

2014

Ontology-based knowledge discovery from unstructured and semi-structured text

Jantima Polpinij
University of Wollongong

UNIVERSITY OF WOLLONGONG

COPYRIGHT WARNING

You may print or download ONE copy of this document for the purpose of your own research or study. The University does not authorise you to copy, communicate or otherwise make available electronically to any other person any copyright material contained on this site. You are reminded of the following:

Copyright owners are entitled to take legal action against persons who infringe their copyright. A reproduction of material that is protected by copyright may be a copyright infringement. A court may impose penalties and award damages in relation to offences and infringements relating to copyright material. Higher penalties may apply, and higher damages may be awarded, for offences and infringements involving the conversion of material into digital or electronic form.

**UNIVERSITY OF
WOLLONGONG**



School of Computer Science and Software Engineering

**Ontology-Based Knowledge Discovery from Unstructured and
Semi-structured Text**

Jantima Polpinij

**This thesis is presented as part of the requirements for the
award of the Degree of Doctor of Philosophy
of the
University of Wollongong**

July and 2014

ABSTRACT

This dissertation proposes a novel methodology for knowledge discovery in large data sets, with a focus on unstructured and semi-structured textual data. To our knowledge, extracting knowledge from unstructured and semi-structured textual data is a major unsolved problem in the area of knowledge discovery in databases (KDD). The problem becomes particularly acute due to ambiguity and lexical variations in natural language. This thesis seeks to address these problems. Firstly, it proposes a unified methodology, called the *Ontology-based Knowledge Discovery in unstructured and semi-structured Text* (On-KDT) methodology, to discover knowledge from unstructured/semi-structured texts. This approach leverages semantic information encoded in ontologies to improve the effectiveness of the knowledge extraction processes. Secondly, the On-KDT methodology is validated in three distinct settings.

In the first setting, we extract scenarios from natural language software requirements. Extracting scenarios from natural language requirements helps improve the efficiency of the requirements process. In this study, the requirements for a course-registration system are used as the case study. The On-KDT methodology is applied to extract scenarios describing three distinct components in the system.

In the second setting, we extract clinical knowledge from PubMed abstracts. PubMed is a very large collection of biomedical abstracts. To be able to make decisions that bring to bear the latest in biomedical research, clinicians need to read each of these. Searching and perusing such a huge repository is near impossible. In this study, PubMed abstracts relating to cervix cancer are used as the case study. In this dissertation, the On-KDT methodology is used to extract knowledge concerning clinical trials from PubMed abstracts. The knowledge thus extracted is represented in the *Clinical Knowledge Markup Language* (CKML). This approach has the potential to make effective use of relevant (and continually updated) medical knowledge contained PubMed abstracts possible, leading to potentially better clinical decisions.

In the third setting, we extract business rules from process model repositories, where process models are encoded as text artefact. Business rules encode important business constraints (including legislative and regulatory compliance constraints) as well as organizational policies. Organizations are often not adequately careful in documenting, encoding and maintaining repositories of their business rules. Instead, business rules are embedded in the design of a variety of operational artefacts, such as business process models. The ability to extract explicit business rules from such artefacts is important in

order to be able to understand, analyse, leverage, deploy and maintain business rules. This dissertation provides an application of the On-KDT methodology in extracting business rules implicit in business process designs.

The empirical results reported in this dissertation provide grounds for confidence that the On-KDT methodology may be effective, not only in the settings described above, but potentially in knowledge extraction from other unstructured or semi-structured data repositories as well.

ACKNOWLEDGEMENTS

First of all I am heartily thankful to my supervisor, Prof. Aditya K. Ghose, whose encouragement, guidance, and support from the initial to the final level enabled me to develop an understanding of the subject.

I also wish to express my appreciation to my co-supervisor, Dr. Hoa Khanh Dam, who made many valuable suggestions and gave constructive advice that improved the quality of this study. Special Gratitude goes to Dr. Andrew Miller, Clinical Associate Professor of Graduate School of Medicine, University of Wollongong, who provided me with many helpful suggestions. I express my sincere thanks to Assoc. Prof. Yi Mu who is the head of School of Computer Science and Software Engineering (SCSSE) and Dr. Lei Wang who is the acting Head of Postgraduate Studies (HPS).

I am also indebted to Evan Morrison for providing a dataset of business process models on using in my work. He is my best friend who always gives me care and moral support. I wish to thank all of my student colleagues in the Decision Systems Laboratory (DSL) for providing a stimulating and fun filled environment. My thanks to Tri A. Kurniawan, Kerry Hinge, Graham Billiau, Andrew Connery, Konstantin Hoesch-Klohe, Hongyun Xu, Katayoun Khodaei, and Chee Fon Chang.

My special thanks are due to Assoc. Prof. Chuleerat Jaruskulchai and Prof. Peter Haddawy, who developed an understanding of the research when I studied for the master degree. Also, I would like to express special thanks to Assoc. Prof. Christopher S.G. Khoo and Assist. Prof. Jin-Cheon Na, who work for the Division of Information Studies at School of Communication & Information (SCI), Nanyang Technological University, Singapore. They were my advisors when I did a research in Singapore.

Also, I wish to express my sincere thanks to Prof. Ramakoti Sadananda and Assoc. Prof. Christopher S.G. Khoo Soo Guan, who are my thesis examiners.

My special appreciation goes to my parents, Niran Polpinij and Jandeang Polpinij, who always supported and encouraged me to concentrate on my study. Also, I would like to express my heartiest thanks to my relatives, who support and encourage me when I face any problem.

Finally, I would like to express special thanks to Mahasarakham University and The Research Student Centre (RSC) of the University of Wollongong. Without their support and help, this study would not have been completed.

TABLE OF CONTENTS

ABSTRACT.....	i
ACKNOWLEDGEMENTS.....	iii
TABLE OF CONTENTS.....	iv
LIST OF FIGURES	vii
LIST OF TABLES	x
1.1 Background	13
1.2 Motivations	14
1.3 Research Questions	14
1.4 Research Contributions	15
1.5 Thesis Content.....	17
2.1 Literature Reviews	19
2.1.1 Knowledge Acquisition.....	19
2.1.2 Data Mining and Knowledge Discovery in Database	21
2.1.3 The KDD Process Model.....	21
2.1.4 Comparison of the KDD Process Models	24
2.1.5 Problems of Applying the KDD process for Informal Data	27
2.2 Improvement of Knowledge Discovery Methodology.....	29
2.3 Related Concepts and Techniques	31
2.3.1 Ontology Development	31
2.3.2 Natural Language Processing.....	39
2.3.3 N-gram.....	46
2.3.4 Text Mining	48
2.3.5 Closed-domain question answering.....	54
2.3.6 Evaluation Measures	57
3.1 The Main Concept for the On-KDT Modelling	60
3.2 The Variant Lexicon Ontology Development	62
3.2.1 Background	63
3.2.2 Definition of the VL-ontology.....	65
3.2.3 The use of the VL-ontology	67
3.2.4 How to implement the VL-ontology	68
3.2.5 Example of the use of the VL-ontology	72
3.2.6 The Experiments of the VL-ontology.....	73
3.3 The Elaboration of the ON-KDT methodology	75
3.3.1 Understanding of the application domain and defining the problem	75

3.3.2	Understanding data	76
3.3.3	Data Preparation	76
3.3.4	Closed-domain knowledge extraction	86
4.1	Background	94
4.2	Motivation	96
4.3	Research Contribution.....	96
4.4	Preliminaries: formal representation of scenarios	96
4.4.1	Formal definition of scenario	96
4.4.2	General form of scenarios	96
4.4.3	A Purposed Format of Scenario in this thesis.....	98
4.5	Extracting Scenarios from Software Requirements	99
4.5.1	Defining of the Problem and Understanding Data: A Case Study of Course Registration System.....	99
4.5.2	Software Requirement Preparation.....	102
4.5.3	Extracting scenarios by Closed-domain Knowledge Extraction	109
4.5.4	Evaluation of Discovered Scenarios.....	119
4.5.5	The Use of Discovered Scenarios.....	121
4.6	Chapter Conclusion.....	122
5.1	Background	123
5.2	Motivation	125
5.3	Research Contribution.....	125
5.4	Preliminary: The <u>cancer term net</u> ontology (CCT-net ontology)	128
5.4.1	Background of the CCT-net ontology	128
5.4.2	Definition of the CCT-net	129
5.4.3	How to implement the CCT-net	134
5.4.4	Size of the CCT-net ontology	141
5.4.5	The example of the use of the CCT-net ontology	142
5.4.6	The experiments of the CCT-net	142
5.4.7	The expected effectiveness of the CCT-net.....	144
5.5	Extracting Clinical Knowledge from PubMed Abstracts.....	144
5.5.1	The Preparation of PubMed abstracts.....	144
5.5.2	The Closed-domain knowledge extraction approached for extracting clinical trial knowledge from PubMed abstracts	149
5.5.3	Evaluation and Discussion	154
5.5.4	Evaluation and Discussion	156
5.6	Chapter Conclusion.....	157

6.1	Introduction	158
6.2	Motivation	159
6.3	Research Contribution.....	159
6.4	Preliminaries	159
6.4.1	Understanding of Business Process Characteristics	159
6.4.2	Definition of transforming the process models into sequence data forms.....	160
6.4.3	The stop-list	161
6.5	Mining Business Rules from Process Models.....	163
6.6	Dataset.....	180
6.6.1	The experiment with real dataset of process models.....	180
6.6.2	The experiment with random process models generated.....	186
6.7	Chapter Conclusion.....	194
7.1	Summary of Background and Motivation of the Thesis	195
7.2	Summary of the Proposed Methodology of the Thesis	199
7.3	Summary of the Experiments and Discussions of the Thesis	201
7.4	Summary of the Common Lesson from the Three Distinct Settings	204
7.5	Awareness in Application	205

LIST OF FIGURES

Figure 2.1 The Overview of five common processing steps in KDD model.....	23
Figure 2.2 Overview of Ontology Development Model	34
Figure 2.3 The enumeration of the concept ‘cancer’	36
Figure 2.4 The enumeration of the concept ‘extract’	36
Figure 2.5 An example of setting relationships between concepts and the class hierarchy..	37
Figure 2.6 An example of properties of the concept ‘extract’	38
Figure 2.7 An example of the XML Schema	39
Figure 2.8 The example tokenization.....	40
Figure 2.9 An example of sentence parsing	45
Figure 2.10 A Vector Space Model (also known as Bag of Words).....	52
Figure 2.11 The example of passage retrieval	58
Figure 3.1 Overview of the On- KDT Methodology	61
Figure 3.2 The information of word ‘extract’ in WordNet.....	65
Figure 3.3 An example of the word ‘block’ in the XSD format	67
Figure 3.4 An example of defining concept (or class) and the class hierarchy.....	72
Figure 3.5 An example of a result of data preparation.....	86
Figure 3.6 The process to analyse questions and identify type of answers.....	88
Figure 3.7 A binary tree representing the format of answer pattern	92
Figure 4.1 Scenario Structure (Manning and Schütze, 1999)	98
Figure 4.2 The semi-structure textual form proposed by Rolland <i>et al.</i> (1999).....	98
Figure 4.3 The purpose format based on XSD used to store scenarios.....	98
Figure 4.4 The On-KDT methodology applied to finding scenarios from software requirements	100
Figure 4.5 An example of finding the actor and action in a requirements sentence	101
Figure 4.6 A Scenarios in XML format	102
Figure 4.7 Separating a requirement document into many sentences.....	103
Figure 4.8 Automatically classifying requirements having similar needs into pre-defined classes.....	108
Figure 4.9 Analysing questions and identifying types of answers.....	110
Figure 4.10 The process to analyse questions and identify type of answers.....	111
Figure 4.11 A binary tree representing the format of answer pattern	114
Figure 4.12 An example of using the answer pattern.....	115

Figure 4.13 An example of using the answer pattern [<modal-verb> <verb> + <object> + <prepositional-phrase>]	116
Figure 4.14 The process of extracting actor and actor’s action in a requirement sentence.	118
Figure 4.15 The final results of extracting the actor and the actor’s action in the requirement sentence “ <i>Students can select four courses for the coming semester</i> ”	118
Figure 4.16 An example of scenario representation.....	119
Figure 4.17 An example of the use of scenario transforming to the use case form	121
Figure 5.1 The On - KDT methodology approach for extracting clinical trial knowledge from PubMed abstracts.....	126
Figure 5.2 An example of the associations in the mapping between term-concepts	129
Figure 5.3 The overview of the CCT-net ontology.....	129
Figure 5.4 Squamous cell carcinoma	130
Figure 5.5 Adenocarcinoma (x10)	130
Figure 5.6 Neuroendocrine carcinoma (x10)	131
Figure 5.7 An example of defining concepts and the class hierarchy of a cancer-term.....	137
Figure 5.8 The concept of cancer histology	138
Figure 5.9 The concept of disease stage	139
Figure 5.10 The concept of therapeutic modalities.....	139
Figure 5.11 The concept of timing of therapeutic modalities	140
Figure 5.12 The concept of specified drugs used.....	141
Figure 5.13 An example of MEDLINE abstract	145
Figure 5.14 An example of MEDLINE abstract	148
Figure 5.15 The method of closed-domain knowledge extraction for finding the clinical trial knowledge from PubMed abstracts	150
Figure 5.16 An example of extracting clinical knowledge from a PubMed’s abstract and storing it in the form of CKML.....	155
Figure 6.1 Examples of process models.....	160
Figure 6.2 An example of transforming process models to the VSM.....	161
Figure 6.3 All of the event elements added in the stop-element list	162
Figure 6.4 The On-KDT methodology approach for finding knowledge from process models	163
Figure 6.5 The core set of BPMN Basics.....	164
Figure 6.6 Some of Event elements	165
Figure 6.7 An example of process model constructed by using BPMN	166
Figure 6.8 An example of transforming a process model into XMI	167
Figure 6.9 An example of the process model containing the XOR gateway	168

Figure 6.10 An example of the process model containing the OR gateway	168
Figure 6.11 An example of of generating a simple process model to L_1 -sequence candidates	173
Figure 6.12 An example of generating a process model containing exclusive gateway to.	174
Figure 6.13 An example of generating a process model containing inclusive gateway to.	174
Figure 6.14 An example of generating a process model containing parallel gateway to....	175
Figure 6.15 An example of generating a process model containing exclusive gateway to.	176
Figure 6.16 Generating L_1 -sequence candidates	177
Figure 6.17 Generating L_2 -sequence, L_3 -sequence, and L_4 -sequence candidates	177
Figure 6.18 Two activity-sequence models have different names but their names are the same meaning	181
Figure 6.19 This presents the vertical and horizontal of business rule extraction	181
Figure 6.20 This presents an example of separating a process model into 3 sub-process models in order to prepare for business rules extraction in horizontal bands	182
Figure 6.21 This presents a process model that is prepared for business rules extraction in vertical bands	183
Figure 6.22 An example of the original process model generated into the form of activity-sequence	183
Figure 6.23 An example of the experimental results	184
Figure 6.24 A business rules that is extracted from the process presents the relationship between pools.....	186

LIST OF TABLES

Table 2.1 Comparison of the KDD process models and application domains.....	26
Table 2.2 Examples of phonology study.....	40
Table 2.3 Examples of inflection.....	41
Table 2.4 Examples of derivation.....	41
Table 2.5 The Penn Treebank Tag Set.....	42
Table 2.6 Examples of Rule-based POS tagging.....	44
Table 2.7 Examples of <i>n</i> -gram.....	46
Table 2.8 The classification of data mining applications.....	48
Table 2.9 The classification of question and answer.....	55
Table 2.10 The confusion matrix.....	58
Table 3.1 Stemming examples.....	63
Table 3.2 An example of the VL-ontology: the word 'block'.....	66
Table 3.3 An example of matching string between the original word and its variant.....	70
Table 3.4 An example of the BOW without the use of the VL-ontology.....	72
Table 3.5 An example of the BOW representation with the use of the VL-ontology.....	73
Table 3.6 Comparison of sentiment classification without the VL-ontology and sentiment classification with the VL-ontology.....	74
Table 3.7 Comparisons of using the VL-ontology through Naïve Bayes text classifications.....	74
Table 3.8 An example of transforming text documents to the BOW.....	78
Table 3.9 The BOW after removing stop-words.....	78
Table 3.10 The BOW after weighting terms by <i>tf-idf</i>	79
Table 3.11 Examples of parsing rules.....	88
Table 3.12 The results after using the POS tagger.....	91
Table 3.13 Examples of answering patterns.....	92
Table 4.1 The definition of each element in the XSD used to store scenarios.....	99
Table 4.2 An example of transforming requirement documents into BOW by using the VL-ontology.....	104
Table 4.3 An example of the BOW of requirement documents after removing stop-words.....	105
Table 4.4 An example of the BOW of requirement documents after weighting by <i>tf-isf</i>	106
Table 4.5 The results of automatic requirement classification by the SVM classifiers.....	108
Table 4.6 Examples of parsing rules.....	110
Table 4.7 The results of POS Tagging.....	113

Table 4.8 Examples of answering pattern for the question “Who perform each task in the system?”	114
Table 4.9 Examples of answering pattern for the question “What is a task of someone in the system?”	116
Table 4.10 The experiments for scenario extraction	120
Table 5.1 An Elaborated description of the TNM system	132
Table 5.2 Examples of utilizing the existing data to search for the associated terms of each ontological concept	136
Table 5.3 Summary of the CCT-net ontology.....	141
Table 5.4 The experimental results of the SVM and the Naive Bayes text classifiers	143
Table 5.5 Example of calculating <i>tf</i> and <i>idf</i>	146
Table 5.6 Example of calculating <i>tf-idf</i>	146
Table 5.7 The experimental results of text filter models	149
Table 5.8 Examples of question classification, corresponding answer type classification, and focus of question	151
Table 5.9 Examples of the keyword used to match and find knowledge from MEDLINE abstracts.....	152
Table 5.10 Examples of the knowledge patterns used to extract the stage of cancer in MEDLINE abstracts.....	153
Table 5.11 Some of knowledge patterns.....	154
Table 5.12 The results of extraction after comparing between the domain expert and the system.....	156
Table 6.1 Examples of transforming process models to activity-sequences.....	161
Table 6.2 The example of process models.....	169
Table 6.3 Examples of generating sequences from process models	170
Table 6.4 Examples of business rules that satisfy the support and confidence thresholds .	178
Table 6.5 The confusion matrix	179
Table 6.6 An example with the confusion matrix.....	179
Table 6.7 The evaluation of business rules that are extracted from process models	185
Table 6.8 The expected business rules.....	186
Table 6.9 The Experimental results with 400 random process models.....	188
Table 6.10 The Experimental results with 800 random process models.....	189
Table 6.11 The Experimental results with 1200 random process models.....	190
Table 6.12 The Experimental results with 400 random process models.....	191
Table 6.13 The Experimental results with 800 random process models.....	192
Table 6.14 The Experimental results with 1200 random process models.....	193

Table 6.15 The experimental results of of process model retrieval by using the actual rules
..... 194

CHAPTER 1

INTRODUCTION

1.1 Background

Knowledge is an intelligent asset in human beings. It can be defined as skills acquired, interpreted, and transformed by a range of experience, education, theorem and practice into a specific purpose. Traditionally, the manner of acquiring knowledge involves “*complex cognitive processes of perception, learning, communication, association and reasoning*”¹. In general, knowledge can be divided into two categories: tacit knowledge and explicit knowledge. The tacit knowledge is personal knowledge that is implicit. It is difficult to transfer to other persons by means of articulating in formal documents. By contrast, the *explicit knowledge* is codified knowledge that is already known. This knowledge can be written down, articulated, and stored in certain repositories such as print, digital media, or a database.

Focusing on the explicit knowledge, as this knowledge can easily write down and stored in various data repositories, it is possible that some useful explicit knowledge may be hidden in these collective data. However, in the last several decades, the problem of information overload has occurred, where advance in computer and information technology has facilitated a significant increase in an enormous amount of digital data and information because of the inexpensive repository and accessibility that are available. As the result, useful knowledge is difficult to locate out of an abundance of data or information in a short period of time.

In order to utilize these data efficiently and effectively, the modern concept of automatically finding knowledge was proposed to support the need of extracting useful knowledge from a collective data or information through computation approach. It has become a challenging task, which is called *knowledge discovery in database (KDD)*. Since the 1960s, the need of obtaining knowledge has made this task more critical than ever and it has grown in many fields (Folorunso and Ogunde, 2005, Hu et al., 2004, Elder and Pregibon, 1996, van der Aalst and Weijters, 2004, Maimon and Rokach, 2005, Hayes et al., 2005). A large body of previous research in this area indicates that there have been a growing number of knowledge discovery applications to different areas in order to discover synopsis and useful knowledge from data through correlations or patterns, and consequently

¹ knowledge acquisition (<http://www.knowmanint.com/what-is-knowledge.php>)

improve an organization's performance. Consequently, several different KDD process models have been developed and proposed since in the 1990s.

However, in reality, there are a variety of data sources. A large body of data is also available as complex unstructured data sources such text. Extracting of useful knowledge from text has become a major challenge in the area of knowledge extraction, where the study of finding knowledge in kind of unstructured data is still recognised as one of the important unsolved problems (Manuja and Garg, 2011). This is because most previous studies of knowledge discovery have focused on structured data. Therefore, to extract significant knowledge / patterns from unstructured data, such text, is extremely hard because of two main problems. Firstly, text is represented in natural language format. This leads to the problem of ambiguity that has long been a challenge in natural language processing (NLP) and information retrieval (IR). Secondly, it may not have a unified process model of discovering useful knowledge from textual data.

1.2 Motivations

The lack of a unified methodology of ontology-based knowledge discovery used to extract useful knowledge from unstructured textual data is the challenge and motivation for this investigation. Moreover, this thesis is motivated by two key drivers.

(1) It is to explore a conceptual unified methodology that may be used to extract knowledge from different informal data repositories, which is valuable as discussed above and offers the following benefits. Even if the unified methodology undergoes some change when applied to other domains, it can lead to efficiencies in reducing the diversity of processes to be supported and thereby improve the process of knowledge discovery. The proposed methodology may therefore provide a better environment leading to better solutions for extracting knowledge, where the same methodological approach to different informal data repositories can be rationalized into the same canonical solution.

(2) It is to illustrate how the proposed methodology of ontology-based knowledge discovery can be extended to extract knowledge from different informal data artefact repositories. Given the different empirical experiments, that may lead to confidence in our unified methodology if it can substantially succeed in extracting knowledge from several informal artefact repositories.

1.3 Research Questions

As suggested in Section 1.2 above, there are two main research questions to address in this thesis.

- (1) What are the main mechanisms and the methodology of ontology-based knowledge discovery which affects for extracting useful knowledge from different unstructured/semi-structured text?
- (2) Is the proposed methodology of ontology-based knowledge discovery effective for extracting useful knowledge from unstructured/semi-structured text?

1.4 Research Contributions

Based on the research questions in Section 1.3, this thesis makes two main contributions.

(1) A novel methodology of knowledge discovery is proposed in this study, where this methodology will be used to find knowledge from different informal datasets. The proposed methodology is called '*the ontology-based knowledge discovery in unstructured and semi-structured text (On-KDT)*'. To the best of our knowledge, there are many researches in the field of knowledge discover, but they have been focused on analysing structured data sources, such as spread-sheet, relational databases, and data tables (Rajman and Besançon, 1997, Soibeman et al., 2008, Mooney and Bunescu, 2005). Nevertheless, there are a variety of data sources in the real world. A large body of data is also available as complex unstructured/semi-structured data sources, such as text, digital images, and web pages (Sanchez et al., 2008, Soibeman et al., 2008). Unstructured/semi-structured data is any data without a well-defined model or schema for accessing data. These data sources cannot contain in any specified structure. This is because,

- With the large number of features of unstructured data, this is difficult to store and manage related data, such as text containing many words, and images containing thousand pixels. Therefore, it is difficult to keep these data sources in the specified structure.
- Some elements contained in the original data format may be lost during transforming the original data to a specified structure. For example, transforming images into a specified structure (e.g. single data table, spread-sheet) lead to the missing-data problem, where texture information of images may be lost.

For our methodology, the backbone of the On-KDT methodology is driven on two main components: ontology and knowledge discovery.

The ontology component is used as a semantic mechanism that will be applied to improve search accuracy for knowledge from unstructured/semi-structured text data, where the main problem in text processing is ambiguity. This problem can open up the possibility of misinterpreting the word during text processing. A main problem of ambiguity is the

change of word-forms. For instances, the word ‘*block*’ can be changed its pattern to ‘*blocking*’, ‘*blocked*’, and ‘*blocks*’. In fact, these words have the same meaning (substance), but an original mechanism (i.e. WordNet) does not support for this problem. Finally, this problem can lead the performance of text analysis and processing being poor. Later, we propose a new ontology, called the Variant Lexical Ontology (VL-ontology). This ontology can support an automatically understanding the contextual meaning of terms that appear in the search space, especially in the case of word variation.

For the knowledge discovery component, it is to discover significant knowledge from the unstructured/semi-structured data. This component consists of two principal processes: data preparation and knowledge extraction.

(2) The confidence in the knowledge discovery model is demonstrated by establishing how effective of the On-KDT methodology applied to extract hidden knowledge from different informal data is. The proposed methodology is validated via its application in three distinct settings.

In the first setting, the On-KDT methodology is applied to extract scenarios from natural language software requirements, where much of software requirements are written in a natural language format, called text-based software requirements specifications (SRSs) (Mu et al., 2009). Finding significant assets in software requirements is still recognised as one of the important problems in software engineering and requirement engineering (Carroll, 1995, Mu et al., 2009). This is because this task is time-consuming and labour-intensive, especially when this is working on a large number of software requirements and each document may contain a large volume of detail. Therefore, automatically extracting useful set of tasks from software requirement and associating them as a scenario are a challenging problem in this proposal.

In the second setting, the On-KDT methodology presents an advantage that this model has driven to extract embedded knowledge from biomedical literatures. We are interested in this matter, because the need for identifying the hidden knowledge from PubMed is required in the last decade. Many researchers also pay attention to study about how to extract knowledge from PubMed documents, where the extracted knowledge can be identified to further advance the clinical and biological studies. Also, the extracted knowledge can offer a better solution to clinical decision-making. In addition, automatic knowledge discovery allows the researchers/specialists to conduct quality work, avoid repetition, and generate new hypotheses (Merialdo, 1994). Consequently, we also apply the On-KDT methodology to find useful clinical trial knowledge relating to cervical cancer from

PubMed abstracts. This case study is a challenging problem, because this disease is one of the major public health problems for woman in the world (Shanta et al., 2000).

In the third setting, we apply the On-KDT methodology to find sequential patterns hidden in process model repositories, where process models are viewed as text. The extracted sequential tasks/activities will be considered as business rules, where business rules can be described the operations, definitions and constraints that apply to an organization. In the original work to leverage business rules, it has been done by the process mining technique. This technique mainly focuses on the derivation of significant information from event logs. However, there are settings in which process mining proves to be inadequate. If a process has never been executed, no event logs information is available, rendering the technique useless. At this, presenting in this section aims to explore an alternative dimension to process mining in which the objective is to extract process constraints (or business rules) as opposed to process models. This approach is valuable in the setting discussed above, in which process mining techniques do not return reliable results.

The expected results of extracting knowledge from each informal repository should be requirement scenarios, clinical knowledge, and business rules. From the perspective of this thesis, scenarios can be used as a main requirement in a software process, and clinical knowledge can be used to support clinical decision-making. We will also show one instance of the use of the On-KDT methodology for business rule extraction from non-textual data (specifically: process models), where process models are viewed as text.

1.5 Thesis Content

Chapter 1 is the introduction that defines the background of this investigation, including motivations, research questions, and contributions.

Chapter 2 is to review the background literature that leads to the research questions. Also, it describes the related approaches used.

Chapter 3 is concerned with the unified methodology of the ontology-based knowledge discovery (On-KDT) from unstructured and semi-structured text. This chapter also gives an overview of the methodology. It also describes the main mechanisms used.

Chapter 4 shows how the On-KDT methodology can be used to extract knowledge from natural language requirement specification repositories. The extracted knowledge is described as scenarios in a process.

Chapter 5 is concerned with the On-KDT methodology used to extract clinical knowledge from medical literatures. This section is derived from PubMed as relating to

clinical trials in the particular domain of cervix cancer in order to support clinicians for decision-making.

Chapter 6 is the extension of the On-KDT methodology used to extract useful knowledge from process model repositories. The extracted knowledge is used as business rules.

In brief, in Chapters 4 to 6 it is shown that the On-KDT methodology can be used to leverage hidden knowledge extracted from the three different types of informal artefact repositories: natural language requirement specification, medical abstract repositories, and business process models.

Finally, the conclusions are drawn in Chapter 7.

CHAPTER 2

RELATED WORKS, CONCEPTS, AND TECHNIQUES

2.1 Literature Reviews

The previous literature is reviewed in this section, in sufficient depth to facilitate an appreciation of the motivation and context of the contributions has made in this thesis.

2.1.1 Knowledge Acquisition

Knowledge is an awareness and familiarity that are gained through reasoning or inference integrated with human practices, understandings, recognitions, and experiences (Gottschalk-Mazouz, 2007). It enables new contexts to produce successful outcomes, because explicit knowledge helps a person decide what it is worthwhile. Knowledge is valuable and costly. It is a most important asset in human beings (Fayyad et al., 1996a, Fayyad et al., 1996b, Smuts et al., 2009).

With the power of knowledge, it can be distributed in a large body of several research areas (Chen et al., 2006, Beliaev and Kraslawski, 2007) such as biomedical data analysis (Blaschke et al., 1999, Marquet et al., 2003), agriculture (Pechsiri et al., 2004), business intelligence (Cantu et al., 2005), software engineering (Cook and Wolf, 1998, Chen et al., 2006), requirement engineering, and Bioinformatics (Hersh, 2005). This is because knowledge can provide solutions to make better decisions and provide better services (Clark et al., 2003, Delic and Riley, 2009). Also, it creates core competencies and introduces new situations for both individuals and organisations, now and in the future. Therefore, knowledge has proven to be successful in solving many important business and scientific problems (Apte et al., 2002, Pechsiri et al., 2004, Cantu et al., 2005, Hersh, 2005, Chen et al., 2006, Annamalai, 2006, Beliaev and Kraslawski, 2007).

Traditionally, the way of obtaining knowledge involves complex cognitive processes such as perception, learning, communication, association and reasoning. However, the traditional way of obtaining knowledge has changed during the rapid growth of computer and information technology that have brought the world into a new era of globalization, called information age. As a result, enormous amounts of data are being collected in many areas because of the advent of increasing densities and falling prices of storage devices. The vast majority of data can reside in a poorly structured collection of reports, memos, and other documents in a myriad of file formats. This has provided a challenge for automated knowledge acquisition (Fayyad et al., 1996a, Fayyad et al., 1996b, Jin et al., 1998, Fayyad,

2000), where useful knowledge can be automatically generated from data repositories through a computational approach. Therefore, during recent decades, many ways of ways of obtaining knowledge have been proposed (Fayyad et al., 1996a, Fayyad et al., 1996b, Fayyad, 2000, Jin et al., 1998). The method of generating useful knowledge from adequate data or information can be called '*knowledge extraction*' (Rao and Sprague, 1999, Yao et al., 2007).

Knowledge extraction can be defined as ways of acquiring hidden knowledge from structured (e.g. Relational databases, XML) or unstructured (e.g. Text, documents, images) sources through a computational approach (Rao and Sprague, 1999, Yao et al., 2007). It can involve a variety of techniques that combine the expertise of both humans and machines (Fayyad et al., 1996a, Fayyad et al., 1996b), while the extracted knowledge is represented in a variety of forms such as rules, attributes, structures, and patterns (Rao and Sprague, 1999). In general, knowledge extraction is computed and evaluated manually using statistical techniques (Asghar and Iqbal, 2009). Initially, traditional efforts in knowledge acquisition arose largely in the field of Information Extraction (IE) (Banko and Etzioni, 2007), where the task is to recognise entities and relations mentioned within bodies of text (Banko and Etzioni, 2007). It is noted that the word '*knowledge extraction*' is closely related to '*knowledge discovery*', where knowledge discovery describes the process of automatically searching large volumes of data for patterns that can be considered knowledge about the data. It is the most well-known branch of data mining, also known as *Knowledge Discovery in Databases (KDD)* (Fayyad et al., 1996a, Fayyad et al., 1996b).

At the beginning, finding knowledge from the wealth of collection data was done by the manual extraction method. Earlier methods of identifying knowledge in data were done by handcrafted data analysis relied on statistical analysis and inference (Elder and Pregibon, 1996) such as Bayes' theorem in the 1700s and predictive regression analysis in the 1800s. With the rapid growth in size and complexity of data in the real world, the methods of obtaining knowledge were like switching from a handcrafted data analysis environment to an automatic data processing. This has been aided by other techniques in computer science, such as artificial neural networks (ANN), cluster analysis, genetic algorithms (GA), decision trees, support vector machines (SVM), and so on. The process of applying these methods to data with the intention of uncovering hidden patterns in large data sets is called *data mining*, which is a part of the overall process of knowledge discovery (Fayyad et al., 1996a, Fayyad et al., 1996b). By this modern concept, knowledge can be automatically extracted from data and discovered knowledge is the output of analysing a dataset and it may be in the form of a pattern that is sufficiently interesting to the user and sufficiently certain.

2.1.2 Data Mining and Knowledge Discovery in Database

Data mining (DM) is the applied science and technology to explore data in order to discover previously unknown patterns (Fayyad et al., 1996a, Fayyad et al., 1996b). The accessibility and abundance of information today makes DM a matter of considerable necessity. Most of the DM techniques are based on inductive learning (Mitchell, 1997), where a model is built explicitly or implicitly by generalizing from an adequate number of training data. Alternative names of DM include knowledge extraction, information discovery, information harvesting, data archaeology, and data pattern processing (Fayyad et al., 1996a, Fayyad et al., 1996b). However, many researchers prefer DM as a synonym for knowledge discovery, where the DM can be considered as a part of the overall process of knowledge discovery (Fayyad et al., 1996a, Fayyad et al., 1996b).

Knowledge discovery in databases (KDD) is the context for development of computational techniques and tools needed to support and handle the finding useful knowledge from the rapidly growing volumes of data. Fayyad *et al.* (1996a) defined the term KDD process as “*the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data*”, while Friedman (1997) considered the KDD process as “*an automatic exploratory data analysis of large database*”. There are several researchers such as Brachman & Anand (1994), Fayyad *et al.* (1996a), Maimon & Last (2005), Reinartz (2002), and Brachman and Anand (1994) have proposed different ways to divide the KDD process into phases.

The difference of KDD and DM can be summarized following. KDD refers to the overall process of nontrivial finding useful knowledge from data. It also includes the pre-processing of the data prior to the data mining step and the evaluation of the novel patterns (or models) that make the decision of what qualifies as knowledge. Meanwhile, DM refers to the application of algorithms for finding patterns from data without the additional steps of a KDD process.

However, in the previous time, different names such as data archaeology and knowledge extraction were also referred to KDD and DM (Devedzic, 2001), where they refer to knowledge discovery from a database (e.g. Structured data, unstructured data). Knowledge extraction is often used as a synonym for KDD and DM (Devedzic, 2001).

2.1.3 The KDD Process Model

At the beginning of finding useful patterns from a dataset, the statistics is a fundamental application used to provide tools (called statistical models) for finding useful knowledge in a dataset (Fayyad et al., 1996a, Fayyad et al., 1996b, Elder and Pregibon,

1996). Therefore, it is not surprised that statistics in particular has much in common with data mining and KDD (Elder and Pregibon, 1996).

Since the 1960s, finding useful information and knowledge from data has had negative connotations in the field of statistics, when computer-based data analysis techniques were introduced. Later, several different KDD process models have been developed and proposed since in the 1990s. The main difference between the KDD process models depends on the number and scope of their specific steps.

Initially, the model of KDD is presented informally, by describing some practical needs of users (Devedzic, 2001). The KDD process combines techniques from a variety of related disciplines, Databases, Statistics, Scientific Discovery, Artificial Intelligence, and visualization (Williams and Huang, 1996). By the classic definition of KDD, the KDD process is “nontrivial and iterative of extracting useful information and knowledge from a large database” (Fayyad et al., 1996a, Fayyad et al., 1996b). It consists of multiple processing steps (one of them is DM). Each step attempts to accomplish a particular discovery task and is completed by the application of a discovery method.

Since the 1990s, many different KDD process models have been developed and proposed. The development of the KDD process model was initiated by academic research, but was quickly followed by industry (Fayyad et al., 1996a, Cabena et al., 1998, Anand and Buchner, 1998). Later, a combination of both models has been presented as a hybrid model in the early 2000s (Chapman et al., 2000).

(a) The KDD process model in the academic research

In academic research, the popular model representations include the KDD process models were developed and proposed by several researchers since in the mid-1990s. Fayyad *et al.* (1996b, 1996a) proposed the classic process of finding useful knowledge in data is interactive and iterative. This model involves nine steps to extract knowledge in data (shown as Figure 2.1).

The overview of finding useful knowledge in data can be summarized to five common steps, which are described as follows.

Selection: this step consists of understanding of problem and data, creating a target dataset, selecting data samples that will be performed in the stage of knowledge discovery.

Pre-processing: this step is to prepare the target data by using many techniques (e.g. Data cleaning) in order to obtain consistent data.

Transformation: this step consists of the transformation of the data using dimensionality reduction or transformation methods.

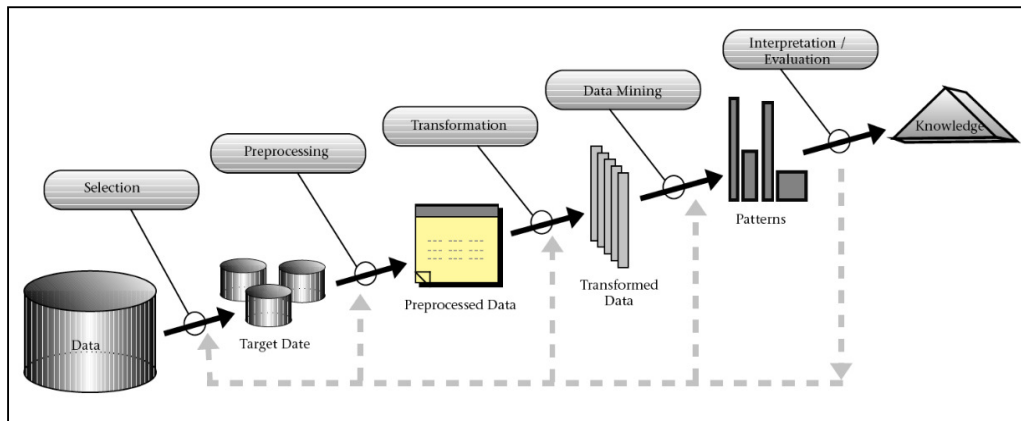


Figure 2.1 The Overview of five common processing steps in KDD model

Data Mining: this consists of the searching for patterns of interest in a particular representational form, depending on the objective of the problem.

Interpretation/Evaluation: this step is to interpret and evaluate the discovered knowledge which may in the form of patterns.

The model proposed by Fayyad *et al.* (1996b, 1996a) is not a methodology because it did not define how to do each of proposed tasks. Although it presents only a set of components in the KDD process model, it is the most popular KDD process model. This KDD process model has been incorporated into a commercial KDD system, which is called MineSet released in 1998 (Referred to <http://www.the-data-mine.com/Software/MineSet>).

The next academic models of finding useful knowledge in data, they were proposed by Cabena *et al.* (1998) and Anand & Brachman (1998). Cabena *et al.* (1998) presented the model involving five steps, while Anand & Brachman (1998) presented the model that consists of eight steps at about the same time. These models were developed by modifying the previous model with a few changes. However, all KDD process models have similar features.

(b) The KDD process model in the industrial area

Subsequently, the growth of the attention paid to industrial area, where the development of the KDD process model has been interested and adopted by industry. Some efforts are being done and established in this area. Initially, the model proposed by Cabena *et al.* (1998) was developed with the support of IBM (Kurgan and Musilek, 2006). In general, the efforts driven by industry concern the definition of processes that can conduct the implementation of the data mining (DM) application. There are many applications have been

proposed and used in practice such as SEMMA and CRISP-DM. The SEMMA (Sample, Explore, Modify, Model, Assess) was developed by the SAS Institute. It is a cycle with five processing steps that are easy to use and understand. It offers an appropriated development and maintenance of the DM tasks. Therefore, it supports for conception, creation, evaluation, and helping to present solution of business problems as well as to find the DM business goals.

The next industrial model, the CRISP-DM (Cross-Industry Standard Process for Data Mining) was developed with the goal of the support of the process model developed (Chapman et al., 2000). It was established in the late 1990s by four companies: Integral Solutions Ltd., NCR, DaimlerChrysler, and OHRA. The last two companies supported as sources of data and case study. The CRISP-DM model consists of six steps. They are business understanding, data understanding, data preparation, modelling, evaluation, and deployment, respectively. At present, the CRISP-DM model has been used in several domains such as marketing and sales. Furthermore, it has been incorporated into a commercial knowledge discovery system called Chementine (referred to SPSS Inc. <http://www.spss.com/chementine>)

(c) The KDD process model as hybrid models

As the results of the development of academic and industrial KDD process models, it has led to the development of *hybrid models* that combines aspects of both. One of popular model was proposed by Cios *et al.* (2005). This model adapted the CRISP-DM model to the needs of the academic research community. Also, it is used to provide a more general and research-oriented description of the steps. Furthermore, the hybrid model has been used in medicine and software development areas. It is included in the development of a computerized diagnostic system for cardiac SPECT image and a grid data mining methodology called GridMiner-Core (Kurgan et al., 2001). It has also been adapted to variety areas such as analysis of health and medical data (such as cystic fibrosis, data of intensive care), and image-based classification of cells (Cios et al., 2007).

2.1.4 Comparison of the KDD Process Models

Although the KDD process models proposed by Fayyad *et al.* (1996a, 1996b) is perceived as the leading academic research model and it has become a cornerstone of the later models (Cios et al., 2007), it can be found some problems. Cios & Kurgan (2005) mentioned that, this KDD process model provides the most guidance, but the step 5 (Choosing the appropriate data mining task) and 6 (Choosing the DM algorithm) are

performed too late. They should be performed during the first step (Understanding of the application domain & defining problem communitys), where the process of data preparation depends on what DM technique is already chosen, while the efficiency of the DM tool strongly depends on the outcome of data preparation. Therefore, the model proposed by Fayyad *et al* (1996a) may cause problems when choosing a DM tool since the prepared data may not be suitable for the tool. Therefore, an unnecessary feedback loop may be needed to change data preparation in the step 2, 3, and 4 (Cios and Kurgan, 2005).

Later, two KDD process models were proposed by Cabena *et al.* (1998) and Anand & Brachman (1998) in academic research model since 1998. Cabena *et al.* (1998) have presented the model involving five steps, while Anand & Brachman (1998) presented the model that consists of eight steps. The models developed by Cabena *et al.* (1998) was mentioned by Hirji (2001), where he used this model in a business domain and he found an incompleteness in the model. Also, Hirji (2001) presented that it was necessary to add one more step between data preparation and data mining, which was called a data audit. By adding another step to the model of Cabena *et al.*, it makes this model very similar to the model of Cios *et al.* (Hirji, 2001, Cios and Kurgan, 2005).

At the same time, the eight-step model by Anand & Brachman (1998) was proposed. The model was modified from the nine-step model proposed by Fayyad *et al.* (1996a) with a few changes on the life cycle. The remarkableness of this model is to identify all of the realistic steps of the KDD approach by adding the consideration of people's involvement in each process and taking into account that the target user is the data engineer. This is called Human-Centred Approach (Marbán *et al.*, 2009). Afterwards, the academic models mentioned in the above have been paid the attention by industry, and it is quickly growing in this area. The well-known industrial model is the CRISP-DM. It was developed and proposed in 1996 because a standard process is required. The main concept is, the data mining process must be reliable by people with a little data mining background. Also, it allows projects to be replicated. It also aids to project planning and management. The first version CRISP-DM 1.0 was released in 1999 (Chapman *et al.*, 2000).

The CRISP-DM is a cycle that comprises six phases: (1) Business understanding, (2) Data understanding, (3) Data preparation, (4) modelling, (5) Evaluation, and (6) Deployment. The detail of each step can be explained broadly following (Jackson, 2002).

Business understanding focuses on understanding the project objectives and requirements from a business perspective. The phase of data understanding commences with an initial data collection and familiarization, including the identification of data quality problems. The data preparation phase is all activities to construct the quality of data set from

the initial raw data. Also, this phase includes data transformation and cleaning. The modelling phase is the techniques of selection and application, including parameter calibration. Next, it is the evaluation phase that will be performed before proceeding to final deployment of the model. This phase is very important because it is to evaluate the model more thoroughly, and review the phase executed to construct the model, to be certain it properly achieves the objectives. At the end of this step, a decision on the use of the data mining results should be reached. Finally, the knowledge or information, which we gain through the data mining process, needs to be presented on the deployment phase. This phase could be as simple as creating a report or as complex as a repeatable data mining process across the organization.

Table 2.1 Comparison of the KDD process models and application domains

Models	Fayyad et al. (1996)	Cabena et al. (1998)	Anand & Brachman (1998)	CRISP-DM (Chapman et al., 2000)	Cios et al. (2000)	
Application Domains	Academic	Academic	Academic	Industry	Hybrid	
No. of Steps	9	5	8	6	6	
Processing Steps	Understanding of the application domain & defining problems	Business Objectives Determination	Human Resource Identification Problem Specification	Business Understanding	Understanding of the problem domain	
	Creating of a target data set	Data Preparation	Data Prospecting	Data Understanding	Understanding the data	
	Data cleaning & pre-processing		Domain Knowledge Elicitation	Data Preparation	Data Preparation	
	Data reduction and projection		Methodology Identification			
	Choosing the appropriate data mining task		Data Pre-processing			
	Choosing the DM algorithm	DM	DM	Pattern Discovery	Modelling	DM
	Evaluating & Interpreting mined patterns	Domain Knowledge Elicitation	Knowledge Post-processing	Evaluation	Evaluation of Discovered Knowledge	
	Using the discovered knowledge	Assimilation of Knowledge		Deployment	Using the Discovered Knowledge	

Although the purpose of the model is to generate knowledge in the data, the knowledge gained may need to be organized and presented in a way that the customer can use it. The CRISP-DM is conducted in 2002, 2004, and 2007 show that it is the leading methodology used by data miners because it can guide people to know how DM can be applied in practice in real systems (Piatetsky-Shapiro, 2002).

2.1.5 Problems of Applying the KDD process for Informal Data

Many research efforts have been focused on analysing structured data sources, such as spreadsheet, relational databases and data tables (Rajman and Besançon, 1997, Soibeman et al., 2008, Mooney and Bunesco, 2005). However, there are a variety of data sources in the real world. A large body of data is also available as complex unstructured data sources, such as text, digital images, and web pages (Sanchez et al., 2008, Soibeman et al., 2008). Unstructured data is any data without a well-defined model or schema for accessing information. These data sources cannot contain in any specified structure (Soibeman et al., 2008). This is because,

(1) With the large number of features of unstructured data, this is difficult to store and manage related data, such as text containing many words, and images containing thousand pixels. Therefore, it is difficult to keep these data sources in the specified structure.

(2) Some elements contained in the original data format may be lost during transforming the original data to a specified structure. For example, transforming images into a specified structure (e.g. single data table, spreadsheet) lead to the missing-data problem, where texture information of images may be lost.

Fundamentally, the KDD processes originally developed for traditional database are utilized to discover useful knowledge from unstructured data sources (Soibeman et al., 2008, Manuja and Garg, 2011). However, a number of existing researches for unstructured data are less studied when comparing with a number of existing researches studied for in structured data (Sanchez et al., 2008, Soibeman et al., 2008, Rajman and Besançon, 1997). This is because analysing of unstructured data needs to be aware of the complexity of the data (Wood and Ross-Kerr, 2006). This has become a major challenge in the area of knowledge extraction from unstructured data, where the study of finding knowledge in kind of unstructured data is still recognized as one of the major unsolved problems (Manuja and Garg, 2011).

Text is an unstructured data that is written in a natural language format. In fact, text may be probably the most form for knowledge repositories (Pandey and Daga, 2007). There are various domains of knowledge repositories such as business, science, medicine, and

education. It is very important for organizations working in this domain to effectively manage the unstructured data describing with text.

Analysing on unstructured textual data need to be aware of the complexity of the data (Wood and Ross-Kerr, 2006), where utilizing text data are subject to content-based analysis of natural language. It should be extremely careful in every step of the analysis of data such text because the degree of ambiguity in natural language is too rigorous (Zhang and Nunamaker, 2004). This makes it extremely hard for a computer to automatically extract useful knowledge from text repositories. Furthermore, it is time-consuming and labour-intensive (Pandey & Daga 2004), especially when working on a large number of text documents and each document may contain a large volume of information.

Unstructured textual data analysis faces not only the problem of natural language ambiguity, but also the problem of lacking a standard model to discover useful knowledge from text repositories. This is because the standards of KDD may be concerned with the processing of structured database (Rajman and Besançon, 1997, Soibeman et al., 2008, Manuja and Garg, 2011). The distinction between text and data contained in a structure is an important because it may effect for the formulation of acquiring useful knowledge (Sanchez et al., 2008). Data are something (e.g. Numeric, categorized value) with a proper model organized into a specified model/schema. Thus, it can acquire their meaning from a fully specified model/schema that is related to the physical data reality. In contrast, text is written in a natural language format that can express any meaning. It is difficult to interpret and understand by automatically computing. As the result, although the process of knowledge discovery in text has considered as an analog of the KDD process model applied to textual data, several research still efforts pay attention to study how to handle and analyse the unstructured data sources. This is because they need to meet the desired standards to find the learned patterns in text and turn them into knowledge. As the problem mentioned above, it leads to the first motivation in this thesis, where we need to propose a unified model of knowledge extraction from unstructured textual data.

However, textual data is ambiguous because this data is written in natural language. The problem of ambiguity has long been a challenge in natural language processing (NLP) and information retrieval (IR) (Goldberg and Elhadad, 2009). Therefore, we also need semantic tool(s) used to support and improve the domain of finding a useful knowledge / pattern from unstructured textual data.

2.2 Improvement of Knowledge Discovery Methodology

By definition, knowledge is defined as “*information combined with user’s ability and experience that can be used to solve a problem or to create new knowledge*” (Lee, 2005). Knowledge can be extracted from a data set through the KDD process (Fayyad et al., 1996b) or Information Extraction (IE) (Mooney and Bunescu, 2005). However, there is a main difference between KDD and IE. That is, IE is a process of finding information that is hidden in unstructured data (e.g. textual data and web data), while many researches related to KDD have been focused on analysing structured data sources. Consequently, finding useful knowledge from unstructured text is also becoming an increasingly important aspect of KDD (Mooney and Bunescu, 2005), where a large body of data is available as complex unstructured data sources, such as text, digital images, and web pages (Sanchez et al., 2008, Soibeman et al., 2008).

Although the method of KDD and IE can be a good solution of finding knowledge from a data set, some problems can be found during the process of knowledge discovery (Wimalasuriya and Dou, 2009, Wimalasuriya and Dou, 2010, Phillips and Buchanan, 2001). First, these processes may need a semantic mechanism to help and guide during the process of knowledge discovery in order to effectively achieve the discover goal (Phillips and Buchanan, 2001). Second, these processes may need to have the reuse of useful knowledge discovered in similar domains (Phillips and Buchanan, 2001). Third, a mechanism used to reduce workload of data interpretation and data transformation may be required (Phillips and Buchanan, 2001, Huang et al., 2010, Coulet et al., 2011).

As above, a solution to improve the quality of KDD and IE methodology is necessary. To the best of our knowledge, an ontology can be used to address the problems that are mentioned above (Diamantini et al., 2009a, Diamantini et al., 2009b). This is because ontology is defined as a formal and explicit specification on shared conceptualization (Gruber, 1993, Studer et al., 1998). The roles of ontologies in KDD/IE are potentially manifold, where it can be used to support for data understanding, results interpretation, and sharing over the semantic data (or model) (Luke et al., 1997, Svatek et al., 2005, Wennerberg, 2005).

In domain of IE, the ontologies can be used to investigate the information extraction process (Wimalasuriya and Dou, 2009). For example, a geopolitical ontology that defines concepts like country, province and city can be used to supplement the information extraction system described earlier. This is the general idea behind “*ontology-based information extraction (OBIE)*” (Wimalasuriya and Dou, 2010). Finally, OBIE can be described as “*A system that processes unstructured or semi-structured natural language text*

through a mechanism guided by ontologies to extract certain types of information and presents the output using ontologies” (Wimalasuriya and Dou, 2010). Early concrete tasks that have been related to this field can be found in Hwang’s work, proposed in 1999 (Hwang, 1999). His work presented the constructing of ontology from text, and he used this ontology to improve the quality of retrieving information. However, IE is methodically similar to KDD at the point of finding specific pieces of information/knowledge in a large volume of data (Mooney and Bunescu, 2005). Therefore, when an ontology is implemented in the KDD, the modification method is also called “*ontology-based knowledge discovery*” (Luke et al., 1997). In recent years, ontologies also play a crucial role in KDD and IE (Wimalasuriya and Dou, 2009, Wimalasuriya and Dou, 2010). There are many IE/KDD works have been proposed and used ontology in practice. Some of the works can be illustrated as follows.

In (Li and Bontcheva, 2007), they identify a formal ontology as one of the system inputs. Also, the ontology is used as the target output as an important characteristic that distinguishes OBIE systems from IE systems. The use of ontology in this work holds true for most OBIE systems. In (Wu and Weld, 2007), they show the OBIE systems that construct the ontology to be used through the information extraction process itself instead of treating it as an input. Although constructing a formal ontology in this case should not disqualify a system from being an OBIE system, it should be careful to remove the requirement to have ontology as an input for the system. However, the requirement to represent the output using ontologies may be reasonable.

In (Berners-Lee et al., 2001), they used ontologies to represent knowledge or meaning, where their ontologies are seen as providing the backbone for the Semantic Web. Similarly, (Ritsko and Seidman, 2004) used an ontology in their information extraction system in order to process the information contained in the World Wide Web automatically. Since a large fraction of this information takes the form of natural language text, manually processing them is becoming increasingly difficult due to their increasing volumes. However, although the success of the Semantic Web relies heavily on the existence of semantic contents that can be processed by software agents, the creation of such contents has been quite slow (Popov et al., 2003, Popov et al., 2004). Therefore, an automatic metadata generation is required. As this, (Popov et al., 2003, Popov et al., 2004) applied OBIE to provide such an automatic mechanism to generate semantic contents, called metadata by Popov et al. The main process of the proposed mechanism to produce the metadata is to convert the information contained in existing web pages into ontologies. This process is also known as the *semantic annotation* of web pages (Cimiano et al., 2004, Popov et al., 2003,

Popov et al., 2004). Then, the Semantic annotation is to tag ontology class instance data and map it into ontology classes (Reeve and Han, 2005).

Furthermore, (Alani et al., 2003) proposed a solution to bring the Semantic Web to life and provide advanced knowledge services. The proposed solution is to link a knowledge extraction tool with an ontology to achieve continuous knowledge support and guide information extraction. The extraction tool searches online documents and it extracts knowledge that matches the given classification structure. It provides this knowledge in a machine-readable format that will be automatically maintained in a knowledge base (KB). Knowledge extraction is further enhanced using a lexicon-based term expansion mechanism that provides extended ontology terminology.

In (Wennerberg, 2005), they proposed the ontology-based social network models to help explicating relationships between these entities that may not be obvious at the first glance, thereby, enabling so-called knowledge discovery. Hence, the knowledge discovered can be facilitated in a knowledge base, which can be used to track, among others, security issues on the (Semantic) Web. Their report can summarize such a system, an ontology based knowledge base that has been developed for the purpose of tracking terrorism related data on the Web.

In (Klien et al., 2006), they combine ontology-based metadata with an ontology-based search for finding geographic information services. This information will be used for estimating potential storm damage in forests. To this end, the approach is shown that through terminological reasoning the request finds an appropriate match in a service on storm hazard classes.

2.3 Related Concepts and Techniques

This section describes about the concepts and techniques relating to this work.

2.3.1 Ontology Development

This section describes ontologies, the need of ontologies, and development of ontologies.

(a) The Need of Ontology

An ontology can be viewed as “*a formal representation of explicit specification of a conceptualization with the set of concepts, relationships among concepts, and axioms about a target domain*” (Gruber, 1993). Ontologies play a major role in many fields of information technology (IT) as knowledge bearing artefacts (Guarino, 1998). This is because ontologies can help enhance the performance of IT systems in general.

Likewise, many particular IT systems such as search engine, information extraction and knowledge extraction (Handschuh and Ciravegna, 2002, Vargas-Vera et al., 2001) use ontologies to improve search accuracy for knowledge and generate more relevant results, where the usage of additional semantic knowledge in ontologies offers the benefit of providing a specification of relevant information (Yildiz and Miksch, 2007). This can be done by understanding searcher intent and the contextual meaning of terms as they appear in the searchable data-space.

In the real world, an underlying implementation in IT systems can make it possible to change their assumptions easily, if the knowledge about the domain changes (Hruby, 2005, Antunes et al., 2007). With a benefit of ontology, it can help make domain assumptions explicit. The explicit specification of domain knowledge is necessary for new users who need to learn the meaning of the terms in the domain (Karanastasi and Matsatsinis, 2010).

As above, it can be concluded that there are many benefits of the use of ontologies. It offers a shared common understanding of the structure of information among people or software agents, allowing reuse of domain knowledge, making domain assumption explicit, separating domain knowledge from the operational knowledge, and analysing domain knowledge. These are necessary for enhancing the performance of IT systems.

(b) Tradition of Ontology Elements

Ontology is the explicit formal specifications of the terms in the domain and relations among them (Hruby, 2005, Antunes et al., 2007). In general, ontology consists of four key elements (Ye et al., 2007):

(1) Class and Class Hierarchy

In general, *classes* are the main focus of most ontology. They describe concepts in the domain. So, a class is also called a '*concept*'. Typically, a class can have sub-classes representing concepts that are more specific than the super-class. This model is called the class hierarchy. There are two main possible approaches in developing a class hierarchy (Uschold and Gruninger, 1996): (1) *top-down development* and (2) *bottom-up development*.

- A *top-down model* identifies the most general classes/concepts in the domain first and subsequent specialization of the concepts (Uschold and Gruninger, 1996).

- A *bottom-up model* identifies the most specific classes/concepts for the leaves of the hierarchy first and subsequent grouping of these classes into more general concepts (Uschold and Gruninger, 1996).

