. 4, pp. 315-319, IEEE Computer Society

7.1 INTRODUCTION

Besides the detection of terms and of semantic relations, discussed in the preceding chapters, our objective of extracting meaningful information from corporate texts also entails document (text) clustering.

Clustering is a data-mining technique, which aims at identifying distinct, non-overlapping sub-groups of similar data objects. Each sub-group thus induced is known as a cluster. Clustering techniques have been employed in a wide variety of domains, including engineering (e.g. pattern recognition), bio-medicine (e.g. genetics) social sciences (e.g. sociology) [158, 159] as well as marketing research [160].

Clustering is one of most popular data-mining techniques for supporting corporate activities, and is at the heart of Customer Relationship Management (CRM) applications. It has traditionally been used in market segmentation efforts [161, 162] , for example, to discover sub-groups of customers with similar purchase frequencies or usage patterns of a particular service. The identified clusters of similar customers are then exploited to determine the customers' preferences and to recommend new products/services to them [163, 164, 165, 166]. In our domain of interest, namely Product Development – Customer Service (PD-CS), the clustering of similar customer complaint reports enables business organizations to acquire more insights on the needs or wishes from their market-base. This information can then be used to enhance customer satisfaction. Also, clusters of engineers' repair notes, which mention similar repair actions performed on malfunctioning products, provide valuable information on the causes of product failures. With these clusters, PD-CS organizations are able to improve the quality of products and of after-sales services. The clusters can also be used to investigate whether the engineers are adhering to stipulated repair/servicing guidelines.

Traditional clustering algorithms, such as *K-means*, have been originally designed to operate on data in structured formats. They exhibit various limitations when applied to unstructured text data as reported by numerous studies such as [167, 168, 169]. For example, the clusters discovered by these techniques are unlabeled, and have to be manually inspected to determine their contents, which is time-consuming and tedious especially when dealing with text [167, 168, 169]. Also, traditional techniques estimate document similarity by comparing high dimensional but sparsely populated vectors, which is detrimental to their scalability for large datasets. Furthermore, manually defining the (number of) initial clusters, which is required by some techniques like K-means prior to the actual clustering, is particularly difficult with unstructured texts.

Recent research efforts have led to the development of a new breed of clustering algorithms specifically targeted at unstructured text data. They identify (groups of) similar documents based on sets of frequent words (features) or *wordsets*. Intuitively, documents that share a larger number of wordsets exhibit a greater degree of similarity, and are assigned to the same cluster. We will refer to these techniques as wordset-based clustering algorithms. Wordset-based clustering algorithms have been shown to achieve higher clustering accuracy over text data than their traditional counterparts [167, 168, 169, 170]. More importantly, they overcome the shortcomings exhibited by traditional techniques for text clustering. For example, they use the wordsets as labels to describe the contents of each cluster. Also, they determine document similarity by comparing low dimensional wordsets instead of high dimensional and sparse document vectors, which is beneficial to their efficiency and scalability. Furthermore, wordset-based algorithms eschew the need to define initial clusters prior to the actual clustering [167, 168].

Wordset-based algorithms developed to date have been predominantly applied to large collections of well-written texts. As already discussed, such standard corpora provide reliable linguistic and statistical evidence (features). These features can be readily acquired from the documents' contents, for example, based on frequency counts, and facilitate the discovery of meaningful cluster labels and the formation of accurate clusters.

However, extant wordset-based algorithms have overlooked the challenges involved in clustering documents generated in corporate disciplines like PD-CS. The main difficulty in clustering these domain-specific documents is that they are informally-written and sparse

(e.g. they are short and contain relatively few words). Hence, they do not explicitly provide reliable features that can be readily acquired from their contents, based on, for example, frequency counts. Instead, sophisticated mechanisms are often needed to extract the desired features. We will refer to this challenge as Challenge 1. This difficulty in acquiring pertinent features from domain-specific texts is detrimental to the performance of wordset-based algorithms for two main reasons. First, it impedes the discovery of valid wordsets that can be used as meaningful cluster labels. Second, it also makes it difficult to precisely discriminate between groups of similar documents, which causes the clustering accuracy to degrade.

We will now provide a summary of other shortcomings of wordset-based algorithms. For the sake of completeness, we will briefly repeat Challenge 1.

1. (Informally-written, sparse texts) Inadequate feature identification. Existing algorithms have overlooked the difficulties in acquiring reliable features from informally-written and sparse domain-specific texts, which affects their performance in discovering accurate clusters with meaningful labels (Challenge 1).

2. Cluster conflicts. These algorithms often fail to precisely select the single, most appropriate cluster for a document [170] (Challenge 2).

3. Skewed clusters. The clusters discovered by these algorithms tend be skewed, with very large or very small number of documents per cluster, which is impractical for real-world usage [170] (Challenge 3).

Once again, it should be noted that the aforementioned challenges are general limitations of wordset-based clustering algorithms, which are more acute when dealing with informally-written and sparse texts. Therefore, our solutions to overcome these limitations, which will be discussed next, are relevant to the field of clustering at large.

In this chapter, we answer our third research question by developing the ClustText framework for text clustering. ClustText is a wordset-based algorithm at its core. It incorporates various innovative aspects that enable it to successfully alleviate the limitations of existing algorithms, especially when dealing with informally-written and sparse texts generated in specialized domains like PD-CS. Our core contributions with ClustText are novel techniques for:

1. Identifying pertinent features from (domain-specific) texts, which facilitate the discovery of precise clusters with meaningful labels (Contribution 1 in response to Challenge 1).

2. Resolving cluster conflicts by precisely assigning documents to their single, most appropriate clusters (Contribution 2).

3. Inhibiting the formation of skewed clusters with extremely disparate sizes, such as a very large cluster that contains most documents in a corpus or many small clusters with few (even 1) document (Challenge 3).

At this juncture, two points are worth mentioning.

- The novel techniques that we propose in this chapter, should, in principle, be applicable to other types of texts, such as standard corpora like newspaper articles. However, in our experiments, we will restrict ourselves to domain-specific (corporate) documents since they have been largely overlooked by previous research efforts, and also given the context of this thesis.
- ClustText, like other wordset-based algorithms, overcomes the limitations of traditional clustering techniques like K-means. It discovers labeled clusters, estimates documents similarity by comparing low dimensional wordsets to improve its scalability, and alleviates the need for specifying initial clusters.

We will elaborate on these challenges and on our contributions in Sections 7.3 and 7.4 respectively.

To validate our ClustText framework, we evaluated its performance in discovering document clusters from a real-life corpus provided by our industrial partners. We compared ClustText's performance against a state-of-the-art text clustering algorithm, which we used as baseline. Our experimental results suggest that ClustText outperformed the baseline, and accurately discovered document clusters, with meaningful labels, from our corpus. These results indicate that ClustText not only overcomes the longstanding (general) challenges in text clustering, but it also addresses the intricacies involved in accurately clustering domain-specific, informally-written and sparse documents. Also, the clusters of similar document discovered by ClustText contain useful information on product failures that required similar repairs, and, can be exploited in corporate activities like determining the cost of non-quality.

This chapter is organized as follows. In Section 7.2, we review some basic notions of clustering, and the traditional difficulties involves in clustering unstructured text data. We will also introduce wordset-based algorithms. Section 7.3 shows that despite their strengths for text clustering, wordset-based algorithms still exhibit various shortcomings. To address these limitations, we present our contributions with our proposed ClustText framework in Section 7.4. The framework is described in details and Section 7.5, and its performance is empirically evaluated in Section 7.6.

7.2 Related Work

7.2.1 Clustering: Unsupervised Learning

Clustering is a data-mining technique, which aims at automatically classifying (categorizing) data-objects into their natural classes. Each natural class corresponds to a *cluster*. There is no agreed-upon definition as to what exactly constitutes a cluster. The majority of studies consider a cluster as a sub-group of homogeneous objects that are maximally similar to each other, and minimally similar (i.e. dissimilar) with the objects of other sub-groups (clusters) [27]. Therefore, an object is assigned to a cluster only if it maximizes the intra-cluster similarity, i.e. between the objects of the cluster, and minimizes the inter-cluster similarity, i.e. the overlaps between different clusters [171, 172].

Clustering is often referred to as *unsupervised learning*. This is because the clusters (natural classes) for the objects are not predefined and are usually unknown prior to the actual clustering. It is up to the clustering algorithm to discover them. Such a learning paradigm contrasts with the *supervised learning* approach adopted by other data-mining techniques, such as classification. In classification, for example, the data objects are already pre-classified into their respective categories. Classification models are then learned from the pre-classified (training) dataset, and used to predict the categories of new objects. In this thesis, we will use clusters and classes interchangeably, wherever more appropriate.

As with most other data-mining techniques, the identification of relevant features is an essential step to improve the accuracy of clustering techniques. Feature identification is discussed in the next section.

7.2.2 Feature Identification

The identification of relevant features has long being considered as one of the most fundamental problem in data-mining [173]. In its simplest form, a relevant feature corresponds to a salient attribute that characterizes the objects in a dataset. In clustering applications, for instance, a (relevant) feature is a discriminative attribute, which enables us to precisely distinguish between different data objects and to assign them to their respective clusters. When dealing with unstructured texts, terms and other important words are often used as features.

Features: A prerequisite for clustering accuracy and meaningful cluster labels
Several studies have shown that reliable features can significantly improve the clustering accuracy. Conversely, invalid features have an adverse effect on the accuracy by blurring the distinction between the clusters [174]. Identifying relevant features is particularly important in wordset-based clustering algorithms (Section7.2.6) since the wordsets that

these algorithms discover from a corpus and that they later use as labels for describing the clusters' contents are composed of features. Thus, in these algorithms, besides ensuring the clusters' accuracy, the detection of pertinent features is also crucial for generating meaningful cluster labels. The need for reliable features is even more pressing when dealing with sparse and informally-written texts. Improperly selected features may lead to irrelevant or incoherent cluster labels, such as "meeting, Friday, room" or "unknown, training, contact", which do not provide any useful information on the clusters' contents. They are misleading to users, and are unsuitable for labeling clusters in real-world applications. Furthermore, users often decide whether a cluster is worth inspecting based on its label. Thus, an incorrectly labeled cluster will be overlooked (skipped) by users even if it contains the documents of interest that they are searching for. For these reasons, we consider the identification of relevant features and the rejection of invalid ones as an indispensable step for discovering appropriately-labeled and accurate clusters.

Next, we will briefly introduce two mechanisms, viz. *feature extraction* and *feature selection*, which are commonly employed for identifying features in data-mining applications. For a more thorough description of these techniques, we refer the reader to [173, 175, 176].

Feature Extraction
Feature extraction has traditionally been employed in unsupervised learning tasks, such as clustering. It transforms the original features into a new set of discriminative features via some functional mappings. A well-known example of such a mapping is the Principal Component Analysis (PCA) [177], which extracts a set of pertinent features (i.e. principal components). These features can then be used by a clustering algorithm to partitioning of the data into clusters [174, 176].

The main drawback of feature extraction is that the transformed features bear little or no resemblance with the original ones [174, 176]. The clusters discovered by such features can be hard to understand, which makes them impractical for real-world corporate usage.

Feature Selection
Feature selection has been more commonly applied in supervised learning tasks, such as classification [174]. Unlike feature extraction, it does not generate a new set of transformed features from the original ones. Instead, it relies on an *evaluation (criterion) function* to select a subset of the most relevant features [173, 178]. The identified features are therefore easier to interpret, and provide more meaningful results than those obtained by feature extraction. Several evaluation functions for feature selection have been proposed. According to Langley [179], they are usually of two types namely, *filter* and *wrapper* methods.

Filter methods adopt a feature selection mechanism that is independent of the classification (or other supervised data-mining) application that will use the identified features. They compute the relevancy of a feature by estimating its dependency with the categories (available in supervised learning) in the data. The higher the dependency of a feature with particular category, the more discriminative and relevant the feature is. Such features will be assigned a relatively high dependency score, and will be selected if their scores satisfy a threshold. A well-known filter method is the *Information Gain (IG)* measure of Yang and Pedersen [175]. It calculates the relevancy of an attribute (in a dataset) as the entropy (e.g. uncertainty) in predicting the category when the attribute is present and when it is absent. If the attribute's removal results in a higher entropy (i.e. the uncertainty in predicting the category is higher), then the attribute can be considered as relevant feature. Another popular filter technique for feature selection, especially from text data, is *Document Frequency (df)* thresholding [175, 176]and its variant *Term Frequency – Inverse Document Frequency (t f-idf)* [61, 180], which we introduced in Chapter 2. The main advantage of these frequency-based techniques is that they are relatively simple, and scale well for large corpora [175, 180]. Since frequency-based techniques are independent of the categories in a corpus, they have often been employed for feature identification in unsupervised learning tasks such as clustering. As will be seen in Section 7.2.6, most of the existing wordset-based algorithms rely on the tf-idf metric to identify features. The main drawback of filter methods, such as the aforementioned IG, df and tf-idf , is in determining an optimal threshold in order to select the most reliable set of relevant features based on their scores. Choosing such a threshold is challenging, and is

often not based on any theoretical grounding. Instead, the threshold value is dictated by the application or the experience of end-users [181].

Conversely to filter methods, the feature selection mechanism of wrapper methods is tightly coupled with (wrapped around) the intended data-mining application. Since these methods are defined for supervised learning tasks, in particular, classification, they use the accuracy of the classifier as an evaluation function. For example, the set of attributes that maximizes the classifier's F1-score is selected [173, 178]. The main strength of wrapper methods is that they are highly accurate as they use the actual classifier performance as an evaluation function. However, they are also computationally expensive, and have to search through the entire subset space of all attributes in a dataset (i.e. all combinations of attributes) in order to find the best set that achieves highest accuracy [178]. Some recent studies have also investigated the use of wrapper methods for unsupervised learning applications as discussed in the work of Roth and Lange [182] .

7.2.3 VECTOR SPACE MODEL, SIMILARITY MEASURES AND CENTROID

A fundamental notion in clustering algorithms is the *Vector Space Model (VSM)* [168, 176], which we briefly introduced when discussing the scalability issue in Section 7.2.1. The VSM encodes a data object as a vector in a multi-dimensional space. For example, a database record, x_i, in a dataset X with a total of p (relevant) features, will be represented as a p-dimensional vector. The vector's j^{th} element, x_{ij} , gives the value of the j^{th} feature in the record x_i. In text clustering, a document, x_i, in a corpus with a total of p features (terms or important words), will also be represented by a p-dimensional vector. The vector's j^{th} element, x_{ij}, then gives the weight of the j^{th} feature, such as its frequency, in document x_i.

Several measures have been employed for computing the similarity(distance) between data objects. These include the Minkowski distance, the Euclidean distance, the city-block (Manhattan) distance and the Pearson correlation measure. A complete review of these techniques is provided in the work of Xu and Wunsch [27]. According to previous studies reported in [27] , the most commonly used and accurate measure is the cosine similarity. This measure computes the similarity between two objects, x_1 and x_2, by considering the cosine of the angle between their corresponding vectors, $\overline{x_1}$ and $\overline{x_2}$, as depicted in equation (7.1).

$$similarity(\overline{x_1}, \overline{x_2}) = \frac{\sum_{f \in F} w_{f,x_1} \times w_{f,x_2}}{\sqrt{\sum_{f \in F} {w_{f,x_1}}^2} \times \sqrt{\sum_{f \in F} {w_{f,x_2}}^2}} \qquad (7.1)$$

In this equation, w_{f,x_i} is the value(weight) of feature f in the object x_i. The cosine value ranges from -1 to 1, indicating different degrees of similarity. A value of 1 indicates that the vectors point in the same direction, implying that the two objects being compared are identical (highly similar) to one another. On the other hand, a value of -1 indicates that the vectors point in opposite directions, implying that the two objects are dissimilar. After computing the similarity between objects, they are grouped into clusters. Objects that are found to be highly similar to each another are grouped together into a single cluster, while those that are dissimilar will be assigned to different clusters.

A cluster of similar data objects can thus be thought of as a group of vectors that are close to each other in a multi-dimensional space. The cluster *centroid* is then a vector that uniquely characterizes a cluster. It is calculated as the mean[24] value of the vectors in the cluster [28, 183]. The centroid is at the core of traditional clustering techniques, which will be described in the next section.

[24] The median vector has also been used as centroid, but the more common practice is to use the mean.

7.2.4 TRADITIONAL CLUSTERING ALGORITHMS

There are two major types of clustering algorithms, namely *hierarchical* and *partitional* [158, 184]. In this thesis, we will refer to them as *traditional* techniques to contrast with the more recent *wordset-based* algorithms, which will be discussed in Section 7.2.6.

Hierarchical Clustering

Hierarchical clustering techniques represent their clusters as a dendogram or a tree structure. In such a representation, the root node typically corresponds to the entire dataset, the internal nodes and edges depict the distance (similarity) between objects, while the leaf nodes are the actual objects.

Hierarchical algorithms can be further classified as *divisive* or *agglomerative*. Divisive techniques consider the entire dataset of N objects as an initial cluster. The cluster is then recursively split into small sub-clusters until some termination criterion is fulfilled, for instance, when singleton clusters are obtained. Agglomerative techniques, on the other hand, consider each object in a dataset as a distinct cluster. The two most similar clusters are then iteratively merged, until a termination criterion is fulfilled, for instance, when all the clusters have been merged. The most basic methods to determine the most similar clusters are the *single-linkage* and the *complete-linkage* [27]. In simple-linkage, the distance between the objects in the two clusters is first estimated using any of the aforementioned similarity measure, such as the Euclidean distance. The similarity between the two clusters is then given as the shortest distance between their (closest) objects. With complete-linkage, the similarity between two clusters is given by the largest distance between their (furthest) objects. A well-known agglomerative clustering technique is the *Unweighted Pair Group Method with Arithmetic Mean (UPGMA)* [172]. It calculates the similarity between two clusters as the average cosine similarity between the clusters' documents [28, 183].

Hierarchical clustering techniques have prohibitively high computational complexity. For instance, the time (run-time) and space (memory) complexity of most agglomerative algorithms when clustering a dataset of size N is of the order of $O(N^2)$, and is even higher for divisive techniques [27]. They are therefore impractical and not scalable for large datasets. It is worth mentioning, however, that recently developed hierarchical algorithms, such as the *Balanced Iterative Reducing and Clustering using Hierarchies* (BIRCH) [185], have significantly improved the clustering efficiency, with a linear time/space complexity in the order of $O(N)$.

Partitional Algorithms

Partitional algorithms are by far the most widely employed clustering techniques [27]. They partition a dataset of N objects into K distinct clusters. A prototypical partitional clustering technique is the *K-means* [186]. K-means operates upon the premise that a cluster can be described by its centroid, which we presented in Section 7.2.3. Initially, K vectors are selected at random or based on some domain knowledge. These K vectors are treated as centroids of K initial clusters. Next, the distance (similarity) between each of the remaining vectors (objects) in the dataset and the centroids is computed using standard measures, like the cosine similarity. An object is then assigned to a cluster if it is found to be most similar (nearest) to the centroid of that cluster. After populating the clusters, their centroids are re-computed, taking into account the newly added objects. K-means then iterates between the two latter phases. That is, for each object, it determines the nearest cluster, assigns it to that cluster and re-computes the centroid. These steps are repeated until there are no changes for each cluster, i.e. each object has been assigned to its nearest cluster.

The K-means algorithm is a relatively simple technique, which has been shown to achieve reasonably high clustering accuracy [183]. It is more efficient than the majority of hierarchical clustering algorithms, and has near linear computational complexity. Considering our dataset of N objects, the time complexity of K-means is of $O(NKdm)$, where m refers to the number of iterations required, and d is the size (dimensionality) of the vectors, which depends on the number of features in the dataset. The corresponding space complexity of K-means is simply $O(N+K)$ [27].

Several enhancements to the basic K-means have been proposed. The *bisecting K-means* algorithm, for instance, combines the strengths of partitional and hierarchical clustering

techniques. With bisecting K-means, the entire dataset is initially treated as a single cluster. It is then split (bisected) into two sub-clusters using the basic K-means. This bisecting step is repeated, until the desired number of clusters is obtained. Bisecting K-means has a linear time complexity of *O(N)*. It is considered as one of the state-of-the-art clustering techniques, and was shown to outperform the basic K-means and hierarchical techniques like UPGMA in generating the most accurate clusters [169, 183].

7.2.5 TRADITIONAL ALGORITHMS FOR TEXT CLUSTERING

Traditional clustering algorithms were originally designed to operate on structured data, consisting of numerical and/or categorical values such as sales frequencies and gender. In their study, Steinbach et al. [28, 183], investigated the performance of these techniques when applied for text clustering. Specifically, they compared the clustering accuracy achieved by partitional techniques, like K-means and bisecting K-means, and by hierarchical (agglomerative) techniques like UPGMA. According to their experiments, the most accurate (text) clustering technique was the bisecting K-means. It achieved an F1 (accuracy) score in the range of 0.59 to 0.89 over various collections of general texts, including the L.A. Times newspaper articles. The least accurate clusters were those discovered by the UPGMA. Its corresponding scores varied in the range of 0.56 to 0.86 [183].

Despite their moderate-reasonable accuracy, traditional techniques fail to meet the specific requirements that text clustering demand. We have already outlined the limitations of these techniques in Section 7.1, and we will now elaborate on them.

Unlabeled Clusters

A longstanding challenge in clustering research is that of cluster labeling. Traditional algorithms generate unlabeled clusters, which do not provide any information on their contents. In the absence of cluster labels, end-users have to manually inspect the clusters' contents in order to find out what they are about, which is time-consuming and tedious, especially with text data. Furthermore, after inspecting the clusters, there is the issue of manually or automatically formulating appropriate labels that accurately characterize their contents. Thus, the unlabeled clusters discovered by traditional algorithms cannot be efficiently exploited to support real-world corporate activities like BI.

Specifying Initial Clusters

Another issue with traditional techniques like K-means is the difficulty in selecting data objects as initial clusters (centroids), prior to the actual clustering. This difficulty is compounded when dealing with unstructured texts. There are no universal and efficient methods for identifying these initial clusters [27]. Very often, they are randomly selected. However, these initial clusters have a significant influence on the performance on clustering algorithms. In some cases, they may even prevent the algorithms from converging to a *global optimum*, in which case, the best set of clusters will not be discovered [27].

Scalability

In text clustering, traditional techniques represent every document as a vector, where each dimension corresponds to a distinct feature (term or important word) in the entire corpus. These document vectors are of high dimension as the total number of features in the whole corpus is usually large. For example, we had in excess of 1000 terms (features) extracted in Chapter 2. Each document in our corpus will therefore be encoded as a vector of at least 1000 dimensions. However, it is highly unlikely for any single document to contain all the features of the entire corpus. Instead, it is more probable that a document will contain only a small fraction of these features, especially if the documents are sparse. As a result, the (high dimensional) document vectors will be sparsely populated; each vector will contain only the few features that are present in its corresponding document.

Comparing these vectors in order to measure the similarity between documents is inefficient and affects the scalability – a phenomenon commonly referred to as the curse of dimensionality[25].

7.2.6 FREQUENT WORDSETS FOR TEXT CLUSTERING

Recently, a new class of clustering algorithms, based on sets of frequently co-occurring features (e.g. words, terms) or *wordsets*, has been proposed [167, 168, 169].

