University of Warsaw
Faculty of Mathematics, Informatics and Mechanics

Towards the Semantic Text Retrieval for Indonesian

Doctoral dissertation

Gloria Virginia

Promotor:
Dr. Hab. Hung Son Nguyen

May, 2013

Declaration

Declaration by the author of the dissertation:
I declare that this dissertation was written by me alone.

.....	.....
date	signature of the author

The thesis statement:
This dissertation is ready for evaluation by the reviewers.

.....	.....
date	signature of the promotor

1. Reviewer:

2. Reviewer:

3. Reviewer:

Day of the defense:

Signature from head of PhD committee:

.....

Abstract

Indonesia is the fourth most populous country in the world and the Asosiasi Penyelenggara Jasa Internet Indonesia (Indonesian Internet Service Providers Association) recorded that Indonesian Internet subscribers and users has been growing rapidly every year. These facts should encourage research such as computer linguistic and information retrieval for Indonesian language which in fact has not been extensively investigated.

The research aims to investigate the tolerance rough sets model (TRSM) in order to propose a framework for a semantic text retrieval system. The proposed framework is intended for Indonesian language specifically hence we are working with Indonesian corpora and applying tools for Indonesian, e.g. Indonesian stemmer, in all of the studies. Cognitive approach is employed particularly during data preparation and analysis. An extensive collaboration with human experts is significant on creating a new Indonesian corpus suitable for our research. The performance of an ad hoc retrieval system becomes the starting point for further analysis in order to learn and understand more about the process and characteristic of TRSM, despite comparing TRSM with other methods and determining the best solution. The results of this process function as the guidance for computational modeling of some TRSM's tasks and finally the framework of a semantic information retrieval system with TRSM as its heart.

In addition to the proposed framework, this thesis proposes three methods based on TRSM, which are the automatic tolerance value generator, thesaurus optimization, and lexicon-based document representation. All methods were developed by the use of our own corpus, namely ICL-corpus, and evaluated by employing an available Indonesian corpus, called Kompas-corpus. The evaluation on the methods achieved satisfactory results, except

for the compact document representation method; this last method seems to work only in limited domain.

To my family.

Acknowledgements

I learned that pursuing a doctoral study requires more than brain and desire. It needs constant stamina to keep it on the track which is impossible to be realized without supports from so many people. The result reaches out of academic matters, internalized into personality. I do realize that this is a beginning of another more exciting journey.

My sincere gratitude goes to Dr. Hab. Hung Son Nguyen whose energy keeps encouraging me to move forward. His guidance, patience, and full confidence have lightened every steps of mine on this journey. I have received a wise reminder from Prof. Andrzej Skowron in order to think globally but act locally, which has brought me to this point. To Dr. Marcin Sczuka who is always ready for any assistance, I am more than grateful. I am filled with gratitude for the opportunities to have discussion with Prof. Piotr Brykczyński in philosophy topics as well as Dr. Danar Hadi who also read the philosophy section of this thesis.

It was a privilege to be a beneficiary of Erasmus Mundus Mobility with Asia (EMMA) and thus fully supported financially for 34 months during my doctoral study. Pertaining to EMMA, I would like to thank Klementyna Kielak who had taken care of me since the very first day of mine in Warsaw. Become part of the University of Warsaw (UW), I was first affiliated with the Interdisciplinary Center for Mathematical and Computational Modeling (ICM) and had experience to be around people with similar passion, i.e. to do research. For this, especially I thank Prof. Marek Niezgódka and Dr. Anna Trykozko. Most of my times were spent in Faculty of Mathematics, Informatics, and Mechanics (MIMUW) which provided me with a supportive environment as a doctoral student; I am thankful for that, in particular

of the service and the kindness of Maria Gamrat. It has been a great advantage to be acquainted with people in SYNAT (System Nauki i Techniki) project, who always welcome me and are exposed to any discussion. There were several circumstances related to publication of scientific article and attending international conference which were supported by the SYNAT project. I do appreciate Duta Wacana Christian University (DWCU) who has been standing behind me since the beginning and endowed me with funds particularly in the last year of my study.

I would like to thank Sławomir Kolasiński, Andrzej Janusz, and Wojciech Świeboda in particular as friends in the similar struggle in UW. Thank you for all of the sharing.

I have been accompanied by Indonesian community in Poland at large. I am indebted to Devy Tarida Augustyn and Tresya Bedkowska who become very good friends of mine.

I am blessed to have a supportive family who are willing to strive for us; my beloved husband and son who love me unconditionally. For my parents, my private supervisors, thank you for your never-ending love and trust. The last but not the least, my Lord, I thank you for every single thing in my life.

Contents

List of Figures

LIST OF FIGURES

List of Tables

LIST OF TABLES

Chapter 1

Introduction

1.1 Information Retrieval

The percentage of individuals using the Internet continues to grow worldwide and in developing countries the numbers doubled between 2007 and 2011[1]. Accessing information by utilizing search systems becomes one habitual activity of million of people in facilitating their business, education, and entertainment in their daily life. The applications, such as web search engines which providing access to information over the Internet, are the most usual applications heavily use information retrieval (IR) service.

Information retrieval is concerned with representing, searching, and manipulating large collections of electronic text and other human-language data (1, p. 2). Clustering systems, categorization systems, summarization systems, information extraction systems, topic detection systems, question answering systems, and multimedia information retrieval systems are other applications utilize IR service.

The main task of information retrieval is to retrieve relevant documents in response to a query (2, p. 85). In a common search application, an *ad hoc* retrieval mode is applied in which a query is submitted (by a user) and then evaluated against a relatively static document collection. A set of query identifying the possible interest to the user may also be supplied in advanced and then evaluated against newly created or discovered documents. This operational mode where the queries remain relatively static is called *filtering*..

[1]Key statistical highlights: ITU data release June 2012. URL: `http://www.itu.int`. Accessed on 25 October 2012.

Documents (i.e. electronic texts and other human-language data) are normally modeled based on the positive occurrence of words while the query is modeled based on the positive words of interest clearly specified. Both models then are examined in similarity basis using a devoted ranking algorithm and the output of information retrieval system (IRS) will be an ordered list of documents considered pertinent to the query at hand.

In the keyword search technique commonly used, the similarity between documents and query is measured based on the occurrence of query words in the documents. Thus, if the query is given by a user, then the relevant documents are those who contain literally one or more words expressed by him/her. The fact is, text documents (and query) highly probable come up in the form of natural language. While human seems effortless to understand and construct sentences, which may consist of ambiguous or colloquial words, it becomes a big challenge for an IRS. The keyword search technique is lack of capability to capture the meaning of words, wherefore the meaning of sentences, semantically on documents and query because it represents the information content as a syntactical structure which is lack of semantical relationship. For example, a document contains words *choir*, *performance*, and *ticket* may talk about a *choir concert*, in spite of the fact that the word *concert* is never mentioned on that particular document. When a user inputs the word *concert* to define his/her information need, the IRS which approximate the documents and query in a set of occurrence words may deliver lots of irrelevant results instead of corresponding documents.

We may expect better effectiveness to IRS by mimicking the human capability of language understanding. We should move from *keyword* to *semantic* search technique, hence the semantic IRS.

1.2 Philosophical Background

Semantic is the study of linguistic meaning (3, p. 1). Sentence and word meaning can be analyzed in terms of what speakers (or utterers) mean of his/her utterances[2] (4). With regard to the intended IRS, we devoted our study to written document, which

[2]Utterances may include sound, marks, gesture, grunts, and groans (anything that can signal an intention)

might be seen as an extension of speech. Hence, a text semantic retrieval system should know to some extent the meaning of words of texts being processed, so to speak.

1.2.1 Searl's and Grice's Accounts on Meaning

1.2.1.1 Intentionality

Among others, Searl (5) and Grice (6) have been on a debate, namely the role of intentionality in the theory of meaning. Intentionality (in Latin: *intendere*; meaning aiming in a certain direction, directing thoughts to something, on the analogy to drawing a bow at a target) has been used to name the property of minds of having content, *aboutness*, being *about* something (7, p. 89). Thus, mental states such as beliefs, fears, hopes, and desires are intentional because they are directed at an object. For example, if I have a belief, it must be a belief of something, or if I have a fear, it must be a fear of something. However, mental states such as undirected anxiety, depression and elation are not intentional because they are undirected at an object (e.g. I may anxious without being anxious about anything), but the directed cases (e.g. I am anxious about something) are intentional.

In addition that intentional is directed, another important characteristic of intentional was proposed by Searl (5) that every intentional state consists of an *intentional content* in a *psychological mode*. The intentional content is a whole proposition which determines a set of condition of satisfaction and the psychological mode (e.g. belief, desire, promise) determines a direction of fit (i.e. mind-to-world or world-to-mind) of its propositional content. An example should make this clear: If I make an assertive utterance that 'it is raining', then the content of my belief is 'it is raining'. So, the conditions of satisfaction are 'it is raining', and not, for example, that the ground is wet or the water is falling out of the sky[3]. And, in my assertive utterance, the psychological mode is a 'belief' of the state in question, so the direction of fit is 'mind-to-world'[4].

[3]The reason is, in the context of speech act, we do not concern about whether the belief of a speaker is true or not, rather we concern about the intention of speaker what he/she wants to represent by his/her utterance. Thus, it might be the case that a speaker represents his/her false belief as a true belief to the audience, e.g. a speaker utters 'it is raining', while in fact 'it is a sunny day'.

[4]In other words, 'the mind to fit the world'. It is because a belief is like a statement, can be true or false; if the statement is false then it is the fault of the statement, not the world. The *world-to-mind* direction of fit is applied for the psychological mode such as desire or promise; if the promise is broken, it is the fault of the promiser.

Further, Searl claimed (5, p. 19-21) that intentional contents do not determine their condition of satisfaction in isolation, rather they are internally related in a holistic way to: *a*) other intentional contents in the Network of intentional states; and *b*) a Background of nonrepresentational mental capacities. The following is Searl's example to describe the role of Network: Suppose there is a man who forms the intention to run for the Presidency of the United States. In order that his desire be a desire to run for the Presidency he must have a whole lot of beliefs such as: the belief that the United States is a republic, that it has a presidential system of government, that it has periodic elections, and so on. And he would normally desire that he receives the nomination of his party, that people work for his candidacy, that voters cast votes for him, and so on. So, in short, we can see that his intention 'refers' to these other intentional states.

The Background is the set of practice, skills, habits, and stance that enable intentional contents to work in the various ways. Consider these sentences: 'Berto opened his book to page 37' and 'The chairman opened the meeting'. The semantic content contributed by the word 'open' is the same in each sentence, but we understand the sentences quite differently. It is because the differences in the Background of practice (and in the Network) produce different understanding of the same verb.

1.2.1.2 Meaning

Language is one of the vehicles of mental states, hence linguistic meaning is a form of derived intentionality.

According to Searle, *meaning* is a notion that literally applies to sentences and speech acts. He mentioned that the problem of meaning in its most general form is the problem of how do we get from the physics to the semantics. For this purpose, there are two aspects to meaning intentions: *a*) the intention to represent; and *b*) the intention to communicate. Here, representing intention is prior to communication intention and the converse is not the case. Hence, we can intend to represent something without intending to communicate it, but we cannot intend to communicate something without intending to represent it before. So to speak, in order to inform anyone that 'it is raining' we need to represent it in our mind that 'it is raining' then utter it. Conversely, we cannot inform anyone anything, i.e. that 'it is raining', when we do not make any representation of the state of affairs of the weather in our mind.

For Grice, when a speaker *mean* something by an utterance, he/she intends to produce certain effects on his/her audience and intends the audience to recognize the intention behind the utterance. By this definition, it seems that Grice has overlooked the intention to represent and overemphasized the intention to communicate. However, a careful analyses showed that Grice's account goes along with Searl's account (8), i.e. representing intention is prior to communication intention. Moreover, Grice definition makes a point that a successful speech act is both meaningful and communicative, i.e. the audience understands nothing when the audience does not recognize the intention behind the utterance, which can be happen when the speaker makes an utterance without intending to mean anything or fails to communicate it.

1.2.2 The Importance of Knowledge

Based on Searl's and Grice's accounts, it should be clear that there is distinction between intentional content and the form of its externalization. To ask for the meaning is to ask for an intentional content that goes with the form of externalization (5). It is maintained that for a successful speech act, a speaker normally choose an expression which is conventionally fixed, i.e. by the community at large, to convey a certain meaning. Thus, before the selection process of appropriate expressions, it is fundamental for a speaker to know about the expression in order to produce an utterance, and consequently the audience is required to be familiar with those conventional expressions in order to understand the utterance.

We may infer now that Searl's and Grice's accounts pertaining the *meaning* suggest knowledge for language production and understanding. This knowledge should consists of concepts who are interrelated and commonly agreed by the community. The communication is satisfied when both sides are active participants and the audience experiences effects at some degree.

1.3 Challenges in Indonesian

1.3.1 Indonesian Studies

Knowledge specifically for Indonesian is fundamental for a semantic retrieval system which processing Indonesian texts. The implication of this claim is far reaching, in particular because each language is unique. There are numerous aspects of monolingual

text retrieval should be investigated for Indonesian, those including indexing and relevance assessment process, i.e. tasks such as tokenization, stopping, stemming, parsing, and similarity functions, are few to mention.

Considerable effort with regard to information retrieval for Indonesian is showed by a research community in University of Indonesia (UI) since mid of 1990s. They reported (9) that their studies range in area of computational lexicography (i.e. creating dictionary and spell-checking), morphological analysis (i.e. creating stemming algorithms and parser), semantic and discourse analysis (i.e. based on lexical semantics and text semantic analysis), document summarization, question-answering, information extraction, cross language retrieval, and geographic information retrieval. Other significant studies conducted by Asian which proposed an effective techniques for Indonesian text retrieval (10) and published the first Indonesian testbed (11). It is worth to mention that despite the long list of works ever mentioned, only limited number of the results is available publicly and among those Indonesian studies, it is hardly to find a work pertaining to automatic ontology constructor specifically.

1.3.2 Indonesian Speakers

The latest data released by Statistics Board of Indonesia (BPS-Statistics Indonesia)[5] pertaining the population of Indonesia, showed that the number reached 237.6 million for the 2010 census. With the population growth rate 1.49 percent per year, the estimation of Indonesia population in 2012 is 245 million. This number ranked Indonesia on the forth most populous country in the world after China, India, and United States[6].

The incredible number is not only related to the population. Indonesia, which is an archipelago country, has around 6,000 inhibited island over 17,508[7]. Administratively, Indonesia consists of 33 provinces in which there are number of ethnics groups comes from each province which has its own regional language; according to Sneddon (12, p. 196), Indonesia has about 550 languages which is roughly one-tenth of all the languages in the world today. However, chosen as the national language, *Bahasa Indonesia* or Indonesian language is taught at all level of education and officially used in

[5]BPS-Statistics Indonesia. URL: `http://www.bps.go.id/`. Accessed on 25 October 2012.

[6]July 2012 estimation of The World Factbook. URL: `https://www.cia.gov`. Accessed on 25 October 2012.

[7]Portal Nasional Indonesia (National Portal of Indonesia). URL: `http://www.indonesia.go.id`. Accessed on 25 October 2012.

domains of formal activity, e.g. mass media, all government business, education, and law. Nowadays, most Indonesians are proficient in using the language; the number of speaker of Indonesian is approaching 100 percent (12, page 201). Therefore, it is not overstated to consider Indonesian language as one of the large number of speakers in the world.

1.3.3 Indonesian Internet Users

Another significant challenge pertains to the growth of Internet users. As the global trend, the percentage of individuals using the Internet continues to grow worldwide and in developing countries the numbers doubled between 2007 and 2011[8]. For Indonesia, the Internet World Stats[9] recorded that there are about 55 million internet users (with 22.4% penetration rate) and 43 million Facebook users (with 17.7% penetration rate) as of Dec. 31, 2011. Figure 1.1 shows the rapid growth of internet users in Indonesia during some previous years[10]. These facts are some indicators of the digital media usage proliferation in Indonesia which is considered to keep on growing.

1.4 The Thesis

1.4.1 Tolerance Rough Sets Model

Basically, an information retrieval system consists of three main tasks: *(1)* modeling the document; *(2)* modeling the query; and *(3)* measure the degree of correlation between document and query models. Thus, the endeavor of improving an IRS revolves around those three tasks. One of the effort is a method called tolerance rough set model (TRSM) which has performed positive results on some studies pertaining to information retrieval. In spite of the fact that TRSM does not require complex linguistic process, it has not been investigated at large extent.

Since it was formulated, tolerance rough sets model (TRSM) is accepted as a tool to model a document in a *richer* way than the base representation which is represented by

[8]Key statistical highlights: International Telecommunication Union (ITU) data release June 2012. URL: `http://www.itu.int`. Accessed on 25 October 2012.

[9]URL: `http://www.internetworldstats.com`. Accessed on 25 October 2012.

[10]The graph was taken from the International Telecommunication Union (ITU). URL: `http://www.itu.int/ITU-D/ict/statistics/explorer/index.html`. Accessed on 25 October 2012.

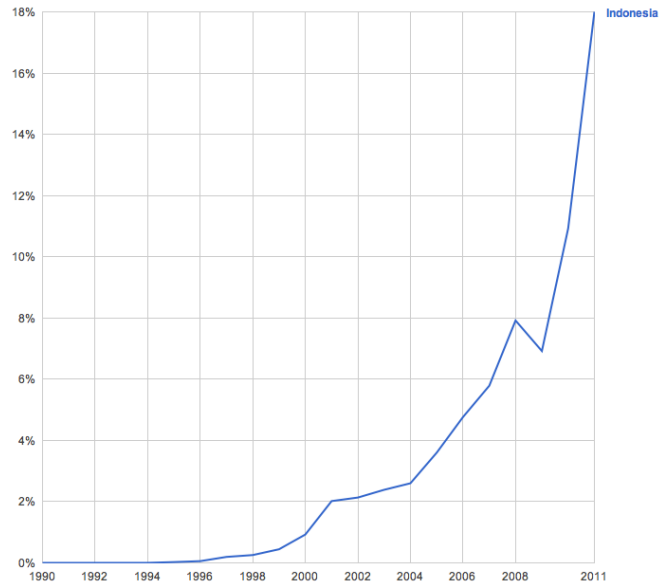

Figure 1.1: The growth of internet users in Indonesia - The figure shows the growth of internet users in Indonesia since 1990 to 2011. On 2011, the penetration rate was close to 18%.

a vector of TF*IDF-weight terms[11] (let us call it TFIDF-representation). The richness of the document representation produced by applying the TRSM (let us call it TRSM-representation) is indicated by the number of index terms put into the model. That is to say, there are more terms belong to TRSM-representation than its base representation.

The power of TRSM is grounded on the knowledge, i.e. thesaurus, which is comprised by index terms and the relationships between them. In TRSM, each set of terms considered as semantically related with a single term t_j is called the tolerance class of a term $I_\theta(t_j)$, hence the thesaurus contains tolerance classes of all index terms. The semantic relatedness is signified by the terms co-occurrence in a corpus in which a tolerance value θ is set to define the threshold of co-occurrence frequency.

1.4.2 Research Objective and Approach

The research aims to investigate the tolerance rough sets model in order to propose a framework for a semantic text retrieval system. The proposed framework is intended for Indonesian language specifically hence we are working with Indonesian corpora and

[11] Appendix A provides an explanation about the TF*IDF weighting scheme.

applying tools for Indonesian, e.g. Indonesian stemmer, in all of the studies.

The researches of TRSM ever conducted pertaining to information retrieval have focused on the system performance and involved a combination of mathematics and engineering in their studies (13, 14, 15, 16, 17). In this thesis, we are trying to look at TRSM from a quite different viewpoint. We are going to do empirical studies involving observations and hypotheses of human behavior as well as experimental confirmation. According to the Artificial Intelligence (AI) view, our studies follow a human-centered approach, particularly the cognitive modeling[12], instead of the rationalist approach (19, p.1-2). Analogous to two faces in a coin, both approaches would result in a comprehensive perspective of TRSM.

In implementing the cognitive approach, we start our analysis from the performance of an ad hoc retrieval system. It is not our intention to compare TRSM with other methods and determine the best solution. Rather, we will take the benefit of the experimental data to learn and understand more about the process and characteristic of TRSM. The results of this process function as the guidance for computational modeling of some TRSM's tasks and finally the framework of a semantic IRS with TRSM as its heart.

1.4.3 Thesis Structure

Our research falls under the information retrieval umbrella. The following chapter provides an explanation about the main tasks of information retrieval and the semantic indexing in order to establish a general understanding of semantic IRS.

Several questions are generated in order to assist us to scrutinize the TRSM. The issues behind the questions should be apparent when we proceed into the nature of TRSM that would be exposed on theoretical basis in Chapter 3. We have selected four subjects of question and will discuss them in the following order:

1. **Is TRSM a viable alternative for a semantic IRS?**
   The simplicity of characteristic and positive result of studies makes TRSM an intriguing method. However, before moving any further, we need to ensure that

[12]The cognitive modeling is an approach employed in the Cognitive Science (CS). Cognitive science is an interdisciplinary study of mental representations and computations and of the physical systems that support those processes (18, p.xv).

TRSM is reasonable to be the ground floor of the intended system. This issue will be the content of Chapter 4.

2. **How to generate the system knowledge automatically?**
   The richer representation of document yielded by TRSM is achieved fundamentally by means of a knowledge, which is a thesaurus. The thesaurus is manually created, in the sense that a parameter, namely *tolerance value* θ, is required to be determined by hand. In Chapter 5 we would propose an algorithm to resolve the matter in question, i.e. to select a value for θ automatically.

3. **How to improve the quality of the thesaurus?**
   The thesaurus of TRSM is generated based on a collection of text documents functions as a data source. In other words, the quality of document representation should depend on the quality of data source at some degree. Speaking of which, the TRSM basically works based on the co-occurrence data, i.e. the raw frequency of terms co-occurrence, and it arises an assumption that other co-occurrence data might bring a benefit for the effort to optimize the thesaurus. These presumptions would be reviewed and discussed in Chapter 6.

4. **How to improve the efficiency of the intended system?**
   The TRSM-representation is claimed to be richer in the sense that it consists of more terms than the base representation. Despite the fact that the terms of TRSM-representation are semantically related, more terms on document vector results in more cost of computation. In other words, system efficiency becomes the trade-off. We came into an idea of a compact document representation that would be explained in Chapter 7.

This thesis proposes three methods based on TRSM for the mentioned problems. All methods, which are discussed in Chapter 5 to 7, were developed by the use of our own corpus, namely ICL-corpus, and evaluated by employing an available Indonesian corpus, called Kompas-corpus[13]; Chapter 8 describes the evaluation process. The evaluation on the methods achieved satisfactory results, except for the compact document representation method; this last method seems to work only in limited domain.

[13]Explanation about all corpora used in this thesis is available in Appendix C.

The final chapter provides our conclusion of the research as well as discussion of some challenges that lead to advance studies in the future.

1.4.4 Contribution

The main contribution of this thesis is the modular framework of text retrieval system based on TRSM for Indonesian. Pertaining to the framework, we introduced novel strategies, which are the automatic tolerance value generator, thesaurus optimization, and lexicon-based document representation. An other contribution is a new Indonesian corpus (ICL-corpus), accompanied by a corpus consists of keywords defined by human experts (WORDS-corpus), in which both follow the format of Text REtrieval Conference (TREC)[14] (20) and ready to be used for an ad hoc evaluation of IRS. These contributions should open wider research directions pertinent to information retrieval.

[14]TREC is a forum for IR community which provides an infrastructure necessary to evaluate an IR system on a broad range of problems. URL: `http://trec.nist.gov/`.

Chapter 2

Semantic Information Retrieval

2.1 Information Retrieval Models

The main problem of information retrieval system is the issue of determining the relevancy of a document with regard to the information need. The decision whether documents are relevant or not relies on the ranking algorithm being used which plays the role of calculating the degree of association between documents and the query as well as defining the order of documents by its degree of association, in which the top documents are considered as the most relevant ones. In order to work, a ranking algorithm considers fundamental premises which are a set of representations of documents in given collection D, a set of representations for user information needs (user queries) Q, and a framework for modeling document/query representation $\mathcal{F}$. These basic premises, together with the ranking function R, determines the IR model as a quadruple $[D, Q, \mathcal{F}, R]$ (21, p. 23).

Baeza-Yates and Ribeiro-Neto (21) structured 15 IR models covered in their book into a taxonomy as well as discussed them theoretically and bibliographically. Figure2.1 presents the summary of the taxonomy. A clear distinction is made on the way a user pursues information: by searching or by browsing. While browsing, a user might explore a document space which is constructed in a flat, hierarchical, or navigational organization. Another user might prefer to submit a query to the system and put the burden of searching process to the system. In order to accomplish the task, the system could analyze each document by reference to the document's content only or combination between the content and the structure of document. The *structured model*

considers the latter while the *classic model* focusses on the former. The classic model is differentiated into three models with regard to the document representation: boolean, vector, and probabilistic. Respectively, in Boolean and probabilistic models, a document is represented based on set theory and probability theory, while vector model will represent a document as a vector in a high-dimensional space.

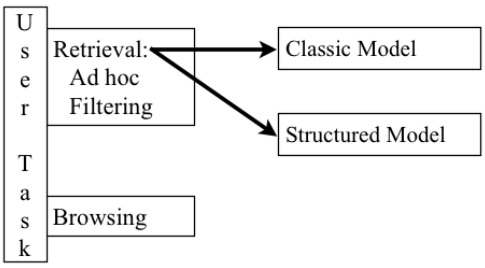

Figure 2.1: A taxonomy of IR models - A summary of the IR models taxonomy structured by Baeza-Yates and Ribeiro-Neto.

In this thesis, we applies the classic vector model where document and query are represented as vectors in a high-dimensional space and each vector corresponds to a term in the vocabulary of the collection. The framework then is composed of a high-dimensional vectorial space and the standard linear algebra operations on vectors. The association degree of documents with regard to the query is quantified by the cosine of the angle between these two vectors[1].

2.2 The Main Tasks of Information Retrieval

Suppose each text document conveys meaning expressed in the form of written words chosen specifically and subjectively by the writer. When text documents are fed into an IRS who employs vector space model, the text documents would be transformed into vectors of a space whose dimension is consistent with the number of index terms in the corpus. A query which conveys information need of a user could be considered as a pseudo-document, thereby analogous scenario and activities occur at user side. In the searching process, a ranking algorithm works over these two representations by measuring the degree of correlation between them.

[1]Appendix B provides explanation about Cosine similarity measure as a document ranking algorithm.

By reference to its process, IR consists of three main tasks which are figured in Fig. 2.2 as filled rectangles: document modeling, query modeling, and matching process. The figure reflects that a successful matching process has two requirements: *(1)* models common to both query and document; and *(2)* system capability to construct a model which represent the information need of the user as well as the content of text document.

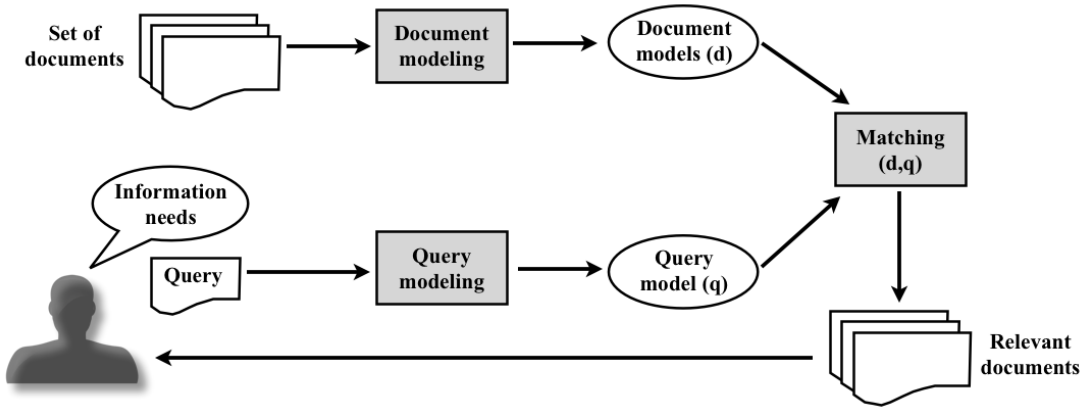

Figure 2.2: The main tasks of information retrieval - Information retrieval consists of three tasks: *(1)* document modeling; *(2)* query modeling; and *(3)* matching process.

Explained in the previous chapter, Searl's and Grice's accounts on meaning suggest knowledge shared by the speaker and its audience (i.e. the user and the system) for a successful communication. Suppose the IRS has knowledge corresponding at some degree to human, still the distinction between intentional content and the form of its externalization rises some complexity for IRS in order to construct representations of user's information need (in the query) and of author's idea (in the document). Language production and understanding are capabilities of most human, achieved through learning activities during his/her life and supported by the biologically mechanism genetically endowed (22), while none of those capabilities and support possessed by the system naturally. Reduced meaning retained by the representations of user's information need and of document's content become the consequence. It is highly probable that user's satisfaction of proper information with regard to his/her need then is sacrificed.