(2) Relationships

This step is to provide the relation between classes (or sub-classes) that can be done by relationship rules (Ye et al., 2007). Some of the relationship rules can be illustrated as follows.

- *Is-a*

This rule means “*Class B is a subclass of A if every instance of A is also an instance of B*”. For an example, suppose the definition of the ‘*extract*’ class is ‘*to pull out*’. Therefore, if the ‘*extracted*’ is a subclass of the ‘*extract*’ class because it is a variation of the word ‘*extract*’, the definition of the ‘*extract*’ class is also the definition of the ‘*extracted*’ class. It can be represented as follows.

$$\textit{Is-a}(\textit{extracted}, \textit{extract}) \Rightarrow \textit{True}$$

- *Kind-of*

Another term to think of the taxonomic relationship is a ‘*kind-of*’ relationship. This rule can use the same meaning of the *is-a* rule. For example, ‘*extracted*’ is a *kind of* sub-class of ‘*extract*’.

- *part-of*

It is a relation that is a strict partial order relation between component concepts if they are a part of a composite concept. For example, the wheel is a part of the car, represented as follows.

$$\textit{Part-of}(\textit{wheel}, \textit{car}) \Rightarrow \textit{true}$$

(3) Slots, Value Types, and Individual Instances

Slots are used to describe properties of classes, while value describes what types of values can fill in the slot (Ye et al., 2007). The common value types of slot can be string, number, Boolean, or individual instance. It is noted that some information of a slot should not change often because they can be inherited to its sub-classes.

(4) Axioms

Ontology can be described as a set of axioms that define relationships between their concepts (Ye et al., 2007). Therefore, ontology axioms are constraints to identify classes (or concepts) and the properties of classes such as relationships between classes, and individual instances. In general, ontology axioms can be described based on *Predicate Logic*. This is

because it is an extension of *Propositional Logic*² used to represent knowledge in the area of artificial intelligence.

(c) A Guideline for Ontology Development

This section describes a method of *semi-automated learning for ontology development* shown as Figure 2.2. It is detailed below based upon the ontology components.

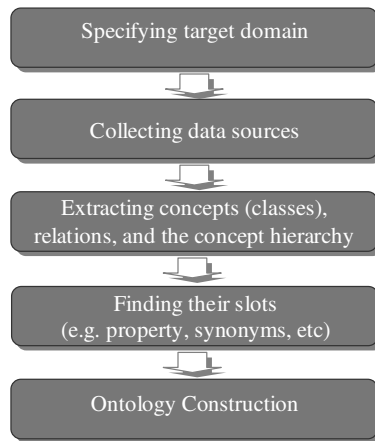


Figure 2.2 Overview of Ontology Development Model

Step 1: Specifying target domain

Before developing an ontology, it needs to be identical to understand and describe the objective for ontology development.

Step 2: Collecting relevant data sources

After obtaining a set of relevant keywords, they are used to collect the relevant data sources. There are two techniques used to collect the relevant data sources.

(1) Using a set of keywords to select and retrieve the specified relevant data collection.

(2) Reusing of pre-existing ontologies. For example, if an ontology that is relevant to cancer will be developed, the pre-existing ontologies such as Dictionary of Cancer³ can be used as a support for the new ontology.

² A propositional logic is a declarative. It is a logic in which the only objects are propositions, that is, objects which themselves have truth values. In the propositional logic, variables represent propositions, and there are no relationships, functions, or quantifiers except for the constants T and ⊥ (representing true and false respectively). The relationships are typically connective ¬, ∧, ∨, and → representing negation, conjunction, disjunction, and implication, repository. (ref: <http://planetmath.org/encyclopedia/PropositionalLogic.html>)

³ <http://www.cancer.gov/dictionary/>

Step 3: Extracting important concepts and setting concept hierarchy

This step identifies important terms as ontological concepts (or classes), and organizing the concept hierarchy in ontology. In general, the concept extraction is an important step for ontology learning. The extracted concepts should be closed to objects and relationships in the domain interest. The objects can be physical or logical. Most existing ontology learning focuses on identifying concepts and taxonomic relations (e.g. *is-a*) (Missikoff et al., 2002). In general, the noun is used to mention the concepts and the verb is used to mention the relationships between concepts. This step consists of three main operations.

(1) Ontological concept provision

This is the first step in determining important terms used as ontological concepts. In this context, there are two solutions to provide the ontological concepts. The first solution is to use the pre-existing ontology to provide the ontological concepts in a new ontology. For example, the original words that are obtained from WordNet will be re-used to produce a new ontology relevant to the lexicon. The second solution is to use the advantage of the domain expert's knowledge to provide the ontological concepts.

(2) Ontological conceptual enumeration

To get a comprehensive list of concepts/terms without worrying about redundancy between concepts, relations among the concepts, or any properties that the concepts may have, it is necessary to make statements to explain about:

- *What are the ontological concepts we would like to talk about? and/or*
- *What properties do those concepts have? and/or*
- *What would we like to say about those concepts?*

Simply speaking, the statement describing something in a specific domain is useful to write down a list of all items relating to the ontological concepts. The list of all items is called *enumeration* (Kamel and Rothenburger, 2010). For example, important cancer-related terms will include type of cancer, histology, disease stage, therapeutic modalities used, timing of therapeutic modalities used, and specified drugs used (shown as Figure 2.3).

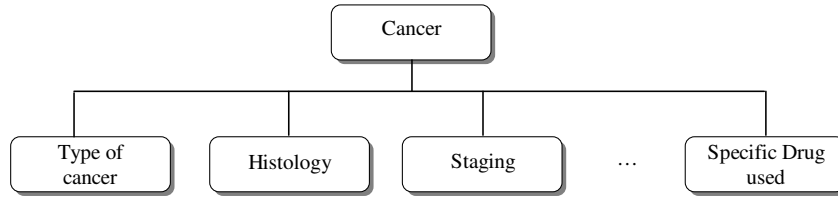


Figure 2.3 The enumeration of the concept ‘cancer’

This performance is also applied to find a set of sub-concepts relating to the relating to the main concepts. For example, the verb ‘extract’ is the original form. It can be changed to other word forms. Its variations are ‘extracts’, ‘extracting’, and ‘extracted’ (shown as Figure 2.4).

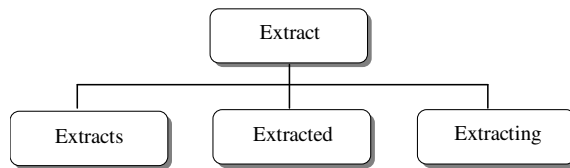


Figure 2.4 The enumeration of the concept ‘extract’

(3) Setting the relationships between concepts and the concept hierarchy

After obtaining a set of concepts, the concept hierarchy is set up by considering the relationships between concepts. In this context, one or more previous ontologies can be also used to support the process of determining the concept (or class) hierarchy.

- *Setting the relationships between concepts*

The taxonomic relationships such ‘is-a’ or ‘kind-of’ can be used to indicate the relationship between concepts, where a class *X* is a subclass of *Y* if every instance of *Y* is also an instance of *X*.

For example, the taxonomic relationships such ‘is-a’ is used to indicate the relationship between concepts, where a class *X* is a subclass of *Y* if every instance of *Y* is also an instance of *X*. It can be illustrated following. Consider the original word ‘extract’. It can be transformed into ‘extracts’, ‘extracted’, and ‘extracting’. Therefore,

$$\langle \text{extracts, extracted, extracting} \rangle \text{ is-a } \langle \text{extract} \rangle$$

These can be described as a set of axioms that define relationships between their concepts. They can be represented in predicate logic as follows.

$$\forall(x): \text{extracts}(x) \Rightarrow \text{extract}(x)$$

$\forall(x): \text{extracted}(x) \Rightarrow \text{extract}(x)$

$\forall(x): \text{extracting}(x) \Rightarrow \text{extract}(x)$

- *Setting the concept hierarchy*

After obtaining the taxonomic relationships between concepts, these can be used to determine the class hierarchy. In general, there are two approaches in developing a class hierarchy (top-down model and bottom-up model).

A *top-down* model is the process to design ontology that starts at the highest level of a design concept (as the most general concepts), and then proceeds towards the lowest level (as the specific concepts).

A *bottom-up* model is the process to design ontology that starts at the lowest level (as the specific concepts), and then proceeds towards the highest level of a design concept (as the most general concepts).

An example of the concept hierarchy can be shown as follows. Consider the relations between the verb ‘*extract*’ and its variations (*extracts*, *extracted*, *extracting*). The class hierarchy of the verb ‘*extract*’ can be shown as Figure 2.5.

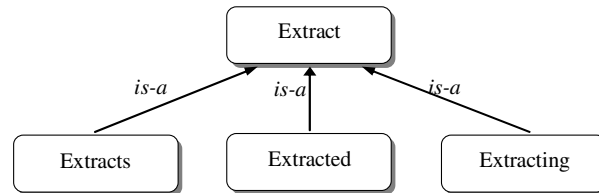


Figure 2.5 An example of setting relationships between concepts and the class hierarchy

Step 4: Defining the properties of concepts

Using only concept cannot provide enough information to answer the questions related to step 1 (specifying the target domain). Therefore, the property can be aspects that are used to describe the general term for attributes and associations relating to the concept. There are several types of object properties such as intrinsic properties, extrinsic properties, and relationships to other individuals.

- Intrinsic property is the significant information required to describe the ontological concept that is discussed. For example, types of grammar (e.g. the verb, the noun, the article) are the intrinsic property of words in WordNet.
- Extrinsic property is the additional information used to extensively describing the ontological concept. For example, WordNet also presents an example of the use of the word. This may be the extrinsic property. This is because if we know

the grammar type of a word, we may recognize how to use the word even if it has no an example.

- Relationships with other individuals present the association of the concepts. For example, the verb ‘bring’ combines with the preposition ‘in’, and then it produces the verb ‘bring in’ meaning of ‘yield’, ‘introduce’ or ‘to produce’.

An example can be presented as Figure 2.6.

```

<elements> concept = 'extract'
  <morphological_information> single word </ morphological_information > //intrinsic property
  <syntactic_information> //intrinsic property
    <class_of_word> verb </class_of_word>
    <its_suffix> -s, -ed, -ing </ its_suffix >
  </ syntactic_information >
  <semantic_information> //intrinsic property
    <ISA> pull, pull out, excerpt, take out, .. </ISA>
    <EQU> none </EQU>
  </semantic_information >
  < example sentence> Oil is extracted from olives. </ example sentence> //extrinsic property
  ...
</elements>

```

Figure 2.6 An example of properties of the concept ‘extract’

Consider Figure 2.6. The concept ‘extract’ contains following information as its properties. The morphological information is the formation of this word. In this case, it shows that the original form of the word ‘extract’ is a single word. In syntactic information, the class of word ‘extract’ is the verb in the English grammar classifications, and the suffixes ‘-s’, ‘-ed’, and ‘-ing’ effect for its variant form. The semantic information presents its synonyms (ISA) and antonyms (EQU). These facts are intrinsic properties. However, an example of the sentence may be extrinsic property because the use of the verb ‘extract’ depends on the context around the word. This effects for the variations of the verb ‘extract’.

(d) Ontology Language

Ontology language is a formal language used to implement the ontology (Corcho and Gómez-Pérez, 2000). There are many ontology languages that have been established such as Knowledge Interchange Format (KIF), XML Schema (XSD), Resource Description Methodology (RDF), and XML Topic Maps (XTM) (Lee and Chu, 2000). Meanwhile, other ontology language such as Common Logic, Web Ontology Languages (OWL), and Ontology Definition Metamodel (ODM) are used to merge languages.

In this context, the XML Schema Definition (XSD) is applied. XSD is a standard form that can be used to construct an ontology (Murata et al., 2005). It is an XML-based

alternative to a Document Type Definition (DTD). An XML schema describes the structure of an XML document. As this, the XML Schema language also refers to XSD. An example of XSD format can be shown as Figure 2.7.

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="note">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="to" type="xs:string"/>
        <xs:element name="from" type="xs:string"/>
        <xs:element name="heading" type="xs:string"/>
        <xs:element name="body" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

Figure 2.7 An example of the XML Schema

XSD is chosen as the ontology language in this work because it is relatively easy to build, where XSD uses XML Syntax (Murata et al., 2005). Therefore, it does not have to learn a new language. However, other advantages of this approach can be summarised as follows.

- (a) An XSD typically represents the inter-relationship between the attributes and elements of an XML object. Likewise, this makes a better understanding of the elements in the ontology and their relations.
- (b) This format may be suitable for computational step, especially the processing and interpreting of natural language. This is because the XSD is represented with a simple structure.

It is noted that the version of XSD that is used in this work was published by the World Wide Web Consortium (W3C) in May 2001 (referred to <http://www.w3.org/XML/Schema>).

2.3.2 Natural Language Processing

Natural language processing (NLP) concerns the computational approach to analysing human language (Dale et al., 2000, Indurkha and Damerou, 2010). NLP is interdisciplinary of theoretical linguistics, computational linguistics, and artificial intelligence. To deal with the complexity of natural language, it regularly classifies computational language into seven levels (Jurafsky and Martin, 2009, Dale et al., 2000) described as follows.

(a) Tokenization

In general, it often refers to word segmentation (also known as tokenization). Word segmentation is to segment a string of characters into a set of tokens (Dale et al., 2000). It is the first and obligatory task in natural language processing because the word is a basic unit in linguistics. Consider about English. It can be delimited by white space or punctuation. An example can be presented as Figure 2.8.

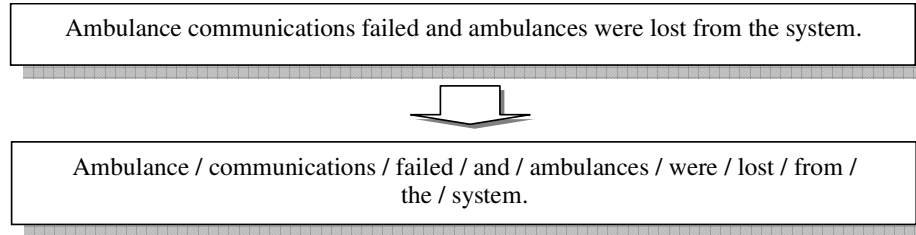


Figure 2.8 The example tokenization

It is noted that word segmentation can be more difficult in specific domains, where some terms are not contained in any standard dictionary because these terms are used in a specific domain. For example, consider the sentence following '*the outcomes of CCRT in locally advanced Cervical Cancer are considered*'. *CCRT* is a specific term in the particular domain of Cancer. This term is often used by a domain expert. Its full name is *concurrent chemoradiation therapy*.

(b) Phonological Analysis

Phonological analysis is the study of linguistic sounds (Dale et al., 2000, Indurkha and Damerau, 2010). The main problem of phonology is phonological ambiguity, where there are two words which sound the same but have different meanings. The examples of phonology study can be represented as Table 2.2.

Table 2.2 Examples of phonology study

Phonetics (sounds)	Words
/b/ + /o/ + /t/	boat
/b/ + /o/ + /th/	both

(c) Morphological Analysis

Morphological analysis is the study of the structure and formation of words, including the meaningful components of words such as the formation of the plural form from

the singular form, prefix and suffix. The morphological analysis covers the studies of stemming and recognition of ending of records. The smallest unit in morphology is called ‘*morpheme*’ which is defined as the “minimal unit of meaning”, while ‘*lexeme*’ is an abstract unit representing a set of forms taken from a single word.

Consider a word like ‘*incorrectness*’. This word consists of three morphemes and each carries a certain amount of meaning. The morpheme ‘*in*’ also means ‘*not*’, while the morpheme ‘*ness*’ means “being in a state or condition”. The morpheme ‘*correct*’ is a free morpheme because it can appear on its own (as a “word” in its own right). In general, bound morphemes will be attached to a free morpheme, and this cannot be words in their own right.

In linguistics, lexeme is an abstract unit of morphological analysis. It represents a set of form taken by a single word. A lexeme can consist of a morpheme (e.g. correct), also several morphemes (e.g. incorrectness). In fact, ‘*lexeme*’ and ‘*morpheme*’ might have the same form, but they are different concepts.

In morphological analysis, inflection and derivation are the two most common ways of word formation.

- *Inflection* is to add an affix to an original word in order to produce a new word. However, it is not a new lexeme and the basic meaning of the word is unchanged. Some examples of inflection can be represented as Table 2.3.

Table 2.3 Examples of inflection

Original word	affix	New lexemes
perform	-s	performs
push	-ed	pushed

- *Derivation* is to add an affix to an original word in order to produce a new word. This will change the original word from one lexeme to another and change the basic meaning of the word. In this case, derivational affixes can be either prefixes or suffixes. Some examples of derivation can be represented as Table 2.4.

Table 2.4 Examples of derivation

Original word	affix	New lexemes
happy	-ness	happyness
happy	un-	unhappy

(d) Syntactic Analysis

Syntactic analysis is the study of structural relationships between words (Dale et al., 2000, Indurkha and Damerau, 2010). Syntactical analysis is a process of analysing text in order to specify the possible organization of words in sentences.

In general, POS tagging is the most important at this level. In English, English grammar classifies words into eight categories: the *verb*, the *noun*, the *conjunction*, the *pronoun*, the *adjective*, the *adverb*, the *preposition*, and the *interjection*. To identify words in a sentence, it always concerns part of speech tagging. *Part of speech tagging (POS)* is a process of assigning the words in a text with their corresponding parts of speech (Dale et al., 2000, Indurkha and Damerau, 2010). It is also called grammatical tagging. In English, traditional grammar classifies words based on eight parts of speech: the verb, the noun, the pronoun, the adjective, the adverb, the preposition, the conjunction, and the determiner.

To tag POS, it firstly needs to choose a standard set of tags to do POS tagging. An example of a standard set of tags is the Penn Treebank Tag Set (referred to <http://www.cis.upenn.edu/~treebank/>).

Table 2.5 The Penn Treebank Tag Set

List	Abbr.	Description	Examples
1.	CC	Coordinating conjunction	and, or
2.	CD	Cardinal number	First, 1
3.	DT	Determiner	the, an, a, that
4.	EX	Existential there	There is
5.	FW	Foreign word	d'hoevre
6.	IN	Preposition or subordinating conjunction	in, of, like
7.	JJ	Adjective	green
8.	JJR	Adjective, comparative	greener
9.	JJS	Adjective, superlative	greenest
10.	LS	List item marker	(1)
11.	MD	Modal	could, will
12.	NN	Noun, singular or mass	car, computer
13.	NNS	Noun, plural	Cars, computers
14.	NP	Proper noun, singular	John, Michelle
15.	NPS	Proper noun, plural	actors

Table 2.5 (cont')

List	Abbr.	Description	Examples
16.	PDT	Predeterminer	both the students
17.	POS	Possessive ending	brother's
18.	PP	Personal pronoun	I, he, she, it
19.	PP\$	Possessive pronoun	My, his, her
20.	RB	Adverb	however, usually, good
21.	RBR	Adverb, comparative	better
22.	RBS	Adverb, superlative	best
23.	RP	Particle	give up
24.	SYM	Symbol	
25.	TO	to	to write, to read
26.	UH	Interjection	ahhhh
27.	VB	Verb, base form	write
28.	VBD	Verb, past tense	wrote
29.	VBG	Verb, gerund or present participle	writing
30.	VBN	Verb, past participle	wrote
31.	VBP	Verb, non-3rd person singular present	write
32.	VBZ	Verb, 3rd person singular present	writes
33.	WDT	Wh-determiner	which
34.	WP	Wh-pronoun	who, what
35.	WP\$	Possessive wh-pronoun	whose
36.	WRB	Wh-adverb	where, when

The methods of tagger can be divided into three types: rule-based POS tagging, transformation-based POS tagging and stochastic/probabilistic POS tagging.

- *Rule-based POS tagging* is the oldest technique that uses rules for tagging (Brill, 1992b). Rules can be determined manually. The main concept of a rule - based tagger is driven on a dictionary or lexicon to get possible tags for each word to be tagged. It can be concluded that this technique depends on grammatical rules. A well-known rule-based POS tagger is ENGTWOL (ENGLISH TWO Level analysis). Some examples of Rule-based POS tagging can be shown as Table 2.6.

Table 2.6 Examples of Rule-based POS tagging

Rules	Meaning	Example
<The> + <NN>	Article 'the' following with the Noun	The house
<JJ> + <NN>/<NNS>	Adjective following with the Noun (single or plural)	white house, yellow birds

- *Transformation-based POS tagging* (also known as Transformation-based learning: TBL) is to utilize a rule-based algorithm for automatic tagging of parts-of-speech to the given text (Brill, 1994). This technique transforms one state to another using transformation rules in order to find the suitable tag for each word. It also allows us to add linguistic knowledge in a readable form. The linguistic knowledge is automatically extracted from corpora. A well-known transformation-based POS tagger is called Brill tagger (Brill, 1992b). Brill rules are of the general form:

$$tag1 \rightarrow tag2 \text{ IF Condition}$$

- *Stochastic/Probabilistic POS tagging* is based on the probability of certain tag occurring given various possibilities (Church, 1988, Cutting et al., 1992). Therefore, this technique requires a training corpus to learn for probability of tag occurring. The simple technique is to choose most frequent tag in training text for each word. In general, the *stochastic/Probabilistic tagging* is based on Hidden Markov Models (HMM). HMM Taggers choose a tag sequence that maximizes this formula:

$$Tag_i = \operatorname{argmax}_j P(\text{word}|\text{tag}) \times P(\text{tag}|\text{previous } n \text{ tags}) \quad (2.1)$$

For example, there are two sentences following.

S_1 : *We need to block spam mails.*

S_2 : *There is a block on the street.*

Consider the word 'to block' in S_1 and the word 'a block' in S_2 . In fact, the word 'block' can be both verb and noun. The word 'block' followed with 'to' can be the verb or the noun. This problem can be described with the HMM presented as below.

to/TO block/???

a/DT block/???

$$Tag_{\text{race as VB}} = \operatorname{argmax} P(\text{VB}|\text{TO}) \times P(\text{'block'}|\text{VB})$$

$$Tag_{\text{race as NN}} = \operatorname{argmax} P(\text{NN}|\text{TO}) \times P(\text{'block'}|\text{NN})$$

Suppose the probability of the word ‘*block*’ performing as the noun is 0.00069, while the probability of the word ‘*block*’ performing as the verb is 0.001. Therefore, the word ‘*block*’ followed with ‘*to*’ should be the verb.

In this work, we utilize the English POS tagger. It was developed by the Stanford Natural Language Processing Group⁴. However, the level of syntactic analysis also includes determining the grammatical structure with respect to a given formal grammar. To analyse the structure of a sentence can be done by representing a sentence in a tree structure. This technique is called *parsing*. An example can be shown as Figure 2.9.

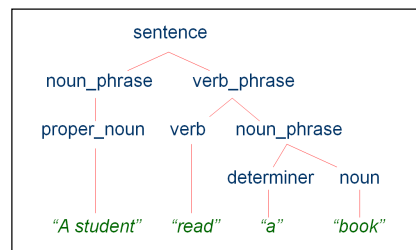


Figure 2.9 An example of sentence parsing

(e) Semantic Analysis

Semantic analysis is the study of the meaning of words/sentences (Dale et al., 2000, Indurkha and Damerau, 2010). Especially, the same word is used in different sentences, but the same event. Basically, before trying to address any semantic problem, the morphological and syntactical analysis should be completed. The morphological and syntactical analyses have focused on the structure of language, not on what things mean because these have been doing natural language processing, but not natural language understanding.

In general, semantic analysis will do about two important things. Firstly, it maps individual words into appropriate objects in the knowledge base or database. Secondly, it creates the correct structures that correspond to the way of the individual word meanings combining with each other.

The main problems of semantic analysis cover more complex tasks like: finding synonyms, word sense disambiguation, constructing question-answering systems, translating from one NL to another, populating base of knowledge.

⁴ The English POS tagger was developed by the Stanford Natural Language Processing Group (<http://nlp.stanford.edu/software/tagger.shtml>)

(f) Pragmatic and Discourse Analysis

Pragmatic analysis is the study of the use of language to accomplish goals (Dale et al., 2000, Indurkha and Damerau, 2010). To study word meaning, it also depends on context (speaker, situation) around that word. This is because it includes the area of studies that goes beyond the study of the meaning of a sentence and attempts to explain what the speaker is expressing.

Discourse analysis is the study of how the previous sentences affect the interpretation of the next sentence (Dale et al., 2000, Indurkha and Damerau, 2010). Also, it includes the study of conventions of dialogue.

(g) Knowledge –based Reasoning and Text Generation

This stage is to generate natural language from a machine representation system such as a knowledge base or a logical form. It includes content planning and surface realization (e.g. syntactic realization, Lexical choice).

2.3.3 *N*-gram

In the field of computational linguistics, *n-gram* refers to a sequence of *n* items from a given sequence of text or speech (Stolcke and Segal, 1994, Pereira et al., 1995, Brown et al., 1992, Manning and Schütze, 1999). Items in *n-gram* can be letters, phonemes, syllables, letters, and words. Regularly, the *n-grams* will be collected from a text or speech corpus. Some examples can be shown in Table 2.7

Table 2.7 Examples of *n-gram*

Type of items	Sizes	Name	Notation	Sample Sequence
character	1	unigram	w_1	r, e, p, o, r, t
character	2	bigram	w_1w_2	re, ep, po, or, rt
character	3	trigram	$w_1w_2w_3$	rep, epo, por, ort
word	1	unigram	w_1	report, of, final, work
word	2	bigram	w_1w_2	report of, of final, final work
word	3	trigram	$w_1w_2w_3$	report of final, of final work

It is noted that larger sizes are sometimes referred to by the value of *n* such as ‘4-gram’, ‘5-gram’, and so on. The *n-gram* model is widely used in statistical natural language processing. The basic idea of the statistical *n-gram* modelling is to predict the next item (such as character, word) given the previous items. If item refers to word, predicting the next

word can be estimated by the following probabilistic function, $P(w_n | w_1, w_2, w_3, \dots, w_{n-1})$ (Manning and Schütze, 1999). However, the probability of having the word w_n after the sequence of words $w_1, w_2, w_3, \dots, w_{n-1}$ can be calculated as follows.

$$P(w_n | w_1, w_2, \dots, w_{n-1}) = \frac{P(w_1, w_2, \dots, w_n)}{P(w_1, w_2, \dots, w_{n-1})} \quad 2.2$$

However, the maximum likelihood estimation (MLE) can be used to solve for the parameters that best fit the data. The MLE can be shown as follows.

$$P_{MLE}(w_1, w_2, \dots, w_n) = \frac{C(w_1, w_2, \dots, w_n)}{N} \quad 2.3$$

where $C(w_1, w_2, \dots, w_n)$ is a frequency of n -gram w_1, w_2, \dots, w_n in training text, while N is the number of training instance. As this, we can modify the formula (2.21) with the MLE and the new formula can be represented as follows.

$$P(w_n | w_1, w_2, \dots, w_{n-1}) = \frac{C(w_1, w_2, \dots, w_n)}{C(w_1, w_2, \dots, w_{n-1})} \quad 2.4$$

An example of the use of n -gram model can be illustrated following. Suppose we need to find the trigram that starts with the sequence of words ‘comes across’. Later, we have to predict the word after the sequence of words ‘comes across’. In a text database, we can count the trigram starting by the sequence of words ‘comes across’ as 10 times shown as below.

$$\begin{aligned} C(\text{comes across as}) &= 8 \\ C(\text{comes across more}) &= 1 \\ C(\text{comes across a}) &= 1 \end{aligned}$$

To predict the next word after the sequence of words ‘comes across’ can be estimated as follows.

$$\begin{aligned} P(\text{as} | \text{come_across}) &= \frac{C(\text{comes_across_as})}{C(\text{comes_across})} = 8/10 = 0.8 \\ P(\text{more} | \text{come_across}) &= \frac{C(\text{comes_across_more})}{C(\text{comes_across})} = 1/10 = 0.1 \end{aligned}$$

$$P(a | come_across) = \frac{C(comes_across_a)}{C(comes_across)} = 1/10 = 0.1$$

The probabilities of the words ‘as’, ‘more’, and ‘a’ following with the sequence of words ‘comes across’ are 0.8, 0.1, and 0.1, respectively.

The *n*-gram model offers some benefits for language model, where it is simple and easy to use and it works well to predict the words.

2.3.4 Text Mining

At present, there is too much textual information such as electronic book, web pages, email, blogs, and electronic research papers. This results in the common perception of *information overload* (Yang et al., 2003). Therefore, accessing to relevant information becomes a serious problem. Some techniques used to solve this problem are information retrieval (IR) (Baeza-Yates and Ribeiro-Neto, 1999), information extraction (IE) (Peng and McCallum, 2006), and text mining (TM) (Srivastava and Sahami, 2009, Indurkha and Damerau, 2010).

Text mining is the process of extracting interesting and non-trivial information and knowledge (e.g. Relations, patterns, and rules) from semi-structured or unstructured textual data (Indurkha and Damerau, 2010, Srivastava and Sahami, 2009). It is also known as knowledge discovery in text (KDT) (Feldman and Dagan, 1995, Brill, 1992a). Text mining is an underage field which has driven on many fields such as data mining, artificial intelligence, machine learning, information retrieval, and statistical and computational linguistics. Although text mining sometimes refers to data mining, they are quite different (Hearst, 1999). A classification of data mining and text mining can be concluded (Hearst, 1999) shown as Table 2.8.

Table 2.8 The classification of data mining applications

Types of data	Finding Patterns	Finding Nuggets	
		Novel	Non-novel
Non-textual data	Standard Data Mining	-	Database Queries
Textual data	Computational Linguistics	Real Text Data Mining	Information Retrieval

As most data or information over 80% is represented as text, text mining is believed to have a high potential value. There are outcomes of text mining. For examples, text mining

technique can detect new trends of research from previous research papers. It also extracts requirements from software specification, and detects conflicts between source code and its documentation. Text mining includes extracting and analysing of information from web sites to detect new products, trends, and the satisfaction of customers.

Types of text mining task include web mining and natural language processing (e.g. Machine translator (MT) (Uszkoreit et al., 2010), question answering (QA) (Denicia-Carral et al., 2006), automatic text summarization (Crangle, 2002), information extraction, and text/document classification and clustering). Hence, some concepts and fundamental from these areas are reviewed.

(a) Classical Categorization of Computational Language

To deal with the complexity of natural language, it regularly classifies Classical categorizations of computational language into six levels (Jurafsky and Martin, 2009)

- *Phonology* is a branch of linguistics. It studies for linguistic sounds.
- *Morphology* is to study the structure of words that relates to the meaningful parts of words.
- *Syntax* is to study the relationship between words in a sentence, including the study of sentence structure.
- *Semantics* is to study the meaning of words/sentences.
- *Pragmatics* is to study about how language is used to reach goals. This study is based on the contextual meaning.
- *Discourse* is to study about the organization and structure with larger linguistic units.

It is noted that phonology only concerns spoken language, while discourse, pragmatics, and even semantics are still rarely used.

(b) The Method of Text Mining

Text mining is a process to discover previously unknown knowledge from text (Stavrianou et al., 2007). Text is free naturally occurring. Therefore, text characteristics are ambiguous (word ambiguity, sentence ambiguity) (Li and Itoh, 1998), noisy data (e.g. Erroneous data) (Vinciarelli, 2005), unstructured format (McCallum, 2005), and high dimensionality (Dhillon et al., 2002). To obtain the quality of mining knowledge from text, it is necessary to identify more a specific individual domain before passing data to the method of text mining. This is because searching a smaller collection should improve effectiveness

and efficiency. Identifying of more specific individual data can be done by the concept of *text classification* or *text clustering*.

After we have a more specific individual domain used for study, the textual datasets will be performed through a text mining methodology (Feldman and Sanger, 2006). The process of knowledge discovery in textual data is also an analog of the KDD process model (Loh et al., 2000). In general, it consists of many processing steps described as follows.

Step 1: Text Pre-processing

Text can be stored in various repositories (e.g. web, blogs). Different repositories have also different meta-data and relation to other data. Therefore, it is necessary to prepare text data by changing raw text data into a representation that is suitable for the text mining process. This step starts with text clean-up and tokenization. Text clean-up is to remove unnecessary information (e.g. Tables and Figures), while tokenization is to separate a string of characters into a set of tokens.

Later, there are three levels of text pre-processing (Makhoul et al., 1999): (1) morphological analyses, (2) syntactical analyses, and (3) semantic analyses. However, each text mining task may not need to prepare raw data by performing every level of text pre-processing.

(1) Morphological Analysis

Morphology is a part of linguistics which deals with words. In general, it often refers to word segmentation (also known as tokenization) and stemming (Makhoul et al., 1999, Dale et al., 2000). Word segmentation is to segment a string of characters into a set of tokens (Dale et al., 2000). It is the first and obligatory task in natural language processing because the word is a basic unit in linguistics. In English, it can be delimited by white space or punctuation.

(2) Syntactic Analysis

Syntactical analysis stage requires full understanding of the sentence by identifying words in a sentence into eight categories (the *verb*, the *pronoun*, the *adjective*, the *adverb*, the *preposition*, the *conjunction*, the *noun*, and the *interjection*) (Makhoul et al., 1999, Dale et al., 2000). To identify words in a sentence, it always concerns part of speech tagging. It is a process of assigning the words in a text with their corresponding parts of speech (Dale et al., 2000).

Most works based on statistical methods have used n-gram models or Hidden Markov Model-based taggers (Church, 1988, DeRose, 1988, Cutting et al., 1992, Daelemans et al., 1995). In the rule-based approaches, the words will be assigned a tag based on a set of rules and a lexicon. These rules can be provided manually (Klein and Simmons, 1963, Daelemans et al., 1995). However, the rules can be learned (Hindle, 1989, Daelemans et al., 1995) or the transformation-based error-driven approach of Brill (1992b).

It is noted that syntactical analysis also determines the grammatical structure with respect to a given formal grammar. The structure of a sentence can be done by representing a sentence in a tree structure, called *a parsing tree* (Dale et al., 2000).

(3) Semantic Analysis

Semantic analysis is a process of understanding of the meaning of a given sentence (Makhoul et al., 1999, Dale et al., 2000). Hence, to obtain the understanding of the semantics of a sentence, it requires a tool containing meta-information used to give semantic knowledge for a sentence. A well-known tool used in natural language processing is WordNet (Miller, 1990). It focuses on a semantic lexicon which exceeds the functionality of a common dictionary or thesaurus.

Step 2: Attribute Generation

After performing by step 1, text data now are ready to be processed. Consequently, the textual data should be transformed into a suitable format for text mining in the next step, called text representation (Baeza-Yates and Ribeiro-Neto, 1999). This step includes feature selection and dimension reduction.

(1) Text Representation

Text document is represented by words (features). To transform a text document to a new form that is suitable for text mining, a well-known technique is called vector space model (VSM) (Salton et al., 1975, Baeza-Yates and Ribeiro-Neto, 1999). It is also known as a bag of words (BOW). The VSM model is one of information retrieval models (IR model) (Korfhage, 1997, Baeza-Yates and Ribeiro-Neto, 1999). It has been developed to retrieve information. This vector representation does not consider the ordering of words in a text. This vector represents the relationship between text documents and words by the number of occurrences of each word in each text document (Korfhage, 1997). Each document can be represented in a vector space, $\vec{d} = \{w_{i,1}, w_{i,2}, \dots, w_{i,t}\}$ called '*bag of words*' (BOW) (Sivic, 2009).

In information retrieval, the vector space model (or BOW) is effective because the returned documents are more relevant to our query, and they are ranked by relevance to the query. It noted that, each term should be weighted in order to evaluate how important a word is to a document in a collection (Korfhage, 1997, Baeza-Yates and Ribeiro-Neto, 1999). We will be described later about the weighting techniques.

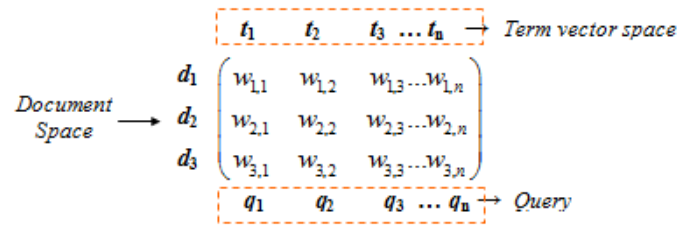


Figure 2.10 A Vector Space Model (also known as Bag of Words)

(2) Feature Selection

Feature selection is a processing step of selecting a subset of features to represent a document (Yang and Pederson, 1997, Cao and Tay, 2000). It is also a process used to reduce the dimension of datasets. In text analysis, there are two approaches used for feature selection.

- *Stop-word removal*

Stop-words are words which are considered as non-descriptive. They typically comprise prepositions, articles, etc. These words usually have very high frequency in the total corpus, and are removed prior to text mining tasks. This is because the most common words are unlikely to help text mining.

- *Weighting feature*

Term frequency-inverse document frequency (tf-idf) is the most common weighting method that has been widely used for determining the weight of a term in the vector space model (Baeza-Yates and Ribeiro-Neto, 1999, Soucy and Mineau, 2005). The term weighting score indicates the number of occurrences of each feature within a document.

In general, *tf-idf* is a weight often used in information retrieval and text mining (Yang and Pederson, 1997, Soucy and Mineau, 2005). This technique is the statistical measure used to evaluate how important a word is to a document in a collection.

The term frequency $tf_{t,d}$ of term t in document d is defined as the number of times that t occurs in d . Relevance does not proportionally increase with term frequency, while the

score is 0 if none of the query terms are present in the document. However, the estimate of $tf_{i,j}$ can be calculated by:

$$tf_{i,j} = 1 + \log(\text{frequency of term}) \quad (2.5)$$

Meanwhile, idf can be calculated from the document frequency df , which is the number of documents in which term t occurs. It is described as follows.

$$idf_t = 1 + \log(|D|/df_t) \quad (2.6)$$

where $|D|$ is the number of documents in the database.

The $tf-idf$ weight of a term is the product of its tf weight and its idf weight and $tf-idf$ is defined as follows.

$$tf-idf = tf \times idf \quad (2.7)$$

A document containing such a term is more probably to be relevant than a document that does not, but it is not an indicator of relevance. Therefore, df_t is the number of documents that contain t and df is a measure of the informativeness of t . For idf_t , it can be calculated from the document frequency df_t , which is the number of documents in which term t occurs. It is described as follows.

$$idf_t = 1 + \log(|D|/df_t) \quad (2.8)$$

where $|D|$ is the number of total documents in the collection and $tf-idf$ is defined as follows.

$$tf-idf = tf \times idf \quad (2.9)$$

Step 3: Text mining task

There are several text mining tasks such as document clustering/classification, information extraction, text summarization, text search engine, and feature extraction (e.g. named entity, relation extraction, and term extraction) (Feldman and Sanger, 2006). Some of text mining can be broadly defined as follows.

- *Document classification* (Stavrianou et al., 2007, Joachims, 1998, Joachims, 1999) is to assign one or more predefined categories to documents. It can be used for document filtering. Also, the document classification is used as data pre-processing mechanism for other text mining task such as information extraction and machine translation.

- *Document clustering* is to group similar text documents together into clusters (Ji and Xu, 2006).
- *Information extraction* is to find and extract relevant information from unstructured textual documents (Rahardjo and Yap, 2001). It also involves computational linguistics and natural language processing. In general, this task will commence with identifying a few nuggets of important text, and pull them out.
- *Text summarization* is a process of summarizing a text, where a text is input to the system and a summarized text is returned (Hovy and Lin, 1998). The summarized text is a non-redundant extract from the original text.
- *Named Entity (NE)* typically involves the identification of *proper names* in texts (Guo et al., 2009), and classifies them into a set of pre-defined classes. The proper names can be person, location, and organization. NE is used as a component technology in other areas such as information extraction, question answering, text mining, and text summarization.

Step 4: Performance Evaluation

This step is to evaluate the mined patterns (e.g. Rails, models, and etc.) that should regard to the goal determined in the first step. Also this step concentrates on the benefit of the patterns (Maimon and Rokach, 2005). We will describe later about how to evaluate the mined patterns in the next section.

2.3.5 Closed-domain question answering

Closed-domain question answering (QA) is a form of question answering that deals with questions over a specific domain. In general, QA is used to extract answers (as knowledge) in natural language, and QA techniques provide an answer to a question expressed in natural language (Dale et al., 2000, Hirschman and Gaizauskas, 2001). However, in this context, closed-domain QA is used not only for extracting answers from text, but also to extract answers from other data forms such as the graphical representation. In general, a question answering system consists of three main modules: *question analysis*, *passage retrieval and scoring*, and *answer extraction*.

Module 1: Question Analysis

This stage is to analyse the question to create a query. In general, this module requires two main tasks of analysis, as follows:

(1) Identifying types of answers

This step identifies types of answers driven by analysing question categories (e.g. what, when, who, how, etc) for distinguishing between different processing strategies and/or answer formats (Hovy et al., 2000, Hermjakob et al., 2002). The results of this indicate what the question might be looking for, called *question focus*. For example, the ‘*who*’ questions focus on ‘*person*’ and the *answer type* of this question category may be ‘*person name*’. Most systems utilize a combination of hand-crafted rules and supervised machine learning to determine the exact answer type for a question (Hovy et al., 2000). The classification of question and answer can be represented as Table 2.9.

Table 2.9 The classification of question and answer

Question Classification	Sub-class of Question	Answer Types	Example
Who	-	Person	Who won the
Why	-	Reason	Why don't we have enough rain this year?
When	-	Date/time	When did rain come yesterday?
Which	Which-who	Person	Which person did invent the instrument of aerology?
	Which-where	Location	Which city has the min temperature?
	Which-when	Date/time	Which month has max rain?
What	What	Whatever e.g. number, title, definition	What is the temperature of Tehran?
	What-who	Person	What is the best meteorologist in Iran?
	What-where	Location	What is the capital of Iran?
	What-when	Date/time	What year do we have max rain?

(2) Leveraging keywords from the question and formulating a query

This is a process in which the question is transformed into a query to be used during document retrieval and answer extraction (Hovy et al., 2000, Hovy et al., 2001). In addition, the keywords will be possibly expanded with semantic variations in order to provide the required context.

For example, consider the question “*Who is the current Prime Minister of Thailand?*” This question has the focus ‘*person*’. To set a query, a tool such thesaurus can be used to find alternatives for the keyword ‘*Prime Minister*’. This keyword can be expanded into ‘*PM*’ and ‘*leader*’. These terms are called ‘*query expansion*’. This supports better performance in both precision and recall (Wu et al., 2008). Finally, all of the keywords will be represented in a vector used as the set of queries.

Module 2: Passage retrieval and scoring

Passage retrieval is a crucial component in the question answering system. It aims to search for more precise and compressed text excerpts in response to queries, rather than giving complete documents. If a passage retrieval module returns too many irrelevant passages, the answer extraction module is likely to fail to detect the correct answer due to too much noise (Lin et al., 2003). Therefore, a passage can sufficiently answer a question.

The traditional passage retrieval systems search only for keywords. For example, consider the question “*Who is the current Prime Minister of Thailand?*” This question focuses on ‘*Prime Minister of Thailand*’. The main keywords from the question are ‘*Prime Minister*’ and ‘*Thailand*’. However, the keyword ‘*Prime Minister*’ can be expanded to ‘*PM*’, and ‘*Leader*’. Consequently, there are four keywords for this question: ‘*Prime Minister*’, ‘*PM*’, ‘*Leader*’, and ‘*Thailand*’. Therefore, the passage that contains these keywords should be retrieved. An example can be described as follows.

Suppose there is a document containing 3 main sentences. It needs to retrieve the only sentence relevant to ‘*Prime Minister of Thailand*’. Consider Figure 3.6. The sentences having the highlight are retrieved because these sentences contain the keywords ‘*Prime Minister*’ and ‘*Thailand*’.

In general, passage retrieval components execute sets of documents and return ranked lists of passages scored with respect to query terms. It is possible that many candidate passages can be retrieved. A simple technique to rank the passages is to consider the position of the keywords. If a passage contains the keywords that are close neighbours, this passage will be retrieved firstly. Therefore, consider Figure 2.12. The first passage is ranked as the first retrieval because the words ‘*Prime Minister*’ and ‘*Thailand*’ are very close.

However, it needs an algorithm score for ranking passages. Some algorithms for ranking passages can be illustrated as follows.

MITRE: This is the simplest passage retrieval method, presented by Light *et al* (2001), and simply counts the number of terms a passage has in common with the query, where each sentence can be treated as a separate passage (Tellex et al., 2003).

MultiText: For this algorithm, passages can be scored on keyword windows. It is a density-based passage retrieval algorithm that favours short passages having many terms with high *idf* (*Inverse Document Frequency*) values (Tellex et al., 2003, Clarke et al., 2000a, Clarke et al., 2000b).

ISI: This algorithm is used to rank sentences based on their similarity to the question of weighing different features such as the exact match of proper names, match of query terms, and match of stem words (Hovy et al., 2001, Tellex et al., 2003).

IBM: This algorithm is a density-based function used by analysing the distance between questionable terms in the candidate passages (Ittycheriah et al., 2000, Ittycheriah et al., 2001, Tellex et al., 2003)

It is noted that these well-described passage retrieval algorithms were chosen from top-performing TREC-10 systems (Tellex et al., 2003).

Module 3: Answer Extraction

This module concerns the extraction of potential answers, called *candidates*, from data repositories such as text. In general, the answer extraction methods can be driven on a pattern matching form, which is a simple common technique used for finding answers. However, this task depends on the complexity of the question. This module consists of two main tasks: extracting answers and ranking answers.

(1) Extracting Answers

The passages that contain answers must be analysed based on pattern matching in order to identify the right one. The extraction also depends on the type of answers. If only an answer is extracted, it may be the right answer. However, it is possible that many answer candidates can be extracted. Therefore, to discover the right answer, these candidates should be ranked.

(2) Ranking Answers

The answer candidates can be ranked by using the definition likelihood. There are several criteria to rank answer candidates such as redundancy, and term statistics in the relevant passages (Han et al., 2006).

2.3.6 Evaluation Measures

The techniques of *recall* (*R*) and *precision* (*P*) are applied (Baeza-Yates and Ribeiro-Neto, 1999). Both of them are based on an understanding and measure of relevance. If recall

score is 1.0, it indicates that the query returns all relevant documents. If precision score is 1.0, it means that the query is so precise to retrieve the relevant and correct documents.

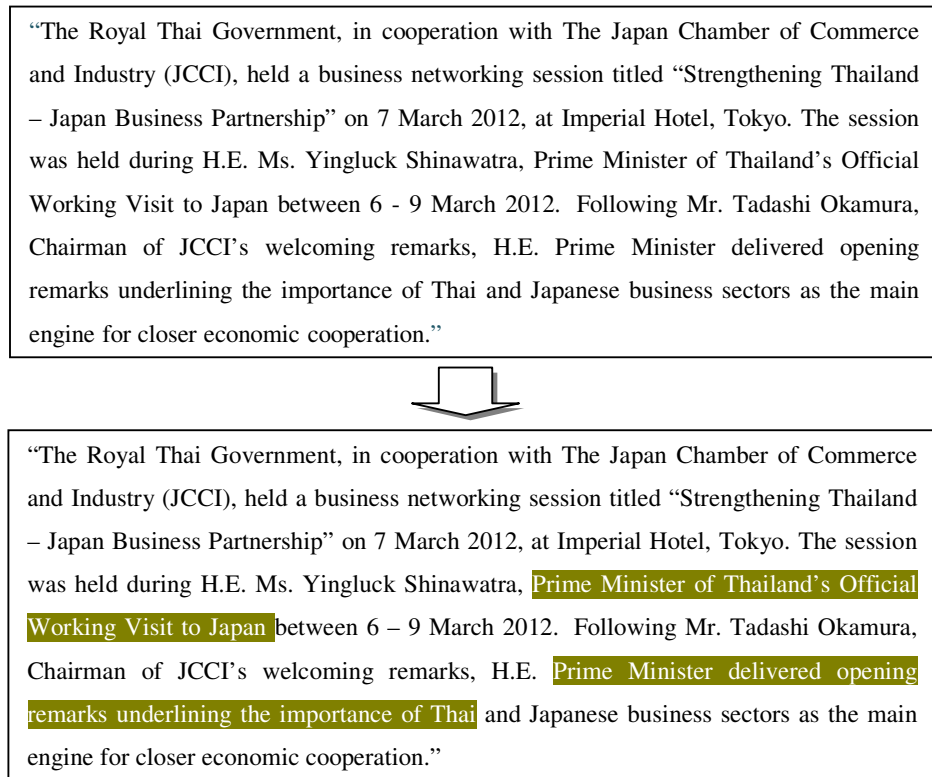


Figure 2.11 The example of passage retrieval

In this context, precision and recall are applied for effectiveness measure. The terms of true positives, true negatives, false positives, and false negatives compare the results of knowledge discovery under testing with trusted external judgments (the expected business rules). The terms ‘*positive*’ and ‘*negative*’ refer to the business rule prediction (sometimes known as the *expectation*), and the terms *true* and *false* refer to whether that prediction corresponds to the external judgment (sometimes known as the *observation*). This is illustrated by the confusion matrix below:

Table 2.10 The confusion matrix

Judgments		Expert	
		Correct	Incorrect
System	Correct	True Positive (TP)	False Positive (FP)
	Incorrect	True Negative (TN)	False Negative (FN)

By using the confusion matrix applied (Baeza-Yates and Ribeiro-Neto, 1999), precision and recall are then defined as:

$$Recall = \frac{TP}{TP + FN} \quad (2.10)$$

$$Precision = \frac{TP}{TP + FP} \quad (2.11)$$

In a good system, precision decreases as either the number of documents retrieved or recall increases. However, recall and precision are difficult for using. This is because these techniques should be used to average over large document collection and the assessments have to be binary. They also need human relevance assessments. Therefore, combined measure that assesses recall and precision trade-off is an *F - measure* (Baeza-Yates and Ribeiro-Neto, 1999). It is the weighted harmonic mean of recall and precision. Note that harmonic mean is a conservative average. It can be defined as follows.

$$F = \frac{2 \times P \times R}{P + R} \quad (2.12)$$

It is noted that these are also commonly used evaluation measure in machine learning, especially in classification work.

For example, suppose a web site contains 40 web pages, and 10 of them are about grain. A search for 'rice' might be retrieved 5 pages which are relevant, and 3 about 'rice paper' which are not relevant. The recall fraction is 0.5 (5 out of 10 truly relevant pages are found), and the precision is 0.625 (5 out of 8 pages are relevant). Finally, the *F*-measure is calculated and the result is 0.55.

CHAPTER 3

MODELLING OF ONTOLOGY-BASED KNOWLEDGE DISCOVERY FROM UNSTRUCTURED/SEMI-STRUCTURED DATA

This dissertation develops a unified conceptual process for ontology-based knowledge discovery from, and applies it in 3 distinct settings: (1) Extracting scenarios from natural language requirements, (2) Extracting rules from PubMed abstracts, (3) Extracting business rules from process models. This chapter describes the conceptual methodology, with the succeeding chapters describing the three application instances. Given that this unified model seeks to leverage the semantics available in ontologies in the KDD process, we shall henceforth refer to it as the ontology-based knowledge discovery in unstructured and semi-structured text (On-KDT) process model.

3.1 The Main Concept for the On-KDT Modelling

As noted in the previous chapter, a significant body of existing work addresses the problem of analysing structured data sources, such as spreadsheet, relational databases and data Tables (Rajman and Besançon, 1997, Soibeman et al., 2008, Mooney and Bunescu, 2005). A large body of data is also available in an unstructured form, such as text, digital images, and web pages (Sanchez et al., 2008, Soibeman et al., 2008). Unstructured data refers to data that does not conform to a well-defined model or schema. Finding useful patterns/knowledge from unstructured and semi-structured textual data is a major research challenge in the area of KDD. The problem becomes particularly acute due to ambiguity in natural language. This thesis seeks to address this problem.

The aim here is to develop an effective and efficient methodology for extracting knowledge from text, leveraging an ontology to address ambiguity in the textual sources and to improve the accuracy of the exercise by understanding the users' intent and the contextual meaning of terms as they appear in the searchable data-space. Our model is called the ontology-based knowledge discovery in unstructured/semi-structured text (On-KDT). On-KDT is an extension of the influential KDD model proposed by Fayyad *et al.* (1996) which in turn has served as the foundation for later models (Cios *et al.* 2007).

The On-KDT methodology consists of six main tasks. The first two tasks in the proposed methodology need to be done manually, while the rest are amenable to automated

support. These steps are repeated in multiple iterations. The On-KDT methodology is shown as Figure 3.1.

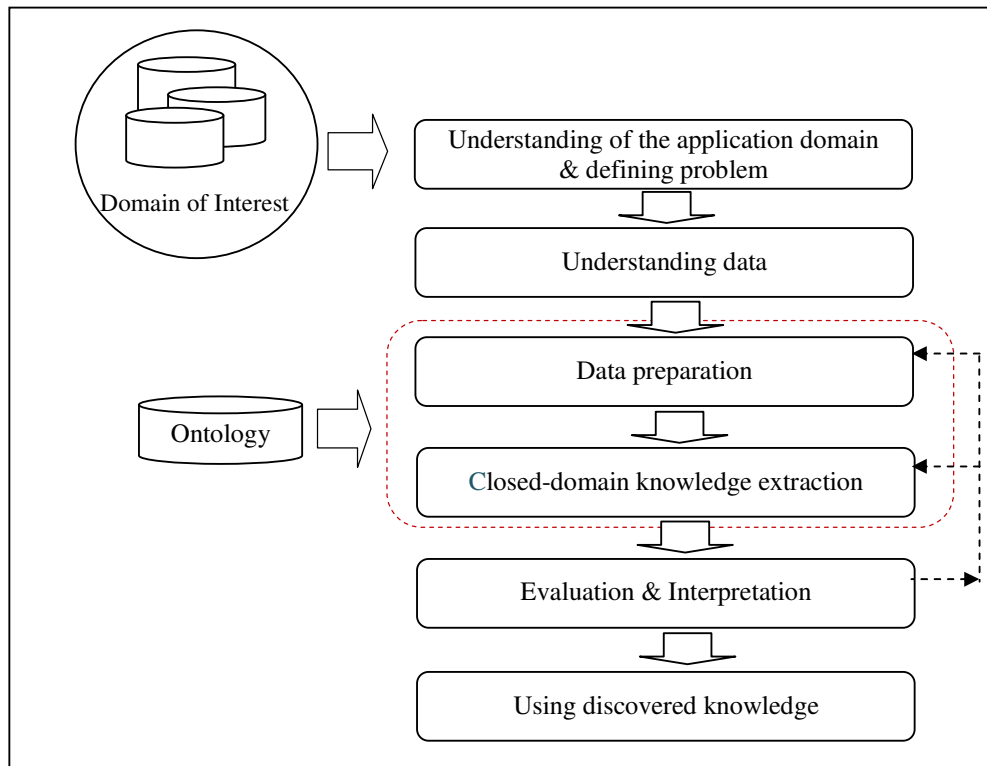


Figure 3.1 Overview of the On- KDT Methodology

In general, the On-KDT methodology commences with two main manual tasks. The first task is to determine the application domain. It includes defining the problem and the goals of the end-user. This task is important, since it identifies target data and determines a plan for knowledge discovery. In general, the task of knowledge discovery is concerned with the application of data mining techniques and tools. After we understand the objectives clearly and make sure to find out what the end-users really want to achieve, the dataset will be collected from available sources that are relevant to the specific problem domain. The second task is to understand the target data and identify data quality problems.

The subsequent tasks of the On-KDT methodology are data preparation, knowledge extraction, including knowledge evaluation and refinement. Data preparation is the process of manipulating data into a new form that is appropriate for further analysis and processing. Basically, there are several techniques for data preparation for data mining such as data selection, data cleaning, and data transformation. However, in this context, the data preparation task comprises two main processes: (1) pre-processing data and (2) identifying

more specific individual data items. Pre-processing the data includes basic operations for data cleaning such as removing noise or outliers, handling missing data fields, and collecting additional information. The second main process identifies more specific individual data items driven by the concept of data partitioning (classification and clustering). After the text dataset is re-organized the outputs are passed to the task of knowledge extraction. Having more specific individual data, it will be an easier task for knowledge extraction. Definitely, closed-domain knowledge extraction also deals with processing task under a specific domain. In general, any knowledge extraction methodology can exploit domain-specific knowledge.

However, analysing unstructured textual data need to be done extremely carefully at every step because the degree of ambiguity in natural language is high (Zhang and Nunamaker, 2004). One of the ambiguity problems in natural language is the change of word-forms, especially the *noun* and the *verb*. For example, consider the following two sentences.

S_1 : *Knowledge can be extracted from text.*

S_2 : *We extract knowledge from text.*

If these sentences are analysed manually, the words ‘*extracted*’ and ‘*extract*’ will be interpreted as having the same meaning. However, automated analysis of these sentences causes problems because the system cannot understand that the terms ‘*extracted*’ and ‘*extract*’ are related. To address this problem, semantic tools need to be used. Therefore, we also need to develop the semantic tool that will be used to address the ambiguity problem of language caused by changing of word-forms. Our semantic tool is called the Variant Lexicon Ontology (VL-ontology). The VL-ontology is used in two main tasks of the On-KDT methodology: data preparation and closed-domain knowledge extraction.

After acquiring knowledge from text data, the next step is to evaluate the patterns/knowledge discovered to make the decision of what qualifies as knowledge. Finally, it is the step of incorporating and taking action on this knowledge into the performance system. We will elaborately describe the detail of the main tasks of the On-KDT methodology later in this chapter.

3.2 The Variant Lexicon Ontology Development

This section describes the development of the main ontology integrated in the On-KDT methodology. It is called the Variant Lexicon ontology (VL-ontology).

3.2.1 Background

A main problem of ambiguity is the change of word-forms. It is well-known that traditional English grammar classifies words based on eight parts of speech (Dale et al., 2000): the *verb* (*V*), the *noun* (*N*), the *pronoun* (*Pron*), the *adjective* (*Adj*), the *adverb* (*Adv*), the *preposition* (*Prep*), the *conjunction* (*Conj*), and the *interjection* (*Int*). The original form of the noun and the verb in parts of speech can be changed. Consider the two sentences below.

S_1 : *A letter should be approved before sending to a customer.*

S_2 : *The manager approves a letter that is sent to a customer.*

If these sentences are analysed manually, they should be classified as belonging to the same group because they contain the same significant terms such as *letter*, *approved* (*approves*), *sending* (*sending*), *customer*. However, automated analysis of these sentences causes problems because the system cannot understand that the terms ‘*approved*’ and ‘*approves*’ are related (also the terms ‘*sending*’ and ‘*sent*’). We will say that ‘*approved*’ and ‘*approves*’ are lexical variants of each other. Similarly, the terms ‘*sending*’ and ‘*sent*’ are also lexical variants. The problem is *ambiguity* which has long been a challenge in natural language processing (NLP) and information retrieval (IR) (Goldberg and Elhadad, 2009). In general, there are two approaches that are used to address this problem: (1) stemming and (2) the use of pre-existing ontologies such as WordNet.