The basic idea underlying these algorithms can be distilled into 3 main steps. First, they select pertinent features (e.g. terms, words) from a corpus. Second, they identify sets of features that co-occur repeatedly across groups of documents in the corpus. These sets have been called termsets [169] or simply itemsets [167, 170, 187]. We will refer to them as wordsets since they are essentially composed of features, which are lexically realized as terms or words. Finally, in the third step, each of the extracted wordset is treated as the label of an initially empty cluster. Documents are assigned to clusters if, for example, they contain the clusters' labels (wordsets). This step can result in overlapping clusters, whereby a document may be assigned to more than 1 cluster [167, 169]. Some algorithms like the *Frequent Itemset-based Hierarchical Clustering (FIHC)* [167] have an additional stage in which they refine the previously populated clusters, such that each document is kept only in its best cluster, and removed from the others. However, as will be discussed in Section 7.3, and shown in our experimental evaluation in Section 7.6, this additional step often proves to be insufficient for effectively resolving the cases of conflicting clusters.

Wordset-based algorithms are particularly attractive for text clustering for several reasons. They have been shown to outperform traditional techniques like K-means, bisecting K-means and UPGMA in discovering more accurate clusters from standard corpora of newspaper articles [167, 168, 169, 170]. More importantly, they address the specific requirements that text clustering demand. For example, they use the wordsets as labels to describe the clusters' contents, and hence, overcome the issue of cluster labeling that plagues traditional techniques like K-means. Also, these algorithms mitigate the curse of dimensionality that affects traditional techniques since they compute document similarity by comparing low dimensional wordsets instead of high dimensional and sparse document vectors. We will further elaborate on these desiderata of wordset-based algorithms when presenting the key characteristics of our ClustText framework in Section 7.4.

Next, we will briefly describe the notion of wordsets, which are at the core of wordset-based clustering algorithms.

Itemsets and Wordsets

Wordsets are analogous to *itemsets* in Association Rule Mining [188] . An itemset is defined as a set of items that co-occur together. It has been commonly used to represent a collection of items purchased in a single transaction by customers for market basket analysis.

The *support* of an itemset is the fraction of transactions in which it occurs. An itemset is said to be *frequent* (i.e. a frequent itemset) if it is found to occur in at least a minimum fraction of all transactions. That is, it satisfies a minimum support threshold, *minsup*. For example, if the minsup is set to 60% (0.6), an itemset is considered as frequent if it is found in at least 60% of all transactions. Frequent itemsets are then used to form association rules, which we do not discus in this thesis. A classic technique for efficiently mining frequent itemsets is the Apriori algorithm of Agarwal and Srikant [189].

When used for text clustering, an itemset is simply a set of features (e.g. words or terms) that co-occur together in a corpus By analogy to frequent itemsets, a wordset is said to be

[25] Some clustering software packages (e.g. CLUTO) and some programming languages (e.g. Perl) include data-structures and libraries that mitigate the curse of dimensionality and improve the scalability to some extent.

frequent (i.e. a frequent wordset) if it appears in at least a minimum fraction of all documents in a corpus. For instance, if the minsup is set to 60%, a wordset is considered as frequent if it occurs in at least 60% of documents in the corpus.

In the next section, we will provide an overview of extant wordset-based clustering algorithms.

Wordset-based Clustering Algorithms

One of the pioneering techniques that investigated the use of wordsets for text clustering is the *Frequent Term-based Clustering (FTC)* algorithm of Beil et al. [169]. Given a collection of documents, FTC starts with basic pre-processing activities, such as the removal of stopwords from the documents' contents. It then identifies those words that are reliable features based on their frequencies, and uses them for clustering the documents. Next, it applies the Apriori algorithm to discover frequent wordsets from the features in corpus. Each (frequent) wordset extracted describes the label of an initially empty cluster. FTC then greedily selects the next available cluster such that it minimally overlaps with the other remaining clusters; for example, their labels share the least number of words. A document is assigned to a cluster if the wordset corresponding to the cluster is contained in the document. The recursive procedure of (greedily) picking the next available cluster and populating it with documents is repeated until all the documents in the corpus have been allocated to their respective clusters. The FTC algorithm was evaluated by estimating its performance over standard corpora employed in text clustering research. These include the CLASSIC3 corpus of scientific articles, the Reuters collection of newspaper articles, and the WAP corpus with documents culled from the Yahoo! Category pages [169]. The F1 scores achieved by FTC were respectively 0.50, 0.49. and 0.35 for the Classic3, Reuters and WAP corpora. These scores compared favorably with those of a 9-secting K-means algorithm employed as a baseline. However, they were inferior to those obtained by a bisecting K-means baseline. Furthermore, the experiments reported in [167] showed that FTC was not scalable for large corpora. For example, it was unable to produce a clustering solution for collections containing 100,000 documents.

Fung et al. [167] proposed the *Frequent Itemset-based Hierarchical Clustering (FIHC)* algorithm. FIHC starts with a pre-processing phase, for removing stopwords from the document contents. Then, it computes the tf-idf [61] scores of the words in the documents for identifying relevant features. Next, FIHC applies the Apriori algorithm to generate wordsets from the features, with each wordset corresponding to the labels of initially empty clusters. A document is assigned to a cluster only if it shares the largest number of features (words) with the other documents of the cluster. The populated clusters are arranged in a tree-structure to facilitate user browsing. FIHC was evaluated based on its performance in clustering documents from 5 corpora. These included the CLASSIC4 collection, similar to the CLASSIC3 corpus discussed earlier, the Hitech collection of San Jose Mercury newspaper articles, the WAP corpus, and two collections of articles from Reuters, referred to as Re0 and Reuters. The experimental results revealed that FIHC outperformed other state-of-the-art clustering techniques, including bisecting K-means and FTC. It achieved maximum F1 scores of 0.62, 0.41, 0.52, 0.45 and 0.60 in respectively clustering documents from the CLASSIC4, Hitech, WAP, Re0 and Reuters corpora. To gauge the efficiency and scalability of the FIHC algorithm, its run-time in clustering a collection of 100,000 documents from the Reuters collection was estimated. FIHC was found to run twice as fast as bisecting K-means, its nearest competitor. The FTC and UPGMA algorithms were unable to find a clustering solution for this corpus of 100,000 documents, and they did not complete. Given its high clustering accuracy and efficiency (scalability), the FIHC algorithm is often used as a baseline to benchmark other wordset-based clustering techniques [170, 187].

In order to further enhance the performance of FIHC, Chen et al. [187] presented the *Fuzzy Frequent Itemset-based Hierarchical Clustering (F^2IHC)* algorithm. The first step performed by F^2IHC is that of document pre-processing. Similar to FIHC and FTC, this involved removing stopwords and word stemming. Then, relevant words (features) are selected and unimportant ones discarded from the corpus based on their tf-idf scores. Compared to the original FIHC, the distinctive feature in F^2IHC is its use of fuzzy itemsets, obtained by *fuzzy Association Rule Mining* [190], for clustering documents. The maximum F1 scores of F^2IHC over the CLASSIC4, Hitech, WAP, Re0 and Reuters corpora were respectively 0.56, 0.48, 0.55, 0.57, and 0.62. Another related clustering technique is the

Maximum Capturing (MC) algorithm. The main difference between MC and other clustering techniques presented thus far is its use of minimum spanning trees to determine document similarity based on wordsets [170].

7.3 WORDSET-BASED CLUSTERING CHALLENGES

In our previous discussions, we distinguished between two broad categories of clustering techniques. The first category comprised of traditional algorithms, including K-means, bisecting K-means and UPGMA. We have seen that while these algorithms can be used to cluster text data with moderately reasonable performance, they fail to satisfy the specific requirements of text clustering such as the need for labeled clusters. Then, we introduced the second category of clustering techniques, namely, wordsets-based algorithms. These algorithms outperform their traditional counterparts like bisecting K-means in generating more accurate clusters, and also exhibit various characteristics that make them suitable for text clustering. For example, they assign meaningful labels to their clusters and have been found to be scalable for large datasets. However, wordset-based algorithms also exhibit certain shortcomings. We already outlined these shortcomings in Section 7.1. We will now describe them in more details.

Challenge 1: Inadequate feature selection

As can be seen from the preceding section, existing wordset-based algorithms have been mostly applied to standard collection of general texts, such as newspaper articles. As mentioned earlier, these documents offer reliable linguistic and statistical features, which can be readily acquired from their contents using basic feature selection mechanisms. For instance, most wordset-based algorithms rely on the frequency counts and the tf-idf scores to determine which words qualify as reliable features. The features thus identified facilitate the discovery of accurate document clusters and of meaningful cluster labels. However, conventional feature selection techniques, like tf-idf, are inadequate for domain-specific texts on two counts, as we will now elaborate.

- The first issue pertains to the cluster accuracy. Domain-specific corpora are sparse, and are characterized by the lack of redundancy. The majority of words in these corpora will be repeated in very few documents. With a tf-idf scoring scheme, these words will be awarded relatively high scores of approximately the same value. This makes it hard to determine which of those words to select as features in order to obtain the most accurate clusters. Based on their tf-idf scores, they will all be equally likely to qualify as features.
- The second issue pertains to the cluster labels. Previous studies in terminology, which were discussed in Chapter 2 of the thesis, have shown that multi-word terms are prominently used in specialized corpora. However, in extant wordset-based algorithms, only single-word expressions are used as features. Thus, given a multi-word term , such as "filament control board" in our domain-specific corpus, 3 distinct single-words, viz. "filament", "control" and "board", will be derived from it. The words will then be selected as features, for example, based on their tf-idf scores, and used as cluster labels. However, these single-word features are less informative as cluster labels than the multi-word terms from which they are derived. In our example, we will have 3 distinct single-word labels, "filament", "control" and "board", which are not as meaningful as the multi-word label "filament control board". Furthermore, as discussed in Section 7.2.2, many of the features identified by conventional techniques from sparse and informally-written texts may correspond to incoherent or irrelevant words, such as "meeting, Friday, room" or "unknown, training, contact" Such unreliable features will lead to the formation of invalid cluster labels, which do not provide any useful information on the contents of their corresponding clusters.

Challenge 2: Cluster conflicts

Most wordset-based algorithms consider a cluster to be suitable for a document if the document maximizes the (cluster's) intra-cluster similarity. For example, a document is assigned to a cluster if it has the largest number of words (features) in common with other documents already in the cluster. However, these algorithms do not consider whether the document-cluster assignment also minimizes the inter-cluster similarity, i.e. the overlap of the cluster with other clusters. With such a clustering strategy, it is highly likely that a

single document maximizes the intra-cluster similarity of multiple clusters. In this case, all the clusters will be considered as equally suitable for the document, leading to cluster conflicts. This clustering strategy of maximizing the intra-cluster similarity, without considering whether the inter-cluster similarity is minimized, is also responsible for the issue of skewed clusters, described below.

Challenge 3: Skewed clusters
Clusters generated by wordset-based algorithms tend to be skewed [170], which hinder their exploitation in real-word applications. For example, the FTC algorithm favors the formation of many small clusters, with few documents per cluster. It can even produce singleton clusters, such that each individual document corresponds to a cluster by itself. Conversely, FIHC is biased towards producing larger clusters. A document is more likely to be assigned to a larger cluster than to a smaller one [170], which further causes the cluster's size to expand.

We will investigate these issues in details during our experimental evaluation, which are presented in Section 7.6. It should also be noted that Challenges 2 and 3 are further exacerbated by the lack of reliable features, i.e. Challenge1.

7.4 CONTRIBUTIONS

The preceding section highlighted the shortcomings of existing wordset-based algorithms for text clustering. To overcome these limitations, and address our third research question, we developed and implemented the ClustText framework. Compared to extant wordset-based algorithms, ClustText incorporates several novel aspects for accurately and efficiently discovering clusters from document collections, and in particular, from informally-written and sparse texts generated in specialized disciplines like PD-CS.

Our core contributions with ClustText are listed below.

1. We propose a novel technique for feature identification. Our technique is a hybrid mechanism that integrates two feature selection methods, namely the filter and the wrapper methods. Since these methods were designed for supervised learning, such as classification, we also adapt them for our unsupervised clustering application. As will be discussed later, such a hybrid approach enables us to leverage upon the high accuracy of wrapper methods, without suffering from their prohibitive computational costs. We will also see during our experimental evaluations that the proposed technique identifies the most informative features from domain-specific texts, and successfully overcomes the challenges posed by the informal language constructs and sparsity that characterize these texts. The identified features, in turn, enable ClustText to accurately discover non-overlapping clusters of similar documents, and to generate meaningful topic descriptions for labeling these clusters (Contribution 1 in response to Challenge 1).

2. The strategy adopted by ClustText for assigning a document to a cluster is based on 3 criteria, namely, the similarity between the topics of the document and of the cluster, the intra-cluster similarity and the inter-cluster similarity. This technique enables ClustText to precisely identify the single, most appropriate cluster for a given document, overcoming the issue of cluster conflicts (Contribution 2).

3. The clustering strategy adopted by ClustText, as described above, also inhibits the formation of skewed clusters (Contribution 3).

Furthermore, ClustText being a wordset-based algorithm, exhibits three other important desiderata that enable it to overcome the limitations of traditional clustering techniques, such as K-means. They are listed below.

4. ClustText automatically labels the discovered clusters with meaningful topic descriptions. The topic of a cluster is the main theme that is recurrently mentioned in the cluster's documents. Hence, such a labeling provides valuable information

on the cluster's contents, and alleviates the need to manually and painstakingly inspect each of the clustered documents. Besides labeling the clusters, ClustText further improves their usability by organizing them in a taxonomy based on their corresponding topics. In this taxonomy, clusters labeled with a more general topic are the parents of other clusters labeled with a more specialized version of the topic. Such a taxonomy is, in effect, a topic tree, which facilitates user browsing.

5. ClustText eschews the need for specifying the initial clusters as input parameters. As was discussed earlier, this parameter is hard to define, but they significantly influence, and may even adversely affect, the clustering accuracy.

6. To compute the similarity between documents, ClustText does not compare their corresponding high dimensional, sparse vectors, which as we saw earlier is inefficient. Instead, its estimates the similarity between documents based on their most informative features and frequent wordsets, which are of much lower dimensions than traditional document vectors. This strategy is more efficient, and ensures the scalability of ClustText even for large corpora.

We will next describe our ClustText framework.

7.5 CLUSTTEXT FRAMEWORK FOR TEXT CLUSTERING

The overall architecture of ClustText is shown in Figure 7.1. Blank arrows represent inputs and outputs, while filled arrows depict the processing performed by the various phases of our ClustText framework.

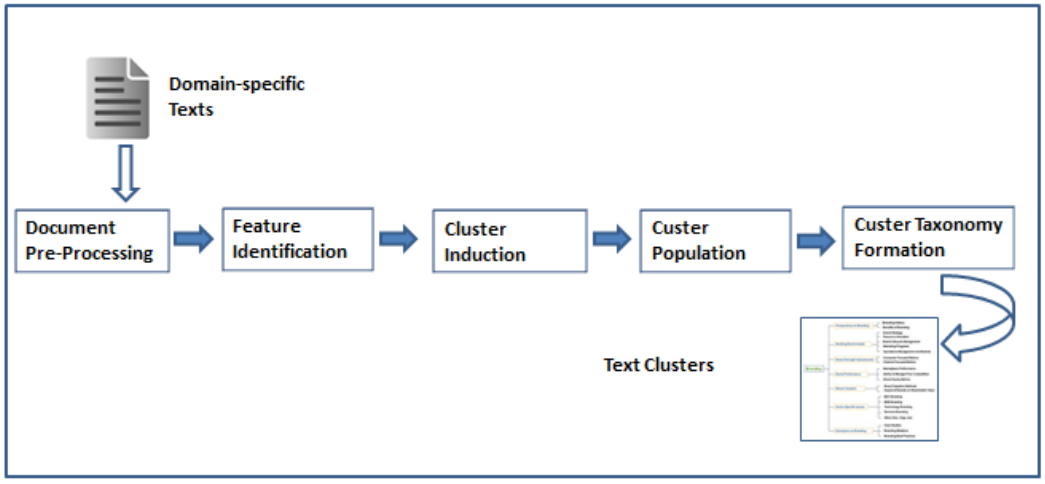


Fig. 7.1: ClustText Architecture.

ClustText takes as input a set of documents. It then pre-processes (Section 7.5.1) the documents to make them more amenable for automatic analyses by its subsequent phases. The documents are next analyzed by the Feature Identification phase (Section 7.5.2), which selects the most informative features from the documents' contents. The selected features are then used in the subsequent stages of ClustText to cluster the documents. During Cluster Induction (Section 7.5.3), we discover a clustering[26] of empty clusters. The documents in the corpus are then allocated to their most appropriate clusters by the Cluster Population phase (Section 7.5.4). Finally, Cluster Taxonomy Formation (Section 7.5.5) organizes the clusters in a taxonomy to facilitate their exploitation in real-life activities.

In the next sections, we will describe the different phases of ClustText, using, as illustrations, real-life examples from a domain-specific, sparse and informally-written PD-CS corpus.

7.5.1 DOCUMENT PRE-PROCESSING

The documents input to ClustText are pre-processed as described in Chapter 2. That is, extraneous contents, which are detrimental to the clustering accuracy, are discarded, and the Part-of-Speech (POS) of the individual tokens (e.g. words) is obtained by POS-tagging.

It should be noted that in the actual implementation, ClustText received as input a collection of pre-processed documents and an accompanying ranked-list of terms extracted from the documents by ExtTerm (Chapter 2).

The pre-processed documents are next examined by the Feature Identification phase.

[26] The noun clustering refers to a collection of clusters. It should not be confused with the verb clustering, which indicates the action.

7.5.2 FEATURE IDENTIFICATION

Feature Identification is one of the most crucial steps in our clustering procedure. As previously discussed, selecting the informative features (and rejecting the invalid ones) is important for two main reasons. First, these features enable ClustText to precisely discriminate between groups of similar documents, leading to the formation of precise, non-overlapping clusters. Second, the wordsets that are used by ClustText to label its clusters consist of features (e.g. words). Hence, to generate meaningful cluster labels, it is imperative that we obtain only the most salient features. In ClustText, feature identification is even of greater importance in order to overcome the challenges posed by our domain-specific texts. We saw earlier that these texts are sparse and abound with informal language constructs, which makes it difficult to acquire reliable features, and hinder the formation of accurate, labeled clusters. As discussed in Section 7.3, basic techniques like tf-idf are often inadequate to detect reliable features from these documents. To overcome this difficulty (Challenge 1), we present a feature identification technique, which detects two types of features, as discussed below.

Feature 1: Terms

The first type of features consists of terms. Our rationales for using terms as features are as follows.

- Terms are informative since they designate pertinent concepts in a domain. They provide us with important domain information, which facilitate the identification and grouping of similar documents into accurate clusters. For example, all documents that mentioned the terms "radiation emitter" and "radiation protector shield" are likely to deal with a common topic, and can be assigned to the same cluster.
- Using terms as features enables us to have multi-word expressions, such as "filament control board", as cluster labels. These are more meaningful than the single-word labels, such as "filament", "control" and "board", used by extant wordset-based techniques, as described in Section 7.3.

Identifying terms from our corpus is straightforward. They are already available as the output of our ExtTerm term extraction framework.

However, terms alone may not be discriminative enough for accurately identifying non-overlapping clusters of similar documents. Furthermore, they are almost always manifested as nouns or noun phrases. Hence, we need to supplement terms with other types of words, such as adjectives, verbs and adverbs, which may also be informative and help us in discovering high quality clusters.

Feature 2: Informative Context Words

Therefore, in addition to terms, we also rely on their context-words as features. A context-word of a term is simply a word that appears in the neighborhood of the term. At this stage of the chapter, it can be safely assumed that the neighbors of a term, i.e. its context-words, correspond to all the other words that co-occur with it in a document. We will elaborate on this point during our experimental evaluations.

However, not all context-words are equally informative. We have to identify as features only those which enable us to precisely distinguish between different groups of documents, leading to the discovery of high quality clusters. We also need to discard all the context-words that are irrelevant since they can adversely affect our clustering accuracy [174].

A possible solution to identify which of the context-words qualify as features is to rely on transformational techniques like Principal Component Analysis (PCA), which we introduced in Section 7.2.2. However, the features extracted by these techniques do not have a clear physical meaning [176], and may even appear to be unrelated to the original data [174]. Consequently, the clusters discovered with these features will be equally hard to interpret. They cannot be efficiently exploited for real-world applications, especially in corporate activities, which, besides accurateness, also demand that the clusters are meaningful (i.e. easy to understand).

A more attractive solution is to employ feature selection techniques like filter or wrapper methods. The features identified by these techniques are easier to interpret since they are

directly selected from the original data. Hence, they facilitate the discovery of meaningful clusters that are suitable for real-world usage. The only drawback of feature selection methods is their supervised nature; they require data that has been pre-classified into corresponding categories. In our clustering application, which is unsupervised, neither pre-classified data nor information on the categories is available. To overcome this difficulty, we have to massage our corpus in such a way that it becomes comparable to the classified datasets employed in supervised learning. We will next elaborate on the corresponding procedure that we develop for this purpose.

Our procedure first collects all the distinct terms in our corpus. Then, for each term, it harvests all the distinct context-words, which co-occur with the term in the documents. This yields a set of terms and a corresponding list of context-words per term. We consider each term as a surrogate for a category. Its context-words are then akin to the attributes that define the category. Such a representation effectively enables us to conceptualize our corpus as a classified database, consisting of attributes (context-words) and their respective categories (terms). Sample context-words and their corresponding terms, extracted from our PD-CS corpus, are listed in Table 7.1. The context-words have been conflated to their common roots (lemmatized), and are annotated with their corresponding POS. In the remainder of this chapter, we will omit the POS-tags in our illustration to improve the readability.

Context-words (Attributes)	Term (Categories)
(activate|V), (faulty|A), (insulate|V)	Radiation protector shield arm
(mount|V), (assemble|V), (resistor|N)	Cable control box

Table 7.1: Context-words and terms.

Our corpus is now more amenable to the application of feature selection methods. Next, we have to identify which of the context-words are informative and can be considered as features. To this aim, we propose a novel feature selection mechanism, incorporating both a filter and a wrapper method.

Our hybrid technique starts with a filter method. Similar to Yang and Pedersen [175] and Galavotti et al. [191], we estimate the correlation between a context-word, c*w*, and its corresponding term, *t* using the chi-squared (χ^2) statistics, as shown in equation (7.2).

$$\chi^2(cw, t) = \frac{N \times (AD - CB)^2}{(A + C) \times (B + D) \times (A + B) \times (C + D)} \qquad (7.2)$$

In this equation, *A* is the frequency with which *cw* and *t* co-occur, *B* is the frequency of *cw* occurring without *t*, C is the frequency of *t* occurring without *cw*, *D* is the frequency of neither *cw* nor *t* occurring, while *N* is the total number of documents. These values are often depicted in a contingency table depicted in Table 8.2.

	$\boldsymbol{t}$	$\boldsymbol{t'}$
$\boldsymbol{cw}$	A	B
$\boldsymbol{cw'}$	C	D

Table 7.2: Contingency table.

With such a procedure, incoherent or irrelevant words like "unknown", "meeting" and "Friday", which are less likely to be strongly associated with terms, will be awarded low scores. It is then easier to identify these types of words, and to reject them. Conversely, other more meaningful words, which are likelier to co-occur with terms, will be awarded higher scores and selected as features. These features then enable ClustText to discover accurate clusters, and to generate meaningful labels for the clusters.

It should also be noted that computing the correlation scores between terms and their context words is a commonly employed technique in distributional semantics for computing term similarity/relatedness. Also, different statistical methods, such as the Lexical Association Measures presented in Chapter 2, can be used to estimate the correlation strength. We rely on the χ^2 measure since it has been used in previous feature selection studies [175, 191].