2.3 Semantic Indexing

Indexing is a process to construct a data structure over the text to speed up searching (21, p. 191). The major data structure in IRS is inverted index (or *inverted file*) which provides a mapping between terms and their locations of occurrence in a text collection (1, p. 33).

The first paragraph of this chapter explained that in order to construct a model of IR, the representation of document (i.e. document indexing) as well as query should be first resolved before specifying the framework; and with these basis, an appropriate ranking function is determined. For a semantic IRS, shifting from traditional indexing into the semantic indexing hence becomes the first consideration. In case that the conventional retrieval strategies employ the bag-of-words representation of document and match directly on keywords, then the semantic indexing requires an enrichment of representation such that the IRS works with bag-of-concepts representation of documents and computes the concept similarity.

Several techniques function for enrichment of document representation are latent semantic indexing (LSI), explicit semantic analysis (ESA), and extended tolerance rough sets model (extended TRSM). These three techniques apply the classic vector space model (VSM) and thus it is possible to use conventional metrics (e.g. Cosine) in matching task. Further, they do not rely on any human-organized knowledge.

LSI, ESA, and extended TRSM naturally use statistical co-occurrence data in order to enrich the document representation, however LSI works by applying singular value decomposition (SVD), ESA relies on knowledge repository (e.g. Wikipedia), and the extended TRSM is based on rough sets theory. As a technique to dimensionality reduction, LSI identifies a number of most prominent dimensions in the data, perceived as the *latent concepts* since these concepts cannot be mapped into the natural concepts manipulated by humans or the concepts generated by system. An opposite condition happens for ESA and extended TRSM, thus the entries of their vectors are *explicit concepts*.

The following sections will describe LSI, ESA, and extended TRSM in the order given. For convenience of the explanation, a matrix is used as data structure where each entry defines the association strength between document and term. The most

common measure used to calculate the strength value is the TF*IDF weighting scheme defined in Equation (A.1).

2.3.1 Latent Semantic Indexing

Latent semantic indexing introduced by Furnas et al. (23) employs singular value decomposition (SVD) in 1988. By running SVD, it approximates the term-document matrix into a lower dimensional space hence removes some of the noise found in the document and locates two documents with similar semantic (whether or not they have matching terms) close to one another in a multi-dimensional space (24).

Running the SVD means that a term-document matrix A is decomposed into the product of three other matrices such that

$$A_{m\times n} = U_{m\times s} D_{s\times s} V^T_{s\times n}. \tag{2.1}$$

Matrix U is the left singular vectors matrix whose columns are eigenvectors of the AA^T and holds the coordinates of term vectors. Matrix V is the right singular vectors matrix whose columns are eigenvectors of the A^TA and holds the coordinates of document vectors. Matrix D refers to a diagonal matrix whose elements are the singular values of A, sorted by magnitude. m is the total number of terms, n is the total number of documents, and $s = min(m, n)$.

The *latent semantic* representation of A is developed by keeping the top k singular values of D along with their corresponding columns in U and V^T matrices. The result is a k-rank matrix A' which is closest in the least squares sense to matrix A; it contains less noisy dimensions and captures the major associational structure of the data (23). Figure 2.3 presents the schematic of SVD for matrix A and its reduced model.

With regard to a query, its vector q is treated similar to matrix A by following this rule

$$q_{1\times k} = q^T_{1\times m} U_{m\times k} S^{-1}_{k\times k}. \tag{2.2}$$

After all, the matching process between query and documents is conducted by computing the similarity coefficient between k-rank query vector q_k and corresponding columns of k-rank matrix V_k.

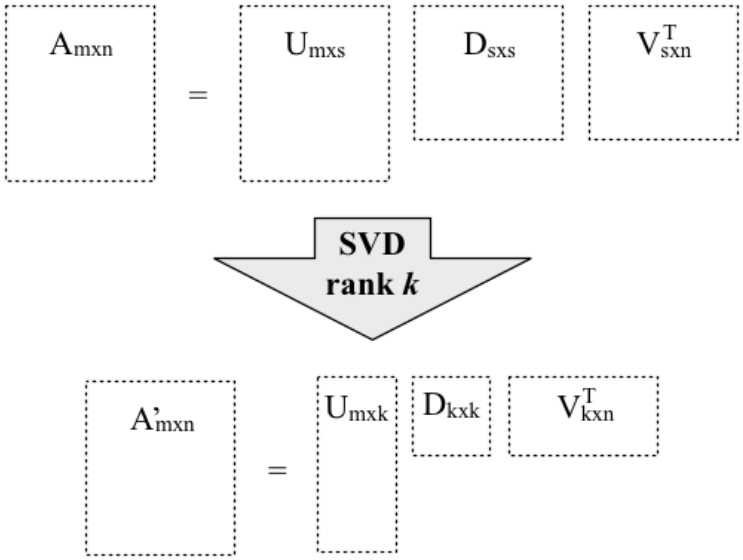

Figure 2.3: The illustration of SVD - SVD illustration of a terms-by-documents matrix A of rank k.

2.3.2 Explicit Semantic Analysis

In 2007, Gabrilovich and Markovitch introduced the notion of explicit semantic analysis (ESA) (25). Later, Wong et al. (26) showed that ESA is a variation of the generalized vector space model (GVSM)[2] who considers term correlation.

ESA represents documents and query as vectors in a high dimensional of *concept space*, instead of *term space*, thus each dimension corresponds to a concept. Each coordinate of concept vector expresses the degree of association between the document and the corresponding concept. Suppose $D = \{d_1, \ldots, d_i, \ldots, d_N\}$ is a set of documents and $T = \{t_1, \ldots, t_j, \ldots, t_M\}$ is the vocabulary of terms, then the association value u_{ik} between document d_i and concept $c_k, k \in \{1, \ldots, K\}$ is defined as

$$u_{ik} = \sum_{t_j \in T} w_{ij} \times c_{jk} \tag{2.3}$$

where w_{ij} denotes the weight of term t_j in document d_i and c_{jk} signifies the correlation between term t_j and concept c_k.

[2]Consistent with VSM, GVSM interprets index term vectors as linearly independent, however they are not orthogonal.

Equation (2.3) describes the association value as the product of weight of term in document (w_{ij}) and weight of concept in knowledge base concept (c_{jk}), hence there are two computations need to be done in advance. Basically, both computations could be done using the TF*IDF weighting scheme, however the former is calculated over a corpus functioned as the system data, while the latter is calculated over a corpus functioned as the knowledge base; thus there are two corpora functioned differently. The merge of system data's and knowledge base's weights yields a new representation for the system data, i.e. bag-of-concepts representation.

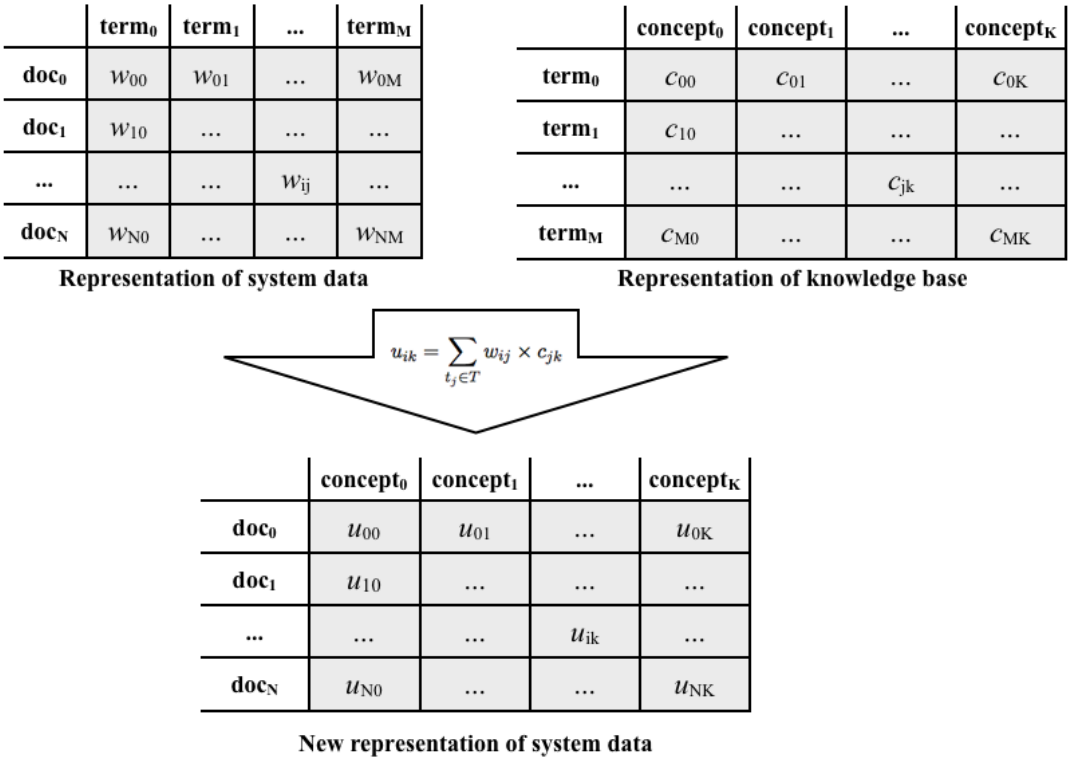

Figure 2.4: The ESA - Visualization of the semantic indexing process in ESA.

Gabrilovich and Markovitch (25) suggests Wikipedia articles for the corpus functioned as the knowledge base considering that it is a vast amount of highly organized human knowledge and undergoes constant development. However, the main reason is Wikipedia treats each description as a separate article, thus each description is perceived as a single concept. By this definition, any collection of documents is possible to be used as the external knowledge base.

Figure 2.4 shows the computation process of ESA in order to convert the bag-of-words representation of system data into the bag-of-concepts representation by utilizing natural language definition of concepts from the knowledge base.

2.3.3 Extended Tolerance Rough Sets Model

As its name, the extended TRSM is an extension of TRSM proposed by Nguyen et al. (27) in 2012. Detail explanation about TRSM is available in the following chapter, hence in this section we focus only on the extension part of TRSM.

The study of Nguyen et al. (27, 28) aimed to enrich the document representation worked in clustering task by incorporating other information than the index terms of document corpus, namely citation and semantic concept. The citation referred to the bibliography of a given scientific article while the semantic concept was constructed based on an additional knowledge source, i.e. DBpedia. Thereby, the extended TRSM was defined as a tuple

$$\mathcal{R}_{Final} = (\mathcal{R}_T, \mathcal{R}_B, \mathcal{R}_C, \alpha_n) \tag{2.4}$$

where $\mathcal{R}_{\mathcal{T}}$, $\mathcal{R}_{\mathcal{B}}$, and $\mathcal{R}_{\mathcal{C}}$ denote the tolerance spaces which are determined respectively over the set of terms T, the set of bibliography items cited by a document B, and the set of concepts in the knowledge domain C. The function $\alpha_n : P(T) \longrightarrow P(C)$ is called the *semantic association* for terms, thus α_n is the set of n concepts most associated with T_i for any $T_i \subset T$ (29).

In this model, each document $d_i \in D$ associated with a pair (T_i, B_i) is represented by a triple

$$\mathbf{U}_{\mathcal{R}}(d_i) = \{\mathbf{U}_{\mathcal{R}_T}(d_i), \mathbf{U}_{\mathcal{R}_B}(d_i), \alpha_n(T_i)\} \tag{2.5}$$

where T_i is the set of terms occurring in document d_i and B_i is the set of bibliography items cited by document d_i. The study of extended TRSM which presented with positive results indicated that the method would be effective to be realized in a real application.

It is obvious from Equation (2.4) and (2.5) that the extended TRSM accommodates different factors at once for a semantic indexing, instead of one factor such as in original TRSM as well as LSI and ESA. Further, the model is more nature considering the real life situation of information retrieval process.

Chapter 3

Tolerance Rough Sets Model

3.1 Rough Sets Theory

In 1982, Pawlak introduced a method called rough sets theory (30) as a tool for data analysis and classification. During the years, this method has been studied and implemented successfully in numerous areas of real-life applications (31). Basically, rough sets theory is a mathematical approach to vagueness which expresses the vagueness of a concept by means of the boundary region of a set; when the boundary region is empty, it is a crisp set. Otherwise, it is a rough set (32). The central point of rough sets theory is an idea that any concept can be approximated by its *lower* and *upper approximations*, and the vagueness of concept is defined by the region between its upper and lower approximations. Consider Fig. 3.1 for illustration.

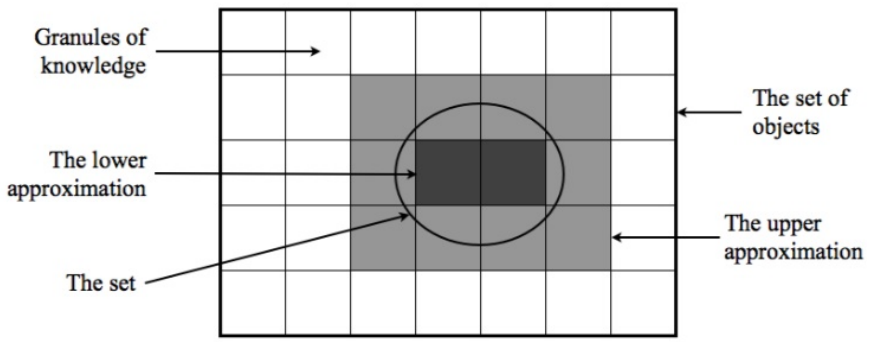

Figure 3.1: Rough Sets - Basic idea of rough sets theory as it is explained in (32)

Let us think of a concept as a subset X of a universe U, $X \subseteq U$, then in a given

approximation space $A = (U, R)$ we can denote the lower approximation of concept X as $L_A(X)$ and the upper approximations of concept X as $U_A(X)$. The boundary region, $BN_A(X)$, is the difference between the upper and lower approximations, hence

$$BN_A(X) = U_A(X) - L_A(X) \tag{3.1}$$

Let $R \subseteq U \times U$ be an *equivalence relation* that will partition the universe into *equivalence classes*, or *granules of knowledge*, thus formal definition of lower and upper approximations are

$$L_A(X) = \bigcup_{x \in U} \{R(x) : R(x) \subseteq X\} \tag{3.2}$$

$$U_A(X) = \bigcup_{x \in U} \{R(x) : R(x) \cap X \neq \emptyset\} \tag{3.3}$$

3.2 Tolerance Rough Sets Model

The equivalence relation $R \subseteq U \times U$ of classical rough sets theory required three properties (31): reflexive (xRx), symmetric ($xRy \rightarrow yRx$), and transitive ($xRy \wedge yRz \rightarrow xRz$); for $\forall x, y, z \in U$, thus the universe of an object would be divided into disjoint classes. These requirements have been showed to be not suitable for some practical applications (viz. working on text data), because the association between terms was better viewed as overlapping classes (see Fig. 3.2), particularly when term co-occurrence was used to identify the semantic relatedness between terms (14) .

The overlapping classes can be generated by a relation called *tolerance relation* which was introduced by Skowron and Stepaniuk (33) as a relation in *generalized approximation space*. The generalized approximation space is denoted as a quadruple $\mathcal{A} = (U, I, \nu, P)$, where U is a non-empty universe of objects, I is the uncertainty function, ν is the vague inclusion function, and P is the structurality function.

Tolerance Rough Sets Model (TRSM) was introduced by Kawasaki, Nguyen, and Ho in 2000 (13) as a document representation model based on generalized approximation space. In the information retrieval context, we can assume a document as a concept. Thus, implementing TRSM means that we approximate concepts determined over the set of terms T on a tolerance approximation space $\mathcal{R} = (T, I, \nu, P)$ by employing the tolerance relation.

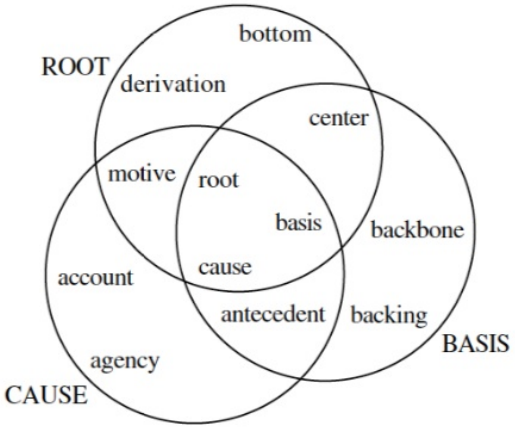

Figure 3.2: Overlapping classes - Overlapping classes between terms *root*, *basis*, and *cause* (14)

In order to generate the document representation, which is claimed to be richer in terms of semantic relatedness, the TRSM needs to create tolerance classes of terms and approximations of subsets of documents. If $D = \{d_1, d_2, ..., d_N\}$ is a set of text documents and $T = \{t_1, t_2, ..., t_M\}$ is a set of index terms from D, then the tolerance classes of terms in T is created based on the co-occurrence of index terms in all documents D. A document representation is represented as a vector of weight $d_i = \{w_{i,1}, w_{i,2}, ..., w_{i,M}\}$, where $w_{i,j}$ denotes the weight of term t_j in document d_i and calculated by considering the upper approximation of document d_i.

3.2.1 Tolerance approximation space

The definitions of tolerance approximation space $\mathcal{R} = (T, I, \nu, P)$ are as follows

Universe: The universe U is the set of index terms T

$$U = \{t_1, t_2, ..., t_M\} = T \tag{3.4}$$

Tolerance class: Skowron and Stepaniuk (33) maintain that an uncertainty function $I : U \to \mathbb{P}(U)$, where $\mathbb{P}(U)$ is a power set of U, is any function from U into $\mathbb{P}(U)$ satisfying the conditions $x \in I(x)$ for $x \in U$ and $y \in I(x) \Leftrightarrow x \in I(y)$ for any $x, y \in U$. This means that we assume the relation $x\mathrm{I}y \Leftrightarrow y \in I(x)$ is a tolerance relation and $I(x)$ is a tolerance class of x.

The parameterized tolerance class I_θ is then defined as

$$I_\theta(t_i) = \{t_j \mid f_D(t_i, t_j) \geq \theta\} \cup \{t_i\} \tag{3.5}$$

where θ is a positive parameter and $f_D(t_i, t_j)$ denotes the number of documents in D that contain both terms t_i and t_j. From Equation (3.5), it is clear that it satisfies the condition of being reflexive ($t_i \in I_\theta(t_i)$) and symmetric ($t_j \in I_\theta(t_i)$) required by a tolerance relation; the tolerance relation $R \subseteq T \times T$ can be defined by means of function I_θ as $t_i R t_j \Leftrightarrow t_j \in I_\theta(t_i)$.

Assuming that a term is a concept, then the tolerance class $I_\theta(t_i)$ consists of terms related to a concept t_i and the precision of the concept determined might be tuned by varying the threshold θ.

Vague inclusion function: the vague inclusion function $\nu : \mathbb{P}(U) \times \mathbb{P}(U) \to [0, 1]$ measures the degree of inclusion between two sets and is defined as

$$\nu(X, Y) = \frac{|X \cap Y|}{|X|} \tag{3.6}$$

where the function ν must be *monotone* w.r.t the second argument, i.e. if $Y \subseteq Z$ then $\nu(X, Y) \leq \nu(X, Z)$ for $X, Y, Z \subseteq U$. Hence, the vague inclusion function can determine the matter whether the tolerance class $I(x)$ of an object $x \in U$ is included in a set X.

Together with the uncertainty function I, the vague inclusion function ν defines the *rough membership function* for $x \in U, X \subseteq U$ as $\mu_{I,\nu}(x, X) = \nu(I(x), X)$. Therefore, the membership function μ for $t_i \in T, X \subseteq T$ is defined as

$$\mu(t_i, X) = \nu(I_\theta(t_i), X) = \frac{|I_\theta(t_i) \cap X|}{|I_\theta(t_i)|} \tag{3.7}$$

Structurality function: with structurality function $P : I(U) \to \{0, 1\}$, where $I(U) = \{I(x) : x \in U\}$, one can construct two subsets based on value of $P(I(x))$, named *structural subset* and *nonstructural subset*, when $P(I(x)) = 1$ and $P(I(x)) = 0$ respectively. In TRSM, all tolerance classes of index terms are considered as structural subsets, hence for all $t_i \in T$

$$P(I_\theta(t_i)) = 1 \tag{3.8}$$

3.2.2 Approximations

With the foregoing definitions, we can define the lower approximation $\mathbf{L}_{\mathcal{R}}(X)$, upper approximation $\mathbf{U}_{\mathcal{R}}(X)$, and boundary region $\mathbf{BN}_{\mathcal{R}}(X)$ of any subset $X \subseteq T$ in a tolerance space $\mathcal{R} = (T, I_\theta, \nu, P)$ as follows

$$L_{\mathcal{R}}(X) = \{t_i \in T \mid \nu(I_\theta(t_i), X) = 1\} \tag{3.9}$$

$$U_{\mathcal{R}}(X) = \{t_i \in T \mid \nu(I_\theta(t_i), X) > 0\} \tag{3.10}$$

$$BN_{\mathcal{R}}(X) = U_{\mathcal{R}}(X) - L_{\mathcal{R}}(X) \tag{3.11}$$

Refers to the basic idea of rough sets theory (32), for any set of X, intuitively we may assume the upper approximation as the set of concepts that share some semantic meanings with X, the lower approximation as the *core* concepts of X, while the boundary region consists of concepts that *cannot be classified uniquely* to the set or its complement, by employing available knowledge.

3.2.3 TRSM document representation

After all, the richer representation of document $d_i \in D$ is achieved by simply representing the document with its upper approximation, i.e.

$$\mathbf{U}_{\mathcal{R}}(d_i) = \{t_i \in T \mid \nu(I_\theta(t_i), d_i) > 0\} \tag{3.12}$$

followed by calculating the weight vector using the extended weighting scheme, i.e.

$$w^*_{ij} = \frac{1}{S} \begin{cases} (1 + \log f_{d_i}(t_j)) \log \frac{N}{f_D(t_j)} & \text{if } t_j \in d_i \\ 0 & \text{if } t_j \notin \mathbf{U}_{\mathcal{R}}(d_i) \\ \min_{t_k \in d_i} w_{ik} \frac{\log \frac{N}{f_D(t_j)}}{1 + \log \frac{N}{f_D(t_j)}} & \text{otherwise} \end{cases} \tag{3.13}$$

where S is a normalisation factor. The extended weighting scheme is defined from the standard TF*IDF weighting scheme and is necessary in order to handle terms that occur in a document's upper approximation but not in the document itself.

By employing TRSM, the final document representation has less zero-valued similarities. This leads to a higher possibility of two documents having non-zero similarities although they do not share any terms. This is the main advantage the TRSM-based algorithm claims to have over traditional approaches.

3.3 The Challenges of TRSM

We identified that there are three fundamental components of TRSM to work which are dependent in sequence: *(1)* the tolerance classes of all index terms; *(2)* the upper document representation; and *(3)* the TRSM weighting scheme. Figure 3.3 displays the basic process of tolerance rough sets model which contains those three components.

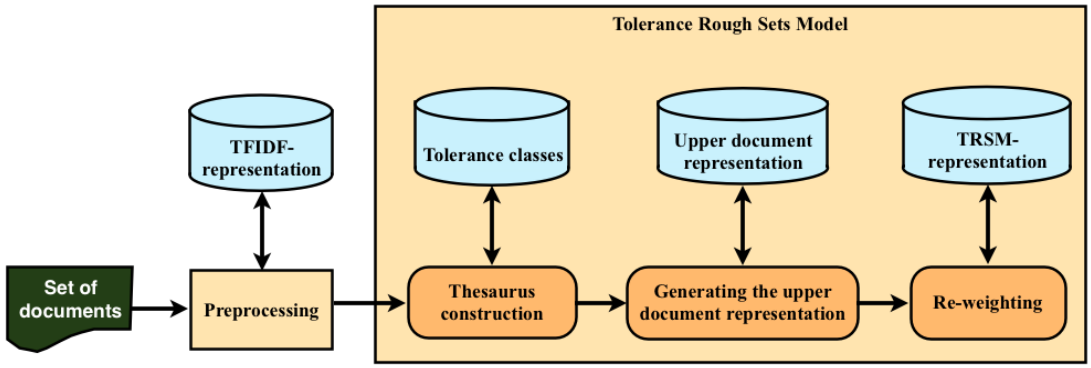

Figure 3.3: Tolerance rough sets model - The process of tolerance rough sets model.

A document representation based on TRSM (TRSM-representation) can be seen as the revised version of a base representation which is recalculated using the TRSM weighting scheme. The base representation is modeled by calculating the term frequency (TF) and the inverse document frequency (IDF) of a term, i.e. commonly called TF*IDF weighting scheme; hence we dub this representation the TFIDF-representation. Suppose the representation of document produced by TRSM and TF*IDF are structured as matrices, thus Figure 3.4 shows the relationship between them, where $tfidf$ and $trsm$ denote the weight of term computed by TF*IDF and TRSM weighting scheme respectively.

During term weight computation, TRSM consults the upper document representation, whereas the upper representation of a document is only possible to be generated when the tolerance classes of all index terms are available. Refer to Equation (3.5), a tolerance class of a term t_i consists of all index terms consider as semantically related with the term t_i and the precision of relatedness between a pair of terms is defined by the tolerance value θ. In other words, the importance of relationship between terms is determined by θ value.

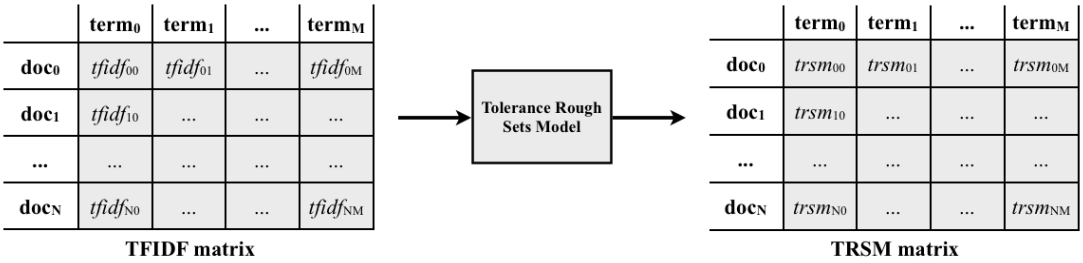

Figure 3.4: Relationship between TFIDF-representation and TRSM-representation - The TRSM-representation is possible to be constructed by taking TFIDF-representation as the input of TRSM.

Based on the nature of TRSM, tolerance classes can be categorized as a thesaurus; a lightweight ontology who reflects the relationship between terms (34). As the heart of TRSM, thesaurus becomes the knowledge of the system who implements it.

It should be clear that in TRSM the quality of document modeling would rely on the thesaurus, and the quality of the thesaurus might depend on the tolerance value θ. Despite the fact that tolerance value is a critical element in TRSM, there is no formal mechanism available for its determination and it is a common practice that the selection is performed manually by the practitioners with regard to their data. This particular issue will be further discussed in Chapter 5.

The thesaurus might be constructed based on an algorithm explained by Nguyen and Ho (15). The algorithm takes a document-by-term matrix (i.e. the TFIDF matrix) as the input and yields the tolerance matrix, which is structured as a binary term-by-term matrix. Figure 3.5 shows the steps of the algorithm. Subsequently, the occurrence binary matrix *OC matrix*, the co-occurrence matrix *COC matrix*, and the tolerance matrix *TOL matrix* were generated in sequence manner by employing Equation (3.14), Equation (3.15), and Equation (3.16). Note that $tfidf_{i,j}$ denotes the weight of term j computed by TF*IDF scheme in document i, card(OC^x AND OC^y) denotes the cardinality of two terms, t_x and t_y, being occurred together in a collection, and θ is the co-occurrence threshold of terms.

$$oc_{i,j} = 1 \quad \Leftrightarrow \quad tfidf_{i,j} > 0 \ . \tag{3.14}$$

$$coc_{x,y} = \text{card}(OC^x \text{ AND } OC^y) \ . \tag{3.15}$$

$$tol_{x,y} = 1 \quad \Leftrightarrow \quad coc_{x,y} \geq \theta \ . \tag{3.16}$$

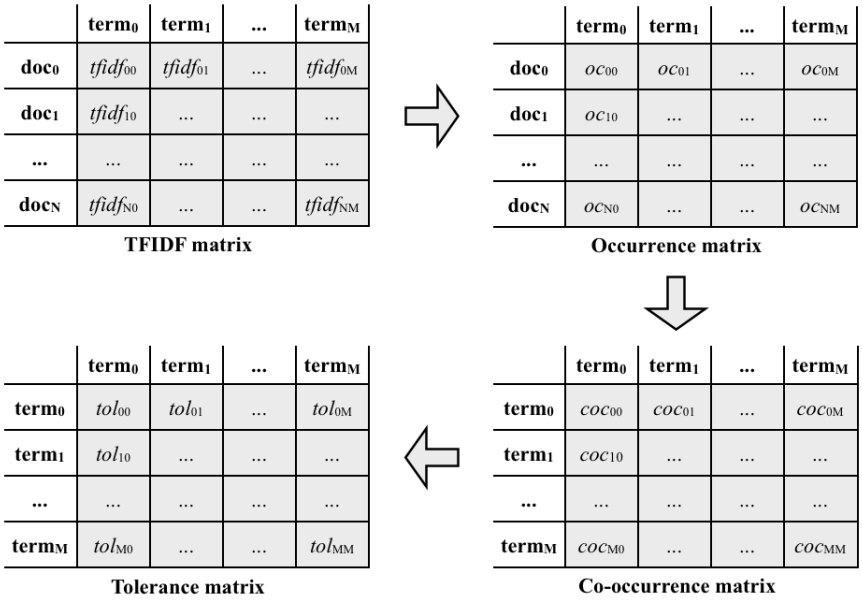

Figure 3.5: Thesaurus construction - The thesaurus construction takes a document-by-term matrix (TFIDF matrix) as the input and yields a term-by-term matrix (Tolerance matrix).