(a) Stemming

Stemming can be used to remove the effect of the inflectional form of terms/words to obtain their stem. Simply speaking, a stemming algorithm removes suffixes and/or prefixes from terms/words leaving a *stem*. The stem is a part of a word. It need not be identical to the morphological root of the word (Paice, 1994, Paice, 1996). An example of finding the stem can be seen in Table 3.1.

Table 3.1 Stemming examples

The Original Term	Stem
Engineer	Engineer
Engineering	Engineer
Engineered	Engineer

There are two objectives of the use of stemming in IR and NLP. The first objective is to reduce ambiguity in natural language caused by morphological variants (Ekmekçioglu and Willett, 2000, Sembok and Bakar, 2011), where grouping words into a common form will improve the efficiency of retrieving/finding relevant documents against a given query (Harman, 1991, Popovic and Willett, 1992). The second objective is to reduce the size of the feature space (also referred to as *dimension reduction*), where a single stem typically corresponds to several original full terms (Wessel and Renee, 1996, Harrag et al., 2010). Therefore, many IR applications treat words with the same stem as synonyms. For example in text search, it permits a search for ‘*computer*’, ‘*compute*’, ‘*computing*’, and ‘*computation*’ to find relevant documents with the common morphological stem ‘*comput*’.

However, stemming can lead to poor search accuracy, if a stemming algorithm lacks efficiency. Transforming a word to its stem opens up the possibility of mis-interpreting the word. Consider these words: *user*, *used*, *uses*, and *using*. After removing their suffixes, their stem turns out to be ‘*us*’. In general, the term ‘*us*’ may be a stop-word⁵. Because of this mistake, all of the words above (with ‘*us*’ as the stem) will be ignored during retrieval which is undesirable.

(b) Using pre-existing ontology WordNet

WordNet (Miller, 1990) can be used as a semantic tool to support text processing. It offers a large body of useful information for processing natural language such as sets of synonyms (called *synsets*), general definitions, and the various semantic relations between synonym sets. A small part of WordNet related to the word ‘*extract*’ is provided in Figure 3.2. The entry has three main elements: (1) semantic/synset relation, (2) word/lexical relation, and (3) an example sentence.

Firstly, semantic relation is a *semantic* network. It is a way of representing *relationships* between concepts. Secondly, word relation presents the word pairs from WordNet. Lastly, WordNet presents an example sentence of each word how to use the word in a sentence.

With the problem of recognizing and generating lexical variants discussed earlier in this section, it can be seen that WordNet does not directly offer any solution to this problem (Miller, 1990). Therefore, one of our key contributions is the definition of a new class of ontologies, called the Variant Lexicon Ontology (VL-ontology), as a means to address this problem.

⁵ Stop words are words which can be filtered out prior to, or after, text processing. (referred to http://en.wikipedia.org/wiki/Stop_words, <http://www.webconfs.com/stop-words.php>)

WordNet Search - 3.1
 - [WordNet home page](#) - [Glossary](#) - [Help](#)

Word to search for:

Display Options:

Key: "S." = Show Synset (semantic) relations, "W." = Show Word (lexical) relations
 Display options for sense: (gloss) "an example sentence"

Noun

- [S.](#) (n) [infusion](#), **extract** (a solution obtained by steeping or soaking a substance (usually in water))
- [S.](#) (n) [excerpt](#), [excerption](#), **extract**, [selection](#) (a passage selected from a larger work)
 "he presented excerpts from William James' philosophical writings"

Verb

- [S.](#) (v) [extract](#), [pull out](#), [pull](#), [pull up](#), [take out](#), [draw out](#), [rip out](#), [tear out](#) (remove, usually with some force or effort; also used in an abstract sense) "pull weeds"; "extract a bad tooth"; "take out a splinter"; "extract information from the telegram"
- [S.](#) (v) **extract** (get despite difficulties or obstacles) "I extracted a promise from the Dean for two new positions"
- [S.](#) (v) [educe](#), [evoke](#), [elicit](#), **extract**, [draw out](#) (deduce (a principle) or construe (a meaning)) "We drew out some interesting linguistic data from the native informant"
- [S.](#) (v) [distill](#), **extract**, [distil](#) (extract by the process of distillation) "distill the essence of this compound"
- [S.](#) (v) **extract** (separate (a metal) from an ore)
- [S.](#) (v) [press out](#), [express](#), **extract** (obtain from a substance, as by mechanical action)
 "Italians express coffee rather than filter it"
- [S.](#) (v) [excerpt](#), **extract**, [take out](#) (take out of a literary work in order to cite or copy)
- [S.](#) (v) **extract** (calculate the root of a number)

Figure 3.2 The information of word 'extract' in WordNet
 (<http://wordnet.princeton.edu/>)

3.2.2 Definition of the VL-ontology

The VL-ontology is a modification of the WordNet. This ontology has been developed based on only single words of the *noun* and the *verb* parts-of-speech. Each entry contains three main facts that are necessary for text processing: (1) morphological information, (2) syntactic information (e.g. grammar class), and (3) semantic information (synset).

Definition 1: Let W be a set of words, $W = \{w_1, w_2, w_3, w_4, \dots\}$. Each member contains significant information, where *significant information* = {*morphological, syntactic, semantic*}. Morphological information is word formation, while the syntactic information gives the grammatical classification of word and word variants. Semantic information is relevant to its synonyms and antonyms.

Definition 2: There are three types of *logical constraints* that are used to be capable of dealing with the absence of relatedness of word meanings: ISA (a kind-of), EQU (synonyms), and NEQ (antonyms). ISA is a conceptual class of a given word, while EQU is a set of words which have a

similar meaning to a given word. NEQ is a set of words which have the opposite meaning to a given word. Finally, the VL-ontology contains a set of distinct and identified concepts C that relates to a set of relations R . A dictionary D_L is an association of ontology concepts C with vocabularies set W_L that is concerned with a language L . We denote this by $D_L: C \rightarrow W_L$. Indeed, the concept c is labelled by a set of vocabularies w_1, w_2, \dots, w_n in the language L . That means, $D_L(c) = \{w_1, w_2, \dots, w_n\}$. In addition, it can determine the mutual relation $R_L: W_L \in C$ by $R_L(w) = \{c \in C \mid w \in D_L(c)\}$. Finally, the word w indicates the concepts c_1, c_2, \dots, c_n . It also denotes $R_L(w) = \{c_1, c_2, \dots, c_n\}$.

To obtain a better understanding, an example of an entry in the *VL-ontology* that defines the word ‘*block*’ is given in Table 3.2.

Consider the word example ‘*block*’ in Table 3.2. The word ‘*block*’ is represented as a concept in the *VL-ontology*. It contains three main facts. The morphological information of the word ‘*block*’ indicates word form. The syntactic information includes the grammatical classification of word (the verb and the noun) and all of possible suffixes that will make variations of the root/original word. Meanwhile, the semantic information presents *synonyms* and *antonyms*, which may be obtained from WordNet.

Table 3.2 An example of the VL-ontology: the word ‘*block*’

Word	02034	# Identify concept ID to linking
	Block	# Word
Morphological information	Single word	# Word form – Single word
Syntactic information	Class{V, N}	# Grammatical classification of word
	Suffix{-s, -ed, -ing}	# Word variation
Semantic information	ISA{prevent}	# A conceptual class of a given word
	EQU{stop}	# A word that has the similar meaning of a given word
	NEQ -	# A word that has the opposite meaning of a given word

It is noted that the *VL-ontology* is represented in an ontology language, called the XML Schema Definition (XSD) (Murata et al., 2005).

```

<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
...
...
<xs:element ID="02034">
  <xs:elementname name="block">
    <xs:elementdata>
      <xs:element name="block">
        <xs:element class="verb" type="xs:string"/>
        <xs:element synset="prevent, "stop" type="xs:string"/>
        <xs:element antonym="allow", "go" type="xs:string"/>
        <xs:element suffix="-ing", "-s", "-ed" type="xs:string"/>
        <xs:element variant="blocking, blocks, blocked" type="xs:string"/>
      </xs:elementdata>
    </xs:elementname>
  </xs:element>
...
...
</xs:schema>

```

Figure 3.3 An example of the word ‘*block*’ in the XSD format

It is noted that all the words in the VL-ontology are obtained from the WordNet. We just add more information relevant to each word, especially the additional information relevant to the verb and the noun. Therefore, the size of the VL-ontology is equivalent to the size of the WordNet.

3.2.3 The use of the VL-ontology

The VL-ontology offers meta-information which describes the word semantics. Therefore, it can be used to support the analytical natural language to reduce the problem of ambiguity. In this context, we utilize two main information from the VL-ontology. Firstly, we utilize the syntactic information (in the case of suffix information) to find a stem of word. This is because this word stem can help to have the original word. Consider the following sentences.

S_1 : A website is *blocking*.

S_2 : An inappropriate website will be ‘*blocked*’.

After tokenizing word by using the VL-ontology, the words ‘*blocking*’ and ‘*blocked*’ will be seen as the word ‘*block*’. Simply speaking, the word ‘*block*’ is the original word of ‘*blocking*’ and ‘*blocked*’.

After we have the original word, we subsequently utilize the semantic information, especially the EQU information. This information gives us two benefits.

(1) It can be used to support for word understanding. Based on Table 3.2 and Figure 3.3, an example can be illustrated following. Suppose three words appear in a document. They are ‘*prevent*’, ‘*block*’, and ‘*stop*’. By using the EQU information in the VL-ontology, a

system can understand that these words have the similar meaning, even though they may be used in different contexts.

(2) The EQU information can be used to expand keyword features in order to improve knowledge discovery performance in textual data. This concept is similar to the query expansion in document retrieval (Zukerman et al., 2003, Carpineto and Romano, 2012), where queries given by user are usually short and the natural language is inherently ambiguous. This can lead to errors and omissions during retrieving information process. Therefore, the technique of query expansion is a solution to improve retrieval effectiveness (Carpineto and Romano, 2012).

It is noted that, NEG is the word that has the opposite meaning of a given word. The NEQ information does not give us any advantage in this study. However, we present this information in the VL-ontology in order to complete our ontology. In addition, this information may be useful in our future works.

3.2.4 How to implement the VL-ontology

This section describes how to develop the VL-ontology. The model of ontology development was described in the Section 2.3.1.

Step 1: Specifying target domain

Before implementing ontology, it is necessary to understand and describe the objective for ontology development. It also narrows the domain and scope of the use of ontology that are required to identify types and sources of data. In this context, we need to develop an ontology that supports semantic analysis in natural language processing, where word variations lead to the ambiguity problem.

Step 2: Collecting relevant data sources for the VL-ontology development

After obtaining the specific target domain of ontology development, it is necessary to collect all of the relevant data sources.

(1) Existing ontologies can efficiently be reused to create a new ontology for new objectives. Therefore, the existing ontology WordNet is used as the basic design of the VL-ontology.

(2) In the case of irregular verbs, these words can be entirely changed because they are outside the standard patterns of word variants. For example, consider the word ‘*arise*’. This word can be changed to ‘*arose*’ and ‘*arisen*’. To reduce time-consuming of ontology development, these words are collected from a list of irregular verbs.

(3) In case of irregular plurals, an original word can be entirely changed when it is transformed into the plural. For example, the word ‘*man*’ is transformed into the plural ‘*men*’. These words are very hard to learn automatically. Therefore, we will collect all them from the list of irregular plurals.

(4) A document collection that will be used to acquire a concept and its sub-classes. To learn about word variation considered as sub-classes, stemming and morphological rules are used to address this problem. Stemming is used to provide the common stem, and the stem is used to produce morphological rules manually. The morphological rules are used to analyse word variation. They can be presented as follows.

<i>stem</i> ^ ‘s’	→ <i>stem</i> ^ ‘ses’
<i>stem</i> ^ ‘z’	→ <i>stem</i> ^ ‘zes’
<i>stem</i> ^ ‘sh’	→ <i>stem</i> ^ ‘shes’
<i>stem</i> ^ ‘ch’	→ <i>stem</i> ^ ‘ches’
<i>stem</i> ^ ‘x’	→ <i>stem</i> ^ ‘xes’
<i>stem</i> ^ ‘cy’	→ <i>stem</i> ^ ‘cies’

For example, the word ‘*box*’ can be transformed into ‘*boxes*’. After analysing the variation by the morphological rules, it can identify that the suffix of variant word ‘*boxes*’ is ‘-es’.

Step 3: Extracting important concepts (as classes) and setting the concept hierarchy

This step identifies important terms as ontological concepts (or classes), and organizes the concept hierarchy into an ontology.

(1) Ontological concept provision

The original words found in WordNet are used as the ontological concepts in the VL-ontology, where this work has used the WordNet synsets for word conflation and identification of concepts. However, more information (as suffix) making the variations of each word are included in the VL-ontology.

(2) Ontological conceptual enumeration

In this context, this step is to determine the words as ontological concepts and use these words to find their variants in the real world dataset. It is to automatically search the variations of the word that is discussed.

After obtaining the concepts from (1), each concept (word) is used to search for its variation by a string matching technique, called the *longest matching algorithm* (or greedy

matching)(Meknavin et al., 1997). This algorithm starts with a word span that could be another word. The method scans an input word from left to right and selects the longest match with a basic word entry at each point. In case the selected match cannot lead the algorithms to find the rest of words, the algorithm will *backtrack* to find the next longest one and continue finding the rest and so on. After matching string between variant forms and original forms, the surplus character(s) of each word is a suffix of each original word. An example is illustrated in Table 3.3.

Table 3.3 An example of matching string between the original word and its variant.

Original word	<i>Extract</i>
Variant word	<i>Extracted</i>
After matching string with the longest matching algorithm	[<i>extract</i>] + [<i>-ed</i>]

However, to evaluate a confidence of acquiring the concept and its sub-classes, we employ the Apriori association algorithm to tag the terms co-occurring between the original words and their suffixes. In this context, *minimum support (minSup)* and *minimum confidence (minConf)* are used as the criteria specified to find all variations of each word. A variant word is a pattern that states when an *original word (X)* occurs, a *suffix (Y)* occurs with certain probability.

$$\text{Support } (X \rightarrow Y) = \frac{\# \text{original word } X \text{ following with suffix } Y}{\# \text{ total of sentences}}$$

$$\text{Confidence } (X \rightarrow Y) = \frac{\# \text{original word } X \text{ following with suffix } Y}{\# \text{ total of sentences having word } X}$$

It is noted that all of the variant words extracted are considered as sub-classes (concepts) of the original word that is considered. An example can be presented as follows.

Suppose we need to find the variations of the word ‘*extract*’. We can evaluate the confidence of each variation of the word ‘*extract*’. This can be done by calculating the frequency of occurrence of the variable word in a document collection. Suppose there are seven sentences shown as follows:

*S*₁: It can be extracted important knowledge from a dataset.

*S*₂: The KDD Process for extracting useful knowledge from a large dataset.

*S*₃: Useful knowledge can be extracted from text data.

*S*₄: New knowledge can be extracted through analysing prior knowledge.

*S*₅: Useful knowledge is hidden in a large dataset.

*S*₆: Knowledge is a significant asset of humans.

S_7 : *Human knowledge is obtained by learning.*

Suppose the minSup and minConf are provided as 0.1 and 0.2 respectively. Consider the seven sentences above. We need to evaluate a confidence of the co-occurrence of ‘*extract*’ and ‘*ed*’. There are three sentences containing ‘*extracted*’. The support ratio of ‘*extracted*’ can be calculated and presented as follows.

$$\begin{aligned} \text{The support ratio of 'extract' following with 'ed'} &= 3/7 \\ &= 0.43 \end{aligned}$$

After comparing the support ratio of ‘*extract*’ following with ‘*ed*’ and the minSup, it can be found that the support ratio of ‘*extract*’ following with ‘*ed*’ is more than the minSup. Therefore, this variant word ‘*extracted*’ is still kept for further analysis.

Consequently, the confident ratio of the co-occurrence of ‘*extract*’ and ‘*ed*’ is calculated and presented as follows.

$$\begin{aligned} \text{Confident ratio of 'extract' following with 'ed'} &= 3/4 \\ &= 0.75 \end{aligned}$$

After comparing the confident ratio of ‘*extract*’ following with ‘*ed*’ and the minSup, it can be found that the confidence ratio of ‘*extract*’ following with ‘*ed*’ is more than the minConf. Therefore, it is possible that the word ‘*extracted*’ is a variation of the word ‘*extract*’. The rule of this representation can be described as follows.

$$\text{'extracted' is-a variation of 'extract' }_{[\text{Sup} = 0.75, \text{Conf} = 0.75]}$$

It is also noted that the irregular verbs do not need to evaluate a confidence of acquiring concept because these verbs are obtained from the list of irregular verbs. However, the significant information (e.g. synset) of these words is still obtained from WordNet.

(3) Setting the relationships between concepts and the concept hierarchy

After obtaining a set of concepts, the concept hierarchy is set up by considering the relationships between concepts. In this case, the taxonomic relationships, such as ‘*is-a*’ are used to indicate the relationship between concepts, where a class X is a subclass of Y if every instance of X is also an instance of Y . It can be illustrated as follows. Consider the original word ‘*block*’. It can be transformed into ‘*blocked*’, ‘*blocks*’, and ‘*blocking*’. Therefore,

$$\langle \text{blocked, blocks, blocking} \rangle \text{ is-a } \langle \text{block} \rangle$$

These can be described as a set of axioms that define relationships between their concepts. They can be represented in predicate logic as follows.

$$\forall(x): \text{blocked}(x) \Rightarrow \text{block}(x)$$

$$\forall(x): \text{blocks}(x) \Rightarrow \text{block}(x)$$

$$\forall(x): \text{blocking}(x) \Rightarrow \text{block}(x)$$

In this context, the concept hierarchy is set up based on the top-down model through consideration of the relationships between concepts. An example of the concept hierarchy can be represented as Figure 3.4.

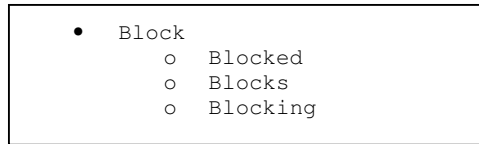


Figure 3.4 An example of defining concept (or class) and the class hierarchy.

Step 4: Finding its properties

This step adds some significant information such as properties, the grammatical classification of words, and synonyms. These can be obtained from WordNet.

Step 5: The VL-Ontology Construction

This is the step to construct the ontology by including concepts gained, their class hierarchy, and morphological, syntactic and semantic information. The VL-ontology is represented in the form of XSD.

3.2.5 Example of the use of the VL-ontology

An example can be illustrated in the domain of text search. Consider a collection that contains three sentences as below.

S_1 : A letter should be approved before sending to a customer.

S_2 : The manager approves a letter that is sent to a customer.

S_3 : A customer writes a letter of complaint.

If we does not utilize the VL-ontology for analysing variation of each term, the vector space model (or BOW) of the collection can be represented as Table 3.4.

Table 3.4 An example of the BOW without the use of the VL-ontology

Sentence	letter	approved	approves	sending	send	manager	complain	Customer
S_1	1	1	0	1	0	0	0	1
S_2	1	0	1	0	1	1	0	1
S_3	1	0	0	0	0	0	1	1

Note: 0 = absence, 1 = presence

Consider Table 3.4. It can be seen that the words ‘*approved*’ and ‘*approves*’ are considered as different words. However, these words should be considered as the same words. Therefore, if you imagine to text classification, the sentences S_1 and S_2 may be classified into different groups.

In contrast, based on the use of the VL-ontology, similar words (having the same stem/synsets and the same grammar class) will be considered as the same concept. Therefore, the words ‘*approved*’ and ‘*approves*’ will be considered as the same word. If you imagine to text classification, the sentences S_1 and S_2 will be classified into the same group. Finally, the sentences above can be represented in the BOW as 3.5.

Table 3.5 An example of the BOW representation with the use of the VL-ontology

Sentence	letter	approve	send	manager	complain	Customer
S_1	1	1	1	0	0	1
S_2	1	1	1	1	0	1
S_3	1	0	0	0	1	1

Note: 0 = absence, 1 = presence

3.2.6 The Experiments of the VL-ontology

To obtain the confidence of the use of the VL-ontology, it is evaluated through two main tasks. The first is to apply the VL-ontology to a sentiment classification. The second is to apply the VL-ontology to simplify the ambiguity of requirement specifications.

- (1) The experimental results of the use of the VL-ontology through Sentiment Classification

Sentiment classification is used to automatically analyse and classify online product reviews of consumers. It is closely related to categorization and clustering of text. Sentiment classification can be also called opinion analysis. There are several advantages of sentiment classification. For example, it can help users to quickly classify and organize things such as on-line reviews of goods and services, political commentaries, etc. Also, it can help businesses to handle 'form free' customer feedback.

We run the experiments with the online product reviews dataset that is gathered from <http://www.reviewcentre.com/>. We randomly select 6,000 the online product review documents for constructing the sentiment classifier models by the support vector machines (SVM) algorithm, while 1,000 documents are used for testing. Finally, the results are estimated by precision, recall, and F-measure (see as Table 3.6).

By using the VL-ontology in sentiment classification, we were able to classify with good accuracy after testing by F-measure. The experimental results are better than classifying sentiment data without the use of the VL-ontology.

Table 3.6 Comparison of sentiment classification without the VL-ontology and sentiment classification with the VL-ontology

Algorithm	The VL-ontology	Recall	Precision	F-measure
SVM	not used	0.81	0.78	0.794
SVM	used	0.97	0.93	0.949

- (2) The experimental results of the use of the VL-ontology through simplifying the ambiguity of requirement specifications

A major problem of software errors is introduced during the requirements phase because much of requirements specification is written in a natural language format. As this, it is hard to identify consistencies because they are too ambiguous for specification purpose. Text classification is applied to analyse and classify requirement specifications having similar detail into the same class. The initially collected requirements should be classified into various categories prior to the analysis phase in order to be usable as input in several requirements analysis methods. We ran experiments with a collection of ambulance dispatch requirements that were obtained from a commercial system. The collection contains about 250 documents. 50 documents are the problem background, 100 documents are the relevant documents to the system requirements, while the rest is the additional requirements. Therefore, the experiment in this case is to automatically classify the collection into the particular classes. We select 60 documents for constructing the classifier models by the Naïve Bayes algorithm. We randomly select 20 documents from each group used for training text classifier model. Finally, the results are estimated by precision, recall, and F-measure are presented in Table 3.7.

Table 3.7 Comparisons of using the VL-ontology through Naïve Bayes text classifications

Algorithm	The VL-ontology	Recall	Precision	F-measure
Naïve Bayes	not used	0.84	0.77	0.80
Naïve Bayes	used	0.95	0.94	0.94

As the results of text classifier and text filters, they can be evaluated with satisfactory accuracy after testing by F -measure. As the experiments above demonstrate, the VL-ontology may provide more effectiveness for automated text analysis.

3.3 The Elaboration of the ON-KDT methodology

As we said at the start of this chapter, this work aims to propose the new conceptual methodology of the ON-KDT, which strongly supports finding useful knowledge from informal data described with text.

Let M be the ON-KDT methodology. The basic concept of the ON-KDT methodology is a set of tasks, denoted as $M = \{T_1, T_2, T_3, T_4, T_5, T_6\}$. T_1 is to understand the application domain and defining problem, while T_2 is to understand data. T_3 is data preparation. T_4 is the closed-domain knowledge extraction. T_5 is the evaluation and interpretation. Finally, T_6 is the use of discovered knowledge. However, the main ontology, called the Variant Lexical Ontology (VL-ontology), will be integrated to the proposed methodology to improve the search accuracy for knowledge from text. Let O be the VL-ontology. It consists of a large number of words, $W = \{w_1, w_2, w_3, w_4, \dots\}$. Each member contains significant information, where *significant information* = {*morphological, syntactic, semantic*}. Morphological information is word formation, while the syntactic information gives the grammatical classification of word and word variants. Semantic information is relevant to its synonyms and antonyms. The result of the ON-KDT methodology may be represented as the form (D, K) . Let D be a document collection, denoted as $D = \{d_1, d_2, d_3, \dots, d_n\}$. K is useful knowledge, denoted as $K = \{k_1, k_2, k_3, \dots, k_n\}$. Finally, the ON-KDT methodology can be denoted as M and $O: D \rightarrow K$.

We describe each of these steps in turn and then continue with the following sections.

3.3.1 Understanding of the application domain and defining the problem

This initial step is to define the problem, understand the objectives clearly, and make sure to find out what the client really wants to achieve. Therefore, the description of the problem, including its restrictions, is prepared. Next, we have to assess the current situation by finding out about the resources, assumptions, constraints and other important factors which should be considered. Finally a good knowledge discovery plan has to be established to achieve the project goals. The plan should be as detailed as possible that have step-by-step to perform during the project including the initial selection of relevant techniques and tools.

3.3.2 Understanding data

This step involves collecting sample data and deciding which data, including format and size, will be needed. Some considerable tasks should be carried out including data load and data integration in order to make the data collection successful. Background knowledge can be used to guide these efforts. Data need to be checked for completeness, redundancy, missing values, and plausibility of attribute values. Finally, the step includes verification of the usefulness of the data with respect to the knowledge discovery goals.

3.3.3 Data Preparation

Data preparation is an important step describing any type of processing performed on raw data to prepare it for another procedure. It is required to be raw because data is dirty (Kotsiantis et al., 2006). A dataset may be incomplete when it lacks attribute values. The problem of dirty data includes the irrelevant data because the data has no attributes of interest or it contains only aggregate data, or the data may be noisy when it contains errors or outliers. Also, having discrepancies in the data (e.g. names, codes, or encryptions) can result in a problem of inconsistent data, while redundant or missing data can cause incorrect or even misleading results. These problems can make knowledge discovery during the training phase more difficult. Simply speaking, if the data that is used has no quality, there will also be no quality in the mining or knowledge extraction results. Therefore, the purpose of data preparation is to make the data easier to search and mine for knowledge. There are many techniques used to prepare the data before passing data to other steps.

In this context, the data preparation step consists of two main tasks: pre-processing data and identifying more specific individuals. The final result of this step is to obtain the dataset having the more specific individuals, where something more specific data effects for the ability of acquiring knowledge (Brill, 1994). This section, therefore, describes the common details of each task in the step of data preparation, as follows.

(a) Pre-processing Data

Pre-processing data is an important step describing any type of processing performed on raw data to prepare it for other procedures. It is required because data in the real world may be incomplete, irrelevant, noisy, missing, and inconsistent. In this context, some approaches of pre-processing can be described as follows.

(i) Data Transformation

The main objective of data transformation is the conversion of data from one format to another. One way to do this is by mapping the data from their given format into the format expected by the appropriate application. This includes value conversions or translation functions as well as normalizing numeric values to conform to minimum and maximum values, and aggregating data by moving up in the concept hierarchy on numeric attributes.

Due to the data used in this context describing with text, a text corpus will be tokenized and transformed into a vector, called *vector space model (VSM)* (Korfhage, 1997). This vector represents the relationship between documents and features by the number of occurrences of each feature in each text document (Korfhage, 1997). Each document can be represented in a vector space, $\vec{d} = \{w_{i,1}, w_{i,2}, \dots, w_{i,t}\}$, also known as ‘*bag of words*’ (*BOW*). However, based on the use of the VL- ontology, the similar words having the same stem (or synsets) and the same grammar class) will be considered as concept. An example can be shown as follows. Consider three documents as follows.

D₁: A manager approves a letter.

D₂: A letter approved is sent to customer.

D₃: A letter delivers to customer.

Consider the term ‘*approves*’ in *D₁* and the term ‘*approached*’ in *D₂*. By using the VL-ontology, these terms can be considered under the concept ‘*approve*’. As this, the sentences above can be represented in the BOW as Table 3.8.

It is noted that we utilize the information relating to synset and word variation that are represented as the detail in each word containing in the VL-ontology (See an example in Figure 3.3). In the case of the use of word variation, after tokenising word, each word will be verified by the VL-ontology.

Consider the term ‘*approves*’ in *D₁* and the term ‘*approached*’ in *D₂*. These words will be traced back to the original word that is ‘*approve*’. Finally, the word ‘*approve*’ is used as the word surrogate of ‘*approves*’ and ‘*approached*’.

In information retrieval, the vector space model (or BOW) is effective because the documents returned are more relevant to our query, and they are ranked by relevance to the query. It noted that, each term should be weighted in order to evaluate how important a word is to a document in a collection (Korfhage, 1997). The weight technique is described in section 2.3.4 (text mining).

Table 3.8 An example of transforming text documents to the BOW

Documents	All distinct terms in the collection								
	A	manager	approve	letter	is	sent	to	customer	Deliver
D_1	1	1	1	1	0	0	0	0	0
D_2	1	0	1	1	1	1	1	1	0
D_3	1	0	0	1	0	0	1	1	1

Note: 1 = presence, 0 = absence

(ii) Data Cleaning

The main concept of data cleaning in this context is to remove the redundant, irrelevant, and inconsistent data because these must cause incorrect or even misleading statistics during the model construction period. In this context, some terms which are in stop-words are removed, where these terms are considered as non-descriptive within the BOW. Stop-words are deemed irrelevant for searching purposes. This is because they usually have very high frequency in the total corpus, but they carry less important meaning than keywords. Therefore, they will be removed from text to return the most relevant result. Stop-words typically comprise prepositions, articles, auxiliary verbs, etc. They must be removed are removed prior to modelling step such as classification, filtering, and knowledge extraction.

It is noted that, a word that occurs only one are also removed. This is because it may not effect for the search space. As the example BOW represented in Table 3.8, it can be modified by removing the stop-words (shown as Table 3.9) in order to obtain the most relevant features.

Table 3.9 The BOW after removing stop-words

Documents	All distinct terms in the collection					
	manager	approve	letter	sent	Customer	Deliver
D_1	1	1	1	0	0	0
D_2	0	1	1	1	1	0
D_3	0	0	1	0	1	1

Note: 1 = presence, 0 = absence

(iii) Weighting terms

Consider the BOW in Table 3.9. We will show the *tf-idf* weight of each term in Table 3.9 as follows. Let $|D|$ be 3, where $|D|$ is the total number of documents in the corpus.

The *df* of the terms ‘manager’, ‘approve’, ‘letter’, ‘sent’, ‘customer’, and ‘deliver’ are 1, 2, 2, 1, 1, and 1, respectively. Therefore, the *idf* of these terms can be calculated as follows.

$$\begin{aligned}
 idf_{\text{'manager'}} &= 1 + \log(3/1) = 1.477 \\
 idf_{\text{'approve'}} &= 1 + \log(3/2) = 1.176 \\
 idf_{\text{'letter'}} &= 1 + \log(3/3) = 1 \\
 idf_{\text{'sent'}} &= 1 + \log(3/1) = 1.477 \\
 idf_{\text{'customer'}} &= 1 + \log(3/2) = 1.176 \\
 idf_{\text{'deliver'}} &= 1 + \log(3/1) = 1.477
 \end{aligned}$$

After gaining the *idf* of all of terms, *tf-idf* can be calculated as Table 3.10.

Table 3.10 The BOW after weighting terms by *tf-idf*

Documents	All distinct terms in the collection					
	manager	approve	letter	sent	customer	Deliver
D_1	1.477	1.176	1	0	0	0
D_2	0	1.176	1	1.477	1.176	0
D_3	0	0	1	0	1.176	1.477

(b) Identifying more a specific individual data

This step aims to identify a more specific individual data before delivering data to the step of knowledge extraction.

The main process of identifying more a specific individual data is driven by both clustering and classification techniques. These techniques effectively archive a partition of the data repository based on appropriate partition criteria. The selection of the appropriate technique used for data specification and simplification depends on the nature of the data. In fact, classification and clustering are similar, because the main concept of these techniques involves the estimation of the parameters in order to partition a dataset into distinct partitions, called classes. However, unlike the clustering concept, classification effectively achieves a prediction by mapping data into predefined classes. It requires the name and number of classes given in the training set, while the name and number of classes are unknown in the clustering technique. Further, classification is much less exploratory than clustering because the objective of classification is not to explore the data to discover interesting partitions, but rather to decide how new data should be classified.

It is noted that the training sets and the target classes are manually identified by domain experts. A binary class is considered in this work. They are relevant and irrelevant classes. This module relies on both techniques in order to make a flexibility of this

component, where types of dataset are different. Such partitioning can be achieved in two kinds of settings:

- *Settings with representative sets of members for each category:*

In such settings, classifiers are used as input sets of (pre-determined) training instances of each category, and produce as output a complete partitioning of the repository, with one partition per category of interest.

Text Classification is a predictive modeling technique based on inductive learning (or supervised learning) (Mitchell, 1997, Han and Kamber, 2006). It involves a dataset that is partitioned into several classes by assigning the same symbol to each member of the class. Therefore, a class contains data having similar data intervals. The basic concept of data classification may be represented as the form (X, Y) . Let X be a vector of observations. Associated to the observations X , it can have a class label y_i where X belongs. This assumes a finite set of known class Y . It means $y_i \in Y$. This is described as a set of pre-defined classes.

Given a dataset D of such data with known labels, the dataset is generally divided into two subsets. The first subset is a randomly selected subset T of D , called *training set*. It is used to construct a function by learning with a classification algorithm. The function learned is also represented as classification rule/model, called *classifier*. It is denoted by $f: X \rightarrow Y$. Finally, given classifier is used to predict the label of new/unknown data. The second subset is the remaining data is used as the *testing set* which is used to determine the accuracy of the model. The methodology of data classification in this context can be described following. In classification problem, the dataset is partitioned into two groups: training set and testing set. In a training set, each collection of data contains a set of attributes, one of which is class. The training set is passed to the stage of pre-processing data, such as data selection, and so on. Hereafter, the pre-processed data is passed to the stage of classifier model building. This is to find a model that is built by a learning algorithm (e.g. Support Vector Machines, Decision Tree, Artificial Neural Network etc) for the class attribute as a function of the values of other attributes. Finally, a test set is used to determine the accuracy of the model. Simply speaking, the test set is used to validate the model.

In this context, as it said at the beginning, this work aims to find useful knowledge from informal data describing with text. Two classification algorithms (machine learning algorithms) that are applied for building the classification models can be described based on textual data, as follows:

(1) Naïve Bayes

The Naïve Bayes (NB) algorithm is a simple probabilistic classifier based on conditional probabilities (McCallum and Nigam, 1998, Mozina et al., 2004). This algorithm has applied Bayes' theorem with strong independence assumptions. It uses Bayes' theorem to estimate a probability by counting the frequency of values, and then combine the values in the exist data. The NB is particularly suited when the dimension of the training set is high. Although the NB is a simple technique, it often offers outperform more sophisticated classification methods. Therefore, the NB is in the top 10 data mining algorithms identified by the IEEE International Conference on Data Mining (ICDM) in December 2006 (Wu et al., 2008).

The idea of NB is to use a set of training documents to estimate parameters and classes. Suppose that we have documents $D = \{d_1, \dots, d_{|D|}\}$, where Φ is a parameter and we use the notation $c_j \in C = \{c_1, \dots, c_{|C|}\}$ to indicate both j -th component and j -th class. We assume that the documents are generated by a mixture model and there is a one-to-one correspondence between the class labels and the mixture component. Each document d_i is produced by selecting a mixture component with the class prior probabilities $P(c_j; \Phi)$, and having this mixture component generate a document according to its own parameters, with distribution $P(d_i | c_j; \Phi)$. Therefore, it can identify the likelihood of a document as a sum of total probability over all generative components:

$$P(d_i | \theta) = \sum_{j=1}^{|C|} P(c_j | \theta) P(d_i | c_j; \theta) \quad (3.1)$$

Document d_i can be considered as an ordered list of word events. We write $w_{d_{ik}}$ for the word in k -th position of document d_i , where the subscript of w shows an index into the vocabulary $v = \{w_1, w_2, \dots, w_{|v|}\}$. We continue to assume that the probability of a word is independent of its position within the document. Using this assumption, the probability of a document given its class becomes:

$$P(d_i | c_j; \theta) = P(|d_i|) \prod_{k=1}^{|d_i|} P(w_{d_{ik}} | C_j; \theta) \quad (3.2)$$

where it assumes the length of the document, $|d_i|$, is grouped independently of class. Each class component is parameterized by the collection of word probabilities, such that $\Phi_{w_k | c_j} = P(w_k | c_j; \Phi)$, where $k \in \{1, \dots, |v|\}$ and $\sum_{k=1}^{|v|} P(w_k | c_j; \Phi) = 1$. The other parameters of the model are the class prior probabilities, $\Phi_{c_j} = P(c_j | \Phi)$, which predicts the

probabilities of choosing each mixture component. Let $\hat{\Phi}$ be the estimates of Φ . The parameters $\Phi_{w_k|c_j}$ can be estimated by using the Laplace smoothing based on a training set. With labelled training web documents, $D = \{d_1, \dots, d_{|D|}\}$, we can calculate estimates for the parameters of the model that generate these documents. To estimate the probability of a word given class, $\Phi_{w_k|c_j}$, we count the frequency with which word w_k occurs among all word occurrences for documents in class c_j . Therefore, these word probability estimates $\hat{\Phi}_{w_k|c_j}$ are:

$$\hat{\theta}_{w_k|c_j} = \frac{1 + \sum_{i=1}^{|D|} N(w_k, d_i) P(c_j | d_i)}{|V| + \sum_{k=1}^{|V|} \sum_{i=1}^{|D|} N(w_k, d_i) P(c_j | d_i)} \quad (3.3)$$

We write w_k for the word in k -th position of document d_i , $N(w_k, d_i)$ denotes the weight of word w_k occurring in document d_i , and where $P(c_j | d_i) \in \{0, 1\}$, is given by the class label. The class prior probability is given as $\Phi_{c_j} = P(c_j | \Phi)$, and can be estimated as

$$\hat{\theta}_c = \frac{\sum_{i=1}^{|D|} P(c_j | d_i)}{|D|} \quad (3.4)$$

Let $|D|$ be the total number of documents. With these parameters, assuming $P(d_i)$ is uniform, we compute the probability that a particular class generated a given document:

$$P(c_j | d_i; \theta) = \frac{P(c_j | \hat{\theta}) \prod_{k=1}^{|d_i|} P(w_{d_{i,k}} | c_j; \hat{\theta})}{\sum_{r=1}^{|C|} P(c_r | \hat{\theta}) \prod_{k=1}^{|d_i|} P(w_{d_{i,k}} | c_r; \hat{\theta})} \quad (3.5)$$

Finally, we must choose $\text{argmax}_j P(c_j | d_i; \Phi)$ as the best class of the document.

(2) Support Vector Machines

Support Vector Machine (SVM) is a supervised machine learning algorithm. It was introduced by Boser, Guyon and Vapnik in 1992 (Boser et al., 1992). It was used to solve the binary classification problem (Rossi and Villa, 2006). The SVM has the background from Vapnik & Chervonenkis' statistical learning theory in the 1960s (Boser et al., 1992). The SVM gained increasing popularity in the late 1990s because it is less sensitive to the dimension of the training set than other methods (Rossi and Villa, 2006). Simply speaking, building effective classification model does not depend on the size of the training set. The SVM is in the top 10 data mining algorithms identified by the IEEE International Conference on Data Mining (ICDM) in December 2006 (Wu et al., 2008).

The basic concept of SVM is to build a function that takes the value +1 in a “relevant” region capturing most of the data points, and -1 elsewhere (Joachims, 1998, Joachims, 1999). In addition, let $\Phi: \mathfrak{R}^N \rightarrow F$ be a nonlinear mapping that maps the training data from \mathfrak{R}^N to a feature space F . So, the dataset will be separated by the following primal optimization problem:

$$\text{Minimize: } \nu(w, \xi, \rho) = \frac{\|w\|^2}{2} + \frac{1}{\nu l} \sum_{i=1}^l \xi_i - \nu \quad (3.6)$$

$$\text{Subject to: } (w \cdot \Phi(x_i)) \geq \rho - \xi_i, \xi_i \geq 0 \quad (3.7)$$

where $\nu \in \{0,1\}$ is a parameter which lets one control the number of support vectors and errors, ξ is a measure of the mis-categorization errors, and ρ is the margin. When we solve the problem, we can obtain w and ρ . Given a new data point x to classify, a label can be assigned according to the decision function that is expressed as follows:

$$f(x) = \text{sign}((w \cdot \Phi(x)) - \rho) \quad (3.8)$$

where α_i are Lagrange multipliers and we apply the Kuhn Tucker condition. Then, it can set the derivatives with respect to the primal variables equal to zero. Finally, we can get:

$$W = \sum \alpha_i \cdot \Phi(x_i) \quad (3.9)$$

There is only a subset of points x_i that lies closest to the hyperplane and has nonzero values α_i . These points are called “support vectors”. To solve the primal optimization problem directly, the dual optimization problem is given by:

$$\text{Minimize: } W(\alpha) = \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j K(x_i, x_j) \quad (3.10)$$

$$\text{Subject to: } 0 \leq \alpha_i \leq \frac{1}{\nu l}, \sum_i \alpha_i = 1 \quad (3.11)$$

where $K(\mathbf{x}_i, \mathbf{x}_j) = (\Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j))$ are the kernels functions performing the non-linear mapping into the feature space based on dot products between mapped pairs of input points. They allow much more general decision functions when the data are nonlinearly separable and the hyperplane can be represented in a feature space. The kernels frequency used is polynomial kernels $K(\mathbf{x}_i, \mathbf{x}_j) = ((\mathbf{x}_i \cdot \mathbf{x}_j) + 1)^d$, Gaussian or RBF (radial-basis function) kernels $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma^2)$. We can eventually write the decision from the equations (16) and (17) and the equation can be illustrated as follows:

$$f(x) = \text{sign} (\sum \alpha_i K(\mathbf{x}_i, \mathbf{x}) - \rho) \quad (3.12)$$

- *Settings with single representative instances for each category:*

In such settings, clustering techniques such as the K -means clustering are employed where users nominate a representative centre-point (centroid) for each partition (cluster) as the clustering criteria, while the output is a partitioning of the repository into clusters, one per nominated centre-point.

Clustering (Han and Kamber, 2006, Mitchell, 1997) is the most important unsupervised algorithm used to the group together of similar data items into clusters. Definition of clustering can be explained following. Let $D \in R^{m \times n}$ a set of data-points/objects representing m points d_i in R^n . The objective is to partition D into K groups C_k , where every data-point belonging to the same group is more ‘*similar*’ than data-point in different groups. Each of K groups is called a cluster. Therefore, the goal of clustering is to determine the intrinsic grouping in a set of unlabelled data shown as an injective mapping $D \rightarrow C$ of data-points D_i to cluster C_k .

The similarity criterion is called *distance*. Two or more objects belong to the same cluster if they are “*close*” according to a given distance, called *distance-based clustering*. The simple *Euclidean Distance* can be used to successfully group similar data instances. It is noted that, even in this case the Euclidean Distance can sometimes be misleading.

Let x be centroid (as the mean of the cluster) and y is data-points of a clustering analysis. The distance between two data-points (items) can be calculated by using a distance function. One of the Euclidean distance techniques is the squared Euclidean distance:

$$d(\bar{x}, \bar{y}) = \sqrt{\sum_{n=1}^N (x_n - y_n)^2} \quad (3.13)$$

However, the centroid of each cluster must be determined, where it will be used to define its boundary. There are many solutions to determine the centroid x . A simple technique can be shown as formula 2.6.

$$\text{Centroid} = \frac{\sum d_i}{\text{total_of_}d_i} \quad (3.14)$$

Furthermore, centroid can be computed by applying the concept of *Rocchio Classification* (Rocchio, 1971, Manning et al., 2009), where this algorithm also uses the centroid to provide the boundary of a class. The centroid of a cluster can be determined as

the vector average of its data-points. The formula used to calculate the centroid presents as follows.

$$centroid = \frac{1}{|D_c|} \sum_{d \in D_c} \vec{v}(d) \quad (3.15)$$

where D_c is the set of all data-points that belong to class c and $v(d)$ is the vector space representation of d . It is noted that, the smaller the value of $d(x, y)$ is, the more similar the two processes are.

The K -means clustering is a partitioned clustering algorithm. It classifies a dataset into their specific clusters (Han and Kamber, 2006). The concept of the K -means clustering is to group objects based on attributes/features into K numbers of groups. The symbol K is a positive integer. The symbol K might be pre-assigned by the user, or it can be an unknown, determined by the algorithm. The grouping is done by minimizing the sum of the squares of distances between data and the corresponding cluster *centroid*. The K -means algorithm can be concluded as follows.

1. *Randomly choose K items and make them as initial centroids.*
2. *For each point, find the nearest centroid and assign the data-point to the cluster associated with the nearest centroid.*
3. *Update the centroid of each cluster based on the items in that cluster. Typically, the new centroid will be the average of all data-points in the cluster.*
4. *Repeats steps (2) and (3), till no data-point switches clusters.*

After data is performed by the step of data preparation, the data should have a quality that is appropriated for the next step, knowledge extraction. Therefore, this step will filter out the sentences that are likely to contain the target information. For example, if we just want to have the sentences relating to the modality treatment in a cancer clinical trial, this can be done by text classification or clustering techniques. An example can be shown as Figure 3.5.

Consider Figure 3.5. The original text is a document that is relevant to the cervical cancer in a clinical trial. Suppose the objective of this work is to find the clinical knowledge relating to treatment modality in this document. Therefore, we will select the relevant sentences before extracting knowledge from the document. In this case, we utilise a text classifier to find the relevant sentences in the document (see in the highlight area in Figure 3.5).

It is noted that the result of data preparation is to obtain more specific individual data because the capability of mining knowledge depends on something more specific than broad, unspecified knowledge (Larose, 2005, Brill, 1994).

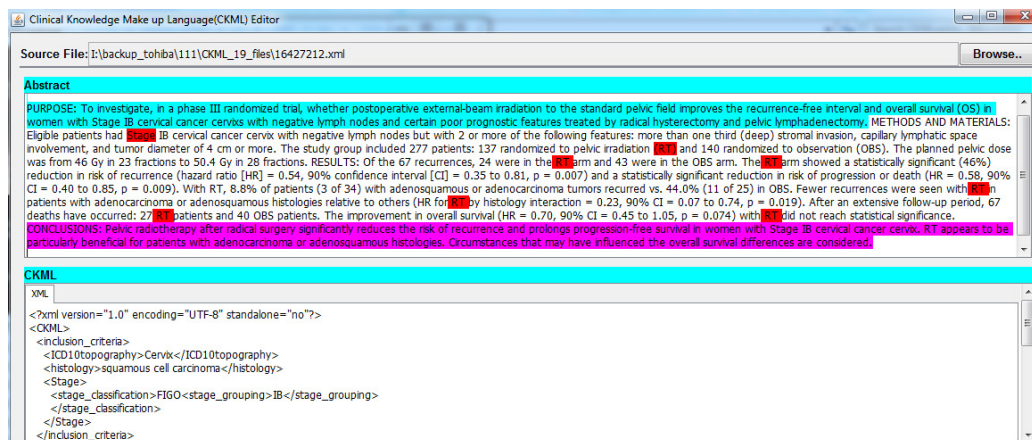


Figure 3.5 An example of a result of data preparation.

3.3.4 Closed-domain knowledge extraction

After identifying more specific through the step of data preparation, extracting knowledge in unstructured textual data can be easier task because NLP systems can exploit domain-specific knowledge frequently formalized in ontologies. Therefore, this step is called *closed-domain knowledge extraction*.

In this context, the term ‘closed-domain’ means that any dataset used under the ON-KDT methodology should be prepared in a specific way (Devedzic, 2001). Therefore, the definition of closed-domain knowledge discovery in the proposed methodology is to search for useful knowledge from unstructured textual data in a specific individual domain. Acquiring of more specific individual domain can be done through the techniques of identifying of more specific individual data that are described in Step 3 (Data Preparation).

To find the useful knowledge, we can utilize text mining techniques, where ‘*the objective of text mining is to exploit information contained in textual documents in various ways, including the discovery of patterns and trends in data, associations among entities, predictive rules*’ (Grobelnik et al., 2000b). Text mining techniques include document clustering, text classification, text filtering, information extraction (e.g. Question answering), and related to information retrieval (e.g. Search engine) (Stolcke and Segal, 1994, Grobelnik et al., 2000b, Pereira et al., 1995, Brown et al., 1992).

From our observation, our unstructured data consists of a set something that are described with text such as words, a sequence of words/items and so on. Therefore, in this context, there are two main solutions for finding the useful knowledge from unstructured/semi-structured text data.

(a) By finding the possible patterns

The concept of finding useful patterns/associations involves finding existing patterns in data describing with text, where the extracted patterns represent the important association of some elements in unstructured data repositories. In general, the patterns often mean association rules (Adamo, 2000, Agrawal and Srikant, 1994, Zhang and Zhang, 2002). As this, a solution of closed-domain knowledge discovery is to search for the important patterns from unstructured data repositories by using the techniques based on the concept of the Association Rule Mining.

(b) By using pattern language models

To use pattern language models, this means that we need to create some pattern language models, and use them to leverage the useful knowledge from textual data repositories. In this practice, the concept of closed-domain knowledge extraction is close to closed-domain question answering (QA) (Hirschman and Gaizauskas, 2001). Closed-domain QA is a form of question answering that deals with questions over a specific domain. This is to find the correct answers from a few documents that are relevant to a specific question. Also the closed domain QA may refer to a situation where only a limited type of questions is accepted. Therefore, the closed domain QA system does not have a large retrieval set abundant of good candidates for selection (Doan-Nguyen and Kosseim, 2004), and the answer should be in the form of a short phrase (Kangavari et al., 2008). We show how to extract useful knowledge from textual data repositories as follows.

(i) Specifying questions

It is to provide the main questions that may help identify the expected results. This can be performed manually. The possible questions should be necessary for project objectives.

(ii) Questions analysis and identifying types of answer

This is to extract the keywords from a question, interprets them, and then these keywords are expanded to other relevant terms.

It is noted that this work has not driven on the TREC standard⁶ (TREC 8, 9, 10, and 11). This is because this work utilizes only the concept of QA to more specific domain of software engineering, where we provide the questions as the direction of analysing the

requirements of a system and the answers are the actors and their actions in the system. Finally, these results are used to design a software application. The process of question analysis and identifying the type of answer can be shown as Figure 3.6.

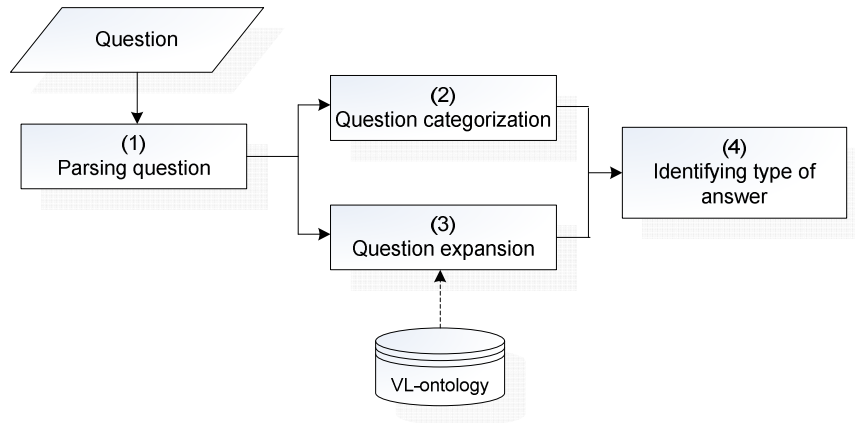


Figure 3.6 The process to analyse questions and identify type of answers

In Figure 3.6, when a question is input to our system, it will be transformed to a structure explaining sequences of words in a natural language, called *parsing* (1). Parsing text has often driven on POS tagging. An example of parsing rules used in this context can be presented as Table 3.11. The parsing rules can be developed based on context free grammar (CFG)⁷.

Table 3.11 Examples of parsing rules

<question-form>:	<question-class>< auxiliary-verb><subject><main-verb><object><complement> <question-class>< auxiliary-verb><main-verb><complement> <question-class><main-verb><complement> <question-class><main-verb><object><complement> <question-class>< auxiliary-verb><subject><main-verb> ...
<question-class>:	what whol when why where
<auxiliary verb>:	do will did have has can shall ...
<main -verb>:	process perform ...

⁶ TREC (Text Retrieval Conference) has had a question answering track since 1999. This track is an effort/standard to bring the benefits of evolution to bear the question answering problem. It aims to develops the true equivalent of a standard retrieval test collection is an open problem. (Referred to <http://trec.nist.gov>)

⁷ Context free grammar (CFG) is a formal grammar in which every production rule is of the form, $V \rightarrow w$, where V is a *single* nonterminal symbol, and w is a string of terminals and/or nonterminals (w can be empty). (Referred to http://en.wikipedia.org/wiki/Context-free_grammar).

Table 3.11 (cont')

<complement>:	noun phrase cause
<subject>:	noun phrase
<object>:	noun phrase
...	...

In Figure 3.6, after processing by (1), the results from (1) will be passed to (2). The questions in QA should be classified into five categories such as *who*, *why*, *when*, *which*, and *what*. The question used in class '*who*' always refers to a person. Meanwhile, the question used in class '*what*' can be classified into four sub-classes: *what* (referring to something, doing something), *what-who* (referring to a person), *what-where* (referring to the location), and *what-when* (referring to date/time). In fact, the questions are provided to be used as mechanisms for identifying the relevant answer. Hereafter, the results from (2) will be passed to (3). This stage is to expand the query, called *query expansion*. In QA, a query is a keyword extracted from the question. In this context, we utilise the VL-ontology and WordNet to expand the keywords that are extracted from main question(s). The VL-ontology is used to group the word variations into the same concept. For example, as we said that, the question class '*who*' always refers to a person. Therefore, '*person*' is a keyword of the question '*who*'. The word '*person*' can expand to the words '*human*', '*individual*', and '*someone*'. However, the word '*individual*' can be changed its form to '*individuals*'. The word '*individuals*' should be an expansion of the word *person*' by analysing through the VL-ontology. Finally, the type of answer can be identified based on the specific question. The process of semi-automatically generating rules can be explained as follows.

Step 1: We provided some general rules for generating the parsing rules manually. The basic rules are illustrated as follows.

- *Question-word + Verb + Object?*
- *Question-word <what, when, why,>*
- *Subject + Verb*
- *Subject + Verb + Object*
- *Article + Noun*

Step 2: It is to collect the questions through the WWW stored in a text file.

Step 3: These questions are grouped into many groups by using the keywords such as *how*, *what*, *who*, *when*, and so on.

Step 4: We utilize a dictionary to tokenize words in a question. Then after the question is tokenized, the results also show a part of speech of each word. For example, suppose

we have a question ‘*What is the Internet?*’ By using a dictionary, this question can be results as follows.

<i>what</i>	<i>is</i>	<i>the</i>	<i>Internet</i>
<Question-word>	<Verb>	<Article>	<Noun>

Step 5: As the result shown in Step 4, our system can automatically produce the paring rules for question analysis. As the example above, the paring rule is <Question-word> <Verb> <Article> <Noun>.

(iii) Development of answering pattern

To extract the answer that is relevant to the question from requirements, it is necessary to provide a tool used to search for answers (person and its action). The main technique used in this work is called *pattern matching based*. This technique is to automatically extract answers from text through formalized pattern based on *N*-gram model⁸. The process of answering pattern development can be concluded as the following algorithm.

1. A document is broken into sentences. A sentence should be simple. A boundary of a sentence can be truncated by a full stop.
2. Tokenizing a sentence and performing with POS tagger to assign the words in a text with their corresponding parts of speech
3. Finding the verb position, and generating the *verb-part* based on *N*-gram model.
4. Finding the object position (if it has). It helps describing what action is. The *object-part* is also generated.
5. Finding the subject position, and generating the *subject-part* based on *N*-gram model.
6. Each sentence is transformed into a binary tree structure. The root node must be the verb.
7. Considering each binary tree as a pattern

An example of the use of the algorithm above can be illustrated following. Consider the sentence “*Students can select four courses for the coming semester*”. The example of developing the answering patterns can be illustrated as follows.

Step 1: The sentence is tokenized, and it can be represented below.

⁸ An *n*-gram is a contiguous sequence of *n* items from a given sequence of text. This model is simple but it is powerful used in computational linguistics. An *n*-gram can be any combination of letters, phonemes, syllables, words or base pairs according to the application. An *n*-gram of size 1 is called a ‘*unigram*’; size 2 is a ‘*bigram*’, size 3 is a ‘*trigram*’. Larger sizes are referred to the value of *n* (e.g. four-gram, five-gram)

Students / can / select / four / courses / for / the / coming / semester

Step 2: The second step uses a POS tagger to assign the words in a text with their corresponding parts of speech. An example can be shown as below.

students / can / select / four / courses / for / the / coming / semester
 <NN> <MD> <VBP> <JJ> <NNS> <IN> <ART> <JJ> <NN>

As above, after using the POS tagger, we determine we can be described the word classes and shown as Table 3.12.

Step 3: This step is to find the verb position based on *N*-gram model.

For example, if the verb is identified, the subject and the complement are consequently found. In this example, the verb position is ‘*can select*’. The word ‘*can*’ is the modal verb, while the word ‘*select*’ is the verb which is in the present tense. As this, a *verb-part* can be generated as <MD>< VBP>.

Table 3.12 The results after using the POS tagger

Words	Abbr.	Description
students	NN	Noun (single)
can	MD	Modal
select	VBP	Verb (present tense)
four	JJ	Adjective
Course	NNS	Noun (plural)
for	IN	Preposition
the	ART	Article
coming	JJ	Adjective
semester	NN	Noun (single)

However, it can be illustrated an example of generating rules based on *N*-gram model. Suppose a corpus contains 100 sentences, there are 10 sentences having <MD><VBP>. By using *N*-gram model, it can be calculated the probability of <MD> followed by <VBP> as follows.

$$Pr(w|<MD>< VBP >) = 10/100$$

$$= 0.1$$

As above, the bigram <MD><VBP> is a model of the verb-parts that can be generated from the corpus, and its probability of the model <MD><VBP> is 0.1.

Step 4: It is to find the object position. It always follows the verb position. In this context, we consider the rest of the sentence that is following the verb as the object. For this example, an *object-part* can be generated as <JJ><NNS><Preposition-phrase>.

Step 5: This step is to find the subject position. It is always represented in the front of the verb position. The word ‘students’ is a plural noun. As this, a noun-form can be generated as <NNS >.