We then sort the context-words in descending order of their χ^2 scores in a ranked-list. The next issue to deal with is how many context-words should be selected as features from the ranked-list. This is achieved by a wrapper method. Intuitively, if the correlation between a context-word and its corresponding term is statistically significant (reliable), then the context-word is likely to be informative and to denote a feature. Otherwise, if the correlation is only spurious, it is highly unlikely that the context-word represents a feature. Based on this assumption, our wrapper method varies an experimental threshold in a given range. At each of the threshold values, only those context-words whose (correlation) scores exceed the threshold are considered to be reliably correlated with their terms. These context-words and their corresponding terms are selected as features. We then keep only these features (i.e. selected context-words and terms) in the documents, and discard all the others. The documents are next fed to the subsequent phases of ClustText, and we estimate the accuracy of the clusters obtained with the different features, selected at each threshold value. The most informative features are those that produce the most accurate clusters. Details on calculating the correlation scores between context-words and terms, and on the threshold values will be presented in our experimental evaluations in Section 7.6.1.

Our hybrid filter-wrapper method, presented above, enables ClustText to leverage upon the high accuracy of wrapper methods, while overcoming their high computational costs. For example, our wrapper method does not have to evaluate all the different possible combinations of context-words when searching for the most informative features. Instead, at each threshold value, it simply relies on the ranked-list output by the filter method, and selects, from that list, only those context-words that satisfy the threshold. As will be demonstrated during our experimental evaluations, our feature identification technique enables ClustText to discover high quality, non-overlapping clusters, labeled with meaningful topic descriptions, from informally-written and sparse texts (Contribution 1).

It should be noted that at each threshold value, we select a set of features. However, we will refer to such a set simply as features. This is done to avoid confusion with *wordsets*, which will be introduced during Cluster Induction (Section 7.5.3).

7.5.3 CLUSTER INDUCTION

Cluster Induction detects a collection of empty clusters from our corpus. These clusters will be subsequently populated with appropriate documents during the Cluster Population stage (Section 7.5.4).

The central idea underlying our cluster formation strategy is that there are groups of documents in a corpus that deal with common topics. These topics can be thought of as the natural classes into which the documents can be organized. For example, in our corpus of engineers' repair notes, we can expect several documents to mention the same type of repair actions, such as "assemble, calibrate". Since they deal with the same topic, these documents can be considered to be similar, and grouped together. Such a sub-group of similar documents constitutes a cluster. We can therefore reformulate the task of discovering clusters from a corpus as one of finding the most common topics discussed in the corpus.

We now have to define what exactly we mean by topics and how we represent them. We define a topic as a set of features that co-occur in at least a minimum fraction, *minsup*, of documents in the corpus. Each feature, as described earlier, is a term or a context-word selected at a particular threshold value. We refer to such a feature set, which satisfies the minsup criterion, as a *frequent wordset*. As mentioned before, wordsets are analogous to the notion of itemsets employed in Association Rule Mining. A wordset with k features is referred to as a k-wordset. For instance, the wordsets ws_1, ws_2, and ws_3 shown in Figure 7.2 are respectively a 1-wordset, a 2-wordset and a 3-wordset.

To extract frequent wordsets from our corpus, any frequent itemset mining algorithms can be used. In ClustText, we employ the Apriori algorithm, mentioned in Section 7.2.6. It is one of the most well-known techniques for frequent itemset mining, and has been used in other wordset-based clustering algorithms [167, 168, 169].

Three frequent wordsets, ws_1, ws_2 and ws_3, extracted by Apriori from our corpus are shown in Figure 7.2.

ws_1 = {"install"}

ws_2 = {"install, configure"}

ws_3 = {"install, configure, verify"}

Fig. 7.2: Frequent wordsets.

Each of these frequent wordsets represents a cluster since they describe the common topics shared by a group of similar documents. In addition, a wordset also conveniently serves as a topic label for the cluster it represents. For example, the wordset ws_2 ={"install, configure"} is the topic label of a corresponding cluster C_2, as shown in Figure 7.3.

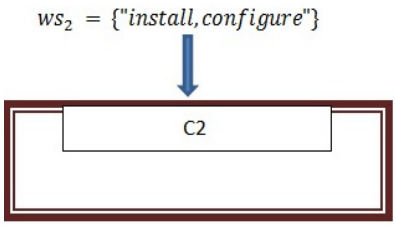


Fig. 7.3: Wordset as topic label for cluster.

Subsequently, all documents having as topic the "installation and configuration of equipment" will be assigned to C_2. By using meaningful topic labels (wordsets) to summarize the contents of each cluster, ClustText overcomes the longstanding challenge cluster labeling. Compared to the unlabeled clusters generated by traditional algorithms, the labeled clusters of ClustText can be more efficiently exploited and are more amenable for use in corporate activities.

ClustText also eschews the need for defining initial clusters prior to the actual clustering, which is difficult and may affect the accuracy. Instead, it automatically detects all the prominent topics (frequent wordsets) in the corpus, and extracts them as clusters.

The only parameter that needs to be specified in our procedure is the minsup of Apriori, which controls the number of frequent wordsets (topics) to be extracted. This parameter is, to a large extent, dependent on the intended application. For example, a user interested in the most predominant topics in the corpus will use higher minsup values so that only the most frequent wordsets, which appear in a large number of documents, are identified as clusters.

Another important observation that can be made from Figure 7.2 is that the 2-wordset ws_2={"install, configure"} is a subset of the 3-wordset ws_3 = {"install, configure, verify"}. This is a direct consequence of the monotonicity property of frequent wordsets (itemsets). According to this property, any (k-1) subset, ws_{k-1}, of a frequent k-wordset, ws_k, is also frequent. This has 2 implications for ClustText:

1. The subset ws_{k-1} will also be extracted (since it is frequent and satisfies the minsup criterion), and it will be considered as the topic label of a corresponding cluster C_{k-1}.
2. The topic described by the k-wordset, ws_k, will be a more specialized version of the topic described its (k-1) subset, ws_{k-1}. This is due to the presence of an additional feature in ws_k. For example, the topic {"install, configure, verify"} of the cluster C_3, described by the 3-wordset ws_3, is a more specialized version of the topic {"install, configure"} of the cluster C_2, described by the 2-wordset ws_2.

We will leverage upon this containment relationship between wordsets, which arise from their monotonicity property, in order to organize the clusters in a topic tree. This will be discussed in the Cluster Hierarchy Formation phase (Section 7.5.5).

At this stage in our framework, we have selected features (at a particular threshold level) from our documents, detected frequent wordsets (sets of features that satisfy the minsup criterion) in the corpus, and used them as topic labels of empty clusters. We will next populate the clusters with documents.

7.5.4 CLUSTER POPULATION

Cluster Population performs the actual clustering by assigning documents to suitable clusters. In order to achieve high quality clustering, we posit that three criteria must be fulfilled when assigning a document to a cluster. First, the document and the label of the cluster to which it is assigned must deal with the same topic. Second, the document must exhibit maximal similarity with other documents already in the cluster, and third, the document should also exhibit minimal similarity with the documents of all other clusters. Based on these propositions, we devise a clustering procedure consisting of two main steps. Step 1 concerns the first criterion, while Step 2 considers both the second and third criteria.

Step 1
In the first step, a document is assigned to a cluster only if it *covers* the cluster's topic. As an illustration, consider a document d_j and a cluster C_i, whose topic label is defined by the (frequent) wordset ws_i. The document d_j is said to cover the cluster C_i if it also contains a wordset ws_i. A similar strategy is also adopted in other wordset-based algorithms, such as FIHC [167] and FTC [169].

Specifically, if we view d_j as a bag-of-wordsets, we can formulate the criterion for d_j to cover C_i as shown in equation (7.3). The variable ws_j in this equation corresponds to some wordset found in d_j ,while ws_i is the wordset that defines the label of cluster C_i.

$$cover(d_j, C_i) \Leftrightarrow \exists\, ws_j \in d_j \;:\; ws_i \subseteq ws_j \tag{7.3}$$

For example, given a document d_j, which contains a wordset ws_j ={"install, configure"}, then it covers the cluster C_x, labeled with the wordset ws_x={"install"}, as well as the cluster C_y, labeled with the wordset ws_y={"configure"}.

As illustrated in the above example, this first step yields overlapping clusters as it is possible for a single document to cover the labels of multiple clusters.

However, cluster overlaps are problematic for two major reasons.

1. Overlapping clusters cannot be precisely evaluated, which makes it difficult to determine the performance of our proposed ClustText framework.
2. The aim of clustering is to partition a collection of objects (corpus of documents) into distinct and homogeneous sub-groups, with the least amount of overlap.

Step 2
The second step of our procedure reduces cluster overlaps. Similar to FIHC of Fung et al. [167], we keep a document only in its *most appropriate* cluster, and remove it from all the other (less appropriate) clusters that it was found to initially cover (Step 1). The next issue is now how to determine the most appropriate clusters for our documents.

Given a clustering C, we consider a cluster $C_i \in C$ to be most appropriate for a document d_j if the assignment of d_j to C_i:

a) Maximizes the intra-cluster similarity, i.e. between the documents of C_i.
b) Minimizes the inter-cluster similarity, i.e. between C_i and all other clusters.

Most existing wordset-based algorithms only take into account the condition 2a). As will be shown in our experiments, such a strategy makes it difficult for these algorithms to refine the initial overlapping clusters (Step 1). Consequently, they suffer from the issues of

cluster conflicts and of skewed clustering, whereby they are unable to precisely locate the single most suitable cluster for a document, and they discover clusters of disparate sizes.

Taking into account the two aforementioned considerations, 2a) and 2b), we propose a technique for estimating the degree to which a given cluster is appropriate for a document. A document is then assigned to (or kept in) that single cluster which achieves the maximum appropriateness score. Our technique for computing the appropriateness score of a cluster C_i for a document d_j is presented in the pseudo-code below.

```
Procedure cluster_appropriateness (C_i, d_j)
1    for each f ∈ d_j
2        for each d_x ∈ C_i , x ≠ j
3            if f ∈ d_x
4                intra_cluster_similarity++

5    for each f ∈ d_j
6        for each C_z ∈ C, z ≠ i
7            for each d_y ∈ C_z
8                if f ∈ d_y
9                    inter_cluster_similarity++
10           document_count++
11   inter_cluster_similarity = inter_cluster_similarity/document_count

12   appropriateness_score = intra_cluster_similarity – inter_cluster_similarity
13   return appropriateness_score
```

Lines 1-4 are concerned with our first criterion, 2a), pertaining to the intra-cluster similarity. The intra-cluster similarity of C_i given d_j is estimated as the number of features, f, that d_j shares with the other documents d_x of C_i. The more features that d_j has in common with these other documents d_x, the more similar they are. As a consequence, the intra-cluster similarity score of C_i after d_j is assigned to it will be large (line 4). This indicates that C_i will still be a homogeneous sub-group of similar documents after the addition of d_j. Conversely, the fewer features that d_j shares with the documents d_x of C_i, the less similar they are. The intra-cluster similarity score of C_i will be much lower, indicating that after the addition of d_j, C_i may no longer constitute a homogeneous sub-group of similar documents.

Lines 5-11 are concerned with our second criterion, 2b), pertaining to the inter-cluster similarity. The inter-cluster similarity of C_i given d_j is estimated as the number of features that d_j has in common with the other documents d_y of all other clusters C_z. The fewer features that d_j has in common with these documents d_y, the smaller the inter-cluster similarity score of C_i after d_j is assigned to it (line 9). This indicates that the addition of d_j will not increase (or deteriorate) the overlaps between C_i and the other clusters C_z. Conversely, the more features shared by d_j and the documents d_y of C_z, the larger the inter-cluster similarity score, indicating that the addition of d_j to C_i will deteriorate the overlaps between C_i and the other clusters C_z.

The appropriateness score of C_i for d_j is then given as the difference between the corresponding intra-cluster similarity score and inter-cluster similarity score (line 12). It should be noted that since the inter-cluster similarity score is computed by considering a much larger number of documents (from all other clusters) than the intra-cluster similarity (from a single cluster), we normalize its value by dividing it with the corresponding number of documents (line 11).

Following the same procedure as described above, we calculate the appropriateness of each cluster in the clustering C for the given document d_j. The most appropriate cluster, $C_{appropriate}$, for d_j is that cluster which achieves the highest appropriateness score, given by equation (7.4).

$$C_{appropriate}(d_j) = \underset{C_i \in C}{argmax}\ cluster\_appropriateness(C_i, d_j) \qquad (7.4)$$

Since $C_{appropriate}$ yields the highest appropriateness score for d_j, and given that this value is calculated as the difference between the corresponding intra-cluster similarity and inter-cluster similarity scores, this entails that addition of d_j to $C_{appropriate}$ maximized its intra-cluster similarity, satisfying criterion 2a), and minimized its inter-cluster similarity, satisfying criterion 2b). In this way, our procedure is able to generate the most accurate (i.e. highest intra-cluster similarity), non-overlapping (i.e. lowest inter-cluster similarity) clusters from a corpus.

Compared to the clustering strategy employed by other wordset-based techniques, our procedure, described above, overcomes the issues of cluster conflicts (Challenge 2) and of skewed clusters (Challenge 3). As will be demonstrated during the experimental evaluations, this procedure precisely identifies the single, most appropriate clusters for each document (Contribution 2), and is not prone to discover clusters of extreme sizes, such as singletons clusters (Contribution 3). In addition, our procedure does not estimate the similarity between documents based on their corresponding high dimensional and sparse vectors, which compromise the scalability. Instead, it simply compares the small sets of reliable features found in each document, which is beneficial to its scalability.

To illustrate our Cluster Population phase, we will consider the case of clustering a number of documents, d_a, d_b, d_c,…. into three clusters C_1, C_2, C_3. For conciseness, we will omit detailed information pertaining to the documents' contents and the clusters' labels.

We assume that the resulting clustering after Step 1 of our overall procedure is as shown in Figure 7.4.

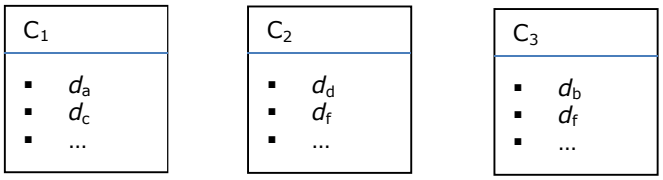


Fig. 7.4: After initial clustering (Step 1).

Then, in Step 2, we compute the appropriateness score between each cluster and document using the *cluster_appropriateness* algorithm, presented earlier. The corresponding scores achieved by the 3 clusters for document d_a are listed in Table 7.3.

Document = d_a	
cluster_appropriateness(C_1, d_a)	0.56
cluster_appropriateness (C_2, d_a)	0.43
cluster_appropriateness (C_3, d_a)	0.75

Table 7.3: Cluster appropriateness scores for document d_a.

The highest score is achieved by cluster C_3, indicating that the assignment of d_a to C_3 maximizes its intra-cluster similarity and minimizes its inter-cluster similarity. Let us assume that this occurs since d_a has the largest number of features in common with the document d_b in C_3 (and the least number of features in common with the other documents of all other clusters). Therefore, we move document d_a from its original cluster C_1 to C_3. Our clustering after this re-assignment is shown in Figure 7.5.

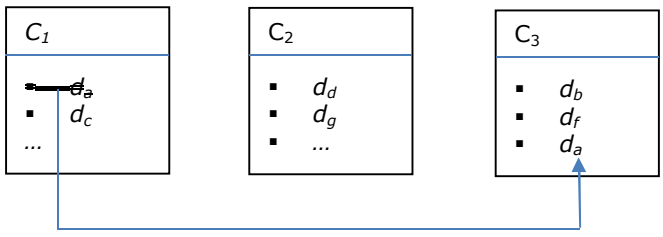


Fig. 7.5: Reassigning document d_a.

Next, we consider the document d_b, and find the most appropriate cluster to accommodate it. The appropriateness scores achieved by our three clusters for d_b are shown in Table 7.4.

Document = d_b	
cluster_appropriateness(C_1, d_b)	0.32
cluster_appropriateness (C_2, d_b)	0.69
cluster_appropriateness (C_3, d_b)	0.55

Table 7.4: Cluster appropriateness scores for document d_b.

The most appropriate cluster for d_b is found to be C_2. Therefore, we move d_b from its original cluster C_3 to C_2, so that our resulting clustering is as shown in Figure 7.6.

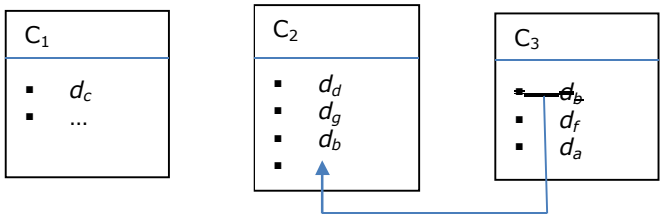


Figure 7.6: Reassigning document d_b.

Our overall clustering procedure seems to be relatively straightforward. However, there is one subtlety that still needs to be taken care of. We saw previously that document d_a was attracted to cluster C_3 because it had the largest number of features in common with the document d_b, which was then residing in cluster C_3. However, we subsequently moved d_b to another cluster, viz. C_2, which was found to be most appropriate for it. Since d_b is no more in cluster C_3, we have to reconsider whether this cluster (i.e. C_3) is still the most appropriate for d_a. Consequently, after having assigned all the documents to their most appropriate clusters, we have to recalculate the clusters' appropriateness scores until a point is reached where all documents remain in their respective clusters. This iterative approach is borrowed from the K-means algorithm, which, after populating the clusters re-computes the centroids, and determines again whether each data object is in its best (closest) cluster.

At the end of this phase, ClustText will produce a collection of high quality clusters of similar documents, which overlap minimally. The clusters will also be given meaningful labels, which provide a summary of the main topics described in the clustered documents. An example of such a clustering solution, as discovered by ClustText, will be presented in Figure 7.7 in the next section.

7.5.5 Cluster Taxonomy Formation

In Cluster Taxonomy Formation, we organize our populated clusters in a taxonomy based on their topics. Our taxonomy construction is hinged upon the monotonicity property of wordsets, discussed in Section 7.5.3. As was mentioned before, two consequences of this property are that a (k-1) subset of a frequent k-wordset will also be extracted as a topic label for a cluster, and that the topic described by a k-wordset is a more specialized version of the topic described its (k-1) subset.

To construct our taxonomy, we adopt the approach of Fung et al. [167]. We proceed in a bottom-up fashion, starting with clusters whose topic labels are defined by k-wordsets. We refer to them as k-clusters. For instance, the cluster C_3, labeled with the wordset ws_3={"install, configure, verify"} is a 3-cluster. Similarly, the cluster C_2, labeled with the wordset ws_2={"install, configure"} is a 2-cluster.

Given a k-cluster C_k, we insert it in the k^{th} level of our taxonomy. For instance, our 3-cluster C_3, with topic label {"install, configure, verify"}, will be inserted at the 3rd level. Then, we leverage upon the monotonicity property of wordsets (cluster labels) to find the

parent of C_k. We define C_k's parent as that (k-1)-cluster, C_{k-1}, such that its topic label (wordset) is a subset of the topic label of C_k. For example, the 2-cluster, C_2, with topic label {"install, configure"}, is a candidate parent for C_3. In this way, a child cluster, e.g. C_3, deals with a more specialized topic than its parent cluster, e.g. C_2.

However, it is possible that several C_{k-1} clusters qualify as candidate parent for C_k. For instance, consider a cluster C_x with topic label {"install, verify"}. C_x's label is also a subset of C_k's label. Therefore, together with C_2, C_x also qualifies as a potential parent for C_k. Our task is now to find the best parent for C_k.

Similar to Fung et al. [167], we reformulate the parent selection problem as one of Cluster Population, as described in Section 7.5.4. We first merge (e.g. concatenate) all the documents in the child cluster, C_k, into a single document, $doc(C_k)$. The (k-1)-clusters, which qualify as parents for C_k, are then viewed as candidate clusters, C_i, for $doc(C_k)$. Then, finding the best parent for C_k is akin to finding the most appropriate cluster for $doc(C_k)$. This is achieved by computing $cluster\_appropriateness(C_i, doc(C_k))$, as discussed previously, where C_i is a (k-1)-cluster that is a potential parent of C_k. Finally, that cluster, which maximizes the appropriateness score according to equation (7.4), is taken as C_k's parent.

After finding the best (k-1)-cluster as C_k's parent, it is inserted in the $(k-1)^{th}$ level of the taxonomy. These steps are repeated until we reach the 1^{st} level. This level is populated with 1-cluster, which contain a single feature in their wordset, e.g. {install"}, {"verify"} or {"configure"}.

Sample clusters discovered by our ClustText application from a domain-specific corpus of engineers' repair notes are depicted in Figure 7.7. For conciseness, we show only 2 clusters and 3 documents per cluster. In this example, "Cluster 1.1" is the child of "Cluster 1". Three important observations can be made from this illustration:

1. Each cluster is assigned a meaningful label (wordset), describing its main topic. For example, "Cluster 1.1" is labeled with the wordset {"Adjust, Monitor"}, indicating that it deals with the topic of "adjustments made to monitor screens".
2. Following from the previous observation, in our clustering, a child cluster, e.g. "Cluster 1.1.", deals with a more specialized topic than its parent, e.g. "Cluster 1".
3. The wordset that defines the topic label of a parent cluster, e.g. {"Adjust"} of "Cluster 1", is a subset of the wordset of the child cluster, e.g. {"Adjust, Monitor"} of "Cluster 1.1".

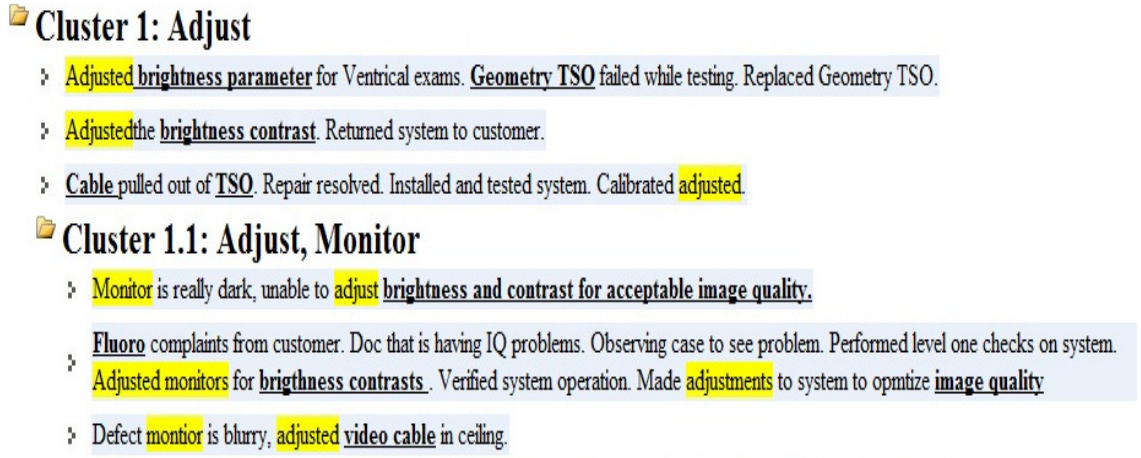


Fig. 7.7: Taxonomy of labeled clusters discovered by ClustText.