In addition to tolerance value, the algorithm demonstrated that a data source might have an impact on the thesaurus quality since it manipulates a set of documents functions as the only input. The effects might be as a consequence of the type of data source or the size of the collection.

Another important subjects relevant to the thesaurus of TRSM is pertaining to the fact that the thesaurus is created based on the quantity of two terms occur together, thereby employs tolerance value θ as the threshold of semantic relatedness. In fact, refer to the term weighting scheme, there are other alternatives of co-occurrence data who takes more factors into consideration, e.g. the TF*IDF weighting scheme. By that means, other similarity measures, i.e. cosine, might be applied. The presumptions pertaining to thesaurus optimization will be examined in Chapter 6.

Refer to the path of TRSM, its computation complexity is the aggregation of each task, i.e. thesaurus construction, upper representation generation, and re-weighting task. The first task requires $O(NM^2)$ (15), while the second and third tasks both requires $O(NM)$, where N defines the number of documents and M defines the number of index terms. Thus totally, the upper bound of TRSM implementation would

be $O(NM^2)$. With regard to the system efficiency, minimizing the dimensionality of document vectors would be a practical alternative for the complexity. Using this as the starting point, in Chapter 7 we are going to introduce a novel model of document namely the lexicon-based representation.

Chapter 4

The Potential of TRSM

4.1 Introduction

We may find studies showing the positive results of TRSM implementation for document clustering task (13, 14, 15, 16), query expansion (17), and document retrieval task (35). Those studies claimed that TRSM-representation was richer than the baseline representation (TFIDF-representation), however none has shown and explained empirically the richness.

It has been known that the richness of TRSM-representation is understood as having less zero-value similarities and having higher possibility that two documents holding non-zero similarities although they do not share any terms. The result of our study presented in this chapter confirmed those affirmations and add another fact. We found that the TRSM-representation consists of terms considered as important by human experts. Further, the study revealed that that rough sets theory seems to work in accordance with the natural way of human thinking. Finally, the study showed that TRSM is a viable option for a semantic IRS.

4.2 Experiment Process

We used two corpora, ICL-corpus and WORDS-corpus[1], with 127 topics. We took an assumption that each topic given by human experts in annotation process was a

[1]ICL-corpus consists of 1,000 documents taken from an Indonesian choral mailing list, while WORDS-corpus consists of 1,000 documents created from ICL-corpus in an annotation process conducted by human experts. Further explanation of these corpora is available in Appendix C.1.

concept, therefore we considered the keywords determined by the human experts[2] as the term variants that highly related with particular concept. These keywords are the content of text body in WORDS-corpus, hence each document of WORDS-corpus contains important terms of particular concept(s) selected by human experts. With regard to the automatic process of the system, we considered these keywords as the relevant terms for each document (which bear one or more topics) that should be selected by the system. Therefore, WORDS-corpus was treated as the ground truth of this study.

Figure 4.1 shows the general process of the study and the dashed rectangle identifies the focus of the experiment, which were performed twice, i.e. with stemming task and without stemming task.

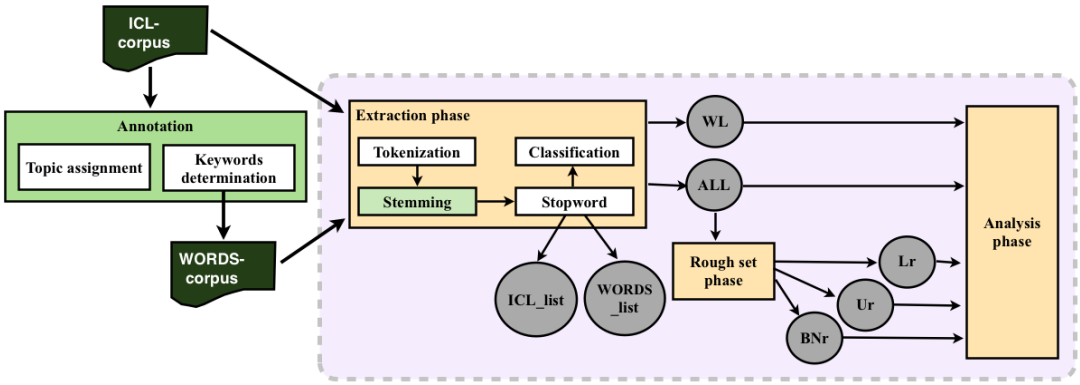

Figure 4.1: Main phases of the study - There were 3 main phases: extraction, rough sets, and analysis. A rectangle represents a process while a circle represent a result.

4.2.1 Extraction Phase

The main objective of extraction phase was preprocessing both corpora. Confix-Stripping stemmer (CS stemmer), a version of Indonesian stemmer, was employed in the stemming task while Vega's stopword (36) was applied in the stopword task[3]. CS stemmer was introduced as a new confix-stripping approach for automatic Indonesian stemming and was showed as the most accurate stemmer among other automated

[2]We collaborated with 3 choral experts during annotation process. Their backgrounds could be reviewed in Appendix C.3.

[3]We used CS stemmer and Vega's stopword in all of our studies presented in this thesis.

Indonesian stemmers (37). Vega's stopword has shown to produce the highest precision@10, R-precision, and recall values (although the differences without stopping words are not significant ($p < 0.05$), except for the recall value (0.038)), among other available Indonesian stopword lists (10).

Documents were tokenized based on character other than alphabetic. The resulted tokens were stemmed by the CS stemmer and then compared to the Vega's stopword. It yielded lists of unique terms and its frequency. There were 9,458 unique terms extracted from ICL-corpus and 3,390 unique terms extracted from WORDS-corpus; called *ICL_list* and *WORDS_list* respectively. When it was run without stemming process, we identified 12,363 unique terms in ICL_list and 4,281 unique terms in WORDS_list.

Both corpora were classified based on 127 topics yielded in preliminary process, i.e. annotation process[4]. Recall that we took an assumption that each topic was a concept and keywords determined by human experts were important variants of a concept hence aggregation of terms appeared in each class were taken as the terms for representative vector of each class. The set of classes resulted from ICL-corpus was called *IL* while the set of classes resulted from WORDS-corpus was called *WL*; each set of class, IL and WL, consists of 127 classes. So, technically speaking, instead of document-term matrix, we worked with topic-by-term matrix.

4.2.2 Rough Set Phase

This phase was conducted in order to generate the lower set, upper set, and boundary set of each class in IL. These sets were possible to be created using Equation (3.9), Equation (3.10), and Equation (3.11) when tolerance classes of all index terms were ready.

The tolerance classes was constructed by following the steps described in Fig. 3.5 of previous chapter, with an exception that in this experiment the algorithm took the topic-by-term frequency matrix as its input. Thereby, Fig. 4.2 displays the steps applied for the thesaurus generation of this particular study.

[4]Please see Appendix C.1 for explanation of annotation process.

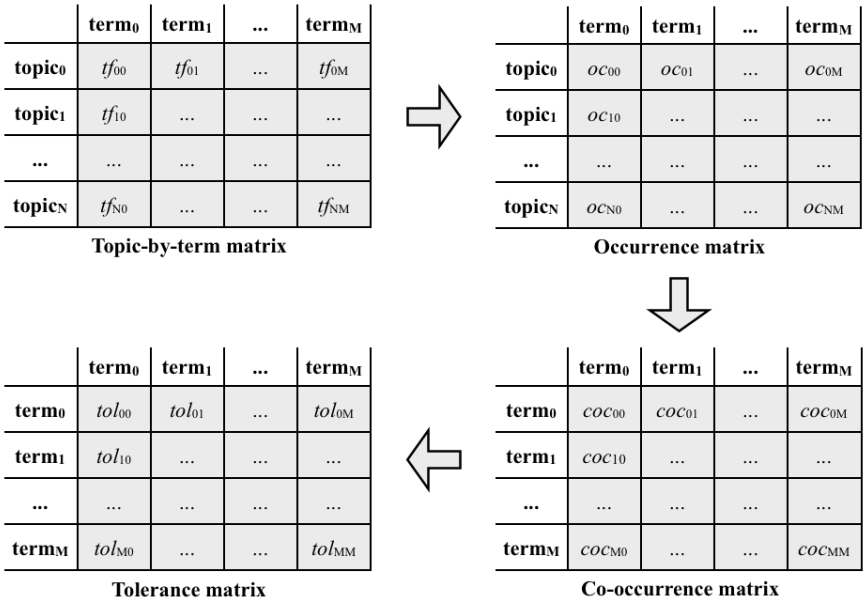

Figure 4.2: Tolerance classes construction - The construction of tolerance classes in this study took topic-term matrix as the input and produced a term-by-term matrix as the output. Here, M denotes the number of index term and N denotes the number of topic.

4.2.3 Analysis Phase

In analysis phase, we examined the mean recall and precision of upper set (US), lower set (LS), and boundary set (BS) of IL by taking the WL as the ground truth. The computations were run for co-occurrence threshold θ between 1 to 75.

Recall and precision are the most frequent and basic measures for information retrieval effectiveness (38). Recall R is the fraction of relevant documents that are retrieved while precision P is the fraction of retrieved documents that are relevant. Suppose *Rel* denotes relevant documents and *Ret* denotes retrieved documents, then recall R and precision P are defined as follows

$$R = \frac{\sharp(relevant\ items\ retrieved)}{\sharp(relevant\ items)} = \frac{|Rel\ \cap\ Ret|}{|Rel|} \tag{4.1}$$

$$P = \frac{\sharp(relevant\ items\ retrieved)}{\sharp(retrieved\ items)} = \frac{|Rel\ \cap\ Ret|}{|Ret|} \tag{4.2}$$

In this study, both measures were used for the *terms* rather than *documents*. That is to say, by considering WORDS_list as the ground truth, then recall R is the fraction

Table 4.1: Formulas for recall and precision calculations.

	US	LS	BS
Recall_{WL}	$\frac{\lvert WL \cap US_{IL}\rvert}{\lvert WL\rvert}$	$\frac{\lvert WL \cap LS_{IL}\rvert}{\lvert WL\rvert}$	$\frac{\lvert WL \cap BS_{IL}\rvert}{\lvert WL\rvert}$
Precision_{WL}	$\frac{\lvert WL \cap US_{IL}\rvert}{\lvert US_{IL}\rvert}$	$\frac{\lvert WL \cap LS_{IL}\rvert}{\lvert LS_{IL}\rvert}$	$\frac{\lvert WL \cap BS_{IL}\rvert}{\lvert BS_{IL}\rvert}$
Recall_{IL}	$\frac{\lvert IL \cap US_{IL}\rvert}{\lvert IL\rvert}$	$\frac{\lvert IL \cap LS_{IL}\rvert}{\lvert IL\rvert}$	$\frac{\lvert IL \cap BS_{IL}\rvert}{\lvert IL\rvert}$

of relevant terms that are retrieved while precision P is the fraction of retrieved terms that are relevant. The formulas applied for recall and precision are displayed in the first and second rows of Table 4.1, where US_{IL}, LS_{IL}, and BS_{IL} respectively denote the upper set, lower set, and boundary region of IL. The Recall_{IL} of the third row is an additional calculation used for evaluating the recall of IL terms in each set. Based on the definition, better recall is preferred than better precision for the reason that better recall would ensure the availability of relevant terms in the set.

4.3 Discussion

With regard to the process of developing WORDS-corpus, the fact that ICL_list could cover almost all WORDS_list terms was not surprising. It was interesting though that there were some terms of WORDS_list did not appear in ICL_list; 17 terms produced by the process without stemming task and 11 terms produced by the process with stemming task. By examining those terms, we found that the CS stemmer could only handle the formal terms (6 terms) and left the informal terms (5 terms) as well as the foreign term (1 term); the other terms caused by typographical error (5 terms) in ICL_corpus.

Despite the fact that CS stemmer succeeded in reducing the size of ICL_list for 23.50% as well as of WORDS_list for 20.81%, it reduced the mean recall of IL about 0.64% for each class from 97.39%. We noticed that the mean precision of IL was increased about 0.25% for each class, however the values themselves were very small (14.56% for process without stemming task and 14.81% for process with stemming task). From these, we could say that the ICL_list was too noisy of containing numerous

Table 4.2: Average Recall and Precision of ICL_list (IL) and WORDS_list (WL).

	With Stemming			Without Stemming		
	US (%)	LS (%)	BS (%)	US (%)	LS (%)	BS (%)
(1)	(2)	(3)	(4)	(5)	(6)	(7)
Recall_{WL}	97.64	5.55	92.08	97.55	4.64	92.91
Precision_{WL}	13.77	27.49	13.50	14.13	26.30	13.75
Recall_{IL}	100.00	5.00	95.00	100.00	4.43	95.57

unimportant terms for particular topic.

Table 4.2 shows the mean values of recall and precision for the sets of IL (i.e. the upper set (US), lower set (LS), and boundary set (BS)) when they were run with and without stemming by considering WORDS_list (WL) as the ground truth. Exceptional is for the third row which is the recall of IL sets over the ICL_list (IL). All of these calculations performed by applying the formulas displayed in Table 4.1.

4.3.1 Recall

It was explained in Section 3.2.2, that for any set of X, the upper set might consist of terms that share some semantic meanings with X. Further, notice that in this study we used a specific domain of corpus, which is a choral corpus. Based on these, the values of Recall_{IL}-US in Table 4.2 for process with and without stemming task, which are 100%, made us confident that the TRSM model has been employed correctly. The upper sets consist of all ICL_list terms due to the fact that generally all index terms are semantically related with choral domain.

One task of annotation process conducted by human experts was keyword determination[5]. It was a fact that during that task our human experts seemed to encounter difficulty in defining keywords for a topic many times. When they were in this position, they preferred to choose sentences on the text or even make their own sentences to describe the topic rather than listing the highly related keywords specifically. The consequence was they introduced numerous number of terms for particular topic. It explains why the value of Recall_{WL}-US for both with and without stemming in Table 4.2 are very high. It is important to be noted that we reckoned all the terms used by

[5]Please see Appendix C.1.

the human experts as relevant terms for the reason that those terms however selected to be used to describe a topic.

This human behavior is reflected by the rough sets theory. We may see on Table 4.2, that the mean recalls of WORDS_list (WL) in lower sets (LS) are very low while the mean recalls of WL in boundary regions (BS) are very high. Refer to Section 3.2.2, intuitively the lower set might consist of the *core* terms while the boundary region might consist of the *uncertain* terms. We can see similar results from the table for ICL-corpus, i.e. the mean recalls of ICL_list (IL) in lower sets (LS) are very low while the mean recalls of IL in boundary regions (BS) are very high. We might infer now that the rough sets theory mimics the natural way of human thinking.

With regard to stemming, we can see that all values in column 3 of Table 4.2 are higher than all values in column 6 while all values in column 4 are lower than all values in column 7. It seems that employing stemming task increases system's capability to retrieve the *core* terms of a concept and to avoid the *uncertain* terms at the same time. Further, the table also shows us that Recall$_{WL}$-US value of process with stemming is higher than the one without stemming, which leads us to an assumption that the stemming task is able to retrieve more relevant terms in general. It supports our confidence so far that stemming task with CS stemmer would bring more benefit in this framework of study.

4.3.2 Precision

Despite the fact that better recall is preferred than better precision, as we mentioned in 4.2.3, we notice that the values of Precision$_{WL}$-US are small (13.77% and 14.13%). With regard to Table 4.1, they were calculated using equation $P = \frac{|WL \cap US_{IL}|}{|US_{IL}|}$. Based on the formula, we may expect to improve the precision value by doing one, or both, of these:

1. increase the co-occurrence terms between WL and US_{IL}; or
2. decrease the total number of US_{IL}.

Refer to Equation (3.16), make the θ value higher will reduce the size of upper sets[6], and refer to Equation (4.2) it will increase the mean precision of upper sets in WL_list.

[6]If the size of tolerance classes are smaller then the size of upper sets will be smaller, and vice versa.

So, technically the total number of terms in an upper set is easily modified by altering the tolerance value θ. However, it raises a typical question, i.e. what is the best θ value and how to set it up? As we have briefly explained the importance and the problem pertaining to θ value in Section 3.3 of previous chapter, this issue seems to support our argumentation that an algorithm to set the θ value automatically is significant.

The index term of WORDS-corpus is clearly constant for we took it as the ground truth, hence there is nothing we can do about WL. Suppose we have a constant number of US (after setting up the θ at a certain value), then the possibility to improve the precision lies on the cardinality of terms in $WL \cap US_{IL}$ set, or in other words on maximizing the availability of relevant terms in upper sets. Based on the nature of TRSM method, this could be happened when we have an optimized thesaurus which defines the relationship between terms appropriately. Knowing that a thesaurus is constructed by a set of documents functioned as data source then we might expect better thesaurus if we know the characteristic of data source we should have. Moreover, based on Equation (3.5), another alternative could be related with the semantic relatedness measure applied in thesaurus construction process.

4.3.3 Tolerance Value

Figure 4.3 shows the mean recall of WORDS_list in upper sets of ICL_list for a process with stemming task when θ value is altered from 1 to 75. It is clear from the figure that the number of relevant terms of WORDS_list drastically filtered out from the upper set of ICL_list at low θ values. However, at some points the changes starts to be stable; Taking one value, e.g. $\theta = 21$. The average number of terms in upper sets when $\theta = 21$ (733.79 terms) is interesting for it was reduced up to 92.24% of the average number of terms in upper sets when $\theta = 0$ (9,458 terms). Whereas from Fig. 4.3, we can see that the mean recall at $\theta = 21$ is maintained to be high (97.58%). By this manual inspection, we are confident to propose $\theta \geq 21$ to be used in similar framework of study.

We urge that the upper sets of ICL_list (US_{IL}) enrich the sets of ICL_list (IL). This assertion is based on two empirical data[7]:

1. the mean recall of WL in IL over 127 topics is 96.75%; while

[7]These values are for the process with stemming task.

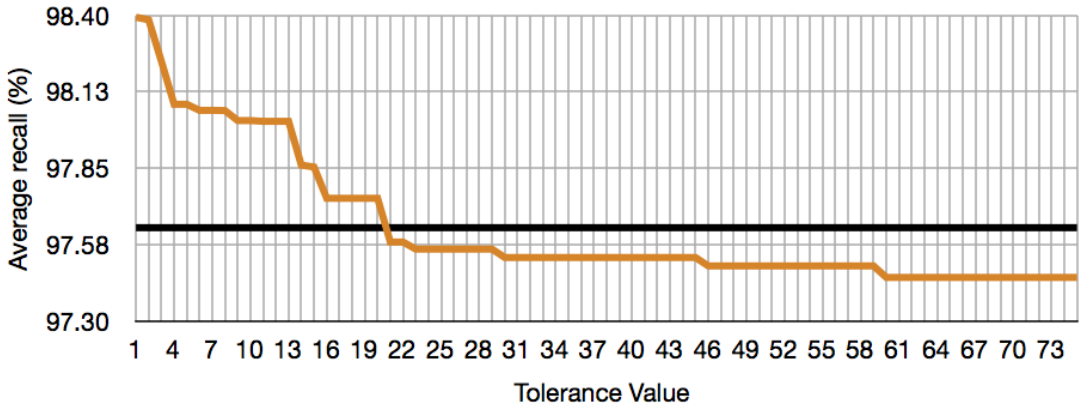

Figure 4.3: The Recall$_{WL}$-US graph - This graph shows the average recall of the sets of WORDS_list in upper sets of ICL_list) for θ value 1 to 75.

2. the mean recall of WL in US_{IL} over 127 topics when θ is altered between 1 to 75 is 97.64%

Thus, we might infer now that the upper sets of ICL_list contain more relevant terms then the sets of ICL_list. In order to construct a document representation, TRSM considers the upper set of a document, hence we might expect that the resulted TRSM-representation consists of more terms and those terms are semantically related. This is a stronger assertion for the claim that tolerance rough sets model enriches the traditional representation of a document and this is a good indicator of TRSM as a feasible method for a semantic IRS.

4.3.4 ICL_list vs. Lexicon

Lexicon is a vocabulary of terms (38). The lexicon utilized by CS stemmer consists of 29,337 Indonesian base words. Comparison between ICL_list and Lexicon showed that there were 3,321 co-occurrence terms. In other words, 64.89% of ICL_list (which is 6,137 terms in total) was different from Lexicon.

We analyzed all of the 6,137 terms with respect to the document frequency and identified that the biggest problem (36.47%) was caused by foreign language[8]. Next two problems were the colloquial terms (27.03%) and proper nouns (21.74%). Combination of foreign and Indonesian terms, e.g. *workshopnya*[9], was considered as colloquial terms.

[8]Most of the foreign terms was English.

[9]It comes from an English term *workshop* and an Indonesian suffix *-nya*.

We also found that the CS stemmer should be improved as there were 48 formal terms left unstemmed in ICL_list.

4.4 Summary

We did a study in order to understand the meaning of *richness* claimed for the representation of document produced by TRSM. The WORDS-corpus who was created by human experts, and contains keywords of each ICL-corpus document, played significant role in the study, for it became the ground truth of the analysis. First of all, the result of the study confirmed that rough sets theory intuitively works as the natural way of human thinking. Being concerned with the meaning of *richness*, we came into conclusion that the TRSM-representation contains more terms than its base representation and those additional terms are semantically related with the topic of the document. After all, with regard to the IRS framework, we infer that TRSM is reasonable for a semantic IRS.

Chapter 5

An Automatic Tolerance Value Generator

5.1 Introduction

Despite the fact, that the value of tolerance value θ is crucial for TRSM implementation, there is no consensus about how we can set a certain number as a θ value. It is usually chosen by the researcher or human expert based on manual inspection through the training data or his/her consideration about the data. It is not deniable that each datum is distinctive hence requires different treatment, however determining the θ value by hand is an exhaustive task before even starting the TRSM paths.

We did a study for an algorithm to generate a tolerance value θ automatically from a set of documents. The idea was based on the fundamental objective of tolerance rough sets model for having a richer representation than the base representation. We took an advantage from the singular value decomposition (SVD) method in order to project all document representations (i.e. TFIDF-representation and TRSM-representation) on a lower dimensional space and then computed the distance between them. The result, together with the analysis of system performance, helped us to understand the pattern of our data and to learn about the principle for a tolerance value determination. In the end, we came up with an intuitive algorithm.

5.2 Experiment Process

The experiment was conducted by following the four phases depicted in Fig. 5.1. Thus, basically we preprocessed the data, constructed the document representation based on TRSM, computed the SVD of TFIDF-representation and TRSM-representation, and finally analyzed them. In the figure, the dashed rectangle identifies the main parts of the experiment that would be run for $\theta = 1$ to 100. In implementation level, we applied the inverted index as the data structure of all document representations[1].

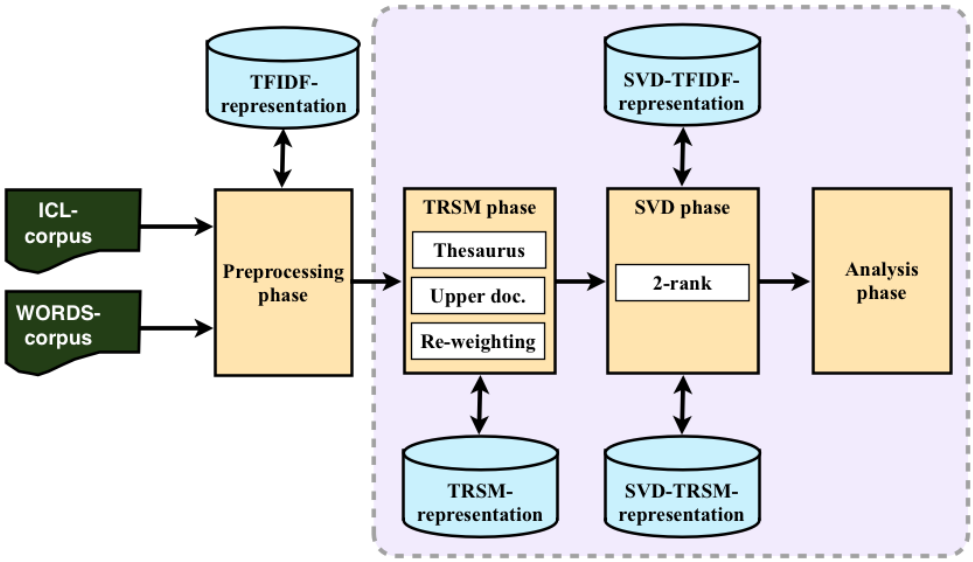

Figure 5.1: Main phases of the study - This study consists of 4 main phases: preprocessing phase, TRSM phase, SVD phase, and analysis phase.

5.2.1 Preprocessing Phase

We used ICL-corpus and WORDS-corpus as the system data and came up with the TFIDF-representations for each corpus. We applied an information retrieval library freely available called Lucene[2] with some modifications in order to embed the Vega's stopword and the CS stemmer.

[1] Inverted index was applied for document representations in all experiments in this thesis.

[2] It is an open source project implemented in Java licensed under the liberal Apache Software License (39). We used `Lucene 3.1.0` in our study. URL for download: `http://lucene.apache.org/core/downloads.html`.

5.2.2 TRSM Phase

The tolerance rough sets model was implemented in this phase, which means we converted the TFIDF-representation into TRSM-representation by following these steps:

1. Construct the thesaurus based on Equation (3.5).
2. Create the upper approximation of documents using Equation (3.12).
3. Generate the TRSM-representation by recalculating the TFIDF-representation using Equation (3.13) and considering the upper approximation of documents.

5.2.3 SVD Phase

The objective of this phase was to compress the high dimensional vector of document so it could be analyzed and plotted on a 2-dimensional graph. We implemented a Java package called JAMA[3] and calculated the SVD, where rank = 2, each for the base representation (TFIDF-representation) and the TRSM-representation; The resulted representation hence called *SVD-TFIDF-representation* and *SVD-TRSM-representation* respectively.

5.2.4 Analysis Phase

In analysis phase, we did two tasks. First of all, we calculated the mean distance and the largest distance between pairs of SVD-representations (i.e. SVD-TFIDF-representation and SVD-TRSM-representation). In order to calculate the distance, we applied the Euclidean function $d(V,U) = \sqrt{\sum_{i=0}^{M}(v_i - u_i)^2}$, where $[v_i]_{i=0}^{M}$ and $[u_i]_{i=0}^{M}$ denote weight vectors of documents V and U. Those distances then were plotted on graphs. We also drew a scatter graph of mean distance for several tolerance values.

Secondly, we generated the recall and precision of retrieval system by employing the 28 topics listed in Table C.3 as the information needs. The recall was computed based on Equation (4.1), while for the precision we were interested in several measurements. First of all, it was the mean average precision (MAP) which is the arithmetic mean of

[3]JAMA has been developed by the MathWorks and NIST. It provides user-level classes for constructing and manipulating real, dense matrices. We used `JAMA 1.0.2` in this study. URL: `http://math.nist.gov/javanumerics/jama/`.

average precision values for individual information needs, thus provides a single-figure measure of quality across recall levels (38). The MAP is defined as follows

$$MAP(Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{|Rel|_j} \sum_{k=1}^{|Rel|_j} Precision(R_{jk}) \tag{5.1}$$

where $q_j \in Q$ is the j^{th} information need, $|Rel|_j$ is the total number of relevant documents for q_j, and R_{jk} is the set of ranked retrieval results from the top result until document d_k.