Step 6: It is to transform the *verb-part*, the *subject-part*, and the *object-part* into a binary tree structure. The root node is the *verb-part*. The left node is the *subject-part*, while the right node is the *object-part*.

Step 7: Finally, each binary tree is transformed as an answer pattern used to extract the reality need in text data.

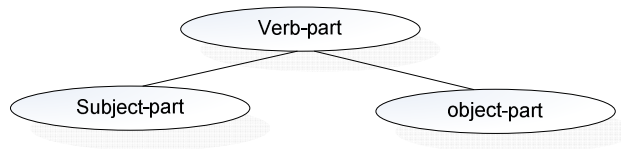


Figure 3.7 A binary tree representing the format of answer pattern

In Figure 3.7, some examples of answering pattern can be generated as shown in Table 3.13.

Table 3.13 Examples of answering patterns

Subject-part:	<Article><Noun> <Noun-phrase> <Article><Noun-phrase> ...
Verb-part:	<auxiliary-verb><verb> <modal-verb><verb> verbl...
Object-part:	<Article><Noun> <Noun-phrase> <Article><Noun-phrase> ...

(iv) Passage Retrieval

The original definition of passage retrieval is to retrieve only the portions of a document, if they are relevant to particular information need (Baeza-Yates and Ribeiro-Neto, 1999, Callan, 1994, Hearst and Plaunt, 1993, Salton et al., 1993). In this context, we utilize the concept of text filter to retrieve only the relevant portions after the document is broken into sentences.

(v) Knowledge Extraction

This step utilizes the answering patterns that are developed in the third step to extract knowledge in the unstructured textual data repositories. The extracted knowledge should be relevant to the specific question (s).

CHAPTER 4

THE ON-KDT METHODOLOGY APPROACH FOR EXTRACTING SCENARIOS FROM SOFTWARE REQUIREMENTS

As we said in early thesis that, given the different empirical experiments, it may lead to have confidence in the On-KDT methodology, if it can substantially succeed in extracting knowledge from several unstructured data repositories. The On-KDT methodology is applied in three distinct settings. One of settings is to extract useful knowledge from software requirements, where the useful knowledge is scenarios of software collected in requirement specifications. The detail of discovery scenarios in natural language requirements can be described as follows.

4.1 Background

A scenario describes a temporal sequence of activities/tasks. It is useful for helping software developers to understand the interactions among the components of a software system and validate software architectures (Sutcliffe, 1998, Sutcliffe et al., 1998, Amyot et al., 2003, Alspaugh and Antón, 2008). Also, scenarios have been used as a means for capturing and refining system requirements. The efficiency and success of a software development project are often highly reliant on the quality of scenarios. Over the past few decades, scenarios have emerged as effective tools for designing and validating in complex software systems (Kazman et al., 1994, Carroll, 1997, Maiden, 1998, Dzida and Freitag, 1998, Alspaugh et al., 1999, Cockburn, 1995, Gargantini et al., 2008). Scenarios represent useful information not just about the operation of the current system but also about its operating context/environment. Also, scenarios provide a solid basis for testing and maintenance activities. A well-known requirements engineering methodology that leverages scenarios is Scenario Based Requirements Analysis (SBRA) (Sutcliffe, 1998, Saiedian et al., 2005, Letier et al., 2005).

In spite of the increasing attention of scenario analysis in requirement engineering (Carroll, 1995), there is a few works have proposed methods to discover scenarios from requirement artefacts. The main cause can be described as follows.

In literature, much of software requirements are written in a natural language format, called text-based software requirements specifications (SRSs) (Mu et al., 2009). It is well-known that natural language is too ambiguous. Therefore, it is laboured to elicit requirements assets from textual requirements.

The study of finding significant assets in software requirements is still recognized as one of the important problems in software engineering and requirement engineering. It is time-consuming and labour-intensive, especially when working on a large number of software requirements and each document may contain a large volume of detail. Some of close studies of identifying the significant assets from software requirements can be presented following.

Mansurov & Probert (2001) proposed a scenario-based approach to re-engineering of legacy telecommunication software into formal specifications. The methodology consists of two approaches. The first is to extract scenarios from the legacy software using a combination of dynamic and static strategies. The second is to automatically synthesize formal specifications from these scenarios. In the second step, they used the MOST toolkit, which is capable of synthesizing state-machine based formal models from scenarios, formalized as extended Message Sequence Charts (MSCs). They also provided detailed descriptions of their re-engineering methodology and compared it with related approaches. The scenario-based methodology was applied to re-engineer a small-sized telecommunications-like software system, called the ToolExchange.

Mu *et al.* (Mu et al., 2009) studied about acquiring of the requirement assets from text-based SRS. This work focused on extracting functional requirements through the analysis of the linguistic characterization of SRSs. Finally, the extended functional requirements can be generated by converting rules.

Bajwa *et al.* (Bajwa et al., 2009) proposed a natural language processing (NLP) based automated system for object-oriented modelling of user requirements and generating code in multi-languages. The model utilizes a rule base to analyse the natural languages and extract the relative and required information from the given software requirements. The requirements written by users should represent based on simple English in a few paragraphs. The designed system incorporates NLP methods to analyse the given script. Firstly, the natural language text is semantically analysed to extract classes, objects and their respective, attributes, methods and associations. Later, UML diagrams are generated on the bases of previously extracted information. The designed system also provides with the respective code automatically of the already generated diagrams. The designed system provides a quick and reliable way to generate UML diagrams to save the time and budget of both the user and system analyst.

Medeni *et al.* (Medeni et al., 2011) presented a proposal that aims to extract a tacit knowledge visualization methodology to support know-where the requirements of the organizational knowledge. They considered that Software Requirement Specification (SRS)

process has field-specific problems that need to be eliminated by using the suitable tacit knowledge extraction techniques.

4.2 Motivation

Scenarios describe not only the significant requirements for developing a system, but also allowing discovery and validation of system requirements. However, it is a time-consuming and labour-intensive process when manually extracting scenarios from requirement artefacts, especially when there are a large number of software requirements. Therefore, automatically extracting useful set of activities from software requirement and associating them as a scenario are a challenging problem in proposal.

4.3 Research Contribution

We apply the On-KDT methodology to extract scenarios from natural language software requirements.

4.4 Preliminaries: formal representation of scenarios

4.4.1 Formal definition of scenario

Formally, a scenario is defined as a “*possible behaviour limited to a set of purposeful interactions taking place among several agents* (Plihon et al., 1998)”. According to Rolland *et al.* (Manning and Schütze, 1999) “*a scenario is composed of one or more actions. The combination of actions in a scenario describes a unique path leading from initial to final states of agents*”. Other authors have used more detailed definitions, such as the following: “*A scenario is a description that contains actors, background information about them, and assumptions about their environment, their goals, or objectives, and sequences of actions and events. It may include obstacles, contingencies, and outcomes. In some application, scenarios may omit one of the elements or express it simply and implicitly* (Go and Carroll, 2004).” Use-case diagrams in the *Unified Modelling Language (UML)*⁹ provide an abstract description of scenarios (Manning and Schütze, 1999).

4.4.2 General form of scenarios

In the account of Rolland *et al.* (Plihon et al., 1998) on how a scenario might be developed, we start by clearly conceptualizing (but not necessarily representing within a

⁹ Rational Software Corp. *Unified Modelling Language Version 1.1.*, available at <http://www.rational.com/uml/documentation.html>, 1998.

scenario) the initial and final states associated with the scenario. The initial state describes the preconditions for the performance of the scenario, while a final state describes the outcomes achieved. A scenario can be classified into two types: normal scenario and exceptional scenario. A normal scenario (Plihon et al., 1998) leads to results that satisfy the needs of users, while an exceptional scenario (Plihon et al., 1998) (Plihon et al., 1998) leads to the achievement of a potentially “second-best” goal if the original goals cannot be achieved. The following example illustrates this.

The scenario of ‘*Withdraw cash from ATM*’ has ‘*The ATM is ready*’ as the initial state and leads to the following final state: ‘*The user has cash*’. This scenario can reach for the goal of withdrawing cash from ATM. This scenario is called the *normal scenario*. However, the scenario of ‘*Withdraw cash from ATM*’ can lead to the final state ‘*The user has no cash*’, where the ATM recognizes that ‘*Withdraw cash from the ATM by treating the exception of three invalid code attempts*’. This scenario is called the *exceptional scenario*. The structure of the scenario can be described as follows.

A scenario contains one or more actions. An action in the scenario can be classified into two types: atomic and flows of actions. An atomic action is an interaction from one agent to another which affects for some parameter objects. For example, consider following case, ‘*The user inserts a card in the ATM*’. This is an atomic action presenting a communication action between two agents (‘*The user*’ and ‘*The ATM*’). In general, atomic action is in the form of a cause (Manning and Schütze, 1999). Flows of actions consist of several actions. For example, consider the following sentence, ‘*The bank customer gets a card from the bank, then the bank customer withdraws cash from the ATM*’. This is a flow of actions, where it comprises of two atomic actions, ‘*The bank customer gets a card from the bank*’ and ‘*the bank customer withdraws cash from the ATM*’.

In addition, a flow action can carry a condition that characterises the course of actions of the scenario. The condition is called *flow condition* (Plihon et al., 1998). For example, consider this sentence ‘*if the code is valid, then a prompt for amount is displayed by the ATM to the use*’. The condition of this flow action is ‘*if the code is valid*’. This condition will be identified a unique case of ATM usage described in the scenario.

As above, the structure of a scenario can be shown as Figure 4.1 and the scenario can be presented as a semi-structure textual form shown as Figure 4.2. This form was proposed by Rolland *et al.* (1999).

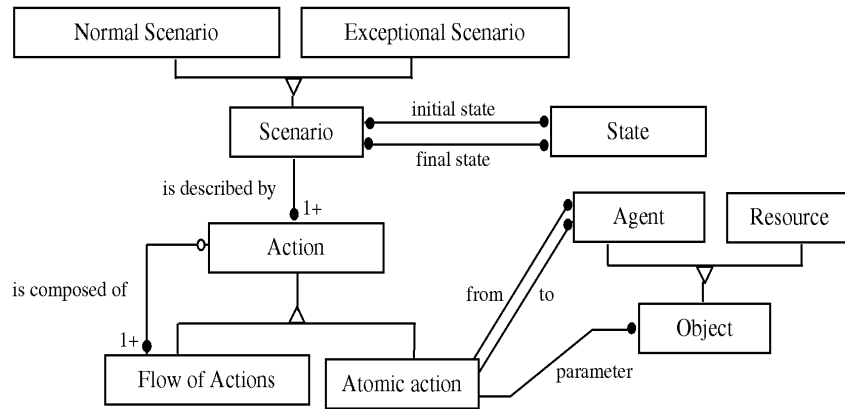


Figure 4.1 Scenario Structure (Manning and Schütze, 1999)

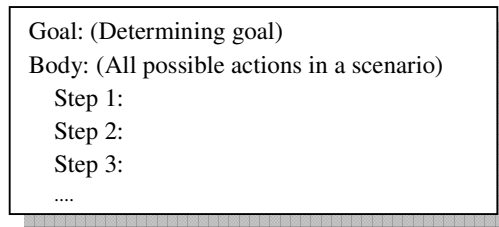


Figure 4.2 The semi-structure textual form proposed by Rolland *et al.* (1999)

4.4.3 A Purposed Format of Scenario in this thesis

For the purposes of implementation, we will store scenarios in the following XML (Extensible Markup Language) schema because it is easier to describe allowable document content.

```

<?xml version="1.0"?>
<xs: scheme = 'system's name'>
  <xs: element name= `name 's element
    <element action class>
      <element task>...<element co-actor>...<element co-actor><element task>
      <element task>...<element co-actor>...<element co-actor><element task>
      ...
    </element action class>
    ...
  </xs: element name>
  ...
</xs: scheme>
  
```

Figure 4.3 The purpose format based on XSD used to store scenarios

In Figure 4.3, it shows the detail of the XML structure used to store scenario in this context. The definition of each element can be described as follows.

Table 4.1 The definition of each element in the XSD used to store scenarios

Elements	Definition
scheme	System's name
element name	Name of element
action class	To represent the main action in the scenario
task	To represent all tasks in the main action
co-actor	To show a co-actor in each task if it can be found

4.5 Extracting Scenarios from Software Requirements

This section shows how to apply the On-KTD methodology for extracting scenarios from software requirements.

4.5.1 Defining of the Problem and Understanding Data: A Case Study of Course Registration System

In this section, we present the application of the On-KDT methodology with a case study that involves extracting scenarios from a text document describing requirements for a course registration system (based on the client-server architecture). The original textual requirements description is about 70 pages long (publicly available at the following site¹⁰). The target system is required to support students registering for courses and accessing enrolment records, marks etc. over the web. It is also required to support professors is managing their courses over the web.

The first step is to understand the structure of sentences containing in the requirements. After the requirements are considered carefully, it can be seen that a requirement sentence consists of three main units: *subject*, *verb*, and *object*.

In general, the verb should be identified firstly, and then the subject and object will be identified. This is because the subject is always in front of the verb, while the object is always followed by the verb.

¹⁰https://docs.google.com/viewer?a=v&q=cache:unl_C7PHwaEJ:online-examination-uet.googlecode.com/files/03_dev475_ex_workbook_main.pdf+ibm+rational+software+%2B+Course+Registration+Requirements&hl=th&gl=th&pid=bl&srcid=ADGEEsGa4T6EJoIVlelPDnnGrdfsPmJjdDrNqMgEZeBkhii2XkPXF9VVsqBNm72PCr5CkL7hq_f9vYEoUfI8y588UBPgflkQJVtErQw7dZCZxsGmzmEXZZwmbFOdsGFRd2awKV31hRWI&sig=AHIEtbSE-tXLmaOmLU4byo2iX_0NvIUx5Q

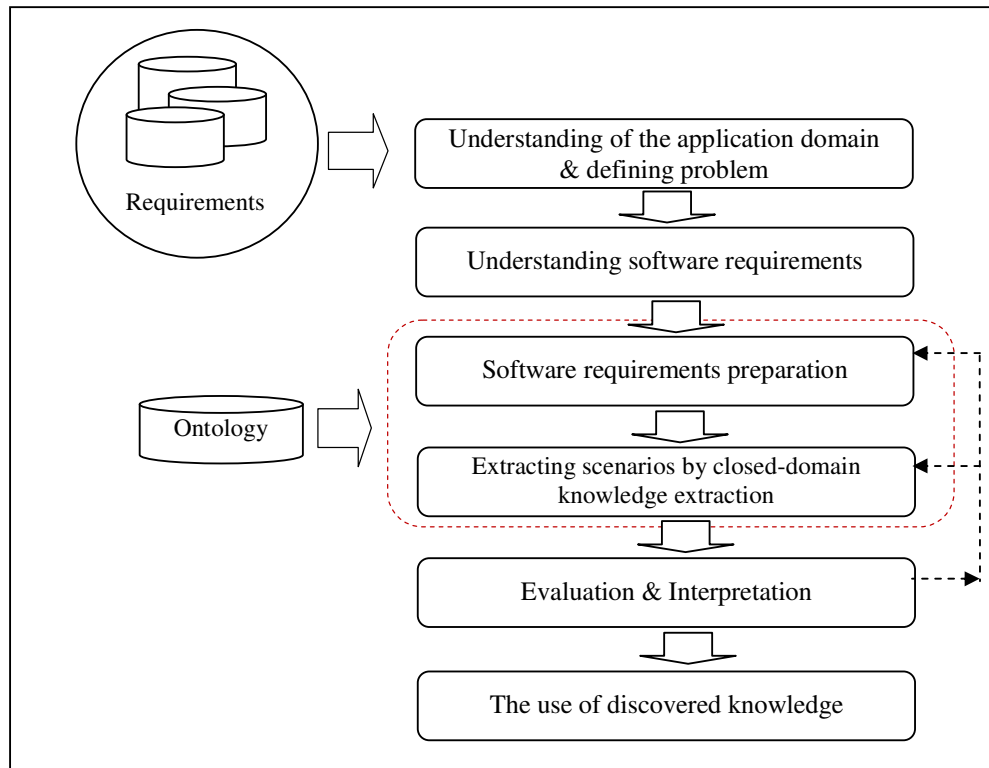


Figure 4.4 The On-KDT methodology applied to finding scenarios from software requirements

By the definitions of English grammar, the *subject* is the person or thing (who/what) performs the action. The *verb* described the action of the subject, while the *object* is the person or thing affected by the action described in the *verb*. The *object* can be nouns, pronouns, phrases, or clauses. In a scenario, the subject refers to the ‘actor’, while the ‘action’ is determined via a combination of the verb and the object.

As above, we can show an example sentence analysis following. Consider this sentence, “*Students can select four courses for the coming semester*”. Ideally, the identification of the ‘actor’ and ‘action’ in the sentence will lead to the result shown in Figure 4.5.

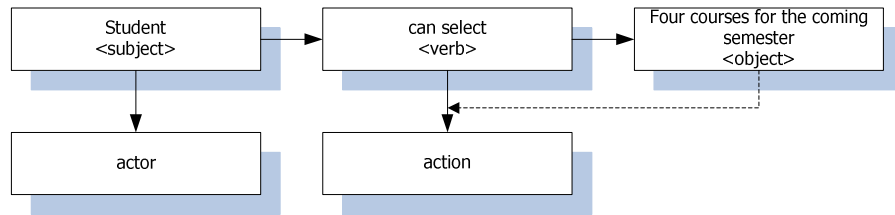


Figure 4.5 An example of finding the actor and action in a requirements sentence

Furthermore, after carefully analysing the characteristic of the requirements of the course registration system, we obtain following assumptions.

(1) Consider about actors involving the course registration system. We concentrate to extract scenarios of three main actors: *professors*, *students*, and *registrar*. Also, this information can be utilized to classify the software requirements into three main categories. This is to re-organize the dataset into more specific individuals before extracting knowledge, where the ability of knowledge extraction depends on more specific data than general data.

(2) As the assumption (1), it may imply that if the requirement dataset is classified into three categories (*professors*, *students*, and *registrar*), the student registration system may have three scenarios: the scenarios used for students, the scenarios used for professors, and the scenarios used for registrars.

(3) A requirement sentence containing two or more actors, it may be presenting an association among actors in the system. For example, the sentence “*For some courses, students must have permission from professors before enrolling for these courses*”, this sentence shows an association between the actor ‘*students*’ and the actor ‘*professors*’.

Figure 4.6 shows a specialization of the general XML format for representing scenarios in the instance of this application. The root element contains three main elements: *students*, *professors*, and *registrars*. Each element contains *task-class* which is a group of specific tasks. A *task-class* contains multiple tasks, and each task will show a relationship between actors in the system. For example, the sentence “*For some courses, students must have permission from professors before enrolling for these courses*”, contains two actors. The student and the professor are co-actors.


```

<?xml version="1.0">
<Scenarios> a course registration system
  <actor="registrar">
    <main-task> update-students
      <sub-task> .. <consequent-task>...</consequent-task><co-actor>...</co-actor></sub-task>
      <sub-task> .. <consequent-task>...</consequent-task><co-actor>...</co-actor></sub-task>
    </main-task>
    <main-task> delete-students
      <sub-task> .. <consequent-task>...</consequent-task><co-actor>...</co-actor></sub-task>
      <sub-task> .. <consequent-task>...</consequent-task><co-actor>...</co-actor></sub-task>
    </main-task>
    ...
  </actor>
  ...
</Scenarios>

```

Figure 4.6 A Scenarios in XML format

4.5.2 Software Requirement Preparation

The data preparation step consists of two main tasks: *data pre-processing* and *identifying more specific individuals*.

(a) Data Pre-processing

This task is performed on raw text data which is transformed into the form of the vector space model (VSM) (Baeza-Yates and Ribeiro-Neto, 1999) that is appropriate for the process of identifying more specific individuals.

(i) Data Transformation

Most NLP techniques rely on separating a document into *passages* (Callan, 1994, Hearst and Plaunt, 1993, Salton et al., 1993). A passage can be one or more paragraphs, sections, or sentences. The extraction of passages is a crucial component of document retrieval, natural language processing, and information extraction techniques, where the aim is to search for more precise and compressed text excerpts in response to queries, rather than giving complete documents. It is important to extract only relevant passages. If too many irrelevant passages are retrieved, extracting knowledge from text is likely to fail due to too much noise (Liu and Croft, 2002).

For our purposes, a passage will be a sentence. Before transforming raw text data to an appropriate format, a document is decomposed into sentences (shown in Figure 5). The main idea of separating document into sentences is to divide documents into coherent units with each unit (i.e., a sentence) corresponding to a requirement. Therefore, each sentence is tokenized, where tokenization is the idea of separating a stream of text (e.g. sentence or document) up into words. Later these sentences will be transformed into the VSM (also

known as ‘*bag of words (BOW)*’ (Baeza-Yates and Ribeiro-Neto, 1999), where documents and queries are both vectors. In VSM, it represents a document as a vector of keywords that are extracted from the document, with associated weights representing the importance of the keywords in the document and within the whole document collection. Furthermore, a query is modelled as a list of keywords with associated weights representing the importance of the keywords in the query. It is noted that, in this work, a sentence will be viewed as a document.

This step utilizes the VL-ontology as the semantic tool to understand similar words (having the same stem (or synonyms) and the same grammar class). These words (terms) will be considered as representing the same concept. For example, consider three sentences below.

S₁: Students can select four courses for the coming semester.

S₂: A student can cancel courses that are selected and contained in their list.

S₃: Professors indicate which courses they will be teaching.

Consider the term ‘*students*’ in *S₁* and the term ‘*student*’ in *S₂*. These refer to the same concept. Consider also the term ‘*select*’ in *S₁* and the term ‘*selected*’ in *S₂*. These also refer to the same concept ‘*select*’ because they contain the same stem and they are the verbs. Therefore, after transforming these documents to BOW, they can be represented in as Table 4.1. As the results, the VSM can then be viewed as a set of concepts instead of a set of words, by leveraging the VL-ontology.

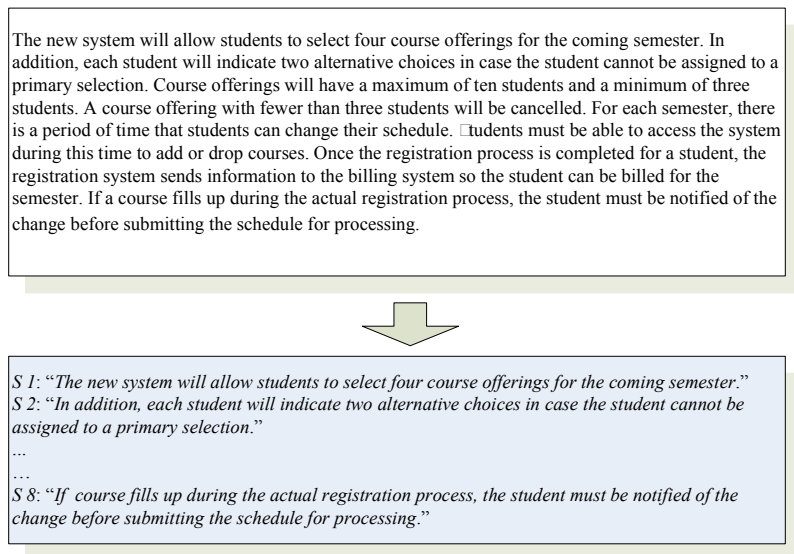


Figure 4.7 Separating a requirement document into many sentences

Table 4.2 An example of transforming requirement documents into BOW by using the VL-ontology

Terms	D_1	D_2	D_3
student / students	1	1	0
can	1	1	0
select / selected	1	1	0
four	1	0	0
course	1	1	1
for	1	0	0
coming	1	0	0
semester	1	0	0
cancel	0	1	0
that	0	1	0
Are	0	1	0
And	0	1	0
contained	0	1	0
in	0	1	0
Their	0	1	0
list	0	1	0
Professors	0	0	1
indicate	0	0	1
which	0	0	1
They	0	0	1
will	0	0	1
be	0	0	1
teaching	0	0	1

(ii) Data Cleaning

After transforming software requirement sentences into the BOW format, we obtain $w = (w_1, w_2, \dots, w_k, \dots, w_v)$, where v is the number of unique words/concepts within the collection. In the BOW, a requirement sentence s_i is composed of a sequence of words, with $s_i = (w_{i1}, w_{i2}, \dots, w_{ik}, \dots, w_{iv})$, where w_{ik} is the frequency of the k -th word in the requirement sentence s_i . After parsing the requirement sentences to extract unique words, stop-words are removed. Stop-words are words which are considered as non-descriptive within a BOW approach. They typically comprise prepositions, articles, etc. These words usually have very high frequency in the total corpus, and are removed prior to classification. Following

common practice, we removed stop-words by using a standard list with 571 stop-words¹¹. Finally, based on Figure 6, we now obtain the BOW after removing stop-words shown as Table 4.3.

Table 4.3 An example of the BOW of requirement documents after removing stop-words

Terms	D_1	D_2	D_3
students / students	1	1	0
select / selected	1	1	0
course	1	1	1
cancel	0	1	0
contained	0	1	0
list	0	1	0
professors	0	0	1
indicate	0	0	1
teaching	0	0	1

(iii) Weighting terms

In *tf-idf*, each unique word w_i corresponds to a feature with $tf(w_i, s_i)$, which is the number of times word w_i occurs in the sentence s_i , as its value. It has been shown (Larocca et al., 2002) that scaling the dimensions of the feature vector with their *inverse sentence frequency* $idf(w_i)$ leads to improved performance. $idf(w_i)$ can be calculated from the sentence frequency $sf(w_i)$, which is the number of sentences in which word w_i occurs. idf is defined as follows:

$$idf(w_i) = 1 + \log(|S| / sf(w_i))$$

According to Table 4.3, it now needs to calculate for term weighting. Firstly, it needs to calculate the *idf* threshold of each term, $idf = 1 + \log(|S|/sf)$. Let $|S|$ be 3, where $|D|$ the total number of requirement sentences in the corpus. The sf is the number of sentences containing term t in the corpus. Therefore, the *idf* of each term can be calculated as follows.

$$idf_{(students)} = 1 + \log(3/2) = 1.176$$

$$idf_{(select)} = 1 + \log(3/2) = 1.176$$

$$idf_{(course)} = 1 + \log(3/3) = 1$$

$$idf_{(cancel)} = 1 + \log(3/1) = 0.477$$

¹¹ <http://www.aifb.uni-karlsruhe.de/WBS/aho/clustering>

$$\begin{aligned}
isf_{(contained)} &= 1 + \log(3/1) = 0.477 \\
isf_{(list)} &= 1 + \log(3/1) = 0.477 \\
isf_{(Professor)} &= 1 + \log(3/1) = 0.477 \\
isf_{(indicate)} &= 1 + \log(3/1) = 0.477 \\
isf_{(teaching)} &= 1 + \log(3/1) = 0.477
\end{aligned}$$

After calculating isf of each term, tf - it can be calculated and is shown as Table 4.3. Finally, these requirement sentences are ready for the next task, identifying more specific individuals.

Table 4.4 An example of the BOW of requirement documents after weighting by tf - isf

Terms	D_1	D_2	D_3
students	1.176	1.176	0
select / selected	1.176	1.176	0
course	1	0.477	1
cancel	0	0.477	0
contained	0	0.477	0
list	0	0.477	0
Professors	0	0	0.477
indicate	0	0	0.477
teaching	0	0	0.477

(b) Identifying more specific individuals

This step aims to group the requirement sentences having similar needs into the same categories prior to mine scenarios in the next step. Slimming down the number of requirements contributes to a better understanding of the impact of the requirements and elaborate them through text categorization discussion.

With the classical search technique, called keyword search, it may be applied. In the study of sentence structure of the course registration system, if his technique is applied, it can be found some problems. Consider following two sentences viewed as documents.

S_1 : *Students can select four courses for the coming semester.*

S_2 : *Students can cancel courses that are selected and contained in their list.*

If the sentences are simple as the example shown in S_1 and S_2 , it may be easily used 'keyword' for grouping the sentences. Consider these sentences. We can classify the sentences having the keyword 'students' into the same category.

However, some sentences can be complex or compound. An example can be illustrated following.

S₁: After professors provide the course for the coming semester, and then they will permit students to select courses that they want to register.

S₂: Students must have permission from professors before enrolling for these courses.

As the sentence above, the keyword search may not be appropriate for this case. This is because full-text searching that relies on matching words in search space, is an inefficient method of finding significant information in text content (Beall, 2008). This is because, this technique may fail to analyse about synonyms, and it also finds and retrieves unwanted homonyms (Mann, 2005, Beall, 2008). Then, the homonym problem occurs in full-text searching when a single word or phrase has more than one meaning.

As this, it can be found in the previous study that clustering is applied when it attempts to solve the homonym problem in subject searches (Joshi and Jiang, 2002, Wang et al., 2009, Beall, 2008), where clustering can group and separate out resources by subject. However, one of the limitations of cluster analysis is that there not official guidelines or conventional approaches to identifying or defining clusters. Therefore, in a specific domain, a supervised learning algorithm may perform classification better. Identically, text processing will be tied to specific areas of known identity. As this, the use of a supervised learning algorithm may be the better solution to group the requirement sentences having similar needs into the same categories prior to mine scenarios. The learned knowledge can then be used by the machine to perform similar actions on these tasks. Simply speaking, it can be easily used for discriminating classification. Also, a classification method driven on supervised learning is usually fast and accurate.

In this study, we utilize the Support Vector Machine (SVM) algorithm to assign a requirement sentences to one or more pre-defined categories (Joachims, 1998), where the SVMs are helpful in text and hypertext categorization as their application can significantly reduce the need for labeled training instances in both the standard inductive and transductive settings. It is noted that the One-Class SVMs is applied (Manevitz and Yousef, 2001). This algorithm just provides the normal training data. The algorithm creates a model of this data. If newly encountered data is too different, according to some measurement, from this model, it is labelled as out-of-class. In addition, this algorithm can be effective with relatively small data sets (Cao and Tay, 2000).

As it said at the section 4.4.1 that, the requirements relating to three main actors (professors, students, and registrar), and we utilize this information to classify the requirement sentences into three main classes before extracting the scenarios.

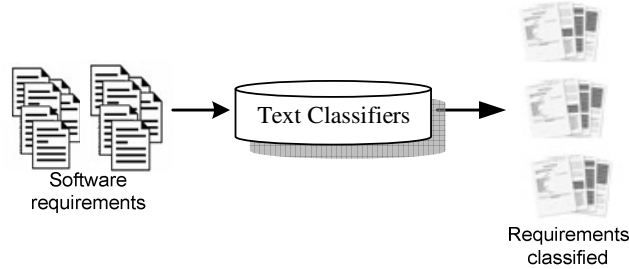


Figure 4.8 Automatically classifying requirements having similar needs into pre-defined classes

We manually select 50 sentences per class used to build the classifier model based on the One-Class SVM algorithm. After the classifier models are built, they are used to automatic classify all requirements into pre-defined classes.

We evaluated the results of the experiments for text classification by using the information retrieval standard (Baeza-Yates and Ribeiro-Neto, 1999). Common performance measures for system evaluation are *precision* (P), *recall* (R), and *F-measure* (F). The results of the evaluation are represented as Table 4.5.

Table 4.5 The results of automatic requirement classification by the SVM classifiers

Categories	Recall (%)	Precision (%)	F-measure (%)
Students	91.66	98	94.72
Professors	90.76	97	93.77
Registrar	88.57	96	92.13

As the results of text requirement classifier, they present the satisfactory accuracy after testing by F -measure. In contrast, they also show the failure rate. This is because word relationships play a vital role in the case. For example, consider following two sentences viewed as documents.

S_1 : *Students can select four courses for the coming semester.*

S_2 : *However, they can cancel courses that are selected and contained in their list.*

Consider two sentences above. Two sentences actually are contained in a sequence activity of the domain acting by ‘students’. As we provide the constraint that the main actors

must be found at the subject position. This may become an ambiguity, where the system of automatic requirement classification cannot understand that the term ‘*they*’ refers to the term ‘*students*’. Therefore, S_2 may be not retrieved and classified into the same group containing S_1 .

However, the results of the SVM classifier models are quite good in terms of accuracy. Thus, these concepts may be able to help to easy to understand and leverage any requirement from a requirement document better. Therefore, in the final, we obtain three main specific requirement categories. They are the categories of requirements relevant to students, professors, and registrar respectively. As the results, we now consider these requirement categories as scenario categories. Consequently, it is to find the activities in each scenario.

4.5.3 Extracting scenarios by Closed-domain Knowledge Extraction

In this step, it shows how to extract scenarios from requirement documents. We utilize the concept of closed-domain question answering to find the real need in the documents. The modified process of extracting scenarios can be described as follows.

(a) Specifying questions

It is to provide the main questions that may help identify the expected results. This can be performed manually. There are two possible questions that are necessary for this task.

- *Who performs each task in the system?* and
- *What is a task of someone in the system?*

(b) Questions analysis and identifying types of answer

This is to extract the keywords from a question, interprets them, and then these keywords are expanded to other relevant terms.

It is noted that this work has not driven on the TREC standard¹² (TREC 8, 9, 10, and 11). This is because this work utilizes only the concept of QA to more specific domain of software engineering, where we provide the questions as the direction of analysing the requirements of a system and the answers are the actors and their actions in the system. Finally, these results are used to design a software application. The process of question analysis and identifying type of answer can be shown as Figure 4.9.

¹² TREC (Text Retrieval Conference) has had a question answering track since 1999. This track is an effort/standard to bring the benefits of evolution to bear the question answering problem. It aims to develop the true equivalent of a standard retrieval test collection is an open problem. (Referred to <http://trec.nist.gov>)

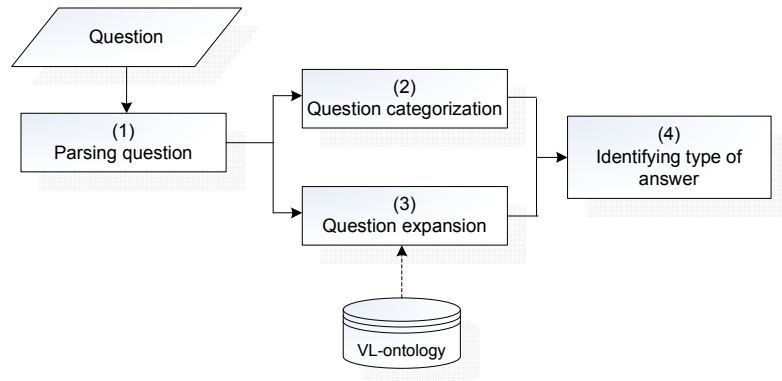


Figure 4.9 Analysing questions and identifying types of answers

In Figure 4.9 when a question is input to our system, it will be transformed to a structure explaining sequences of words in a natural language, called *parsing*. Parsing text has often driven on POS tagging. An example of parsing rules used in this context can be presented as Table 4.6. The parsing rules can be developed based on context free grammar (CFG)¹³.

Table 4.6 Examples of parsing rules

<question-form>:	<question-class>< auxiliary-verb><subject><main-verb><object><complement> <question-word>< auxiliary-verb><main-verb><complement> <question-word><main-verb><complement> <question-word><main-verb><object><complement> <question-word>< auxiliary-verb><subject><main-verb> ...
<question-class>:	what whol when why where
<auxiliary verb>:	do will did have has can shall ...
<main -verb>:	process perform ...
<complement>:	noun phrase cause
<subject>:	noun phrase
<object>:	noun phrase
...	...

Consider the question, “*who perform each task in the system?*”. An example of parsing can be shown as Figure 4.10. After parsing the question, two sub-processes are consequently performed: question categorization and query expansion. Question

categorization is to identify the question types. This sup-process affects for the step of answer extraction, where extracting answers from sentences has driven on the question types.

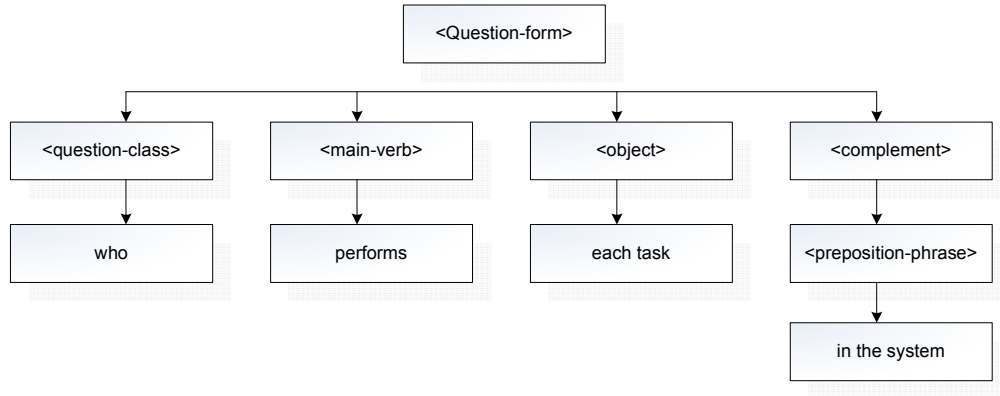


Figure 4.10 The process to analyse questions and identify type of answers

In general, questions in QA can be classified into five categories such as who, why, when, which, and what. The question used in class ‘who’ always refers to a person. Meanwhile, the question used in class ‘what’ can be classified into four sub-classes: what (referring to something, doing something), what-who (referring to a person), what-where (referring to the location), and what-when (referring to date/time). In fact, the questions are provided to be used as mechanisms for identifying the relevant answer. Hereafter, this is to expand the query, called query expansion. In QA, a query is a keyword extracted from the question. In this context, we utilise the VL-ontology and WordNet to expand the keywords that are extracted from main question(s). The VL-ontology is used to group the word variations into the same concept. For example, the question used in class ‘who’ always refers to a person. Therefore, ‘person’ is a keyword of the question ‘who’. The word ‘person’ can expand to the words ‘human’, ‘individual’, and ‘someone’. However, the word ‘individual’ can be changed its form to ‘individuals’. The word ‘individuals’ should be an expansion of the word person’ by analysing through the VL-ontology.

Consider the first question, “who perform each task in the system?”. The answer of ‘who’ is a person. The keyword ‘person’ can expand to ‘human’, ‘individual’, and ‘someone’. In this context, ‘person’ can refer to ‘students’, ‘Professors’, and ‘registrar’ in final.

¹³ Context free grammar (CFG) is a formal grammar in which every production rule is of the form, $V \rightarrow w$, where V is a *single* nonterminal symbol, and w is a string of terminals and/or nonterminals (w can be empty). (Referred to http://en.wikipedia.org/wiki/Context-free_grammar)

Consider the second question, “*what is a task of someone in the system?*”. The question class ‘*what*’ indicates to ‘*doing something*’, where the keywords of this question are ‘*action*’ and ‘*someone*’. The keyword ‘*action*’ can be expanded to ‘*activity*’, ‘*performance*’, and ‘*operation*’. Meanwhile, the keyword ‘*someone*’ can be expanded to ‘*somebody*’, ‘*individual*’, and ‘*one*’. This implies that ‘*someone*’ refers to the keyword ‘*person*’. Finally, the type of answer of each question can be identified.

(c) Development of answering pattern

To extract the answer that is relevant to the question from requirements, it is necessary to provide a tool used to search for answers (person and its action). The main technique used in this work is called ‘*pattern matching based*’. This technique is to extract answers from text through formalized pattern based on *N*-gram model. The process of answering pattern development can be concluded as the following algorithm.

1. A document is broken into sentences. A sentence should be simple. A boundary of a sentence can be truncated by a full stop.
2. Tokenizing a sentence and performing with POS tagger to assign the words in a text with their corresponding parts of speech
3. Finding the verb position, and generating the *verb-part* based on *N*-gram model.
4. Finding the object position (if it has). It helps describing what action is. The *object-part* is also generated.
5. Finding the subject position, and generating the *subject-part* based on *N*-gram model.
6. Each sentence is transformed into a binary tree structure. The root node must be the verb.
7. Considering each binary tree as a pattern

For example, consider the sentence ‘*Students can select four courses for the coming semester*’. Firstly, this sentence is tokenized, and it can be represented below.

Students / can / select / four / courses / for / the / coming / semester

The second step uses a POS tagger to assign the words in a text with their corresponding parts of speech. An example can be shown as Table 4.7.

Table 4.7 The results of POS Tagging

Words	Abbr.	Description
students	NN	Noun (single)
can	MD	Modal
select	VBP	Verb (present tense)
four	JJ	Adjective
course	NNS	Noun (plural)
for	IN	Preposition
the	ART	Article
coming	JJ	Adjective
semester	NN	Noun (single)

The third step is to find the verb position based on N -gram model. If the verb is identified, the subject and the complement are consequently found. In this example, the verb position is ‘*can select*’. The word ‘*can*’ is the modal verb, while the word ‘*select*’ is the verb which is in the present tense. As this, a *verb-part* can be generated as $\langle MD \rangle \langle VBP \rangle$. However, it can be illustrated an example of generating rules based on N -gram model. Suppose a corpus contains 100 sentences, there are 10 sentences having $\langle MD \rangle \langle VBP \rangle$. By using N -gram model, it can be calculated the probability of $\langle MD \rangle$ followed by $\langle VBP \rangle$ as follows.

$$\begin{aligned} Pr(w|\langle MD \rangle \langle VBP \rangle) &= 10/100 \\ &= 0.1 \end{aligned}$$

As above, the bigram $\langle MD \rangle \langle VBP \rangle$ is a model of the verb-parts that can be generated from the corpus, and its probability of the model $\langle MD \rangle \langle VBP \rangle$ is 0.1 .

The fourth step is to find the object position. It always follows the verb position. In this context, we consider the rest of the sentence that is following the verb as the object. For this example, an *object-part* can be generated as $\langle JJ \rangle \langle NNS \rangle \langle \text{Preposition-phrase} \rangle$.

The fifth step is to find the subject position. It is always represented in the front of the verb position. The word ‘*students*’ is a plural noun. As this, a noun-form can be generated as $\langle NNS \rangle$.

The sixth step is to transform the *verb-part*, the *subject-part*, and the *object-part* into a binary tree structure. The root node is the *verb-part*. The left node is the *subject-part*, while the right node is the *object-part*.

Finally, each binary tree is transformed as an answer pattern used to extract the reality need in requirement specification.

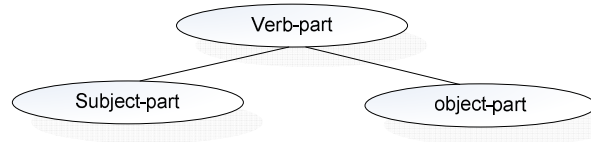


Figure 4.11 A binary tree representing the format of answer pattern

However, the answer patterns are developed based on two main questions, ‘*who perform each task in the system?*’ and ‘*What is a task of someone in the system?*’. Therefore, we develop two types of answer patterns.

(i) The answering pattern for the question ‘*who perform each task in the system?*’

As we said that, the question class ‘*who*’ always refers to a person. In this context, the term ‘*person*’ can be ‘*students*’, ‘*professors*’, and ‘*registrar*’. These terms are used as queries. The objective of this question is to find ‘*actor*’ that performs a task in the system. In general, the subject of a sentence indicates to the actor. Some patterns of answer used to find *actor* in a requirement sentence can be presented as Table 4.8.

Table 4.8 Examples of answering pattern for the question “*Who perform each task in the system?*”

Question:	Who does perform each action in the system?
Major target answer:	Person (as actor)
Standard Form:	<subject> <verb-part> <articl><subject><verb-part>
Subject form:	<noun> <noun-phrase>
Specific Query:	‘students’, ‘student’, ‘professors’, ‘professor’, and ‘registrar’
General Pattern:	<query> <verb-part> <query> <verb-part>
Examples of specific answering patterns:	<students>+<verb-part> ‘students’ + <verb-part> + ... + (.) They + <verb-part> <article><students>+<verb-part> <Professors>+<verb-part> <article><processors>+<verb-part> ...

Consider two examples of the answering patterns. The pattern [‘*students*’ + <verb-part>] means that if the query/keyword ‘*students*’ are found, the rest of the sentence containing the verb and something, called <verb-part>. This verb-form is an action of the actor ‘*students*’. Another pattern [‘*students*’ + <verb-part> + ... + (.) They + <verb-part>]

means that if the query ‘*students*’ is found at the subject position of the first sentence; the next sentence may be the consequent action of the same subject.

As the example of an answering pattern in Table 4.7, it can show how to use the answering pattern to extract the actor from a requirement sentence following. Consider the sentence ‘*the professors can record the grades for the students in each class*’. After tokenizing this sentence, it can be represented below.

The / professors / can / record / the / grades / for / the / students / in / each / class

By using the answering pattern [<article>‘*professors*’+<verb-part>], this sentence will be separated into two parts shown as Figure 4.12.

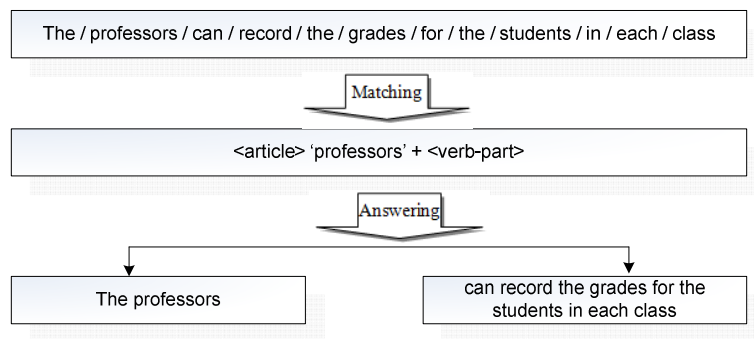


Figure 4.12 An example of using the answer pattern

(ii) The answering pattern for the question ‘What is a task of someone in the system?’

Similarly, consider the question ‘*What is a task of someone in the system?*’. In this case, the question is to find an *action* that is performed by *actors*. An action can be found in the verb position following the subject (also known as an *actor*). Some patterns of answer used to find *action* in a requirement sentence can be presented as Table 4.9.

As the example of answering pattern in Table 4.9, it can show how to use the answering pattern to extract the actor from a requirement sentence following. Consider the rest of the requirement sentence ‘*the professors can record the grades for the students in each class*’. It is ‘*can record the grades for the students in each class*’. Later, by using the answering pattern [<modal-verb><verb> + <object> + <preposition-phrase>], this part will be separated into three parts shown as Figure 4.13.

Table 4.9 Examples of answering pattern for the question “What is a task of someone in the system?”

Question:	What is a task that is performed by person?
Major target answer:	Doing something (activity/ task/ action)
Standard Form:	<Verb> + <object> <Verb> + <object> + <complement>
Query:	-
General Pattern:	-
Examples of answering patterns:	<verb> + <object> <verb> + <object> + <preposition-phrase> <auxiliary-verb><verb> + <object> + <preposition-phrase> <auxiliary-verb> not <verb> + <object> + <preposition-phrase> <auxiliary-verb> not <verb> + <object> <modal-verb><verb> + <object> <modal-verb><verb> + <object> + <preposition-phrase> <model-verb> not <verb> + <object> ...

The process to develop the answering patterns has driven on semi-automated answering pattern extraction, where the rules (as patterns) are generated manually based on the English grammar rules, and then these patterns are automatically tested and selected the appropriate patterns by using a simple probabilistic technique.

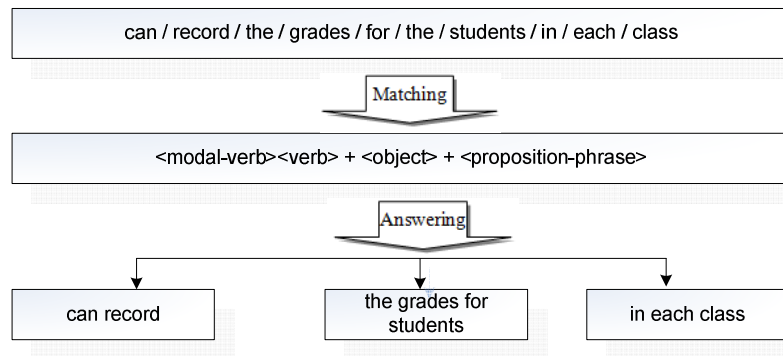


Figure 4.13 An example of using the answer pattern [<modal-verb> <verb> + <object> + <prepositional-phrase>]

(d) Passage Retrieval

The original definition of passage retrieval is to retrieve only the portions of a document, if they are relevant to particular information need (Baeza-Yates and Ribeiro-Neto, 1999, Callan, 1994, Hearst and Plaunt, 1993, Salton et al., 1993).

After the requirement document is broken into sentences. Next, we used text classifier models based on the One-Class SVM algorithm to group the relevant requirements into the same class. Finally, the requirement sentences can be classified into three classes. They are requirements for *students' performance*, *professors' performance*, and *registrar's performance*.

However, to design a software application, it is necessary to make relationship among actors. Therefore, passage retrieval in this context is a process to retrieve some documents (sentences) containing two or more actors. Such these documents bring us to draw the association among actors in the system. Consider following document.

“For some course, students must have permission from professors before enrolling for the course”

This document shows an association between two actors in the system (*students* and *professors*). We use the keyword *students* and *professors* to retrieve this document. It is to imply Boolean logic with keyword searching technique. Boolean logic allows us to combine words and phrases into search statements to retrieve relevant documents from document repositories. In this case, we use the Boolean logic *AND*. Therefore, only documents containing the keyword *students* and *professors* are retrieved. All of pairs of keywords are (*students AND professors*), (*students AND registrar*), and (*professor AND registrar*).

(e) Scenario Discovery in Software Requirements

This process utilizes the concept of answer extraction to find actor and actor's action in a requirement sentence. When all of the actions are extracted, these are integrated as the detail of a scenario. The process of extracting actor and actor's action in a requirement sentence consists of three sub-processes (shows as Figure 4.14): (i) *text pre-processing*, (ii) *extracting actor* and (iii) *extracting actor's action*. They can be elaborated as follows.

In Figure 4.14, when a requirement sentence is input to our system, the first sub-process is *text pre-processing*. It is firstly performed with the morphological analysis. It is to segment words shown as below.

Students/ can / select/ four / courses/ for / the / coming / semester

Next, it is a syntactic analysis used to specify the possible organization of word in a sentence. A POS tagging is used to assign the words in a text with their corresponding parts of speech. An example can be shown as below.

<i>students /</i>	<i>can /</i>	<i>select /</i>	<i>four /</i>	<i>courses /</i>	<i>for /</i>	<i>the /</i>	<i>coming /</i>	<i>semester</i>
<NN>	<MD>	<VBP>	<JJ>	<NNS>	<IN>	<ART>	<JJ>	<NN>

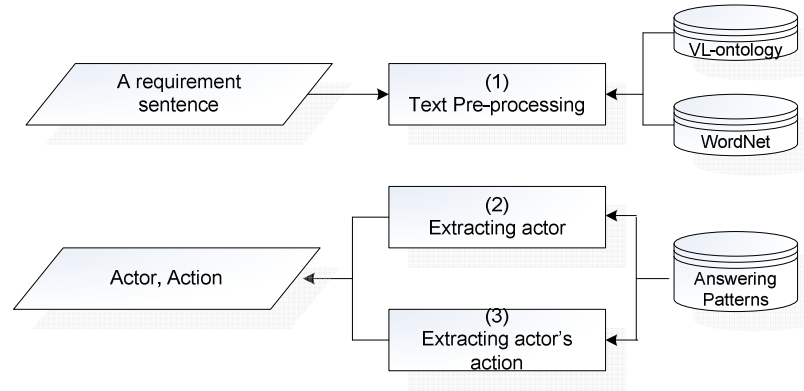


Figure 4.14 The process of extracting actor and actor’s action in a requirement sentence

The next sub-process is to extract the ‘actor’ of the requirement sentence. As we said at the start of this paper that the subject is consequently found, if the verb is identified. After pre-processing of text, the verb position is found. Therefore, in front of the verb is the subject. By using the answer pattern [‘students’ + <verb-part>], the term ‘students’ is extracted as the ‘actor’ of this requirement.

Consequently, it is to find the action of the actor ‘students’ in this case. The rest of the requirement sentence is analysed to find the real action in this requirement. By using the pattern [<modal-verb><verb><object> + <prepositional-phrase>], the rest of the requirement sentence can be separated into three portions shown as follows.

<modal-verb><verb><object>: can select four courses
 <prepositional-phrase>: for the coming semester

As above, the actor’s action in this example is ‘can select four courses’. The summary of answer extraction applied to find the actor and the actor’s action of the requirement sentence ‘Students can select four courses for the coming semester’ can be concluded as 4.15.

```

    <?xml version="1.0" encoding="windows-874"?>
    <Scenarios> a course registration system
      <actor="students">
        <main-task> course-registration
          <sub-task> select four course </sub-task>
        </main-task>
      </actor>
    </Scenarios>
  
```

Figure 4.15 The final results of extracting the actor and the actor’s action in the requirement sentence “Students can select four courses for the coming semester”

4.5.4 Evaluation of Discovered Scenarios

This section shows an example of scenarios that is extracted from software requirements, representation of scenarios, including the performance evaluation.

(a) Scenario Representation

After a scenario is extracted from software requirements, it will be stored in the form of XML schema. An example can be shown as Figure 4.16.

In Figure 4.16, the example presents the results of finding scenarios based on the ON-KDT methodology. An original requirement will be executed to search for the relevant activities in the same story (or class), and combine them as a scenario. In the example, the course-registration is a process of scenario that will be done by students. This scenario shows what students can do to take action in this process.

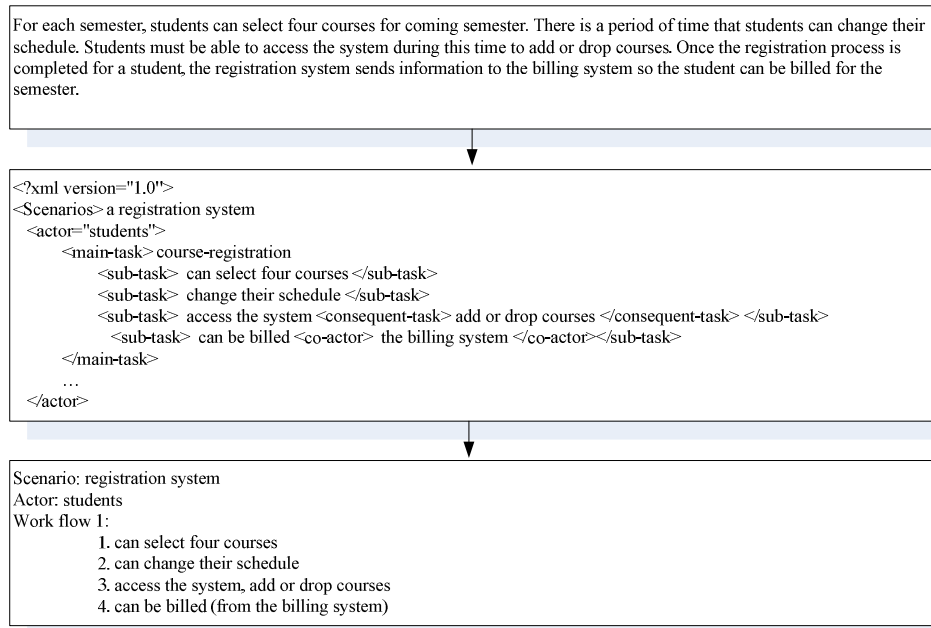


Figure 4.16 An example of scenario representation

(b) Performance evaluation and discussion

In this context, we apply three evaluation measure techniques: recall, precision, and *F*-measure. We evaluated three results of the experiments for scenario extraction: scenario of students, scenario of professors, and scenario of registrar. In fact, the scenario of students contains 60 requirement sentences, the scenario of professors contains 65 requirement sentences, and the scenario of registrar contains 70 requirement sentences. In this work, there are 24 answering patterns for actor extraction and 35 answering patterns for action extraction

that are provided to extract all activities of each scenario. The experiments for scenario extraction can be presented as Table 4.10.

In Table 4.10, it can be seen that the results of scenario extraction are quite good in terms of accuracy. Consider the recall measurements. In fact, these measurements obtained from the results of automatic requirement classification by the SVM classifiers shown in Table 4.10.

Meanwhile, the precision measurements are the results of scenario extraction evaluated by a domain expert. Although the results of the precision measurements are quite satisfactory, they also show the failure rate. The background of this problem is form sentence structure. If a sentence is simple, it has no any problem. However, if the sentence is a compound or complex, we may face a big problem. For example, consider following complex sentence.

Table 4.10 The experiments for scenario extraction

Information	Scenarios		
	Students	professors	registrar
All requirement sentences	60	65	70
Retrieved (the number of sentences)	55	59	62
Retrieved & Correct (the number of sentences)	47	50	51
Evaluations (%)	Students	professors	registrar
Recall	91.66	90.76	84.47
Precision	78.33	76.92	83.27
<i>F</i> -measure	84.47	72.86	79.95

“The Student can also update or delete course selections if changes are made within the add/drop period at the beginning of the semester.”

The actor of this performance is ‘students’. However, it may confuse about the actions of the students in this sentence. In fact, if this is performed manually, ‘*can also update or delete*’ is extracted as the action. Unfortunately, our answering patterns for extracting action do not cover for this case, where <modal><adv><verb>+ or + <verb> is not provided.

As the problem above, it may lead to the accuracy of scenario extraction being decreased. Therefore, to handle this problem, a better solution is to provide many pattern rules to support any problem.

4.5.5 The Use of Discovered Scenarios

As the scenario is defined as a description of a set of users, a work context, and a set of tasks that users perform or want to perform, the description sufficiently detailed so the design implications can be inferred and reasoned about (Carroll, 1995).

Scenarios can be applied and used in several stages of the software development life cycle, often in different ways, although they are better known to be used at the requirements stage.

Apart from using scenarios for capturing requirements, there are also attempts to use them in requirements analysis stage (Sutcliffe, 1998). Sometimes, scenarios are linked to use cases in the Unified Modelling Language (UML). In that context, each scenario is an instance of a use case (Sutcliffe et al., 1998, Grobelnik et al., 2000a).

We will show one of basic strategies is to identify a path through a use case. An example of the usage of scenarios that are extracted from the software requirements can be described following. Consider the scenario in Figure 4.16. The scenario can be transformed to the form of the use case and shown it as Figure 4.17.