Such a clustering facilitates user browsing and makes it easier to employ the results of ClustText for supporting various types of corporate activities. For example, it provides end-users with an overview of topics (i.e. a topic tree) and their relations (taxonomy). The users can then decide which of the topics are of most interest to them by traversing the topic tree. Subsequently, they can zoom into the relevant documents in the corresponding

cluster, or dive into deeper levels of the topic tree to investigate more specialized topics. This form of user browsing, which we discussed in Chapter 1, offers numerous benefits over traditional keyword searches. For example, it is particularly suitable when users are unable to precisely formulate what they are looking for or when they are unable to anticipate the keywords that are used to express the desired information in the documents. In addition, the clusters and topics that are visited during user browsing provide useful contextual information, which is not available by keyword searching. To further assist the end-user in efficiently finding the required information, we also explicitly indicate the occurrences of terms, by underlining them and using stronger fonts, in the clustered documents, as shown in Figure 7.7.

The clusters presented above can be used to identify groups of products that require similar repair actions, for example, "adjustments" and "replacements". Based on the information derived from the clusters, corrective actions can be implemented to improve the quality of these products, leading to more satisfied customers.

7.6 EXPERIMENTAL EVALUATION

In this section, we evaluate the performance of our proposed ClustText framework in clustering our informally-written and sparse domain-specific documents. In Sections 7.6.1 – 7.6.3, we will discuss the results of the various phases of ClustText. Then, in Section 7.6.4, we will describe how we gauged the accuracy of the clustering solution produced by ClustText, and present the actual performance (accuracy) scores. Next, in Section 7.6.5, we compare the accuracy achieved by ClustText with that of FIHC, which we used as a baseline. Finally, in 7.6.6, we benchmark the scalability of ClustText against the FIHC baseline. All the experiments reported in this section were executed on a high performance computing cluster[27].

7.6.1 FEATURE IDENTIFICATION

We identified two types of features from our documents. Our first type of features consisted of terms extracted from the documents' contents. To this aim, we automatically selected the top-n expressions in the ranked-list output by the ExtTerm framework (Chapter 2). In our experiments, n was set to 1000. The reason for considering the top-1000 expressions was that they were more likely to correspond to real terms and were more accurate than lower ranked entries. Next, we searched for the occurrences of the selected 1000 terms in our corpus, and extracted the 2115 documents in which they were mentioned. This sub-corpus of 2115 documents was used in the remainder of our experiments.

Then, we identified our second type of features. For each term in a document, we extracted all the context-words that co-occurred with it in the document. Thus, the window was taken to be the entire document. Our main justification in using such a large window is that it enabled us to capture as many informative context-words as possible from our sparse documents. Using a smaller window, for instance, consisting only of the words immediately preceding/succeeding a term, could have resulted in the premature loss of many informative features. This would have further exacerbated the data sparsity issue, with an ensuing drop in our performance. The only difficulty with our large window is that it captured a number of irrelevant context-words. However, these would be later discarded based on their relevancy scores, and thus, they did not constitute a problem for our clustering accuracy.

[27] http://www.rug.nl/cit/hpcv/faciliteiten/HPCCluster

Some statistics on the sub-corpus and on the number of terms and context-words identified as presented in Table 7.5.

Number of documents	2115
Number of distinct terms	1000
Number of distinct context-words	17,775
Total (terms and context words)	18,775

Table 7.5: Statistics on sub-corpus.

Then, after extracting the context-words, we identified those that were informative, and that could qualify as features. Using our hybrid filter-wrapper method described in Section 7.5.2, we first estimated the correlation between the context-words and their corresponding terms. This was computed using the χ^2 statistics. For context-words that co-occurred with more than 1 term, the maximum correlation score was used. The context-words were then sorted in descending order of the correlation scores with their terms in a ranked-list. An example of such a ranked-list, depicting the top-10 context-words based on their scores, is presented in Table 7.6.

	Context-word: *cw*	**Term:*t***	**Correlation score: $\chi^2(cw,t)$**
1	repair	radiation protector shield arm	145.45
2	connect	cable control box	143.21
3	remove	filament control board	140.03
4	brightness	monitor screen	120.45
5	faulty	body guard sensor button	100.97
6	pull	coaxial cable	97.67
7	change	control known	94.67
8	error	grid switch	91.45
9	calibrate	geometry rotation	89.67
10	intermittent	monitor screen	88.01

Table 7.6: Top-10 context-words and terms.

We considered a context-word to be informative only if its correlation with a corresponding term was found to be statistically significant. All other context-words, which were spuriously correlated with their corresponding terms, were dropped from the documents. This was achieved using a standard hypothesis testing procedure, described below.

Given a context-word, *cw*, and its corresponding term, *t*, we formulated our null and alternative hypotheses, H0 and Ha, as

H0: *cw* and *t* are independent (i.e. not correlated).
Ha: *cw* and *t* are correlated.

We defined three different levels of significance, α_1=0.05, α_2=0.01 and α_3=0.001, to indicate the probability of incorrectly considering *cw* and *t* to be correlated when they are not (i.e. a Type 1 error). For example, at α_2=0.01, the probability of making a Type 1 error is 0.01. These values of 0.05, 0.01 and 0.001 were chosen since, by convention, they have been the most commonly used in practice [192].

Then, we defined our *degrees of freedom*, *df*. Considering our contingency table, shown earlier as Table 7.2, with *r*=2 rows and *c*=2 columns, *df* is calculated as

$$df = (r-1) \times (c-1) = (2-1)(2-1) = 1$$

Using the chi-squared distribution table [193, 194] with 1 degree of freedom, we next determined the three critical values for each of the significance levels α_1, α_2 and α_3. They were respectively, θ_1=3.841, θ_2=6.635 and θ_3=10.828, as shown in the 2nd column of Table 7.7.

These critical values represent a threshold that needs to be exceeded in order for the null hypothesis to be rejected. As an illustration, consider our context-word *cw* and its corresponding term *t* with a correlation score $\chi^2(cw,t)$ computed earlier. Assuming a significance level of α_2=0.01 and a corresponding critical value θ_2 =6.635, if the score between *cw* and *t* exceeds the threshold θ_2, as shown in equation (7.5), we can reject the null hypothesis that *cw* and *t* are independent; the probability of making an error (i.e. of being wrong) in rejecting this hypothesis is only of α_2=0.01. This means that we can accept the alternative hypothesis that *cw* and *t* are indeed correlated. The correlation between *cw* and *t* is then considered to be statistically significant, and *cw* is likely to correspond to a feature.

$$\chi^2(cw,t) > \theta_2 \qquad (7.5)$$

On the other hand, if the score between *cw* and *t* is less than the threshold θ_2, then any correlation between them is purely coincidental, and we do not consider *cw* as a feature.

Therefore, in our experiments, we used the three critical values θ_1, θ_2 and θ_3 as thresholds. At each of these three thresholds, we selected as features only those context-words whose scores exceeded the threshold value, as well as the corresponding terms of these context-words. We will refer to the features harvested at the three thresholds θ_1, θ_2 and θ_3 respectively as ϕ_1, ϕ_2 and ϕ_3. As we stated earlier, ϕ_1, ϕ_2 and ϕ_3, are in fact sets of features. But we will simply call them features to avoid confusion with wordsets.

The number features collected at the different threshold (critical) values is shown in the last column of Table 7.7.

Significance levels α	**Critical values θ, df=1**	**Num. distinct features |ϕ|**
α_1=0.05	θ_1=3.841	$\|\phi_1\|$=873
α_2=0.01	θ_2=6.635	$\|\phi_2\|$=492
α_3=0.001	θ_3=10.828	$\|\phi_3\|$=145

Table 7.7 :Features identified at different threshold levels.

At each of the thresholds, only the selected features were kept in the documents; all others were considered irrelevant and discarded. The documents were then fed to the subsequent phases of ClustText, and we estimated the accuracy of the clusters produced by ClustText (separately for each of the different features ϕ_1, ϕ_2 and ϕ_3). As will be shown in Section 7.6.4, the most informative features, which yielded the most accurate clusters, were ϕ_2, collected at the threshold θ_2.

We will experimentally demonstrate in Section 7.6.5 that our feature selection technique plays an important role in overcoming the challenges posed by the informal language constructs and sparsity that characterize our documents (Challenge 1). The features that it acquires from these documents enable ClustText to group them into high quality, non-overlapping clusters, labeled with meaningful topic descriptions (Contribution 1).

7.6.2 CLUSTER INDUCTION

The input to the Cluster Induction phase consisted of documents and their features, selected at a particular threshold as described previously.

We applied the Apriori algorithm for discovering frequent wordsets (sets of frequently co-occurring features) from the documents. In our experiments, we employed the Apriori implementation available as part of the WEKA Machine Learning library [195, 196], (version 3.6.5) [197].

Each extracted wordset was used as a topic label for an empty cluster. In this way, ClustText addressed the limitations of traditional algorithms, which generate unlabeled clusters that are difficult to exploit in real-world applications. Also, ClustText did not require the definition of initial clusters, and overcame the difficulties in manually

identifying these clusters prior to the clustering. Instead, it automatically detected all the relevant clusters (and their topic labels) from our corpus.

In our experiments, all parameters of Apriori were left unchanged, except for the minsup. As already mentioned, this parameter indicated the minimum fraction of documents in which a wordset should appear in order to be considered as frequent, and to be extracted as a cluster. There are no guidelines stipulating how to define an optimal minsup value, which is application dependent. For our work, we followed the approach adopted by other wordset-based clustering algorithms [167, 187]. These algorithms discovered different clusterings, consisting of 3, 15, 30 and 60 clusters. Their performance was then estimated by averaging their accuracy for the different clusterings.

Therefore, in our experiments, we also varied our minsup parameter so as to obtain the desired number of clusters. For ease of evaluation, which was manual in our case, we considered 3, 30 and 60 clusters. Thus, we varied our minsup to obtain 3 different clusterings with 3, 30 and 60 clusters respectively. As an illustration, we present in Table 7.8, our minsup values that yielded the desired clusters (wordsets) from our corpus when using the features ϕ_2. We will populate these empty clusters and evaluate them in Sections 7.6.3 and 7.6.4 respectively.

Minsup	Number of Clusters (Wordsets)
0.15	60
0.27	30
0.72	3

Table 7.8: Minsup values and number of clusters.

It can be observed that higher values of minsup were required to extract smaller clustering with fewer clusters. This is expected since an equally small number of wordsets occurred in a sufficient number of documents in order to satisfy these higher minsup values. Therefore, very few wordsets were considered as frequent, and extracted as clusters. For instance, to obtain 3 clusters, we set our minsup to 0.72, indicating that only those wordsets occurring in at least 72% of all documents should be extracted as clusters. Conversely, to obtain larger clusterings with many clusters, lower minsup values were required so that the equally large number of wordsets, which satisfied these low minsup, were extracted as clusters.

7.6.3 Cluster Population

Each of our clusterings, with 3, 30 and 60 clusters, was populated with documents, and the accuracy scores of the resulting clusters were then estimated. This was performed (separately) for each of the features, ϕ_1, ϕ_2 and ϕ_3, harvested during Feature Identification (Section 7.6.1), as will be next described.

7.6.4 Evaluating ClustText's Accuracy and Selecting Threshold

In this section, we will evaluate the accuracy of the clusters discovered by ClustText from our PD-CS corpus of engineers' repair notes. We will also determine which of the features, ϕ_1, ϕ_2 and ϕ_3, were the most informative and yielded the most accurate clusters.

The quality of a clustering solution is commonly evaluated as the accuracy of the clusters' contents. This is often computed using traditional metrics like the F1 or purity measures [4, 198]. Broadly speaking, these measures estimate the percentage of correct documents in the clusters. However, one of the strengths of ClustText is the discovery of labeled clusters. Therefore, our evaluation procedure must also take into account the accuracy of the clusters' labels, besides that of the clusters' contents. In fact, evaluating the labels of clusters is as important as – if not even more important than – evaluating their contents. This is because the labels provide users with an indication of the main topics discussed in the clustered documents. Incorrect or meaningless labels can be misleading to users, and impede the clusters' usage in real-world applications. Furthermore, cluster labels can be considered as the first contact point between users and a cluster. Users often decide whether a cluster's contents are worth investigating based on its label.

Therefore, to estimate the quality of our clustering solution, we define a measure which incorporates 2 components. The first component, which we call the *topic_significance* assesses the topic label of a cluster, while the second component, called the *cluster_precision*, is concerned with accuracy of the clustered contents. We will now elaborate on these two components.

Topic_Significance for Cluster's Labels

The topic_ significance of a cluster label indicates whether it describes a relevant topic/theme in our corpus/domain. Determining the topic_significance of labels requires domain-specific knowledge. Therefore, in the absence of any additional knowledge resources, we relied on domain experts to estimate the topic_significance of cluster labels. Two human judges were independently asked to gauge the relevancy of the cluster labels on a scale of 0 (minimum) to 10(maximum). Labels that were found to be closely related to the domain were awarded higher scores than irrelevant ones.

Estimating the topic_significance scores involves an element of subjectivity, which can compromise the quality of our evaluation. To improve the reliability of our results, we computed the topic_significance of a cluster C_i, labeled with a wordset, ws_i, by averaging its scores from the two judges.

In addition, to ensure that the judges agreed on what labels they considered relevant and on the scores they awarded to these labels, we proceeded as follows. We sorted the cluster labels in descending order of their topic_significance scores, i.e. labels at rank 1 were those with the maximum score (of 1), and lower-ranked labels had smaller scores. If multiple labels had the same scores, i.e. tied values, their corresponding ranks were computed as the average of the ranks that they would have otherwise occupied. In this way, two ranked lists of labels were created – one for each judge. For example, in our clustering of 30 clusters, we had 2 ranked lists, each with 30 cluster labels and their scores. Then, we estimated the Spearman rank correlation (with correction for tied values) between the ranked lists of the two judges. The corresponding correlation between the two judges, for each of our clustering of 3, 30, and 60 labels, are presented in Table 7.9.

Clusters (labels)	**Spearman Rank Correlation (topic_significance)**
3	0.87
30	0.80
60	0.74

Table 7.9: Correlation between judges for topic-significance.

The values in Table 7.9 indicate a high degree of correlation (agreement) between the judges in assigning the topic-significance scores (ranks) to the labels. This suggests that the agreement between the judges were not coincidental, and that our performance scores can be considered as reliable.

Given a clustering, C, with $|C|$ clusters (e.g. 3, 30, and 60), we computed its overall topic_significance as shown in equation (7.6). In this equation, C_i is a cluster in the clustering C, and $topic\_significance(C_i)$ is the average of the topic_significance score awarded to C_i by the 2 judges.

$$topic\_significance(C) = \frac{\sum_i^{|C|} topic\_significance(C_i)}{|C|} \qquad (7.6)$$

To illustrate the computation of the overall topic_significance, we will consider a clustering C with $|C|=3$ clusters, C_1, C_2 and C_3. They are shown in Table 7.11, together with the average topic_significance scores awarded to them by the 2 judges. Using equation (7.6), the overall topic_significance for the entire clustering C is then calculated as

$$topic\_significance(C) = \frac{0.7 + 0.6 + 0.8}{3} = 0.70$$

Cluster precision for Cluster's Contents

To measure the accuracy of the clustered contents, we can employ either the F1 or the purity measure, as mentioned earlier. However, computing the F1 score usually requires the existence of pre-classified data as gold-standards. These gold-standards are then used to compare the clustering solution produced by an algorithm. In our experiments, no such gold-standards were available, and their creation was impeded by the Knowledge Acquisition bottleneck.

The purity measure, which is easier to compute without a gold-standard, seemed to be a more attractive alternative. The conventional definition of the purity of a cluster C_i is given as the largest fraction of objects (documents) in C_i that belong to the same class (deal with the same topic). However, this definition is incomplete for our solution, which consists of labeled clusters. It does not consider whether the documents in the cluster are related to the cluster's topic label. Consequently, the conventional purity measure may provide us with an overestimated value of our clusters' accuracy. As a simple example, consider a cluster whose topic is labeled as (the wordset) {"install, configure"}. Even if the majority, say 90%, of documents in the cluster deals with a different topic, like {"replacement"}, it will still be awarded a high purity score of 0.9, and will be incorrectly considered to be accurate. Such a clustering, in which the labels fail to clearly describe the main topics of the clusters can be misleading to users, as already discussed.

Therefore, when estimating the accuracy of the cluster's contents, we need to consider whether the documents in the cluster indeed deal with the same topic as described by the cluster's label. That is, we need to determine the *precision* with which the documents in the cluster characterize the cluster's topic. One efficient strategy to compute the cluster's precision is to automatically search for occurrences of its topic label (wordset) in the clustered documents. This strategy is intuitive and easy to implement. However, it can also overestimate the cluster accuracy score. This is because a document in a cluster may not always be related to the cluster even if it contains the wordset that describes the cluster's topic. For instance, the same wordset can convey different meanings when it is used as the cluster's label and when it appears in the document. As an illustration, consider a cluster, C_i, labeled with the wordset {"file transfer"}, and dealing with the topic of "file/data/information transfer between remote equipment". Some of the clustered documents are depicted in Figure 7.8.

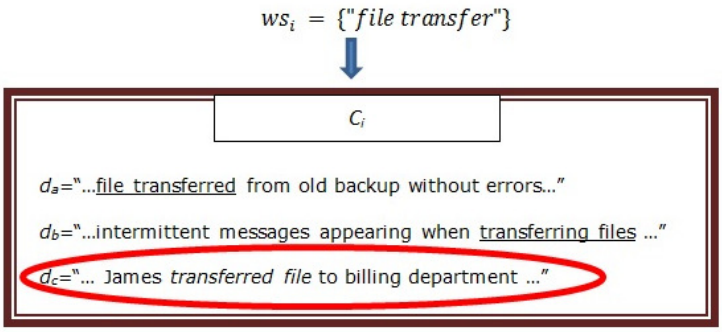


Fig. 7.8: Documents and their relations to clusters.

All of the clustered documents, d_a, d_b and d_c, contain the cluster's wordset (topic label) {"file transfer"} (in inflected forms). However, in d_c, "file transfer" refers to the physical transfer of a file (e.g. work orders) to a physical location. It has a different connotation from the intended meaning, which is the transfer of digital files between equipment. Therefore, d_c is not related to the cluster even though it contains the topic label.

Consequently, in order to determine which documents are actually related to the cluster, we cannot merely search for the occurrences of the cluster's topic label in the documents. Due to these difficulties, in our experiments, we had to adopt a manual, but more reliable, approach to determine the clusters' precision. Our 2 judges inspected each of the discovered clusters, and identified the documents that dealt with the clusters' topics, as indicated by their labels. The precision of a cluster, C_i, was then estimated by each judge

using equation (7.7), where n_i is the number of documents in C_i that were found to deal with the topic of C_i, and $|C_i|$ is the total number of documents contained in C_i.

In the event that a cluster had an invalid (poor quality) label, such as {"cable, unknown"}, then we ignored the label. The value of n_i was given as the largest fraction of documents in the cluster that dealt with a common topic, regardless of whether that topic was related to the (invalid) cluster label. In effect, we measured the cluster's purity.

$$cluster\_precision(C_i) = \frac{n_i}{|C_i|} \qquad (7.7)$$

As before, to improve the reliability of our results, we computed the cluster_precision of a cluster C_i by averaging its scores from the two judges. We also computed the Spearman rank correlation as follows. We sorted the cluster_precision scores awarded to the clusters by the two judges in descending order, and created two ranked lists (one per judge). For tied values, i.e. multiple clusters with the same score, their ranks were computed as the average of the ranks that they would have otherwise occupied. We represented the clusters in the ranked lists using unique identifiers. We did not use the clusters' labels to avoid confusion with the topic_significance scores, computed earlier. For example, in the ranked-list for the clustering of 30 clusters, we had 30 identifiers, C_0 … C_{29} representing the clusters and their corresponding cluster_precision scores. The Spearman rank correlation scores (with correction for tied values) between the 2 judges are given in Table 7.10.

Clusters	Spearman Rank Correlation (cluster_precision)
3	0.75
30	0.73
60	0.65

Table 7.10: Correlation between judges for cluster_precision.

The correlation scores shown in the above table suggest a relatively high degree of correlation in the decision of both judges when awarding the cluster_precision scores to the clusters. They indicate that our performance scores can be considered as reliable.

To calculate the overall cluster_precision of a clustering C, with $|C|$ clusters (e.g. 3, 30, and 60), we used equation (7.8).

$$cluster\_precision(C) = \frac{\sum_i^{|C|} cluster\_precision(C_i)}{|C|} \qquad (7.8)$$

We will now illustrate the computation of the overall cluster_precision using our clustering C with $|C|=3$ clusters, C_1, C_2 and C_3, and their respective cluster_precision scores, listed in Table 7.11. Using equation (7.8), the overall cluster_precision for the entire clustering C is then calculated as

$$cluster\_precision(C) = \frac{0.9 + 0.9 + 0.8}{3} = 0.87$$

Cluster, C_i	Topic_significance(C_i)	Cluster_precision(C_i)
C_1	0.7	0.9
C_2	0.6	0.9
C_3	0.8	0.8

Table 7.11: Sample clusters and scores.

Finally, we estimate the accuracy of a clustering C as the product of its cluster_precision and topic_significance scores, as illustrated in equation (7.9).

$$accuracy(C) = \ topic\_significance(C) \times cluster\_precision(C) \qquad (7.9)$$

Considering our previous clustering example, its accuracy according to equation (7.9) is given as

$$accuracy(C) = 0.70 \times 0.87 = 0.61$$

In this way, we estimated the accuracy of our 3 different clusterings, consisting of 3, 30 and 60 clusters. The corresponding accuracy scores, as computed by equation (7.9), are presented in Table 7.12 for each of the different features ϕ_1, ϕ_2 and ϕ_3.

Features	Num. of Clusters	Topic_significance	Cluster_precision	Accuracy
ϕ_1	3	0.81	0.80	0.65
	30	0.82	0.83	0.68
	60	0.81	0.84	0.68
	Avg.	0.81	0.82	0.67
ϕ_2	3	0.84	0.84	0.71
	30	0.85	0.86	0.73
	60	0.88	0.88	0.77
	Avg.	0.86	0.86	0.74
ϕ_3	3	0.82	0.82	0.67
	30	0.84	0.83	0.70
	60	0.84	0.85	0.71
	Avg.	0.83	0.83	0.69

Table 7.12: Accuracy scores of ClustText for different features.

Three main observations can made from the results presented in Table 7.12. They are discussed next.

Observation 1: Contribution of topic_significance and cluster_precision
The contributions of the topic_significance and of the cluster_precision components are both equally significant to our accuracy. This indicates that ClustText not only discovers relatively precise clusters (high cluster_precision), but that it also generates meaningful labels for these clusters. In addition, these results indicate that our novel feature identification mechanism (Section 7.5.2) identified informative features from our corpus of domain-specific texts. These features then contributed to the formation of meaningful labels and precise clusters. Furthermore, the above results also show that our cluster population approach (Section 7.5.4), which optimizes both the intra-cluster similarity and inter-cluster similarity, successfully identified similar documents and grouped them together into non-overlapping clusters.

Observation 2: Higher Accuracy for Larger Clusterings
It can be seen that the clustering accuracy improved as the number of clusters increased. This was mainly because in large clusterings, for e.g. with 60 clusters, the document collection could be more precisely partitioned as more clusters were available. Each cluster was then a small sub-group of documents, which were closely related to the cluster's topic. These clusters were awarded high cluster_precision scores.

Conversely, when fewer clusters were available, for e.g. only 3, more documents were assigned to the same clusters. Since these documents often dealt with unrelated topics, the clusters were awarded lower cluster_precision scores.

Observation 3: Most Informative Features

The most accurate clusters were those obtained when the features ϕ_2 were used to cluster the documents. Therefore, these features were considered to be most informative, and were used in the remainder of our experiments.