The other measurements of precision we used were Precision@10, Precision@20, and Precision@30, in which all of them are the measures at fixed low level of retrieved result and hence are referred as *precision at k* (38), where k defines the amount of top documents would be examined that retrieved by the system. The last precision measure we concerned was R-Precision which is basically similar with the *precision at k* measures, except that the k is the amount of relevant documents for each query.

In order to guarantee consistency with published results, we applied the *trec_eval*[4] program created and maintained by Chris Buckley to compute the recall and MAP of the retrieved documents.

5.3 Discussion

5.3.1 Learning from WORDS-corpus

We kept the assumption that each document of WORDS-corpus consists of essential keywords, which should appear in corresponding document representation of ICL-corpus. The distance between document representations of both corpora measures how far an ICL-corpus document from a WORDS-corpus document is. Thus, the assumption brings us to a preference of smaller value of distance; When we had a smaller value of distance, we might expect more keywords appear in an ICL-corpus document.

Figure 5.2 depicts the distances between TRSM-representations of ICL-corpus and WORDS-corpus after they were reduced into 2 dimensions for tolerance value 0 to 50. Figure 5.2(a) shows the result of mean distance (let us call it `mean_distance`) which was calculated by taking the mean average of the distances of all TRSM documents at certain tolerance value. The largest distance (let us call it `largest_distance`) displayed

[4]We used the `trec_eval.9.0` which is publicly available on `http://trec.nist.gov/trec_eval/`.

in Fig. 5.2(b) reveals the largest value of distance, hence it gives us a clue about the size of document cluster at each particular tolerance value. The `mean_distance` graph simply tells us that the higher tolerance value, the farther the distance, and thus the less relevant terms should appear in TRSM-representation of ICL-corpus. It seems that large `largest_distance` might lead to large `mean_distance`.

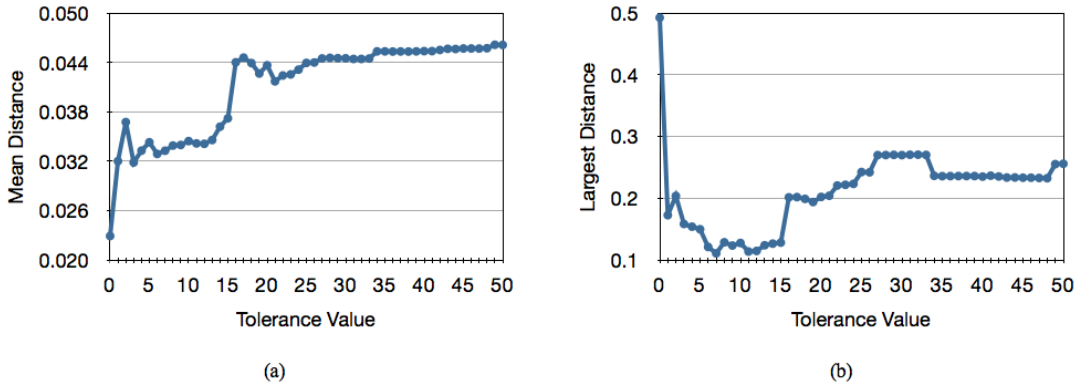

Figure 5.2: Distances between document of ICL-corpus and WORDS-corpus - The distances between TRSM-representations of ICL-corpus and of WORDS-corpus where $1 \leq \theta \leq 50$. Graph (a) is the mean distance, while (b) is the largest distance.

Analyzing scatter graph of distance between each document of ICL-corpus and WORDS-corpus after the TRSM method should give us more understanding about the relationship between those corpora and the alteration of tolerance value. Figure 5.3 depicts the clusters of TRSM-documents of ICL-corpus which at certain distance with TRSM-documents of WORDS-corpus when tolerance values are set to 0, 10, 15, and 41.

Concerning that the graphs reflect the distances between ICL-corpus and WORDS-corpus, the ideal graph in Fig. 5.3 would be a single line on X-axis. In this situation, when the documents of ICL-corpus have zero distance with of WORDS-corpus, we might be certain that terms considered relevant in WORDS-corpus are successfully retrieved by TRSM method and put into the TRSM-representation of ICL documents while the other irrelevant ones are filtered out. Suppose we take the WORDS-corpus as the ground truth, then we might expect high recall in low tolerance value.

We know that the corpora we used in this study lie on a single domain specific[5], i.e.

[5]WORDS-corpus is generated based on ICL-corpus hence they dwell in a single domain.

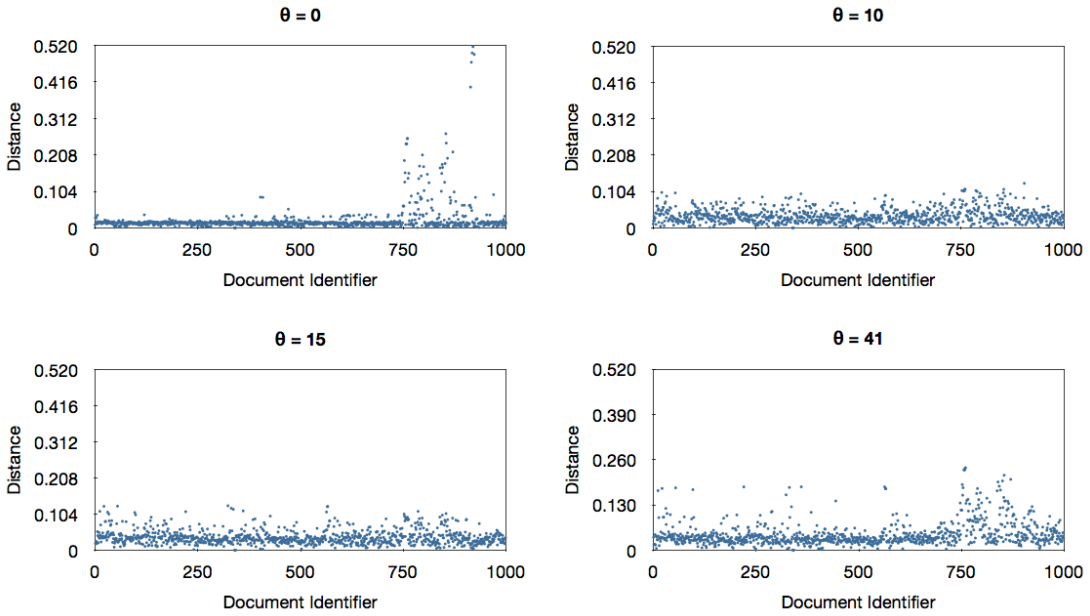

Figure 5.3: Scatter graph of distance - The scatter graph of distance between TRSM documents of ICL-corpus and of WORDS-corpus when $\theta = 0$, $\theta = 10$, $\theta = 15$, and $\theta = 41$.

choral, hence all index terms from both corpora are generally semantically related, even though in a very remote relationship. Therefore, in Fig. 5.3, if the resulted cluster is a line-formed on X-axis, then we would have common documents which contain common terms. Similar circumstances should be happened at any line-formed clusters parallel to X-axis for the reason that similar distance comes from similar document. In other words, we have the least discrimination power of document at this state. Based on these, a cluster with scattered documents inside should be preferred, and in order to have such cluster, its size should big enough.

The largest size of cluster in Fig. 5.3 occurs when $\theta = 0$, in which the documents are much less scattered and even tends to be a line-formed. Consider that θ value is a threshold to filter the index term out from document representation, $\theta = 0$ means that all index terms are determined to be semantically related to each other even though any pair of terms never occurs together. Consequently, the TRSM-representation yielded would have most of index terms within. As the result, we are standing in similar position of foregoing paragraph and it confirms that a parallel line with X-axis signify the commonality of documents in the cluster. Further, comparison between the cluster of $\theta = 0$ and the other clusters in Fig. 5.3 indicates that the tolerance value has a

significant role in removing irrelevant terms as well as relevant terms, for the other clusters are smaller in size and the documents within are more disseminated.

Nevertheless, refer to the context of richness in TRSM method, merely having all index terms in the document representation is out of the intention. Therefore, $\theta = 0$ should be out of our consideration when determining a good tolerance value for any set of documents.

Pertaining to the relationship between `mean_distance` and `largest_distance`, the four tolerance values, i.e. $\theta = 0$, 10, 15, and 41, were assumed to reflect four conditions. Those are the condition when we have, respectively: *a*) small `mean_distance` and large `largest_distance`; *b*) small `mean_distance` and small `largest_distance`; *c*) large `mean_distance` and small `largest_distance`; and *d*) large `mean_distance` and large `largest_distance`. To be more clear, Fig. 5.4 depicts these four conditions in extreme way which will be useful for the discussion in further sections.

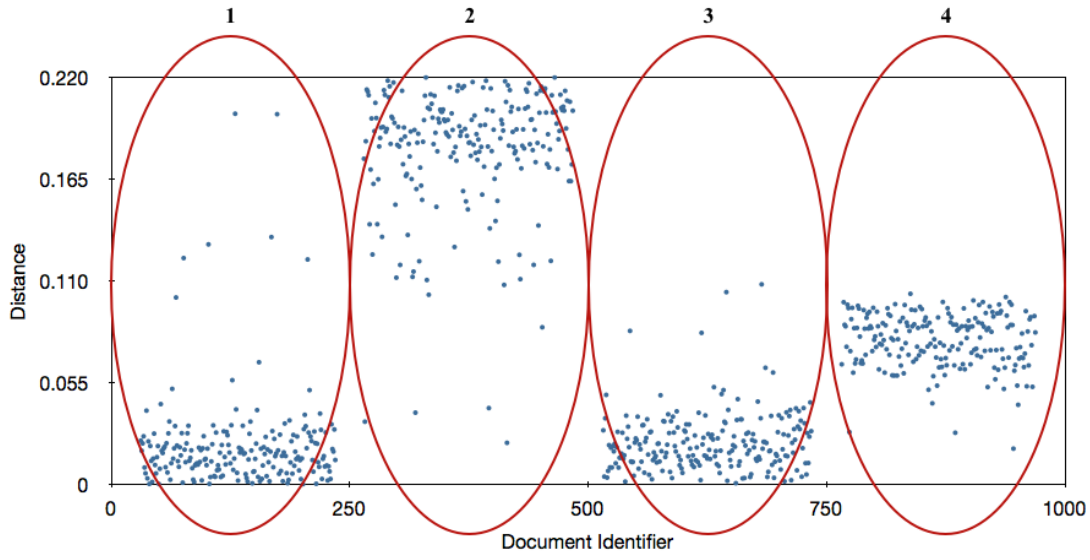

Figure 5.4: Extreme conditions of mean distance and largest distance - The four extreme conditions of the mean distance and largest distance: *(1)* small `mean_distance` and large `largest_distance`; *(2)* large `mean_distance` and large `largest_distance`; *(3)* small `mean_distance` and small `largest_distance`; and *(4)* large `mean_distance` and small `largest_distance`.

5.3.2 Learning from ICL-corpus

In this section, we present and discuss results based on two grounds: *(1)* distance calculation; and *(2)* retrieval system performance

Distance calculation

We computed the distances between TFIDF-representation and TRSM-representation of single corpus, i.e. ICL-corpus, after the dimensionality of those representations were reduced into 2 dimensions using the SVD method. Refer to the capability of TRSM which is to enrich a document representation, larger distance is preferred since it gives an indication that TRSM-representation is richer than the base representation. So, taking the characteristic of document representation into account, we should treat the distance value differently; When we are learning from WORD-corpus (as in previous section), we prefer smaller distance value, whereas when we are analysing ICL-corpus (as in this section) we prefer larger distance value.

In similar fashion with Fig. 5.2, Fig. 5.5 displays the `mean_distance` and Fig. 5.6 shows the largest value of distances between TFIDF-representation and TRSM representation for each tolerance value ranging from 1 to 100. The green horizontal line in each figure reflect the average of `mean_distance` and of `largest_distance`, thus let us call this green lines as `average_distance`.

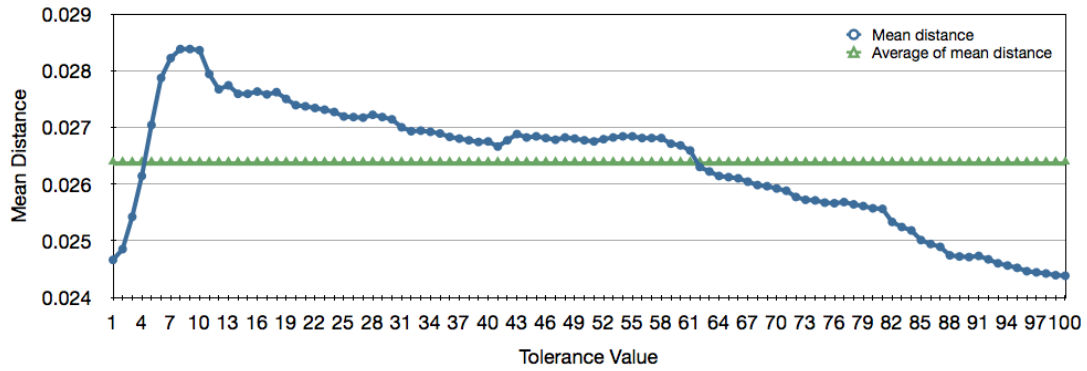

Figure 5.5: Mean Distance - The mean distance between TFIDF-representation and TRSM-representation of ICL-corpus for SVD 2-rank where $1 \leq \theta \leq 100$. The horizontal line is the average of the mean distance values.

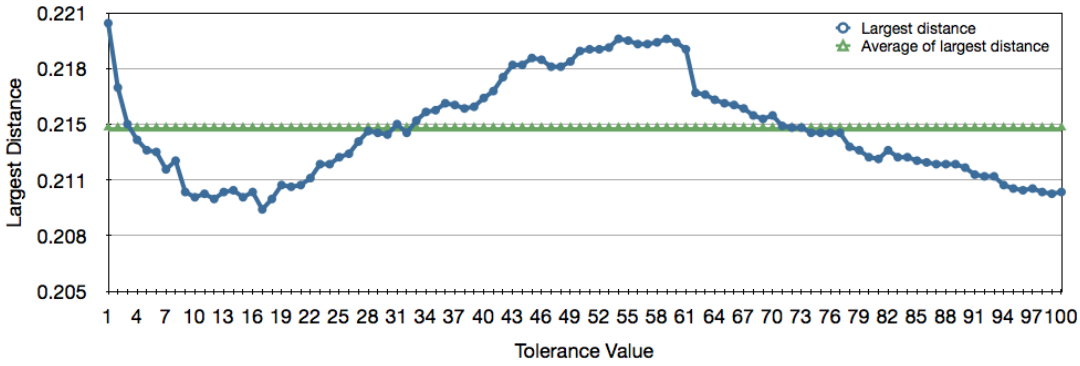

Figure 5.6: Largest Distance - The largest distance between TFIDF-representation and TRSM-representation of ICL-corpus for SVD 2-rank where $1 \leq \theta \leq 100$. The horizontal line is the average of the largest distance values.

Based on the nature of TRSM we prefer large value of `mean_distance`, and as learning from WORDS-corpus in Section 5.3.1, we were suggested to take large value of `largest_distance`. Further, our study in Chapter 4 proposed $\theta \geq 21$, owing to the fact that at $\theta = 21$ the average size of upper documents has been reduced sufficiently up to 92.24% while the average recall (of WORDS-corpus' index terms in ICL-corpus documents) was kept high (97.58%); The average sizes of upper documents were smaller afterward but the changes were observed not significant. Suppose we consider the *large* value for both `mean_distance` and `largest_distance` as having value more than or equal to its `average_distance`, thus Fig. 5.5 and Fig. 5.6 recommend us to focus on $31 \leq \theta \leq 61$.

Retrieval system performance

We put the ICL-corpus into a framework of information retrieval system and generated several results based on the performance measures. Figure 5.7 up to Fig. 5.11 exhibit the results in the form of graphs which goes from the general level to the specific low level, all for tolerance value 1 to 100.

The recall and MAP calculations shown by Fig. 5.7(a) and (b) clearly define that we can rely on TRSM method whose effectiveness is better then the base method[6], nonetheless the performance of `TRSM` has a progressive decline at higher tolerance value.

[6]Base method means that we employed the TF*IDF weighting scheme only without TRSM implementation.

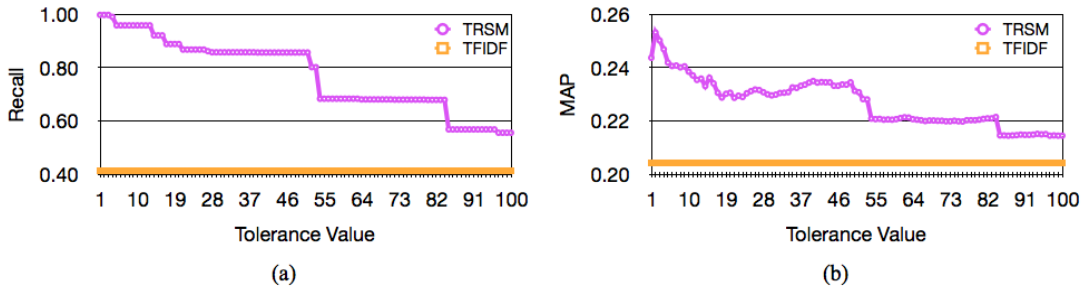

Figure 5.7: Recall and MAP - The system performances while implementing TRSM method and base method in terms of (a) recall and (b) mean average precision (MAP).

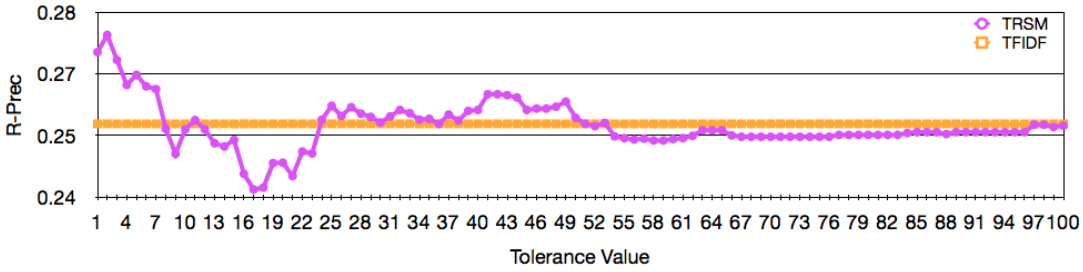

Figure 5.8: R-Precision - The precision of ICL-corpus at top $|R|$ documents, where $|R|$ is the total of relevant documents for each topic.

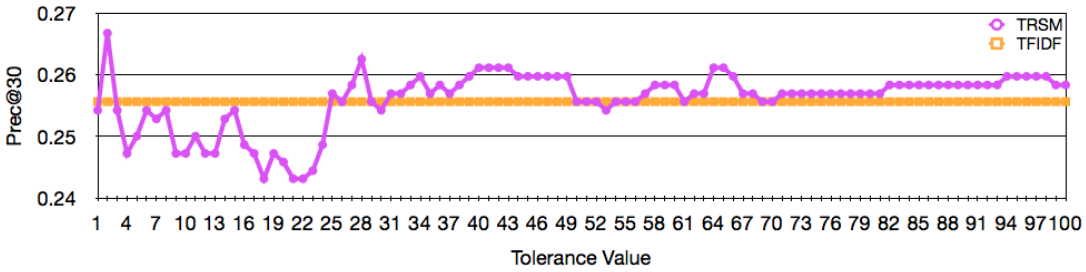

Figure 5.9: Precision@30 - The precision of ICL-corpus at top 30 documents.

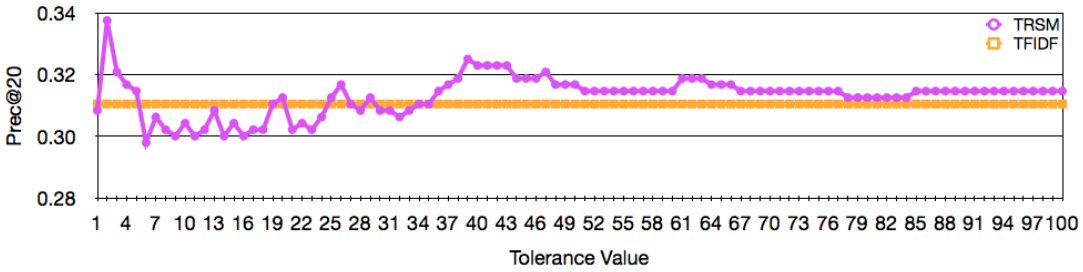

Figure 5.10: Precision@20 - The precision of ICL-corpus at top 20 documents.

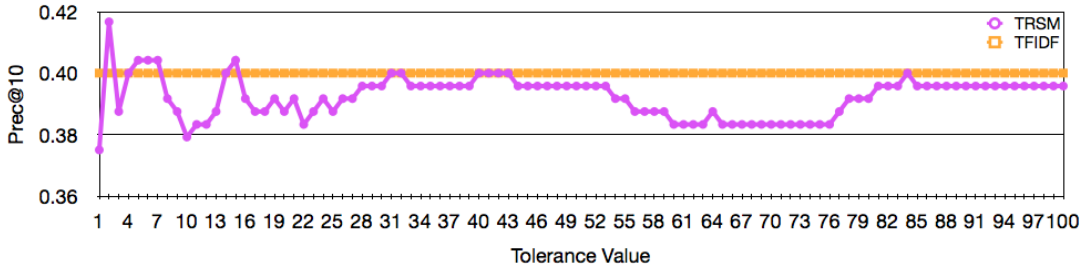

Figure 5.11: Precision@10 - The precision of ICL-corpus at top 10 documents.

Table 5.1: The tolerance values with high precision based on several measurements.

Measures	Tolerance values
R-Precision	31 - 53
Precision@30	31 - 61, but 53
Precision@20	34 - 61
Precision@10	31 - 32, 40 - 43

Correlated with the `mean_distance` and `largest_distance` Fig. 5.7 say no more, despite the recall graph confirms that we might have high value of document recall on lower θ.

For analysis, we went further and came with the results of *precision at k* computations which are displayed in Fig. 5.8, 5.9, 5.10, and 5.11, for R-Precision, Precision@30, Precision@20, and Precision@10 sequentially. We applied our finding of distance calculation (i.e. that we should adjust our attention on tolerance values between 31 to 61) on those graphs by intersecting the tolerance values of each graph whose precision values (of `TRSM` method) are higher or equal to the base method (`TFIDF` method) with the tolerance values between 31 to 61. In conclusion, we have tolerance values between 40 to 43. Table 5.1 lists the tolerance values we manually observed whose values are high for several precision measurements.

With regard to the `mean_distance` and `largest_distance` graphs, at $40 \leq \theta \leq 43$ the distances are adjacent to their `average_distance`. Suppose we apply this into Fig. 5.4, instead of those extreme conditions, we would have considerable large of cluster in which the documents are scattered. In other words, at those tolerance values, the TRSM method might yield fairly richer documents representation and at the same time

sill maintain the distinction between documents. Figure 5.12(a) and (b) are the scatter graph of distance when $\theta = 41$ and, for comparison purpose, $\theta = 0$. Despite the slight difference between the distances, it is still possible to see that the document cluster of $\theta = 0$ is more solid than of $\theta = 41$.

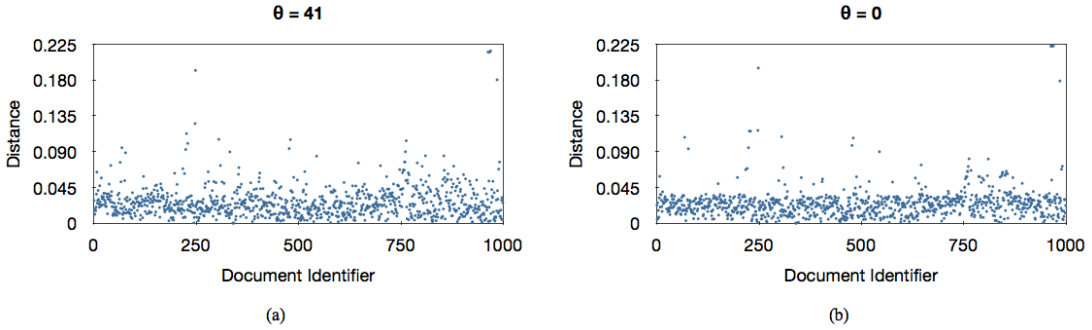

Figure 5.12: Scatter graph of distance - The distance of TRSM-representation from TFIDF-representation when (a) $\theta = 41$ and (b) $\theta = 0$.

Examination on the scatter graph of distance for tolerance values 40 to 43 produced identical results, hence we might infer that those tolerance values would bring us equivalent benefit. However, it is reflected by Fig. 5.5 and Fig. 5.6 that the graphs have tendency to be close to their `average_distance`, thus we prefer θ with the closest `mean_distance` and `largest_distance`, as for this case $\theta = 41$.

5.4 Tolerance Value Generator

We had already introduced the first version of the algorithm to generate a tolerance value θ automatically from a set of documents (40) in which we took the `mean_distance` as a single parameter for consideration. In this chapter, through a more careful analysis, we came into an understanding that both `mean_distance` and `largest_distance` have significant contribution in determining a single tolerance value from a set of documents.

Based on our analysis, a good tolerance value belongs to a fair size of document cluster in which its documents are scattered. Associated with the distances of TRSM-representation from TFIDF-representation, the preferred tolerance value is characterized by the `mean_distance` and the `largest_distance` whose distances are larger or equal to its `average_distance`, in which the closer the distances to its `average_distance`, the better.

Unfortunately, when this rules were applied on Fig. 5.5 and Fig. 5.6, we came with $\theta = 61$, whose `mean_distance` is the closest to its `average_distance` but the `largest_distance` is very large. On that account, the R-Precision and Precision@10 of `TRSM` are beneath the `TFIDF`. So we learned, when the size of document cluster is very large, it is an indication that the TRSM method a little bit out of line in discriminating the document.

For that reason, further restriction needs to be added for the acceptable limit of the `largest_distance` in order to ensure that the `largest_distance` will not have very large value. By observing Fig. 5.6, setting the maximum limit to half of the length between maximum distance and the `average_distance` seems to be appropriate.

Algorithm 1 presents the core idea of our algorithm. Line 1 up to 16 of Algorithm 1 are the initialization and the rest is the main process. The main process says that we choose the tolerance value (namely $finalTheta$) based on values of mean distance ($mean\_dist$) and of largest distance ($largest\_dist$) for certain range of θ whose distances to its average ($md\_toAverage$ for mean distance and $ld\_toAverage$ for largest distance) are the smallest. When searching the tolerance value, we only consider those whose value of $mean\_dist$ is larger than its average (md_avg) and of $largest\_dist$ exists between its average (ld_avg) and its limit (ld_limit). The limit is computed as the average plus half of the length between maximum value of distance and the average ($ld\_avg + (ld\_max - ld\_avg)/2$).

Suppose we apply the Golub-Kahan SVD algorithm (41, p. 455) for dimensionality reduction of TFIDF-representation and TRSM-representation, then in order to compute singular values matrix and V matrix it needs $4MN^2 + 8N^3$ floating-point operations (flops) (41, p. 254), where M is the number of index terms and N is the number of documents. Whereas, for TRSM implementation, the complexity is $O(NM^2)$. Combining these together, the computation of Algorithm 1 does not grow faster than $O(N^3M^2K)$, where K defines the number of tolerance values being examined.

Training Documents

Naturally the number of input data for the algorithm should be all documents in the corpus, so the resulted thesaurus consists of all index terms occurs in the corpus and the chosen tolerance value might suggest the best relationship between those index terms. Our study in Chapter 6 showed that the number of documents used as the data

Algorithm 1 Main Idea of Tolerance Value Generator

Require: A set of documents as data source

Ensure: A tolerance value

```
1: tfidf ← construct TFIDF-representation
2: svd_tfidf ← construct SVD 2-rank of tfidf
3: theta ← lowerBound
4: while theta ≤ upperBound do
5:     trsm ← construct TRSM-representation
6:     svd_trsm ← construct SVD 2-rank of trsm
7:     mean_dist ← mean distance between svd_tfidf and svd_trsm
8:     largest_dist ← the largest distance between svd_tfidf and svd_trsm
9:     theta++
10: end while
11: md_avg ← average of mean_dist
12: md_toAverage_min ← Integer.MAX_VALUE
13: ld_max ← maximum of largest_dist
14: ld_avg ← average of largest_dist
15: ld_limit ← ld_avg + (ld_max − ld_avg)/2
16: ld_toAverage_min ← Double.MAX_VALUE
17: for i ← 0, (|mean_dist| − 1) do
18:     md_toAverage ← mean_dist[i] − md_avg
19:     ld_toAverage ← largest_dist[i] − ld_avg
20:     if md_avg ≤ mean_dist[i] and ld_avg ≤ largest_dist[i] ≤ ld_limit then
21:         if md_toAverage == md_toAverage_min and ld_toAverage ≤
    ld_toAverage_min then
22:             finalTheta ← theta of mean_dist[i]
23:             ld_toAverage_min ← ld_toAverage
24:         else if md_toAverage < md_toAverage_min then
25:             finalTheta ← theta of mean_dist[i]
26:             md_toAverage_min ← md_toAverage
27:             ld_toAverage_min ← ld_toAverage
28:         end if
29:     end if
30: end for
31: return finalTheta
```

source for thesaurus does not guarantee the thesaurus to be more qualified, but the total number of unique terms and the type of input documents should. Further study related to this issue is necessary in the future, particularly concerning the efficiency issue.