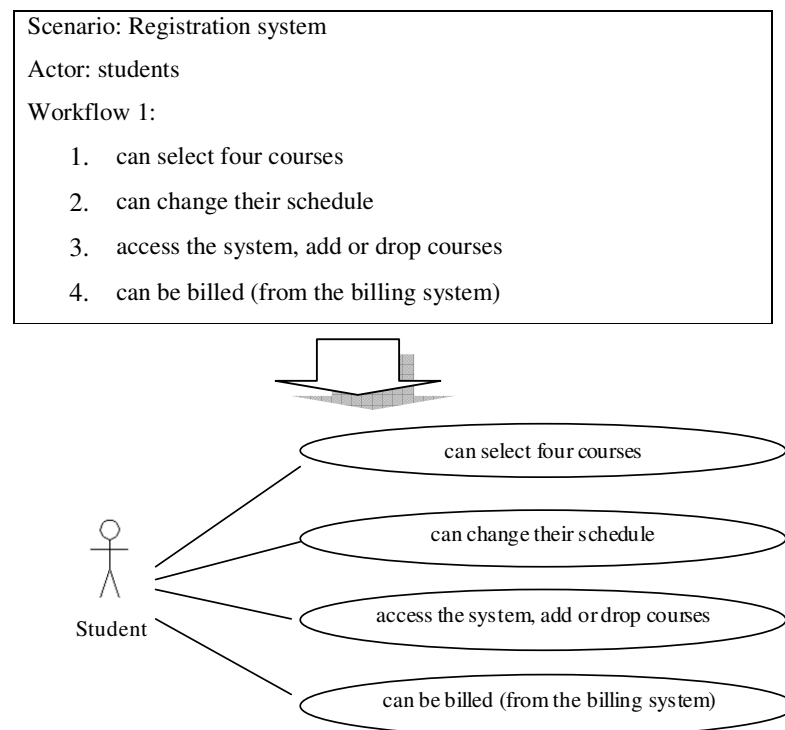


Figure 4.17 An example of the use of scenario transforming to the use case form

4.6 Chapter Conclusion

Scenarios are sufficient for existing systems because they can be used to portray paths of possible behaviours or activities or tasks, and they have been used as a way to deduce the system requirements. Despite the increasing attention of scenario analysis in requirement engineering, there are few works that have proposed methods to discover scenarios from requirement artefacts. This is because it may not be easy to elicit scenarios from requirement artefacts. In addition, many requirement specifications are written in a natural language format, which makes it hard to identify the real needs of requirements because they are ambiguous for specification purposes. Therefore, in this particular domain, the dataset is a collection of requirement specification documents, and the motivation of this part is driven by finding and extracting knowledge as scenarios from text. Developing scenarios from requirements is an important activity in requirements engineering, because they are very useful for software analysis, design and testing.

In this chapter, we also apply the conceptual model of the On-KDT methodology to automatically extract scenarios from software requirements. The empirical results show that our approach can provide more effectiveness for extracting scenarios from requirement specification documents. However, it is also noted that, the sequence of tasks, that are extracted, depend on the sequence of sentences in requirements.

CHAPTER 5

THE ON-KDT METHODOLOGY APPROACH FOR EXTRACTING CLINICAL KNOWLEDGE FROM PUBMED ABSTRACTS

As it said in the chapter 3 that, this work aims to propose the conceptual model of the On-KDT methodology which supports for extracting useful knowledge from unstructured data describing with text. In this chapter, it aims to apply the proposed methodology to extract clinical trial knowledge from PubMed abstracts which are in particular cervix cancer.

5.1 Background

PubMed is the National Library of Medicine's search engine designed to search MEDLINE¹⁴. MEDLINE (Medical Literature Analysis and Retrieval System Online) is a large repository of health information which covers a huge range of results regarding different disease types, symptoms, treatments, disease causing factors (genetic and environmental) as well as candidate genes that could be responsible for the onset of these diseases. Information regarding illness is dispersed over various areas such as health research and disease treatment. It has accumulated 18 million biomedical-related abstracts of bibliographic information stored in electronic format, and they continue to accrue at an exponentially increasing rate. To cope with this increase in published information, an appropriate, effective, efficient, and easy-to-use retrieval tool is necessary. PubMed uses MeSH (Medical Subject Headings) to retrieve relevant documents from MEDLINE, where MeSH is the PubMed assistant to support users locating appropriate terms for MEDLINE searches. MeSH was developed over tree structure. It is a corpus of vocabularies used to control PubMed search. For example, if users search for '*cervix cancer*' documents, all other possible definition, implied meaning, or literal meanings (e.g. *Uterine Cervical Neoplasm*, *Cervical Neoplasms*, *Cervical Neoplasm*, *Uterine Cervical Cancer*, *Cancer of Cervix*, and *etc.*) can be used to support the search in MEDLINE in order to obtain more relevant documents.

The traditional knowledge discovery from PubMed abstracts involves the use of several manual processes. The health professionals enter search terms into a web portal to produce a clinically relevant abstract retrieval (Hadzic et al., 2008). In Yu *et al.* (1988), they

¹⁴ www.ncbi.nlm.nih.gov/pubmed

also said that many biomedical researchers have to spend a crucial amount of times in the literature search process. This is because searching relevant documents from the PubMed has been challenging because of two main problems. Firstly, PubMed is the biggest biomedical literature repository that contains more than 18 million articles and still keeps growing (Cutting et al., 1992, Church, 1988). Large volume of search can be made more difficult for accessing to find relevant documents. Secondly, it is hard to express the specific relevance of users in the given keyword query (Church, 1988). As the result, many results are typically retrieved. Therefore, many researchers still pay attention to improve the quality of search on PubMed.

However, biomedical researchers require not only an efficient search engine system, but also a tool used to automatically search for synopsis knowledge from PubMed. In reality, it is never easy to extract useful knowledge from PubMed. Traditionally searching for relevant knowledge from PubMed is a process that, the user reads the article's title to consider appropriateness and may then read the abstract to determine relevance. Indeed, this is time-consuming and labour-intensive, especially when a search returns a large number of abstracts and where each abstract contains a large volume of information (Vaka and Mukhopadhyay, 2009).

In the last decade, the need for discovering the hidden knowledge from PubMed is also required, where automatic knowledge discovery allows the researchers to conduct quality work, avoid repetition, and generate new hypotheses (Merialdo, 1994). Therefore, many researches also pay attention to study about how to extract knowledge from PubMed. For examples, Mendonça & Cimino (2000) studied the possibility of utilizing the co-occurrence of MeSH terms in MEDLINE citations associated with the search strategies optimal for evidence-based medicine. Afterwards, all evidence-based medicines will be stored as a knowledge base.

Pustejovsky *et al.* (2001) presented a system, called Acromed, which finds acronym-meaning pairs as part of a set of information extraction tools designed for processing and extracting data from abstract in the MEDLINE database. The results of this system are entered into a large, continuously updated database of acronyms for the biomedical literatures.

Kumar *et al.* (1994) investigated and developed a complete knowledge base, called BioMap. This database uses the entire MEDLINE collection of (over 12 million) bibliographic citations and author abstracts from over 4600 biomedical journals worldwide. They also developed an interactive knowledge network for users to access this secondary knowledge (BioMap) along with its primary databases such as the MEDLINE.

Demner-Fushman & Lin (2007) proposed a series of knowledge extractors, which employ a combination of knowledge-based and statistical techniques, for automatically identifying clinically relevant aspects of MEDLINE abstracts. The system started with an initial list of citations retrieved by PubMed, and then their system brings relevant abstracts into higher ranking positions. Finally, the MEDLINE abstracts will generate answer that directly responses to physicians' questions. They showed the experiments on a collection of real-world clinical questions that this approach significantly outperforms the already competitive PubMed baseline.

Song et al. (2011) proposed how to extract procedural knowledge rather than declarative knowledge. This technique applies machine learning method with deep language processing features.

5.2 Motivation

In order to fully employing these on-line documents effectively, it is crucial to be able to find and extract the synopsis information in these documents. Finding useful clinical trial knowledge related to cervix cancer from PubMed abstracts is a challenging problem in this chapter, where the goal of this chapter is to extract clinical knowledge relevant to cervix cancer from MEDLINE abstracts.

5.3 Research Contribution

In order to fully employing these on-line documents effectively, it is crucial to be able to find and extract the synopsis information in these documents. Extracting useful clinical trial knowledge relating to cervix cancer from PubMed abstracts becomes a challenging problem in this chapter. Therefore, the goal of this chapter is to discover clinical knowledge relevant to cervix cancer from MEDLINE abstracts.

This chapter presents an advantage that the On-KDT methodology driven over natural language processing technology can bring to extract embedded knowledge from biomedical literatures. It aims to extract clinical knowledge relevant to cervix cancer from MEDLINE abstracts. The overview of the approach can be explained as follows.

This approach starts with two manual steps: (1) *Understanding of the application domain and defining the problem* and (2) *Understanding MEDLINE abstracts*. The first step is to acquire the relevant prior knowledge and the goals of knowledge discover. This is because we need to know what to look for in the MEDLINE abstracts. In this case, the directions for finding useful knowledge are in form of questions determined by an expert. Therefore, the type of answer required is related to the question. This can provide constraints

on what constitutes relevant data, which can help other modules to correctly locate and verify an answer (or knowledge).

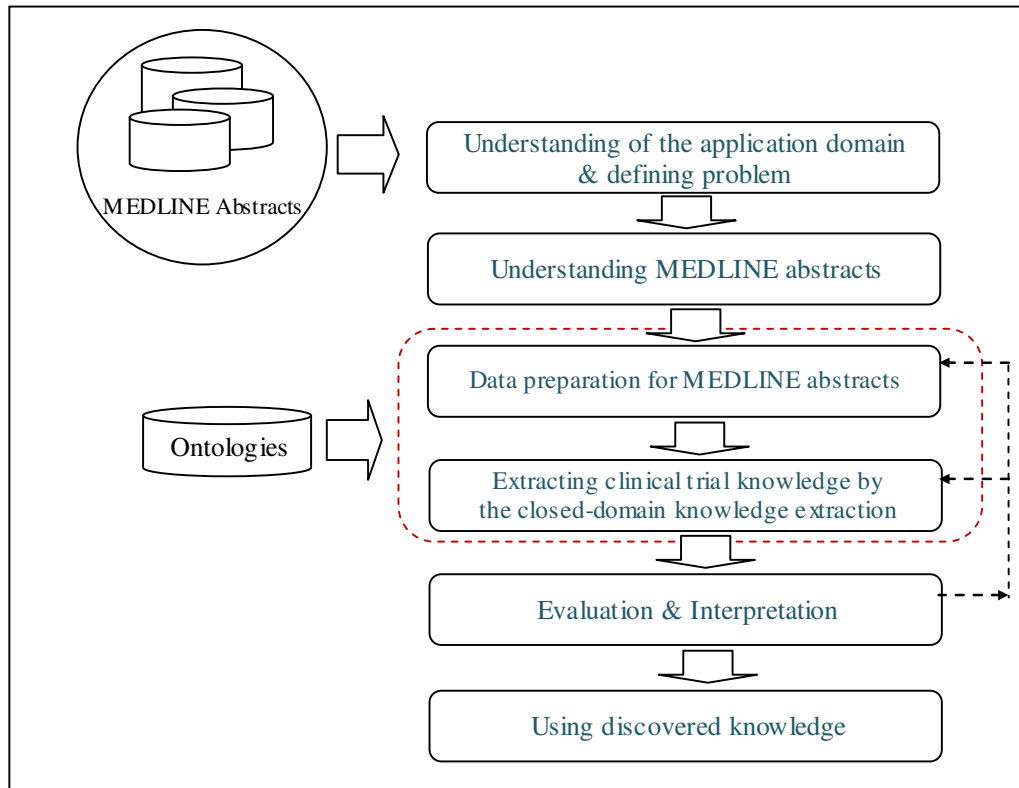


Figure 5.1 The On - KDT methodology approach for extracting clinical trial knowledge from PubMed abstracts

In this context, the PubMed abstracts relevant to cervix cancer are used as the case study because cervix cancer was reported by WHO (World Health Organization) (Pagliusi, 2010) that cervix cancer is the second most common cause of cancer-related disease and death among women worldwide with 288,000 deaths yearly. About 510,000 cases of cervical cancer are reported each year with nearly 83% in developing countries, representing 15% of female cancers: 68,000 in Africa, 77,000 in Latin America, and 245,000 in Asia. In Australia, cervix cancer is the 13th most common cancer in women¹⁵, while in Thailand¹⁶,

¹⁵ <http://www.abc.net.au/health/library/stories/2005/11/29/1829319.htm>

¹⁶ WHO (2010) Thailand: Human Papillomavirus and Related Cancers, WHO/ICO HPV Information Centre (3rd edition).

cervix cancer is second in frequency in women. As the results, it can be seen that cervix cancer is one of the major public health problems for woman in the world which should be controlled in order to maximize the sufferers' chance of survival.

To obtain the most relevant knowledge from a context, we commence our work by providing seven main questions as follows.

- (1) *What is the histology of cervix cancer?*
- (2) *What is the stage of cervical cancer?*
- (3) *How many patients are in a study?*
- (4) *What are arm-names used?*
- (5) *What treatment modalities are used?*
- (6) *What is timing of each treatment modality used?*
- (7) *What are drugs used for treatment?*

The questions above are provided by a domain expert. These questions are used as goals in this context. Therefore, the primary objective is to find knowledge from MEDLINE abstracts relevant to the cervical cancer in clinical trials.

The second step focuses on the selection of data samples that are relevant to the objective of knowledge discovery. Also, this step starts with an initial list of relevant abstracts retrieved by the PubMed search system.

Later, the third step is *data preparation*. The goal of this step is to identify more specific individual data because the ability of extracting knowledge from PubMed depends on something more specific than broad, unspecified knowledge. In this step, it consists of two main tasks: (1) *data pre-processing* and (2) *identifying more specific individual data*. Data pre-processing is a task to transform the desired subset into a format that is suitable for the next step, *identifying more specific individual data*. In this case, the sub-process of identifying more specific individual data is to construct a text classifier (used as text filters). We develop text classifiers based on the questions determined by the domain expert. Therefore, in this context, we have seven text classifiers, and then we use these text classifiers to search for the most relevant passages in an abstracts that are related to the question. This is because the answer of the question may be in the selected passages.

The fourth step, called *closed-domain knowledge extraction*, is a process to extract knowledge. Due to applying the concept of closed-domain question answering, extracting knowledge from MEDLINE abstracts can be done by answering patterns. Therefore, we need to provide the answering patterns used to identify answer (also known as knowledge) that satisfies the specific questions.

It is noted that the steps of *data preparation* and *closed-domain knowledge extraction* utilize the VL-ontology to understand the contextual meaning of terms as they appear in the searchable data-space, if the unstructured data is text. However, using only the VL-ontology to understand the contextual meaning of terms may not be sufficient because biomedical literatures contain specific biomedical terms and some of these terms are changed into their forms by the domain experts. Therefore, in this chapter, it presents a new ontology used to understand the meaning of biomedical terms, especially the terms used in cancer. The new ontology is called the *cancer term net ontology* (*CCT-net ontology*).

The fifth step is to evaluate the patterns/knowledge discovered to make the decision of what qualifies as knowledge. The standard techniques with precision, *recall*, and *F*-measure can be applied to evaluate the knowledge discovered. Finally, it describes how to use the extracted knowledge in the reality.

5.4 Preliminary: The cancer term net ontology (CCT-net ontology)

This section describes how to implement the CCT-net ontology which was developed under the supervision of an oncology domain expert.

5.4.1 Background of the CCT-net ontology

The CCT-net (Cancer Term Net Ontology) was developed to support the process of content-based text analysis. It is useful for two reasons. First, a set of prior knowledge can help to identify the particular domain that users employ when searching. Second, the ontology can handle the problem of term variation in the specific domain.

The background problem that points to the need for the ontology is illustrated here. In cancer medicine, the term ‘*chemotherapy*’ and ‘*radiotherapy*’ are changed by domain experts when they are used together. They can be represented as abbreviations, e.g., ‘*CTh/RTh*’, ‘*CT/RT*’, ‘*CTRT*’, or as a combined term, e.g., ‘*chemoradiotherapy*’, or ‘*chemoradiation*’. If these terms are made understanding of human, term variation is not a problem. However, if these terms are automatically analysed by a computing system, this becomes a significant problem, called *the problem of ambiguity in language*. This problem can be seen to occur in an article that describes *chemoradiation* in the abstract has no MEDLINE heading coded that explicitly indicates radiotherapy occurred (Accessed at DOI: // 10.1111/j. 1440-1673.2005.01546. X; ID:16499727).

At present, it has no any tool that can be used to address this problem. The pre-existing ontologies such the National Cancer Institute’s Thesaurus and Ontology (Golbeck et al., 2003) do not offer information relating to a domain specific variations in terms used by domain experts.

5.4.2 Definition of the CCT-net

The entries of CCT-net are term-concepts mapping. The associations in the mapping are conceptual and term-concept relations harness synonyms or term variations. For example, the concept ‘*specific drugs used*’ is similar to the concept ‘*chemotherapy*’ because chemotherapy describes a particular form of drug therapy. However, ‘*chemotherapy*’ is also a distinct treatment concept contained within ‘*therapeutic modalities*’. These associations can be shown as Figure 5.2.

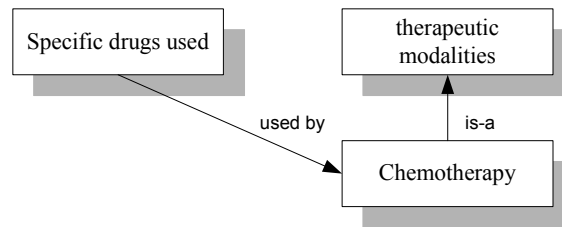


Figure 5.2 An example of the associations in the mapping between term-concepts

The CCT-net consists of five main concept mappings: (1) histology, (2) disease stage, (3) therapeutic modalities, (4) timing of therapeutic modalities, and (5) specified drugs used.

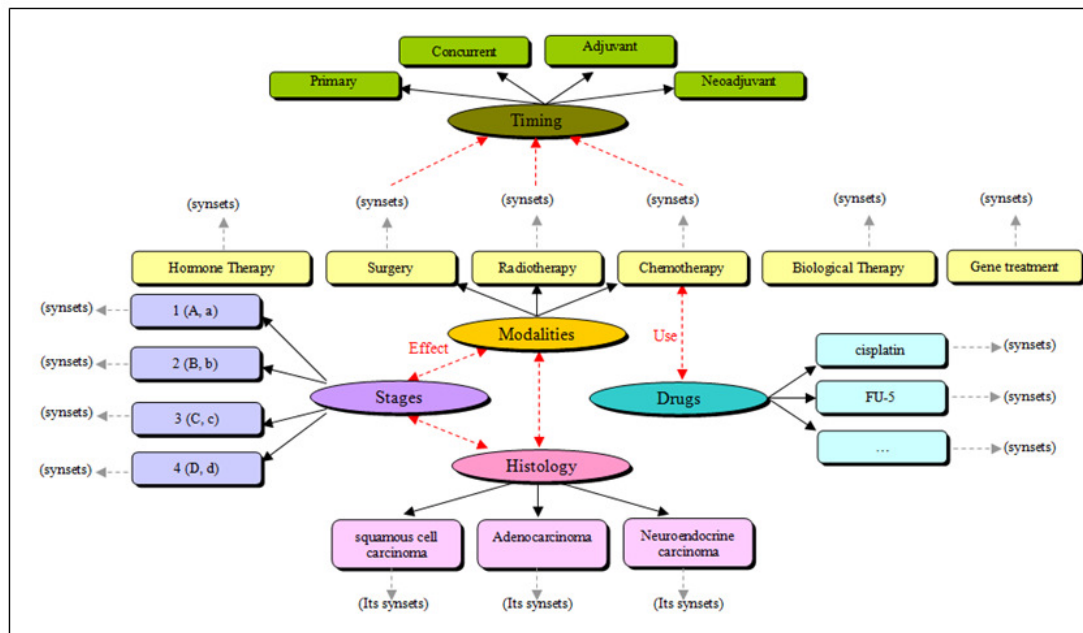


Figure 5.3 The overview of the CCT-net ontology

The importance of each concept can be described as follows.

(a) Histology

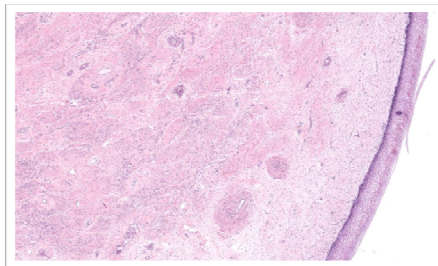
The tissues and cells of cancer are described on the basis of their light microscopic appearance. Cervical cancer histology is broadly classified into three main types: squamous cell carcinoma, adenocarcinoma, and neuroendocrine carcinoma.

Squamous cell carcinoma (SCC) arises in epithelial tissues and can also arise in multiple organs such as skin, lips, mouth, lungs, vagina, and cervix. SCC develops in tissues lined by squamous *epithelium* which usually means that they experience high wear and tear and/or contact with air. Some examples are shown as Figure 5.4.

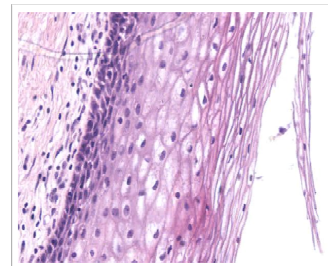
Adenocarcinoma is another epithelial type of cancer that develops in the lining of glandular tissues. Glandular epithelial tissues include sweat glands in the skin and a variety of organs that produce secretions such as lung, breasts, colon, prostate, stomach, and pancreas. An example is shown as Figure 5.5.

Adenosquamous carcinoma is a type of cancer containing both types of cancer cells, that is a mixture of squamous and adenocarcinoma cells.

Neuroendocrine carcinoma is the most aggressive histopathological type and is thankfully rare. They are usually detected when in an advanced stage, which makes cure impossible. An example is shown as Figure 5.6.



(a) A closer view of the squamous epithelium showing the layers of flattened cells (squames) (x20).



(b) Squamous cell carcinoma of the cervix (x10)

Figure 5.4 Squamous cell carcinoma

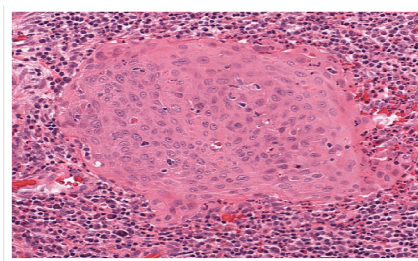


Figure 5.5 Adenocarcinoma (x10)

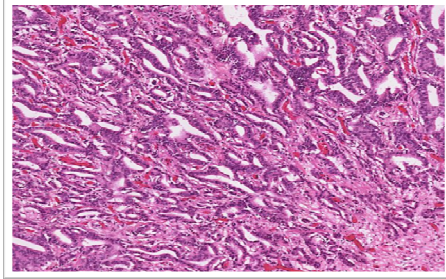


Figure 5.6 Neuroendocrine carcinoma (x10)

(b) Disease stage

This concept relates to a formal description of the extent of cancer spread. There is an overall stage of a cancer which is usually represented as numbers *I* to *IV* with *IV* being the worst. More specifically, cancer staging is formalised in the *TNM Classification of Malignant Tumours*, where *T* describes the size and extent of the primary *tumour*, *N* describes regional *lymph node involvement*, and *M* describes the presence and site of any distant *metastasis*. The detail of the TNM classification for cervix cancer is described in Table 5.1.

Using the TNM representation, a cervical cancer classified as T3a N1, M1 refers to a cervical cancer that “*involves the lower third of the vagina, no extension to pelvic wall*” with “*regional lymph node metastases*” and “*distant metastases (including peritoneal spread, involvement of supraclavicular or mediastinal lymph nodes, lung, liver, or bone)*” which has an overall stage grouping of Stage IVB.

The TNM system was developed by the International Union Against Cancer (UICC)¹⁷ and the American Joint Committee on Cancer (AJCC)¹⁸ and so most cancer facilities uses the TNM system as their main method for cancer reporting.

(c) Therapeutic modalities

Therapeutic treatment of cancer can involve any of several modalities. Modalities can be combined to create a treatment program that is appropriate for the patient. Also, it is based on patient and tumour characteristics as well as patient preferences. In general, there are six standard treatments for cancer: surgery, chemotherapy, radiotherapy, immunotherapy and biological therapy, hormone therapy, and gene therapy. However, clinical trials may be an option for some as cancer treatment who meets certain study criteria.

¹⁷ The International Union Against Cancer (UICC) (<http://www.uicc.org/>)

¹⁸ The American Joint Committee on Cancer (AJCC) (<http://www.cancerstaging.org/>)

Table 5.1 An Elaborated description of the TNM system¹⁹

TNM	Abbreviation	Description
Primary Tumor (T)	TX	Primary tumor cannot be evaluated
	T0	No evidence of primary tumor
	Tis	Carcinoma in situ (CIS; abnormal cells are present but have not spread to neighboring tissue; although not cancer, CIS may become cancer and is sometimes called preinvasive cancer)
	T1, T2, T3, T4	T1 - The cancer is found only in the cervix. T2 - The cancer has grown beyond the uterus (womb) but not to the pelvic wall or to the lower third of the vagina. T3 - The tumor has spread to the pelvic wall; involves the lower third of the vagina; stops the kidney from working properly; or blocks the tubes (ureters) that connect the kidneys to the bladder. T4- The tumor has invaded the lining (mucosa) of the bladder or rectum and grown beyond the pelvis.
	T3a	It involves the lower third of the vagina, no extension to the pelvic wall
Regional Lymph Nodes (N)	NX	Regional lymph nodes cannot be evaluated
	N0	No regional lymph node involvement
	N1, N2, N3	Involvement of regional lymph nodes (number of lymph nodes and/or extent of spread)
Distant Metastasis (M)	MX	Distant metastasis cannot be evaluated
	M0	No distant metastasis
	M1	Distant metastases (including peritoneal spread, involvement of supraclavicular or mediastinal lymph nodes, lung, liver, or bone)

- Surgery can be used to remove tumours or as much of the cancerous tissue as possible. It is often performed in conjunction with chemotherapy or radiation therapy. However, surgery can be prescribed alone.

- Chemotherapy is a technique of cancer treatment that uses drugs to eliminate cancer cells. Unlike surgery, chemotherapy affects the entire body, not just a specific part.

¹⁹ Cancer Staging (<http://www.cancer.gov/cancertopics/factsheet/detection/staging>)

However, there is an option like surgery. Chemotherapy can be prescribed alone, or it can be supplemented in conjunction with other treatments.

- Radiotherapy is a cancer treatment. Its goal is to shrink tumours or eliminate cancer cells by damaging a cancer cell's DNA. This therapeutic modality can be given alone, along with chemotherapy, and/or with surgery. The main decision to combine radiotherapy with other therapeutic modalities depends on the stage of cancer.

- Immunotherapy and biological therapy is an innovative therapeutic modality that helps to attack and control the side effects of cancer treatment by given with conventional cancer treatments (adjuvant therapy). This is because the well-known therapeutic modalities (radiation, chemotherapy, and surgery) also affect for healthy surrounding tissues and even parts of our bodies remote from the area of the cancer, although they have served well in the battle against of cancer. Then, these therapies help to boost the immune system response and usually specify a target gene. It is noted that biological therapy for cancer includes immunotherapy, targeted therapy, and anti-angiogenesis treatment.

- Hormone therapy can be used in some types of cancer such as breast and prostate cancer. It is designed to alter hormone production in the body, so that cancer cells stop growing or they are killed completely.

- Gene therapy is to replace damaged genes with ones that work to address a root cause of cancer: damage to DNA.

(d) Timing of therapeutic modalities

There are three main terms used to describe the role of therapy relative to other cancer treatments. There are three main terms of timing of therapeutic modalities used: primary, adjuvant, neoadjuvant, and concurrent treatments.

The status of a treatment modality can be described as primary (the most important one in producing a cure, which will be surgery if used, or radiotherapy in the absence of surgery) otherwise its secondary nature will be described in relation to its timing in relation to the primary treatment modality.

Later, the term '*adjuvant*' therapy is relevant to use a systemic therapy (such as radiotherapy, chemotherapy, and biological therapy) after giving the primary therapy such surgery to enhance the chance of cure.

In contrast to adjuvant therapy, '*neoadjuvant*' therapy is given before the main treatment, where it is used to reduce the size of the tumour so as to facilitate more effective surgery.

'*Concurrent*' treatment is relevant to administering cancer treatments at the same time as other therapies (such as concurrent chemotherapy and radiotherapy).

(e) Specified drugs used

Chemotherapy is related to specific drugs used. It means if a professional health considers using this technique as a therapeutic modality, a specific drug is also used with this therapeutic modality. There are many drugs that are used with the Chemotherapy treatment. The majority of chemotherapeutic drugs for therapeutic modalities used can be divided into Alkylating Agents, Anti-metabolites, Plant Alkaloids and Topoisomerase inhibitors, Antibiotics, Miscellaneous, and other Antitumour Agents.

5.4.3 How to implement the CCT-net

This section describes how to develop the CCT-net. The development of the CCT-net is explained below.

Step 1: Specifying the target domain with initial domain knowledge

The development of CCT-net commences by determining the domain and scope of the use of ontology in order to be used as a guide for collecting a set of data source from which the ontology will be developed. The target domain is specified by a set of keywords that are relevant to the cancer domain. The set of keywords can be obtained from answers that satisfy the specific questions. The specific questions are provided below.

- *What is the specific domain that the ontology will cover?*
- *What is the ontology going to be used?*

If the answer of the first question '*What is the specific domain that the ontology will cover?*' is '*cancer*', the keyword '*cancer*' will be expanded to include other relevant keywords such as histology, cancer staging, modality, specific drug used (e.g. Cisplatin, 5-fluorouracil). As demonstrated, some of these keywords can be expanded continuously. Consider the keyword '*modality*' can be expanded to chemotherapy, surgery, radiotherapy. Similarity, the keyword '*histology*' can be expanded to squamous cell carcinoma, adenocarcinoma, adenosquamous carcinoma, and neuroendocrine carcinoma. These keywords will be used to identify, retrieve, and collect the relevant data sources.

Step 2: Collecting data

After acquiring the specific target domain of ontology development, it is necessary to collect all relevant data sources.

(1) Existing data, such cancer dictionary and a list of drugs used can efficiently be reused to create a new ontology for a new objective (referred to <http://www.cancer.gov/dictionary>).

(2) The pre-existing ontologies such National Cancer Institute's Thesaurus and Ontology (referred to <http://www.mindswap.org/2003/CancerOntology/>).

(3) The biomedical literatures relevant to cervix cancer are retrieved from MEDLINE. In this context, we utilize the keywords that are obtained from step (1) to search and retrieve the relevant abstracts from MEDLINE through the PubMed search engine.

Step 3: Extracting important concepts (as classes) and setting concept hierarchy

This step identifies important terms as ontological concepts (or classes), and organizing the concept hierarchy in ontology.

(1) Ontological concept provision

In this case, some of specific terms determined by the domain experts are used as the ontological concepts in the CCT-net ontology. The ontological concepts are histology, disease stage, therapeutic modalities, timing of therapeutic modalities, and specified drugs used. These concepts will be expanded to their associated terms.

(2) Ontological conceptual enumeration

After obtaining the ontological concepts from (1), this step is to determine term variations and related terms of each ontological concept. There are three solutions to search for the associated terms.

Firstly, some of associated terms can be searched automatically based on rules and a simple statistic theory is applied to estimate probability of occurrence of each extracted term. For example, if we need to find alternative names of the term '*radiotherapy*'. Suppose the rule, '*if term X followed by (Y) then Y is pulled*', is determined. Let *X* be a term and *Y* be an alternative name of *X*. Consider following sentence.

'Abbreviated course of radiotherapy (RT) for breast cancer'

By using the rule, we will obtain '*RT*' as an alternative name of '*radiotherapy*'. It is noted that all of the rules will be determined manually.

Secondly, we can utilize the existing data (e.g. Cancer dictionary, a list of drugs used, and MeSH database) to search for the associated terms of each ontological concept. Some examples are shown as Table 5.2.

Thirdly, some of associated terms should be defined by the domain expert, where these terms are provided by health professionals. An example can be illustrated. The terms *CTRT* or *CRT* indicate to *chemoradiation*. It is a treatment that combines chemotherapy with radiation therapy. Sometimes, health professionals use the term '*CTRT*' or *CRT* alone

without the full name ‘*chemoradiation*’. For example, consider the following sentence ‘*Both were provided over 3 weeks before and 12 weeks after CRT*’. This sentence explains that the therapeutic modality in this case is *chemoradiation*. Therefore, general users may not be able to understand for this term because the term ‘*CRT*’ is not described in the cancer dictionary (<http://www.cancer.gov/dictionary>).

Table 5.2 Examples of utilizing the existing data to search for the associated terms of each ontological concept

Ontological concept	Subclasses	Associated terms (Alternative names)	The source of the existing data
Therapeutic modalities	Chemotherapy	Drug therapy Chemoradiotherapy	MeSH
specified drugs used	Cisplatin	Chemotherapy	A list of drugs used + MeSH
	5-Fluorouracil	Chemotherapy	a list of drugs used + MeSH

(3) Setting the relationships between concepts and the concept hierarchy

After obtaining a set of concepts, the concept hierarchy is set up by considering the relationships between concepts. The taxonomic relations such ‘*is-a*’ or ‘*kind-of*’ can be used to indicate the relationship between concepts, where a class *X* is a subclass of *Y* if every instance of *Y* is also an instance of *X*.

For example, the term ‘*cervix cancer*’ is defined as a concept in the CCT-net. However, by considering it in conjunction with the cancer dictionary, it can be summarized that ‘*cervix cancer*’ is a subclass of ‘*cancer*’ and ‘*cancer*’ is a subclass of ‘*disease*’. Therefore, transitivity of the subclass relationship means that the class ‘*cervix cancer*’ is a subclass of ‘*disease*’ as well.

< Cervix cancer > is-a < cancer >
< Cancer > is-a < disease >
< Cervix cancer > is-a < disease >

These can be represented in predicate logic as follows.

$\forall(x): cervix_cancer(x) \Rightarrow cancer(x)$
 $\forall(x): cancer(x) \Rightarrow disease(x)$
 $\forall(x): cervix_cancer(x) \Rightarrow disease(x)$

Meanwhile, ‘stage’ is a description of the extent that cancer has spread. ‘Squamous cell carcinoma’ is a ‘cancer histology’, while ‘surgery’ and ‘chemotherapy’ are the kinds of ‘cancer treatment modality’. These are important aspects of information about cancer that are necessary for clinical decision-making. Some of the relationships can be represented in predicate logic.

$$\forall(x): surgery(x) \vee radiotherapy(x) \vee chemotherapy(x) \Rightarrow modality(x)$$

$$\forall(x): squamous_cell_carcinoma(x) \Rightarrow histology(x)$$

$$\forall(x): cancer\ staging(x) \Rightarrow cancer(x)$$

$$\forall(x): histology(x) \Rightarrow cancer(x)$$

$$\forall(x): cancer\ treatment\ modality(x) \Rightarrow cancer(x)$$

As the results, it can be concluded that ‘cancer histology’, ‘cancer staging’, and ‘cancer treatment modality’ are subclasses of ‘cancer’.

In addition, consider the original word ‘Chemotherapy’. It can be transformed into ‘CT/RT’, ‘CTRT’, ‘CTN/RTN’ and ‘Chemoradiation’. Therefore,

$$\langle CT/RT, CTRT, Chemoradiation, CTN/RTN \rangle is-a \langle chemotherapy \rangle$$

These can be described as a set of axioms that define relationships between their concepts. They can be represented in predicate logic as follows.

$$\forall(x): CT/RT(x) \Rightarrow Chemotherapy(x)$$

$$\forall(x): CTRT(x) \Rightarrow Chemotherapy(x)$$

$$\forall(x): CTN/RTN(x) \Rightarrow Chemotherapy(x)$$

$$\forall(x): Chemoradiation(x) \Rightarrow Chemotherapy(x)$$

In this context, the concept hierarchy is set up based on the top-down model through consideration of the relationships between concepts. An example of the concept hierarchy can be represented as Figure 5.7.

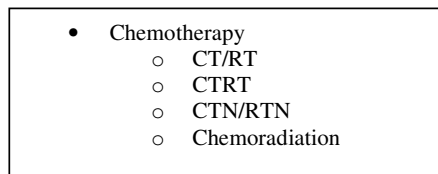


Figure 5.7 An example of defining concepts and the class hierarchy of a cancer-term.

Furthermore, consider the association between a main concept and a sub-concept of another main concept. It is well-known in the domain expert that if a drug is used during a

treatment modality, the therapeutic modality is ‘*chemotherapy*’. Therefore, if a drug is used, it instantly indicates to ‘*chemotherapy*’ used as the treatment modality, although it is not found the cancer-term ‘*chemotherapy*’ in text. Therefore,

< All specific drugs used > relate-to < chemotherapy >

Step 4: Finding its properties

After identifying the important concepts and the concept hierarchy, each concept needs to add some significant information such as the association among concepts, the varied terms set as synonyms.

- *Histology*

A part of the concept of cancer histology can be presented as Figure 5.8. The example document in Figure 5.8 consists of a root element, called ‘*histology*’. It contains a required attribute called ‘*Squamous cell carcinoma*’. The ‘*histology*’ element contains two different child elements: ‘*alternative-name*’ and ‘*part-of-body-found*’. The element ‘*alternative-name*’ represents all of term variations. The element ‘*part-of-body-found*’ describes the position in a body that can be found each histological type of cancer. These data can be collected from the cancer dictionary.

It is noted that the root element ‘*histology*’ can be squamous cell carcinoma, adenocarcinoma, adenosquamous carcinoma, and neuroendocrine carcinoma.

- *Disease stage*

A part of the concept of disease stage can be presented as Figure 5.9. The root element of this concept is ‘*disease-stage*’. It contains a required attribute called ‘*Stage I*’. The ‘*disease-stage*’ element contains three different child elements: ‘*stage-synonyms*’, ‘*sub-stage*’ and ‘*therapeutic-modality*’.

```
<Histology> Squamous cell carcinoma
  <Alternative-name> SCC, Squamous </alternative-name>
  <Part-of-body-found>
    Skin, lips, mouth, lungs, vagina, and cervix
  </Part-of-body-found>
</histology>
```

Figure 5.8 The concept of cancer histology

The ‘*disease-stage*’ presents a measure of how much the cancer has grown and spread. Most tumours can be described as stage I, stage II, stage III, or stage IV. The element ‘*stage-synonyms*’ shows alternative symbols of cancer staging. The element ‘*sub-stage*’ presents cancer sub-staging. The element ‘*therapeutic-modality*’ shows what therapeutic modality can be used. These data can be collected from the cancer dictionary, the domain expert, and automatic extraction by the rules. Our rules are developed based on *n-gram* technique. An example rule used to extract ‘*stage-synonyms*’ can be shown as, *Stages + x*. The meaning of this rule is that, the keyword ‘*stage*’ always followed by *x*, where *x* represents the *disease-stage*’.

```

< Disease-stage> stage I
  <Stage-synonyms> 1, I </stage-synonyms>
  <sub-stage> A, a, <stage-example>1A, IA, ia, 1a</stage_example></sub_stage>
  <sub-stage> B, b, <stage-example>1B, IB, ib, 1b</stage_example></sub_stage>
  <therapeutic-modality> surgery</therapeutic-modality>
  <therapeutic-modality> chemotherapy </therapeutic-modality>
  <therapeutic-modality> radiotherapy </therapeutic-modality>
</disease-stage>

```

Figure 5.9 The concept of disease stage

- *Therapeutic modalities*

A part of the concept of therapeutic modalities can be presented as Figure 5.10. The example document consists of a root element, ‘*therapeutic-modalities*’, that contains a required attribute called ‘*chemotherapy*’. The ‘*therapeutic-modalities*’ element contains four different child elements: ‘*synonyms*’, ‘*kind-of-modality*’, ‘*disease-stage*’, and ‘*co-modalities*’.

```

< Therapeutic-modalities> chemotherapy
  <Synonyms> CTRT, CT/RT, CTN/RTN, Chemoradition, ... </synonyms>
  <Kind-of-modality> specific drugs used </kind-of-modality>
  <disease-stage> I, II, III, IV </disease-stage>
  <co-modalities> surgery, radiotherapy </ co-modalities >
</therapeutic-modalities>

```

Figure 5.10 The concept of therapeutic modalities

The child element ‘*synonyms*’ presents alternative names of ‘*chemotherapy*’, while the child element ‘*kind-of-modality*’ describes that *chemotherapy* always uses with specific

drugs. *Chemotherapy* can be used for every ‘*disease-stage*’ and it can be used as *co-modalities* with surgery and radiotherapy. These data can be collected from the cancer dictionary, the domain expert, and automatic extraction by the rules. An example rule used to extract ‘*synonyms*’ can be shown following, $x + (y)$. The meaning of this rule is that, x is the keyword (such as chemotherapy and radiotherapy). If ‘ x ’ is followed by ‘ y ’, ‘ y ’ may be a *synonym* of ‘ x ’.

- *Timing of therapeutic modalities*

A part of the concept of timing of therapeutic modalities can be presented as Figure 5.11. The example document consists of a root element, called ‘*timing of therapeutic modalities*’, that contains a required attribute called ‘*adjuvant*’. The ‘*timing of therapeutic modalities*’ element contains only one child element, called ‘*description-of-usage*’. It contains some phrases in MEDLINE abstracts that indicate to ‘*timing of therapeutic modalities*’.

```

<Timing of therapeutic modalities> adjuvant
  <description-of-usage> after surgery, ... </ description-of-usage >
</Timing of therapeutic modalities> neoadjuvant
  < description-of-usage > before surgery, ... </ description-of-usage >
</Timing of therapeutic modalities>

```

Figure 5.11 The concept of timing of therapeutic modalities

These data can be collected from the cancer dictionary, the domain expert, and automatic extraction by the rules. An example rule is ‘*a phrase containing $x + after surgery$* ’, where x can be ‘*chemotherapy*’ or ‘*radiotherapy*’. It means that, if a phrase containing x is found and it is followed by ‘*after surgery*’, the timing of therapeutic modality is ‘*adjuvant*’. This rule also utilizes the definition of ‘*adjuvant*’, where its definition is a treatment that is given in addition to the primary. An example of adjuvant therapy is the additional treatment usually *given after surgery*.

- *Specified drugs used*

A part of the concept of specified drugs used can be presented as Figure 5.12. The example document consists of a root element, ‘*specified-drugs-used*’, that contains a required attribute called ‘*drugs*’. The ‘*specified-drugs-used*’ element contains two different child elements: ‘*relevant-modality*’ and ‘*drug-class*’. The child element ‘*drug-class*’ consists

of an element, called ‘*specific-drug*’. This sub-element contains the element ‘*alternative-drug-name*’.

These data can be collected from a list of cancer drugs used for *drug treatment* (also known as *chemotherapy*).

```

<specified-drugs-used> drugs
  <relevant-modality> chemotherapy </relevant-modality>
  <drug-class> Alkylating Agents
    <specific-drug> crispatin
      <alternative-drug-name> Abiplatin, Biocisplatinum, .. </alternative-drug-name>
    </specific-drug>
    ...
  </drug-class>
  ...
</specified-drugs-used>

```

Figure 5.12 The concept of specified drugs used

Step 5: Ontology Construction

Finally, the CCT-net is developed based on five main concept mappings (histology, disease stage, therapeutic modalities, timing of therapeutic modalities, and specified drugs used). In this context, the XML Schema Definition Language (XSD) that is described in Chapter 2 is used as the ontology language.

5.4.4 Size of the CCT-net ontology

In the CCT-net ontology, it consists of five main concepts: histology, therapeutic modality, disease stage, timing of therapeutic modality, and specific drugs used. Each concept contains many sub-concepts that are shown as Table 5.3.

Table 5.3 Summary of the CCT-net ontology

List	Main Concept Name	Number of Sub-concepts	Examples
1	Histology	7	Squamous cell carcinoma Adenocarcinoma Neuroendocrine carcinoma
2	Therapeutic modality	7	Surgery Chemotherapy Radiotherapy
3	Disease stage	5	0, I, II

Table 5.3 (cont')

List	Main Concept Name	Number of Sub-concepts	Examples
4	Timing of therapeutic modality	3	Adjuvant Concurrent
5	Specific drugs used	366	Cispatin 5-FU

5.4.5 The example of the use of the CCT-net ontology

The main benefit of the CCT-net ontology is to support for the understanding of cancer terms, and this ontology can link a cancer term to other terms.

An example can be illustrated in the domain of text analysis. Consider a following collection that contains three phrases as below.

S_1 : Chemotherapy of squamous cell carcinoma of the skin.

S_2 : The effect of chemoradiation on thymidine phosphorylase and dihydropyrimidinase expression in cancer of the uterine cervix.

S_3 : Levels of 5-FU metabolic or related enzymes.

If this collection does not utilize the CCT-net for analysing the variation of cancer-terms, a system cannot understand that the cancer-term '*Chemotherapy*' and '*chemoradiation*' are similar. Furthermore, the system cannot know that the third phrase is also related to '*Chemotherapy*', where '*5-FU*' is a specific drug used with chemotherapy. By using the CCT-net, the system should recognize and interpret that '*chemoradiation*', and '*5-FU*' are related to '*Chemotherapy*'.

5.4.6 The experiments of the CCT-net

We show the approach of using the CCT-net through text retrieval, where the objective of this stage is to compare the different search methods between the method without the CCT-net and the method with the CCT-net. It is noted that the method of relevant literature retrieval is done through text classification.

This method commences with gathering the cervix cancer abstracts by keywords search on PubMed system, and then these abstracts are used to build a text classifier in order to use for retrieving clinically relevant cervix cancer abstracts.

The basic concept of text classification is formalized as the task of approximating the unknown target function $\Phi: D \times C \rightarrow \{T, F\}$ by means of a function $\hat{\Phi}: D \times C \rightarrow \{T, F\}$ - called the classifier - where $C = \{c_1, c_2, \dots, c_{|C|}\}$ is a predefined set of categories, and D is a set of documents. The documents $D = \{d_1, \dots, d_{|D|}\}$ to be classified are described by a vector of

words $\omega = \{w_1, w_2, \dots, w_i\}$. If $\Phi(d_i, c_j) = T$, then d_i is called a positive member of c_j , while if $\Phi(d_i, c_j) = F$, it is called a negative member of c_j . In this context, text classifiers were implemented by two algorithms: Support Vector Machines (SVM) and Naïve Bayes algorithms.

Before building a text classifier, a collection of cervix cancer abstracts is represented in the structured ‘*bag of words* (BOW)’. We obtain $w = (w_1, w_2, \dots, w_k, \dots, w_v)$, where w_{ik} is the number of the unique words within the collection. In the BOW, a cervix cancer abstract d_i is composed of a sequence of words, with $d_i = (w_{i1}, w_{i2}, \dots, w_{ik}, \dots, w_{iv})$, where w_{ik} is the frequency of the k -th word in an oncology document d_i . Significantly, each word is weighted by *tf-idf*, which is used for providing a pre-defined set of features for exchanging information. The *tf-idf* weight of a term is the product of its *tf* weight and *idf* weight. The term frequency *tf* of term t in document d is defined as the number of times that t occurs in d . For *idf* weighting, let *df* be the document frequency of t , where t is the number of documents that contain t . It can define the *idf* of t as $idf = 1 + \log(N/df)$. In general, word segmentation is the first and obligatory task in natural language processing because the word is a basic unit in linguistics. In case of analysing the specific words in the cancer area, this task relies on the CCT-net. A list of relevant keywords can be generated by using this ontology. The CCT-net is used as a dictionary. Meanwhile, the general words relies on the VL-ontology. By using the CCT-net, the process of analysis should understand that the words ‘*chemotherapy*’, ‘*chemoradiotherapy*’, and ‘*CT*’ are the same meaning, although they are different for representation. Finally, the SVM and the Naïve Bayes text classifiers are evaluated by the information retrieval standard. Common performance measure for system evaluation can be *F-measure* (F), where the *F - measure* is the weighted harmonic mean of precision and recall. The results of the evaluation can be shown as Table 5.4.

Table 5.4 The experimental results of the SVM and the Naïve Bayes text classifiers

Text classifier without the CCT-net		Text classifier with the CCT-net	
Algorithms	F-measure (%)	Algorithms	F-measure (%)
SVM	78.00	SVM	95.00
Naïve Bayes	72.50	Naïve Bayes	91.50

Consider Table 5.4. After testing by *F-measure*, It can be seen that text classifiers based SVM and Naïve Bayes are more effective for retrieving the relevant cervix cancer abstracts relating to clinical trials from PubMed. This is because the average accuracies can

be increased. This demonstrates that the CCT-net is more effective for selecting and retrieving the relevant cervical cancer abstracts from PubMed.

5.4.7 The expected effectiveness of the CCT-net

The CCT-net ontology offers three main efficiencies. It helps to identify the particular domain that users satisfy for the needs, where users need to retrieve the relevant Literature.

(1) It also helps to handle a problem of terms variation in the domain specific. This can lead to correctly analyse and interpret some cancer technical terms which are presented in different styles but they should be actually understood in the same meaning. For example the combined simultaneous use of radiotherapy and chemotherapy can be described as ‘*chemotherapy and radiotherapy*’, ‘*CT/RT*’, ‘*chemoradiotherapy*’, ‘*chemoradiation*’, or ‘*RT/CT*’. In the domain of text processing, the problem of ambiguity can lead the performance of text retrieval being poor.

(2) The CCT-net may help to understand an association between concepts. For example, the stage of cancer can refer to the process of treatment. If patients are in cancer stage 0, the patients can be treated by radical surgery alone. For another example, if *cisplatin* which is a cancer drug is used, it means that the clinicians use chemotherapy as the modality treatment.

5.5 Extracting Clinical Knowledge from PubMed Abstracts

In this section, we shall describe an instance of the general On-KDT methodology described above which is effective in supporting a largely automated process of extracting clinical knowledge from MEDLINE abstracts.

5.5.1 The Preparation of PubMed abstracts

The data preparation step consists of two main tasks: *data pre-processing* and *identifying more specific individuals*. These can be elaborated as follows.

(a) Data Pre-processing

This task is performed on MEDLINE abstracts. It will be transformed into a format that is appropriate for the process of identifying more specific individuals.

(i) Data Transformation

Each abstract is tokenized and transformed it into the vector space model, $\vec{d} = \{w_{i,1}, w_{i,2}, \dots, w_{i,t}\}$, and it is now called ‘*bag of words (BOW)*’. This step utilizes both

ontologies (the VL-ontology and the CCT-net) as the semantic tools to understand similar words (having the same stem (or synonyms) and the same grammar class). These words (terms) will be considered as the same concept. An example of MEDLINE abstract can be shown as Figure 5.13.

BACKGROUND: To evaluate the efficacy of low or high-dose immunomodulator, Z-100, in combination with radiotherapy for cervical cancer cervix. METHODS: Between 1995 and 1999, 221 patients with stage IIIB squamous cell carcinoma of the cervix were randomly assigned to treatment with Z-100 either at 0.2 microg or 40 microg in a double-blind manner in combination with radiotherapy. RESULTS: The 5-year survival of patients with high-dose and low-dose Z-100 was 41.5% (95% CI: 31.7-51.3%) and 58.2% (95% CI: 48.7-67.7%), respectively, showing a 30% reduction in the death rate (hazard ratio: 0.670 [95% CI: 0.458-0.980], P = 0.039). Survival of high-dose group was equivalent to the 4-year survival of the radiotherapy plus hydroxyurea arm (49.7%) of GOG120 study, and that of low-dose group was similar to the survival of the cisplatin-based chemoradiation arm. The progression-free survival was also significantly improved in favor of low-dose group (hazard ratio: 0.667 [95% CI: 0.447-0.997], P = 0.048). The survival of low-dose group was similar to the survival of the cisplatin-based chemoradiation arms of the GOG120 study. CONCLUSIONS: Unexpectedly, the survival of patients with advanced cervical cancer cervix treated by lower dose of Z-100 in combination with radiotherapy was significantly better than those treated with higher dose Z-100, which was equivalent to the survival with radiotherapy alone. The hypothesis that lower dose of Z-100 enhances the efficacy of radiation therapy is now being tested by placebo-controlled randomized trial.

Figure 5.13 An example of MEDLINE abstract

(ii) Data Cleaning

After transforming MEDLINE abstracts to BOW, we obtain $w = (w_1, w_2, \dots, w_k, \dots, w_v)$, where v is the number of unique words within the collection. In the BOW, MEDLINE abstract d_i is composed of a sequence of words, with $d_i = (w_{i1}, w_{i2}, \dots, w_{ik}, \dots, w_{iv})$, where w_{ik} is the frequency of the k -th word in an MEDLINE abstract d_i .

After parsing MEDLINE abstracts to extract unique words, stop-words are removed. Stop-words are words which are considered as non-descriptive within a BOW approach. They typically comprise prepositions, articles, etc. These words usually have very high frequency in the total corpus, and are removed prior to classification. Following common practice, we removed stop-words by using a standard list with 571 stop-words²⁰.

(iii) Weighting terms

Each word is weighted by *tf-idf*. It is used for providing a pre-defined set of features for exchanging information. Each unique word w_i corresponds to a feature with $tf(w_i, d_i)$, the number of times word w_i occurs in the article d_i , as its value. Refining the requirement representation, it has been shown that scaling the dimensions of the feature vector with their *inverse article frequency* $idf(w_i)$ leads to improved performance. $idf(w_i)$ can be calculated

²⁰ <http://www.aifb.uni-karlsruhe.de/WBS/aho/clustering>

from the article frequency $df(w_i)$, which is the number of articles in which word w_i occurs. It is described as follows.

$$idf(w_i) = 1 + \log(|D| / df(w_i))$$

An example of calculating term weight can be presented as follows. Consider the two sentences as follows.

D_1 : “To evaluate the efficacy of low or high-dose immunomodulator, Z-100, in combination with **radiotherapy** for cervical cancer”

D_2 : “To compare weekly and three-weekly cisplatin as an adjunct to **radiation therapy** in high-risk early-stage cervical cancer after surgery with regard to treatment compliance.”

Suppose there are five selected feature keywords. Suppose there are five features: immunomodulator, cervical, cancer, surgery, and radiotherapy (also known as Radiation therapy). By using the CCT-net, it can be analysed that the technical terms ‘radiotherapy’ and ‘radiation therapy’ are presented in different writing styles but they share the same meaning. Therefore, they will be analysed as a same feature. Finally, tf and idf of these terms can be calculated and shown as Table 5.5. Later, the $tf-idf$ can be calculated and shown as Table 5.6.

Table 5.5 Example of calculating tf and idf

Documents	immunomodulator		cervical		cancer		surgery		Radiotherapy Radiation therapy	
	tf	idf	tf	idf	tf	idf	tf	idf	tf	idf
D_1	1	1.301	1	1	1	1	0	1.301	1	1
D_2	0		1		1		1			1

1 = presence, 0 = absence

Table 5.6 Example of calculating $tf-idf$

Documents	immunomodulator	cervical	cancer	surgery	Radiotherapy Radiation therapy
D_1	1.301	1	1	0	1
D_2	0	1	1	1.301	1

(b) Identifying more specific individuals

As we said at the Section 5.5 that, we may obtain the most relevant knowledge from MEDLINE abstracts by using seven main questions as follows.

- What is the histology of cervix cancer?
- What is the stage of cervix cancer?

- *How many patients are in a study?*
- *What are arm-names used?*
- *What treatment modalities are used?*
- *What is timing of each treatment modality used?*
- *What are drugs used for treatment?*

To identify more specific individual data, we also utilize these questions to construct text classifier models that are used as text filters. Text filter is required, where it can be used to filter irrelevant information from a MEDLINE abstract in order to improve efficiency of text analysis and obtain the quality of knowledge extraction in the next step. In this context, it will select the most relevant passage in an abstract relating to the question. Therefore, there are seven text filter models.

For example, consider the first question, ‘*What is the histology of cervix cancer?*’. In this question, the term ‘*histology*’ is the keyword. By using the CCT-net, this keyword can be expanded to squamous cell carcinoma, adenocarcinoma, adenosquamous carcinoma, and neuroendocrine carcinoma. We use these keywords to construct a text filter model. An example of using the text filter model can be presented as Figure 5.14.

To construct text filter models, we provide a content-based text classification (CBCT) by using the Support Vector Machine (SVM) algorithm. The SVM has strategy to find the best hyperplane on input space called the structural minimization principle of statistical learning theory. In fact, the basic concept of the SVM is to build a function that takes the value +1 in a “small” region capturing most of the data points, and -1 elsewhere. However, our approach applies a method of the one-class SVM which is to only consider the positive points.

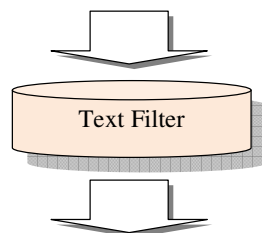
In this context, let x_1, x_2, \dots, x_l be training document set belonging to one class X , where X is a compact subset of R^N . Let $\Phi : X \rightarrow h$ be a kernel map which transforms the training set to another space. Indeed, to separate the data set from the origin, one needs to solve the following quadratic programming problem:

$$\begin{aligned}
 \text{problem: } & \min \frac{1}{2} \|w\|^2 + \frac{1}{vl} \sum_{i=1}^l \xi_i - \rho \\
 \text{subject to: } & (w \cdot \Phi(x_i)) \geq \rho - \xi_i \quad i = 1, 2, \dots, l \quad \xi_i \geq 0
 \end{aligned}$$

If w and ρ solve this problem, then the decision function will be positive for most examples x_i contained in the training document set.

$$f(x) = \text{sign} ((w \cdot \Phi(x)) - \rho)$$

BACKGROUND: To evaluate the efficacy of low or high-dose immunomodulator, Z-100, in combination with radiotherapy for cervical cancer cervix. METHODS: Between 1995 and 1999, 221 patients with stage IIIB squamous cell carcinoma of the cervix were randomly assigned to treatment with Z-100 either at 0.2 microg or 40 microg in a double-blind manner in combination with radiotherapy. RESULTS: The 5-year survival of patients with high-dose and low-dose Z-100 was 41.5% (95% CI: 31.7-51.3%) and 58.2% (95% CI: 48.7-67.7%), respectively, showing a 30% reduction in the death rate (hazard ratio: 0.670 [95% CI: 0.458-0.980], P = 0.039). Survival of high-dose group was equivalent to the 4-year survival of the radiotherapy plus hydroxyurea arm (49.7%) of GOG120 study, and that of low-dose group was similar to the survival of the cisplatin-based chemoradiation arm. The progression-free survival was also significantly improved in favor of low-dose group (hazard ratio: 0.667 [95% CI: 0.447-0.997], P = 0.048). The survival of low-dose group was similar to the survival of the cisplatin-based chemoradiation arms of the GOG120 study. CONCLUSIONS: Unexpectedly, the survival of patients with advanced cervical cancer cervix treated by lower dose of Z-100 in combination with radiotherapy was significantly better than those treated with higher dose Z-100, which was equivalent to the survival with radiotherapy alone. The hypothesis that lower dose of Z-100 enhances the efficacy of radiation therapy is now being tested by placebo-controlled randomized trial.