7.6.5 CLUSTTEXT VS. FIHC BASELINE

To benchmark the performance of ClustText, we compared its output against that of the FIHC baseline. FIHC was chosen as baseline since it was shown to outperform other clustering techniques in discovering more accurate clusters from large corpora of well-written documents, such as newspaper articles. FIHC is also scalable for large datasets [167, 168, 170], and similar to ClustText, it discovers labeled clusters. FIHC can be freely downloaded from [199]. For our experiments, we found that implementing our own FIHC version, as described in [167], was more convenient.

To ensure a fair comparison with ClustText, we evaluated FIHC on the same corpus of 2115 documents, which were obtained as described in 7.61. FIHC started by a pre-processing the input documents. As mentioned earlier, this involved discarding stopwords, and stemming all the other words. Next, the tf-idf scores of words were computed. In the actual implementation of FIHC, as described in [167], the authors do not mention the exact number of words that were used as features. Subsequently, we decided to select the top-n words as features based on their tf-idf scores. In our experiments, we set n to $|\phi_2|$=492 to correspond to the same number of features that yielded the best performance for ClustText. That is, the top-492 words were selected based on their tf-idf scores, and used as features by FIHC to cluster our corpus. Then, we estimated the accuracy of FIHC using equation (7.9) for the 3 desired clusterings of 3, 30 and 60 clusters. This was performed by our 2 judges. We ensured the reliability of our results by computing the Spearman rank correlation between the judges. The correlation values between the 2 judges when evaluating the topic_significance for our clustering of 3, 30 and 60 clusters were in the range of 0.71-0.80. The corresponding correlation values between the judges for the cluster_precision were in the range of 0.67-0.73. These correlation scores indicated a relatively high level of agreement between the two judges, and thus, our evaluation can be considered reliable.

The topic_significance (TS), cluster_precision (CS) and accuracy scores achieved by FIHC is shown in Table 7.13.We also repeat the (best) scores of ClustText (using ϕ_2 features) for ease of comparison.

Num. of Clusters	FIHC			ClustText		
	TS	**CS**	**Accuracy**	**TS**	**CS**	**Accuracy**
3	0.77	0.78	0.60	0.84	0.84	0.71
30	0.79	0.80	0.63	0.85	0.86	0.73
60	0.81	0.83	0.67	0.88	0.88	0.77
Avg.	0.79	0.80	0.63	0.86	0.86	0.74

Table 7.13: Accuracy scores of FIHC (Baseline) vs. ClustText.

From the results, it can be seen that our proposed ClustText framework was more accurate than the FIHC baseline, as reflected in its larger accuracy score. Specifically, ClustText discovered more precise clusters of documents, as indicated by its higher cluster_precision, and generated more meaningful labels for these clusters, as indicated by its higher topic_significance.

To investigate the lower performance of FIHC, we inspected its clusters and made 3 main observations. We will elaborate upon them, and compare these results (of FIHC) with those of our proposed ClustText framework, which were obtained in the preceding section.

Issue 1: Irrelevant, Unintelligible Topic Labels

Many of the cluster labels extracted by FIHC denoted topics that were either irrelevant to our corpus (domain) or were unintelligible word sequences. Typical examples of such labels included {"meeting, Friday"} and {"cable, training, center, unknown"}. These results indicate that the tf-idf technique, employed by FIHC, detected fewer meaningful features from our sparse, informally-written corpus (Challenge 1). Instead, it identified a

large number of less reliable features, which were subsequently extracted as invalid cluster labels (wordsets), as illustrated above. These labels were then awarded low topic_significance scores during our evaluations.

Conversely, the cluster labels extracted by ClustText provided meaningful descriptions of topics, which were closely related to our corpus. These descriptive topic labels were awarded much higher topic_significance scores than those extracted by FIHC. An example of such a label was "{cable, connection, replace}", which described the topic of "replacement of connection cables" in our corpus. These results illustrate that our proposed feature selection mechanism accurately identified the most informative features from our domain-specific texts (Contribution 1), and overcame the challenges posed by the informal language constructs and sparsity that characterized these texts.

Issue 2: Cluster Conflicts
The features extracted by tf-idf from our corpus were less discriminative, and did not allow FIHC to precisely distinguish between different document sub-groups. We observed that these irrelevant features were responsible for the issue of cluster conflicts (Challenge 2). They occurred frequently, and appeared recurrently in many clusters and documents. It was therefore possible that a single document had the largest number of features in common with more than 1 cluster, i.e. the document maximized the intra-cluster similarity of multiple clusters. FIHC's clustering strategy, which focused on optimizing the intra-cluster similarity, then identified these (multiple) clusters as being most suitable for the (single) document. This led to the issue of cluster conflicts, where a single document could be assigned to more than one cluster. These clusters were awarded relatively low cluster_precision scores during our evaluations. We shall see in Issue 3 that the formation of skewed clusters by FIHC could also be attributed to its clustering strategy, which focused on the intra-cluster similarity, and to its irrelevant features.

ClustText, on the other hand, relied on more informative features, discovered by our feature identification mechanism, which enabled it to precisely discriminate between different sub-groups of documents. Furthermore, the clustering strategy employed in ClustText (Section 7.5.4) did not solely take into account whether a document and a cluster shared the maximum number of features, i.e. optimizing (maximizing) the intra-cluster similarity. Instead, ClustText also optimized (minimized) the inter-cluster similarity by considering whether the document had the least number of features in common with other clusters. Since it was highly unlikely for a single document to optimize both the intra-cluster similarity and inter-cluster similarity of more than 1 cluster, our strategy effectively mitigated the issue of cluster conflicts (Contribution 2). It enabled ClustText to precisely identify the most appropriate cluster for each document. Consequently, the clusters discovered by ClustText were of higher quality than those of the FIHC baseline, and they were awarded higher cluster_precision scores.

Issue 3: Skewed Clusters
Some of the clusters discovered by FIHC were much larger than others. As we described before, this occurred because the non-discriminative features that FIHC identified from our domain-specific corpus appeared frequently in many documents. They behaved as common features shared by the documents, which resulted in a relatively high, but fallacious, similarity between these documents. The clustering strategy of FIHC, which focused on optimizing the intra-cluster similarity of a cluster, then assigned these large numbers of highly similar documents to a single cluster. Subsequently, the size of the cluster increased considerably. In turn, the clustered documents started to attract other documents to the cluster if they shared some features, even if they dealt with different topics. This led to the formation of skewed clusters of disparate sizes (Challenge 3). For example, in our clustering of 30 clusters, around 90% (1900) of all (2215) documents were assigned to only 6 clusters. The remaining 10% (215 documents) were then distributed across the remaining 24 clusters. Thus, we had 6 extremely large clusters with their sizes averaging 1900/6= (approx.) 317 documents per cluster, and 24 other much smaller clusters with an average size of 215/24=9 documents per cluster. These skewed clusters were inaccurate, and were awarded low cluster_precision scores during our evaluations.

Conversely, as we saw earlier, the clustering strategy of ClustText was stricter. It did not solely optimize the intra-cluster similarity by considering whether the documents and a

given cluster shared the largest number of features. It also optimized the inter-cluster similarity by determining whether the documents had the least number of features in common with other clusters. Since it was highly unlikely for an unreasonably large number of different documents to optimize both the intra-cluster similarity and the inter-cluster similarity of a single cluster, this strategy inhibited the formation of skewed clusters (Contribution 3). That is, the documents were more evenly distributed. The clusters of ClustText were consequently more accurate, and were awarded higher cluster_precision scores.

So far, we have empirically demonstrated that

- Our feature identification technique identified informative features from our informally-written and sparse texts. It played an important role in accurately grouping these texts into corresponding clusters, and in generating meaningful labels for the clusters (Contribution 1).
- The clustering strategy employed by ClustText, which optimized both the intra-cluster similarity and the inter-cluster similarity, addressed the issues of cluster conflicts and of skewed clustering (Contributions 2 and 3).

In the next section, we will investigate the scalability of ClustText to determine whether it can efficiently cluster large datasets.

7.6.6 ANALYZING THE SCALABILITY

To analyze the scalability of our ClustText framework, we adopted an approach similar to that employed by Fung et al. [167] and Chen et al. [187]. It involved duplicating the documents in a small corpus in order to create a larger corpus with more documents. The time taken by a clustering algorithm to cluster the resulting larger corpus was then measured. The latter two steps of replicating the documents to increase the corpus size, and of measuring the clustering time were repeated until a corpus of maximum size N was obtained. In the aforementioned studies, N was set to 100,000

Therefore, starting with our corpus of size 2115, we repeatedly duplicated (i.e. doubled) its documents until a maximum corpus size of 135,360 documents was reached. The value of 135,360 documents was chosen as it was approximately equal to the 100,000 employed by previous studies.

The different corpora thus obtained are presented in the 2nd column of Table 7.14. Next, both ClustText and FIHC were applied over each of these corpora. The number of clusters was fixed to 30 (by setting the minsup value) for both algorithms. Their time taken to produce a clustering solution was then measured, as shown respectively in the 3rd and 4th columns of Table 7.14. The results are also presented graphically in Figure 7.9.

Corpus	Size	Clustering Time (ms)	
		ClustText	FIHC
1	2115	3450	3390
2	4230	4989	5679
3	8460	5612	6578
4	16,920	7453	8213
5	33,840	9345	10,345
6	67,680	11,509	12,393
7	135,360	13,764	14,859

Table 7.14: Clustering time of ClustText vs. FIHC.

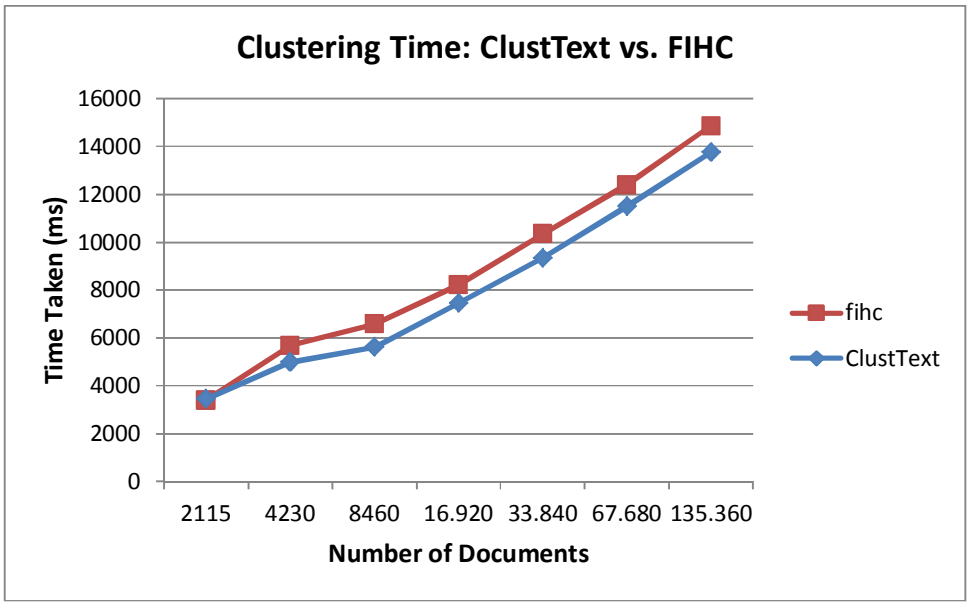


Figure 7.9: Clustering time of ClustText vs. FIHC.

The above results indicate that the scalability of ClustText was comparable to that of FIHC, if not, marginally better. It can also be seen from Figure 7.9 that the run-time of ClustText (and of FIHC) increases almost linearly with the corpus size. Since FIHC has been shown to run twice as fast as state-of-the-art techniques like bisecting K-means over corpora of different sizes [167, 168, 170], our results suggest that ClustText is also scalable.

One caveat with our procedure for evaluating the scalability by duplicating documents is that it focuses solely on the run-time. It does not take into account the accuracy of the algorithms. Larger corpora, consisting of distinct documents (not of duplicated ones) will usually contain a larger number of irrelevant features. Therefore, even if an algorithm is shown to be scalable over these corpora, their accuracy can, in fact, have deteriorated.

7.7 Conclusion

In this chapter, we were concerned with our third research question, *RQ3: How to discover cohesive and homogeneous clusters of similar documents from a collection of domain-specific, sparse and informally-written collection of corporate texts?*

We addressed this question by developing the ClustText framework for text clustering. The clustering strategy that we adopted in ClustText was based on the relatively new approach of exploiting sets of words (features) co-occurring frequently in the documents of a corpus. The fundamental idea in this approach is to cluster together documents that share a large number of wordsets since they presumably deal with similar topics. Wordset-based algorithms have been shown to be more efficient than traditional techniques like K-means for clustering unstructured text data.

One of the crucial aspects in our clustering approach was the identification of informative features. Feature identification was critical in ClustText for two main reasons. First, the use of informative features enabled us to precisely discriminate between different sub-groups of documents, and to subsequently discover accurate, non-overlapping document clusters. Irrelevant features, on the other hand, would have had an adverse effect on the formation of high quality clusters by blurring the distinction between document sub-groups. Second, we used the features to generate wordsets, which described the main topics latent in the corpus. The wordsets were then exploited as topic labels of clusters. Hence, to obtain meaningful cluster labels, it was essential for us to first acquire the most pertinent and

useful features from the corpus. The need for informative features was even more pressing given our informally-written and sparse domain-specific texts. Improperly selected features would have resulted in incoherent cluster labels. As we mentioned earlier, these invalid cluster labels are unsuitable for use in real-life applications since they can be misleading to users. Furthermore, users often decide whether a cluster is worth inspecting further based on its labels. Inappropriately labeled clusters will therefore be overlooked by users if even they contain the documents being searched for.

In ClustText, we relied on two types of features. The first type consisted of terms. The main benefit of using terms as features was that they were informative since they designated domain-specific concepts. Also, they were realized both as single and multi-word expressions. This allowed us to have multi-word expressions as labels, which are more meaningful than the single-word labels of existing wordset-based algorithms. Our second type of features consisted of context-words that appeared with the terms in the documents. The next difficulty we faced was to identify which of these context-words were the most informative and could qualify as feature. We found out that conventional techniques, such as tf-idf, which have been commonly employed in other wordset-based algorithms, could not identify the most discriminative features from our domain-specific texts. To overcome this challenge, we developed a novel feature identification mechanism. It was hinged on a hybrid strategy, and combined both a filter and wrapper methods. With such a hybrid strategy, we were able to leverage upon the high accuracy levels of wrapper methods, without suffering from their high computational costs. For example, our wrapper method did not have to search through all the possible context-words combinations to identify the most promising features. Instead, it simply selected the most promising context-words based on their scores, computed by the filter method. We then used the selected context-words and their corresponding terms as features. To discover frequent wordsets, i.e. sets of features that appeared frequently in our documents, we employed the Apriori algorithm. As already described, these wordsets conveniently served as cluster labels, and enabled us to overcome the longstanding challenge of cluster labeling that plagues traditional clustering algorithms like K-means. To assign documents to the clusters, we relied on three criteria. The first one was that a document dealt with the same topic as indicated by the cluster's label. The second one was that the assignment of the document to the cluster maximized the intra-cluster similarity. This criterion was important to ensure the formation of homogeneous clusters. The last criterion, related to the previous one, was that the assignment of the document to the cluster also minimized (did not deteriorate) the overlap between the cluster and all other clusters. The latter criterion was important for the formation of distinct (non-overlapping) clusters. After populating the clusters, ClustText arranged them in a hierarchy or a topic tree. This was done by exploiting the monotonicity property of the wordsets (cluster labels). The clusters were arranged such that a child cluster dealt with a more specialized topic that its parent. Such a topic tree is useful for user browsing and navigation.

We evaluated ClustText on a collection of real-life, domain-specific texts provided by our industrial partners. We also compared it against the FIHC algorithm, a state-of-the-art wordset-based clustering technique. The experimental evaluations showed that our novel feature identification technique was able to extract more meaningful features from the domain-specific texts than the tf-idf technique employed by FIHC. Subsequently, the cluster labels discovered by ClustText were more informative and useful for practical applications than those of FIHC. We also found out that the cluster population strategy of ClustText, which optimized both the intra-cluster and inter-cluster similarity, led to the formation of precise clusters of similar documents. Conversely, FIHC, which only optimized only the intra-cluster similarity, was often unable to identify the most precise clusters for its documents, resulting in the issue of conflicting clusters. This issue was compounded by the lack of reliable features in FIHC. Also, with ClustText, the documents were more evenly distributed in the clusters. Conversely, the clustering solution discovered by FIHC from the domain-specific corpus was skewed, and consisted of clusters with extremely disparate sizes, for example, a few very large clusters and many much smaller clusters. Consequently, ClustText outperformed the FIHC baseline. It discovered more precise, non-overlapping and appropriately-labeled clusters from the domain-specific collection than FIHC.

The main findings/conclusions of this chapter can be summarized as follows:

1. The identification of the most pertinent features is an essential step towards the discovery of precise clusters. For techniques that generate labeled clusters, such as ClustText and FIHC, the features are also important for the formation of meaningful labels.
2. Traditional techniques like tf-idf are often inadequate for selecting informative features from domain-specific texts, which are sparse and informally-written. Feature identifications from these texts often require more sophisticated mechanisms.
3. Although wrapper and filter methods are originally designed for feature selection in supervised learning applications (e.g. classification), they can also be adapted for unsupervised applications (e.g. clustering).
4. Wordset-based algorithms present several characteristics that make them particularly attractive for clustering texts. One of their main strength is that they generate labeled clusters. Also, they are scalable and efficient since they estimate document similarity by comparing low dimensional, compact set of features (wordsets). This is unlike traditional algorithms like K-means, which have to compare high dimensional and sparse vectors, which compromises their scalability for large datasets.
5. To prevent cluster conflicts and the formation of skewed clusters, it is important that a clustering strategy attempts to maximize the intra-cluster similarity and to minimize the inter-cluster similarity.
6. Traditional cluster evaluation metrics are not always suitable for labeled clusters. They do not take into account whether the labels are meaningful (i.e. relevant to the domain) and whether the documents in a cluster are indeed related to the cluster's label.

We will discuss the results of our text clustering framework with respect to RQ3 in Chapter 8.

Concerning the business applications, ClustText is useful as a decision support tool to identify similarity between product failures. This information can then be exploited to improve product quality. It can also be applied to identify similar repair actions to enable managers in determining whether engineers are following the stipulated guidelines when repairing/maintaining products. In addition, ClustText can be employed for customer segmentation to support targeted advertisement and product recommendation.

PART V - CONCLUSION

CHAPTER 8 – SUMMARY AND DISUSSIONS

8.1 BACKGROUND

The main research question (RQ) that we addressed in this dissertation, as formulated in Chapter 1 was
"How to automatically extract meaningful information from sparse, domain-specific and informally-written corporate texts?

This question was motivated by two main factors. The first factor, with an industrial/corporate underpinning, pertained to the proliferation of unstructured text data, which has resulted in a drastic shift in the information space of business organizations. According to various studies, the traditional structured data, such as sales transactions in databases and data warehouses, currently account for a (relatively) insignificant 20% of all corporate data. Conversely, an overwhelming 80% of corporate data is in unstructured text format, such as customer complaints and repair notes of service engineers. Corporate organizations have also become increasingly aware that buried within these massive amounts of unstructured texts are meaningful information nuggets, which are valuable for supporting a wide-range of Business Intelligence activities, leading to products of better quality and more satisfied customers.

The second factor that motivated our main RQ had an academic underpinning, and was the main focus of this thesis. It pertained to the challenges involved in identifying meaningful information from corporate texts. Several Natural Language Processing (NLP) algorithms have been developed for discovering meaningful information from texts. Most of the existing NLP techniques developed to date have been applied to large collections of well-written texts. These texts provide reliable linguistic and statistical evidence, which makes it easier for NLP algorithms to identify meaningful information items from their contents. However, texts generated from specialized domains, such as Product Development-Customer Service, pose a set of new challenges that extant NLP algorithms fail to adequately address. For example, these domain-specific texts are informally-written, and reliable linguistic features cannot be obtained from their contents. Also, they are sparse, and do not offer substantial statistical evidence. Due to the lack of reliable linguistic/statistical features, most NLP algorithms exhibit various shortcomings when applied to these domain-specific texts, and are inaccurate for extracting meaningful information from their contents. The lack of reliable features is further compounded by the scarcity of domain-specific knowledge resources, such as ontologies, which could have been exploited to facilitate the detection of meaningful information from texts.

The above two factors highlighted the need for novel NLP techniques that successfully overcome the challenges posed by domain-specific corporate documents. They established our objectives with this research. We had two main objectives. Our first objective, with an industrial underpinning, was to provide business organizations with our algorithms and corresponding prototypes so that they could use them to unlock the valuable corporate information, buried in the massive amount of texts. Our second objective, which was the main aim of this thesis, had an academic underpinning. It was to fill the lacuna in extant NLP research by developing state-of-the-art techniques, which successfully alleviated the limitations of existing ones and addressed the challenges involved in extracting meaningful information from sparse, informally-written domain-specific texts.

To realize these objectives and to answer our main RQ, we identified 3 main components of *meaningful information*, namely terms, semantic relations and clusters of similar documents. From these 3 components, we formulated 3 more specific research questions, namely, RQ1, RQ2, and RQ3, as described below. Together, the solution to each of these specific questions constituted our answer to the main research question, presented earlier.

RQ1: How to accurately identify and extract terms from domain-specific, sparse and informally-written corporate texts?

RQ2: How to accurately detect occurrences of semantic relations from domain-specific, sparse and informally-written corporate texts?

RQ3: How to discover cohesive and homogeneous clusters of similar documents from a collection of domain-specific, sparse and informally-written corporate texts?

The remainder of this chapter is organized as follows. In Section 8.2, we will summarize our proposed solutions, presented in Chapters 3-7 of this thesis, in response to the aforementioned challenges and the 3 RQs. In our discussion, we will also describe how successful our solutions have been in addressing these RQs and the academic objective, and we will highlight areas of potential improvements. Our industrial objective will be analyzed in Section 8.3. Then, in Section 8.4 we present future research avenues.

8.2 ACADEMIC OBJECTIVE

8.2.1 RESEARCH QUESTION 1 (RQ1) - TERM EXTRACTION

In Chapter 2, we addressed RQ1. We identified 5 main challenges that existing algorithms are unable to adequately address, and that hinder the accurate extraction of terms from domain-specific texts. First, domain-specific documents are sparse and usually contain many rare (i.e. low frequency) terms, which are difficult to detect due to the lack of significant statistical evidence. Second, these documents are often informally-written, which makes it difficult to identify terms based on linguistic evidence, such as by using Part-of-Speech (POS) filters. The third challenge is that of detecting multi-word terms, containing three or more words, which are commonly employed in technical domains, such as PD-CS. The fourth challenge, related to the previous one, is to precisely discriminate between valid multi-word terms and invalid word sequences. The fifth challenge is the unavailability of domain-specific knowledge resources like ontologies to support the identification of terms from documents.