Upper and Lower Bound

Recall that tolerance value is required in thesaurus construction in order to filter the index terms out based on the co-occurrence frequency between terms in the corpus. Consequently, the fair scenario is to evaluate all possibilities of tolerance value, i.e. by setting the `lowerBound` to 1 and the `upperBound` to the maximum number of co-occurrence between terms (namely `maxCOC`). The `upperBound` thus is subject to change with regard to the size of data source used.

We have no objection about setting the `lowerBound` to 1. Nevertheless, the `upperBound` needs to be determined prudently. Here are the details of three idea specifically for the `upperBound`.

We urge not to use the idea above (using the maximum number of co-occurrence frequency) alone for the `upperBound`, because it will give us an extensive search range. Take an example, for the ICL-corpus which consists of 1,000 documents and 9,742 index terms, the maximum number of co-occurrence frequency is 329. Thus, if we applied the idea, the `upperBound` is set to 329. For another reason, by manually observing the co-occurrence data, we identified that there were limited number of terms having the co-occurrence frequency bigger than 164 (about half of 329), and much less index terms to be preserved when we increased the tolerance value. This behavior might decrease the ability of TRSM to enrich the base document representation.

We took an advantage of the knowledge, that TRSM is able to enrich the base representation in terms of having more semantic terms, for the second alternative of `upperBound`. Technically speaking, enriching the base representation means that the TRSM-representation contains more terms than the TFIDF-representation. So, the comparison between the average length of TRSM-representation (in the algorithm it is the `avgLengthTRSM`) and TFIDF-representation (namely `avgLengthTFIDF`) would be good for the search termination process. This idea seems to be more affirmative than the use of co-occurrence frequency. However, it gives us an uncertainty state of the real search scope at some extent.

We came into the third idea for the `upperBound` based on our real experience when conducting the experiment. Initially, we set the `upperBound` in a low value and had the first result. Based on the analysis of the results, we decided whether to have another run with higher `upperBound`. This particular process might be happened for several times and we stopped the procedure when we were confident that there would be no other significant changes as if we had another run. In order to be confident, we tried to grasp the pattern of the `mean_distance` manually and decided to stop the procedure if we identified that the best `mean_distance` was located in about 2/3 of the range (namely `certainty_range`), which meant that the resulted tolerance value (namely `finalTheta`) was lower than a certain threshold (namely `threshold_theta`). If the resulted tolerance value was higher than the threshold, we set a new value to the `upperBound` as well as the `threshold_theta` and went for another run. Our experience is implemented mainly as described in Algorithm 2.

Algorithm 2 Set Up the `upperBound`

```
finalTheta ← 0
lowerBound ← 1
range ← r
certainty_range = (2/3) * range
upperBound = lowerBound + range
threshold_theta = lowerBound + certainty_range
theta ← lowerBound
while theta ≤ upperBound do
    finalTheta ← compute the final tolerance value
    if finalTheta > threshold_theta then
        threshold_theta = upperBound + certainty_range
        upperBound+ = range
    end if
    theta + +
end while
```

We put all those three alternatives of `upperBound` into our algorithm as it is shown in Algorithm 3 for reason. The experiment results suggest us to have a high tolerance value, however it is possible to have a low tolerance value in implementation, e.g. when we have a small number of index term in a set of documents for the data source. The

third alternative of `upperBound` should be effective for this particular circumstance since it ensures us to have reasonable search range of tolerance value. The second alternative which make use of the comparison between the average length of TFIDF-representation and of TRSM-representation should guarantee that we would have a tolerance value whose TRSM-representation is richer than the base representation. At last, the maximum co-occurrence frequency might be useful as the final termination process.

Another advantage of knowing the maximum co-occurrence frequency is to set the `range`, for example, by setting it to about 1/3 of the maximum number. Suppose, for the ICL-corpus whose maximum co-occurrence frequency is 329, we set $range = 111$, then we will have maximum 3 times runs.

Eventually, putting all together, we came with Algorithm 3 which is the final version of our algorithm.

5.5 Summary

In this chapter we put forward a revised version of algorithm for defining a tolerance value automatically from a set of documents. The heart of the algorithm is measuring the distances between document representations of data source, i.e. one computed using base method while the other using TRSM method, in their 2-dimensional space which are constructed by utilizing the singular value decomposition (SVD) method over a range of θ values.

We learned from two corpora, ICL-corpus and WORDS-corpus, in order to generate some principles that served as the foundation for the algorithm. We found that we should consider both, the `mean_distance` as well as the `largest_distance`, for realizing a fairly big document cluster in which the documents are adequately scattered. Further, we discussed the parameters used in the algorithm.

Algorithm 3 Tolerance Value Generator

Require: A set of documents as data source

Ensure: A tolerance value

1: $tfidf \leftarrow$ construct TFIDF-representation
2: $svd\_tfidf \leftarrow$ construct SVD 2-rank of $tfidf$
3: $finalTheta \leftarrow 0$
4: $lowerBound \leftarrow 1$
5: $range \leftarrow r$
6: $certainty\_range = (2/3) * range$
7: $upperBound = lowerBound + range$
8: $threshold\_theta = lowerBound + certainty\_range$
9: $avgLengthTRSM \leftarrow$ Integer.MAX_VALUE
10: $avgLengthTFIDF \leftarrow$ the average length of $tfidf$
11: $maxCOC \leftarrow$ the maximum co-occurrence frequency between terms
12: $theta \leftarrow lowerBound$
13: **while** $theta \leq upperBound$ **and** $avgLengthTFIDF < avgLengthTRSM$ **and** $theta \leq maxCOC$ **do**
14: **while** $theta \leq upperBound$ **do**
15: $trsm \leftarrow$ construct TRSM-representation
16: $svd\_trsm \leftarrow$ construct SVD 2-rank of $trsm$
17: $mean\_dist \leftarrow$ the mean distance between svd_tfidf and svd_trsm
18: $largest\_dist \leftarrow$ the largest distance between svd_tfidf and svd_trsm
19: $theta++$
20: **end while**
21: $md\_avg \leftarrow$ average of $mean\_dist$
22: $md\_toAverage\_min \leftarrow$ Integer.MAX_VALUE
23: $ld\_max \leftarrow$ maximum of $largest\_dist$
24: $ld\_avg \leftarrow$ average of $largest\_dist$
25: $ld\_limit \leftarrow ld\_avg + (ld\_max - ld\_avg)/2$
26: $ld\_toAverage\_min \leftarrow$ Double.MAX_VALUE

Algorithm 4 Tolerance Value Generator (continued)

```
    for i ← 0, (|mean_dist| − 1) do
        md_toAverage ← mean_dist[i] − md_avg
        ld_toAverage ← largest_dist[i] − ld_avg
        if md_avg ≤ mean_dist[i] and ld_avg ≤ largest_dist[i] ≤ ld_limit then
            if md_toAverage == md_toAverage_min and ld_toAverage ≤ ld_toAverage_min then
                finalTheta ← theta of mean_dist[i]
                ld_toAverage_min ← ld_toAverage
            else if md_toAverage < md_toAverage_min then
                finalTheta ← theta of mean_dist[i]
                md_toAverage_min ← md_toAverage
                ld_toAverage_min ← ld_toAverage
            end if
        end if
    end for
    avgLengthTRSM ← the average length of trsm at theta
    if finalTheta > threshold_theta then
        threshold_theta = upperBound + certainty_range
        upperBound+ = range
    end if
end while
```

Chapter 6

Optimizing the Thesaurus

6.1 Introduction

Based on the process of modeling a document in TRSM, thesaurus is the heart of TRSM, in which the relationship between terms in the thesaurus is determined by a tolerance value θ. Thus, choosing the right θ value is essential in TRSM implementation. In the previous chapter, we have seen that it is possible to determine a value for θ by considering the mean and the largest distances between TRSM-representation and TFIDF-representation. We also proposed a new version of algorithm to generate the tolerance value automatically. In this chapter, we move the focus on the important issues relevant to the quality of the thesaurus.

We might find most of the graphs presented in the last chapter support the fact that the values of distances between TRSM-representation and TFIDF-representation vary with regard to the tolerance value, and so is the the quality of the TRSM-representation. Therefore, it seems that the thesaurus, which stores information about terms relationship exploited to enrich a document representation, is influenced by the tolerance value. Moreover, the thesaurus of TRSM is created from a collection of text documents as a data source and relied on the co-occurrence of terms as the semantic relatedness measure. These facts imply that the data source and the semantic measure have capacity to produce effect on the quality of the thesaurus.

Tolerance rough sets model uses the frequency of co-occurrence in order to define the semantic relatedness between terms. Despite the raw frequency, there are several ways to calculate the degree of association between pairs of terms from co-occurrence data,

i.e. Cosine, Dice, and Tanimoto measure. Further, with regard to the term-weighting scheme, the term frequency (TF) is the simplest approach to assign the weight for a term but it suffers from a critical problem that it considers all terms as equally important no matter how often it occurs in the set of documents. The inverse document frequency (IDF) is well known to be used in order to enhance the discriminating power of a term in determining relevance by considering the document frequency of the term in the corpus. Combination of term frequency and inverse document frequency produces a composite weight commonly assigned to a term, which is known as TF*IDF weighting scheme. After all, in spite of the raw frequency of co-occurrence, we might get different results from the same co-occurrence data with different formula.

We conducted a study to investigate the quality of the thesaurus of TRSM with regard to these three factors (i.e. tolerance value, data source of thesaurus, and semantic measure) in the framework of an information retrieval system (IRS). We used different corpus as data sources of the thesaurus, implemented different semantic relatedness measure, and altered the tolerance value. In order to analyze the results, we calculated the performance measure of an information retrieval system, i.e. recall and precision, and compared the results with the base representation (TFIDF-representation).

6.2 Experiment Process

We did two experiments. The first experiment focused on the data source of thesaurus, while the second experiment focused on the semantic measure of thesaurus.

For the first experiment, we maintained the frequency of co-occurrence as the measure of semantic relatedness in the thesaurus construction and we used our primary corpus, i.e. ICL-corpus, as the main data which was processed by the IRS. Specifically for the data source of thesaurus, we employed several corpora as listed in Table 6.1; *Total document* column defines the total number of documents in each set of documents which served as the data source, *Total unique term* column defines the total number of index terms, and *Total term* column is the total number of terms appear in the set.

ICL_1000 is actually the ICL-corpus which is a set of the first 1,000 emails of Indonesian Choral Lovers (ICL) Yahoo! Groups, while the ICL_2000 and ICL_3000 are the extension of ICL_1000, which contain the first 2,000 emails and the first 3,000 emails respectively. WORDS_1000 is the WORDS-corpus, hence it is a set of 1,000

Table 6.1: List of data source for thesaurus. This table presents the list of data source used specifically for thesaurus construction.

No.	Data source	Total document	Total unique term	Total term
1.	ICL_1000	1,000	9,742	129,566
2.	ICL_2000	2,000	14,727	245,529
3.	ICL_3000	3,000	21,101	407,805
4.	ICL_1000 + WORDS_1000	2,000	9,754	146,980
5.	ICL_1000 + WIKI_1800	2,800	17,319	191,784

documents which are the keywords defined by our choral experts with regard to each corresponding document in ICL-corpus. Finally, WIKI_1800 is a set of 1,800 short abstracts of Indonesian Wikipedia articles[1]

In the second experiment, we only used single corpus, which was the ICL-corpus that acted as the main data processed by the IRS as well as the data source. For the semantic measure in thesaurus construction, we considered the Cosine measure which is probably the similarity measure that has been most extensively used in information retrieval research. The Cosine was calculated over the TF*IDF weight of term.

For both experiments conducted in this study, we followed three phases displayed in Fig. 6.1 which are preprocessing phase, TRSM phase, and analysis phase; The main differences between experiments were on the TRSM phase. The dashed rectangle shows the central activities of the study, i.e. the TRSM phase and analysis phase, which were iterated for a range of tolerance value (i.e. for $\theta = 1$ to 100). Below are the description of each phase.

6.2.1 Preprocessing Phase

There were no special treatment in this phase. What we did in this phase was similar with the preprocessing phase of the preceding study explained in Chapter 5, in the sense that the Lucene library was implemented and both the Vega's stopword and the CS stemmer were embedded in Lucene. In this study, we separated the data for IRS system

[1]Please see Appendix C.2.

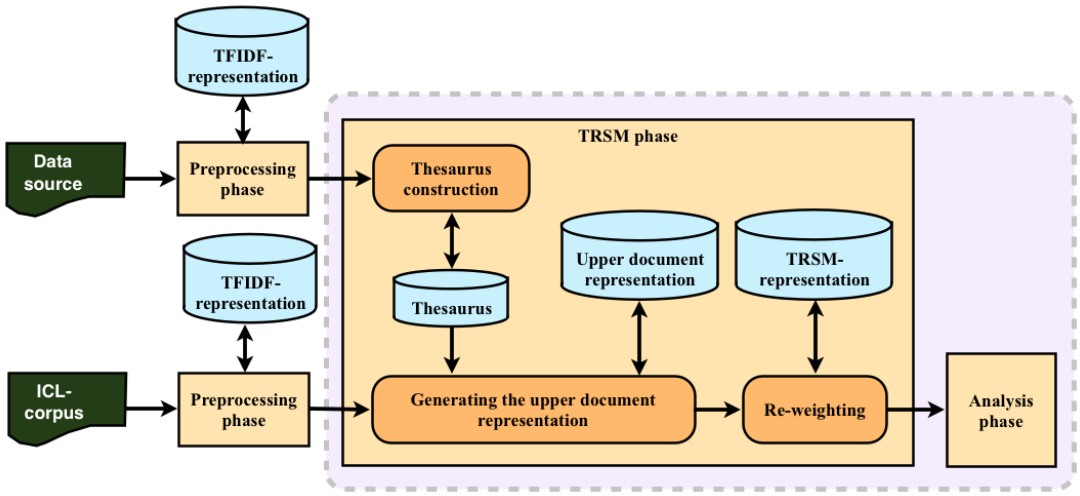

Figure 6.1: Main phases of the study - This figure shows the three main phases of the study in the IRS framework, which are preprocessing phase, TRSM phase, and analysis phase.

from the data source for thesaurus construction, in which both operations yielded the TFIDF-representations.

6.2.2 TRSM Phase

Basically, we followed the three steps of tolerance rough sets model, which were constructing the thesaurus, creating the upper document representation, and re-weighting the TFIDF-representation using the TRSM-weighting scheme. However, we applied the first step only for the data source in order to generate the thesaurus, while the other two steps were applied to the system data, that is the ICL-corpus.

In the first experiment, the thesaurus were constructed from each data source listed in Table 6.1 based on the frequency of co-occurrence terms, while in the second experiment the thesaurus were constructed only using the ICL_corpus and calculated based on Cosine semantic measure. Then, for both experiments, the TRSM-representation was re-weighted by considering the TFIDF-representation and the upper document representation of ICL-corpus, in which the thesaurus became the bottom layer of the upper representation generation.

6.2.3 Analysis Phase

We applied the Cosine similarity[2] in order to retrieve documents from the corpus relevant to the 28 information needs. The queries, which were the 28 topics determined by our choral experts, were modeled into TRSM-query-representations based on the following rule

$$w_j = \begin{cases} (1 + \log f_q(t_j)) \log \frac{N}{f_D(t_j)} & \text{if } t_j \in q \\ 1 & \text{if } t_j \notin \mathbf{L}_{\mathcal{R}}(q) \\ \frac{|I_\theta(t_j) \cap q|}{|I_\theta(t_j)|} & \text{if } t_j \notin \mathbf{U}_{\mathcal{R}}(q) \\ 0 & \text{otherwise} \end{cases} \tag{6.1}$$

where w_j defines the weight of term t_j in a query, $f_q(t_j)$ is the occurrence frequency of term t_j in the query, $f_D(t_j)$ is the document frequency of term t_j in a corpus, N is the total document in the corpus, and $\frac{|I_\theta(t_j) \cap q|}{|I_\theta(t_j)|}$ is the rough membership function between tolerance class of term t_j and the query. We considered a query as a new document in a corpus, thus we add 1 to the total document N and the document frequency $f_D(t_j)$, if term t_j occurs in the query.

Our primary data to analyze the thesaurus were the calculation results of recall and precision of the TRSM-representations created. As the experiment in previous chapter, we calculated the recall and mean average precision (MAP) based on Equation (4.1) and Equation (5.1) sequentially. We compared them to the computation result of the base representation, i.e. the TFIDF-representation.

6.3 Discussion

6.3.1 Result of First Experiment: Data Source of Thesaurus

Figure 6.2 shows the recall values of ICL-corpus by implementing TRSM in which the thesaurus was generated based on the co-occurrence frequency between terms of data sources listed in Table 6.1 and the tolerance value was altered between 0 to 100. The `TFIDF` in the graph is the recall values of TFIDF-representation.

Generally, all data sources perform similar pattern. When $\theta = 1$ they have very high recall values (0.9967 for `ICL_1000` and `ICL_1000 + WORDS_1000` data sources, and 0.9968 for `ICL_2000`, `ICL_3000`, and `ICL_1000 + WIKI_1800` data sources) and the values are gradually decreased when the tolerance value is increased. It is also clear from

[2]Explanation about Cosine as a document ranking is available in Appendix B.

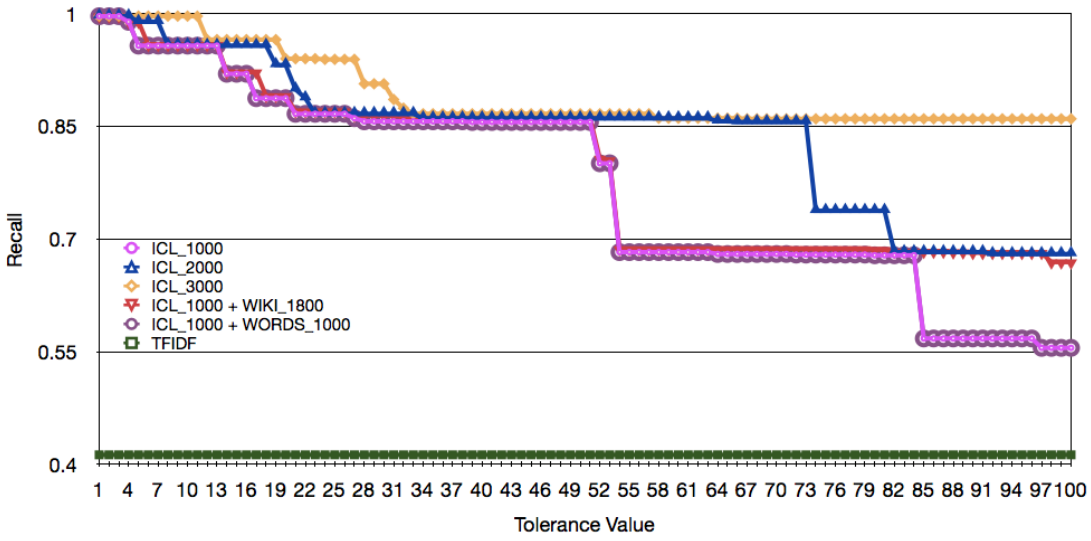

Figure 6.2: Recall - Recall values based on several data sources of thesaurus.

the graph that all the recall values of TRSM-representations outperform the TFIDF-representation's recall value (0.4125).

Refer to the nature of tolerance rough sets model, the general result of recall values shown in Fig. 6.2 is predictable. When we set $\theta = 0$, we put all the index terms into all TRSM-representations that leads to the retrieval of all documents in the corpus, including the relevant ones, for all the queries. When θ is set to 1, a number of index terms which co-occur with no other index terms are removed. It reduces a number of index term appear in document vector at some degree and decreases the retrieval of relevant documents. If we set the θ even higher, more index terms are filtered out and lesser recall values are obtained.

A careful analysis to Fig 6.2 gave us several interesting points. First, we cannot expect anything from adding the `WORDS_1000` as the data source; It has the same result with the `ICL_1000`. The fact that it consists of keywords defined by human experts seems to be not significant for thesaurus optimization. Secondly, it is interesting that adding `WIKI_1800` as data source unpredictably came with similar result to `ICL_1000` up to $\theta = 85$. Thirdly, some improvement were achieved by adding the `ICL_1000` with similar documents as the data source, as it is shown by the `ICL_2000` and `ICL_3000`.

Considering Table 6.1, we learned that the number of unique terms and total terms in the set contribute more to the quality of thesaurus than the number of documents.

Put our focus on adding the ICL_1000 with WORDS_1000 and WIKI_1800, it seems that the kind of unique terms in a set are also count. From Table 6.1, we can see that adding WORDS_1000 (which have 3,477 unique terms) for the data source gives us 12 new unique terms. It means that most of index terms contained in WORDS_1000 are also the index terms of ICL_1000 and we may infer by Fig. 6.2 that the condition brings no improvement to the thesaurus. On the contrary, the index terms of WIKI_1800 are different from the ICL_1000 to a considerable extent; From Table 6.1, we can see that the ICL_1000 has 9,742 unique terms and aggregation of ICL_1000 + WIKI_1800 has 17,319 unique terms, while there are 10,549 unique terms in WIKI_1800 solely. Refer to Fig. 6.2, this fact also gives no significant improvement.

The results are a little bit different by implementing the ICL_2000 and ICL_3000 as the data source for thesaurus. Compared to the ICL_1000, both of them have more unique terms as well as total terms in their sets, and we could be certain that most documents inside them are corresponding in topic, i.e. choral, with ICL_1000. As in Fig. 6.2, these conditions lead to some improvement in recall values. Thus, we may conclude that merely introducing new unique terms does not guarantee any improvement for thesaurus. It should be provided by terms in documents of related domain.

In similar fashion with Fig. 6.2, Fig. 6.3 presents the mean average precision (MAP). One obvious note from Fig. 6.3 is all results of TRSM-representations outperform the result of TFIDF-representation. Specifically, ICL_1000 shows to have the highest MAP value in a very low tolerance value ($\theta = 2$) and its graph tends to decrease as the tolerance value is increased. With regard to the nature of TRSM, this fact is predictable with similar reason we explained in a paragraph above. However, we can see that there are some points where the graph looks to be stable for several tolerance values; After drastic changes in the beginning, the graph tends to be stable at $\theta = 18$, $\theta = 54$, and $\theta = 84$.

As the recall values, the MAP values of combining ICL_1000 with WORDS_1000 are the same with utilizing ICL_1000 separately[3]. It seems to confirm that a set of keywords defined by human experts does not serve as a contributor to the quality improvement of thesaurus.

[3]In fact, we found the same result between ICL_1000 and ICL_1000 + WORDS_1000 in all calculations we made, such as in R-Precision, Precision@10, Precision@20, and Precision@30.

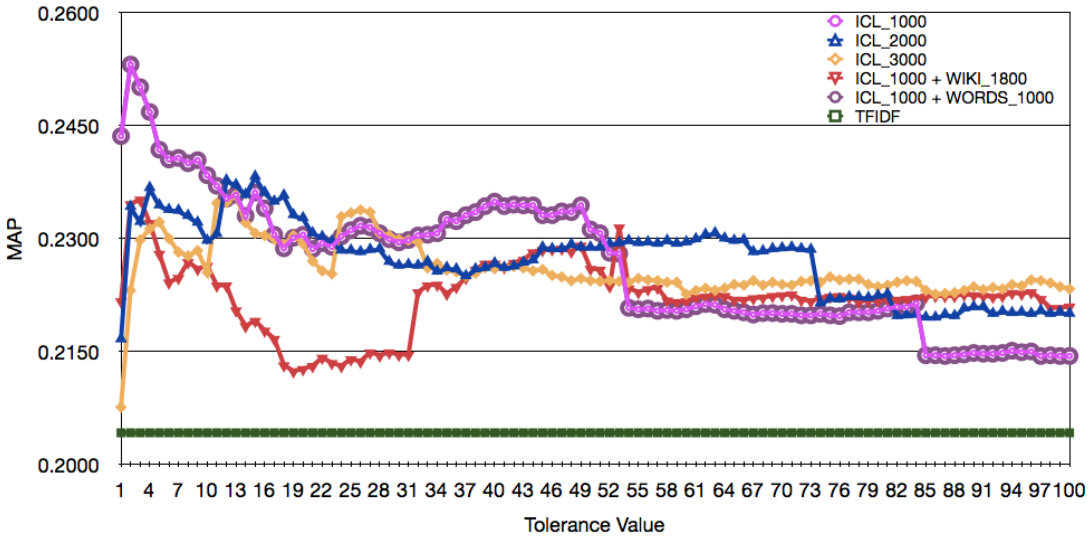

Figure 6.3: Mean Average Precision - Mean Average Precision (MAP) values based on several data sources of thesaurus.

The `ICL_1000 + WIKI_1800` shows to be comparable with the others in high tolerance values ($\theta \geq 32$), even though in low tolerance values, its performance is the worst. The other data sources, `ICL_2000` and `ICL_3000`, perform similar pattern where they both have tendency to decrease as the tolerance value is increased. However, their performances are more stable than the `ICL_1000`. Based on these facts, Fig. 6.3 also indicates that documents in a corresponding domain with the system data (such as `ICL_2000` and `ICL_3000`) may give some contribution to thesaurus improvement.

6.3.2 Result of Second Experiment: Similarity Measure of Thesaurus

Instead of raw frequency of co-occurrence between terms, in the second experiment we considered the Cosine value based on TF*IDF weight of each term in order to define the semantic relatedness between terms of ICL-corpus. With regard to the nature of Cosine measure, the value of relatedness are between 0 to 1, hence in this experiment each θ value was divided by 100. Thus, for θ value 1 to 100, it was read by the thesaurus construction module as 0.01 to 1. Figure 6.4 and Fig. 6.5 display the recall and MAP values of ICL-corpus based on Cosine measure in thesaurus construction.

At first glance, both figures show perfect performances, where most of the results outperform the TFIDF-representation and TRSM-representation based on co-

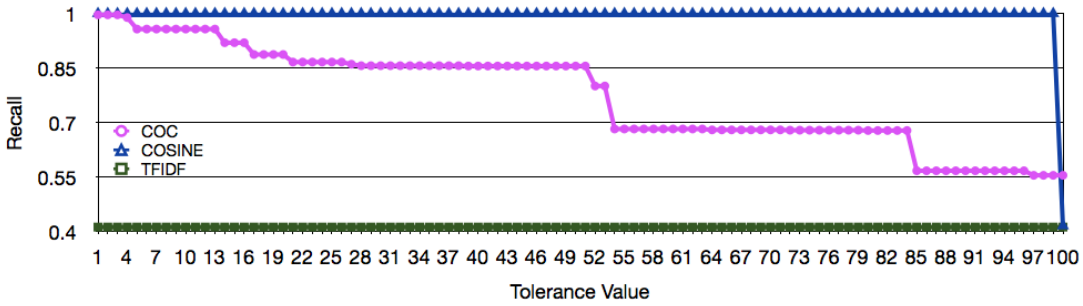

Figure 6.4: Recall - Recall values where the co-occurrence (COC) and Cosine (COSINE) measures were applied to define the semantic relatedness between terms in thesaurus construction.

occurrence measure (COC) at tolerance values 1 to 99. Those performances obtained because most of the index terms occurred in almost all of TRSM-representation. In fact, there were more than 9,000 index terms out of 9,742 occurred in almost all of TRSM-representation, and the changes of amount of index terms occurred in TRSM-representation between tolerance value 1 to 99 were very small; Table 6.2 lists the total number of distinct length of document vector yielded by TRSM when the Cosine measure was implemented for tolerance value 1 to 100. It is not an ideal condition we are looking for. The condition signify that the TRSM has successfully enrich the base representation but it lessen the uniqueness of document at large extent.