METHODS: Between 1995 and 1999, 221 patients with stage IIIB squamous cell carcinoma of the cervix were randomly assigned to treatment with Z-100 either at 0.2 microg or 40 microg in a double-blind manner in combination with radiotherapy.

Figure 5.14 An example of MEDLINE abstract

In this context, we used the default parameter σ and d of the LIBSVM²¹ tool for support vector classification. Finally, the one-class SVM algorithm gives the text classifier for selecting the relevant cervix cancer documents having treatment information.

We experiment text filter models with 200 MEDLINE abstract relevant to cervix cancer in clinical trials. The results of the experiments for text filters are evaluated by using the information retrieval standard. Common performance measures for system evaluation are *precision (P)*, *recall (R)*, and *F-measure (F)*. The results of the evaluation are represented as Table 5.7.

Consider Table 5.7. Although the results of the one-class SVM text classifiers are still good in terms of accuracy, they show a failure rate, especially the models used for the fourth question and the fifth question. The fourth question indicates to the ‘*arm-name*’ in clinical trials, while the fifth question indicates to treatment modalities (e.g. Chemotherapy, surgery, radiotherapy). However, the arm - name may be closely related to treatment modalities. Furthermore, some of MEDLINE abstracts, the representation in the abstracts are

²¹ <http://www.csie.ntu.edu.tw/~cjlin/libsvm>

very complex. Therefore, this may lead to have confusion. For example, consider the following abstract.

OBJECTIVES: The use of preoperative neoadjuvant chemotherapy (NAC) in locally advanced cervical cancer cervix (LACC) was hindered by the disadvantages of a delay of curative treatment for nonresponders and the development of radioresistant cells. However, these disadvantages may be overcome by a 'quick' high-dose scheme administered in a short period before surgery. Our purpose is to assess the efficacy of NAC with short cycle-length, high-dose agents for LACC.

Table 5.7 The experimental results of text filter models

Models	Recall (%)	Precision (%)	F-measure (%)
Text filter for the first question	100	100	100
Text filter for the second question	100	100	100
Text filter for the third question	100	95	97.44
Text filter for the fourth question	83	75	72.95
Text filter for the fifth question	85	82	83.47
Text filter for the sixth question	100	100	100
Text filter for the seventh question	100	100	100
Average	94.28	92.57	93.41

If this abstract is analysed manually, there is only one arm-name in this clinical trials: neoadjuvant chemotherapy (NAC) followed by surgery. However, chemotherapy and surgery are treatment modalities. Therefore, this can lead to mis-understanding and the miss-filtering. Finally, it may lead to the accuracy of the one-class SVM text filter being poor.

5.5.2 The Closed-domain knowledge extraction approached for extracting clinical trial knowledge from PubMed abstracts

In this step, it shows how to extract clinical knowledge from MEDLINE abstracts. We have said in Chapter 3 that, we utilize the concept of closed-domain question answering to find the real need in the abstracts. In this chapter, the method of closed-domain knowledge extraction can be shown in Figure 5.15. It consists of three tasks: *question analysis and knowledge pattern development, passage retrieval, and knowledge extraction.*

Given a question, we commence with the Question Analyzer. Focusing question is used to provide the list of keywords from the question. Afterwards, the keywords in

question are passed to the Relevant Sentences Retrieval, which select the relevant sentence that may contain the answer. Finally, these selected sentences are analysed by the Answer Extractor in order to extract the final answer. The detail of each component can be described as follows.

Task 1: Question analysis and knowledge pattern development

This component is to analyse the question, and to create some representation of information requested. This is crucial to determine the question type, the expected answer type, and the question focus. It is noted that there are only two question types used in this work: *what-is/are* and *how-many*.

(1) Specifying questions

It is to provide the main questions that may help identify the expected results. This can be performed manually. There are seven questions that are necessary for this task.

- *What is the histology of cervix cancer?*
- *What is the stage of cervical cancer?*
- *How many patients are in a study?*
- *What are arm-names used?*
- *What treatment modalities are used?*
- *What is timing of each treatment modality used?*
- *What are drugs used for treatment?*

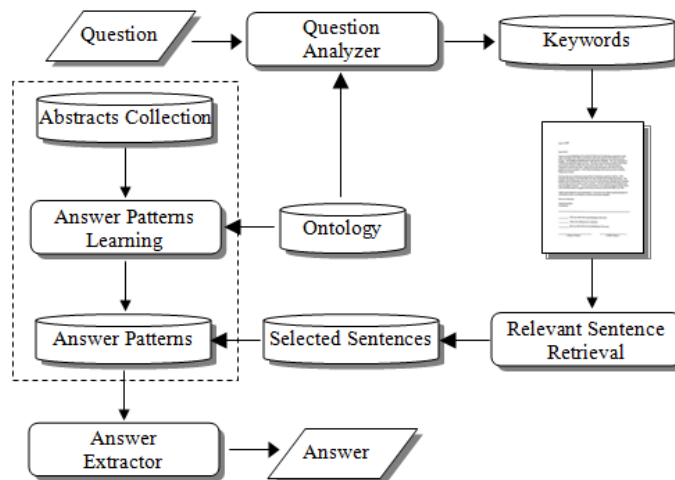


Figure 5.15 The method of closed-domain knowledge extraction for finding the clinical trial knowledge from PubMed abstracts

(2) Questions analysis and identifying type of answer (or knowledge)

This step commences with finding the focus of question, where the focus of question can indicate what information is being asked for in the question. In this context, identifying the focus can be done by *keywords matching* based on the CCT-net. For an example, the question ‘*What is the stage of cervical cancer?*’ has the focus ‘*stage*’.

Table 5.8 Examples of question classification, corresponding answer type classification, and focus of question

Question classes	Sub-question classes	Answer Types	Question Focus
What	What-is	Issues or cells	histology
	What-are	modalities used/modality use	modality
How	How-many	patients are in a study/ number	number of patient

Simply speaking, the question focus is used to determine the list of keywords based on expanding the keywords. These keywords are used as queries. One illustrated, the question “*What is the stage of cervix cancer?*” has the focus “*stage*”. The stages of cervix cancer are numbered from 0 to 4. Each stage is sometimes divided into A and B. Therefore, the keyword “*stage*” can be expanded to ‘1’, ‘2’, ‘3’, and ‘4’. Also, each expanded term and format can be expanded terms to other terms (its synonyms). For an example, the format ‘2’ is related to the formats ‘II’ and ‘ii’.

(3) Development of answer pattern (used as knowledge pattern)

To extract the answer (as knowledge) that is relevant to the specific question, it is necessary to provide patterns used to search and extract clinical knowledge from MEDLINE abstracts, called *knowledge patterns*. These patterns are developed based on the concept of *pattern-match rules or pattern matching* (Califf and Mooney, 1997). Pattern matching is the technique of searching a string containing text/token data for some set of characters based on a specific search pattern (Califf and Mooney, 1997).

To construct patterns in this way, we apply two techniques to develop the knowledge pattern: (1) *keyword matching* and (2) *N-gram model* (Cavnar, 1994). The keyword matching is a process of finding all occurrences of keywords from a given set as a substring in a given string. A *n-gram* model is a contiguous sequence of *n* items from a given sequence of text (Cavnar, 1994). This model is simple but it is powerful used in computational linguistics and statistical language modeling. A *n-gram* can be any combination of letters, phonemes,

syllables, words or base pairs according to the application. A n -gram of size 1 is called a ‘*unigram*’; size 2 is a ‘*bigram*’, size 3 is a ‘*trigram*’. Larger sizes refer to the value of n (e.g. four-gram, five-gram).

- *Pattern-match rules based on keywords*

The process of knowledge pattern development based on based on keyword matching can be concluded as the following algorithm.

1. Extracting the keyword in each specific question
2. Expand the keyword to other words that are relevant by using the CCT-net and the domain expert
3. Group these keywords into the same class re

An example of keyword rules can be shown as Table 5.9.

Table 5.9 Examples of the keyword used to match and find knowledge from MEDLINE abstracts

Question classes	Keyword	Keyword expansion	Related terms
What is the histology of cervix cancer?	Histology	Squamous cell carcinoma	SCC, Squamous
		adenocarcinoma	Glandular tissue, malignant tumour of a gland
		adenosquamous carcinoma	-
		neuroendcrine carcinoma	-

- *Pattern-match rules based on N-gram model*

The process of knowledge pattern development based on based on N -gram model can be concluded as the following algorithm.

1. Use text filter relating to the specific question to retrieve the most relevant passage in a MEDLINE abstract
2. Use the keywords as the mark point (or clue word).
3. Finding the answer relating to the question
4. Generating the *knowledge pattern* based on N -gram model.

To have more understanding, we describe through another example. Consider the question ‘*What is the stage of cervix cancer?*’. The question focus is ‘*stage*’ which is used as the key feature to find a knowledge pattern. In general, the staging of cancer is numbered from 0 to 4 (or I - IV). Each stage is sometimes divided into A and B. An example of

generating the knowledge pattern based on N -gram model can be described following. Suppose there are three MEDLINE abstracts. The most relevant passage of each abstract can be present as follows.

P_1 : *Between 1995 and 1999, 221 patients with **stage IIIB** squamous cell carcinoma of the cervix were randomly assigned to treatment with Z-100 either at 0.2 microg or 40 microg in a double-blind manner in combination with radiotherapy.*

P_2 : *From June 1st, 2003 to February 29th, 2004, the authors performed a randomized trial of radiotherapy in combination with two concurrent chemotherapy regimens - weekly or three-weekly cisplatin--in patients with high-risk cervical cancer cervix FIGO **stage I-IIA** after surgery. Women with primary invasive squamous-cell carcinoma, adenocarcinoma, or adenosquamous carcinoma of the cervix were enrolled.*

P_3 : *Consecutive patients affected by **stage IB-IIB** cervical carcinoma scheduled for radical surgery entered the study.*

Consider three passages above. It can be found ‘stage’ always follows with a number (e.g. stage IIIB) or range of number (e.g. stage IB-IIB). By the N -gram concept, the pattern used for extracting cancer staging can be ‘stage + $\langle x \rangle$ ’ or ‘stage + $\langle x-y \rangle$ ’, where x and y are the number from 1 to 4 having more progression. The first pattern ‘stage + $\langle x \rangle$ ’ is unigram, while the second pattern ‘stage + $\langle x-y \rangle$ ’ is trigram. These can be shown as Table 5.10.

Table 5.10 Examples of the knowledge patterns used to extract the stage of cancer in MEDLINE abstracts.

Question focus	Background of Rules	Knowledge Patterns	Notes
stage	unigram	‘Stage’ + $\langle x \rangle$	x and y are numbered from 0 to 4 (or I - IV).
	trigram	‘Stage’ + $\langle x - y \rangle$	

Task 2: Passage Retrieval

In this context, we use text filter models to retrieve the passage that is relevant to the specific question.

Task 3: Knowledge Extraction

After we obtain the knowledge pattern developed in task (1), these patterns are stored as a rule base. Some of knowledge patterns can be shown as Table 5.11.

Table 5.11 Some of knowledge patterns

Question	Examples of Knowledge Pattern	Type of Pattern
Histology	squamous cell carcinoma	Keyword matching
	SCC	Keyword matching
Stage	'stage' + <x>	Pattern-match rules
	'stage' + <x - y>	Pattern-match rules
	'stage' + <x and y>	Pattern-match rules
Number of patients	<number> + 'patients'	Pattern-match rules
	<number> + 'women'	Pattern-match rules
Arm-names used	To compare <x> and <y>	Pattern-match rules
Treatment modalities	Chemotherapy	Keyword matching
	Surgery	Keyword matching
	Radiotherapy	Keyword matching

The patterns will be used to extract in clinical knowledge from MEDLINE abstracts, and stored them in an XML format that replaces the unstructured abstract text of the PubMed mark up, called the Clinical Knowledge Markup Language (CKML) (Daelemans et al., 1995). An example of extracting knowledge from cervix cancer abstracts can be shown as Figure 5.16.

5.5.3 Evaluation and Discussion

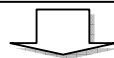
In order to observe the characteristics of our system, we have gathered 200 cervix cancer abstracts in the clinical trial and use them as our experimental datasets. The results of clinical knowledge extraction are evaluated by comparing the results between the domain expert and our system. The experimental results can be presented in the Table 5.12, while an example of the clinical knowledge extracted from PubMed's abstract can be shown as Figure 5.16.

The average of the results is satisfactory. However, the result of extracting arm names is poor. This is because the answer for this question sometimes needs to use advanced semantic rules to interpret the exact meaning of its.

BACKGROUND: To evaluate the efficacy of low or high-dose immunomodulator, Z-100, in combination with radiotherapy for cervical cancer cervix. **METHODS:** Between 1995 and 1999, 221 patients with stage IIIB squamous cell carcinoma of the cervix were randomly assigned to treatment with Z-100 either at 0.2 microg or 40 microg in a double-blind manner in combination with radiotherapy. **RESULTS:** The 5-year survival of patients with high-dose and low-dose Z-100 was 41.5% (95% CI: 31.7-51.3%) and 58.2% (95% CI: 48.7-67.7%), respectively, showing a 30% reduction in the death rate (hazard ratio: 0.670 [95% CI: 0.458-0.980], P = 0.039). Survival of high-dose group was equivalent to the 4-year survival of the radiotherapy plus hydroxyurea arm (49.7%) of GOG120 study, and that of low-dose group was similar to the survival of the cisplatin-based chemoradiation arm. The progression-free survival was also significantly improved in favor of low-dose group (hazard ratio: 0.667 [95% CI: 0.447-0.997], P = 0.048). The survival of low-dose group was similar to the survival of the cisplatin-based chemoradiation arms of the GOG120 study. **CONCLUSIONS:** Unexpectedly, the survival of patients with advanced cervical cancer cervix treated by lower dose of Z-100 in combination with radiotherapy was significantly better than those treated with higher dose Z-100, which was equivalent to the survival with radiotherapy alone. The hypothesis that lower dose of Z-100 enhances the efficacy of radiation therapy is now being tested by placebo-controlled randomized trial.



```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<CKML>
  <inclusion_criteria>
    <ICD10topography>Cervix</ICD10topography>
    <histology>squamous cell carcinoma</histology>
    <Stage>
      <stage_classification>FIGO</stage_classification>
      <stage_grouping>IIIB</stage_grouping>
    </Stage>
  </inclusion_criteria>
  <intent>
    <number_of_patients>221</number_of_patients>
  </intent>
  <Arm>
    <Arm_name>low Z-100 immunomodulator</Arm_name>
    <Arm_name>high-dose Z-100 immunomodulator</Arm_name>
    <Therapy>Radiotherapy<Radiotherapy_Timing>primary</Radiotherapy_Timing>
  </Therapy>
    <Therapy>Chemotherapy<Chemotherapy_Timing>adjuvant</Chemotherapy_Timing>
    <Prescription_DrugName>Cisplatin</Prescription_DrugName>
    <Prescription_DrugName>hydroxyurea</Prescription_DrugName>
    <Prescription_DrugName>immunomodulator</Prescription_DrugName>
    <Prescription_DrugName>Z-100</Prescription_DrugName>
  </Therapy>
  </Arm>
</intent>
</CKML>
```



Summary:
 Type of cancer: cervix cancer
 Histology: Squamous cell carcinoma
 Staging based FIGO: IIIB
 Arm name: (1) low Z-100 immunomodulator, (2) high-dose Z-100 immunomodulator
 Timing: Radiotherapy (primary), Chemotherapy (adjuvant)
 Drugs used: Cispatin, hydroxyurea, Immunomodulator, Z-100

Figure 5.16 An example of extracting clinical knowledge from a PubMed's abstract and storing it in the form of CKML

5.5.4 Evaluation and Discussion

In order to observe the characteristics of our system, we have gathered 200 cervix cancer abstracts in the clinical trial and use them as our experimental datasets. The results of clinical knowledge extraction are evaluated by comparing the results between the domain expert and our system. The experimental results can be presented in the Table 5.12, while an example of the clinical knowledge extracted from PubMed’s abstract can be shown as Figure 5.16.

Table 5.12 The results of extraction after comparing between the domain expert and the system

The Questions	Precision of Answers after comparing with the domain expert (%)
What-histology	100
What-stage	100
How many patients	90
What-arm names used	52
What-modalities used	100
What-timing of each treatment modality	75
What-drugs used	90
Average accuracy	86.71

The average of the results is satisfactory. However, the result of extracting arm names is poor. This is because the answer for this question sometimes needs to use advanced semantic rules to interpret the exact meaning of its.

Consider following sentence, “*The overall clinical response rate was 84.6% and included a complete response (CR) in four patients (7.7%), partial response (PR) in 40 patients (76.9%), and disease (SD) in the remaining eight patients (15.4%). Surgery revealed positive nodes in 9.6% neoadjuvant chemotherapy group patients and in 29.6% primary surgery group patients (P = 0.014).*” If it is analysed by an expert human, *using neoadjuvant chemotherapy* and *using surgery alone* can be extracted as the arm-names. However, this sentence cannot be extracted clinical knowledge by using the current rules. Therefore, these proposed rules or templates should be improved to support the process of analysis of the system.

5.6 Chapter Conclusion

This chapter describes a contribution to the On-KDT methodology to search knowledge within the MEDLINE database relevant to clinical trials for cervix cancer. The extracted knowledge is called '*clinical trial knowledge related to cervix cancer*' and it is represented in a named Clinical Knowledge Markup Language (CKML).

Evidence-based medicine is a difficult topic. When finding relevant knowledge in PubMed to support clinical decision-making is imprecise, a solution to this problem is required. This is because the traditional knowledge discovery from PubMed abstracts involves the use of several manual processes. Most clinicians use PubMed to retrieve abstracts and then to read the acquired abstract for relevance. This selection process is time-consuming and labour-intensive, especially when a search returns a large number of abstracts and each abstract may contain a large volume of information.

In order to fully utilize these on-line documents effectively, it is crucial to be able to take advantage of the synopsis knowledge of PubMed abstracts. Therefore, this extracted knowledge may offer the following benefits. Having a knowledge base would increase efficiency immensely as it would reduce the time needed to select the most relevant knowledge of the requested domain. More timely provision of more relevant knowledge provides a better environment, leading to better clinical outcomes.

In medical domains, where data and analytics-driven research is successfully applied, new and novel research directions can be identified to further advance the clinical and biological studies. This is because; the extracted knowledge can offer a better solution to clinical decision-making. In this context, the PubMed abstracts relevant to cervix cancer are used as a dataset, where cervix cancer is an important disease causing of cancer-related disease and death among women worldwide. As the previous study, it has no a system that offers the previous clinical decision treatments used as the prior knowledge for future clinical decisions. As this, it becomes the motivation for the investigation in this thesis.

This approach may offer the several benefits such as increased efficiency for PubMed searches by reducing the time to select and collect the most relevant abstracts for the requested domain, leading to better clinical decisions because of better assimilation of the literatures relevant to their researches and clinical decision.

CHAPTER 6

THE ON-KDT METHODOLOGY APPROACH FOR EXTRACTING BUSINESS RULES FROM PROCESS MODEL REPOSITORIES

6.1 Introduction

A business process is defined as consisting of a set of logically related activities, performed by their relevant roles or collaborators, to intentionally achieve the common business goals (Ye et al., 2007). Business processes have become the core assets of many organizations since they generate revenue and often represent a significant proportion of costs. Nowadays, modelling and managing business processes is an important approach for managing organizations from an operational perspective. In fact, a recent study (Hill et al., 2007) has shown that the business process management (BPM) software market reached nearly \$1.7 billion in total software revenue in 2006 and this number continues to grow. *It now becomes common for most medium to large to have collections of hundreds or even thousands of business process models.* For example, the BIT Process Library has 735 process models, (Larose, 2005), the SAP Reference Model contains 604 process models (May et al., 2008), and there are 6,000+ process models in Suncorp's process model repository for insurance (Johnson, 1967). Such large collections of process models form a valuable source of data which knowledge can be harvested. To the best of our knowledge, obtaining of the business knowledge can be done by a well-known technique, also known as process mining (van der Aalst and Weijter, 2003, van der Aalst and Weijters, 2004, van der Aalst et al., 2008). Process mining provides the capability to discover, detect, control, organize, and monitor actual process execution by extracting useful knowledge from event logs (i.e. process instances).

However, there has been limited work in mining process models, where a large body of research work in the area of process mining (van der Aalst and Weijter, 2003, van der Aalst and Weijters, 2004, van der Aalst et al., 2008) mainly focuses on the derivation of process models from event logs. Nevertheless, there are settings in which process mining proves to be inadequate. If a process has never been performed, no event logs information is available, rendering the technique useless. In addition, learning/data mining techniques, such as those used in this approach to process, only return reliable results when there are sizeable datasets (in the case of process mining, event logs) available. If the size of the set of event logs is small (which is often the case with infrequently executed processes, such as those

associated with disaster management, or with Greenfield applications), the reliability of the knowledge extracted cannot be guaranteed.

6.2 Motivation

With the limitation in process mining, the work presented in this chapter aims to explore an alternative dimension to process mining, where the original concept of process mining techniques do not return reliable results. The aim of this work is to concentrate on an alternative dataset - process models as opposed to process instances (i.e. event logs), and extract process constraints as opposed to process models. The expected results of mining process model repositories are business rules, where a business rule defines one aspect of IT system that is intended to assert IT system structure or influence the behaviour of IT system. Such business rules represent business constraints that have been encoded in a process model.

6.3 Research Contribution

We apply the On-KDT methodology to find sequential patterns hidden in process model repositories, where process models are viewed as text. The extracted sequential patterns will be considered as business rules. In this context, business rules that are automatically extracted in our approach represent significant and valid sequential correlations among business activities belonging to a particular organization.

6.4 Preliminaries

In this section, we describe three more preliminaries that are required to make more understanding about the process model repositories.

6.4.1 Understanding of Business Process Characteristics

A business process is an activity or set of activities that will accomplish a specific organisational goal. In a software system, it can be produced from the software requirements and conditions. Simply speaking, the business process is actually often a collection of interrelated processes that function in a logical sequence to achieve the ultimate goal. In general, the business process can be designed and represented by using graphical notation such as flow charts, data flow diagram, and BPMN. An example of the business process can be shown as Figure 6.1.

It is possible that a sequence of activities that can be found in many business processes may be business rules, where it can define or constrain some aspect of a

system. Besides, it can represent both the software requirements and conditions to which the system should conform (Wan-Kadir and Loucopoulos, 2004).

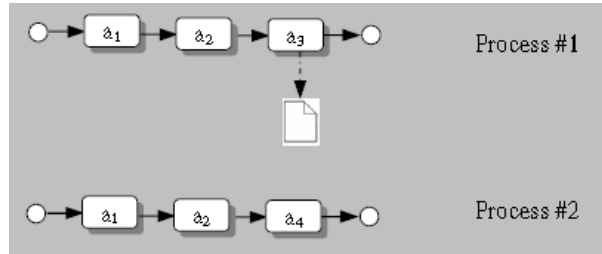


Figure 6.1 Examples of process models

In general, a sequential data consists of ordered elements or events (e), denoted as $\langle e_1 e_2 \dots e_n \rangle$ (Agrawal and Srikant, 1995, Srikant and Agrawal, 1996). Consider the business process in Figure 6.1. It can be seen that a business process can be viewed as a similarity of the sequential data. Therefore, the process model 1 and 2 shown in Figure 6.1 can be denoted as $\langle e_1 e_2, e_3 \rangle$ and $\langle e_1 e_2, e_4 \rangle$ respectively, where ‘ a ’ is an activity (or a task).

6.4.2 Definition of transforming the process models into sequence data forms

This section provides a set of definitions that can be used to transform process model to sequences.

Definition 1: A process can be transformed into a number of activity-sequences. Each activity-sequence is denoted as $S = \langle s_1, s_2, \dots, s_i \rangle$, where s_i can be an *activity-set* or a *conditional activity-sequence*.

Definition 2: An activity-set can be a set of activities and/or activity-sequences that are performed together (i.e. in parallel). For example, $\{a_1, a_2\}$ denotes activity a_1 and a_2 performed together whilst $\{a_1, \langle \{b_1\}, \{b_2\} \rangle\}$ denotes activity a_1 and activity sequence $\langle \{b_1\}, \{b_2\} \rangle$ performed together.

Definition 3: A conditional activity-sequence is denoted as $\langle s_1, s_2, \dots, s_n \rangle_{cond}$, meaning that s_1, s_2, \dots, s_n would be performed (in sequence) under the condition *cond* is true.

The following examples demonstrate how process models with different gateways can be transformed into the form of activity-sequences based on the above definitions.

Table 6.1 Examples of transforming process models to activity-sequences

Original process models	Gateway	Activity-sequences
	None	S1: <{A}, {B}, {C}>
	AND or parallel	S1: <{A}, <{B}, {M}>, <{Y}>>
	OR	S1: <{A}, {B}, <{E}, {F}>, {D}> S2: <{A}, {B}, <{C}>, {D}> S3: <{A}, {B}, <{<{E}, {F}>, <{C}>}>, {D}>
	XOR	S1: <{A}, <{B}, {M}>_{C1}, {D}> S2: <{A}, <{C}>_{C2}, {D}>

6.4.3 The stop-list

In the case of general data preparation in the ON-KDT methodology, if we have a large dataset of process models, the step of identifying more specific individual data will be required. At the beginning, the process model collection should be transformed into a form that is easier for analysis. This work applies the classical *vector space model (VSM)*, to create a space in which both documents and queries (document's elements) are represented by vectors. Each process model P_i contains a sequence of activities (a) or tasks and it can be represented by a t -dimensional vector.

$$P_i = \{a_{i1}, a_{i2}, a_{i3}, \dots, a_{it}\}$$

The VSM has been widely used in the traditional *information retrieval (IR)* field (Baeza-Yates & Ribeiro-Neto 1999). In this context, a process model is equivalent to a document, while the queries are gateways and tasks. An example can be illustrated as Figure 6.2.

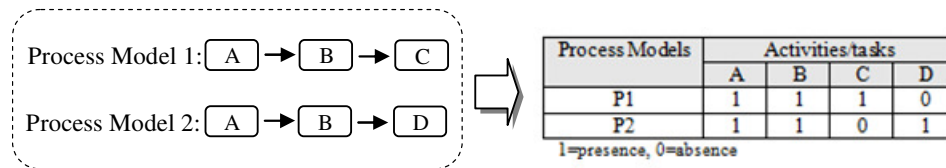


Figure 6.2 An example of transforming process models to the VSM

After transforming the process model into the VSM, data will be improved the quality by a data cleaning technique. In this case, some elements of the process model will be filtered out prior to other processes of the ON-KDT methodology. This is because these elements are frequently occurring, but meaningless for searching purposes. These elements are the ‘start’, ‘intermediate’ and ‘end’ events. The start events indicate where a process will begin, while the end events indicate where a process will end. The intermediate events indicate that the activity should be interrupted when the event is triggered. These elements occur frequently in the process models, but they do not help much narrow down search results. Therefore, they can be ignored at search time. The list of elements is called the *stop-element list* (shown as Figure 6.3).

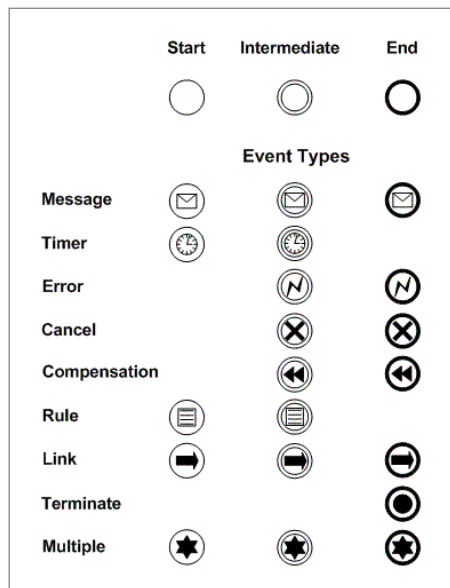


Figure 6.3 All of the event elements added in the stop-element list

In fact, the concept above is similar to the concept of *stop-word* (or *stop-list*) in natural language processing and information retrieval (Fox 1990²², Baeza-Yates & Ribeiro-Neto 1999). In computing, words which do not contain important significance to be used in text processing will be removed or ignored. This is because these words return vast amount of unnecessary information, but they are deemed irrelevant for searching purposes. There are many stop words using in natural language processing such as ‘a’, ‘an’, ‘the’, ‘about’, ‘in’, ‘on’, ‘before’, and so on. Therefore, they are dropped at indexing time and then ignored at search time (Baeza-Yates & Ribeiro-Neto 1999).

22 Christopher Fox (1990) A Stop list for Gneral Text, ACM SIGIR Forum, 24(1-2)

6.5 Mining Business Rules from Process Models

The elaboration of utilizing the On-KDT methodology to extract business rules from process model repositories can be described as follows.

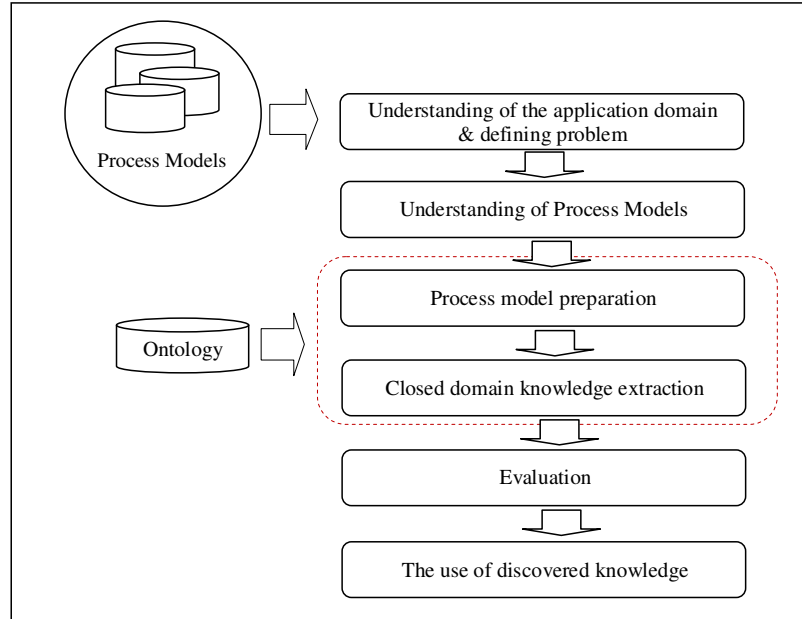


Figure 6.4 The On-KDT methodology approach for finding knowledge from process models

Step 1: Understanding of the application domain and defining problem

This step is to acquire the relevant prior knowledge and the goals of knowledge discovery, where we need to know what to look for in the dataset of process models. In this context, it aims to find business rules from process model repositories. The business rules are significant and valid sequential correlations among business activities belonging to a particular organization/system (Grobelnik et al., 2000b).

Step 2: Understanding of data (process model repositories)

This step focuses on the selection of data samples that are relevant to the objective of knowledge discovery and this step includes querying the existing data to select the desired subset. In this context, the dataset is a set of process models. Before the dataset will be performed in the next steps, we need to make more understanding about the dataset (e.g. structure of data) in order to make the resulting dataset more effective for the purposes.

In this setting, our process models are represented based on Business Process Model Notation (BPMN) (Miers and White, 2008), which is a graphical representation for defining business processes²³.

BPMN defines a business process diagram (BPD) for creating graphical models of business process operations. BPD is based on a flowcharting technique. A business process model is a network of graphical objects representing activities/works/tasks and the flow controls that defines their order of performances. There are four basic categories of elements: Flow Objects, Connecting Objects, Swimlanes, and Artifacts. Flow objects consist of three objects: Event, activity, and Gateway. The flow objects are connected together in a diagram by the connecting objects. The connecting objects consist of sequence flow, message flow, and association. For Swimlanes, they are two types of swimlanes: pool and lane. A pool is used to represent a participant in a process, while lane is used to organize and categorize activities. For Artifacts, they can be used to provide the additional detail and context in a specific model. The core set of BPMN basics can be represented in Figure 6.5.

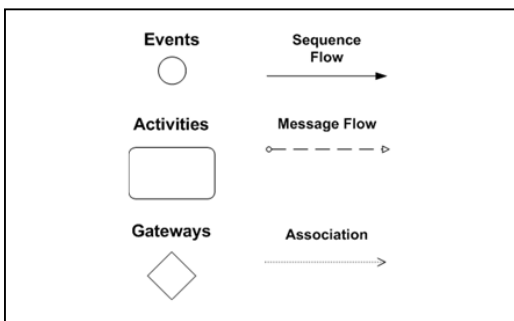


Figure 6.5 The core set of BPMN Basics

Each BPMN type can be extended to other elements. For example, all of the event elements can be shown in Figure 6.6.

After the dataset of process models is carefully considered, it can be found that directly extracting business rules from process models represented on graphical diagram such BPMN is a difficult task. Therefore, the collection of process models will be transformed into a textual format from which it is easier to extract knowledge. This work uses the XML Metadata Interchange (XMI)²⁴ standard, which is an Object Management

²³ BPMN: <http://www.bpmn.org/>, <http://www.omg.org/spec/BPMN/>

²⁴ <http://www.omg.org/technology/documents/formal/xmi.htm>

Group (OMG) standard for exchanging metadata information through Extensible Markup Language (XML).

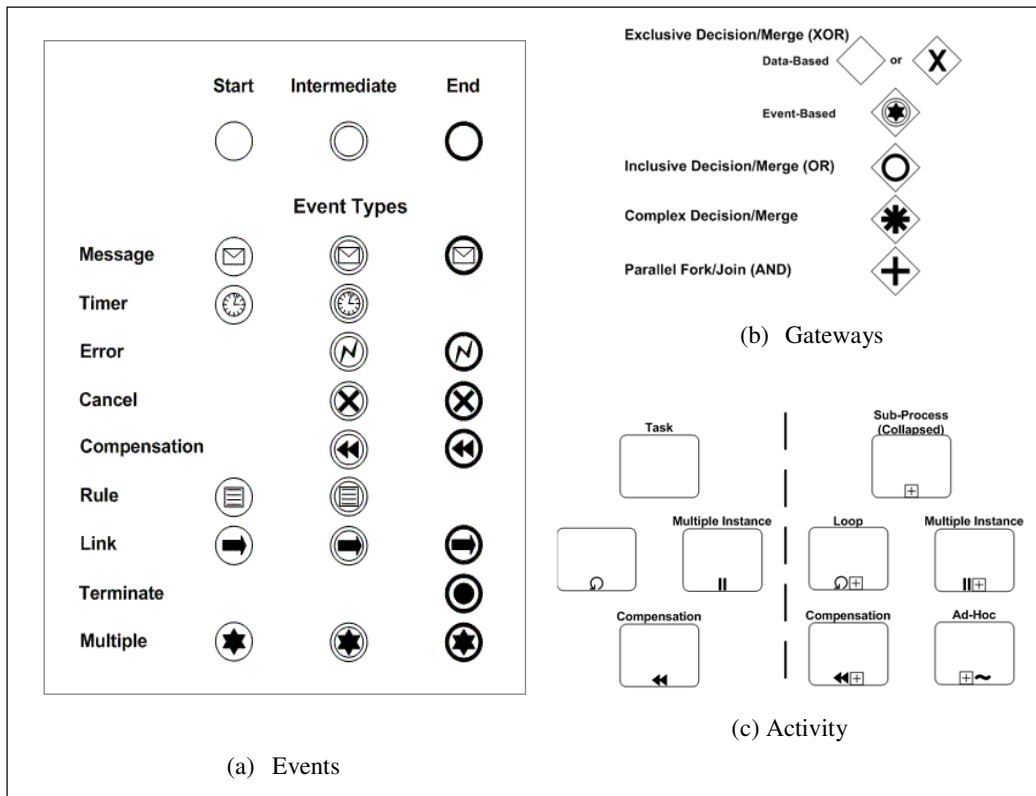


Figure 6.6 Some of Event elements

A simple example of transformation can be presented as Figure 6.7. It can be found that each process model contains too much detail of text after transforming to XMI. As this, we present one instance of the use of this methodology of extracting knowledge from non-textual data, such process models, where we view this data as text.

A simple example of transformation can be presented as Figure 6.7. It can be found that each process model contains too much detail of text after transforming to XMI. As this, we present one instance of the usage of this methodology of extracting knowledge from non-textual data such process models, where we view this data as text.

Consider the examples of process models in Figure 6.7. It can be determined that a process model contains a set of tasks/activities, which define any workflow of a system execution. The workflow in the process model is an ordered sequence of activities to run on the specific system.

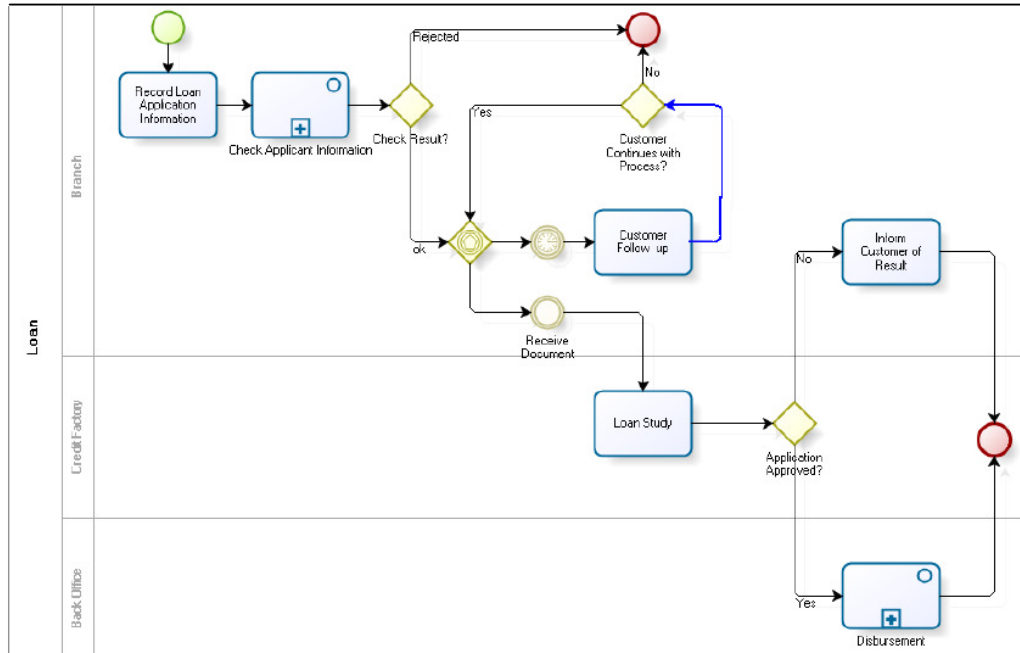
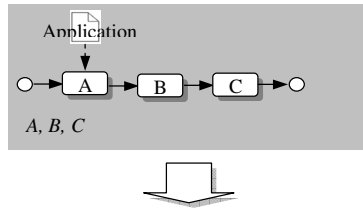


Figure 6.7 An example of process model constructed by using BPMN

(<http://www.bizagi.com/eng/downloads/BPMNbyExample.pdf>)

Formally, a repository of process models can be defined as a set of process models $P = \{p_1, p_2, p_3, \dots, p_n\}$. Each process model p_i consists of a set of elements $E = \{e_1, e_2, e_3, \dots, e_n\}$, where each element e_i is an ordered list (i.e. sequence) of tasks/activities. As this, it can be seen that the characteristic of the process model is similar to the sequential data, where a sequential data contains the sequence of ordered elements/events.

The detail of the sequential data can be described following. Let I be a set of items, $I = \{i_1, i_2, \dots, i_m\}$. An element may contain a set of items. Items within an element are unordered. An element (or an itemset) is denoted by $\{x_1, x_2, \dots, x_k\}$, where $x_j \in I$ is an item. Therefore, a sequence s is an ordered list of elements, denoted as $\langle e_1 e_2 \dots e_i \rangle$, where e_i is also called an element of the sequence s . Furthermore, a sequence of length k is called k -sequence, while the size of a sequence is the number of elements (or itemsets) in the sequence. An example of a process model transforming to XMI can be presented as Figure 6.8.



```

<?xml version="1.0" encoding="UTF-8"?>
<bpmn:BpmnDiagram xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:bpmn="http://stp.eclipse.org/bpmn
xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore" xmi:id="_XfmYwWUZEd6o3OQ7dzE7fg"
ID="_XfmYwWUZEd6o3OQ7dzE7fg">
...
...
<artifacts xmi:type="bpmn:DataObject" xmi:id="_tr-7YWUZEd6o3OQ7dzE7fg" iD="_tr-7YGUZEd6o3OQ7dzE7fg"
name="Application"> <associations xmi:type="bpmn:Association" xmi:id="_y74HsGUZEd6o3OQ7dzE7fg"
target="_XgV_o2UZEd6o3OQ7dzE7fg"/></artifacts>
<vertices xmi:type="bpmn:Activity" xmi:id="_XgV_o2UZEd6o3OQ7dzE7fg" iD="_XgV_omUZEd6o3OQ7dzE7fg"
associations="_y74HsGUZEd6o3OQ7dzE7fg" outgoingEdges="_6Du7MWUZEd6o3OQ7dzE7fg"
incomingEdges="_yhw-kWUZEd6o3OQ7dzE7fg" name="A" activityType="Task"/>
<vertices xmi:type="bpmn:Activity" xmi:id="_1JB7wWUZEd6o3OQ7dzE7fg" iD="_1JB7wWUZEd6o3OQ7dzE7fg"
associations="_y74HsGUZEd6o3OQ7dzE7fg" outgoingEdges="_6Du7MWUZEd6o3OQ7dzE7fg"
incomingEdges="_yhw-kWUZEd6o3OQ7dzE7fg" name="B" activityType="Task"/>
...
...
</bpmn:BpmnDiagram>

```

Figure 6.8 An example of transforming a process model into XMI

In the previous study, most researches relating to sequential database focus on two main issues. The first issue is sequential pattern mining. It aims to find the frequently occurred sequences to describe the data or predict future data (Zhao and Bhowmick, 2003). The second issue is to find for statistical relevant patterns between occurrences of sequential events/elements. This technique is called sequential pattern mining (Aalst and Weijter, 2004, Agrawal and Srikant, 1995, Zhao and Bhowmick, 2003). The final results of sequential pattern mining can be called *sequential rules* that satisfy the minimum support threshold. Finding of useful knowledge from a sequential database is very similar to the Association Rule mining, but sequence in this case matters. As this, if we need to extract business rules in the form of sequential pattern from the process model repositories, these data should be re-formatted as the sequential form in order to be appropriate for the method of useful pattern extraction.

Unfortunately, the process model may be more complicated than the general sequential data, where the process model is also modelled with decision points, called Gateway(s). As this, the process model containing a gateway can be many alternative meanings based on condition expressions for each gate of the gateway. The important gateways used in modeling process models based on BPMN can be OR, AND, and XOR gateways.

The ‘XOR’ gateway (also called exclusive gateway) represents an out-going path where the alternatives are based on condition expressions for each gate of the gateway (Owen and Raj, 2003, Miers and White, 2008). An example can be shown as Figure 6.9.

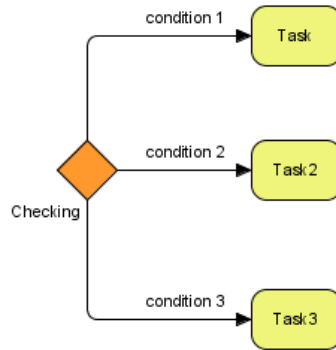


Figure 6.9 An example of the process model containing the XOR gateway

As Figure 6.9, all possible paths of the business process separated can be represented following.

1. If condition 1 is true, then taks-1 is performed.
2. If condition 2 is true, then taks-2 is performed.
3. If condition 3 is true, then taks-3 is performed.

The ‘OR’ gateway (also called inclusive gateway) can represent more than 1 out-going paths. The condition checking process of this gateway will have a little bit different the exclusive gateway. An example can be illustrated as follows.

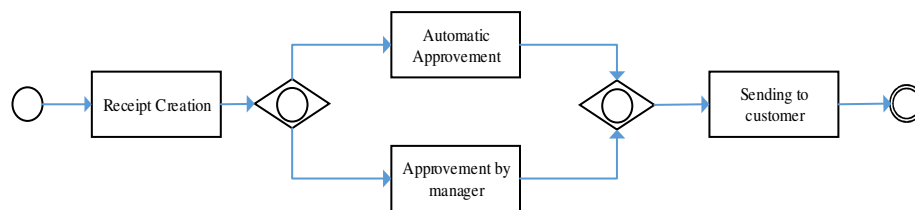


Figure 6.10 An example of the process model containing the OR gateway

Consider Figure 6.10. All possible paths of the business process separated can be represented following.

1. A receipt can be automatically approved.
2. A receipt can be approved by manager.
3. A receipt can be approved by system and manager.

The 'AND' gateway (also called parallel gateway) must receive an input signal from all input sequential flows used as the output to be taken.

Step 3: Process model preparation

Process model preparation is an important step describing any type of processing performed on raw data to prepare it for another procedure. The purpose of process model preparation is to make this data easier to search and discover for hidden knowledge.

The step of data preparation in the On-KDT methodology consists of two main tasks: *pre-processing data* and *identifying a specific individual data*. The main objective of the pre-processing data is to detect and remove errors and inconsistencies from data in order to improve the quality of data. Also, data pre-processing includes transforming the specific data to a new form that will be more easily and effectively processed to meet the needs of the user. The second step is to identify more specific individual data driven by the concept of data classification/clustering.

As the objective of the data preparation in the On-KDT methodology is to identify more a specific individual data before passing to the step of extracting useful knowledge. This is because the ability of acquiring knowledge depends on something more specific than general data. However, in this approach, it is noted that the general data preparation in the ON-KDT methodology will be effected for the very large set of database. Unfortunately, our process model dataset used in this experiment is quite small. Therefore, the data preparation step in this context should be modified as the step of transforming the process model repositories of the sequence data. By using the definition of transforming the process model to the new form of sequential data, we can show an example as follows. Suppose there are seven process models shown as Table 6.2.

Table 6.2 The example of process models

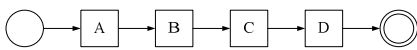
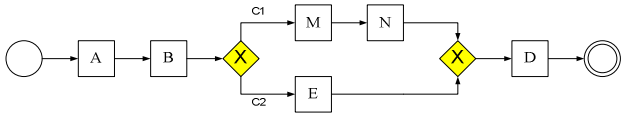
Process models	Original process models
P1	
P2	

Table 6.2 (cont')

Process models	Original process models
P3	
P4	
P5	
P6	
P7	

These process models must be transformed into the sequence form that is suitable for the process of business rule extraction. The sequence data that are generated from the original process models (presented as Table 6.2) can be shown in Table 6.3.

Table 6.3 Examples of generating sequences from process models

Process Model	Generating sequences from process models	
P1	Original form	
	Sequentail form	$S_{1,1}: \langle \{A\}, \{B\}, \{C\}, \{D\} \rangle$

Table 6.3 (cont')

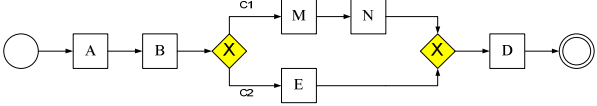
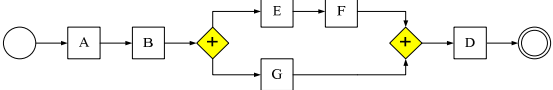
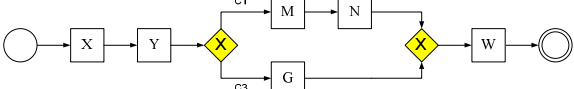
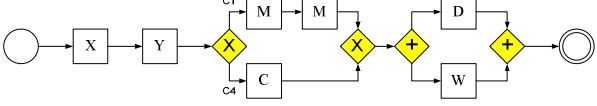
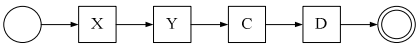
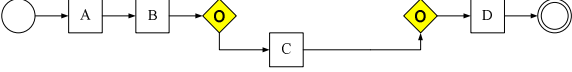
Process Model	Generating sequences from process models	
P2	Original form	
	Sequential form	$S_{2,1}: v\langle\{A\}, \{B\}, \langle\{M\}, \{N\}\rangle_{c1}, \{D\}\rangle$ $S_{2,2}: \langle\{A\}, \{B\}, \langle\{E\}\rangle_{c2}, \{D\}\rangle$
P3	Original form	
	Sequential form	$S_{3,1}: \langle\{A\}, \{B\}, \langle\langle\{E\}, \{F\}\rangle, \langle\{G\}\rangle\rangle, \{D\}\rangle$
P4	Original form	
	Sequential form	$S_{4,1}: \langle\{X\}, \{Y\}, \langle\{M\}, \{N\}\rangle_{c1}, \{W\}\rangle$ $S_{4,2}: \langle\{X\}, \{Y\}, \langle\{G\}\rangle_{c3}, \{W\}\rangle$
P5	Original form	
	Sequential form	$S_{5,1}: \langle\{X\}, \{Y\}, \langle\{M\}, \{N\}\rangle_{c1}, \{\langle\{D\}\rangle, \langle\{W\}\rangle\}\rangle$ $S_{5,2}: \langle\{X\}, \{Y\}, \langle\{C\}\rangle_{c4}, \{\langle\{D\}\rangle, \langle\{W\}\rangle\}\rangle$
	Original form	
	Sequential form	$S_{6,1}: \langle\{X\}, \{Y\}, \{C\}, \{D\}\rangle$

Table 6.3 (cont')

Process Model	Generating sequences from process models	
P7	Original form	
	Sequential form	$S_{7,1}: \langle \{A\}, \{B\}, \langle \{E\}, \{F\} \rangle, \{D\} \rangle$ $S_{7,2}: \langle \{A\}, \{B\}, \langle \{C\} \rangle, \{D\} \rangle$ $S_{7,3}: \langle \{A\}, \{B\}, \langle \langle \{E\}, \{F\} \rangle, \langle \{C\} \rangle \rangle, \{D\} \rangle$

Finally, these sequence data will be passed to the next step, the step of extracting business rules by closed-domain knowledge extraction.

As it is said in Chapter 3, the ON-KDT methodology is the conceptual model designed for finding knowledge from unstructured/semi-structured data describing with text. The ON-KDT methodology integrates the ontology's semantic data to improve search accuracy for knowledge discovery by understanding the contextual meaning of terms. The main ontology integrated in the proposed methodology is the VL-ontology. In this context, this step will utilize the VL-ontology to interpret the contextual meaning of terms as they appear in context, when the names of activities containing in process models are written in natural language. Some process's names can be represented in the same meaning but different representations. Therefore, the VL-ontology can help to address this problem.

Step 4: Extracting Business Rules by Closed-domain Knowledge Extraction

As we have discussed in the chapter 3 that, there two solutions for finding the useful knowledge from unstructured/semi-structured text.

In this context, the main method of business rule extraction through the ON-KDT has driven on the concept of sequential pattern mining (Adamo, 2000, Agrawal and Srikant, 1995, Han and Pei, 2000, Pei et al., 2007, Qiao et al., 2012, Srikant and Agrawal, 1996, Zhao and Bhowmick, 2003). Then, we have been applied the Generalized Sequential Pattern Mining (GSP mining) (Srikant & Agrawal 1996) to find business rules represented in the form of *sequential-process pattern*.

Let P be a set of process models, $P = \{p_1, p_2, p_3, \dots, p_i\}$. A sequence is an ordered list of *elements*, denoted as $\langle e_1 e_2 \dots e_l \rangle$, where an element can be an activity-set, a set of sub-sequences of process model performed on a condition gateway (described in the section

7.4). After transforming the process models in the form of sequential data, each process model will be called *activity sequence*, denoted as p .

The number of elements in activity sequence p is called the *length of the sequence* (L), denoted as $|p|$. An activity sequence with length k is called k -sequence. The i -th element in the activity sequence p is represented by p_i . The empty activity sequence is denoted by $\langle \dots \rangle$. The result of the concatenation of two activity sequences a and b is a new activity sequence p , denoted as $p = ab$.

In addition, although our data are in the form of sequence, they are different from others. This is because a process model can contain the decision gateway. This notation can make a process model generated to sub-sequences. Therefore, we have to provide some constraints that are used to handle about generating of *sequence candidates*. All of constraints can be described following.

Let β be a set of constraints, $\beta = \{C_1, C_2, C_3, \dots, C_n\}$. The constraints can be represented below.

Definition C₁: If a process model represented in the form of $\langle \{a_1\}, \{a_2\}, \{a_3\}, \dots, \{a_n\} \rangle$, L_1 -sequence can be generated and it is denoted as $\{activity_n\}$.

An example of generating a process model that has no any decision gateway to L_1 -candidates can be shown as Figure 6.11.

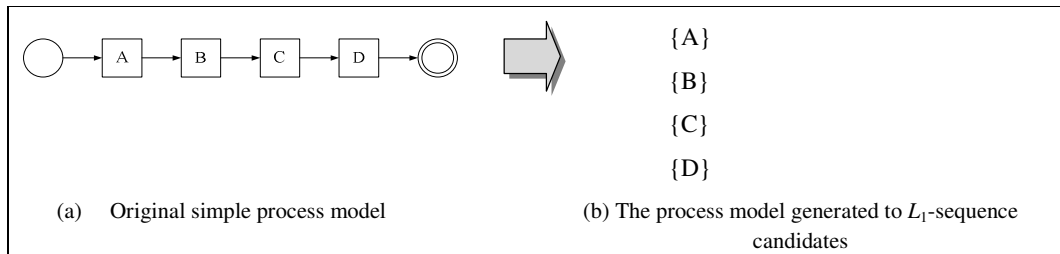


Figure 6.11 An example of of generating a simple process model to L_1 -sequence candidates

Definition C₂: If a process model contains the exclusive gateway, the sub-sequences of process model that are in the domain of exclusive gateway can be generated to L_1 -sequence, denoted as $\langle \{activity\} \rangle_{condition-i}$.

An example of generating a process model that contains the inclusive gateway to L_1 -candidates can be shown as Figure 6.12.

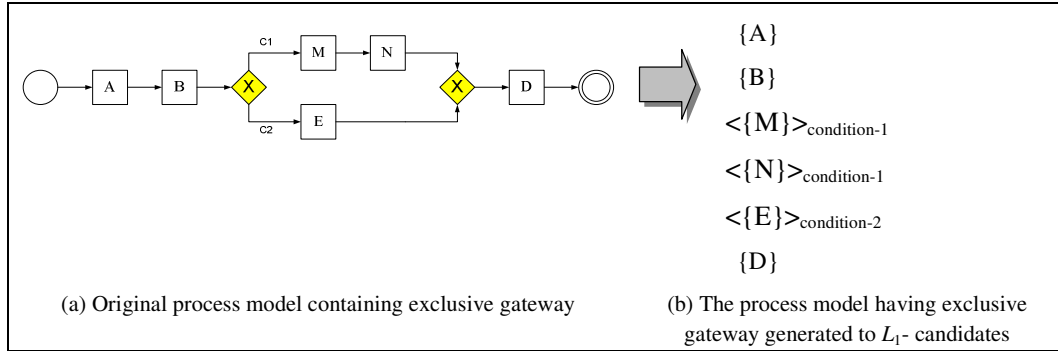


Figure 6.12 An example of generating a process model containing exclusive gateway to L_1 -sequence candidates

Definition C₃: If a process model contains inclusive gateway (OR), the sub-sequences of process model that are in the domain of inclusive gateway can be generated to L_1 -sequence, denoted as $\langle \{activity\} \rangle_{OR}$.

An example of generating a process model that contains the inclusive gateway to L_1 -candidates can be shown as Figure 6.13.

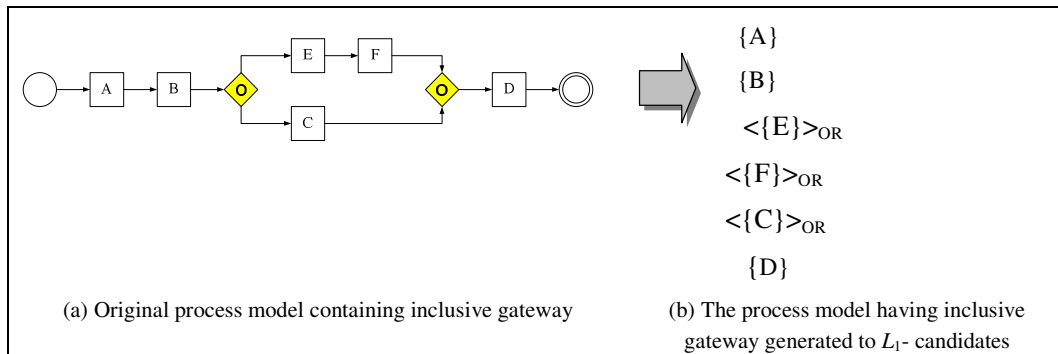


Figure 6.13 An example of generating a process model containing inclusive gateway to L_1 -sequence candidates

Definition C₄: If a process model contain a parallel gateway (AND), the sub-sequences of process model that are in the domain of parallel gateway can be generated to L_1 -sequence, denoted as $\langle \{activity\} \rangle_{AND}$.

An example of generating a process model that contains the parallel gateway to L_1 -candidates can be shown as Figure 6.14.

It is noted that if the candidates are in different forms, they also have different meanings. For example, suppose there are three process models as follows.

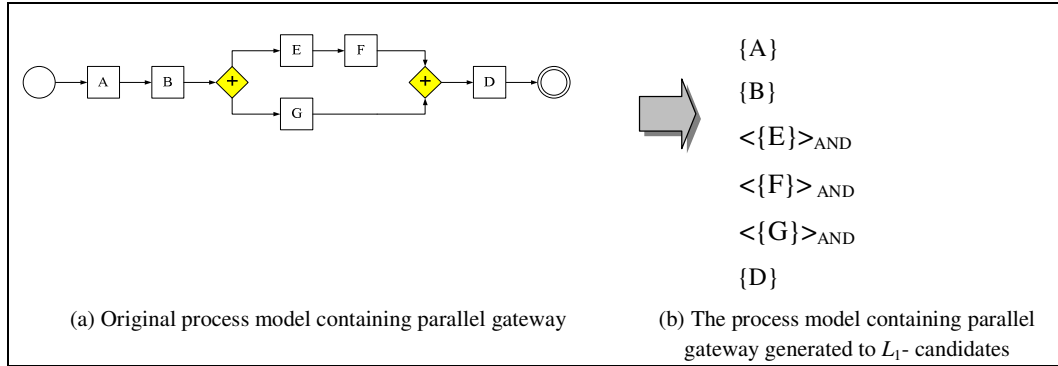


Figure 6.14 An example of generating a process model containing parallel gateway to L_1 -sequence candidates

$$p_1 = \langle \{a\}, \langle \{b\}, \{c\} \rangle, \{d\} \rangle$$

$$p_2 = \langle \{a\}, \langle \{b\}, \{c\} \rangle_{\text{cond-1}} \rangle$$

$$p_3 = \langle \{a\}, \langle \{b\} \rangle_{\text{OR}}, \{d\} \rangle.$$

L_1 -sequence candidates that are generated from p_1 are $\{a\}$, $\langle \{b\} \rangle$, $\langle \{c\} \rangle$, and $\{d\}$.