To overcome these challenges and address RQ1, we developed the ExtTerm term extraction framework. Our framework is unsupervised in that its analyses do not rely on additional information from domain-specific knowledge resources like ontologies, which are scarce in corporate domains like PD-CS. To detect rare terms, we exploit the termhood property, which is the degree to which an expression is related to a domain, regardless of its occurrence frequency. We compute the termhood of an expression by devising a corpus-comparison strategy. It involves estimating the expression's relative probability of occurring in our domain-specific texts and in a general corpus. In this way, an expression with a low frequency in the domain-specific texts will still be awarded a high termhood if it was found to be less likely in the general corpus (i.e. it was likelier in the domain-specific texts than in the general corpus). These domain-specific expressions with high termhood can then be selected as terms (candidates). To ensure the accuracy of the corpus-comparison approach, the two corpora have to exhibit comparable and contrastive characteristics. In ExtTerm, we used Wikipedia as the general corpus for comparing our domain-specific texts. To deal with the issue of informal language constructs, we define a linguistic filter that accommodates POS-tagging errors. The filter design is also hinged on the term formation process, which enables it to capture terms with complex syntactic structures. Concerning the detection of multi-word terms, we develop a novel algorithm for determining the unithood of expressions of arbitrary lengths. The crux of the algorithm lies in decomposing an expression with n words into a number of sub-expressions of size 2…(n-1). The unithood scores of these sub-expressions are recursively calculated, starting with those of size 2 whose unithood can be computed using traditional Lexical Association

Measures (LAMs). In ExtTerm, we used the cube mutual information as LAM since it achieved the best performance on our domain-specific corpus in preliminary experiments. Then, the unithood of the expression of size n is obtained by aggregating the unithood of its sub-expressions. Such a strategy ensured that valid terms were awarded much higher unithood scores than invalid ones. Thus, besides the detection of multi-word terms, our approach was also useful to identify and reject invalid word sequences.

Results for RQ1
We evaluated the performance of the proposed ExtTerm framework on a real-life corpus of informally-written and sparse texts. To determine how successful ExtTerm is in extracting domain-specific terms, we measured its accuracy using the performance measures of precision, recall and F1. The maximum precision, recall and F1 scores achieved by ExtTerm, as reported in our experiments, were respectively of 0.87, 0.89 and 0.88.

To determine whether ExtTerm effectively addresses the challenges posed by domain-specific texts and overcomes the limitations of extant term extraction algorithms, we also evaluated the performance of the C-value algorithm on the same corpus. The corresponding precision, recall and F1 scores achieved by C-value were respectively of 0.71, 0.84, and 0.77. A closer inspection of the results of ExtTerm and C-value revealed the following. The linguistic filter of C-value incorrectly rejected many of the valid terms that had complex syntactic structures. This filter was also susceptible to POS-tagging errors in the domain-specific texts, and detected term fragments instead of the entire terms. Conversely, the filter of ExtTerm was able to accurately detect these complex terms and to circumvent POS-tagging errors. The C-value baseline also missed the low frequency terms in the corpus, especially those occurring less than three times. ExtTerm, on the other hand, was equally accurate in detecting terms of different frequencies, including those occurring very rarely (e.g. twice) in the entire corpus. In addition, C-value could not precisely discriminate between valid multi-word terms and other invalid multi-word expressions, while the performance of ExtTerm was relatively stable in extracting terms of different lengths, including those with more than 3 words.

Our experimental results and observations suggest that our approach accurately extracts terms from sparse, informally-written domain-specific texts. It effectively addresses the challenges posed by such texts and overcomes the limitations of existing techniques, thereby answering our first research question, RQ1.

8.2.2 Research Question 2 (RQ2) - Relation Extraction

Prior to actually addressing RQ2, we provided an overview of the basic notions pertaining to semantic relations in Chapter 3. We defined a semantic relation as a triple, consisting of a pair of terms and a pattern that connects the pair in a document. For example, the triple <"consist of ", "engine", "car"> is made up of the pattern *"consist of"*, which establishes a part-whole relation between the (part) "engine" and the (whole) "car", as in "car *consists of* engine". In Chapter 3, we also motivated our interest in two types of semantic relations, namely part-whole, between parts and wholes, and causal, between causes and their effects. Our main justification in focusing on these two specific relations was that they express valuable information, which can be useful for PD-CS organizations to improve product quality and to enhance customer satisfaction.

In Chapter 4 we addressed RQ2, and investigated the extraction of part-whole relations from domain-specific texts. We adopted a minimally-supervised approach. Unlike supervised approaches, it eschewed the need for domain-specific resources (e.g. annotated training data), which are scarce in most specialized disciplines. Our algorithm was initialized with a handful of term pairs, called seeds, that instantiate the target relation (i.e. part-whole). Then, it employed a recursive learning procedure for extracting patterns connecting the pairs and vice-versa. We identified 4 main shortcomings of minimally-supervised techniques, which are further exacerbated when they are applied to domain-specific texts, and hinder the accurate extraction of (part-whole) relations. The first challenge is the sparse characteristics of domain-specific documents, which is detrimental to the statistical analyses of minimally-supervised algorithms, and may affect the precision. Data sparsity also compromises the recall (coverage) since a sparse corpus is less likely to contain the wide variety of patterns that realize part-whole relations. The second issue is that of selecting fertile seeds from domain-specific texts, which requires proficiency in the domain terminology. The third challenge is that of semantic-drift, especially when dealing

with relations like part-whole, which are realized by a large number of ambiguous patterns. These ambiguous patterns promote the extraction of invalid part-whole pairs, which pollute the recursive learning procedure of minimally-supervised techniques. As a result, the extracted relations will be different from the intended relation. The fourth challenge concerns the limitation of the surface-strings, which are commonly employed by minimally-supervised techniques to represent patterns. These strings are susceptible to word order and morphological variations, and lead to the extraction of a large number of redundant (similar) patterns. They also cannot detect related instances/patterns that do not appear in adjacent positions in the surface texts, i.e. they do not capture long-range dependencies.

To overcome these difficulties and address RQ2, we developed a novel framework for extracting (part-whole) relations. Instead of extracting part-whole relations directly from the domain-specific, sparse texts, we first acquire a set of patterns that reliably express these relations from a knowledge-base. As knowledge-base, we considered two options, namely the British National Corpus (BNC) and the (English) Wikipedia collection. We compared the key characteristics of these two resources, and concluded that Wikipedia was more attractive than BNC as a knowledge-base. Being an encyclopedic resource, it has a broader coverage, and hence is likelier to contain the large variety of patterns that express part-whole relations. Wikipedia is larger in size, and provides us with more reliable statistical evidence to overcome the data sparsity issue posed by domain-specific texts. A larger number of prototypical part-whole pairs, to be used as seeds for initializing our algorithm, can also be found from Wikipedia. To mitigate the issue of semantic-drift, we define mechanism for ensuring that valid part-whole pairs are awarded much higher reliability scores than invalid ones. The invalid pairs can then be discarded, and prevented from triggering semantic-drift. On the other hand, the valid part-whole pairs are bootstrapped, and used to harvest reliable part-whole patterns, which is beneficial to the performance. We represent the part-whole patterns extracted from Wikipedia using lexico-syntactic information, derived from the parse-trees of corresponding sentences. Such a representation prevents the generation of a large number of redundant patterns (e.g. due to word order variations or inflections), and identifies related pairs even if they do not appear in adjacent positions in surface-texts. Thus, our lexico-syntactic patterns overcome the shortcomings of the traditional surface-strings employed by most minimally-supervised techniques. We then use the reliable patterns harvested from our Wikipedia knowledge-base in order to extract part-whole relations from the domain-specific texts. Each relation was encoded a triple, consisting of a pattern from the knowledge-base and a pair of domain-specific terms identified from the domain-specific corpus by the ExtTerm framework (Chapter 2).

Chapter 5 was also concerned with RQ2. We investigated the extraction of causal relations from domain-specific texts. The challenges involved with causal relations were the same as those we faced when learning part-whole relations. These include the issues of data sparsity and semantic-drift. An additional difficulty is that of detecting causal relations that are implicitly expressed by verbal and non-verbal patterns. These implicit patterns are usually devoid of any causal connotation, such as the verb *"reduce"* or the adjectival phrase *"due to"*. Most existing techniques have targeted the extraction of explicit causal relations, which are expressed by the verb *"to cause"* and its synonyms. To address the challenges, we relied on our relation extraction approach that we presented earlier for mining part-whole relations. Thus, this study also served as a test-bed for evaluating whether our proposed approach is generic.

Results for RQ2

As before, we determined how successful our approach was in extracting semantic relations by computing its accuracy over the same domain-specific (real-life, informally-written and sparse) corpus, which we used in term extraction. The precision, recall, and F1 scores achieved in leaning part-whole relations were respectively 0.79, 0.80 and 0.79. The corresponding scores achieved when extracting causal relations were 0.77, 0.82 and 0.79.

We also compared the performance of our approach in extracting part-whole relations with the Espresso state-of-the-art minimally-supervised technique. The corresponding precision, recall and F1 scores of Espresso were respectively 0.44, 0.45, and 0.44. The lower performance of Espresso was attributed to the issues of data sparsity and semantic-drift. Reliable part-whole patterns, such as *"contain"*, did not occur very frequently in the

domain-specific texts, and in the absence of statistical evidence, these patterns were discarded by Espresso. Instead, ambiguous patterns, which did not always express part-whole relations, were more frequent, and were incorrectly selected. These patterns then introduced invalid part-whole pairs, which in turn, triggered the issue of semantic-drift, and deteriorated the performance. Our approach, on the other hand, leveraged upon Wikipedia as a knowledge-base for overcoming the issue of data sparsity, and also effectively mitigated the effect of semantic-drift by detecting and rejecting invalid part-whole pairs that were introduced by ambiguous part-whole patterns. We observed that Espresso extracted a large number of redundant surface-patterns, with word order or morphological variations (e.g. *"consisted of"*, *"consists of"*), which further exacerbated the issue of data sparsity. The lexico-syntactic pattern representation that we employed in our relation extraction approach overcame these limitations.

Our experimental results and observations suggest that our proposed approach accurately extracts semantic relations from sparse, informally-written domain-specific texts. It effectively addresses the challenges posed by such texts and overcomes the limitations of existing techniques, thereby answering our second research question, RQ2. We also showed that our approach was generic, and could be applied to mine different types of relations, including part-whole and causality.

We performed additional experiments in Chapter 6, which was devoted to studying the behavior of minimally-supervised algorithms; in particular, we were interested on the influence of the initializing seeds on the algorithms' performance. These experiments provided us with valuable results that can be used to improve the performance of minimally-supervised techniques in future. We focused on the part-whole relations since they have been precisely classified into different (sub)types in various taxonomies. The seeds (instance pairs) that can participate in each type are also defined in these taxonomies. In our work, we relied on the formal taxonomy of Keet and Artale [114], which lists 8 types of part-whole relations. Four of these subtypes are transitive and are known as mereological, while the remaining are intransitive and are known as meronymic. An example of a mereological relation is the *"structural part-of"* relation, such as between "engine" and "car". An example of a meronymic relation is the *"member-of"* relation, such as between "player" and "team". In our experiments, we defined distinct sets of seeds for each of the relation types, and used these seed sets to initialize our minimally-supervised algorithm. We also followed the conventional practice of defining a single set that mixed seeds of different types. We had employed such a mixed seed set in our earlier experiments reported in Chapter 4. We observed that using mereological seeds yielded the most precise relations. Also, we were still able to discover relations of different types even if the initializing seeds belonged to one particular type. Finally, we saw that the results obtained with the traditional practice of using a single set of mixed seeds could not be accurately predicted. Furthermore, with the mixed set, we could not detect many of the specialized patterns that were peculiar to some types of part-whole relations. We will discuss the implications of these findings when describing our areas of future work in Section 8.4.

8.2.3 Research Question 3 (RQ3) - Text Clustering

In Chapter 7 we addressed RQ3, and investigated the discovery of document clusters from domain-specific texts. We formulated three main desirable characteristics of text clustering algorithms, namely 1) assigning labels to the clusters, which overcomes the need for manually inspecting the clusters' contents to determine what they are about; 2) ensuring scalability since comparing the high dimensional but sparse document vectors to estimate document similarity is inefficient, and 3) eschewing the need for defining initial clusters, required by some algorithms as an input parameter. Then, we introduced wordset-based clustering algorithms, which compared to traditional techniques like K-means, satisfy the specific requirements of text clustering, listed above. A wordset is a set of features (e.g. terms or words) that appear frequently in the documents of a corpus, and hence, can be thought of as describing a prominent topic in the corpus. The basic idea underlying wordset-based algorithms is that documents sharing a larger number wordsets are likely to deal with the same topic, and thus, can be assigned to the same cluster. Then, we saw that wordset-based algorithms, despite their strengths, exhibit 3 main shortcomings, especially when dealing with sparse, informally-written domain-specific texts. The first challenge is that of feature identification. Domain-specific texts often do not explicitly provide reliable features, which can be readily acquired from their contents using

conventional techniques like tf-idf. We considered the identification of informative features to be crucial for wordset-based techniques. This is because the features are used by these algorithms to generate wordsets, which then serve as labels for clusters. Therefore, the generation of meaningful cluster labels, entails the prior identification of the most promising features. The need for reliable features is even more pressing given our informally-written (and sparse) documents since improperly selected features can result in incoherent or irrelevant cluster labels (wordsets). Detecting meaningful features is also important for precisely discriminating between sub-groups of similar documents, leading to the formation of more accurate non-overlapping clusters. The second challenge of wordset-based algorithms is that of resolving cluster conflicts and precisely identifying the single most appropriate cluster for a given document. The third challenge, related to the previous one, is that of preventing the formation of skewed clusters with extremely disparate sizes.

To address these challenges and answer RQ3, we developed ClustText, a wordset-based technique for text clustering. In ClustText, we identify two types of features for clustering our documents. The first type consists of terms, which provide us with meaningful information on pertinent domain concepts. A benefit of using terms as features is that they are realized as multi-word expressions (in addition to single-word expressions), which allows us to have multi-word features. These multi-word features, such as "filament control board" are more informative as labels than those consisting of single words, such as "filament", "control" and "board", which are used by extant wordset-based techniques. Terms were straightforwardly available from the output of the ExtTerm framework (Chapter 2). However, terms alone, despite being informative, may be insufficient to precisely cluster our documents. Also, they are predominantly realized as noun phrases. Therefore, to supplement terms, we use, as our second type of features, the context-words, which co-occur in the same documents as the terms. To determine which of those context-words are informative enough to qualify as features, we develop a novel feature selection mechanism. Our mechanism is a hybrid one, incorporating both a filter and a wrapper method. Since these methods are designed for supervised learning tasks (e.g. classification), we adapt them (and our data) for our (unsupervised) clustering application. The main strength of our hybrid feature selection technique is that it provides us with the high accuracy levels of wrapper methods without suffering from their high computational costs. For example, our wrapper method does not have to search through the entire subset space of context-words, and to evaluate the performance of each context-word combination in order to determine which one yields the optimal performance. Instead, it only selects those context-words whose relevancy scores, as computed by the filter method, exceed a threshold. The set of context-words (and terms) that yield the best performance is then selected as features. After feature identification, we apply the Apriori algorithm on our corpus to discover frequent wordsets. As already discussed, a wordset is a set of features (terms and/or context-words) that occur in at least a minimum fraction of documents (i.e. frequently) in the corpus. Since each wordset describes a recurrent topic in the corpus, we treat them as labels of initially empty clusters. In this way, we overcome the issue of cluster labeling that plagues traditional clustering techniques like K-means. To populate the clusters, we devise a strategy that assigns a document to a cluster only if such an assignment satisfies three criteria. First, the document and cluster (label) deal with the same topic; second, the document exhibits maximum similarity with other documents already in the cluster, i.e. we maximize the intra-cluster similarity; and third, assigning the document to the cluster does not deteriorate its overlap with the other clusters, i.e. we minimize the inter-cluster similarity. Most existing wordset-based algorithms satisfy only the first two criteria. As demonstrated in our experiments, the strategy employed by existing algorithms is responsible for the issues of cluster conflicts and skewed clusters. Incorporating the additional (third) criterion makes the clustering technique of ClustText stricter and more accurate. It plays an important role in resolving cluster conflicts and in inhibiting the formation of skewed clusters.

Results for RQ3

We determined how successful our approach was in discovering clusters by computing its accuracy over our domain-specific (real-life, informally-written and sparse) corpus. Traditional cluster evaluation measures such as the F1 and the purity are not designed for labeled clusters; they only take into account the accuracy of the clustered contents. In our case, we also had to gauge the quality of the clusters' labels in addition to their contents. As we discussed before, we considered the quality of the labels to be as important as the

quality of the clusters' contents. Incorrectly labeled clusters can be misleading to users. Also, users will often decide whether it is worthwhile to inspect a cluster based on its label. Thus, incorrectly labeled clusters will be skipped by users even if these clusters contain the documents of interest.

Therefore, our clustering accuracy measure was made up of 2 components, viz. topic_significance, which gives the relevancy of the cluster labels, and the cluster_precision, which gives the precision of the clustered contents (i.e. the degree to which the cluster's documents are related to its topic label). The (overall) accuracy of a clustering solution was then the product of the topic_significance and the cluster_precision of all the clusters (in the clustering). The average scores achieved by our ClustText framework over the domain-specific texts were of 0.86 for the topic_significance, 0.86 for the cluster_precision and 0.74 (i.e. 0.86 x 0.86) for the (overall) accuracy.

As a baseline to benchmark the performance of ClustText, we relied on the Frequent Itemset-based Hierarchical Clustering (FIHC) algorithm, which is a state-of-the-art wordset-based technique. The topic_significance, cluster_precision and accuracy scores achieved by FIHC were respectively 0.79, 0.80 and 0.63. In our experiments, we observed that our feature identification mechanism, employed by ClustText, detected more meaningful (i.e. coherent and relevant) features than the traditional tf-idf technique employed by FIHC. As a result, the labels discovered by ClustText were more informative than those of FIHC. Also, the clustering strategy of ClustText, which involves optimizing both the intra-cluster similarity and the inter-cluster similarity, led to the formation of more precise and non-overlapping clusters. Concerning FIHC, we noted that it was often unable to detect the single most appropriate cluster for many documents. Thus, it suffered from the issue of conflicting clusters. This was compounded by the lack of meaningful features for distinguishing documents of different clusters. In addition, with ClustText, the documents were more evenly distributed in the clusters, whereas the clustering solution produced by FIHC was skewed, consisting of a few very large clusters and many small clusters.

Our experimental results and observations suggest that our proposed approach accurately discovers non-overlapping clusters of similar documents from sparse, informally-written domain-specific texts. It effectively addresses the challenges posed by such texts and overcomes the limitations of existing techniques, thereby answering our third research question, RQ3.

8.2.4 FURTHER DISCUSSIONS AND IMPROVEMENTS

In the above discussions, we have seen that the approaches developed in this thesis successfully address the 3 research questions, RQ1, RQ2, and RQ3, and consequently, enable us to realize our academic objective. Before analyzing our industrial objective, there are some aspects that deserve further mention, as will be described below.

a) Concerning our term extraction framework, our corpus-comparison approach computes the termhood of an expression as its relative probabilities in the domain-specific texts and in a general corpus. An alternative technique is to compare the corpora using statistical tests, such as the log-likelihood or χ^2 measures. The null hypothesis in this case is that there are no differences between the observed frequencies for the same expression in the domain-specific and in the general corpus. The alternative hypothesis is that the candidate's distribution is different in the domain-specific and general corpora. Based on the corresponding statistical score of the expression, we can reject the null hypothesis, in which case, the expression is considered to be domain-specific (i.e. relevant to the domain-specific corpus), or we can accept the null hypothesis and consider the expression to be irrelevant. The main benefit of using statistical tests for comparing the corpora is that they are more accurate in determining whether the higher frequency of an expression in the domain-specific text is really (statistically) significant or is only spurious (i.e. due to chance). Another potential improvement is in our use of Wikipedia as a general corpus for comparing the domain-specific texts. An attractive alternative to Wikipedia is to consider other large, general corpora like the British National Corpus (BNC). In fact, the BNC can be considered to be more representative (of the standard English language) than Wikipedia, and hence, could have improved our termhood computations.

b) Concerning our approach for mining causal relations, we can compare its performance against that of a suitable baseline, as was done for the case of part-whole relation extraction. This will enable us to estimate the gains in performance when leveraging upon Wikipedia to overcome the issue of data sparsity, and when relying on the Web to mitigate the negative influence of semantic-drift. A related experiment would be to investigate the effect of the search engine API, which we use to query the Web, on our performance. Currently, our approach is based on the Yahoo! API. Another search engine API that can be readily incorporated in our framework is that of Google.

c) Many of the challenges that we addressed are not peculiar to our specialized corporate (PD-CS) domain. They have also been observed when processing texts generated in other disciplines, and can thus be considered as general challenges in NLP. These include the issues of data sparsity (for term and relation extraction), semantic-drift (for relation extraction), and feature identification (for clustering). These difficulties, however, are more acute in our domain-specific texts. It is therefore reasonable to expect that the algorithms we propose in response to these challenges can also be applied, with relatively high performance, to general texts, such as newspaper articles, or to other types of domain-specific texts, such as those generated in healthcare (e.g. symptoms descriptions). The latter texts, in particular, exhibit many similarities with the corporate documents that we treated in this thesis.

d) Related to the above, we can evaluate the performance of our algorithms on traditional benchmarks datasets. Examples of such benchmark datasets include the L.A. Times corpus for evaluating our term extraction framework, the TREC collection or ACE corpus for our relation extraction framework, and the Reuters corpus for our text clustering framework. This will enable us to further validate our results, and compare them with the large number of related research efforts that have been also evaluated on these benchmark datasets.

8.3 INDUSTRIAL OBJECTIVE

Our industrial objective was to provide our algorithms and corresponding prototypes to industrial partners so that they could use them to unlock the valuable information, buried in their massive amount of texts.

As already mentioned, many of the algorithms (or modules thereof), developed in this thesis, have been implemented into applications and delivered to our industrial partners. For example, our term extraction framework has been employed in a text classification application. We are currently working on clustering customer complaints using our text clustering framework. Our relation extraction techniques will be used as part of a Question-Answering system to make the results of our algorithms more accessible to the users. Hence, to a large extent, our industrial objective has also been achieved.

There are still some issues concerning the industrial objective, which need to be taken care of. They were outside the scope of our project, and we will briefly describe them below.

- A crucial aspect that needs further consideration is the seamless integration of our applications, not only with the corporate IT ecosystem, but also, and even more importantly, with the existing business processes. An application that drastically disrupts the business processes, for example, in handling customer complaint cases, may lead to more inefficiencies.
- Another pertinent aspect is the usability (user-friendliness) of the deployed applications. Users will be reluctant to rely on applications that are cumbersome to use for supporting their daily operations, even if these applications are technically sound.
- Related to the issue of usability is that of formatting or presenting the results to suit the requirements or preferences of different users. For instance, an engineer and a data analyst will both be interested in determining the number of product failures and related details such as their causes, their status (solved, unsolved, in-progress) and the cost involved in fixing these failures. However, they are likely to have different preferences on how the information is presented to them. The engineer may be more interested in the technical details, while the data analyst may prefer an aggregated view, for example, failures by product types with the possibility of diving deeper into the details.

8.4 Future Directions

8.4.1 Effects of Seeds on Domain-Specific Part-Whole Relation Extraction

In our previous experiments, we have shown that the traditional practice of initializing a minimally-supervised algorithm with a single seed set, which mixes seeds of different part-whole types, may fail to extract some of the specialized part-whole patterns.

An interesting area of future work is to determine whether adopting a more principled approach to seed selection enables us to discover more accurate part-whole relations. It involves defining distinct sets of seeds for each part-whole relation type, and initializing our algorithm with each set to harvest patterns from the Wikipedia knowledge-base. The patterns are then used, as before, to extract the domain-specific relations from the domain-specific corpus. One can expect higher recall since, as we saw before, the use of separate seed sets for each individual part-whole type enables us to capture a larger number of specialized patterns. Such a strategy can also be beneficial to the precision. We observed during our previous experiments that the precision of the relations harvested when using distinct seed sets was as high, if not higher than, the precision obtained with sets of mixed seeds.