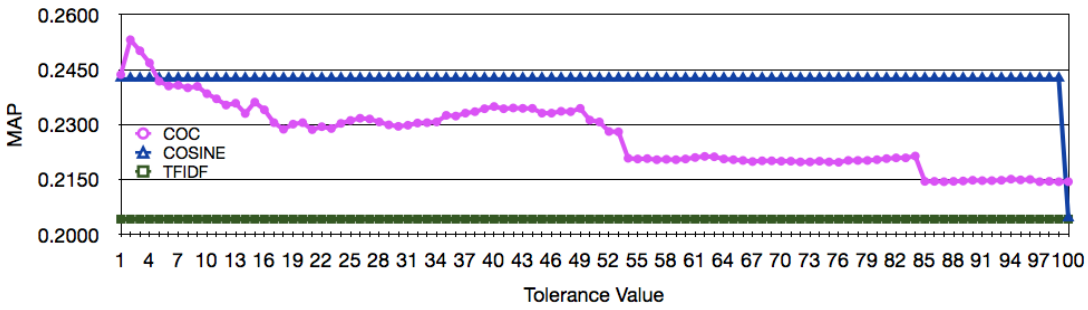

Figure 6.5: Mean Average Precision - Mean Average Precision (MAP) values where the co-occurrence (COC) and Cosine (COSINE) measures were applied to define the semantic relatedness between terms in thesaurus construction.

Both figures also shows that the graphs are suddenly drop at $\theta = 100$ to the TFIDF level. In the thesaurus construction, when $\theta = 100$, the system filtered out index terms whose Cosine values less than 1. It made the tolerance class of each index terms

Table 6.2: Total number of distinct vector length. This table presents the total number of distinct length of TRSM-representation based on Cosine measure for tolerance value 1 to 100.

Tolerance Value	Total Distinction	Tolerance Value	Total Distinction	Tolerance Value	Total Distinction
1 - 64	425	80	426	92	419
65 - 69	423	81 - 82	425	93	425
70 - 72	424	83	423	94	427
73	426	84	427	95	430
74 - 75	427	85	423	96	429
76 - 77	426	86	422	97	425
78	423	87 - 89	421	98	430
79	425	90 - 91	417	99	442
				100	237

consisted of the term itself and thus the final TRSM-representation was exactly the same with the TFIDF-representation. So, it is reasonable that the recall and MAP values of `COSINE` and `TFIDF` displayed in the Fig. 6.4 and Fig. 6.5 are the same.

In this particular experiment, we calculated the Cosine value based on TF*IDF weight of index terms. We applied the TF*IDF weighting scheme in order to refine the discriminating power of each index term. Refer to Equation (B.2), the denominator of Cosine measure functions as the length-normalization of each vector being calculated in order to counterbalance the effect of various document length.

So, philosophically, the Cosine measure seems to be an ideal measure. Further, we found that implementing the Cosine measure in thesaurus construction has lessened the discrepancy of document in the corpus at large extent when Cosine value was less than 1. The fact that ICL-corpus is a set of documents in a specific domain (hence the index terms are generally related) might be the reason why most of the index terms occurred in the TRSM-representation. If this is the reason, it contradicts the result of our first experiment explained in Section 6.3.2 which indicated that we might expect having better contribution in order to improve the quality of thesaurus from a set of documents which was in the same domain with the system data.

Mathematically, the Cosine behavior might be explained by the nature of its equation, in which the association between pairs of terms is basically computed based on

the co-occurrence data (even though in this particular experiment we have refined the raw frequency into the TF*IDF weight). Empirically, there were numerous pairs of terms occurred together in documents which leads to high values of Cosine and little changes in the values. Notice that conventionally a document is written using the common words of a subject. Thus, the fact that ICL-corpus came from a mailing lis of a specific domain confirms that its documents should contain general words of particular domain. Based on this, we urge that the characteristic of ICL-corpus is the primary cause of the Cosine behavior in this experiment.

After all, we might infer that Cosine measure is not appropriate to define the semantic relatedness between terms in thesaurus construction of tolerance rough sets model.

6.4 Summary

The result of the study confirmed that tolerance value, data source of thesaurus, and semantic measure influence the quality of the thesaurus. Even though we could not say affirmatively what kind of data source for an effective thesaurus, but empirically the result of study indicated that a set of documents in a corresponding domain with the system data might give better contribution to improve the quality of thesaurus. We also learned that the number of unique terms and total terms in the set contribute more to the quality of thesaurus than the total number of documents. Related to the semantic measure, we suggested to maintain the raw frequency of co-occurrence between terms rather than implementing the other measures, i.e. Cosine.

Chapter 7

Lexicon-Based Document Representation

7.1 Introduction

TRSM employs a vector space model hence it represents the document as a vector of term weight in a high dimensional space. The richer representation claimed as the benefit of TRSM means that there is less zero-valued in document vector. Despite the fact that it can increase the possibility of two documents having non-zero similarity although they do not share any terms in original document, this fact leads us to a presumption that higher computational cost may become a significant trade-off in TRSM.

In Chapter 4, we showed that TRSM was able to fetch the important terms which should be retrieved by the automated process of the system. Nevertheless, based on comparison between the lexicon[1] and of the indexed terms, we identified 64.89% did not occur in lexicon; the contributors were foreign terms (mostly in English), colloquial terms, and proper nouns. The following are the example of colloquial terms: *yoi* (it has the same meaning with word *iya* (in Indonesian) and *yes* (in English)), *terus* (it has the same meaning with word *lalu* (in Indonesian) and *and then* (in English)), *rekans* (it has the same meaning with word *teman-teman* (in Indonesian) and *friends* (in English).

[1]It is an Indonesian lexicon created by the University of Indonesia described in a study of Nazief and Adriani in 1996 (42) which consists of 29,337 Indonesian root words. The lexicon has been used in other studies (10, 37)

In this chapter, we propose a novel method, called a lexicon-based document representation, for a compact document representation. The heart of our method is the mapping process of terms occurring in TRSM-representation to terms occurring in lexicon, which gives us a new document representation consisting only of terms occurring in lexicon (we refer to this representation as LEX-representation) as an output. Consider Fig. 7.1 for depiction of the idea. Hence this method represents a document

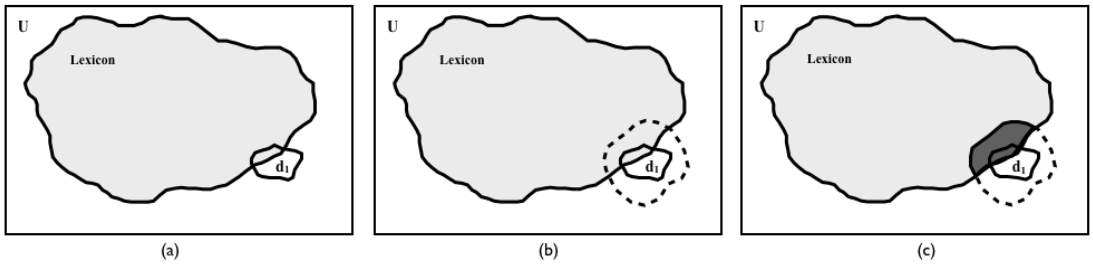

Figure 7.1: The idea of mapping process - Picture (a) shows relation between lexicon and a TFIDF-representation (d_1), picture (b) shows relation between lexicon and a TRSM-representation (depicted by area inside dashed line), while picture (c) shows relation between lexicon and a LEX-representation (depicted by the darkest area).

as a vector in a lower dimensional space and eliminates all informal terms previously occurring in TRSM-representation. By this fact, we can expect less computational cost. For analysis, we take advantage of recall and precision commonly used in information retrieval research to measure the effectiveness of LEX-representation. We also did manual investigation into the list of terms considered as highly related with a particular concept in order to assess the quality of the representations.

7.2 Experiment Process

Experiment in this chapter used two corpora, i.e. ICL-corpus and WORDS-corpus, and employed two types of topic, i.e. 127 topics and 28 topics. The experiment was conducted by following four main phases which were preprocessing phase, TRSM phase, mapping phase and analysis phase as depicted in Fig 7.2. Generally we did the first three phases over both corpora individually and analysed them in the analysis process.

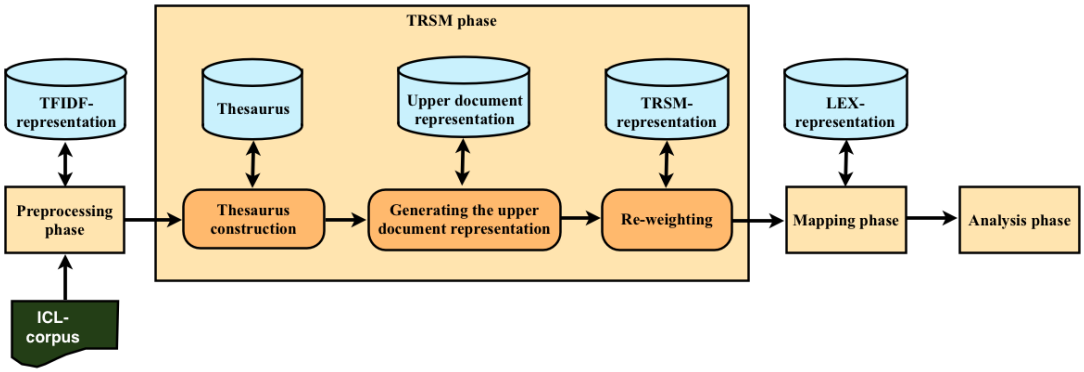

Figure 7.2: Main phases of the study - The process consisted of 4 phases: preprocessing phase, TRSM phase, mapping phase, and analysis phase.

7.2.1 Preprocessing Phase

The goal of this phase is to generate document representation based on the TF*IDF weighting scheme. So, the activities in this study basically the same with the preprocessing phase of experiments in Chapter 4 and 6. This is a phase when we did tokenisation, stopword elimination, stemming, and finally generated TFIDF-representation. This phase was powered by Lucene in which Vega's stopword list and CS stemmer were embedded in it.

In order to work, the CS stemmer requires a dictionary called DICT-UI which showed to produce more accurate results than the use of the other dictionary, i.e. *Kamus Besar Bahasa Indonesia* (KBBI)[2] (37). The DICT-UI is actually the lexicon of this study.

7.2.2 TRSM Phase

In this phase we acted in accordance with the consecutive steps of tolerance rough sets model and came up with TRSM-representation for both corpora. Let us call the TRSM-representation for ICL-corpus and WORDS-corpus as *ICL-TRSM-representation* and *WORDS-TRSM-representation* respectively. In thesaurus construction, we maintained the use of raw frequency of co-occurrence between terms and altered the tolerance value from 1 to 100.

[2]KBBI is a dictionary copyrighted by *Pusat Bahasa* (in English: Language Center), Indonesian Ministry of Education, which consists of 27,828 root words

7.2.3 Mapping Phase

Our intention in this phase is to map the index terms of TRSM-representation into the terms of the lexicon.

We noticed that the total number of terms in the lexicon (29,337 terms) was much bigger than the total number of index terms in ICL-corpus (9,742 terms) and WORDS-corpus (3,477 terms). We also noted that relationship between terms of tolerance classes were constructed based on term co-occurrence in a set of documents, hence there would be no relationship to other terms outside the corresponding set. However, there must be an intersection between lexicon and each document in a corpus because all documents must have some *formal* terms in order to be understood. Consequently, there would be no benefit in considering all terms in the lexicon during the mapping process.

In order to make the process faster, we intersected the lexicon with each corpus and called the result as *known-terms* K. Let $D = \{d_1, d_2, ..., d_N\}$ is a set of text documents, $T = \{t_1, t_2, ..., t_M\}$ is a set of index terms from D, and $B = \{b_1, b_2, ..., b_P\}$ is a set of terms in the lexicon, then $K = \{t_i \in T \mid t_i \cap b_j\} = \{k_1, k_2, ..., k_C\}$, for all $b_j \in B$. The terms appeared in known-terms then became the index terms of LEX-representation. The total number of known-terms for ICL-corpus and WORDS-corpus were 3,444 and 1,566 respectively. The mapping process was conducted as follows

Input:

Matrix of TRSM-representation $TRSM_{matrix} = [trsm_{i,j}]_{NxM}$ for all $t_j \in T$ and $d_i \in D$, where $trsm_{i,j}$ denotes weight of term t_j in document d_i.

Output:

Matrix of LEX-representation $LEX_{matrix} = [lex_{i,l}]_{NxC}$ for all $k_l \in K$ and $d_i \in D$, where $lex_{i,l}$ denotes weight of term k_l in document d_i.

Process:

Generate LEX_{matrix} based on Equation (7.1) for all $t_j \in T$, $k_l \in K$, and $d_i \in D$

$$lex_{i,l} = \begin{cases} trsm_{i,j} & \text{if } k_l = t_j \\ 0 & \text{otherwise} \end{cases} \tag{7.1}$$

Even though we describe the document representations in terms of matrix, in implementation level of experiment we applied the inverted index as the data structure.

We should mention that, during the annotation process, we found that our human experts seemed to encounter difficulty in determining keywords. Thus, rather than listing the keywords, our experts chose sentence(s) from the text or made their own sentence(s). This action made the WORDS-corpus contain both formal and informal terms. Based on this fact, we decided to run the mapping process not only on ICL-corpus but also on WORDS-corpus, in order to remove the informal terms occurring in both corpora. Let us call the resulting representation *ICL-LEX-representation* for ICL-corpus and *WORDS-LEX-representation* for WORDS-corpus.

7.2.4 Analysis Phase

There were two tasks committed in the analysis phase. We named them categorisation and calculation. Categorisation was the task when we clustered documents of the same topics together. The motivation behind this task was based on the annotation process conducted by our human experts, i.e. keywords determination for each document. Thus, we perceived each topic as a concept and considered the keywords in WORDS-corpus as variants of terms semantically related with a particular concept. For this task, we used the 127-topics defined by our human experts, therefore we got 127 classes. Let us call the output of this process *ICL-topic-representation* and *WORDS-topic-representation* for each corpus. Technically, those representations were topic-term matrices.

In a calculation task, we used the notions of recall and precision, which are defined as Equation (4.1) and Equation (4.2) respectively, in terms of calculating the *documents* as well as the *terms*. The first calculation computed the *terms* while the second calculation computed the *documents*. Thus, for the *terms-calculation*, the recall R is the fraction of relevant *terms* that are retrieved while precision P is the fraction of retrieved *terms* that are relevant. Notice that our WORDS-corpus consists of keywords defined by human experts, hence we considered WORDS-corpus as the *ground truth*, i.e. WORDS-corpus consists of *relevant terms* which should be retrieved by automated system.

Briefly, in the *terms-calculation*, we categorised LEX-representation of each corpus and then computed the recall and precision of topic-representations generated with and without a mapping process. Whereas, in the *documents-calculation*, we computed the standard recall and mean average precision (MAP) of all representations.

7.3 Discussion

7.3.1 Calculating the Terms

Based on the *terms-calculation*, our findings are summarised by Fig. 7.3. Those graphs show that the mean of recall and precision values across 127 topics vary by the alternation of tolerance values θ and tend to be smaller as the tolerance value becomes higher.

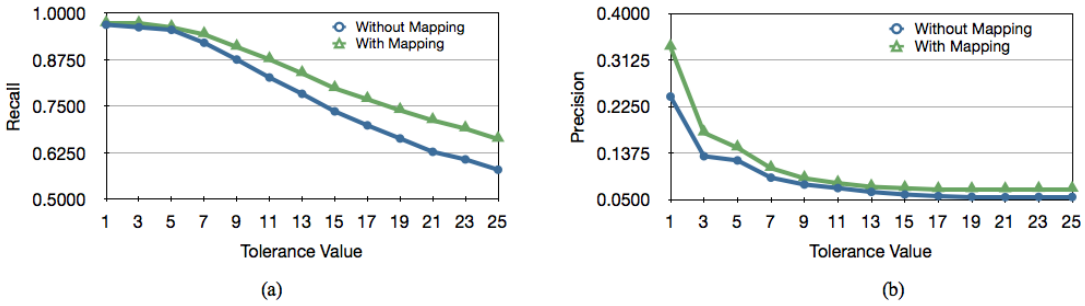

Figure 7.3: Mean of Recall and Precision. - Graph (a) shows the mean of recall values, while graph (b) shows the mean of precision values. The mean were calculated over 127 topics.

We have mentioned in section 7.2.4 that in the *terms-calculation* we focused on *terms* rather than *documents* when calculating recall and precision. Instead of document representation, the recall and precision values were computed over the terms of topic-representation. We measured the quality of topic-representation of ICL-corpus based on the occurrence of relevant terms in it; the relevant terms were the index terms of topic-representation of WORDS-corpus.

Pertaining to the mapping process, we perceive the recall as a value which expresses the ability of the mapping process to keep the relevant terms out of the irrelevant ones. Thus, from Fig. 7.3(a) we can say that the mapping process outperforms the original TRSM method in terms of preserving the relevant terms. A gradual reduction of the ability is shown as the tolerance value θ gets higher, yet the mapping process seems to work better.

From another point of view, by the nature of TRSM method, a greater tolerance value should increase the number of index terms discarded from being introduced into the document representation. Considering Fig. 7.3(b), the behavior seems to shield not

only the irrelevant index terms but also the relevant ones to be chosen to extend the base representation, even though at some point the change is not significant anymore, which happens at $\theta > 17$. However, the mapping process performs better once again in this figure.

7.3.2 Calculating the Documents

In this task, the standard recall and precision were computed using the *trec_eval* program based on TFIDF-representation, TRSM-representation and LEX-representation of ICL-corpus over 28 topics for $\theta = 1$ to 100. Figure 7.4 is the graph of recall while Fig. 7.5 is the graph of mean average precision (MAP). In the figures, `LEX` is the LEX-representation, `TRSM` is the TRSM-representation, and `TFIDF` is the TFIDF-representation.

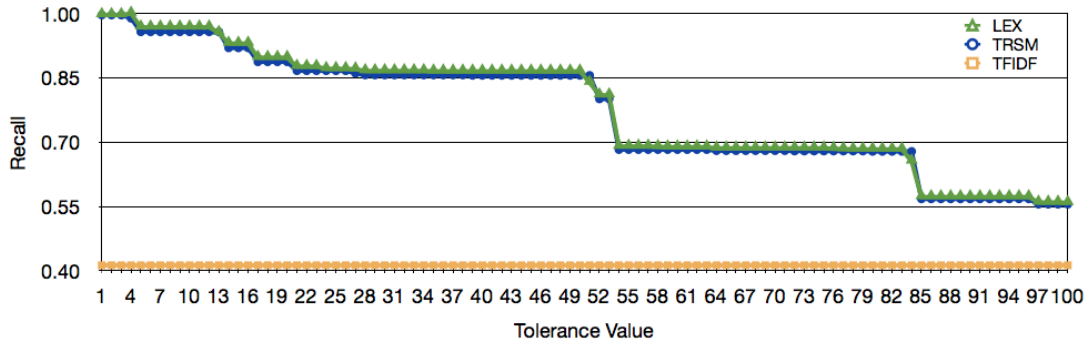

Figure 7.4: Recall. - This graph shows the recall values based on TFID-representation, TRSM-representation, and LEX-representation.

Figure 7.4 displays that LEX-representation works better than TFIDF-representation, even has slightly higher recall values than TRSM-representation at almost all level of $\theta = 1$ to 100. The trade-off to the recall values can be seen in Fig. 7.5. Here, the performance of LEX-representation is shown to be similar with TRSM-representation on low tolerance values ($\theta < 22$) and has slightly better precision at $\theta = 5$ to 15. Compared with TFIDF-representation, it performs better at $\theta < 85$.

The result depicted by Fig. 7.4 and Fig. 7.5 looks consistent with the result presented by Fig. 7.3. Those figures say that the increment of tolerance value leads to the less relevant terms in topic-representation and the more incapable the system to retrieve relevant documents. Even though the mapping process proved to be more

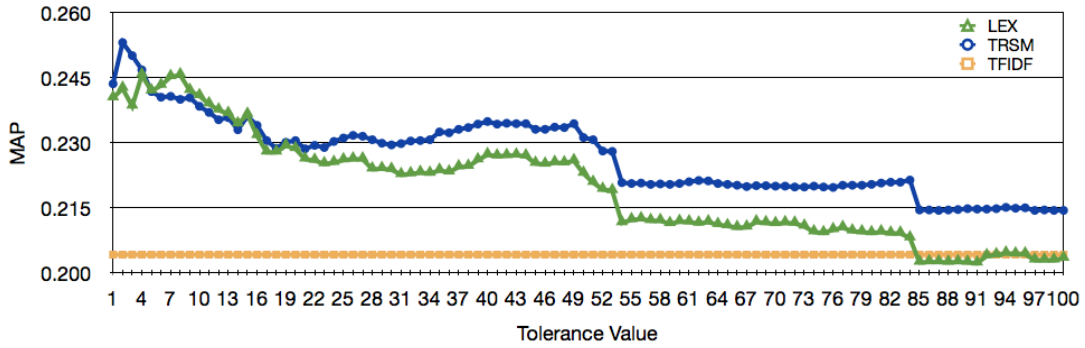

Figure 7.5: Mean Average Precision. - This graph shows the mean average precision (MAP) values based on TFID-representation, TRSM-representation, and LEX-representation.

capable to maintain the relevant terms and the recall value of LEX-representation have proportional result with of TRSM-representation, the mean average precision (MAP) value shows that TRSM-representation performs better in general. Figure 7.6, which presents the mean length of TRSM-representation and LEX-representation for tolerance values between 1 and 100, seems to explain that the vector length has contribution at some degree.

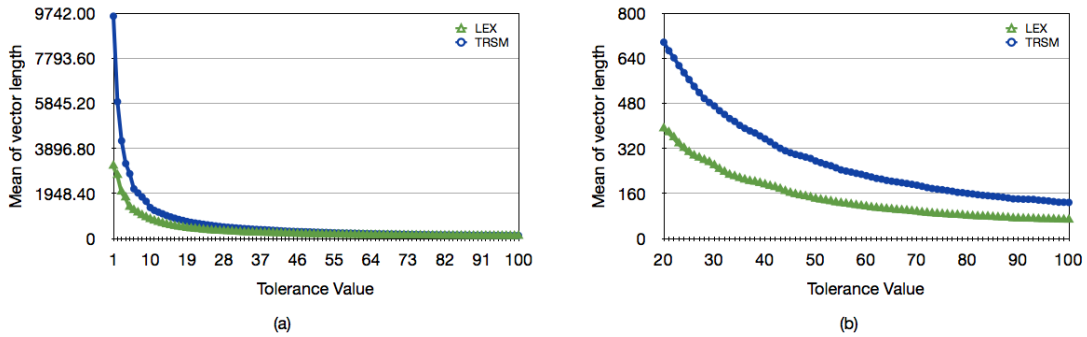

Figure 7.6: Length of vector. - Graph (a) shows the mean length of TRSM-representation and LEX-representation for $\theta = 1$ to 100 while graph (b) is the inset of graph (a) for $\theta = 20$ to 100.

Figure 7.6 tells us that document representation of TRSM tends to be longer than document representation yielded by mapping process. In fact, our observation through all document vectors for $\theta = 1$ to 100 yielded $\overrightarrow{x} \geq \overrightarrow{y}$, where $\overrightarrow{x}$ is document vector of TRSM and $\overrightarrow{y}$ is document vector of mapping process. It is not a surprising result due

to the fact that mapping process conducted based on TRSM, in which the index terms of LEX-representation are those of TRSM-representation which appear in the lexicon.

The document ranking method we used in this study is the Cosine similarity method (B.2), which implies that the largest value of $similarity(Q, D)$ are obtained when the query Q and the document D are the same. Refer to this method, longer vector should have more benefit than the shorter one. Therefore, it is predictable that TRSM-representation outperforms the others when document vector of TRSM is the longest.

It is interesting though that at some levels between tolerance values 1 to 100 the LEX-representation has better performance than of TRSM. So, instead of the vector length, there must be another factor which give significant contribution to similarity computation based on Cosine method. The investigation went further to the tolerance classes which constructed the thesaurus.

7.3.3 Tolerance Class

We picked 3 topics out of 28 which were the most frequent topics in ICL-corpus as it appears in Table C.3. These were *kompetisi* (in English: competition), *konser* (in English: concert), and *karya musik* (in English: musical work), and made an assumption that those topics were concepts which could be represented by a single term for each, namely *kompetisi*, *konser*, and *partitur* (in English: musical score)[3].

We generated the tolerance classes of those terms at several particular tolerance values, i.e. $\theta = 2$, $\theta = 8$, $\theta = 41$, and $\theta = 88$[4]. Specifically, we generated all terms considered semantically related with terms *kompetisi*, *konser*, and *partitur* (based on its occurrence in thesaurus) which appeared on the most relevant document retrieved by the system for each particular topic (i.e. *kompetisi*, *konser*, and *karya musik* respectively). Let us call this term sets as `TolClass_in_document`.

Table 7.1 and Table 7.2 summarise the results; column 1 lists the terms being investigated, while `TFIDF`, `TRSM`, and `LEX` columns present the number of related terms

[3]The index terms of thesaurus are in the form of single term, hence we choose term *partitur* as the representative of the *karya musik* concept.

[4]Figure 7.5 serves as a basis for the choice of θ values in which the TRSM-representation, LEX-representation, TRSM-representation, and TFIDF-representation outperform the other representations at $\theta = 2$, $\theta = 8$, $\theta = 41$, and $\theta = 88$ in respective order. However, particularly at $\theta = 88$, the TFIDF-representation only performs better than the LEX-representation.

Table 7.1: Total number of terms considered as highly related with terms *kompetisi*, *konser*, and *partitur* at tolerance values 2 and 8 in a top-retrieved document representation generated based on TF*IDF weighting scheme (`TFIDF`), TRSM model (`TRSM`) and mapping process (`LEX`). The `Total` column is the total terms of respective tolerance class in thesaurus.

	$\theta = 2$				$\theta = 8$			
Term	TFIDF	TRSM	LEX	Total	TFIDF	TRSM	LEX	Total
Kompetisi	54	1,587	883	1,589	31	315	203	320
Konser	37	3,508	1,664	3,513	23	902	513	909
Partitur	141	2,023	1,037	2,030	30	590	325	597

Table 7.2: The number of terms considered as highly related with terms *kompetisi*, *konser*, and *partitur* at tolerance values 41 and 88 in a top-retrieved document representation generated based on TF*IDF weighting scheme (`TFIDF`), TRSM model (`TRSM`) and mapping process (`LEX`). The `Total` column is the total terms of respective tolerance class in thesaurus.

	$\theta = 41$				$\theta = 88$			
Term	TFIDF	TRSM	LEX	Total	TFIDF	TRSM	LEX	Total
Kompetisi	4	7	4	7	1	1	1	1
Konser	4	92	46	96	3	21	7	21
Partitur	18	54	23	56	1	4	2	4

appeared in TFIDF-representation, TRSM-representation, and LEX-representation sequentially (i.e. the cardinality of `TolClass_in_document`). When $\theta = 2$ we considered those representations with regard to the top-retrieved document calculated based on TRSM model. In similar fashion, for $\theta = 8$, $\theta = 41$, and $\theta = 88$, we considered ones with regard to the top retrieved document based on mapping process, TRSM method, and base model[5]. The `Total` column is the cardinality of particular tolerance class in thesaurus. In other words, it specifies the total terms defined semantically related with term *kompetisi*, *konser*, and *partitur* at $\theta = 8$, $\theta = 41$, and $\theta = 88$.

In a glance we should notice that document vector of TRSM consists of most related terms defined in thesaurus, even at high tolerance value ($\theta = 88$) it includes all of them.

[5]The base model means that we employed the TF*IDF weighting scheme without TRSM implementation nor the mapping process.

Table 7.3: The list of index terms considered manually as highly related with terms *kompetisi*, *konser*, and *partitur*. The last column is the comparable English translation for each related index term mentioned in the middle column.

Term	Related index terms	Comparable English translation (in respective order)
Kompetisi	*kompetisi, festival, lomba, kategori, seleksi, juri, menang, juara, hasil, atur, nilai, jadwal, serta*	competition, festival, contest, category, selection, jury, win, champion, result, regulate, grade, schedule, participate
Konser	*konser, tiket, tonton, tampil, informasi, kontak, tempat, publikasi, poster, kritik, acara, panitia*	concert, ticket, watch, perform, information, contact, place, publication, poster, criticism, event, committee
Partitur	*partitur, lagu, karya, musik, koleksi, aransemen, interpretasi, komposisi, komposer*	musical score, song, creation, music, collection, arrangement, interpretation, composition, composer

It is also clear in both tables that the cardinality of `TolClass_in_document` in TFIDF-representation (showed by the `TFIDF` columns) are mostly the least.

In order to assess the quality of document vector in terms of the relevant terms, we manually made a short list of terms we considered as semantically related with terms *kompetisi*, *konser*, and *partitur*. Table 7.3 displays the lists. By cross referencing our manual list with the `TolClass_in_document`, we found that `TolClass_in_document` consists of at least one term of our manual list. And as predicted, the `TolClass_in_document` of TRSM includes our terms the most.