L_1 -sequence candidates that are generated from p_2 are $\{a\}$, $\langle \{b\} \rangle_{\text{cond-1}}$, and $\langle \{c\} \rangle_{\text{cond-1}}$. L_1 -sequence candidates that are generated from p_3 are $\{a\}$, $\langle \{b\} \rangle_{\text{OR}}$, and $\{d\}$.

Consider the activity b . It is represented in different forms after generating these process models to L_1 -sequence candidates. The L_1 -sequence candidates are $\{b\}$, $\langle \{b\} \rangle_{\text{cond-1}}$, and $\langle \{b\} \rangle_{\text{OR}}$. These candidates have different meanings.

Definition C5: To generate candidate length $(k+1)$ sequence from length- k frequent sequences using Apriori, if some candidates contains the same condition gateway, they can be generated to L_{k+1} -sequence, denoted as $\langle \{activity_1\} \{activity_2\} \dots \{activity_n\} \rangle_{\text{condition gateway}}$. An Example is shown in Figure 6.15.

All above, the modification of the GSP mining algorithm used in this context can be presented as follows.

1. Scan the process repository based on the constraints C_1 , C_2 , C_3 and C_4 , and take sequences as length-1 (L_1) candidate with min-support.
2. For each level (e.g. sequence of length- k) do
 - 2.1 Scan process repository to collect support count for each candidate sequence
 - 2.3 Generate candidate length $(k+1)$ sequence from length- k frequent sequences using Apriori through the consideration of constraints C5

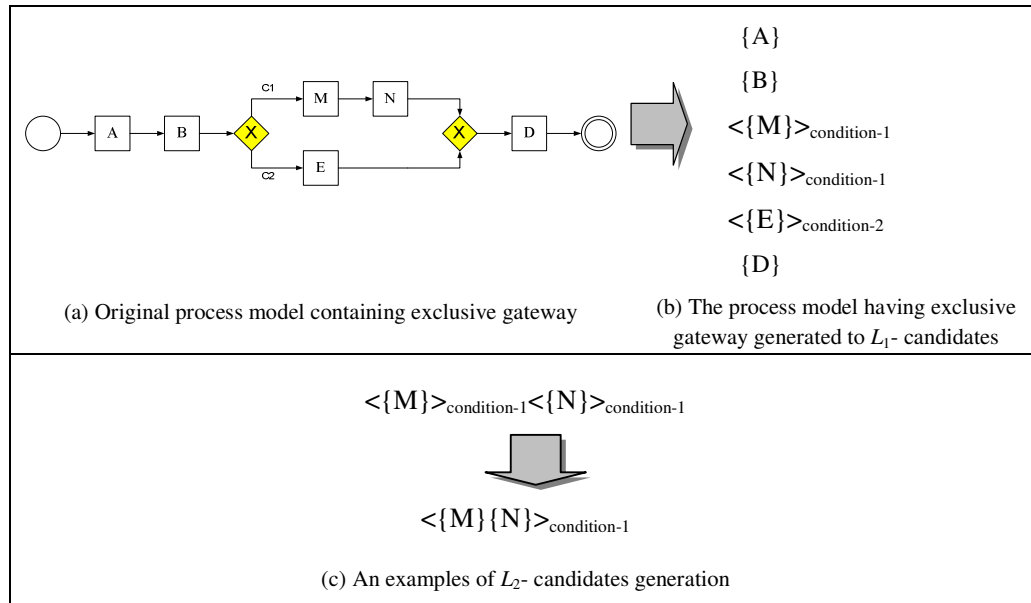


Figure 6.15 An example of generating a process model containing exclusive gateway to L_2 -sequence candidates

3. Repeat until no frequent sequence / no candidate can be found
4. Generate sequential rules that satisfy the support and confidence thresholds

An example of applying the constraint-based GSP mining algorithm to extract business rules from process model repositories can be represented as follows.

Step 1: Scan the database based on the constraints (C_1 - C_4), and take sequences as length-1 (L_1) candidate with min-support.

Consider the activity-sequences shown as Table 6.3. If we determine the min-support as 2, L_1 -sequence candidates can be shown as Figure 6.15

- Step 2:* For each level (e.g. sequence of length- k) do
- 2.1 Scan database to collect support count for each candidate sequence
 - 2.2 Generate candidate length ($k+1$) sequence from length- k frequent sequences based on the constraint C_5
 - 2.3 Candidates pruning by Apriori

Step 3: Repeat until no frequent sequence / no candidate can be found

The min-support is 2. L_2 -sequence, L_3 -sequence, and L_4 -sequence candidates can be shown as Figure 6.16 – 6.17.

L1-sequence	Support	Pruned
{A}	8	
{B}	8	
{C}	2	
{D}	8	
{X}	4	
{Y}	4	
{W}	2	
<{M}> _{c1}	3	
<{N}> _{c1}	3	
<{E}> _{c2}	1	Less than min-supp
<{G}> _{c3}	1	Less than min-supp
<{C}> _{c4}	1	Less than min-supp
<{E}>	3	
<{F}>	3	
<{G}>	1	Less than min-supp
<{D}>	2	
<{W}>	2	
<{C}>	1	Less than min-supp

L1-sequence	Support
{A}	8
{B}	8
{C}	2
{D}	8
{X}	4
{Y}	4
{W}	2
<{M}> _{c1}	3
<{N}> _{c1}	3
<{E}>	3
<{F}>	3
<{D}>	2
<{W}>	2

Figure 6.16 Generating L_1 -sequence candidates

L2-sequence	Support
{A}{B}	7
{A}{D}	7
{A}<{E}>	3
{A}<{F}>	3
{B}{D}	7
{B}<{E}>	3
{B}<{F}>	3
{C}{D}	2
{X}{Y}	5
{X}{W}	2
{X}<{M}> _{c1}	2
{X}<{N}> _{c1}	2
{X}<{D}>	2
{X}<{W}>	2
{Y}{W}	2
{Y}<{M}> _{c1}	2
{Y}<{N}> _{c1}	2
{Y}<{D}>	2
{Y}<{W}>	2
<{M}> _{c1} <{N}> _{c1}	3
<{E}><{F}>	3
<{E}><{D}>	3

L3-sequence	Support
{A}{B}{D}	5
{A}{B}<{E}>	3
{A}{B}<{F}>	3
{A}<{E}><{F}>	3
{A}<{E}><{D}>	3
{B}<{E}><{F}>	3
{B}<{E}><{D}>	3
{X}{Y}{W}	2
{X}{Y}<{M}> _{c1}	2
{X}{Y}<{N}> _{c1}	2
{X}{Y}<{D}>	2
{X}{Y}<{W}>	2
{X}<{M}> _{c1} <{N}> _{c1}	2
{Y}<{M}> _{c1} <{N}> _{c1}	2

L3-sequence	Support
{A}{B}<{E}><{F}>	2
{A}{B}<{E}><{D}>	2
{X}{Y}<{M}> _{c1} <{N}> _{c1}	2

Figure 6.17 Generating L_2 -sequence, L_3 -sequence, and L_4 -sequence candidates

Step 4: Generate sequential rules that satisfy the support and confidence thresholds

The business rules that are generated from process model repositories can be represented in Table 6.4.

Table 6.4 Examples of business rules that satisfy the support and confidence thresholds

Length of sequence	Rules	Support-count	Support	Confidence
3	{A}{B}{D}	5	0.416	0.714
3	{A}{B}<{E}>	3	0.250	0.428
3	{A}{B}<{F}>	3	0.250	0.428
3	{A}<{E}><{F}>	3	0.250	0.428
3	{A}<{E}><{D}>	3	0.250	0.428
3	{B}<{E}><{F}>	3	0.250	0.428
3	{B}<{E}><{D}>	3	0.250	0.428
3	{X}{Y}{W}	2	0.167	0.4
3	{X}{Y}<{M}> _{C1}	2	0.167	0.4
3	{X}{Y}<{N}> _{C1}	2	0.167	0.4
3	{X}{Y}<{D}>	2	0.167	0.4
3	{X}{Y}<{W}>	2	0.167	0.4
3	{X}<{M}> _{C1} <{N}> _{C1}	2	0.167	0.4
3	{Y}<{M}> _{C1} <{N}> _{C1}	2	0.167	0.4
4	{A}{B}<{E}><{F}>	2	0.167	0.286
4	{A}{B}<{E}><{D}>	2	0.167	0.286
4	{X}{Y}<{M}> _{C1} <{N}> _{C1}	2	0.167	0.4

Step 5: Evaluation of Business Rules

This step is to evaluate the precision of the business rules generated by the Apriori Algorithm. In this context, we apply the IR evaluation standard to evaluate our results. These techniques are *recall (R)* and *precision (P)*. Both of them are based on an understanding and measure of relevance. If recall score is 1.0, it indicates that the query returns all relevant documents. If precision score is 1.0, it means that the query is so precise to retrieve the relevant and correct documents.

In this context, precision and recall are applied for effectiveness measure. The term true positives, true negatives, false positives, and false negatives compare the results of the

business rule extraction under test with trusted external judgments (the expected business rules). The terms *positive* and *negative* refer to the business rule prediction (sometimes known as the *expectation*), and the terms *true* and *false* refer to whether that prediction corresponds to the external judgment (sometimes known as the *observation*). This is illustrated by the confusion matrix below:

Table 6.5 The confusion matrix

Judgements		Expert	
		Correct	Incorrect
System	Correct	True Positive (TP)	False Positive (FP)
	Incorrect	True Negative (TN)	False Negative (FN)

By using the confusion matrix (Kohavi and Provost, 1998) that contains information about the expected business rules and predicted business rules (actual business rules) done by the rule extractor system, precision and recall are then defined as:

$$Recall = \frac{TP}{TP + FN} \quad (7.1)$$

$$Precision = \frac{TP}{TP + FP} \quad (7.2)$$

Suppose there are 100 expected business rules. In that stage of business rule extraction, we extract 115 the actual business rules from a set of process models. It can be found that there are 100 actual business rules that are matched to the expected business rules provided by an expert. This is illustrated by the *confusion matrix* below:

Table 6.6 An example with the confusion matrix

Rule Extraction		Expert Judgments	
		Correct	Incorrect
Rule Extractor	Correct	TP = 100	FP = 15
	Incorrect	FN = 0	TN = 0

Consider all observations in Table 6.6. The recall and the precision of this evaluation can be resulted as follows.

$$Recall = 100 / (100 + 0) = 1.00$$

$$Precision = 100 / (100 + 15) = 0.87$$

The result of recall indicates that, the actual business rules can be matched to the expected business rules at 100%. Also, the performance of extracting business rule that are correct is 87% after testing by precision.

6.6 Dataset

This section illustrates the experiment of business rule extraction through ON-KDT process model. In order to obtain the confidence and reliable of the ON-KDT methodology, we experiment with two datasets. The first dataset contains the real-world process models. The second is the dataset of process models that is generated by a random process model generator. Both the real-world process models and the random process model generator are based on research being undertaken by the Decision Systems Laboratory (DSL), School of Computer Science and Software Engineering (SCSSE), University of Wollongong. The detail of the experiments can be elaborated as follows.

6.6.1 The experiment with real dataset of process models

The process models used in this experiment capture “*as is*” workflows. Each process was modelled in a series of interviews with various actors within a financial organization. Furthermore, each model has been created in text as well as at various abstraction layers using BPMN. There are 31 process models in the collection. Some of process models contain 4 Pools²⁵, which is a type of Swimlanes.

After the dataset is carefully considered, it can be found that directly extracting business rules from process models represented in graphical diagram such BPMN is a difficult task. Therefore, the collection of process models will be transformed into a textual format from which it is easier to extract knowledge. This work uses the XML Metadata Interchange (XMI)²⁶ standard, which is an Object Management Group (OMG) standard for exchanging metadata information through Extensible Markup Language (XML).

After understanding the dataset of process models, this dataset will be passed to the step of data preparation. This step may be affected with the problem of ambiguity of natural language, where XMI is now viewed as text. Afterwards, we will extract all of activity-sequences from the XML.

²⁵ Pool represents a participant in a process. It is also acts as a graphical container for partitioning a set of activities from other Pools. A Lane is a sub-partition within a Pool and will extend the entire length of the Pool, either vertically or horizontally. Lanes are used to organize and categorize activities. (<http://www.omg.org/bpmn/index.htm>)

²⁶ <http://www.omg.org/technology/documents/formal/xmi.htm>

In the real world, it is possible that a set of process models can be produced by many process designers. Sometimes, given name of tasks/activities may be different, although these tasks indicate to the same thing. It can be illustrated following. Consider two process models in Figure 6.18.

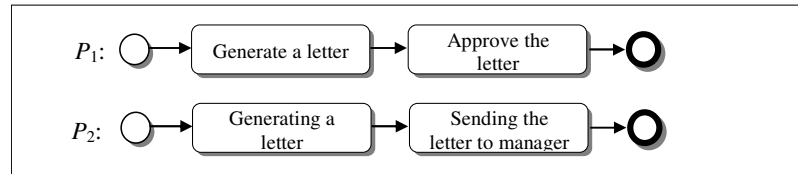


Figure 6.18 Two activity-sequence models have different names but their names are the same meaning

In Figure 6.18, the given name of the task ‘*Generate a letter*’ in P_1 and the given name of the task ‘*Generating a letter*’ in P_2 are different, if they are considered by an automatic analysis. In this case, these activities will be considered as the same activity by using the VL-ontology.

Furthermore, each real-world process model used in this context contains 4 Pools. Pool is a type of Swimlanes, which are used to describe the responsibility in process lines shown as horizontal bands. However, A Pool represents activities designated for a single participant in a process. Also, it can be a graphical container for partitioning a set of activities from other Pools (Miers and White, 2008). As this, it can be interpreted that among Pools may have association shown as vertical bands. Therefore, we consider extracting of business rules in two-dimensions of analysis: (1) *horizontal analysis* and (2) *vertical analysis* (shown as Figure 6.19).

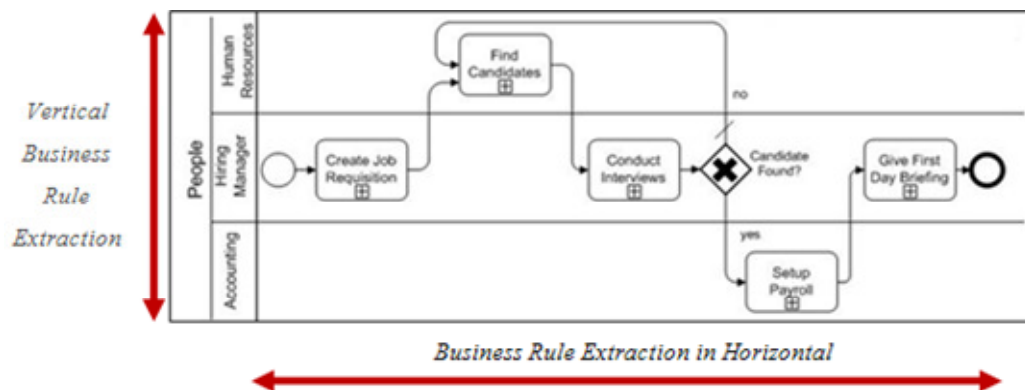


Figure 6.19 This presents the vertical and horizontal of business rule extraction

First, *the horizontal analysis*, it is to extract business rules from sub-process models that contain in pools. A process model will be separated n sub-process models, where n is a number of Pools. An example can be shown as Figure 6.19.

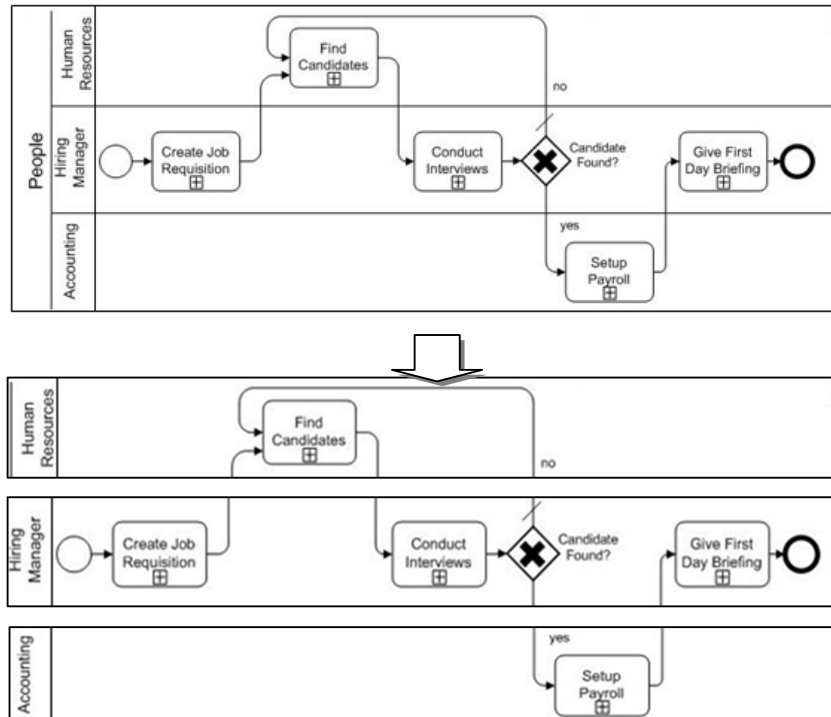


Figure 6.20 This presents an example of separating a process model into 3 sub-process models in order to prepare for business rules extraction in horizontal bands

Second, *the vertical analysis*, it is to extract business rules by considering the correlation among Lanes/Pools (shown as Figure 6.21). It is noted that a format to present the given name of each task is provided by manually. The format is '*pool-name.given-name*'. For example, the task named '*find candidates*' is stored in the Pool named '*Human resource*'. It can be re-written as *Human-resource.find-candidates*.

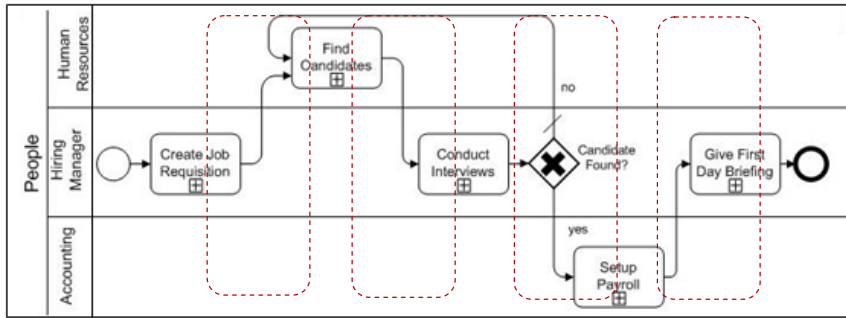


Figure 6.21 This presents a process model that is prepared for business rules extraction in vertical bands

(a) The experimental results of extracting business rules in the horizontal bands

As we said that the number of the original process model is 31. However, these process models can be generated to 325 sub-process models.. An example of the original process models can be shown as Figure 6.22.

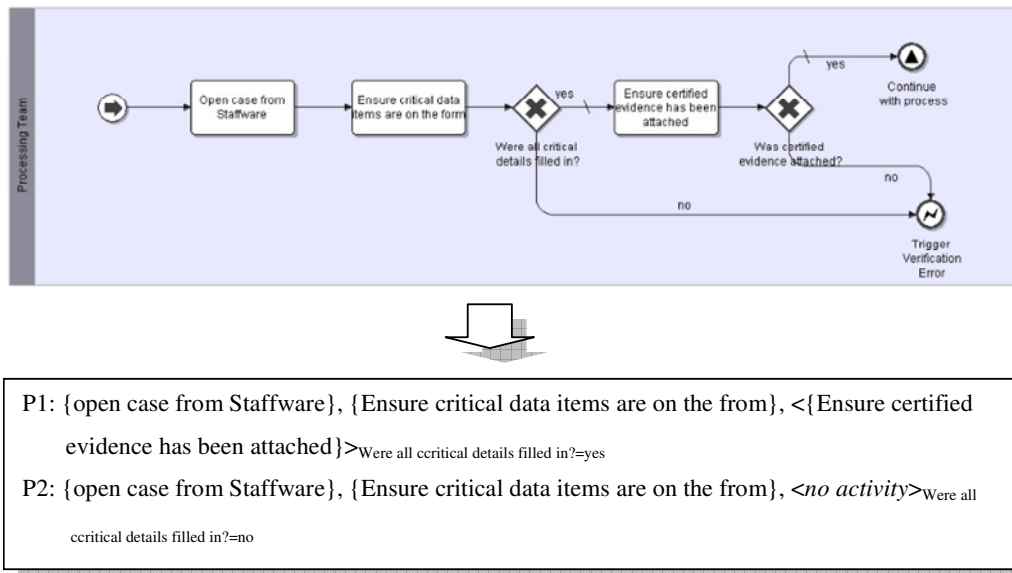
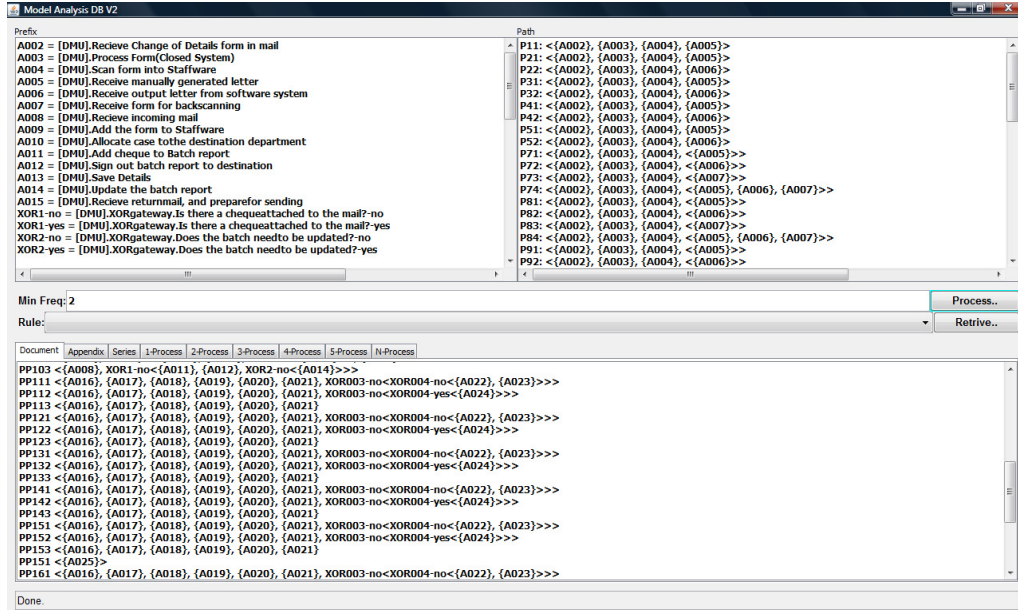


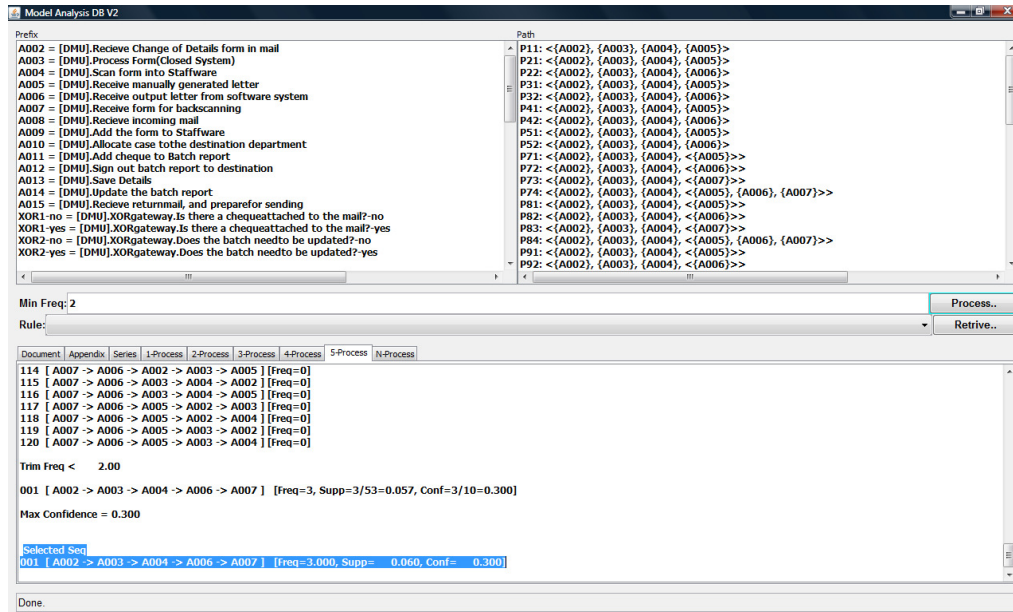
Figure 6.22 An example of the original process model generated into the form of activity-sequence

By using the modification of the GSP mining algorithm, we can extract the significant sequence of activities in the collection, considered as the business rules. It is noted that the min-support used in this experiment is 2. An experiment result can be shown

as Figure 6.23. This figure shows some of activities and conditions (based on gateways) and some of the activity-sequences that are generated from the original process models. In addition, it shows the longest sequence that can be extracted from the collection. This sequence satisfies the support and confidence thresholds and it is considered as a business rule.



(a) Some of activity-sequences in the experiment



(b) The extracted business rule that contains five activities in the sequence

Figure 6.23 An example of the experimental results

If we consider only business rules that contain activity at least 4 activities, we will obtain 8 business rules. It is noted that there is no exact rule, because it has no any business rule having the confidence threshold as 1.00. In this context, the extracted business rules will be evaluated by considering the impact value. Each business rule will be used to retrieve and count the number of process models that contain this business rule. After evaluating all of business rules, the results are presented in Table 6.7.

Table 6.7 The evaluation of business rules that are extracted from process models

Business rules	Recall (%)	Precision (%)
1	35.48	100
2	32.26	100
3	35.48	100
4	32.26	100
5	35.48	100
6	32.26	100
7	32.26	100
8	19.35	100
Average	31.85	100

The results in Table 6.7 show that the average of recall is 31.85% and the average of precision is 100%. In terms of precision score, it means these business rules may be the key activities or critical roles that can reach the success. This is because many process models in the test set contain these business rules extracted. In the area of business process management, these business rules are very important, especially for process model re-design.

(b) The experimental results of extracting business rules in the vertical bands

In this case, after extracting business rules by the modification of the GSP mining algorithm, we obtain ten business rules, when we use the min-support as 2. These rules present the relationship between pools. An example can be described in Figure 6.23.

In Figure 6.24, it can be seen that the pools ‘DMU’ and ‘Staffware’ can have the relationship with the connection between the activities ‘DMU.Scan form in Staffware’ and ‘Staffware’. Form scanned into the workflow system’.

It is noted that there is no exact rule, because it has no any business rule having the confidence threshold as 1.00. Furthermore, each rule contains only two activities.

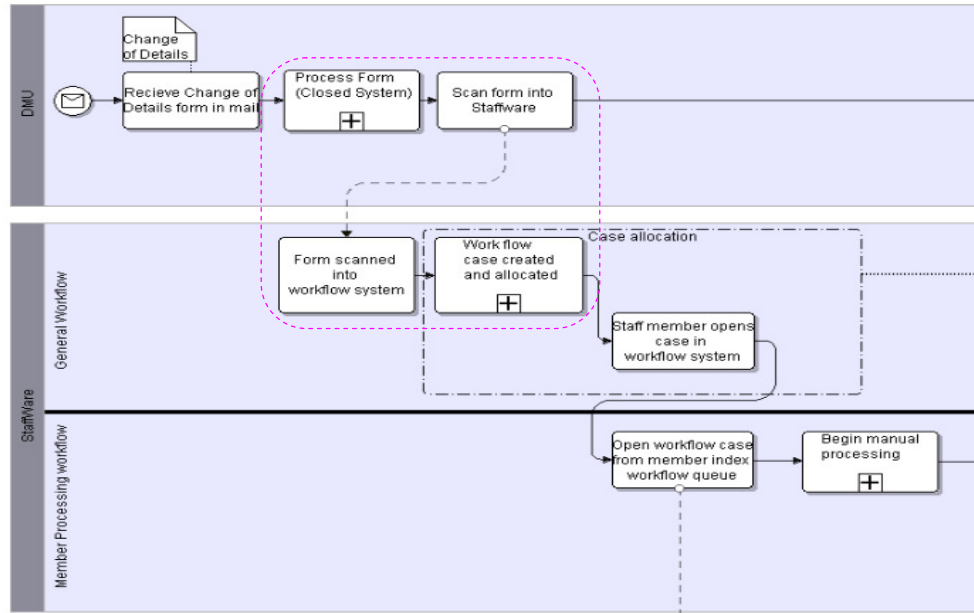


Figure 6.24 A business rules that is extracted from the process presents the relationship between pools

6.6.2 The experiment with random process models generated

In order to get the confidence of the On-KDT, we also experiment this methodology with the large dataset of process models, where we have created large sets of random process models.

Firstly, we have created a set of business rules, called *the expected business rules*. The dataset consist of 20 expected business rules and is used as a benchmark for our evaluation. All of expected business rules can be as shown in Table 6.8.

Table 6.8 The expected business rules

List	Expected business rules
1	$a_0 \rightarrow a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow a_4$
2	$a_6 \rightarrow a_7 \rightarrow XOR-gateway_2 \rightarrow a_8 \rightarrow a_{10}$
3	$a_6 \rightarrow a_7 \rightarrow XOR-gateway_2 \rightarrow a_9 \rightarrow a_{10}$
4	$b_3 \rightarrow b_4 \rightarrow b_5 \rightarrow b_6 \rightarrow b_7$
5	$b_{11} \rightarrow b_{12} \rightarrow XOR-gateway_3 \rightarrow b_{13} \rightarrow b_{14}$

Table 6.8 (cont')

List	Expected business rules
6	$b_{11} \rightarrow b_{12} \rightarrow XOR\text{-gateway}_3 \rightarrow b_{15} \rightarrow b_{16}$
7	$c_1 \rightarrow c_2 \rightarrow c_3 \rightarrow c_4 \rightarrow c_5$
8	$d9 \rightarrow d10 \rightarrow d11 \rightarrow d12 \rightarrow d13$
9	$h_4 \rightarrow h_5 \rightarrow h_6 \rightarrow XOR\text{-gateway}_1 \rightarrow h_7$
10	$h_4 \rightarrow h_5 \rightarrow h_6 \rightarrow XOR\text{-gateway}_1 \rightarrow h_8$
11	$f_4 \rightarrow f_5 \rightarrow f_6 \rightarrow f_7 \rightarrow f_8$
12	$f_{11} \rightarrow XOR\text{-gateway}_4 \rightarrow f_{15} \rightarrow f_{16} \rightarrow f_{17}$
13	$f_{11} \rightarrow XOR\text{-gateway}_4 \rightarrow f_{12} \rightarrow f_{13} \rightarrow f_{17}$
14	$m_1 \rightarrow m_2 \rightarrow m_3 \rightarrow m_4 \rightarrow m_5$
15	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_3$
16	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_4$
17	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_5$
18	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_3$
19	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_4$
20	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_5$

We generated three datasets that contain 500, 1000, 1500, and 2000 random process models, respectively. We then applied our technique to extract the significant longest sequence of activities in the collection, called *the actual business rules*. Also, we used two values of the min-support as 2 and 4, respectively. The experimental results can be shown as follows.

(a) The Experimental results with the use of min-support as 2

The first case is to experiment with the min-support as 2. We test with three datasets that contain 400, 800, and 1200 random process models, respectively. Also, we evaluate the performance by checking time used.

(i) The experiment with 400 random process models

By using with 400 random process models, there are 20 actual business rules that are extracted by the business rule extraction engine and running time is 50.76 seconds. The experiment results showing with their support and confidence can be as shown in Table 6.9.

Table 6.9 The Experimental results with 400 random process models

List	Actual-set of Business Rules	Support	Confidence
1	$a_0 \rightarrow a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow a_4$	0.2	0.6
2	$a_6 \rightarrow a_7 \rightarrow XOR\text{-gateway}_2 \rightarrow a_8 \rightarrow a_{10}$	0.2	0.6
3	$a_6 \rightarrow a_7 \rightarrow XOR\text{-gateway}_2 \rightarrow a_9 \rightarrow a_{10}$	0.2	0.6
4	$b_3 \rightarrow b_4 \rightarrow b_5 \rightarrow b_6 \rightarrow b_7$	0.2	0.5
5	$b_{11} \rightarrow b_{12} \rightarrow XOR\text{-gateway}_3 \rightarrow b_{13} \rightarrow b_{14}$	0.2	0.5
6	$b_{11} \rightarrow b_{12} \rightarrow XOR\text{-gateway}_3 \rightarrow b_{15} \rightarrow b_{16}$	0.2	0.5
7	$c_1 \rightarrow c_2 \rightarrow c_3 \rightarrow c_4 \rightarrow c_5$	0.2	0.4
8	$d_9 \rightarrow d_{10} \rightarrow d_{11} \rightarrow d_{12} \rightarrow d_{13}$	0.2	0.4
9	$h_4 \rightarrow h_5 \rightarrow h_6 \rightarrow XOR\text{-gateway}_1 \rightarrow h_7$	0.2	0.6
10	$h_4 \rightarrow h_5 \rightarrow h_6 \rightarrow XOR\text{-gateway}_1 \rightarrow h_8$	0.2	0.6
11	$f_4 \rightarrow f_5 \rightarrow f_6 \rightarrow f_7 \rightarrow f_8$	0.2	0.6
12	$f_{11} \rightarrow XOR\text{-gateway}_4 \rightarrow f_{15} \rightarrow f_{16} \rightarrow f_{17}$	0.2	0.6
13	$f_{11} \rightarrow XOR\text{-gateway}_4 \rightarrow f_{12} \rightarrow f_{13} \rightarrow f_{17}$	0.2	0.6
14	$m_1 \rightarrow m_2 \rightarrow m_3 \rightarrow m_4 \rightarrow m_5$	0.2	0.5
15	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_3$	0.2	0.6
16	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_4$	0.2	0.6
17	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_5$	0.2	0.6
18	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_3$	0.2	0.6
19	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_4$	0.2	0.6
20	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_5$	0.2	0.6

(ii) The experiment with 800 random process models

By using with 800 random process models, there are 20 actual business rules that are extracted by the business rule extraction engine and running time is 110.20 seconds. The experiment results showing with their support and confidence can be as shown in Table 6.10.

Table 6.10 The Experimental results with 800 random process models

List	Actual-set of Business Rules	Support	Confidence
1	$a_0 \rightarrow a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow a_4$	0.2	0.6
2	$a_6 \rightarrow a_7 \rightarrow XOR\text{-gateway}_2 \rightarrow a_8 \rightarrow a_{10}$	0.2	0.6
3	$a_6 \rightarrow a_7 \rightarrow XOR\text{-gateway}_2 \rightarrow a_9 \rightarrow a_{10}$	0.2	0.6
4	$b_3 \rightarrow b_4 \rightarrow b_5 \rightarrow b_6 \rightarrow b_7$	0.2	0.5
5	$b_{11} \rightarrow b_{12} \rightarrow XOR\text{-gateway}_3 \rightarrow b_{13} \rightarrow b_{14}$	0.2	0.5
6	$b_{11} \rightarrow b_{12} \rightarrow XOR\text{-gateway}_3 \rightarrow b_{15} \rightarrow b_{16}$	0.2	0.5
7	$c_1 \rightarrow c_2 \rightarrow c_3 \rightarrow c_4 \rightarrow c_5$	0.2	0.4
8	$d_9 \rightarrow d_{10} \rightarrow d_{11} \rightarrow d_{12} \rightarrow d_{13}$	0.2	0.4
9	$h_4 \rightarrow h_5 \rightarrow h_6 \rightarrow XOR\text{-gateway}_1 \rightarrow h_7$	0.2	0.6
10	$h_4 \rightarrow h_5 \rightarrow h_6 \rightarrow XOR\text{-gateway}_1 \rightarrow h_8$	0.2	0.6
11	$f_4 \rightarrow f_5 \rightarrow f_6 \rightarrow f_7 \rightarrow f_8$	0.2	0.6
12	$f_{11} \rightarrow XOR\text{-gateway}_4 \rightarrow f_{15} \rightarrow f_{16} \rightarrow f_{17}$	0.2	0.6
13	$f_{11} \rightarrow XOR\text{-gateway}_4 \rightarrow f_{12} \rightarrow f_{13} \rightarrow f_{17}$	0.2	0.6
14	$m_1 \rightarrow m_2 \rightarrow m_3 \rightarrow m_4 \rightarrow m_5$	0.2	0.5
15	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_3$	0.2	0.6
16	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_4$	0.2	0.6
17	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_5$	0.2	0.6
18	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_3$	0.2	0.6
19	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_4$	0.2	0.6
20	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_5$	0.2	0.6

(iii) The experiment with 1200 random process models

By using with 1200 random process models, there are 20 actual business rules that are extracted by the business rule extraction engine and running time is 190.20 seconds. The experiment results showing with their support and confidence can be as shown in Table 6.11.

Table 6.11 The Experimental results with 1200 random process models

List	Actual-set of Business Rules	Support	Confidence
1	$a_0 \rightarrow a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow a_4$	0.2	0.6
2	$a_6 \rightarrow a_7 \rightarrow XOR\text{-gateway}_2 \rightarrow a_8 \rightarrow a_{10}$	0.2	0.6
3	$a_6 \rightarrow a_7 \rightarrow XOR\text{-gateway}_2 \rightarrow a_9 \rightarrow a_{10}$	0.2	0.6
4	$b_3 \rightarrow b_4 \rightarrow b_5 \rightarrow b_6 \rightarrow b_7$	0.2	0.5
5	$b_{11} \rightarrow b_{12} \rightarrow XOR\text{-gateway}_3 \rightarrow b_{13} \rightarrow b_{14}$	0.2	0.5
6	$b_{11} \rightarrow b_{12} \rightarrow XOR\text{-gateway}_3 \rightarrow b_{15} \rightarrow b_{16}$	0.2	0.5
7	$c_1 \rightarrow c_2 \rightarrow c_3 \rightarrow c_4 \rightarrow c_5$	0.2	0.4
8	$d_9 \rightarrow d_{10} \rightarrow d_{11} \rightarrow d_{12} \rightarrow d_{13}$	0.2	0.4
9	$h_4 \rightarrow h_5 \rightarrow h_6 \rightarrow XOR\text{-gateway}_1 \rightarrow h_7$	0.2	0.6
10	$h_4 \rightarrow h_5 \rightarrow h_6 \rightarrow XOR\text{-gateway}_1 \rightarrow h_8$	0.2	0.6
11	$f_4 \rightarrow f_5 \rightarrow f_6 \rightarrow f_7 \rightarrow f_8$	0.2	0.6
12	$f_{11} \rightarrow XOR\text{-gateway}_4 \rightarrow f_{15} \rightarrow f_{16} \rightarrow f_{17}$	0.2	0.6
13	$f_{11} \rightarrow XOR\text{-gateway}_4 \rightarrow f_{12} \rightarrow f_{13} \rightarrow f_{17}$	0.2	0.6
14	$m_1 \rightarrow m_2 \rightarrow m_3 \rightarrow m_4 \rightarrow m_5$	0.2	0.5
15	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_3$	0.2	0.6
16	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_4$	0.2	0.6
17	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_5$	0.2	0.6
18	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_3$	0.2	0.5
19	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_4$	0.2	0.5
20	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_5$	0.2	0.5

(b) The Experimental results with the use of min-support as 4

The second case is to experiment with the min-support as 4. We test with three datasets that contain 400, 800, and 1200 random process models, respectively. Also, we evaluate the performance by checking time used.

(i) The experiment with 400 random process models

By using with 400 random process models, there are 20 actual business rules that are extracted by the business rule extraction engine and running time is 60.16 seconds. The experiment results showing with their support and confidence can be as shown in Table 6.12.

Table 6.12 The Experimental results with 400 random process models

List	Actual-set of Business Rules	Support	Confidence
1	$a_0 \rightarrow a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow a_4$	0.4	0.6
2	$a_6 \rightarrow a_7 \rightarrow XOR\text{-gateway}_2 \rightarrow a_8 \rightarrow a_{10}$	0.4	0.6
3	$a_6 \rightarrow a_7 \rightarrow XOR\text{-gateway}_2 \rightarrow a_9 \rightarrow a_{10}$	0.4	0.6
4	$b_3 \rightarrow b_4 \rightarrow b_5 \rightarrow b_6 \rightarrow b_7$	0.4	0.5
5	$b_{11} \rightarrow b_{12} \rightarrow XOR\text{-gateway}_3 \rightarrow b_{13} \rightarrow b_{14}$	0.4	0.5
6	$b_{11} \rightarrow b_{12} \rightarrow XOR\text{-gateway}_3 \rightarrow b_{15} \rightarrow b_{16}$	0.4	0.5
7	$c_1 \rightarrow c_2 \rightarrow c_3 \rightarrow c_4 \rightarrow c_5$	0.4	0.4
8	$d_9 \rightarrow d_{10} \rightarrow d_{11} \rightarrow d_{12} \rightarrow d_{13}$	0.4	0.4
9	$h_4 \rightarrow h_5 \rightarrow h_6 \rightarrow XOR\text{-gateway}_1 \rightarrow h_7$	0.4	0.6
10	$h_4 \rightarrow h_5 \rightarrow h_6 \rightarrow XOR\text{-gateway}_1 \rightarrow h_8$	0.4	0.6
11	$f_4 \rightarrow f_5 \rightarrow f_6 \rightarrow f_7 \rightarrow f_8$	0.4	0.6
12	$f_{11} \rightarrow XOR\text{-gateway}_4 \rightarrow f_{15} \rightarrow f_{16} \rightarrow f_{17}$	0.4	0.6
13	$f_{11} \rightarrow XOR\text{-gateway}_4 \rightarrow f_{12} \rightarrow f_{13} \rightarrow f_{17}$	0.4	0.6
14	$m_1 \rightarrow m_2 \rightarrow m_3 \rightarrow m_4 \rightarrow m_5$	0.4	0.5
15	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_3$	0.4	0.6
16	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_4$	0.4	0.6
17	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_5$	0.4	0.6
18	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_3$	0.4	0.6
19	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_4$	0.4	0.6
20	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_5$	0.4	0.6

(ii) The experiment with 800 random process models

By using with 800 random process models, there are 20 actual business rules that are extracted by the business rule extraction engine and its running time is 124.32 seconds. The experiment results showing with their support and confidence can be as shown in Table 6.13.

Table 6.13 The Experimental results with 800 random process models

List	Actual-set of Business Rules	Support	Confidence
1	$a_0 \rightarrow a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow a_4$	0.4	0.4
2	$a_6 \rightarrow a_7 \rightarrow XOR\text{-gateway}_2 \rightarrow a_8 \rightarrow a_{10}$	0.4	0.6
3	$a_6 \rightarrow a_7 \rightarrow XOR\text{-gateway}_2 \rightarrow a_9 \rightarrow a_{10}$	0.4	0.6
4	$b_3 \rightarrow b_4 \rightarrow b_5 \rightarrow b_6 \rightarrow b_7$	0.4	0.4
5	$b_{11} \rightarrow b_{12} \rightarrow XOR\text{-gateway}_3 \rightarrow b_{13} \rightarrow b_{14}$	0.4	0.4
6	$b_{11} \rightarrow b_{12} \rightarrow XOR\text{-gateway}_3 \rightarrow b_{15} \rightarrow b_{16}$	0.4	0.4
7	$c_1 \rightarrow c_2 \rightarrow c_3 \rightarrow c_4 \rightarrow c_5$	0.4	0.6
8	$d_9 \rightarrow d_{10} \rightarrow d_{11} \rightarrow d_{12} \rightarrow d_{13}$	0.4	0.6
9	$h_4 \rightarrow h_5 \rightarrow h_6 \rightarrow XOR\text{-gateway}_1 \rightarrow h_7$	0.4	0.6
10	$h_4 \rightarrow h_5 \rightarrow h_6 \rightarrow XOR\text{-gateway}_1 \rightarrow h_8$	0.4	0.6
11	$f_4 \rightarrow f_5 \rightarrow f_6 \rightarrow f_7 \rightarrow f_8$	0.4	0.4
12	$f_{11} \rightarrow XOR\text{-gateway}_4 \rightarrow f_{15} \rightarrow f_{16} \rightarrow f_{17}$	0.4	0.4
13	$f_{11} \rightarrow XOR\text{-gateway}_4 \rightarrow f_{12} \rightarrow f_{13} \rightarrow f_{17}$	0.4	0.4
14	$m_1 \rightarrow m_2 \rightarrow m_3 \rightarrow m_4 \rightarrow m_5$	0.4	0.5
15	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_3$	0.4	0.6
16	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_4$	0.4	0.6
17	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_5$	0.4	0.6
18	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_3$	0.4	0.6
19	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_4$	0.4	0.6
20	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_5$	0.4	0.6

(iii) The experiment with 1200 random process models

By using with 1200 random process models, there are 20 actual business rules that are extracted by the business rule extraction engine and running time is 200.02 seconds. The experiment results showing with their support and confidence can be as shown in Table 6.14.

Table 6.14 The Experimental results with 1200 random process models

List	Actual-set of Business Rules	Support	Confidence
1	$a_0 \rightarrow a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow a_4$	0.4	0.4
2	$a_6 \rightarrow a_7 \rightarrow XOR\text{-gateway}_2 \rightarrow a_8 \rightarrow a_{10}$	0.4	0.6
3	$a_6 \rightarrow a_7 \rightarrow XOR\text{-gateway}_2 \rightarrow a_9 \rightarrow a_{10}$	0.4	0.6
4	$b_3 \rightarrow b_4 \rightarrow b_5 \rightarrow b_6 \rightarrow b_7$	0.4	0.4
5	$b_{11} \rightarrow b_{12} \rightarrow XOR\text{-gateway}_3 \rightarrow b_{13} \rightarrow b_{14}$	0.4	0.4
6	$b_{11} \rightarrow b_{12} \rightarrow XOR\text{-gateway}_3 \rightarrow b_{15} \rightarrow b_{16}$	0.4	0.4
7	$c_1 \rightarrow c_2 \rightarrow c_3 \rightarrow c_4 \rightarrow c_5$	0.4	0.6
8	$d_9 \rightarrow d_{10} \rightarrow d_{11} \rightarrow d_{12} \rightarrow d_{13}$	0.4	0.6
9	$h_4 \rightarrow h_5 \rightarrow h_6 \rightarrow XOR\text{-gateway}_1 \rightarrow h_7$	0.4	0.6
10	$h_4 \rightarrow h_5 \rightarrow h_6 \rightarrow XOR\text{-gateway}_1 \rightarrow h_8$	0.4	0.6
11	$f_4 \rightarrow f_5 \rightarrow f_6 \rightarrow f_7 \rightarrow f_8$	0.4	0.4
12	$f_{11} \rightarrow XOR\text{-gateway}_4 \rightarrow f_{15} \rightarrow f_{16} \rightarrow f_{17}$	0.4	0.4
13	$f_{11} \rightarrow XOR\text{-gateway}_4 \rightarrow f_{12} \rightarrow f_{13} \rightarrow f_{17}$	0.4	0.4
14	$m_1 \rightarrow m_2 \rightarrow m_3 \rightarrow m_4 \rightarrow m_5$	0.4	0.5
15	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_3$	0.4	0.6
16	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_4$	0.4	0.6
17	$w_0 \rightarrow w_1 \rightarrow w_2 \rightarrow XOR\text{-gateway}_5 \rightarrow w_5$	0.4	0.6
18	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_3$	0.4	0.6
19	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_4$	0.4	0.6
20	$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow XOR\text{-gateway}_6 \rightarrow x_5$	0.4	0.6

(c) Summary of the experimental results

Consider the results of business rule extraction under the traditional concept of the Association Rule Mining, it can be found that no any exact rule is extracted because there is no any business rule having the confidence threshold as 1.00. We therefore apply a confidence threshold of 0.6 and a min-support of 2 and 4. Table 6.16 shows recall and precision of the business rules extracted against the expect rules.

As can be seen from Table 6.16, all the business rules that were extracted by our technique are expected (precision is 1 for all cases) and recalls are the same for both min-support of 2 and 4. These results have demonstrated the effectiveness of our approach in terms of extracting the correct business rules.

Table 6.15 The experimental results of of process model retrieval by using the actual rules

Min-support	Number of random process models	Recall (%)	Precision (%)	Time used (seconds)
2	400	71.64	100	50.76
	800	71.37	100	110.20
	1200	72.29	100	190.20
4	400	71.64	100	60.16
	800	71.37	100	124.32
	1200	72.29	100	200.32
Average of performance		71.77	100	122.66

6.7 Chapter Conclusion

It is well-known that process mining is the research area that mainly focuses on the derivation of process models from event logs. Process mining provides the capability to discover, detect, control, organize, and monitor actual process execution by extracting useful knowledge from event logs (i.e. process instances). However, if a process has never been performed, no event logs information is available, rendering the technique useless, there are settings in which process mining proves to be inadequate. Furthermore, if the size of the set of event logs is small (which is often the case with infrequently executed processes, such as those associated with disaster management, or with Greenfield applications), the reliability of the knowledge extracted cannot be guaranteed.

As the problem relating to the problem of dataset, we explore an alternative dimension to process mining in which the objective is to extract process constraints (or business rules) as opposed to process models. It also focuses on an alternative dataset - process models as opposed to event logs. In this context, we use process models as the resource of prior knowledge. This approach is valuable in the settings discussed above, in which process mining techniques do not return reliable results. The expected results of mining process model repositories are business rules, where a business rule defines one aspect of IT system that is intended to assert IT system structure or influence the behaviour of IT system. Such business rules represent business constraints that have been encoded in a process model. Therefore, this chapter describes a contribution to the unified methodology of the ON-KDT to extract business rules from process model repositories, where we view these data as text.

CHAPTER 7

CONCLUSION

7.1 Summary of Background and Motivation of the Thesis

The problem of information overload has occurred, where advance in computer and information technology has facilitated a significant increase in an enormous amount of digital data and information because of the inexpensive repository and accessibility that are available. As the result, useful knowledge is difficult to locate out of an abundance of data or information in a short period of time. In order to utilize these data efficiently and effectively, the modern concept of automatically finding knowledge was proposed to support the need of extracting useful knowledge from a collective data or information through computation approach. It has become a challenging task, which is called *knowledge discovery in database (KDD)*. Since the 1960s, the need of obtaining knowledge has made this task more critical than ever and it has grown in many fields.

Traditionally, the way of obtaining knowledge involves complex cognitive processes such as perception, learning, communication, association and reasoning. At the beginning, finding knowledge from the wealth of data collection was done by the manual extraction method. Early methods of identifying knowledge in data were done by handcrafted data analysis relied on statistical analysis and inference (Elder and Pregibon, 1996) such as Bayes' theorem in the 1700s and predictive regression analysis in the 1800s.

However, with the rapid growth in size and complexity of data in the real world, this has provided a challenge for automated knowledge acquisition (Fayyad et al., 1996a, Fayyad et al., 1996b, Jin et al., 1998, Fayyad, 2000). The methods of obtaining knowledge were like switching from a handcrafted data analysis environment to an automatic data processing. Many solutions of obtaining knowledge have been proposed (Fayyad et al., 1996a, Fayyad et al., 1996b, Fayyad, 2000, Jin et al., 1998). The process of generating useful knowledge from adequate data is called '*knowledge extraction*' (Rao and Sprague, 1999, Yao et al., 2007).

Knowledge extraction can be defined as ways of acquiring hidden knowledge from structured (e.g. Relational databases, XML) or unstructured (e.g. Text, documents, images) sources through a computational approach (Rao and Sprague, 1999, Yao et al., 2007). It involves a variety of techniques that combine the expertise of both humans and machines (Fayyad et al., 1996a, Fayyad et al., 1996b), while the extracted knowledge can be represented in a variety of representations such as rules, attributes, structures, and patterns (Rao and Sprague, 1999). In general, knowledge extraction is computed and evaluated

manually using statistical techniques (Asghar and Iqbal, 2009). Initially, traditional efforts in knowledge acquisition arose largely in the area of Information Extraction (IE) (Banko and Etzioni, 2007), where this task is to recognise entities and relations mentioned within bodies of text (Banko and Etzioni, 2007). It is remarked that the word ‘*knowledge extraction*’ is closely related to ‘*knowledge discovery*’. This is because the knowledge discovery also describes the process of automatically searching large volumes of data for patterns that can be considered as knowledge about the data. It is the most well-known branch of data mining, also known as *Knowledge Discovery in Databases (KDD)* (Fayyad et al., 1996a, Fayyad et al., 1996b).

The KDD methodology is the context for development of computational techniques and tools that are required to support and handle the finding useful knowledge from the rapidly growing volumes of data. Fayyad *et al.* (1996a) defined the term of KDD as “*the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data*”, while Friedman (1997) considered the KDD methodology as “*an automatic exploratory data analysis of large database*”. There are several researchers such as Brachman & Anand (1994), Fayyad *et al.* (1996a), Maimon & Last (2005), Reinartz (2002), and Brachman and Anand (1994), that have proposed different ways to divide the KDD methodology into phases.

In general, the model of KDD is presented informally, by describing some practical needs of users (Devedzic, 2001). The KDD process combines techniques from a variety of related disciplines, Databases, Statistics, Scientific Discovery, Artificial Intelligence, and visualization (Williams and Huang, 1996). By the classic definition of KDD, the KDD process is “*nontrivial and iterative of extracting useful information and knowledge from a large database*” (Fayyad et al., 1996a, Fayyad et al., 1996b). It consists of multiple processing steps (one of them is data mining). Each step attempts to accomplish a particular discovery task and is completed by the application of a discovery method.

Since the 1990s, many different KDD models have been developed and proposed. The development of the KDD model was initiated by academic research, but it was quickly followed by industry (Fayyad et al., 1996a, Cabena et al., 1998, Anand and Buchner, 1998). Later, a combination of both models has been presented as a hybrid model in the early 2000s (Chapman et al., 2000).

In academic research, the popular model representations include the KDD process models were developed and proposed by several researchers since in the mid-1990s. Fayyad *et al.* (1996b, 1996a) proposed the classic process of finding useful knowledge in data is interactive and iterative. The overview of finding useful knowledge in data can be

summarized to five common steps: (1) Data Selection, (2) Data pre-processing, (3) Data transformation, (4) Data Mining, and (5) Interpretation/Evaluation.

The model proposed by Fayyad *et al.* (1996b, 1996a) is not a methodology because it did not define how to do each of proposed tasks. Although it presents only a set of components in the KDD process model, it is the most popular KDD process model. For example, this KDD process model has been incorporated into a commercial KDD system, which is called MineSet released in 1998 (Referred to <http://www.the-data-mine.com/Software/MineSet>). Later, many models were proposed by modifying the previous model with a few changes. However, all KDD process models have similar features.

Subsequently, the growth of the attention paid to industrial area, where the development of the KDD model has been interested and adopted by industry. Some efforts are being done and established in this area. Initially, the model proposed by Cabena *et al.* (1998) was developed with the support of IBM (Kurgan and Musilek, 2006), and then there are many applications have been proposed and used in practice such as SEMMA and CRISP-DM. The SEMMA (Sample, Explore, Modify, Model, Assess) was developed by the SAS Institute.

The next industrial model is the CRISP-DM (Cross-Industry Standard Process for Data Mining). This model was developed with the goal of the support of the process model developed (Chapman *et al.*, 2000). This model consists of six steps: (1) business understanding, (2) data understanding, (3) data preparation, (4) modelling, (5) evaluation, and (6) deployment. The CRISP-DM model has been used in several domains such as marketing and sales. Furthermore, it has been incorporated into a commercial knowledge discovery system called Chementine (see SPSS Inc. <http://www.spss.com/chementine>).

To the best of our knowledge, many works in the KDD have been focused on analysing structured data sources, such as spreadsheet, relational databases and data tables (Rajman and Besançon, 1997, Soibeman *et al.*, 2008, Mooney and Bunescu, 2005). Unfortunately, a large body of data in the real world is available as complex unstructured data sources, such as text, digital images, and web pages (Sanchez *et al.*, 2008, Soibeman *et al.*, 2008). Unstructured data is any data without a well-defined model or schema for accessing information. These data sources cannot contain in any specified structure (Soibeman *et al.*, 2008). This is because,

(1) With the large number of features of unstructured data, this is difficult to store and manage related data, such as text containing many words, and images containing thousand pixels. Therefore, it is difficult to keep these data sources in the specified structure.

(2) Some elements contained in the original data format may be lost during transforming the original data to a specified structure. For example, transforming images into a specified structure (e.g. single data table, spread sheet) lead to the missing-data problem, where texture information of images may be lost.

In the previous study, a number of existing researches for unstructured data are less studied when comparing with a number of existing researches studied for in structured data (Sanchez et al., 2008, Soibeman et al., 2008, Rajman and Besançon, 1997). This is because analysing of unstructured data needs to be aware of the complexity of the data (Wood and Ross-Kerr, 2006). This has become a major challenge in the area of knowledge extraction from unstructured data, where the study of finding knowledge in kind of unstructured data is still recognized as one of the major unsolved problems (Manuja and Garg, 2011).

Analysing of unstructured data, i.e. textual data, need to be aware of the complexity of the data (Wood and Ross-Kerr, 2006). This is because utilising text data are subject to content-based analysis of natural language. It should be extremely careful in every step of the analysis of data such text because the degree of ambiguity in natural language is too rigorous (Zhang and Nunamaker, 2004). This makes it extremely hard for a computer to automatically extract useful knowledge from text repositories. Furthermore, it is time-consuming and labour-intensive (Pandey & Daga 2004), especially when working on a large number of text documents and each document may contain a large volume of information.

In addition, unstructured textual data analysis faces not only the problem of natural language ambiguity, but also the problem of lacking a standard model to discover useful knowledge from text repositories. As this, it leads to the first motivation in this thesis, where we need to propose a unified model of knowledge discovery from unstructured /semi-structured data.