8.4.2 Causal Relation Extraction

Our approach for mining causal relation has focused on intra-sentential relations, where the causal agent and the effect are realized as events in the same sentence. However, in many cases, causal relations can also exist between larger units of texts, such as between sentences and even between entire paragraphs. Detecting these types of causal relations has been overlooked in current research, and is an interesting, albeit challenging, area of future work.

8.4.3 Incorporating Semantic Information for Clustering

We are also interested in determining whether the incorporation of more semantic information can help to improve our clustering accuracy. In the absence of domain-specific knowledge resources, we can investigate the use of general resources like WordNet or Wikipedia as sources of semantic information. For example, we can leverage on WordNet's synsets to acquire synonyms of the context-words that we use as features. Other related features appearing in the documents can be identified based on Wikipedia's category system (e.g. hypernyms). In this way, documents that share many synonymous or related features can be considered to be similar, and clustered together. The use of additional information from WordNet and Wikipedia may be beneficial to overcome the issue of data sparsity, and to facilitate the detection of more accurate clusters.

8.4.4 Multi-Lingual Information Extraction

In our work, we have mostly focused on extracting terms and relations from English texts. However, the rapid proliferation of electronic texts, such as web pages, can also be observed in languages other than English. In addition, many corporate organizations have forayed into new markets, dispersed over different geographical locations. Most corporate documents generated in these markets are likely to be expressed in the native language. For example, Spanish customers may find it more convenient to describe their complaints in Spanish. Similarly, a service engineer in France may prefer to narrate a service/repair action in French instead of English. There is therefore also a need for novel algorithms to extract meaningful information from these corporate (domain-specific) documents.

Since Wikipedia is available in multiple languages, our algorithms, which rely on Wikipedia, should, in principle, be relatively easy to extend for other languages. The only foreseeable difficulty is the availability of NLP tools, such as POS-taggers and syntactic parsers for non-English languages.

8.4.5 INFORMATION INTEGRATION

As already mentioned, our algorithms have mostly targeted documents generated in the PD-CS domain. The terms, semantic relations and clusters discovered from these documents provide useful information that can be used for improving product quality and customer satisfaction. More accurate information can be obtained by considering other sources, such as documents generated in the marketing domain. The information extracted from these disparate sources can then be integrated to provide a more coherent view on product quality and customer satisfaction. Such a task entails a number of interesting challenges, such as resolving conflicting information from different sources and determining the trustworthiness of the data sources.

8.4.6 ONTOLOGY LEARNING

Our terms and semantic relations can be encoded into ontology triples using a suitable language like OWL, and used to learn a domain-specific ontology. Such an ontology can then be used as a domain-specific knowledge resource to support future term and relation extraction endeavors. Furthermore, the inferencing mechanisms of an ontology can be applied to reason over the facts (terms and relations) extracted by our algorithms in order to automatically deduce or predict new information, such as the causes of product failures.

An interesting research question with ontologies is how to update an ontology with new information (terms, relations) extracted from new documents. It also entails the issues of conflicting information and trustworthiness of data sources.

8.4.7 QUESTION-ANSWERING SYSTEMS

Another attractive application that is worth investigating in future is that Question-Answering (QA) systems. A QA system will enable corporate users to pose natural language questions, such as "what are the main causes of blinking monitor", and will return as answers relevant information nuggets, such as "the blinking monitor was due to a short-circuit", extracted by our relation extraction approach. It is therefore particularly useful to make our results more accessible to the users. Developing a QA system from our existing work should not entail many difficulties. For example, the relations, and even the clusters, that we discovered from the corpus, can be used to populate the "answer table" of a QA system. One of the main challenges will be in dealing with non-factoid questions, which are common in our PD-CS domain when dealing with product failures/repairs, such as "why is the monitor still blinking after changing the frequency oscillator?".

BIBLIOGRAPHY

[1] R. Blumberg and S. Atre, "The problem with unstructured data," *DM REVIEW,* vol. 13, pp. 42--49, 2003.

[2] P. Russom, "BI Search and Text Analytics:TDWI Best Practices Report," 2007.

[3] A. van den Bosch and G. Bouma, Interactive Multi-modal Question-answering, Springer-Verlag New York Inc, 2011.

[4] C. Manning, P. Raghavan and H. Schutze, Introduction to information retrieval, 2008: Cambridge University Press.

[5] R. Rao, "From unstructured data to actionable intelligence," *IT professional,* vol. 5, no. 6, pp. 29--35, 2003.

[6] C. Manning, P. Raghavan and H. Schutze, Introduction to information retrieval, Cambridge University Press, 2008.

[7] R. Baeza-Yates and B. Ribeiro-Neto, Modern information retrieval, ACM press New York, 1999.

[8] D. Volitich, IBM Cognos 8 Business Intelligence: The Official Guide, McGraw-Hill, Inc., 2008.

[9] A. Berger, R. Caruana, D. Cohn, D. Freitag and V. Mittal, "Bridging the lexical chasm: statistical approaches to answer-finding," in *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, 2000.

[10] R. Soricut and E. Brill, "Automatic question answering: Beyond the factoid," in *Proceedings of HLT-NAACL*, 2004.

[11] C. Olston and E. Chi, "ScentTrails: Integrating browsing and searching on the Web," *ACM Transactions on Computer-Human Interaction,* vol. 10, no. 3, pp. 177--197, 2003.

[12] M. Katz and M. Byrne, "Effects of scent and breadth on use of site-specific search on e-commerce Web sites," *ACM Transactions on Computer-Human Interaction ,* vol. 10, no. 3, pp. 198--220, 2003.

[13] D. Klein and C. Manning, "Accurate unlexicalized parsing," in *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, 2003.

[14] A. McCray, S. Srinivasan and A. Browne, "Lexical methods for managing variation in biomedical terminologies.," in *Proceedings of the Annual Symposium on Computer Application in Medical Care*, 2005.

[15] D. Sanchez and A. Moreno, "Learning non-taxonomic relationships from web documents for domain ontology construction," *Data \& Knowledge Engineering,* vol. 64, no. 3, pp. 600--623, 2008.

[16] H. Harkema, R. Gaizauskas, M. Hepple, A. Roberts, I. Roberts, N. Davis and Y. Guo, "A large scale terminology resource for biomedical text processing," in *In Proceedings of Biolink Workshop, ACL*, 2004.

[17] S. Ponzetto and M. Strube, "Knowledge derived from Wikipedia for computing semantic relatedness," *Journal of Artificial Intelligence Research,* vol. 30, no. 1, pp. 181--212, 2007.

[18] M. Ashburner, C. Ball, J. Blake, D. Botstein, H. Butler, J. Cherry, A. Davis, K. Dolinski, S. Dwight, J. Eppig and others, "Gene Ontology: tool for the unification of biology," *Nature genetics,* vol. 25, no. 1, 2000.

[19] C. Fellbaum, WordNet: An electronic lexical database, The MIT press, 1998.

[20] B. Farley, "Extracting information from free-text aircraft repair notes," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing,* vol. 15, no. 4, pp. 295--305, 2001.

[21] S. Kim, R. Bracewell and K. Wallace, "Answering engineers' questions using semantic annotations," *Artificial Intelligence for Engineering Design, Analysis, and Manufacturin,* vol. 21, no. 2, pp. 155--171, 2007.

[22] A. Auger and C. Barriere, Probing Semantic Relations: Exploration and identification in specialized texts, John Benjamins Pub Co, 2010.

[23] S. a. C. P. Blohm, "Using the web to reduce data sparseness in pattern-based

information extraction," in *Knowledge Discovery in Databases: PKDD 2007*, 2007.

[24] W. Kuechler, "Business applications of unstructured text," *Communications of the ACM,* vol. 50, no. 10, pp. 86--93, 2007.

[25] "Merriam Webster," [Online]. Available: http://www.merriam-webster.com/. [Accessed 4 August 2011].

[26] C. Khoo and J. Na, "Semantic relations in information science," *Annual Review of Information Science and Technology,* vol. 40, p. 157, 2006.

[27] R. Xu and D. Wunsch, "Survey of clustering algorithms," *Neural Networks, IEEE Transactions on,* vol. 16, no. 3, pp. 645--678, 2005.

[28] M. Steinbach, G. Karypis and V. Kumar, "A comparison of document clustering techniques".

[29] D. Biber, S. Conrad and R. Reppen, in *Corpus linguistics: Investigating language structure and use*, Cambridge Univ Pr, 1998, p. 8.

[30] A. O'Keeffe and M. McCarthy, The Routledge handbook of corpus linguistics, Routledge, 2010.

[31] C. Manning and H. Schutze, Foundations of statistical natural language processing, MIT Press, 1999, p. 7.

[32] N. Chomsky, Syntactic structures, Walter de Gruyter, 2002.

[33] M. Dunn, S. Greenhill, S. Levinson and R. Gray, "Universals, Evolved structure of language shows lineage-specific trends in word-order," *Nature,* vol. 473, no. 7345, pp. 79--82, 2011.

[34] G. Nenadic, I. Spasic and S. Ananiadou, "Mining term similarities from corpora," *Terminology,* vol. 10, no. 1, pp. 55--80, 2004.

[35] P. Buitelaar, Ontology learning and population: bridging the gap between text and knowledge, Ios Pr Inc, 2008.

[36] P. Buitelaar, P. Cimiano and B. Magnini, "Ontology learning from text: methods, evaluation and applications," *Computational Linguistics,* vol. 32, no. 4, 2005.

[37] K. Frantzi, S. Ananiadou and H. Mima, "Automatic recognition of multi-word terms:. the C-value/NC-value method," *International Journal on Digital Libraries,* vol. 3, no. 2, pp. 115--130, 2000.

[38] K. Frantzi and S. Ananiadou, "The C-value/NC-value domain independent method for multi-word term extraction," *Journal of Natural Language Processing,* vol. 6, no. 3, pp. 145--179, 1999.

[39] S. Blohm and P. Cimiano, "Using the web to reduce data sparseness in pattern-based information extraction," in *Knowledge Discovery in Databases: PKDD* , 2007.

[40] M. Lapata and A. Lascarides, "Detecting novel compounds: The role of distributional evidence," in *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics*, 2003.

[41] J. Vivaldi and H. Rodriguez, "Improving term extraction by combining different techniques," *Terminology,* vol. 7, no. 1, pp. 31--48, 2001.

[42] D. Maynard and S. Ananiadou, "Identifying terms by their family and friends," in *Proceedings of the 18th conference on Computational linguistics*, 2000.

[43] D. Maynard and S. Ananiadou, "Term extraction using a similarity-based approach," *Recent advances in computational terminology,* 1999.

[44] D. Maynard and S. Ananiadou, *TRUCKS: A model for automatic multi-word term recognition,* 2000.

[45] S. Aubin and H. T., "Improving Term Extraction with Terminological Resources," in *Lecture Notes in Artificial Intelligence*, T. Salakoski, Ed., Springer-Verlag, 2006, pp. 380--387.

[46] V. Hoste, K. Vanopstal and E. Lefever, "The Automatic Detection of Scientific Terms in Patient Information," in *In Proceedings of RANLP*, 2007.

[47] "International Organization for Standardization," [Online]. Available: http://www.iso.org/iso/catalogue_detail.htm?csnumber=31696. [Accessed 4 August 2011].

[48] N. Sager, "Sublanguage: Linguistic phenomenon, computational tool," in *Analyzing language in restricted domains: sublanguage description and processing*, R.

Grishman and R. Kittredge, Eds., Hillsdale, New Jersey, Lawrence Erlbaum Associates, 1986, pp. 1--18.

[49] R. Grishman, "Adaptive information extraction and sublanguage analysis," in *Proc. of IJCAI 2001*, 2001.

[50] R. Kittredge and I. Mel'cuk, "Towards a computable model of meaning-text relations within a natural sublanguage," in *The Proceedings of IJCAI*, 1983.

[51] B. Daille, B. Habert, C. Jacquemin and J. Royaut'e, "Empirical observation of term variations and principles for their description," *Terminology,* vol. 3, no. 2, pp. 197--257, 1996.

[52] J. Sager, "In search of a foundation: Towards a theory of the term," *Terminology,* vol. 5, no. 1, pp. 41--57, 1998.

[53] J. Sager, "Term formation," *Handbook of terminology management,* vol. 1, pp. 25--41, 1997.

[54] K. Kageura and B. Umino, "Methods of automatic term recognition: A review," *Terminology,* vol. 3, no. 2, pp. 259--289, 1996.

[55] M. Pazienza, M. Pennacchiotti and F. Zanzotto, "Terminology extraction: an analysis of linguistic and statistical approaches," *Knowledge Mining,* pp. 255--279, 2005.

[56] S. Wright and G. Budin, Handbook of Terminology Management: Basic aspects of terminology management, John Benjamins Pub Co, 1997.

[57] I. Dagan and K. Church, "Termight: Identifying and translating technical terminology," in *Proceedings of the fourth conference on Applied natural language processing*, 1994.

[58] J. Justeson and S. Katz, "Technical terminology: some linguistic properties and an algorithm for identification in text," *Natural language engineering,* vol. 1, no. 1, pp. 9--27, 1995.

[59] S. Petrovic, J. Snajder and B. Basic, "Extending lexical association measures for collocation extraction," *Computer Speech & Language,* vol. 24, no. 2, pp. 383--394, 2010.

[60] P. Schone and D. Jurafsky, "Is knowledge-free induction of multiword unit dictionary headwords a solved problem," in *Proceedings of the 6th Conference on Empirical Methods in Natural Language Processing* , 2001.

[61] G. Salton, "Developments in automatic text retrieval," *Science,* vol. 253, no. 5023, p. 974, 1991.

[62] K. Ahmad, A. Davies, H. Fulford and M. Rogers, "What is a term? The semi-automatic extraction of terms from text," *Translation Studies: An interdiscipline,* pp. 267--278, 1994.

[63] T. Chung, "A corpus comparison approach for terminology extraction," *Terminology,* vol. 9, no. 2, pp. 221--246, 2003.

[64] K. Uchimoto, S. Sekine, M. Murata, H. Ozaku and H. Isahara, "Term recognition using corpora from different fields," *Terminology,* vol. 6, no. 2, pp. 233--256, 2001.

[65] P. Rayson and R. Garside, "Comparing corpora using frequency profiling," in *Proceedings of the workshop on Comparing corpora, Association for Computational Linguistics*, 2000.

[66] A. Kilgarriff, "Comparing corpora," *International journal of corpus linguistics,* vol. 6, no. 1, pp. 97--133, 2001.

[67] "Reference Guide for the British National Corpus (XML Edition)," 4 August 2011. [Online]. Available: ttp://www.natcorp.ox.ac.uk/XMLedition/URG/.

[68] J. Vivaldi and H. Rodriguez, "Evaluation of terms and term extraction systems: A practical approach," *Terminology,* vol. 13, no. 2, pp. 225--248, 2007.

[69] K. Church and P. Hanks, "Word association norms, mutual information, and lexicography," *Computational linguistics,* vol. 16, no. 1, pp. 22--29, 1990.

[70] B. Daille, "Approche mixte pour l'extraction de terminologie: statistique lexicale et filtres linguistiques," 1994.

[71] T. Dunning, "Accurate methods for the statistics of surprise and coincidence," *Computational linguistics,* vol. 19, no. 1, pp. 61--74, 1993.

[72] P. Vossen, "EuroWordNet a multilingual database with lexical semantic network,"

Computational Linguistics, vol. 25, no. 4, 1999.

[73] F. Xu, D. Kurz, J. Piskorski and S. Schmeier, "Term extraction and mining of term relations from unrestricted texts in the financial domain," in *Business Information Systems*, Springer, 2002.

[74] B. Hamp and H. Feldweg, "Germanet-a lexical-semantic net for german," in *Proceedings of ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, 1995.

[75] T. Vu, A. Aw and M. Zhang, "Term extraction through unithood and termhood unification," in *Proceedings of the 3rd International Joint Conference on Natural Language Processing*, 2008.

[76] J. Wermter and U. Hahn, "Paradigmatic modifiability statistics for the extraction of complex multi-word terms," in *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, 2005.

[77] Z. Zhang, J. Iria, C. Brewster and F. Ciravegna, "A comparative evaluation of term recognition algorithms," in *Proceedings of the sixth international conference of Language Resources and Evaluation (LREC 2008)*, 2008.

[78] E. Milios, Y. Zhang, B. He and L. Dong, "Automatic term extraction and document similarity in special text corpora," in *Proceedings of the Sixth Conference of the Pacific Association for Computational Linguistics*, 2003.

[79] A. Hliaoutakis, K. Zervanou and E. Petrakis, "The AMTEx approach in the medical document indexing and retrieval application," *Data \& Knowledge Engineering,* vol. 68, no. 3, pp. 380--392, 2009.

[80] P. Drouin, "Term extraction using non-technical corpora as a point of leverage," *Terminology,* vol. 9, no. 1, pp. 99--115, 2003.

[81] P. Pecina and P. Schlesinger, "Combining association measures for collocation extraction," in *Proceedings of the COLING/ACL on Main conference poster sessions*, 2006.

[82] S. Agarwal, S. Godbole, D. Punjani and S. Roy, "How much noise is too much: A study in automatic text classification," in *Seventh IEEE International Conference on Data Mining*, 2007.

[83] R. Ananthanarayanan, V. Chenthamarakshan, P. Deshpande and R. Krishnapuram, "Rule based synonyms for entity extraction from noisy text," in *Proceedings of the second workshop on Analytics for noisy unstructured text data*, 2008.

[84] L. Subramaniam, S. Roy, T. Faruquie and S. Negi, "A survey of types of text noise and techniques to handle noisy text," in *Proceedings of The Third Workshop on Analytics for Noisy Unstructured Text Data*, 2009.

[85] A. Mikheev, "Periods, capitalized words, etc.," *Computational Linguistics,* vol. 28, no. 3, pp. 289--318.

[86] H. Nakagawa and T. Mori, "Nested collocation and compound noun for term recognition," in *Proceedings of the First Workshop on Computational Terminology COMPUTERM*, 64--70.

[87] "Medical Subject Headings," [Online]. Available: http://www.nlm.nih.gov/mesh/. [Accessed 4 August 2011].

[88] "ziekenhuis.nl," [Online]. Available: http://ziekenhuis.nl/. [Accessed 04 August 2011].

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

[90] M. Marcus, M. Marcinkiewicz and B. Santorini, "Building a large annotated corpus of English: The Penn Treebank," *Computational linguistics,* vol. 19, no. 2, pp. 313--330.

[91] "WikiXML Collection," ISLA, University of Amsterdam, [Online]. Available: http://ilps.science.uva.nl/WikiXML/.

[92] C. Guinaudeau, G. Gravier and P. Sebillot, "Enhancing lexical cohesion measure with confidence measures, semantic relations and language model interpolation for multimedia spoken content topic segmentation," *Computer Speech \& Language,* 2011.

[93] S. Piao, P. Rayson, D. Archer and T. McEnery, "Comparing and combining a semantic tagger and a statistical tool for MWE extraction," *Computer Speech \& Language,* vol. 19, no. 4, pp. 378--397, 2005.

[94] S. Evert and B. Krenn, "Using small random samples for the manual evaluation of statistical association measures," *Computer Speech \& Language,* vol. 19, no. 4, pp. 450--466, 2005.

[95] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and psychological measurement,* vol. 20, no. 1, pp. 37--46, 1960.

[96] C. van Rijsbergen, Information Retrieval, London: Butterworths, 1979.

[97] F. d. Saussure, B. C, S. A and T. De Mauro, Cours de linguistique general, 1916.

[98] J. Lyons, Semantics. 2 vols, Cambridge: Cambridge University Press, 1977.

[99] J. Lyons, Linguistic semantics: an introduction, Cambridge Univ Pr, 1995.

[100] R. Chaffin and D. J. Herrmann, "The nature of semantic relations: A comparison of two approaches," in *Relational models of the lexicon: Representing knowledge in semantic networks*, M. Evens, Ed., Cambridge, Cambridge University Press, 1988, pp. 289--334.

[101] J. F. Sowa, Conceptual structures: Information processing in mind and machine, Reading, MA: Addison-Wesley Pub, 1983.

[102] McKeon, The basic works of Aristotle, R. McKeon, Ed., New York: The Modern Library, 2001.

[103] M. Evens, B. Litowitz, J. Markowitz, R. Smith and O. Werner, "Lexical-Semantic Relations: A Comparative Survey. Linguistic Research," in *Linguistic Research*, 1980.

[104] R. Chaffin and D. Herrmann, "The similarity and diversity of semantic relations," *Memory \& cognition,* vol. 12, no. 2, pp. 134--141, 1984.

[105] M. Hudon, "Relationships in multilingual thesauri," in *Relationships in the organization of knowledge*, C. Bean and R. Green, Eds., Dordrecht: Kluwer, 2001, pp. 67--80.

[106] R. Girju, A. Badulescu and D. Moldovan, "Learning semantic constraints for the automatic discovery of part-whole relations," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, 2003.

[107] R. Girju, A. Badulescu and D. Moldovan, "Automatic discovery of part-whole relations," *Computational Linguistics,* vol. 32, no. 1, pp. 83--135, 2006.

[108] B. Beamer, A. Rozovskaya and R. Girju, "Automatic semantic relation extraction with multiple boundary generation," in *Proceedings of the 23rd national conference on Artificial intelligence*, 2008.

[109] C. Khoo, J. Kornfilt, R. Oddy and S. Myaeng, "Automatic extraction of cause-effect information from newspaper text without knowledge-based inferencing," *Literary and Linguistic Computing,* vol. 13, no. 4, 1998.

[110] P. Pantel and M. Pennacchiotti, "Espresso: Leveraging generic patterns for automatically harvesting semantic relations," in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, 2006.

[111] M. Winston, R. Chaffin and D. Herrmann, "A taxonomy of part-whole relations," *Cognitive science,* vol. 11, no. 4, pp. 417--444, 1987.

[112] P. Gerstl and S. Pribbenow, "Midwinters, end games, and body parts: a classification of part-whole relations," *International Journal of Human Computer Studies,* vol. 43, no. 5, pp. 865--890, 1995.

[113] J. Odell, "Six different kinds of composition," *Journal of Object-Oriented Programming,* vol. 5, no. 8, pp. 10--15, 1994.

[114] C. Keet and A. Artale, "Representing and reasoning over a taxonomy of part--whole relations," *Applied Ontology,* vol. 3, no. 1, pp. 91--110, 2008.

[115] M. Berland and E. Charniak, "Finding parts in very large corpora," in *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, 1999.

[116] W. van Hage, H. Kolb and G. Schreiber, "A method for learning part-whole relations," in *Lecture Notes in Computer Science*, vol. 4273, 2006, pp. 723--735.

[117] E. Voorhees and D. Tice, "Overview of the TREC-9 question answering track," in *The Ninth Text REtrieval Conference (TREC9)*, 2001.

[118] P. Cimiano, A. Pivk, L. Schmidt-Thieme and S. Staab, "Learning taxonomic relations from heterogeneous sources of evidence," in *Ontology Learning from Text: Methods, evaluation and applications*, P. Buitelaar, P. Cimiano and B. Magnini, Eds., Amsterdam, IOS-Press, 2005.

[119] T. McIntosh and J. Curran, "Reducing semantic drift with bagging and distributional similarity," in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 2009.

[120] A. Gangemi, N. Guarino, C. Masolo, A. Oltramari and L. Schneider, "Sweetening ontologies with DOLCE," in *Proceedings of Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web*, 2002.

[121] G. Miller, C. Leacock, R. Tengi and R. Bunker, "A semantic concordance," in *Proceedings of the workshop on Human Language Technology*, 1993.