Let us focus on Table 7.1 when tolerance value is 8. At $\theta = 8$, refers to Fig. 7.5, the LEX-representation performs better than the others, whereas refers to Fig. 7.6 the mean length of its vectors is shorter than of TRSM. Note that the cardinality of `TolClass_in_document` of mapping process (showed by the `LEX` column in Table 7.1 and Table 7.2) for those three terms are smaller than of TRSM. A close observation to the vectors as well as the `TolClass_in_document` turned out that the length difference of both vectors were not too big and most of our manual terms (listed in Table 7.3) were found to sit on top ranks.

Indeed, based on the nature of mapping process, all of relevant terms we confronted occurred in `TolClass_in_document` of mapping process were always at higher rank than of TRSM. It happened because the index terms of LEX-representation were actually those of TRSM-representation which were not dropped out by the lexicon's.

Further, manual inspection yielded that numerous terms in `TolClass_in_document` of TRSM were remotely related to the terms *kompetisi*, *konser*, and *partitur*. With regard to the problem we mentioned in the beginning of this chapter (i.e. the existence of informal terms such as foreign terms, colloquial terms, and proper nouns), the LEX-representation had the most satisfactory result, i.e. it contained only the formal terms, which were index terms of lexicon.

We may infer now, when the total terms in LEX-representation is not in big difference with the total terms in TRSM-representation, we might expect better performance from LEX-representation, which has shorter length but the same relevant terms whose ranks are higher, or in other words, which is more compact. It is practically feasible to improve the quality of LEX-representation by processing the terms more carefully in the preprocessing phase which have never been done by any of our experiments in this thesis.

7.3.4 Time and Space Complexity

The computation cost of constructing the tolerance classes is $O(NM^2)$ (15). In order to generate the LEX-representation, we need to construct the upper document representation and the TRSM-representation which are both $O(NM)$. Going from TRSM-representation to LEX-representation the computation cost is also $O(NM)$. After all, the total cost of mapping process is $O(NM^2)$.

We have mentioned before that the total number of index terms in ICL-corpus was 9,742 and WORDS-corpus was 3,477. As a result, the total numbers of index terms of TRSM-representations for ICL-corpus and WORDS-corpus were the same, 9,742 and 3,477 respectively. After the mapping process, we found that the total number of index terms in both corpora were reduced significantly, 64.65% for ICL-corpus and 54.93% for WORDS-corpus. The mapping process reduces the dimensionality of document vector quantitatively, thus we might expect more efficient computation when we further process the LEX-representation, e.g. for retrieval, categorization, or clustering process.

The use of LEX-representation should give much benefit in applications when efficiency is put on the high priority.

7.4 Summary

We have presented a novel approach for an alternative to a document representation by employing the TRSM method and then run the mapping process, and finally come up with a compact representation of document. The mapping process is the process of mapping the index terms in TRSM-representation to terms in the lexicon.

We analyzed the LEX-representation based on the terms of topic-representation as well as of document representation. By a comparison between topic-representation with and without mapping we have seen that the mapping process should yield a better representation of document, concerning its nature ability to preserve the relevant terms. We have explained that the use of LEX-representation should lead to an effective process of retrieval due to the fact that the mean of recall and precision calculation gave comparable results with TRSM-representation. We might also expect a more efficient process of retrieval based on the finding that LEX-representation has much lower dimensional space than TRSM-representation. We conclude that the result of this study is promising.

Chapter 8

Evaluation

8.1 Introduction

With regard to the intended retrieval system, we proposed some strategies pertaining to the implementation of tolerance rough sets model as we described in Chapter 5 to Chapter 7. All of the strategies were formulated by exploiting our domain specific testbed, namely ICL-corpus.

In this chapter, we are going to present our evaluation on those strategies when they were applied on a retrieval system with different corpus. The aim of evaluation is to validate all of our proposed strategies. Consecutively in following sections, we will discuss the effectiveness of tolerance value generator algorithm, the contributive factors of thesaurus optimization, and the lexicon-based document representation by means of employing another Indonesian corpus, called Kompas-corpus[1] (11), into the retrieval system.

Due to the fact that Kompas-corpus is the only Indonesian testbed available, we generated several corpora from Kompas-corpus as listed in Table 8.1. We named the variations using term `Kompas_X`, where `X` is a number specifies the amount of documents inside it, hence `Kompas_3000` is the original Kompas-corpus who consists of 3,000 documents. In Kompas-corpus, not all documents are relevant with any topic defined in the topic file (i.e. information needs file) of Kompas-corpus. In fact, there are only 433 documents who have relevancy with at least one topic, and those 433 documents were

[1]Kompas-corpus is a TREC-like Indonesian testbed which is composed of 3,000 newswire articles and is accompanied by 20 topics. Please see Appendix C.4 for more explanation.

Table 8.1: The variation of Kompas-corpus.

No.	Variation	Total document	Total unique term	Total term
1.	Kompas_433	433	8,245	85,063
2.	Kompas_1000	1,000	13,288	183,812
3.	Kompas_2000	2,000	19,766	370,472
4.	Kompas_3000	3,000	24,689	554,689

assembled together into the `Kompas_433`. Respectively, `Kompas_1000` and `Kompas_2000` are composed of 1,000 and 2,000 documents in which all documents of `Kompas_433` becomes part of them.

The evaluation data were acquired from experiments following a process depicted in Fig. 8.1 which is a schema for a retrieval system based on TRSM followed by calculating the LEX-representation. We employed all variations of Kompas-corpus listed in Table 8.1 for data source of the thesaurus and used only single corpus, `Kompas_433`, as the main data of the retrieval system for all runs. In the retrieval phase, the information needs and relevance judgments files were loaded in order to produce sets of ranked documents based on TFIDF-representation, TRSM-representation, and LEX-representation.

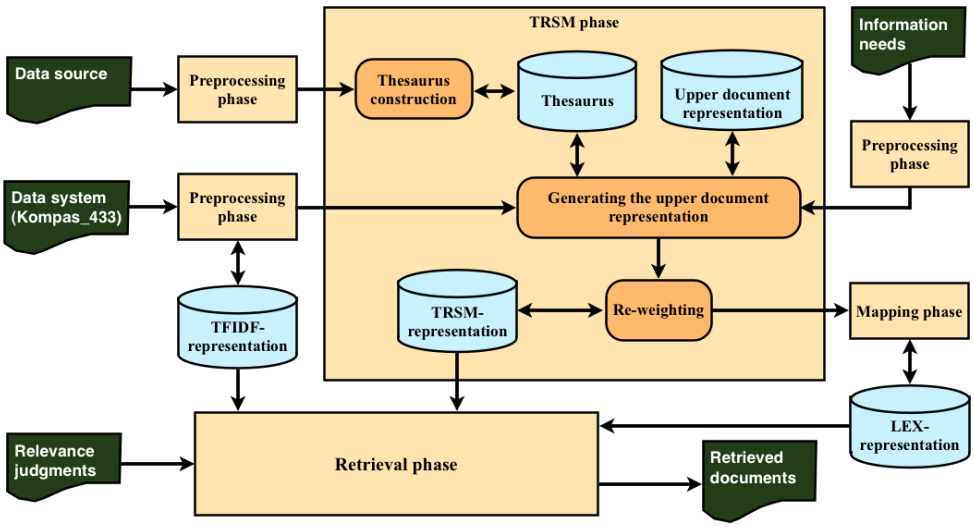

Figure 8.1: IRS based on TRSM - The evaluation was conducted as a retrieval system in which tolerance rough sets model and the mapping process were implemented.

Table 8.2: Tolerance values generated by the TolValGen for each variant of Kompas-corpus functioned as the data source of thesaurus.

No.	Data system	Data source	Tolerance value
1.	Kompas_433	Kompas_433	37
2.	Kompas_433	Kompas_1000	43
3.	Kompas_433	Kompas_2000	46
4.	Kompas_433	Kompas_3000	47

8.2 Evaluation on Tolerance Value Generator

In addition to retrieval system displayed in Fig. 8.1, we ran our tolerance value generator (let us call it *TolValGen* for short) for all variants of Kompas-corpus that served as the data source of thesaurus. Table 8.2 records the tolerance values provided by the TolValGen for each run of different variant.

Figure 8.2 shows the compilation of recall and MAP values of retrieval system for all data sources. From these graphs, we can see that the tolerance values yielded by TolValGen are appropriate since at each resulted θ value the associated corpus performs better then the `TFIDF`.

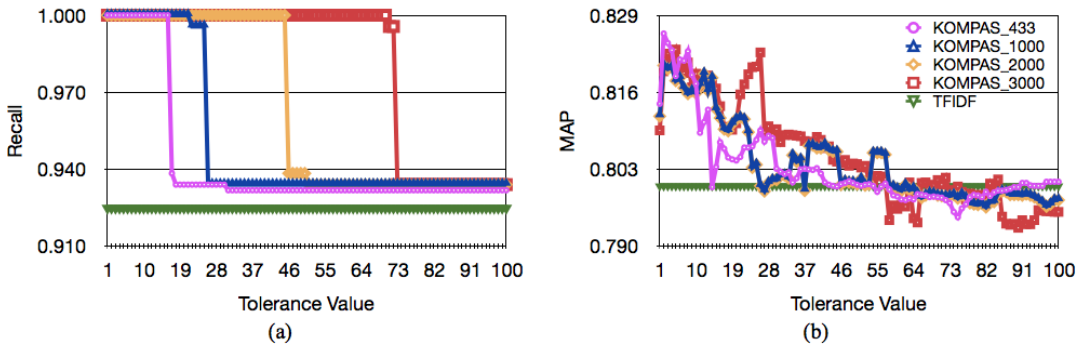

Figure 8.2: Compilation of recall and MAP for TFIDF-representation and TRSM-representation - Graph (a) presents the recall values and graph (b) presents the MAP values of Kompas-corpus variants at $1 \leq \theta \leq 100$.

8.3 Evaluation on Thesaurus Optimization

In Chapter 6 we argued that tolerance value, data source, and semantic measure influence the quality of thesaurus in TRSM. Figure 8.2 that shows the compilation results of recall and MAP of the retrieval system for all variants of Kompas-corpus seems to agree with our argumentation. First of all, those graphs clearly confirm that we might have different quality of thesaurus that leads to different system performance by altering the tolerance value. This claim is supported by Obadi, et. al. (43) who did a study by implementing TRSM in a journal recommendation system based on topic search. They concluded that TRSM is very sensitive to parameter setting.

It is obvious from Table 8.1 that all corpora, `Kompas_433`, `Kompas_1000`, `Kompas_2000`, and `Kompas_3000` have an increasing number of total term and distinct term respectively from one to another. Notice that all variants came from single corpus, hence those corpora are in the same domain with the data system. Considering the amount of terms in each corpus, Figure 8.2 indicates that it agrees with our strategy in maximizing thesaurus quality by applying a set of corresponding documents whose total term and unique terms are larger in number.

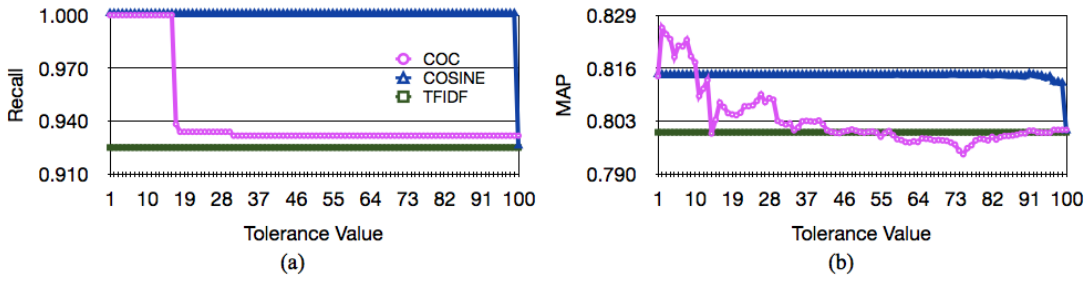

Figure 8.3: Recall and MAP of different measures in thesaurus construction - Graph (a) presents the recall values and graph (b) presents the MAP values of `Kompas_433` corpus at $1 \leq \theta \leq 100$ in which the co-occurrence (COC) and Cosine (COSINE) measures were applied to define the semantic relatedness between terms in thesaurus construction

With regard to the semantic measure in thesaurus construction, working with `Kompas_433` as the main data as well as data source of the retrieval system while applying two different measures (i.e. raw frequency of co-occurrence and Cosine) produced recall and MAP graphs as they are depicted in Fig. 8.3. Put our concern on the Cosine measure (denoted by `COSINE` in the graphs), Fig. 8.3 is similar with Fig. 6.4

and Fig. 6.5 in Chapter 6. Based on this, we acknowledge that Cosine behavior occurs not only for a domain specific corpus such as ICL-corpus, but also for Kompas-corpus whose documents are more differ in topic. However, this finding affirms our assertion that the raw frequency of co-occurrence between terms is more suitable for thesaurus construction in TRSM.

8.4 Evaluation on Lexicon-Based Document Representation

The idea of document representation based on lexicon was confronted by the experimental results shown in Figure 8.4 in which `Kompas_433` served as the data system and Kompas-corpus variants functioned as the data sources. It should come to our notice that the results are not as promising as ones of ICL-corpus.

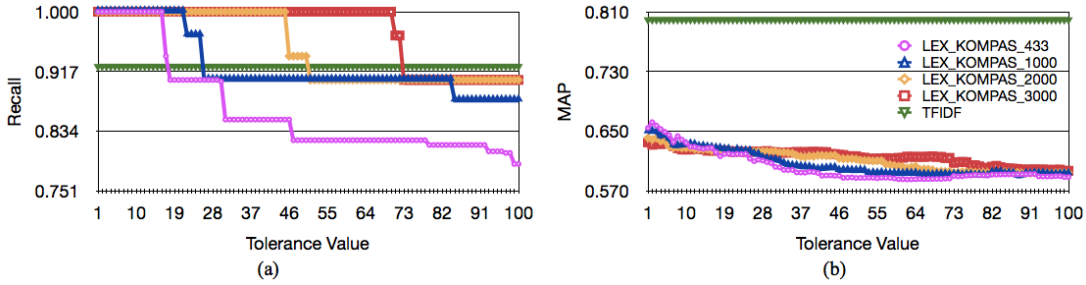

Figure 8.4: Compilation of recall and MAP for LEX-representation - Graph (a) presents the recall values and graph (b) presents the MAP values of LEX-representation of Kompas-corpus variants at $1 \leq \theta \leq 100$.

We did the observation into tolerance classes of `Kompas_433` for terms *banjir* (in English: flood) and *sidang* (in English: trial) at several tolerance values, i.e. 1, 16, 17, and 37. The terms were chosen in order to represent topic *situasi banjir Jakarta* (in English: the flood situation in Jakarta) and topic *persidangan Tommy Soeharto* (in English: the Tommy Soeharto's trial) which have document frequency 40 and 45 respectively. From the study, we found identical characteristic with of ICL-corpus in similar observation, that *(1)* TRSM-representation had most of related terms defined in thesaurus; and *(2)* the cardinality of `TolClass_in_document` in TFIDF-representation were mostly the least.

However, comparison between tolerance classes of TRSM-representation and of LEX-representation made us realize that the lexicon has removed some terms with high relevancy with the topic, which mostly were proper noun. For example, the topics *situasi banjir Jakarta* and *persidangan Tommy Soeharto* include significant proper nouns *Jakarta* (which is the name of Indonesian's capital city) and *Tommy Soeharto* (which is the name of Indonesian second president's youngest son) respectively, and none of those proper nouns are part of the lexicon.

Table C.6 lists the 20 topics of Kompas-corpus and is comprised of 75 unique terms. First of all, it is obvious that almost all topics have proper nouns. Further, we identified that 26.6% of the topic unique terms would be useless in retrieval phase because those terms have been removed from LEX-representation by the lexicon during mapping process, whereas most of the removed terms are proper nouns which are significant in defining the topics. The situation was quite different with the ICL-corpus due to the fact that the topics of ICL-corpus which is comprised of 41 unique terms only have 1 proper noun, i.e. ICL, and thus yielded a compact LEX-representation.

For generalization, we acknowledge that this is a serious problem for LEX-represen tation for it might be corrupted and thus become much less reliable. Considering the fact that a lexicon consists of base words, we may infer that lexicon-based representation is not suitable for general use.

Chapter 9

Conclusion

9.1 The TRSM-based Text Retrieval System

The research of extended TRSM, along with other researches of TRSM ever conducted, acted in accordance with the rational approach of AI perspective. This thesis presented studies who complied with the contrary path, i.e. a cognitive approach, for an objective of a modular framework of semantic text retrieval system based on TRSM specifically for Indonesian.

Figure 9.1 exhibits the schema of the intended framework which consists of three principal phases, namely preprocessing phase, TRSM phase, and retrieval phase. The framework supports a distinction between corpora functioned as data source and data system. In the framework, the query is converted into TRSM-representation by putting the thesaurus and the Equation (6.1) to use while generating the upper approximation and re-weighting the query representation respectively. The mapping phase is included for an alternative and subject to change.

The proposed framework is in Java and takes a benefit of using Lucene 3.1 while indexing. Indonesian stemmer (i.e. CS stemmer), lexicon (i.e. created by University of Indonesia), and stopword (i.e. Vega's stopword) which are embedded make the framework works specifically for Indonesian language; altering them specific to one language would make the framework dependent to that particular language.

It consists of 9 primary classes, in which one of it is the main class (i.e. `IRS_TRSM`[1]), plus single class for the optional mapping phase. Three classes are included in pre-

[1]The source code of `IRS_TRSM` can be found in Appendix D

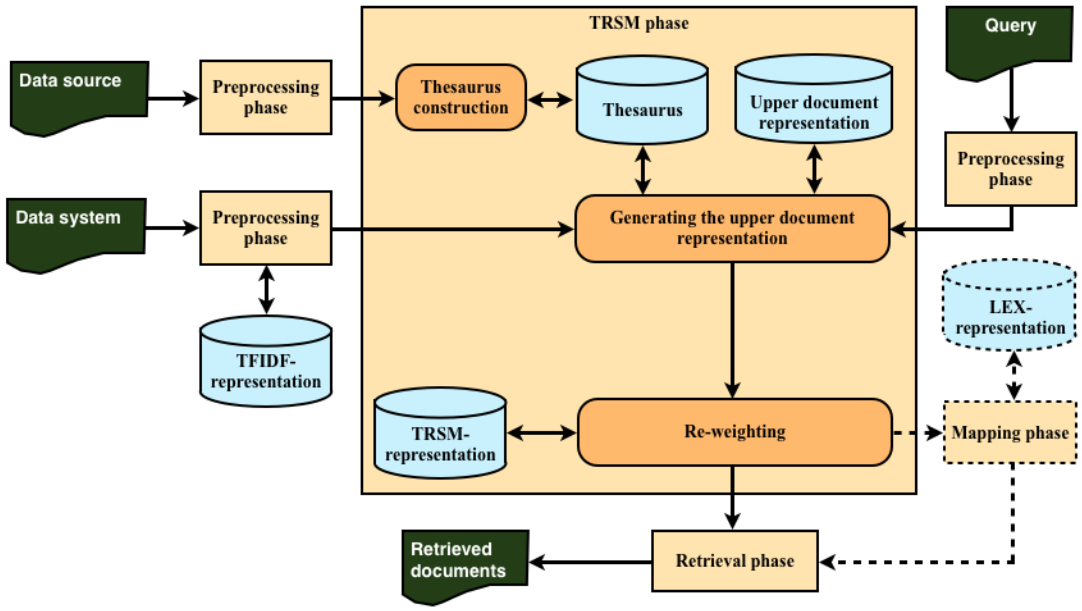

Figure 9.1: The schema of the IRS - The schema of text retrieval system based on TRSM. The dashed line shapes are optional.

processing phase, two classes work for TRSM phase, and three classes are needed in retrieval phase. Figure 9.2 shows the classification of those classes based on the phases of the resulted IRS.

9.2 Novel Strategies for The TRSM-based Text Retrieval System

With regard to the framework of retrieval system, we delved into four issues based on the nature of TRSM. The very first issue questioned about the capacity of TRSM for the intended system, while the other three touched the system effectiveness.

In order to answer the first question, we did a feasibility study whose aim was to explain the meaning of *richness* of the TRSM-representation, rather than listing the strengths and weaknesses of TRSM. By working in close cooperation with human experts, we were able to reveal that the representation of document produced by TRSM does not merely contain more terms than the base representation, it rather contains more semantically related terms. Concerning our approach, we deem this as a stronger affirmation for the meaning of *richness* of the TRSM-representation as well as a sat-

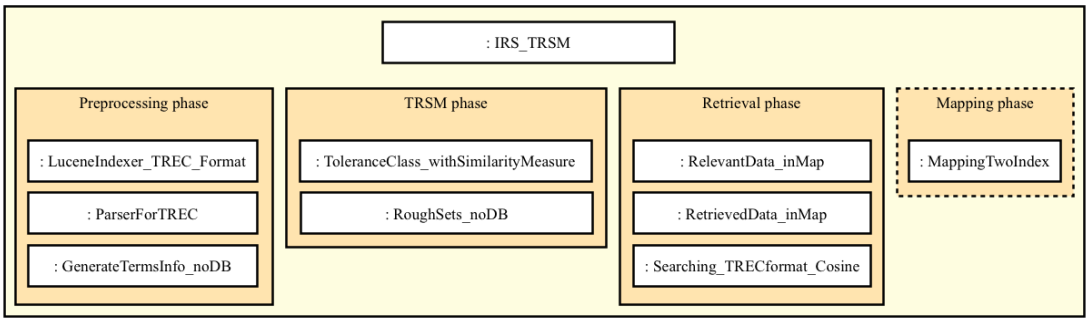

Figure 9.2: Primary classes of the IRS - In total there are 10 primary classes in the proposed IRS, including the `MappingTwoIndex` for the optional mapping phase. The classes are classified based on the three phases of the IRS. The `IRS_TRSM` is the main class.

isfactory indicator in an endeavor to have a semantic retrieval system. Moreover, our analysis confirmed that rough sets theory intuitively works as the natural process of human thought.

Since the TRSM was introduced, no one has ever discussed or examined TRSM's parameter (i.e. tolerance value θ) pertaining to its determination, whereas we consider it as fundamental for TRSM implementation. Obadi et. al. (43) seemed to realize this particular issue by stating that TRSM is very sensitive to parameter setting in their conclusion, however they did not explain or suggest anything about how to initiate it. In Chapter 5 we proposed a novel algorithm to define a tolerance value automatically by learning from a set of documents; and later we named it *TolValGen*. The algorithm was a result from careful observation and analysis performed through our corpora (i.e. ICL-corpus and WORDS-corpus) in which we learned some principles for a tolerance value resolution. The TolValGen was evaluated using another Indonesian corpus (i.e. Kompas-corpus) and yielded positive result. It was capable to produce an appropriate tolerance value for each variants of Kompas-corpus. Figure 9.3 displays the flowchart of TolValGen which works based on SVD.

We recognized that the thesaurus dominates TRSM in its work, hence optimizing the quality of thesaurus became another important issue we discussed. We admit that our idea to enhance the quality of thesaurus by adding more documents specifically for data source of thesaurus did not come up with a promising result as of Nguyen et al. (27, 28) which performed much more clever idea by extending the TRSM such that it accommodates more than one factors for a composite weight value of document

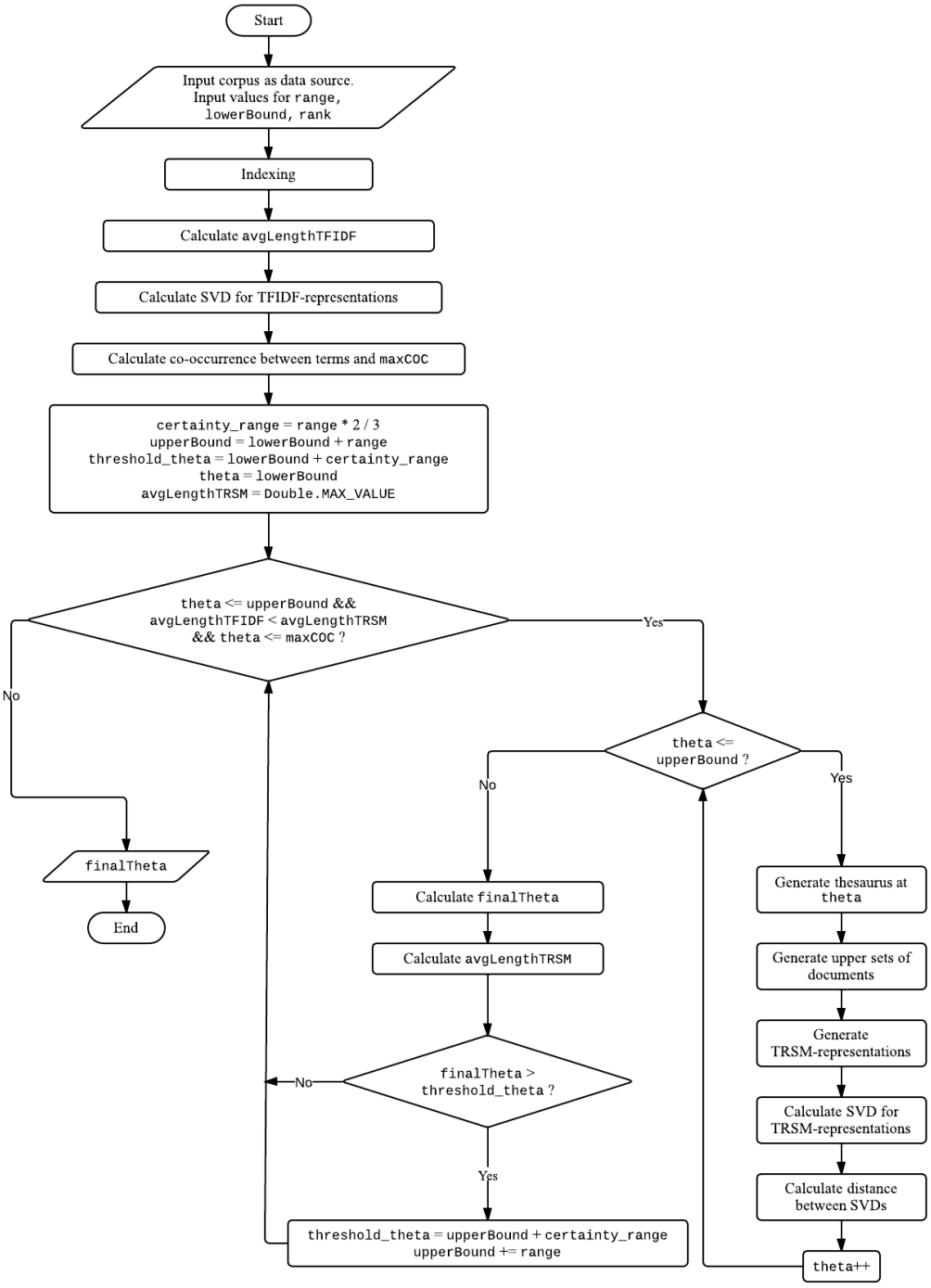

Figure 9.3: Flowchart - The flowchart of TolValGen.

vector. However, from the analysis carried out through several corpora (the variants of ICL-corpus and the Wiki_1800), we learned that tolerance value, data source of thesaurus, and semantic measure determine the quality of thesaurus. Specifically, a data source which is in a corresponding domain with the system data and is larger in number might bring more benefit. We also found that the total number of terms and index terms contribute more to the quality of thesaurus, despite the size of corpus. Finally, we suggested to keep the raw frequency of co-occurrence to define the semantic relatedness between terms for it gave better results in experiment rather than other measure, i.e. Cosine. All of these findings were validated by means of evaluation using Kompas-corpus.

The last issue discussed in this thesis associated with both the effectiveness and efficiency of system. It was motivated by a fact that the richer representation of TRSM is indicated by the larger number of index terms put into the model. Concerning the size of vector dimension, we came into an idea of a compact model of document based on the mapping process between index terms and lexicon after the document enriched by TRSM. The experimental data over ICL-corpus expressed a promising result, however the evaluation through Kompas-corpus remarked differently. Even though numerous irrelevant terms successfully removed from LEX-representation by the lexicon, we learned that our model cannot be applied for general use. The LEX-representation might be easily corrupted and thus become much less reliable when a query comprised of many terms which are not part of the lexicon and those terms are considered significant. Whereas, this particular situation is highly probable to occur in natural language.

9.3 Future Directions

The proposed framework is lack of comparison result. The studies presented in this thesis focused only on the use of TRSM which were compared to the result of TF*IDF. Comparison studies of methods, such those explained in Chapter 2 for semantic indexing, would put TRSM on certain position and bring some suggestion for further development.