However, the problem of ambiguity in text has long been a challenge in text processing such as natural language processing (NLP) and information retrieval (IR) (Goldberg and Elhadad, 2009). Therefore, we also need a semantic tool that can be used to support and improve the domain of finding a useful knowledge/pattern from unstructured/semi-structured textual data. This leads to the second motivation in this thesis, where we also need to propose a semantic mechanism that will be contained in the knowledge discovery methodology. This mechanism can be used to support for finding knowledge from unstructured/semi-structured data.

7.2 Summary of the Proposed Methodology of the Thesis

As we have said in Section 7.1 that unstructured/semi-structured data, such textual data, is any data without a well-defined model or schema for accessing information. Finding a useful pattern/knowledge from unstructured/semi-structured data (e.g. textual data) are a major unsolved problem in the area of KDD. The problem becomes particularly acute due to ambiguity in natural language. Therefore, this thesis seeks to address this problem. It aims to propose a novel conceptual methodology that can be used as a guide for finding useful knowledge from unstructured/semi-structured data, where they are viewed as text.

As the motivation above, we aim to design a conceptual methodology of finding useful knowledge from informal data described with text. An ontology will be integrated to the proposed methodology in order to reduce the ambiguity in text. Also, this ontology helps to improve the search accuracy for knowledge from text by understanding the searchers' intent and the contextual meaning of terms as they appear in the searchable data-space. Our methodology is called the *Ontology-based Knowledge Discovery in unstructured/semi-structured Text* (On-KDT). In this context, the KDD model proposed by Fayyad *et al.* (1996) is used as the cornerstone of the On-KDT methodology because it is recognised as the leading academic research model which has become a foundation of the later models (Cios *et al.* 2007).

The On-KDT methodology consists of six tasks: understanding of the application domain & defining problem, understanding data, data preparation, closed-domain knowledge extraction, evaluation & interpretation, and using discovered knowledge.

The On-KDT methodology commences with two main manual tasks. The first task is to determine the application domain. It includes defining the problem and the goals of the end-user. This task is important, where it identifies target data and determines a plan of knowledge discovery. In general, the task of knowledge discovery is concerned with data mining techniques and tools. After we understand the objectives clearly and make sure to find out what the end-users really want to achieve, the dataset will be collected from available sources that are familiar and relevant to the specific problem domain. The second task is to collect an initial data and proceed with activities in order to make more understanding the target data and identify data quality problems.

Consequently, the data preparation is the process of manipulating data into a new form that is appropriate for further analysis and processing. Basically, there are several techniques for data preparation for data mining such as data selection, data cleaning, and data transformation. However, in this context, the data preparation task comprises two main processes: (1) *pre-processing data* and (2) *identifying more specific individual data*. Firstly,

pre-processing the data includes basic operations for data cleaning such as removing noise, handling missing data fields, and collecting the necessary information to the model. Also, it transforms the specific data used in a new form that will be more easily and effectively processed to meet the needs of the user. The second main process identifies more specific individual data driven by the concept of data partition (classification and clustering), where the ability of acquiring knowledge depends on something more specific than broad, unspecified knowledge. After text dataset is re-organized to more specific individuals, the outputs are passed to the task of knowledge extraction in order to acquire accumulated knowledge from textual data repositories. As the data are identified to a specific individual data before passing the data to knowledge extraction task, this task is called *closed-domain knowledge extraction*.

After acquiring knowledge from unstructured/semi-structured data, the next step is to evaluate the patterns/knowledge discovered to make the decision of what qualifies as knowledge. Finally, it is the step of incorporating and taking action on this knowledge into the performance system.

However, with the problem of ambiguity in text mentioned earlier in this section, analysing of unstructured/semi-structured data (e.g. text data) needs to be extremely careful in every step of the data analysis because the degree of ambiguity in natural language is rigorous (Zhang and Nunamaker, 2004). Some of the ambiguity problems in natural language can be explained following.

First, it is the problem of word-form changing, especially the *noun* and the *verb*. For example, consider the word “*extract*”. It can change its form to “*extracts*”, “*extracting*”, and “*extracted*”, where the forms of words depend on the context of the sentence. Second, it is the problem of words that have the same meaning, but they are different words. These words are called ‘*synonyms*’.

As the problems above, they can lead to poor accuracy of text analysis, where an automatic text analysis system cannot understand that these words have the same meaning.

To address these problems, a semantic tool(s) used to support and improve the domain of finding useful knowledge / patterns from textual data is required. Therefore, we develop the semantic tool that will be used to address the ambiguity problem of language caused by changing of word-forms. Also, it helps to make more understanding of words that are contained in the content. Finally, this ontology can support the search accuracy for knowledge from text by understanding the searchers’ intent and the contextual meaning of terms as they appear in the searchable data-space. The proposed semantic mechanism is called the Variant Lexicon Ontology (VL-ontology).

In the ON-KDT methodology, the proposed ontology will be used in two main tasks: data preparation and closed-domain knowledge extraction.

7.3 Summary of the Experiments and Discussions of the Thesis

To provide the confidence of the proposed methodological effectiveness, the On-KDT methodology is validated using a developed with three distinct settings.

(1) The first setting is to apply the On-KDT methodology for extracting scenarios from natural language software requirements. Extracting scenarios are necessary in order to advise methodology to manage the amount of information in software requirements becomes more apparent. Utilising automatic extraction methodology is very practical as it can help to establish the scope of software systems and facilitate information reuse. In addition, this can serve as a bridge that is used to transition from the stage of requirements collecting to the stage of designing the architecture for software systems. In this setting, the software requirement of the course-registration is used as the case study. The original textual requirements description is about 70 pages long.

Firstly, it needs to understand the structure of requirement sentences. A requirement sentence consists of three main units: *subject*, *verb*, and *object*. In general, the first step of analysing a sentence should be to identify the verb. The subject and object can be identified only after the verb has been identified. The subject is always in front of the verb, while the object is always followed by the verb. By the well-known definitions of English grammar, the *subject* is the person or thing (who/what) performs the action. The *verb* described the action of the subject, while the *object* is the person or thing affected by the action described in the *verb*. The *object* can be nouns, pronouns, phrases, or clauses. In a scenario, the subject refers to the actor, while the action is determined via a combination of the verb and the object.

As above, we provide 24 answering patterns for actor extraction and 35 answering patterns for action extraction. These patterns are provided to extract all activities of each scenario. After testing by recall and precision, the precision measurements are the results of scenario extraction evaluated by a domain expert. It can be seen that, although the results of the precision measurements are satisfactory, they also show the failure rate. In our study, if a requirement sentence is the simple sentence, it has no any problem. However, if the sentence is more complex, our proposed system may face a big problem. For example, consider following complex sentence.

“The student can also update or delete course selections if changes are made within the add/drop period at the beginning of the semester.”

The actor of this performance is ‘*students*’. However, it may confuse about the actions of the students in this sentence. In fact, if this is performed manually, ‘*can also update or delete*’ is extracted as the action. Unfortunately, our answering patterns for extracting action do not cover in this case.

After a scenario is extracted from the software requirements, it will be stored in the form of the XML schema. As the results, these scenarios can be used to support in the next stage of the software development life cycle (SDLC), the requirement analysis and design stage. This is because these scenarios can be linked to use cases in the Unified Modelling Language (UML). In that context, each scenario can be regarded as an instance of a use case. Therefore, this study presents our approach complements domain of software requirement analysis by quickly offering insights into scenario of each system functionality.

(2) The second setting is to extract knowledge relating clinical practice from PubMed abstracts. In general of finding medical knowledge from PubMed abstracts, users will use PubMed to retrieve the documents, and then they will read the present document for relevance. As this, finding knowledge from PubMed abstracts is time-consuming and labour-intensive process, especially when there are a large number of documents and each document contains more detail. In order to fully employ on-line medical documents effectively, it is crucial to advise methodology to find important knowledge from medical documents retrieved from PubMed. In this context, PubMed abstracts relating to cervix cancer are used as the case study.

This setting presents an advantage that the On-KDT methodology driven over natural language processing technology can bring to extract embedded knowledge from biomedical literatures. It aims to extract clinical knowledge relevant to cervix cancer from MEDLINE abstracts. In this context, the On-KDT methodology not only uses the VL-ontology to support the search accuracy for knowledge from text, but also the CCT-net. The CCT-net (Cancer Term Net Ontology) was developed to support the process of content-based text analysis. It is useful for two reasons. First, a set of prior knowledge can help to identify the particular domain that users employ when searching. Second, the ontology can handle the problem of term variation in the specific domain.

In order to observe the characteristics of our system, we have gathered 200 cervix cancer abstracts in the clinical trial and use them as our experimental datasets. The results of clinical knowledge extraction are evaluated by comparing the results between the domain expert and our system. The average of the results is satisfactory. However, the result of

extracting arm names is poor. This is because the answer for this question sometimes needs to use advanced semantic rules to interpret the exact meaning of its. Consider following sentence, “*The overall clinical response rate was 84.6% and included a complete response (CR) in four patients (7.7%), partial response (PR) in 40 patients (76.9%), and disease (SD) in the remaining eight patients (15.4%). Surgery revealed positive nodes in 9.6% neoadjuvant chemotherapy group patients and in 29.6% primary surgery group patients (P = 0.014).*” If it is analysed by an expert human, *using neoadjuvant chemotherapy* and *using surgery alone* can be extracted as the arm-names. However, this sentence cannot be extracted clinical knowledge by using the current rules. Therefore, the proposed rules should be improved to support the process of analysis of the system.

After obtaining the clinical knowledge from PubMed abstracts, this knowledge will be represented in a format, named Clinical Knowledge Markup Language (CKML). This approach may offer the several benefits such as increasing efficiency for PubMed searches by reducing the time to select and collect the most relevant abstracts for the requested domain, leading to better clinical decisions because of better assimilation of the literatures relevant to their researches and clinical decision.

(3) Business rules are defined and constrained from various aspects of businesses. They can be used as a guide for the operation of an organization. When a business changes in market places, its applications are also required to change. Then, application development teams are expected to release changes and improvements in a short time. Consequently, when an organization requires improving its business policies, it needs to know current business rules used. In general, discovering business rules is very costly in large and complex systems. Nevertheless, it can be found that many organizations today have their business rules embedded in several existing artefacts such as business process models, applications, or databases. As this, it is possible to extract business rules from these existing applications.

In our literature review, business knowledge can be done by a well-known technique, called process mining (van der Aalst and Weijter, 2003, van der Aalst and Weijters, 2004, van der Aalst et al., 2008). There has been limited work in mining process models. A large body of research work in this area mainly focuses on the derivation of process models from event logs. Nevertheless, there are settings in which process mining proves to be inadequate. If a process has never been performed, no event logs information is available, rendering the technique useless. In addition, learning/data mining techniques, such as those used in this approach to process, only return reliable results when there are sizeable datasets (in the case of process mining, event logs) available. If the size of the set of event

logs is small (which is often the case with infrequently executed processes, such as those associated with disaster management, or with Greenfield applications), the reliability of the knowledge extracted cannot be guaranteed.

As above, we also apply the On-KDT methodology to find sequential patterns hidden in process model repositories, where process models are viewed as text. The extracted sequential patterns will be considered as business rules. In this context, business rules that are automatically extracted in our approach represent significant and valid sequential correlations among business activities belonging to a particular organization.

After the business rules that are extracted from process models is tested by retrieving relevant process models from process model repositories. The results of process model retrieval are satisfactory. These outcomes have demonstrated the effectiveness of our approach in terms of extracting the correct business rules.

7.4 Summary of the Common Lesson from the Three Distinct Settings

As the lack of a unified methodology of knowledge discovery from unstructured/semi-structured textual data is the challenge and motivation, we propose a novel conceptual methodology that can be utilised as a guide for discovering useful knowledge from unstructured/semi-structured textual data. It is noted that, unstructured/semi-structured data should be viewed as text.

In this study, three different datasets are used to evaluate the proposed methodology, and then the empirical experiments present the satisfying results. The expected results of extracting knowledge from each informal repository should be requirement scenarios, clinical knowledge, and business rules. Based on the perspective of this thesis, scenarios can be used as a main requirement in a software process, and clinical knowledge can be used to support clinical decision-making. We also show one instance of the use of the On-KDT methodology for business rule extraction from non-textual data (specifically: process models), where process models are viewed as text. As this, it seems that, the results from three different settings may be different. However, they also represented the common lessons that are obtained from the three case studies. They can be concluded as follows.

First, if a target data is unstructured/semi-structured textual data, it is possible to apply the On-KDT methodology for discovering useful knowledge from the target dataset.

Second, the consolidation of the VL-ontology to the KDD is a resolution to solve the semantic ambiguity in English text during text analysis and processing, particularly in the case of word variation.

Third, the common lesson of getting useful knowledge from different unstructured/semi-structured textual data through the ON-KDT methodology is to have the results in the form of regulation (rules or constraints).

7.5 Awareness in Application

In this setting, although the On-KDT methodology can substantially succeed in identifying useful knowledge from unstructured/semi-structured data, it may accept a restriction. This is because the On-KDT methodology returns the satisfying effects, where unstructured/semi-structured data, which is used as the dataset, is interpreted in the English text.

As above, if the On-KDT is required to apply to other languages, it can just be applied as the guideline for knowledge discovery in unstructured/semi-structured text. Simply speaking, it does not effect for other languages. This is because; the VL-ontology is only driven on the English language. If the On-KDT methodology is applied to other languages such as Thai, Chinese, and Japanese, the main ontology, VL-ontology, should be modified because it cannot support for other languages. Therefore, we need to develop a new ontology based on the language that is meditating. Besides, the complex languages require the extreme care in every step of text analysis and processing. Simply speaking, the On-KDT methodology can be applied as the concept of knowledge discovery on unstructured/semi-structured data, when it is applied to other languages.

Difficulty in a complex language (i.e. Thai) can be illustrated following. Consider text processing in the step of word tokenisation, which is an important task which has a significant impact on higher language processing levels. By characteristics of Thai language, it does not have any explicit word boundary delimiters, such as a space to separate between each word. This means, Thai text tokenisation is harder than Latin-based language such as English, French, and Spanish. Therefore, to address this problem, a dictionary is required for tokenizing text which is traditionally written without word boundary. As this, it can be stated that the tokenisation accuracy depends significantly on the quality of the dictionary. If the dictionary is not good enough, it may lead to a critical problem of unknown words, and then it will contribute to the problem of language ambiguity. Finally, this problem can lead the performance of text analysis and processing being poor.

REFERENCES

- AALST, W. M. P. V. D. & WEIJTER, A. J. M. M. 2004. Process Mining: A Research Agenda. *Computers and Industry*, 53, 231-244.
- ADAMO, J. M. 2000. *Data Mining for Association Rules and Sequential Patterns Sequential and Parallel Algorithms*, Springer.
- AGRAWAL, R. & SRIKANT, A. 1995. Mining Sequential Patterns. *Proceedings of the Eleventh International Conference on Data Engineering (ICDE)*.
- AGRAWAL, R. & SRIKANT, R. 1994. Fast algorithms for mining association rules in large databases. *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB)*.
- ALANI, H., SANGHEE, K., MILLARD, D. E., WEAL, K. J., HALL, W., LEWIS, P. H. & SHADBOLT, N. R. 2003. Automatic ontology-based knowledge extraction from Web documents. *IEEE Intelligent Systems*, 18, 14 - 21.
- ALSPAUGH, T. A., ANTON, A. I., BARNES, T. & MOTT, B. W. 1999. An Integrated Scenario Management Strategy. *Requirement Engineering*.
- ALSPAUGH, T. A. & ANTON, A. I. 2008. Scenario support for effective requirements. *Information and Software Technology*, 50.
- AMYOT, D., HE, X., HE, Y. & CHO, D. Y. 2003. Generating Scenarios from Use Case Map Specifications. *The International Conference on Quality Software (QSIC)*.
- ANAND, S. & BUCHNER, A. 1998. *Decision Support Using Data Mining*, Financial Times Pitman.
- ANNAMALAI, P. Year. Extracting Knowledge in the Internet Age. In: *Proceedings of the 44th annual Southeast regional conference*, 2006.
- ANTUNES, B., SECO, N. & GOMES, P. 2007. Using Ontologies for Software Development Knowledge Reuse. *Lecture Notes in Computer Science*, 874/2007, 357-368.
- APTE, C., LIU, B., PEDNAULT, E. P. D. & SMYTH, P. 2002. Business Applications of Data Mining. *Communications of the ACM*, 45.
- ASGHAR, S. & IGBAL, K. 2009. Automated Data Mining Techniques: A Critical Literature Review. *International Conference on Information Management and Engineering (ICIME)*.
- BAEZA-YATES, R. & RIBEIRO-NETO, B. 1999. *Modern Information Retrieval*, New York, ACM Press, Addison-Wesley.

- BAJWA, I. S. S., SAMAD, A. & MUMTAZ, S. 2009. Object Oriented Software Modeling Using NLP Based Knowledge Extraction. *European Journal of Scientific Research*, 35.
- BANKO, M. & ETZIONI, O. 2007. Strategies for Lifelong Knowledge Extraction from the Web. *The Forth International Conference on Knowledge Capture (K-CAP)*.
- BEALL, J. 2008. The Weakness of Full-Text Searching. *The Journal of Academic Librarianship*, 34, 438-444.
- BELIAEV, S. & KRASLAWSKI, A. 2007. Knowledge discovery method for the identification of solvents for the bio-catalytic reactions. *The 38th European Symposium of the Working Party on Computer Aided Process Engineering*.
- BERNERS-LEE, T., HENDLER, J. & LASSILA, O. 2001. The Semantic Web, *Scientific American* 284.
- BLASCHKE, C., A.M., A., OUZOUNIS, C. & VALENCIA, A. 1999. Automatic extraction of biological information from scientific text: protein-protein interactions. *Proc Int Conf Intell Syst Mol Biol*.
- BOSER, B. E., GUYON, I. M. & VAPNIK, V. N. 1992. A training algorithm for optimal margin classifiers. In: HAUSSLER, D. (ed.) *5th Annual ACM Workshop on Conference on Learning Theory (COLT)*. ACM Press.
- BRACHMAN, R. J. & ANAND, T. 1994. The process of knowledge discovery in databases: A first sketch. *Processdings 1994 AAAI Workshop on Knowledge Discovery in Database*.
- BRILL, E. 1992a. A simple rule-based part of speech tagger. *Proceedings of the third conference on Applied natural language processing (ANLC '92)*. Stroudsburg, PA, USA: Association for Computational Linguistics.
- BRILL, E. Year. A simple rule-based part of speech tagger. In: *Proceedings of the third conference on Applied natural language processing (ANLC '92)*, 1992b. Stroudsburg, PA: Association for Computational Linguistics, 152-155.
- BRILL, E. 1994. Some Advances in Transformation-Based Part of Speech Tagging. *Proceedings of the twelfth national conference on artificial intelligence*.
- BROWN, P. F., DESOUZA, P. V., MERCER, R. L., PIETRA, V. J. D. & LAI, J. C. 1992. Class-based N-gram Models of Natural Language. *Computational Linguistics*, 18, 467-479.
- CABENA, P., HADJINIAN, P., STADLER, R., VERHEES, J. & ZANASI, A. 1998. *Discovering Data Mining: From Concepts to Implementation*, Prentice Hall.

- CALIFF, M. E. & MOONEY, R. J. 1997. Relational Learning of Pattern-Match Rules for Information Extraction. In: ELLISON, T. M. (ed.) *Computational Natural Language Learning (CoNLL)*. ACL.
- CALLAN, J. P. 1994. Passage-level evidence in document retrieval. In: CROFT, B. W. & RIJSBERGEN, C. J. V. (eds.) *Proceedings of the 17th annual international ACM-SIGIR conference on research and developments in information retrieval*. ACM.
- CANTU, F. J., MORA, S. P., DÍAZ, A., CEBALLOS, H., MARTINEZ, S. O. & JIMENEZ, D. R. 2005. A knowledge-based entrepreneurial approach for business intelligence in strategic technologies: Bio-MEMS. *Proceedings of the Eleventh Americas Conference on Information Systems*.
- CAO, L. J. & TAY, F. H. 2000. Feature Selection for Support Vector Machines in Financial Time Series Forecasting. *Intelligent Data Engineering and Automated Learning (IDEAL)*, LNCS 1983, 268-273.
- CARPINETO, C. & ROMANO, G. 2012. A Survey of Automatic Query Expansion in Information Retrieval. *Journal of ACM Computing Surveys (CSUR)*, 44.
- CARROLL, J. M. 1995. The scenario perspective on system development. In: CARROLL, J. M. (ed.) *Scenario-based design: envisioning work and technology in system development*. J. Wiley, New York.
- CARROLL, J. M. 1997. Scenario-based Design. In: HELANDER, M. G., LANDAUER, T. K. & PRABHU, P. V. (eds.) *Handbook of Computer Interaction*. 2nd ed. Amsterdam: North Holland.
- CAVNAR, W. B. 1994. N-gram-Based Text Filtering. *TREC1993*, 171-180.
- CHAPMAN, P., CLINTON, J., KERBER, R., KHABAZA, T., REINARTZ, T., SHEARER, C. & WIRTH, R. 2000. CRISPDM 1.0 step-by-step data mining guide. Technical report, CRISP-DM.
- CHEN, R. S., CHANG, C. C. & CHI, I. 2006. Ontology-Based Knowledge Extraction-A Case Study of Software Development. *The 7th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD'06)*.
- CHURCH, K. Year. A stochastic parts program and noun phrase parser for unrestricted text. In: *Proceedings Second ACL Applied NLP*, 1988. 136-143.
- CIMIANO, P., HANDSCHUH, S. & STAAB, S. Year. Towards the self-annotating web. In: *Proceedings of the 13th International Conference on World Wide Web*, 2004. ACM, New York.

- CIOS, K. & KURGAN, L. 2005. Trends in Data Mining and Knowledge Discovery. *In: N.R., P., JAIN, L. C. & TEODERESKU, N. (eds.) Knowledge Discovery in Advanced Information Systems.*
- CIOS, K. J., PEDRYCZ, W., SWINIARSKI, R. W. & KURGAN, L. A. 2007. The Knowledge Discovery Process. *Data Mining: A Knowledge Discovery Approach.* Springer.
- CLARK, D. D., PARTRIDGE, C., RAMMING, J. C. & WROCLAWSKI, J. T. 2003. A Knowledge Plane for the Internet. *The ACM Special Interest Group on Computer Communication (SIGCOMM).*
- CLARKE, C., CORMACK, G. & TUDHOPE, E. 2000a. Relevance ranking for one to three term queries. *Information Processing and Management*, 36:291–311.
- CLARKE, C. L. A., CORMACK, G. V., KISMAN, D. I. E. & LYNAM, T. R. 2000b. Question answering by passage selection (multitext experiments for trec-9). *Proceedings of the Ninth Text REtrieval Conference (TREC-9).*
- COCKBURN, A. 1995. *Structuring Use Cases with Goals* [Online]. Salt Lake City, Utah. Available: <http://members.aol.com/acocburn/papers/usecases.htm> [Accessed 1 March 2012].
- COOK, J. E. & WOLF, J. E. 1998. Discovering Models of Software Processes from Event-Based Data. *ACM Transactions on Software Engineering and Methodology*, 7, 215-249.
- CORCHO, O. & GÓMEZ-PÉREZ, A. 2000. A Roadmap to Ontology Specification Languages. *12th International Conference on Knowledge Engineering and Knowledge Management.*
- COULET, A., SMAIL-TABBONE, M., NAPOLI, A. & DEVIGNES, M.-D. 2011. Ontology-based Knowledge Discovery in Pharmacogenomics. *Software Tools and Algorithms for Biological Systems*, 357-366.
- CRANGLE, C. E. 2002. Text Summarization in Data Mining. *Computing in an Imperfect World, LNCS*, 2311/2002, 605-648.
- CUTTING, D., KUPIEC, J., PEDERSON, J. & P. SIBUN, P. Year. A practical part of speech tagger. *In: Proceedings Third ACL Applied NLP*, 1992. 133-140.
- DAELEMANS, W., ZAVREL, J., BERCK, P. & GILLIS, S. Year. MBT: A Memory-Based Part of Speech Tagger-Generator. *In: Proceedings of 4th Workshop on Very Large Corpora*, 1995.
- DALE, R., MOISL, H. & SOMERS, H. (eds.) 2000. *Handbook of Natural Language Processing: MerceL Dekker Inc., New York.*

- DELIC, K. A. & RILEY, J. A. 2009. Enterprise Knowledge Clouds: Next Generation KM Systems? *International Conference on Information, Process, and Knowledge Management*.
- DEMNER-FUSMAN, D. & LIN, J. 2007. Answering Clinical Questions with Knowledge-Based and Statistical Techniques. *Journal of Computational Linguistics*, 33 63-103.
- DENICIA-CARRAL, C., MONTES-Y-GÓMEZ, M., VILLASEÑOR-PINEDA, L. & HERNÁNDEZ, R. G. 2006. A text mining approach for definition question answering. *Proceedings of the 5th international conference on Advances in Natural Language Processing*
- DEROSE, S. 1988. Grammatical category disambiguation by statistical optimization. *Computational Linguistics*, 14, 31-39.
- DEVEDZIC, V. 2001. *Knowledge Discovery and Data Mining in Databases*, World Scientific Publishing.
- DHILLON, I. S., GUAN, Y. & KOGAN, J. 2002. Iterative Clustering of High Dimensional Text Data Augmented by Local Search. *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002)*. IEEE Computer Society.
- DIAMANTINI, C., POTENA, D. & STORTI, E. Year. KDDONTO: An Ontology for Discovery and Composition of KDD Algorithms. In: *International Workshop on the Third Generation Data Mining: Towards Service-oriented Knowledge Discovery*, 2009a.
- DIAMANTINI, C., POTENA, D. & STORTI, E. 2009b. Ontology-driven KDD Process Composition. In: ADAMS, N. M., ROBARDET, C., SIEBES, A. & BOULICAUT, J.-F. (eds.) *Advances in Intelligent Data Analysis VIII*. Springer Berlin Heidelberg.
- DOAN-NGUYEN, H. & KOSSEIM, L. 2004. Improving the Precision of a Closed-Domain Question-Answering System with Semantic Information. In: FLUHR, C., GREFENSTETTE, G. & CROFT, W. B. (eds.) *RIAO 2004 in Computer-Assisted Information Retrieval*.
- DZIDA, W. & FREITAG, R. 1998. Making Use of Scenarios for Validating Analysis and Design. *IEEE Transactions on Software Engineering*, 24, 1182-1196.
- EKMEKÇIOĞLU, F. Ç. & WILLETT, P. 2000. Effectiveness of stemming for Turkish text retrieval. *The Association for Information Management*, 34.
- ELDER, J. F. & PREGIBON, D. 1996. A Statistical Perspective on Knowledge Discovery in Databases. In: U.M.FAYYAD, G.PIATETSKY-SHAPIRO, P.SMITH & R.UTHURUSAMY (eds.) *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press.

- FAYYAD, U. 2000. Knowledge discovery in databases: An overview. In: DĚZEROSKI, S. (ed.) *Relational Data Mining*.
- FAYYAD, U., PIATETSKY-SHAPIO, G. & SMYTH, P. 1996a. From Data Mining to Knowledge Discovery in Databases. *AI Magazine*.
- FAYYAD, U., PIATETSKY-SHAPIO, G. & SMYTH, P. 1996b. The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM*, 39
- FELDMAN, R. & DAGAN, I. 1995. Knowledge Discovery in Textual Databases (KDT) *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD)*.
- FELDMAN, R. & SANGER, J. 2006. *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*, Cambridge University Press.
- FOLORUNSO, O. & OGUNDE, A. O. 2005. Data mining as a technique for knowledge management in business process redesign. *Information Management and Computer Security*, 13, 274-280.
- FRIEDMAN, J. 1997. Data mining and statistics: What's the connection? . *Proceedings of the 29th Symposium on the Interface: Computing Science and Statistics*.
- GARGANTINI, A., RICCOBENE, E., SCANDURRA, P. & CARIONI, A. 2008. Scenario-based Validation of Embedded Systems. *Forum on specification and Design Languages (FDL)*.
- GO, K. & CARROLL, J. M. 2004. Scenario-Based Task Analysis. In: DIAPER, D. & STANTON, N. A. (eds.) *The Handbook Of Task Analysis For Human-Computer Interaction*. Mahwah, NJ.: Lawrence Erlbaum Associates.
- GOLBECK, J., FRAGOSO, G., HARTEL, F., HENDLER, J., OBERTHALER, J. & PARSIA, B. 2003. The National Cancer Institute's Thesaurus and Ontology. *Journal of Web Semantics*, 1.
- GOLDBERG, Y. & ELHADAD, M. 2009. On the Role of Lexical Features in Sequence Labeling. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*.
- GOTTSCHALK-MAZOUZ, N. 2007. Internet and the flow of knowledge. In: HRACHOVEC, H. & PICHLER, A. (eds.) *Philosophy of the Information Society: Proceedings of the 30th International Ludwig Wittgenstein Symposium Kirchberg am Wechsel*.
- GROBELNIK, M., MLADENIC, D. & MILIC-FRAYLING, N. 2000a. Text mining as integration of several related research areas: report on KDD's workshop on text mining 2000. *ACM SIGKDD Explorations Newsletter*, 2, 99-102.

- GROBELNIK, M., MLADENIC, D. & MILIC-FRAYLING, N. 2000b. Text Mining as Integration of Several Related Research Areas: Report on KDD'2000. *Workshop on Text Mining*. KDD.
- GRUBER, T. R. 1993. A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition*, 5(2), 199-220.
- GUARINO, N. 1998. Formal Ontology and Information Systems. In: *Proceedings of the First International Conference on Formal Ontologies in Information Systems (FOIS)*.
- GUO, J., XU, G., CHENG, X. & LI, H. 2009. Named entity recognition in query. In: *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*.
- HADZIC, M., D'SOUZA, R., HADZIC, F. & DILLON, T. 2008. Thinking PubMed: an Innovative System for Mental Health Domain. *The 21st IEEE International Symposium on Computer-Based Medical Systems*.
- HAN, J. & KAMBER, M. 2006. *Data Mining: Concepts and Techniques*, Elsevier.
- HAN, J. & PEI, J. 2000. Mining frequent patterns by pattern-growth: methodology and implications. *ACM SIGKDD Explorations Newsletter*, 14-20.
- HAN, K., SONG, Y. & RIM, H. 2006. Probabilistic Model for Definitional Question Answering. *Proceedings of SIGIR 2006*.
- HANDSCHUH, S. S. & CIRAVEGNA, F. 2002. S-CREAM - Semi-Automatic Creation of Metadata. *Workshop of Semantic Authoring, Annotation and Markup in 15th European Conference on Artificial Intelligence (ECAI'02)*.
- HARMAN, D. 1991. How effective is suffixing? *Journal of the American Society for Information Science*, 42, 7-15.
- HARRAG, F., AL-QAWASMAH, E. & A.S.AL-SALMAN. 2010. Comparing Dimension Reduction Techniques for Arabic Text Classification using BPNN algorithm. *International Conference on Integrated Intelligent Computing*.
- HAYES, J. H., DEKHTYAR, A. & SUNDARAM, S. 2005. Text mining for software engineering: how analyst feedback impacts final results. *Proceedings of the 2005 international workshop on mining software repositories*.
- HEARST, M. 1999. Untangling Text Data Mining. In: *The Proceedings of ACL: the 37th Annual Meeting of the Association for Computational Linguistics*.
- HEARST, M. A. & PLAUNT, C. 1993. Subtopic structuring for full-length document access. In: KORFHAGE, R., RASMUSSEN, E. & WILLET, P. (eds.) *Proceedings*

of the 16th annual international ACM-SIGIR conference on research and development in information retrieval. ACM.

- HERMJAKOB, U., ECHIHABI, A. & MARCU, D. 2002. Natural Language Based Reformulation Resource and Web Exploitation for Question Answering. *Proceedings of the TREC-2002 Conference*.
- HERSH, W. 2005. Evaluation of biomedical text-mining systems: Lessons learned from information retrieval. *Briefings in Bioinformatics*, 6, 344-356.
- HILL, J. B., CANTARA, M., DEITERT, E. & KERREMANS, M. 2007. Magic quadrant for business process management suites. Tech. rep., Gartner Research.
- HINDLE, D. Year. Acquiring disambiguation rules from text. *In: Proceedings 27th Annual Meeting of the Association for Computational Linguistics*, 1989.
- HIRJI, K. K. 2001. Exploring Data Mining Implementation. *Communications of the ACM*, 44(7), 87-93.
- HIRSCHMAN, L. & GAIZAUSKAS, R. 2001. Natural language question answering: The view from here. *Natural Language Engineering*, 7(4), 275-300.
- HOVY, E., HERMJAKOB, U. & LIN, C.-Y. 2001. The use of external knowledge in factoid QA. *In: Proceedings of the Tenth Text REtrieval Conference (TREC 2001)*.
- HOVY, E. H., GERBER, L., HERMJAKOB, U., JUNK, M. & LIN, C.-Y. 2000. Question Answering in Webclopedia. *In: Proceedings of the TREC-9 Conference*, Gaithersburg, MD. NIST.
- HOVY, E. H. & LIN, C.-Y. 1998. Automated Text Summarization in SUMMARIST. *In: MAYBURY, M. & MANI, I. (eds.) Intelligent Scalable Summarization Text Summarization*. Forthcoming.
- HRUBY, P. 2005. Role of Domain Ontologies in Software Factories. *Workshop on Software Factories in the Conference of Object-Oriented Programming, Systems, Languages & Applications (OOPSLA)*.
- HU, X., LIN, T. Y., SONG, I. Y., LIN, X., I.YOOI, LECHNER, M. & SONG, M. 2004. Ontology-based Scalable and Portable Information Extraction System to Extract Biological Knowledge from Huge Collection of Biomedical Web Documents. *Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*. IEEE Computer Society.
- HUANG, J., DOU, D., HE, L. & DANG, J. 2010. Ontology-based knowledge discovery and sharing in bioinformatics and medical informatics: A brief survey. *In: 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, pp. 2203 - 2208.

- HWANG, C. 1999. Incompletely and imprecisely speaking: using dynamic ontologies for representing and retrieving information. *In: FRANCONI, E. & KIFER, M., eds. The 6th International Workshop on Knowledge Representation Meets Databases, ACM, New York.*
- INDURKHYA, N. & DAMERAU, F. 2010. *Handbook Of Natural Language Processing*, Boca Raton, FL: CRC Press.
- JACKSON, J. 2002. Data Mining: A Conceptual Overview. *Journal of Communications of the Association for Information Systems*, 8, 267-296.
- JI, X. & XU, W. 2006. Document clustering with prior knowledge. *ACM Special Interest Group on Information Retrieval (SIGIR)*, 405-412.
- JIN, Y., VON SEELEN, W. & SENDHOFF, B. 1998. An Approach to Rule-Based Knowledge Extraction. *Proceedings of IEEE Conference on Fuzzy System*.
- JOACHIMS, T. 1998. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. *Proceedings of the European Conference on Machine Learning (ECML)*. Springer.
- JOACHIMS, T. 1999. Transductive Inference for Text Classification using Support Vector Machines. *Proceedings of the International Conference on Machine Learning (ICML)*.
- JOHNSON, S. C. 1967. Hierarchical Clustering Schemes. *Psychometrika*, 2, 241-254.
- JOSHI, A. & JIANG, Z. 2002. Retriever: Improving Web Search Engine Results Using Clustering. *In: GANGOPADHYAY, A. (ed.) Managing Business with Electronic Commerce: Issues and Trends*. Idea Gropu Publishing.
- JURAFSKY, D. & MARTIN, J. H. 2009. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, Pearson Prentice Hall.
- KAMEL, M. & ROTHENBURGER, B. 2010. Ontology Building Using Parallel Enumerative Structures. *Proceedings of the International Conference on Knowledge Engineering and Ontology Development*
- KANGAVARI, M. R., GHANDICHI, S. & GOLPOUR, M. 2008. Information Retrieval: Improving Question Answering Systems by Query Reformulation and Answer Validation. *Proceedings of World Academy of Science, Engineering, and Technology*.
- KARANASTASI, A. & MATSATSINIS, N. 2010. Agent Technology Meets the Semantic Web: Interoperability and Communication Issues. *In: BADR, Y., CHBEIR, R.,*

- ABRAHAM, A. & HASSANIEN, A.-E. (eds.) *Emergent Web Intelligence: Advanced Semantic Technologies*. Springer-Verlag.
- KAZMAN, R., ABOWN, G., BASS, L. & CLEMANT, P. 1994. Scenario-based Analysis of Software Architecture. *IEEE Transactions on Software Engineering*.
- KLEIN, S. & SIMMONS, R. 1963. A grammatical approach to grammatical coding of English words. *The Journal of the ACM (JACM)*, 10, 334-347.
- KLIEN, E., LUTZ, M. & KUHN, W. 2006. Ontology-based discovery of geographic information services - An application in disaster management. *Computers, Environment and Urban Systems*, 30, 102-123.
- KORFHAGE, R. R. 1997. *Information storage and retrieval*, Wiley Computer Pub.
- KOTSIANTIS, S., KANELLOPOULOS, D. & PINTELAS, P. 2006. Data Preprocessing for Supervised Learning. *International Journal of Computer Science*, 1, 111-117.
- KURGAN, L. A., CIOS, K. J. & TADEUSIEWICZ, R. 2001. Knowledge discovery approach to automated cardiac SPECT diagnosis. *Artificial Intelligence in Medicine*, 23/2, 149-169.
- KURGAN, L. A. & MUSILEK, P. 2006. A survey of Knowledge Discovery and Data Mining process models. *The Knowledge Engineering Review*, 21(1).
- LAROCCA, J., FREITAS, N. A. A. & KAESTNER, C. A. A. 2002. Automatic Text Summarization using a Machine Learning Approach. In: BITTENCOURT, G. & RAMALHO, G. (eds.) *16th Brazilian Symposium on Artificial Intelligence*. Advances in Artificial Intelligence.
- LAROSE, D. T. 2005. *Discovering Knowledge in Data: An Introduction to Data Mining*, Wiley-Interscience.
- LEE, D. & CHU, W. W. 2000. Comparative Analysis of Six XML Schema Languages *ACM SIGMOD Record*, 29(3), 76-87.
- LEE, H.-W. 2005. Knowledge Management and the Role of Libraries. *The 3rd China-US Library Conference*.
- LETIER, E., KRAMER, J., MAGEE, J. & UCHITEL, S. 2005. Monitoring and control in scenario-based requirements analysis. *Proceedings of the 27th international conference on Software engineering*.
- LI, S. & ITOH, Y. 1998. On Removing Ambiguity in Text Understanding. *Language, Information and Computation (PACLIC12)*, 271-282.
- LI, Y. & BONTCHEVA, K. 2007. Hierarchical, perceptron-like learning for ontology-based information extraction In: *Proceedings of the 16th International Conference on World Wide Web 2007*. ACM, New York

- LIGHT, M., MANN, G. S., RILOFF, E. & BRECK, E. 2001. Analyses for elucidating current question answering technology. *Natural Language Engineering*, 7(4).
- LIN, J., QUAN, D., SINHA, V., BAKSHI, K., HUYNH, D., KATZ, B. & KARGER, D. R. 2003. What makes a good answer? The role of context in question answering. In: *Proceedings of the ninth IFIP TC13 International Conference on Human-Computer Interaction*.
- LIU, X. & CROFT, W. B. 2002. Passage Retrieval Based On Language Models. *Conference on Information and Knowledge Management (CIKM)*.
- LOH, S., WIVES, L. K. & PALAZZO, J. 2000. Concept-Based Knowledge Discovery in Texts Extracted from the Web. *SIGKDD Explorations*, 2, 29-39.
- LUKE, S., SPECTOR, L. & RAGER, D. 1997. Ontology-Based Knowledge Discovery on the World-Wide Web. In: *Proceedings of the Workshop on Internet-based Information Systems*, AAAI.
- MAIDEN, N. A. M. 1998. CREWS-SAVRE: Scenarios for Acquiring and Validating Requirements. *Automated Software Engineering*, Kluwer Academic Publishers, 5, 419-446.
- MAIMON, O. & ROKACH, L. 2005. Introduction to Knowledge Discovery in Database. In: MAIMON, O. & ROKACH, L. (eds.) *The Data Mining and Knowledge Discovery Handbook*. Springer.
- MAKHOUL, J., KUBALA, F., SCHWARTZ, R. & WEISCHEDEL, R. 1999. *Performance measures for information extraction* [Online]. Available: <http://www.itl.nist.gov/iad/mig/publications/proceedings/darpa99/html/dir10/dir10.htm> [Accessed 15 January 2012].
- MANEVITZ, L. M. & YOUSEF, M. 2001. One-Class SVMs for Document Classification. *Journal of Machine Learning Research*, 139-154.
- MANN, T. 2005. *Will Google's Keyword Searching Eliminate the Need for LC Cataloging and Classification?* [Online]. Available: <http://cdigital.uv.mx/bitstream/123456789/6488/2/Thomas%20Mann.pdf> [Accessed 15 February 2014].
- MANNING, C. D., RAGHAVAN, P. & SCHÜTZE, H. 2009. *An Introduction to Information Retrieval*, Cambridge University Press.
- MANNING, C. D. & SCHÜTZE, H. 1999. *Foundations of Statistical Natural Language Processing*, MIT Press.
- MANSUROV, N. N. & PROBERT, R. L. 2001. Scenario-based approach to evolution of communication software. *IEEE Communications*, 94-100.

- MANUJA, M. & GARG, D. 2011. Semantic Web Mining of Un-structured Data: Challenges and Opportunities. *International Journal of Engineering (IJE)*, 5.
- MARBÁN, Ó., MARISCAL, G. & SEGOVIA, J. 2009. A Data Mining & Knowledge Discovery Process Model In: PONCE, J. & KARAHOCA, A. (eds.) *Data Mining and Knowledge Discovery in Real Life Applications*.
- MARQUET, G., BURGUN, A., MOUSSOUNI, F., GUERIN, E., LEDUFF, F. & LOREAL, O. 2003. BioMeKe: an ontology-based biomedical knowledge extraction system devoted to transcriptome analysis. *Stud Health Technol Inform*, 95.
- MAY, M., BERENDT, B., CORNUÉJOLS, A., GAMA, J., GIANNOTTI, F., HOTHO, A., MALERBA, D., MENESALVAS, E., MORIK, K., PEDERSEN, R., SAITTA, L., SAYGIN, Y., SCHUSTER, A. & VANHOOF, K. 2008. *Research Challenges in Ubiquitous Knowledge Discovery* Chapman & Hall.
- MCCALLUM, A. 2005. Extraction: Distilling Structured Data from Unstructured Text *ACM Queue*.
- MCCALLUM, A. & NIGAM, K. 1998. A Comparison of Event Models for Naive Bayes Text Classification. In *AAAI/ICML-98 Workshop on Learning for Text Categorization*. AAAI Press.
- MEDENI, T., ÜNSAL, S., AYAS, M. & MEDENI, I. T. 2011. Tacit Knowledge Extraction for Software Requirements specification (SRS): A Proposal of Research Methodology Design and Execution for Knowledge Visualization. In: *Proceedings of the 55th Annual Meeting of the ISSS*.
- MEKNAVIN, S., CHAROENPORNSAWAT, P. & KIJSIRIKUL, B. 1997. Feature-based Thai Word segmentation. *Proceeding of the Natural Language Processing Pacific RIM Symposium*.
- MENDONÇA, E. A. & CIMINO, J. J. 2000. Automated knowledge extraction from MEDLINE citations. *Proceeding of AMIA Symposium*.
- MERIALDO, B. 1994. Tagging English Text with a Probabilistic Model. *Computational Linguistics*, 20, 155-172.
- MIERS, D. & WHITE, S. A. 2008. *BPMN Modeling and Reference Guide*, Future Strategies Inc.
- MILLER, G. 1990. WordNet: An on-line lexical database. *International Journal of Lexicography*, 3.
- MISSIKOFF, M., NAVIGLI, R. & VELARDI, P. 2002. The Usable Ontology: An Environment for Building and Assessing a Domain Ontology. *Proceedings of the First International Conference on the Semantic Web*. Springer-Verlag.

- MITCHELL, T. 1997. *Machine Learning*, McGraw Hill.
- MOONEY, R. J. & BUNESCU, R. 2005. Mining Knowledge from Text Using Information Extraction. *CM SIGKDD Explorations Newsletter - Natural language processing and text mining*, 7(1).
- MOZINA, M., DEMSAR, J., KATTAN, M. & ZUPAN, B. 2004. Nomograms for Visualization of Naive Bayesian Classifier. *The proceedings of European Conference on Principles of Data Mining and Knowledge Discovery (PKDD)*.
- MU, Y., WANG, Y. & GUO, J. 2009. Extracting Software Functional Requirements from Free Text Documents. *International Conference on Information and Multimedia Technology*.
- MURATA, M., LEE, D., MANI, M. & KAWAGUCHI, K. 2005. Taxonomy of XML Schema Languages using Formal Language Theory. *In ACM Trans. on Internet Technology (TOIT)*, 5, 1-45.
- OWEN, M. & RAJ, J. 2003. BPMN and Business Process Management: Introduction to the New Business Process Modeling Standard. Popkin Software.
- PAICE, C. D. 1994. An evaluation method for stemming algorithms. *In: CROFT, W. B. & RIJSBERGEN, C. J. V. (eds.) Proceedings of the 17th ACM SIGIR conference held at Dublin*.
- PAICE, C. D. 1996. A method for the evaluation of stemming algorithms based on error counting. *Journal of the American Society for Information Science*, 47(8), 632-649.
- PANDEY, G. & DAGA, R. 2007. On Extracting Structured Knowledge from Unstructured Business Documents. *In: Proc IJCAI Workshop on Analytics for Noisy Unstructured Text Data*, pp 155-162.
- PECHSIRI, C., KONGWAN, A. & KAWTRAKUL, A. 2004. Agricultural Knowledge Discovery from Semi-Structured Text. *The Joint conferences of the Asian Federation of Information Technology in Agriculture (AFITA) and World Congress on Computers in Agriculture (WCCA)*.
- PEI, J., HAN, J. & WANG, W. 2007. Constraint-based sequential pattern mining: the pattern-growth methods. *Journal of Intelligent Information Systems*, 22(8), 133-160.
- PENG, F. & MCCALLUM, A. 2006. Information extraction from research papers using conditional random fields. *Information Processing and Management*, 42, 963.
- PEREIRA, F. C., SINGER, Y. & TISHBY, N. 1995. Beyond Word N-grams. *Proceedings of the Third Workshop on Very Large Corpora*.
- PHILLIPS, J. & BUCHANAN, B. G. 2001. Ontology-Guided Knowledge Discovery in Databases. *In: The International Conference on Knowledge Capture (K-CAP'01)*.

- PIATETSKY-SHAPIRO, G. 2002. *KDnuggets Methodology Poll* [Online]. Available: <http://www.kdnuggets.com/polls/2002/methodology.htm> [Accessed 15 September 2010].
- PLIHON, V., RALYTÉ, J., BENJAMEN, A., MAIDEN, N. A. M., SUTCLIFFE, A., DUBOIS, E. & HEYMANS, P. 1998. A Reuse-Oriented Approach for the Construction of Scenario Based Methods. *Proc. Int'l Software Process Assoc. Fifth Int'l Conf. Software Process (ICSP '98)*.
- POPOV, B., KIRYAKOV, A., KIRILOV, A., MANOV, D., OGNYANOFF, D. & GORANOV, M. Year. KIM - semantic annotation platform. In: *Proceedings of the 2nd International Semantic Web Conference, 2003*. Springer-Verlag, Berlin.
- POPOV, B., KIRYAKOV, A., OGNYANOFF, D., MANOV, D. & KIRILOV, A. 2004. KIM - a semantic platform for information extraction and retrieval. *Journal of Natural Language Engineering* 10, 375-392.
- POPOVIC, M. & WILLETT, P. 1992. The effectiveness of stemming for natural language access to Slovene textual data. *Journal of the American Society for Information Science*, 43, 384-390.
- PUSTEJOVSKY, J., CASTANO, J., COCHRAN, B., KOTECKI, M., MORRELL, M. & RUMSHISKY, A. 2001. Linguistic knowledge extraction from MedLine: Automatic construction of an acronym database. *Proceedings of the 10th World Congress on Health and Medical Informatics*.
- QIAO, S., LI, T., PENG, J. & QIU, J. 2012. Parallel Sequential Pattern Mining of Massive Trajectory Data. *International Journal of Computational Intelligence Systems*.
- RAHARDJO, B. & YAP, R. H. C. 2001. Automatic information extraction from web pages. In: *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval 2001*.
- RAJMAN, M. & BESANÇON, R. 1997. Text Mining: Natural Language techniques and Text Mining applications. *Association extraction from indexed data*. Chapman & Hall.
- RAO, R. & SPRAGUE, R. H. 1999. *Natural navigation* [Online]. The original knowledge management publication, 3. Available: <http://www.ikmagazine.com> [Accessed 2 March 2010].
- REEVE, L. & HAN, H. 2005. Survey of semantic annotation platforms. In: *Proceedings of the 2005 ACM symposium on Applied computing (SAC'05)*, ACM, New York, 1634-1638.

- REINARTZ, T. 2002. A Unifying View on Instance Selection. *Data Mining and Knowledge Discovery*, 191-210.
- RITSKO, J. J. & SEIDMAN, D. I. 2004. Preface. *IBM Systems Journal*, 43, 449-450
- ROCCHIO, J. J. 1971. Relevance feedback in information retrieval. In: SALTON, G. (ed.) *The SMART Retrieval System: Experiments in Automatic Document Processing*. Englewood Cliffs NJ: Prentice-Hall Series in Automatic Computation.
- ROSSI, F. & VILLA, N. 2006. Support vector machine for functional data classification. *Neurocomputing*, 730-742.
- SAIEDIAN, H., KUMARAKULASINGAM, P. & ANAN, M. 2005. Scenario-based requirements analysis techniques for real-time software systems: a comparative evaluation. *Requirements Engineering*.
- SALTON, G., ALLAN, J. & BUCKLEY, C. 1993. Approaches to passage retrieval in full text information systems. In: KORFHAGE, R., RASMUSSEN, E. & WILLET, P. (eds.) *Proceedings of the 16th annual international ACM-SIGIR conference on research and development in information retrieval*. ACM.
- SALTON, G., WONG, A. & YANG, C. S. 1975. A Vector Space Model for Automatic Indexing. *Communications of the ACM*, 18, 613-620.
- SANCHEZ, D., MARTÍN-BAUTISTA, M. J. & BLANCO, I. 2008. Text Knowledge Mining: An Alternative to Text Data Mining. *IEEE International Conference on Data Mining Workshops*.
- SEMBOK, T. M. T. & BAKAR, Z. A. 2011. Effectiveness of Stemming and n-grams String Similarity Matching on Malay Documents. *International Journal of Applied Mathematics and Informatics*, 5.
- SHANTA, V., KRISHNAMURTHI, S., GAJALAKSHMI, C. K., SWAMINATHAN, R. & RAVICHANDRAN, K. 2000. Epidemiology of cancer of the cervix: global and national perspective. *Journal Indian Medicine Association* 98, 49-52.
- SIVIC, J. 2009. Efficient visual search of videos cast as text retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31, 591 - 605.
- SMUTS, H., VAN DER MERWE, A., LOOCK, M. & KOTZÉ, P. 2009. A Methodology and Methodology for Knowledge Management System Implementation. *Annual Conference of South African Institute of Computer Scientists and Information Technologists (SAICSIT)*.
- SOIBEMAN, L., WU, J., CALDAS, C., BRILKIS, I. & LIN, K.-Y. 2008. Management and analysis of unstructured construction data types. *Advanced Engineering Informatics*, 22(1).

- SONG, S.-K., OH, H.-S., MYAENG, S. H., CHOI, S.-P., CHUN, H.-W., CHOI, Y.-S. & JEONG, C.-H. 2011. Procedural knowledge extraction on MEDLINE abstracts. *Proceedings of the 7th international conference on Active media technology (AMT)*.
- SOUICY, P. & MINEAU, G. W. 2005. Beyond TFIDF Weighting for Text Categorization in the Vector Space Model. *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI)*.
- SRIKANT, R. & AGRAWAL, R. 1996. Mining Sequential Patterns: Generalizations and Performance Improvements. *5th International Conference on Extending Database Technology (EDBT)*.
- SRIVASTAVA, A. & SAHAMI, M. 2009. *Text Mining: Classification, Clustering, and Applications*, Boca Raton, FL: CRC Press.
- STAVRIANOU, A., ANDRITSOS, P. & NICOLOYANNIS, N. 2007. Overview and Semantic Issues of Text Mining. *SIGMOD Record*, 36.
- STOLCKE, A. & SEGAL, J. 1994. Precise N-gram Probabilities from Stochastic Context-Free Grammars. *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*
- STUDER, R., BENJAMINS, V. R. & FENSEL, D. 1998. Knowledge Engineering: Principles and methods. *Data Knowledge Engineering*, 25, 161-197.
- SUTCLIFFE, A. G. 1998. Scenario-based requirements analysis. *Journal of Requirements Engineering*, 3, 48-65.
- SUTCLIFFE, A. G., MAIDEN, N. A. M., MINOCHA, S. & MANUEL, D. 1998. Supporting scenario-based requirements engineering. *IEEE transactions on software engineering*, 24, 1033-1196.
- SVATEK, V., RAUCH, J. & RALBOVSKÝ, M. 2005. Ontology-Enhanced Association Mining. *In: Proceedings of the 2005 joint international conference on Semantics, Web and Mining*, pp 163-179
- TELLEX, S., KATZ, B., LIN, J., FERNANDES, A. & MARTON, G. 2003. Quantitative Evaluation of Passage Retrieval Algorithms for Question Answering. *SIGIR'03*.
- USCHOLD, M. & GRUNINGER, M. 1996. Ontologies: Principles, Methods and Applications. *Knowledge Engineering Review*.
- USZKOREIT, J., PONTE, J. M., POPAT, A. C. & DUBINER, M. 2010. Large Scale Parallel Document Mining for Machine Translation. *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*.

- VAKA, H. G. G. & MUKHOPADHYAY, S. 2009. Knowledge Extraction and Extrapolation Using Ancient and Modern Biomedical Literature. *International Conference on Advanced Information Networking and Applications Workshops*.
- VAN DER AALST, W. M. P., RUBIN, V., VERBEEK, H. M. W., VAN DONGEN, B. F., KINDLER, E. & GÜNTHER, C. W. 2008. Process Mining: A Two-Step Approach to Balance Between Underfitting and Overfitting. *Software and Systems Modeling*. Springer Berlin / Heidelberg.
- VAN DER AALST, W. M. P. & WEIJTER, A. J. M. M. 2003. Process Mining: A Research Agenda. *Computers and Industry*, 53, 231-244.
- VAN DER AALST, W. M. P. & WEIJTERS, A. J. M. M. 2004. Process mining: a research agenda. *Computers in Industry*, 53, 231 - 244.
- VARGAS-VERA, M., MOTTA, E., DOMINGUE, J., SHUM, S. B. & LANZONI, M. 2001. Knowledge Extraction by using an Ontology-based Annotation Tool. *Workshop on Knowledge Markup & Semantic Annotation in the first Conference of Knowledge Capture (K-CAP'01)*.
- VINCIARELLI, A. 2005. Noisy Text Categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27.
- WAN-KADIR, W. M. N. & LOUCOPOULOS, P. 2004 Relating evolving business rules to software design. *Journal of Systems Architecture*, 50, 367 - 382
- WANG, J., ZHOU, Y., LI, L., HU, B. & HU, X. 2009. Improving Short Text Clustering Performance with Keyword Expansion. *The Sixth International Symposium on Neural Networks*.
- WENNERBERG, P. O. 2005. Ontology Based Knowledge Discovery in Social Networks. JRC Joint Research Center, European Commission, Institute for the Protection and Security of the Citizen (IPSC).
- WESSEL, K. & RENEE, P. 1996. Viewing stemming as recall enhancement. *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*.
- WILLIAMS, G. J. & HUANG, Z. 1996. *Modelling the KDD Process* [Online]. TR DM 96013, CSIRO Division of Information Technology. Available: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.4> [Accessed 12 January 2012].
- WIMALASURIYA, D. C. & DOU, D. 2009. Using Multiple Ontologies in Information Extraction. In: *The 18th ACM Conference on Information and Knowledge Management (CIKM)*.

- WIMALASURIYA, D. C. & DOU, D. 2010. Ontology-based information extraction: An introduction and a survey of current approaches. *Journal of Information Science*, 3, 306-323.
- WOOD, M. J. & ROSS-KERR, J. C. 2006. *Basic steps on planning nursing research from question to proposal*, Jones and Bartlett Publisher.
- WU, F. & WELD, D. S. Year. Autonomously semantifying wikipedia. *In: Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management*, 2007. ACM, New York.
- WU, X., KUMAR, V., QUINLAN, J. R., GHOSH, J., YANG, Q., MOTODA, H., MCLACHLAN, G. J., NG, A., LIU, B., YU, P. S., ZHOU, Z.-H., STEINBACH, M., HAND, D. J. & STEINBERG, D. 2008. Top 10 Algorithms in Data Mining. *Knowledge and Information Systems*, 14(1), 1-37.
- YANG, C. C., CHEN, H. & HONG, K. 2003. Visualization of large category map for Internet browsing. *Decision Support Systems*, 35 (1), 89–102.
- YANG, Y. & PEDERSON, J. O. 1997. A Comparative Study on Features selection in Text Categorization. *Proceedings of the 14th international conference on Machine Learning (ICML)*. Nashville, Tennessee.
- YAO, Y., ZENG, Y., ZHONG, N. & HUANG, X. 2007. Knowledge Retrieval (KR). *IEEE/WIC/ACM International Conference on Web Intelligence*.
- YE, J., COYLE, L., DOBSON, S. & NIXON, P. 2007. Ontology-based models in pervasive computing systems. *The Knowledge Engineering Review*, 22, 315-347.
- YILDIZ, B. & MIKSCH, S. 2007. Ontology-Driven Information Systems: Challenges and Requirements. *International Conference on Semantic Web and Digital Libraries*.
- ZHANG, C. & ZHANG, S. 2002. *Association Rule Mining: Models and Algorithms*, Springer.
- ZHANG, D. & NUNAMAKER, J. F. 2004. A Natural Language Approach to Content-Based Video Indexing and Retrieval for Interactive E-Learning. *IEEE Transactions on Multimedia*, 6.
- ZHAO, Q. & BHOWMICK, S. S. 2003. Sequential Pattern Mining: A Survey. Technical Report, CAIS, Nanyang Technological University, Singapore, No. 2003118.
- ZUKERMAN, I., RASKUTTI, B. & WEN, Y. 2003. Query Expansion and Query Reduction in Document Retrieval. *In: The 15th IEEE International Conference on Tools with Artificial Intelligence*.