[122] "AGROVOC," [Online]. Available: http://aims.fao.org/website/AGROVOC-Thesaurus/sub. [Accessed 04 August 2011].

[123] D. Lenat, R. Guha, K. Pittman, D. Pratt and M. Shepherd, "Cyc: toward programs with common sense," *Communications of the ACM,* vol. 33, no. 8, pp. 30--49.

[124] C. Matuszek, M. Witbrock, R. Kahlert, J. Cabral, D. Schneider, P. Shah and D. Lenat, "Searching for common sense: Populating Cyc™ from the Web," in *Proceedings of the National Conference on Artificial Intelligence*, 2005.

[125] "LDC-Projects-ACE (Automatic Content Extraction)," [Online]. Available: http://projects.ldc.upenn.edu/ace/.

[126] R. Mihalcea, "Using wikipedia for automatic word sense disambiguation," in *Proceedings of NAACL HLT*, 2007.

[127] F. a. K. G. a. W. G. Suchanek, "Yago: a core of semantic knowledge," in *Proceedings of the 16th international conference on World Wide Web*, 2007.

[128] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak and Z. Ives, "Dbpedia: A nucleus for a web of open data," in *Proceedings of 6th International Semantic Web Conference and Second Asian Semantic Web Conference (ISWC/ASWC)*, 2007.

[129] O. Medelyan, D. Milne, C. Legg and I. Witten, "Mining meaning from Wikipedia," *International Journal of Human-Computer Studies,* vol. 67, no. 9, pp. 716--754, 2009.

[130] E. Morin, "Automatic acquisition of semantic relations between terms from technical corpora," in *Proc. of the Fifth International Congress on Terminology and Knowledge Engineering-TKE'99*, 1999.

[131] A. Condamines, "Corpus analysis and conceptual relation patterns," *Terminology,* vol. 8, no. 1, pp. 141--162, 2002.

[132] M. Iris, B. Litowitz and M. Evens, "Problems of the part-whole relation," in *Relations models of the lexicon*, M. Evens, Ed., Cambridge University Press, 1988, p. 261–28.

[133] S. Brin, "Extracting patterns and relations from the world wide web," in *Proceedings of World Wide Web and Databases*, 1999.

[134] E. Agichtein and L. Gravano, "Snowball: Extracting relations from large plain-text collections," in *Proceedings of the fifth ACM conference on Digital libraries*, 2000.

[135] M. Stevenson and M. Greenwood, "Dependency pattern models for information extraction," *Research on Language \& Computation,* vol. 7, no. 1, pp. 13--39, 2009.

[136] D. Seaghdha and A. Copestake, "Co-occurrence contexts for noun compound interpretation," in *Proceedings of Workshop on a Broader Perspective on Multiword Expressions*, 2007.

[137] R. Mihalcea and A. Csomai, "Wikify!: linking documents to encyclopedic knowledge," in *Proceedings of 16th Conference on Information and Knowledge Management*, 2007.

[138] A. Ittoo and G. Bouma, "On learning subtypes of the part-whole relation: do not mix your seeds," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 2010.

[139] P. Pantel and D. Lin, "A statistical corpus-based term extractor," in *AI '01*

Proceedings of the 14th Biennial Conference of the Canadian Society on Computational Studies of Intelligence: Advances in Artificial Intelligence, 2001.

[140] P. Pantel and D. Ravichandran, "Automatically labeling semantic classes," in *Proceedings of HLT/NAACL*, 2004.

[141] P. Turney, "The latent relation mapping engine: Algorithm and experiments," *Journal of Artificial Intelligence Research,* vol. 33, no. 1, pp. 615--655, 2008.

[142] A. Ittoo and G. Bouma, "Semantic Selectional Restrictions for Disambiguating Meronymy Relations," in *Computational Linguistics in the Netherlands*, B. Plank, E. Tjong Kim Sang and T. Van de Cruys, Eds., LOT, 2009, pp. 83--98.

[143] P. Pantel, D. Ravichandran and E. Hovy, "Towards terascale knowledge acquisition," in *Proceedings of the 20th international conference on Computational Linguistics*, 2004.

[144] T. Kiso, M. Shimbo, M. Komachi and Y. Matsumoto, "HITS-based seed selection and stop list construction for bootstrapping," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, 2011.

[145] R. Girju, "Automatic detection of causal relations for question answering," in *Proceedings of the ACL 2003 workshop on Multilingual summarization and question answering-Volume 12*, 2003.

[146] B. Rink, C. Bejan and S. Harabagiu, "Learning Textual Graph Patterns to Detect Causal Event Relations," in *Proceedings of the 23rd Florida Artificial Intelligence Research Society International Conference*, 2010.

[147] S. Bethard, W. Corvey, S. Klingenstein and J. Martin, "Building a corpus of temporal-causal structure," in *Proceedings of the Sixth International Language Resources and Evaluation Conference*, 2008.

[148] B. Beamer, S. Bhat, B. Chee, A. Fister, A. Rozovskaya and R. Girju, "UIUC: A knowledge-rich approach to identifying semantic relations between nominals," in *Proceedings of the Fourth International Workshop on Semantic Evaluations*, 2007.

[149] C. Khoo, S. Chan and Y. Niu, "The many facets of the cause-effect relation," in *The semantics of relationships: an interdisciplinary perspective*, Kluwer Academic Pub, 2002.

[150] F. Keil, Concepts, kinds, and cognitive development, The MIT Press, 1992.

[151] D. Hume, An abstract of a treatise of human nature, CUP Archive, 1938.

[152] C. Barriere, "Hierarchical refinement and representation of the causal relation," *Terminology,* vol. 8, no. 1, pp. 91--111, 2002.

[153] C. Khoo, "Automatic identification of causal relations in text and their use for improving precision in information retrieval," 1995.

[154] C. Khoo, S. Chan and Y. Niu, "Extracting causal knowledge from a medical database using graphical patterns," in *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, 2000.

[155] "MEDLINE," [Online]. Available: http://www.nlm.nih.gov/bsd/pmresources.html. [Accessed 4 8 2011].

[156] B. Levin and M. Rappaport Hovav, "Wiping the Slate Clean: A Lexical Semantic Exploration," in *Lexical and conceptual semantics*, B. Levin and S. Pinker, Eds., Blackwell, 1995, pp. 123-151.

[157] G. van Noord, "At last parsing is now operational.," in *13e conference sur le traitement automatique des langues naturelles*, 2006.

[158] B. Everitt, S. Landau and M. Leese, Cluster analysis., London: Arnold, 2001.

[159] J. Hartigan, Clustering algorithms, John Wiley \& Sons, Inc., 1975.

[160] M. Wedel and W. Kamakura, Market Segmentation: Conceptual and Methodological Foundations, Kluwer Academic Publishers, 2000.

[161] A. Machauer and S. Morgner, "Segmentation of bank customers by expected benefits and attitudes," *International Journal of Bank Marketing,* vol. 19, no. 1, pp. 6--18, 2001.

[162] B. aglam, F. Salman, S. Sayin and M. Turkay, "A mixed-integer programming approach to the clustering problem with an application in customer segmentation," *European journal of operational research,* vol. 173, no. 3, pp. 866--879, 2006.

[163] S. Sohn and Y. Kim, "Searching customer patterns of mobile service using clustering and quantitative association rule," *Expert Systems with Applications,* vol. 34, no. 2, pp. 1070--1077, 2007.

[164] H. Shin and S. Sohn, "Segmentation of stock trading customers according to potential value," *Expert systems with applications,* vol. 27, no. 1, pp. 27--33, 2004.

[165] J. Ahn and S. Sohn, "Customer pattern search for after-sales service in manufacturing," *Expert Systems with Applications,* vol. 36, no. 3, pp. 5371--5375, 2009.

[166] A. Ittoo, Y. Zhang and J. Jiao, "A Text Mining-Based Recommendation System for Customer Decision Making in Online Product Customizatio," in *Management of Innovation and Technology, 2006 IEEE International Conference on*, IEEE, 2006, pp. 473--477.

[167] B. Fung, K. Wang and M. Ester, "Hierarchical document clustering using frequent itemsets," in *Proceedings of the SIAM International Conference on Data Mining*, 2003.

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

[169] F. Beil, M. Ester and X. Xu, "requent term-based text clusterin," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*.

[170] W. Zhang, T. Yoshida, X. Tang and Q. Wang, "Text clustering using frequent itemsets," *Knowledge-Based Systems,* vol. 23, no. 5, pp. 379--388, 2010.

[171] P. Hansen and B. Jaumard, "Cluster analysis and mathematical programming," *Mathematical programming,* vol. 79, no. 1, pp. 191--215, 1997.

[172] A. Jain and R. Dubes, Algorithms for clustering data, Prentice-Hall, Inc., 1988.

[173] G. Forman, "An extensive empirical study of feature selection metrics for text classification," *The Journal of Machine Learning Research,* vol. 3, pp. 1289--1305.

[174] M. Dash and H. Liu, "Feature Selection for Clustering," in *Proceeding PADKK '00 Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Current Issues and New Applications*, 2000.

[175] Y. Yang and J. Pedersen, "A comparative study on feature selection in text categorization," in *International Conference on Machine Learning (ICML)*, 1997.

[176] T. Liu, S. Liu, Z. Chen and W. Ma, "An evaluation on feature selection for text clustering," in *International Conference on Machine Learning (ICML)*, 2003.

[177] I. Jolliffe, "Principal component analysis," in *Encyclopedia of Statistics in Behavioral Science*, Wiley Online Library, 2002.

[178] M. Dash and H. Liu, "Feature selection for classification," *Intelligent data analysis,* vol. 1, no. 3, pp. 131--156, 1997.

[179] P. Langley, "Selection of relevant features in machine learning," in *Proceedings of the AAAI Fall symposium on relevance*, 1994.

[180] B. Tang, M. Shepherd, E. Milios and H. M.I., "Comparing and combining dimension reduction techniques for efficient text clustering," in *International Workshop on Feature Selection for Data Mining – Interfacing Machine Learning and Statistics in conjunction with 2005 SIAM International Conference on Data Mining*, 2005.

[181] Y. Li and L. Song, "Threshold Determining Method for Feature Selection}," in *Electronic Commerce and Security, 2009. ISECS'09. Second International Symposium on*, 2009.

[182] V. Roth and T. Lange, "Feature selection in clustering problem," *Advances in neural information processing systems,* vol. 16, pp. 473--480, 2004.

[183] M. Steinbach, G. Karypis and V. Kumar, "A comparison of document clustering techniques," in *KDD workshop on text mining*, 2000.

[184] A. Jain, M. Murty and P. Flynn, "Data clustering: a review," *ACM computing surveys (CSUR),* vol. 31, no. 3, pp. 264--323, 1999.

[185] T. Zhang, R. Ramakrishnan and M. Livny, "BIRCH: an efficient data clustering method for very large databases," in *Proceedings of ACM SIGMOD Conf. Management of Data*, 1996.

[186] J. MacQueen, "Some methods for classification and analysis of multivariate

observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability}*, 1967.

[187] C. Chen, F. Tseng and T. Liang, "Mining fuzzy frequent itemsets for hierarchical document clustering," *Information Processing \& Management,* vol. 46, no. 2, pp. 193--211, 2010.

[188] B. Liu, Web data mining, Springer, 2007.

[189] R. Agrawal and Srikant, "Fast algorithms for mining association rules," in *Proceedings of 20th Int. Conference on Very Large Data Bases, VLDB*, 1994.

[190] T. Hong, K. Lin and S. Wang, "Fuzzy data mining for interesting generalized association rules," *Fuzzy sets and systems,* vol. 138, no. 2, pp. 255--269, 2003.

[191] L. Galavotti, F. Sebastiani and M. Simi, "Feature selection and negative evidence in automated text categorization," in *roceedings of KDD-00*, 2000.

[192] F. Gravetter and L. Wallnau, Essentials of statistics for the behavioral sciences, Wadsworth Pub Co, 2000, p. 194.

[193] "Critical Values of the Chi-Square Distribution," [Online]. Available: http://itl.nist.gov/div898/handbook/eda/section3/eda3674.htm. [Accessed 04 August 2011].

[194] "Chi-square distribution table," [Online]. Available: http://www.medcalc.org/manual/chi-square-table.php. [Accessed 04 August 2011].

[195] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. Witten, "The WEKA data mining software: an update," *ACM SIGKDD Explorations Newsletter,* vol. 11, no. 1, pp. 10--18, 2009.

[196] I. a. F. E. Witten, Data Mining: Practical machine learning tools and techniques, Morgan Kaufmann, 2005.

[197] "Machine Learning Group at University of Waikato - Downloading and installing Weka," [Online]. Available: http://www.cs.waikato.ac.nz/~ml/weka/. [Accessed 04 August 2011].

[198] A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure," in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2007.

[199] "FIHC 1.0," [Online]. Available: http://ddm.cs.sfu.ca/software.html.

[200] J. Svartvik, "Corpus linguistics 25+ years on," *Language and Computer,* vol. 62, no. 1, pp. 11--25, 2007.

[201] M. A. K. Halliday, W. Teubert and C. Yallop, "Language and Corpus Linguistics," in *Lexicology and corpus linguistics: an introduction*, Continuum Intl Pub Group, 2004, p. 98.

SAMENVATTING

INLEIDING

Geschat wordt dat 80% van alle bedrijfsinformatie bestaat uit documenten waar informatie is beschreven in de vorm van vrije tekst, terwijl maar 20% bestaat uit gestructureerde informatie. Verborgen in deze documenten is kostbare informatie die van strategisch belang kan zijn voor ondernemingen, zoals klachten van klanten en beschrijvingen van reparaties die door onderhoudspersoneel zijn uitgevoerd. Bestaande 'Business Intelligence' software is vooral gericht op gestructureerde data, zoals databases, en is niet in staat om informatie te ontdekken in tekstdocumenten. Als gevolg hiervan kunnen ondernemingen onvoldoende profiteren van belangrijke inzichten die aanwezig zijn in de documenten die het bedrijf produceert. Dit leidt tot gemiste kansen, risico's, en onvermogen om bijvoorbeeld op onvrede en niet vervulde verwachtingen van klanten te reageren.

Binnen de Natuurlijke-taalverwerking (Natural Language Processing, NLP) zijn verschillende technieken ontwikkeld om informatie te ontdekken in teksten. Dit zijn onder andere technieken voor informatie-extractie (IE) en tekst-clustering (TC). Deze technieken zijn succesvol toegepast op grote collecties van goed geschreven tekst, zoals verzamelingen krantenartikelen en (medisch-) wetenschappelijke artikelen. Zulke documenten bevatten betrouwbare taalkundige en statistische patronen, die het ontdekken van informatie mogelijk maken. Documenten binnen ondernemingen, zoals die bijvoorbeeld gebruikt worden tijdens productontwikkeling en klantcontacten, vormen een uitdaging voor bestaande NLP-technieken. Zulke documenten zijn vaak zeer informeel, en bevatten regelmatig ongrammaticale zinsconstructies. Ze zijn meestal ook veel beknopter dan documenten waarop NLP-technieken getest zijn, en bevatten daarom ook minder betrouwbare taalkundige en statistische signalen. Bestaande NLP-technieken zijn daarom vaak niet in staat om accuraat informatie aan zulke teksten te onttrekken. Deze tekortkoming in bestaande NLP-technieken was de motivatie voor ons onderzoek.

De onderzoeksvraag die centraal staat in dit werk is:

> "Hoe kan automatisch informatie gevonden worden in beknopte, domein-specifieke, en informele teksten?"

Meer in het bijzonder onderzochten we de volgende drie deelvragen:

> RQ1: Hoe kunnen termen en concepten accuraat worden geïdentificeerd in domein-specifieke, beknopte, en informele teksten binnen ondernemingen?
>
> RQ2: Hoe kunnen semantische relaties binnen domein-specifieke, beknopte, en informale teksten automatisch gevonden worden?
>
> RQ3: Hoe kunnen coherente en homogene clusters van vergelijkbare documenten worden gevonden in een verzameling domein-specifieke, beknopte, en informele teksten?

ONDERZOEKSVRAAG 1 (RQ1) – TERMINOLOGIE-EXTRACTIE

In hoofdstuk 2 gaven we een antwoord op vraag RQ1 in de vorm van ExtTerm, een zelflerend model voor terminologie-extractie. Om het probleem van het identificeren van zeldzame, infrequente, termen op te lossen maakt ExtTerm gebruik van de 'termhood'-eigenschap. De 'termhood' van een uitdrukking wordt berekend door de relatieve frequentie waarmee een term in een domein-specifiek en een algemeen corpus voorkomt te vergelijken. Hierdoor kunnen termen die relatief vaak voorkomen in een domein-specifiek corpus als term worden aangemerkt, ook als ze in absolute zin weinig frequent zijn. Om te kunnen omgaan met informeel taalgebruik hebben we een taalkundig filter gedefinieerd dat foutieve, automatisch toegekende, woordsoorten kan verbeteren. Voor

het het ontdekken van termen die bestaan uit meerdere woorden ontwikkelden we een nieuw algoritme dat kijkt naar de 'unithood' van delen van zulke uitdrukkingen. 'Unithood' bepalen we met behulp van de 'mutual information' score.

We testten de prestaties van ExtTerm op een realistisch corpus van informeel en beknopt taalgebruik.
De prestaties werden gemeten door de gebruikelijke maten precisie, recall, en F1-score. De resultaten zijn vergeleken met dat van het bekende C-value algoritme. Onze resultaten laten zien dat ExtTerm op een dergelijk corpus beter werkt dan C-Value. De tekortkomingen van bestaande technieken worden door ExtTerm opgelost, en daarmee is een antwoord gegeven op vraag RQ1.

ONDERZOEKSVRAAG 2 (RQ2) - RELATIE-EXTRACTIE

In hoofdstuk 3 definieerden we een semantische relatie als een tripel, bestaande uit een tweetal termen en een patroon dat dit paar verbindt. Het tripel <"consist of ", "engine", "car"> bestaat bijvoorbeeld uit het patroon *"consist of"*, dat een deel-geheel relatie uitdrukt, een deel ("engine") en een geheel ("engine"). We concentreerden ons op deel-geheel en oorzaak-gevolg relaties, omdat die een belangrijk deel uitmaken van informatie in documenten binnen ondernemingen.

In hoofdstuk 4 en 5 gingen we in op vraag RQ2 en onderzochten we de deel-geheel en oorzaak-gevolg relatie in domein-specifieke teksten. Ons algoritme maakt gebruik van een klein aantal voorbeelden van de relatie (zogenaamde "seeds") om patronen voor de relatie te leren. Voor de deel-geheel relatie gebruikten we bijvoorbeeld "engine-car" en voor de oorzaak-gevolg relatie "hiv-aids". Het achterliggende idee is dat algoritme recursief patronen leert die voorbeeldparen verbinden, daarmee nieuwe voorbeelden kan vinden, die weer nieuwe patronen opleveren, enzovoort tot een stop-conditie wordt bereikt.

In onze aanpak proberen we niet direct patronen voor deel-geheel en oorzaak-gevolg relaties te ontdekken in de domein-specifieke teksten, maar leren we zulke patronen op basis van een algemeen corpus. Als algemeen corpus gebruikten we Wikipedia. Wikipedia is omvangrijk en zeer divers wat onderwerp betreft. Dit garandeert dat verschillende patronen voor een semantische relatie geleerd kunnen worden. Om te voorkomen dat op basis van een klein aantal voorbeelden ook patronen worden geleerd die een andere relatie typeren ('semantic-drift') maken we gebruik van een mechanisme dat is gebaseerd op de 'Latent Relational Hypothesis'. De patronen die we leren zijn gebaseerd op taalkundige analyse van de tekst. Dit maakt het mogelijk afhankelijkheden te leren tussen woorden die niet naast elkaar staan, en voorbeelden te ontdekken onafhankelijk van hun volgorde in de tekst. Nadat we een verzameling patronen voor de deel-geheel en oorzaak-gevolg relatie hebben gevonden in de algemene tekst, gebruiken we deze om domein-specifieke instanties van deze relaties te vinden in de domein-specifieke teksten. Een instantie bestaat uit een taalkundig patroon, en een tweetal termen die zijn geïdentificeerd door ExtTerm.

Net als in hoofdstuk 2 bepaalden we hoe succesvol deze benadering is door de 'accuratesse' te bepalen in het corpus dat ook gebruikt werd voor de evaluatie van terminologie-extractie. We vergeleken deze aanpak met het 'state-of-the-art' Espresso algoritme voor informatie-extractie.

De resultaten geven aan dat onze benadering beter werkt dan Espresso op beknopte, informele, en domein-specifieke teksten. Het biedt een oplossing voor de problemen die zulke teksten opleveren voor standaard informatie-extractie technieken, en geeft zo een antwoord op vraag RQ2. De resultaten geven ook aan dat de aanpak generiek is, en kan worden gebruikt om verschillende soorten relaties te leren.

Hoofdstuk 6 bevat een aantal aanvullende experimenten om te onderzoeken wat de rol is van de voorbeelden waarmee het leerproces wordt opgestart. We beperkten ons tot deel-geheel relaties, omdat die eenvoudig in verschillende subtypes in te delen zijn. Onze voornaamste conclusie is dat keuze van de initiële voorbeelden een vrij groot effect heeft op de accuratesse van het uiteindelijke resultaat. Transitieve deel-geheel relaties (ook wel 'mereologische' relaties) zijn het meest vruchtbaar en geven het meest accurate resultaat. Niet-transitieve deel-geheel relaties (ook wel: 'meronymische' relaties) geven het minst

goede resultaat. De gebruikelijke methode om geen onderscheid te maken tussen subtypes van een relatie bij het selecteren van voorbeelden werkt relatief slecht en geeft onvoorspelbare resultaten.

ODERZOEKSVRAAG 3 (RQ3) – TEKST-CLUSTERING

In hoofdstuk 7 gingen we in op onderzoeksvraag RQ3 door een algoritme (ClustTExt) te ontwikkelen voor tekst-clustering gebaseerd op 'wordsets'. Om kwalitatief goede, niet overlappende, clusters te vinden ontwikkelden we een feature-selectie methode gebaseerd op een combinatie van 'wrapper' en 'filter' technieken. Dit leidt tot dezelfde hoge kwaliteit als gebruikelijke 'wrapper' methoden, maar zonder de hoge computationele kosten. Nadat we de meest informatieve woorden in het corpus bepaald hebben, gebruiken we het Apriori algoritme om veel voorkomende 'wordsets' te ontdekken. Dit zijn woordcombinaties die in een aantal documenten samen voorkomen. Iedere wordset beschrijft een thema in het corpus, en kan bovendien gebruikt worden als etiket voor het bijbehorende cluster. Clusters zijn in eerste instantie leeg, en worden gevuld met documenten op basis van de volgende criteria: i) de overeenkomst tussen het cluster-etiket en het document, ii) een hoge mate van overeenkomst tussen documenten in hetzelfde cluster, en iii) een hoge mate van verschil tussen (documenten in) clusters onderling.

We testten deze aanpak op ons corpus van domein-specifieke teksten. Bij gebrek aan een standaard voor het evalueren van clusters hebben we onze eigen maat ontwikkeld. De maat houdt rekening met de kwaliteit van het etiket van een cluster en de vraag in hoeverre een etiket correct is voor ieder document binnen het cluster. We vergeleken de prestaties van ons algoritme met een 'state-of-the-art' wordset-techniek (Frequent Itemset-based Hierarchical Clustering, FIHC). Onze benadering werkt beter dat FIHC voor dit corpus. Het algoritme is in staat domein-specifieke, informele, beknopte documenten accuraat te clusteren, en biedt daarmee een antwoord op vraag RQ3.