The high complexity of our framework is the consequence of TRSM implementation. The application of Lucene module supports the indexing task in preprocessing phase of the framework, however we failed in the attempt to alter the index directly after

TRSM phase which forced us to store the revised-index in different space. We found that it reduced the efficiency of IRS significantly, even though index file was applied. Studies focus on indexing in TRSM implementation is thus essential.

The proposed framework was developed for laboratory environment which is effective for restricted format and type of documents, i.e. follow the TREC-format and written in a *.txt* file. For a real application, our proposed framework should be extended to have the ability to deal with various format and type of documents. Much further, we should consider the recent phenomena of big data[2].

The TolValGen has showed to work on our corpora and their variations. However, it suffers from the expensive time and space to operate. In order to have cheaper complexity for tolerance value generator, further study on this theme with different methods is needed. We might expect some advantage by the use of machine learning method that accommodates the dynamic change of data source.

The lexicon-based document representation is an attempt on system efficiency. Despite the result of evaluation in Chapter 7 which signifies that it is lack of scalability, the fact that we did not implement any other linguistic methods arose our confident that those computations (such as tagging, feature selection, n-gram) might give us benefit in the effort of refining the thesaurus that serves as the basis of tolerance rough sets model, and thus the knowledge of our IRS.

In accordance with Searl's and Grice's accounts on meaning, Ingwersen (45, p. 33) defined that the concept of information, from a perspective of information science, has to satisfy dual requirements: *(1)* being the result of a transformation of generator's knowledge structures (by intentionality, model of recipients' states of knowledge, and in the form of signs); and *(2)* being something which when perceived, affects and transforms the recipients's state of knowledge. Thus, the endeavor of a semantic IRS is the effort to retrieve *information* and not merely terms with similar meaning. This thesis is a step toward the objective.

[2]Big data is a term to describe the enormity of data, both structured and unstructured, in volume, velocity, and variety (44).

Appendices

Appendix A

Weighting Scheme: The TF*IDF

Salton and Buckley summarised clearly in their paper (46) the insights gained in automatic term weighting and provided baseline single-term-indexing models with which other more elaborate content analysis procedures can be compared. The main function of a term-weighting system is the enhancement of retrieval effectiveness where this result depends crucially on the choice of effective term-weighting systems. *Recall* and *Precision* are two measures normally used to assess the ability of a system to retrieve the relevant and reject the non-relevant items of a collection. Considering the trade-off between recall and precision, in practice compromises are normally made by using terms that are broad enough to achieve a reasonable recall level without at the same time producing unreasonably low precision.

Salton and Buckley further explained that, with regard to the differing recall and precision requirements, three main considerations appear important:

1. *Term frequency* (tf). The frequent terms in individual documents appear to be useful as recall-enhancing devices.

2. *Inverse document frequency* (idf). The *idf* factor varies inversely with the number of documents df_t to which a term t is assigned in a collection of N documents. It favors terms concentrated in a few documents of a collection and avoids the effect of high frequency terms which are widespread in the entirety of documents.

3. *Normalisation.* Normally, all relevant documents should be treated as equally important for retrieval purposes. The normalisation factor is suggested to equalise the length of the document vectors.

A. WEIGHTING SCHEME: THE TF*IDF

Table A.1 summarises some of the term weighting schemes together with the mnemonic which is sometimes called SMART notation. One example of the mnemonic is *lnc.ltc*. The first triplet (i.e. *lnc*) represents the weighting combination for the document vector, while the second triplet (i.e. *ltc*) represents the weighting combination for the query vector. For each triplet, it describes the form of *tf* component, *idf* component, and *normalization* component being used. Thus, mnemonic *lnc.ltc* means that the document vector employs log-weighted term frequency, no idf for collection component, and cosine normalisation, while the query vector employs log-weighted term frequency, idf weighting for collection component, and cosine normalisation. Equation A.1 is the common weighting scheme used for a term in a document, i.e. mnemonic *ntn*, which is called TF*IDF weighting scheme.

$$w_{t,d} = tf \cdot idf = tf_{t,d} \cdot \log \frac{N}{df_t} \tag{A.1}$$

Table A.1: Term-weighting components with SMART notation (38). Here, $tf_{t,d}$ is the term frequency of term t in document d, N is the size of document collection, df_t is document frequency of term t, w_i is the weight of term t in document i, u is the number of unique terms in document d, and *CharLength* is the number of characters in the document.

Term Frequency Component	
n (natural)	$tf_{t,d}$
l (logarithm)	$1 + log(tf_{t,d})$
a (augmented)	$0.5 + \frac{0.5 \times tf_{t,d}}{max_t(tf_{t,d})}$
b (boolean)	$\begin{cases} 1 & \text{if } tf_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$
L (log ave)	$\frac{1+log(tf_{t,d})}{1+log(ave_{t \in d}(tf_{t,d}))}$
Collection Frequency Component	
n (no)	1
t (idf)	$log\frac{N}{df_t}$
p (prob idf)	$max\{0, log\frac{N-df_t}{df_t}\}$
Normalisation Component	
n (none)	1
c (cosine)	$\frac{1}{\sqrt{\sum_i (w_i)^2}}$
u (pivoted unique)	$\frac{1}{u}$
b (byte size)	$\frac{1}{CharLength^\alpha}, \alpha < 1$

A. WEIGHTING SCHEME: THE TF*IDF

Appendix B

Document Ranking Method: The Cosine Measure

Manning et al. (38) stated that cosine similarity is fundamental to IR systems that use any form of vector space scoring. Given a query vector and a set of document vectors in a high dimensional space, we may rank the documents by comparing the angle between the query vector and each document vector; the smaller the angle, the more similar the vectors. In linear algebra, the angle θ between two vectors, $\overrightarrow{x}$ and $\overrightarrow{y}$, can be measured as follows:

$$\overrightarrow{x} \cdot \overrightarrow{y} = |\overrightarrow{x}| * |\overrightarrow{y}| * cos(\theta) \tag{B.1}$$

where $\overrightarrow{x} \cdot \overrightarrow{y}$ represents the *dot product* while $|\overrightarrow{x}|$ and $|\overrightarrow{y}|$ represent the lenght of the vectors. The dot product $\overrightarrow{x} \cdot \overrightarrow{y}$ of two vectors is defined as $\sum_{j=1}^{M} x_j * y_j$ and the Euclidean length of a vector $|\overrightarrow{x}|$ is defined as $\sqrt{\sum_{j=1}^{M}(x_j)^2}$. Thus, formula (B.2) can be used to measure the similarity between a query vector Q and a document vector D:

$$similarity(Q, D) = \frac{\sum_{j=1}^{M} w_{qj} * w_{dj}}{\sqrt{\sum_{j=1}^{M}(w_{qj})^2 * \sum_{j=1}^{M}(w_{dj})^2}} \tag{B.2}$$

Appendix C

The Corpora

C.1 ICL-Corpus and WORDS-Corpus

Our original corpus, called ICL-corpus, consists of 1,000 first emails of Indonesian Choral Lovers (ICL) Yahoo! Groups and are formatted as of the Text REtrieval Conference (TREC) format (20). Therefore our test collections consist of three parts (a set of documents, a set of information needs, and relevance judgments) and all documents are marked up in a TREC-like format, i.e. *each document* is marked up by <DOC> and </DOC> tags, the *document number* is marked up by <DOCNO> and </DOCNO> tags, the *subject of email* is marked up by <SUBJECT> and </SUBJECT> tags, the *date of email* is marked up by <DATE> and </DATE> tags, the *sender* is marked up by <FROM> and </FROM> tags, and the *text body* is marked up by <TEXT> and </TEXT> tags.

We worked with two choral experts intensively in the annotation process in order to construct the information needs and relevance judgments for our testbed. The annotation process consisted of two tasks which were *a*) topic assignment, where the human experts assigned topic(s) for each document within the original corpus; and *b*) keywords determination, where they determined terms considered as highly related with the topic(s) given. The annotation process aimed to grasp how the topic(s) could be assigned to a particular document which was mainly described by the keywords determined. We take benefit from these keywords as the list of terms closely related with the topic of document, as well as the document itself, and assume that the other terms not listed are less important terms. The first step of topic assignment yielded 127

topics and the keywords determination yielded a new corpus, called WORDS-corpus.

Consult Fig. C.1 to see the content of both corpora. Notice that the main difference between documents in ICL-corpus and WORDS-corpus lies in the *text body*, i.e. the document of ICL-corpus consists of a body of emails while the document of WORDS-corpus consists of keywords defined by human experts. Fig. C.2 shows the relationship between both corpora.

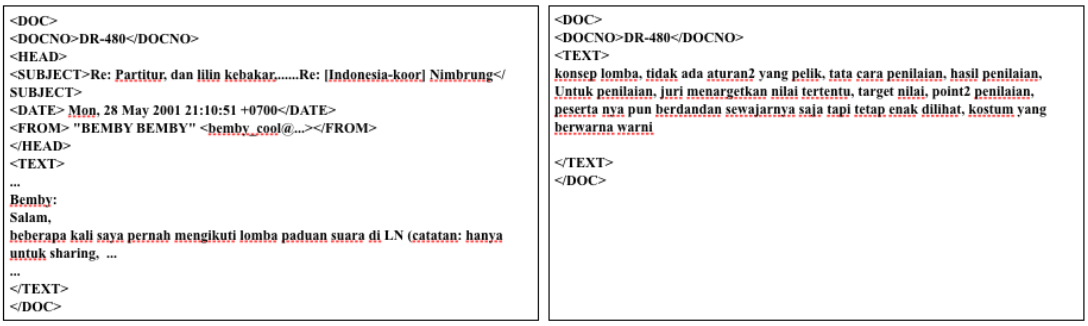

Figure C.1: The content of corpora - Picture on the left is an example of ICL-corpus document which consists of original document, while picture on the right is an example of WORDS-corpus document which consists of keywords given by human expert manually for particular ICL-corpus document, i.e. in this case, the ICL-corpus document with number "DR-480" which is shown on the left.

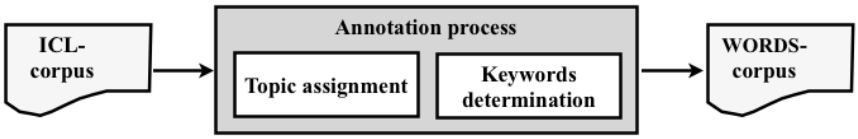

Figure C.2: Corpus relationship - The WORDS-corpus was yielded by human expert in annotation process over ICL-corpus

As we mentioned above, the topic assignment yielded 127 topics of which many have low document frequency; 81.10% of them have document frequency < 10 and 32.28% of them have document frequency 1. We further processed the 127-topics, as it is shown by Table C.1 and C.2, and came up with 28 topics as listed in Table C.3. Thus, we have two version of relevance judgments *a*) relevance judgment which consists of 127 topics; and *b*) relevance judgment which consists of 28 topics.

For the 127-topics, distribution of topics is showed by Table C.4 while list of topics with document frequency ≥ 10 is showed by Table C.5. For all the tables here, *ID*

Table C.1: List of topics. This is a list of 127 topics of ICL-corpus and the total number (document frequency) of relevant documents for each topic with ID 0 to 63.

ID	Topic	DF	ID	Topic	DF
0	Konser	134	32	Manajemen PS	8
1	Partitur	125	33	Peraturan lomba	8
2	ICL	80	34	Manajemen penyanyi	7
3	ICL baru	75	35	Organisasi PS	7
4	Lomba	73	36	Perkenalan	7
5	Tanggapan konser	46	37	Analisa lagu	6
6	KPS Unpar	39	38	Kualitas penyanyi	6
7	Pertemuan	37	39	Melatih PS	6
8	Dokumentasi	35	40	UCV	6
9	Tanggapan lomba	34	41	Bel Canto	5
10	Media PS	33	42	CKO	5
11	Manajemen dana	32	43	Impromptu	5
12	Aplikasi	30	44	LPSAPTI	5
13	Buku vokal	26	45	Pakar PS anak	5
14	Teknikal milis	25	46	Pembayaran	5
15	Festival	17	47	Penampilan	5
16	Interpretasi	17	48	Piano	5
17	Warna tangga nada	17	49	Ad Maiorem	4
18	Kompetisi PS	15	50	Choral sound	4
19	Garpu tala	14	51	File uploaded	4
20	Lokakarya	13	52	ICL file	4
21	Seminar	12	53	KCI	4
22	Lagu sacred	11	54	Pembicara choir building	4
23	Publikasi	10	55	Poling	4
24	Hasil lomba	9	56	PS anak	4
25	Konser bersama	9	57	Alamat	3
26	Koor gereja	9	58	Arti konser	3
27	Penilaian lomba	9	59	Demam panggung	3
28	PS sekolah	9	60	Folklore	3
29	Spam	9	61	FX. Soetopo	3
30	Aturan spam	8	62	Hak cipta	3
31	Istilah musik	8	63	ICL perkenalan	3

Table C.2: List of topics. This is a list of 127 topics of ICL-corpus and the total number (document frequency) of relevant documents for each topic with ID 64 to 126.

ID	Topic	DF	ID	Topic	DF
64	Teknik pernafasan	3	96	FPS ITB	1
65	Teknik vokal	3	97	FPS Unpar	1
66	Tempat konser	3	98	Harga lagu	1
67	Tempat latihan	3	99	Hari haki	1
68	Tommy Prabowo	3	100	Himpunan seniman	1
69	Acara	2		remaja bandung	
70	Aransemen	2	101	ICL poling	1
71	Berita	2	102	Informasi	1
72	Chamber choir	2	103	Interpretasi lagu	1
73	Informasi umum	2	104	Jepang	1
74	Juri lomba	2	105	Kategori lomba	1
75	Memasyarakatkan PS	2	106	Kategori PS	1
76	Pembicara	2	107	Ketua PSM	1
77	Pertanyaan	2	108	Lagu	1
78	Pitch	2	109	Lokakarya musik	1
79	PS GSS	2	110	Maria Luciana Dharmadi	1
80	PS SD	2	111	Pemanasan	1
81	Teknik pengucapan	2	112	Pesan foto	1
82	Tiket konser	2	113	Poster	1
83	Usul	2	114	Poster konser	1
84	Website PS	2	115	PS Perbanas	1
85	Workshop PS	2	116	PS Petra	1
86	Agenda	1	117	PSM Petra	1
87	Artikel konser	1	118	PSM UGM	1
88	Blocking	1	119	PSM Unpad	1
89	BMS	1	120	Respon ICL baru	1
90	Children Choir Network	1	121	Salam	1
91	Choir building	1	122	Sponsor	1
92	Database PS	1	123	Tangga nada	1
93	File	1	124	Tiket	1
94	File konser	1	125	VCD	1
95	Forum	1	126	VCD FPS ITB	1

Table C.3: List of topics. This is a list of 28 topics of ICL-corpus and the total number (document frequency) of relevant documents for each topic.

ID	Topic	DF	ID	Topic	DF
0	Komenter kegiatan	80	14	Orang	16
1	Internal milis ICL	100	15	Referensi	27
2	Kompetisi	181	16	Media paduan suara	33
3	Konser	158	17	Latihan	12
4	Karya musik	125	18	Pertemuan anggota milis ICL	37
5	Perkenalan anggota milis ICL	87	19	Spam	14
6	Manajemen	46	20	Instrumen	19
7	Kelompok musik	52	21	Genre	14
8	Aplikasi	38	22	Tangga nada	18
9	Hal teknis milis	33	23	Seminar atau pelatihan	28
10	Teknik vokal	13	24	Hak cipta	11
11	Performa	14	25	Terminologi	11
12	Dokumentasi	38	26	Forum	15
13	Interpretasi karya musik	24	27	Publikasi	14

Table C.4: Topic distribution. This table shows the total number of topic which has document frequency < 10 out of 127 topics.

Document frequency	9	8	7	6	5	4	3	2	1
Number of topic	6	4	3	4	8	8	12	17	41

Table C.5: List of topics. This table presents topics of ICL-corpus with document frequency ≥ 10 out of 127 topics.

DF	Topic	DF	Topic
134	Konser	30	Aplikasi
125	Partitur	26	Buku vokal
80	ICL	25	Teknikal milis
75	ICL baru	17	Festival
73	Lomba	17	Interpretasi
46	Tanggapan konser	17	Warna tangga nada
39	KPS Unpar	15	Kompetisi PS
37	Pertemuan	14	Garpu tala
35	Dokumentasi	13	Lokakarya
34	Tanggapan lomba	12	Seminar
33	Media PS	11	Lagu sacred
32	Manajemen dana	10	Publikasi

column defines the topic identifier, *Topic* column is the topic in Indonesian, and *DF* column is the document frequency or total number of relevant documents with regard to the topic.

Refer to the TREC format, Fig. C.3 is an example of relevance judgment file while Fig. C.4 is an example of the information needs file. For the relevance judgment file, the first column defines the topic identifier, the third column defines the document identifier, and the fourth column defines the relevancy, i.e. 1 if the document is relevant to the topic, and 0 otherwise. The second column is an arbitrary string and in this case brings no information. The information needs file consists of topics (string between <TITLE> and </TITLE> tags) with its description (string between <DESC> and </DESC> tags) and narrative (string between <NARR> and </NARR> tags). It follows the TREC format, thereby marked up by some tags in which each topic is enclosed by <TOP> and </TOP> tags.

```
0   0   DR-882   0
1   0   DR-882   1
2   0   DR-882   0
3   0   DR-882   0
4   0   DR-882   1
5   0   DR-882   0
6   0   DR-882   0
7   0   DR-882   0
```

Figure C.3: The relevance judgment file - This picture is an inset of the relevance judgment file. Respectively, column 1 to 4 are the topic identifier, random string, document identifier, and document relevancy with topic.

<TOP>
<NUM>0</NUM>
<TITLE>komentar kegiatan</TITLE>
<DESC>laporan pandangan mata atau kesan atau tanggapan atau komentar atau kritik suatu kegiatan misalnya konser atau kompetisi atau seminar atau pelatihan atau lokakarya atau choral clinic</DESC>
<NARR>Dokumen yang relevan berisi tentang laporan pandangan mata, kesan, tanggapan, kritik, dan saran tentang sebuah kegiatan (konser, kompetisi, seminar, pelatihan, lokakarya, choral clinic) baik secara mendetil ataupun umum. Pembahasan bisa berkisar tentang lagu yang dibawakan ketika konser atau kompetisi, penampilan penampil secara fisik maupun secara teknis ketika bernyanyi, tempat penyelenggaraan, panggung, dekorasi, dan panitia penyelenggara, materi yang dibawakan ketika seminar/pelatihan, pembicara atau pembawa materi dalam seminar/pelatihan, cara pendaftaran suatu kegiatan, atau besarnya biaya.</NARR>
</TOP>

<TOP>
<NUM>1</NUM>
<TITLE>internal milis icl</TITLE>
<DESC>hal internal milis indonesian choral lovers atau icl</DESC>
<NARR>Dokumen yang relevan berisi informasi internal milis ICL secara khusus, misalnya sejarah milis, penggagas milis, dan anggotanya. Dokumen yang berisi alamat surat elektronik pribadi yang baru, rencana kegiatan anggota milis, atau surat pribadi antar anggota juga termasuk relevan. Dokumen yang bersisi ucapan selamat datang untuk anggota baru tidak termasuk di sini.</NARR>
</TOP>

Figure C.4: The information needs file - This picture is an inset of the information needs file.

C.1.1 Annotation Process

We have mentioned above that the annotation process consisted of two tasks, namely topic assignment and keywords determination, and yielded WORDS-corpus and two lists of topics (127-topics and 28-topics). This was a collaborative work with three choral experts in which four phases were carried out as it is presented in Fig. C.5.

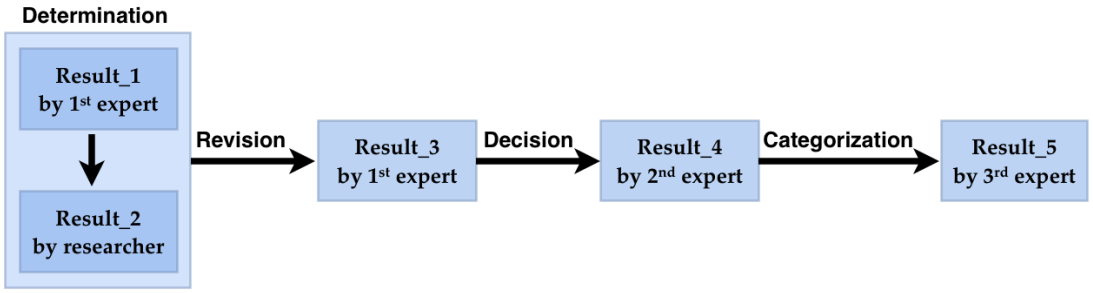

Figure C.5: Annotation process - The annotation process had four phases: determination, revision, decision, and categorization.

First of all, the first expert did the topic assignment and keyword determination for 1,000 documents of ICL-corpus. Considering his work (namely `Result_1`), we did the same thing and came with a different result (namely `Result_2`). Based on `Result_1` and `Result_2`, the first expert did the revision of his previous result and produced the new result (namely `Result_3`). The second expert made a decision (`Result_4`) by analyzing `Result_1`, `Result_2`, and `Result_3`.

On this stage, we had 127 topics and decided to make the list smaller by categorizing it. Thus, we analyzed those topics and agreed on 28 topics. Refer to the 28-topics, the third expert reassigned each documents of ICL-corpus.

In addition to the construction process, this is another main difference of our corpus with an Indonesian corpus made by Jelita Asian from Kompas newswire articles (Kompas-corpus)[1]. In ICL-corpus, each document must be assigned by at least one topic while in Kompas-corpus it is not the case, i.e. there are documents that are not designated to any topics.

[1]Please see Appendix C.4 for more explanation about Kompas-corpus.

C.2 WIKI_1800

WIKI-1800 is a corpus consists of 1,800 text documents in music domain which are the short abstract of Indonesian Wikipedia articles[2]. The full version of the corpus consists of 85,601 short abstracts in variety of topics and was downloaded from DBpedia[3]. The WIKI_1800 employed in this study was obtained by filtering out the 85,610 abstracts specifically based on music domain which was conducted by our third expert.

<http://dbpedia.org/resource/Indonesia_Raya> <http://www.w3.org/2000/01/rdf-schema#comment> "Indonesia Raya adalah lagu kebangsaan Republik Indonesia. Lagu ini pertama kali diperkenalkan oleh komponisnya, Wage Rudolf Soepratman, pada tanggal 28 Oktober 1928 pada saat Kongres Pemuda II di Batavia. Lagu ini menandakan kelahiran pergerakan nasionalisme seluruh nusantara di Indonesia yang mendukung ide satu \"Indonesia\" sebagai penerus Hindia Belanda, daripada dipecah menjadi beberapa koloni."@in <http://id.wikipedia.org/wiki/Indonesia_Raya#> .

<http://dbpedia.org/resource/Music> <http://www.w3.org/2000/01/rdf-schema#comment> "Musik adalah bunyi yang diterima oleh individu dan berbeda-beda berdasarkan sejarah, lokasi, budaya dan selera seseorang. Definisi sejati tentang musik juga bermacam-macam: Bunyi/kesan terhadap sesuatu yang ditangkap oleh indera pendengar Suatu karya seni dengan segenap unsur pokok dan pendukungnya. Segala bunyi yang dihasilkan secara sengaja oleh seseorang atau kumpulan dan disajikan sebagai musik Beberapa orang menganggap musik tidak berwujud sama sekali."@in <http://id.wikipedia.org/wiki/Musik#> .

Figure C.6: WIKI-1800 - An example of WIKI-1800 document.

Figure C.6 shows a small chunk of WIKI-1800 document. Each document is represented as an RDF triple notation which contains three components (i.e. subject, predicate, and object), plus the URL of the Web page. In Fig. C.6, the `<http://dbpedia.org/resource/Indonesia_Raya>`, which acts as the subject, is an URI reference to the resource of *Indonesia Raya*. The `<http://www.w3.org/2000/01/rdf-schema#comment>` (or `rdfs:comment` for short), which acts as the predicate, is an URI reference that refers to the property used to provide a human-readable description of a resource; R `rdfs:comment` L states that L is a human-readable description of R (47). Therefore,

[2]Indonesian Wikipedia: `http://id.wikipedia.org/wiki/Halaman_Utama`.

[3]DBpedia is a community project which was started and is administered by research group from Universität Leipzig, Freie Universität Berlin, and OpenLink Software. The project is an effort to extract information from Wikipedia, make this information available on the Web under an open license, and interlink the DBpedia dataset with other open datasets on the Web. The Indonesian short abstracts of DBpedia was downloaded from `http://downloads.dbpedia.org/3.7/id/`.

the string inside the quotes next to the `rdfs:comment` is the human-readable description of *Indonesia Raya*, which is actually the short abstract of the *Indonesia Raya* article. Finally, the `<http://id.wikipedia.org/wiki/Indonesia_Raya#` is the URL that will go to the Web page of *Indonesia Raya*.

C.3 The Choral Experts

In data preparation of our study, we worked in collaboration with three people who have experiences in choral for years. They were Agastya Rama Listya, Kristoforus Kuntarahadi, and Inke Kusumastuti; in Section C.1.1, we called them the first expert, second expert, and third expert respectively. Figure C.7 displays the pictures of them.

Figure C.7: The choral experts - The choral experts involved in annotation process of our study: (1) Agastya Rama Listya, (2) Kristoforus Kuntarahadi, and (3) Inke Kusumastuti.

Agastya Rama Listya was born in Yogyakarta on February 18, 1968, and now is living in Salatiga, Central Java, Indonesia. He obtained his Bachelor of Arts in Theory and Music Composition from the Indonesian Arts Institute, Yogyakarta, Indonesia, in 1992. In 2001, he received his Master of Sacred Music in Choral Conducting from Luther Seminary and St. Olaf College, Minnesota, USA. He was the Dean of the Faculty of Performing Arts, Satya Wacana Christian University at Salatiga for two periods (2011-2009) and was affiliated as the committee member of Badan Kerjasama Gereja-Gereja se-Salatiga (2007-2010), Lembaga Pengembangan Pesparawi Daerah Jawa Ten-

gah (2007-2010), and Badan Pembina Seni Mahasiswa Indonesia Jawa Tengah (2008-2010). Agastya has published 7 books, 6 articles in journals, and 16 essays. He is a productive music composer and arranger in which many of his choral works were performed by numerous choirs in Indonesia. He is also an active choral coach of a number of choirs where under his direction have made some prominent achievements regionally, nationally, and internationally. Individually, he was the winner of 4 different national choral composition contests during 1998-2009 and the winner of Yazeed Djamin Award for Piano Composition Contest in 2006. Agastya Rama Listya's name was included in the 30th Pearl Anniversary of Marquis Who's Who in The World (November 2012)

Kristoforus Kuntarahadi was born in Yogyakarta on January 14, 1979. He is now a staff in the office of Bishop's Conference of Indonesia, in Jakarta. He was the student of several well-known Indonesian vocalists and chorister, i.e. Avip Priatna, Lucia Kusumawardhani, Yoseph Chang, and Tommy Prabowo. He has been an active singers in some choirs since 1990, including the famous Indonesian choir, Batavia Madrigal Singers in Jakarta, and the tenor solo performer in some concerts. He obtained several achievements on regional singing festival during 1993-1997. Nationally, as a classical singer, he was the runner-up of Bintang Radio dan Televisi (a national radio and television singing competition) in 1995 and the third prize winner of PEKSIMINAS V (a national singing competition for student) in 1999. He received an award from Governor of Yogyakarta as an outstanding vocal artist in 1997.

Inke Kusumastuti is a medical doctor and currently continuing her education in Psychiatry in Udayana University, Denpasar, Bali. She was born in Blitar on April 17, 1986. She did not receive any formal education in music specifically, but she is practically a motivated self-learner when it comes to singing. She got numerous prizes in individual regional singing contests since she was in elementary school (1992-2001). In 2001-2004 she was involved in a band as the vocalist and the band won several regional competitions. In 2003, she experienced to be a cafe singer for a year. After that, while pursuing her medical education, she had been an active sopranos in some choirs, including the Eternal Choir, a well-known semi-professional small choir in Yogyakarta. As a chorister, she was involved in numerous concerts and choral competitions and received some achievements. In 2007, she followed a *conducting* workshop given by Andrew deQuadros in the First Asian Choir Games and, in 2010, she joined a choral clinic given by Marc Anthony Carpio, a choirmaster of Phillippine Madrigal Singers.

Table C.6: List of topics. This is a list of 20 topics of Kompas-corpus and the document frequency, DF, of relevant documents for each topic.

ID	Topic	DF
0	Hubungan Indonesia Australia setelah Timor Timur	11
1	Dampak terorisme terhadap penurunan jumlah turis	2
2	Kecelakaan pesawat udara Indonesia	22
3	Pemberantasan narkoba	18
4	Pemilu presiden Prancis	1
5	Ulang tahun Megawati Sukarnoputri	1
6	Situasi banjir Jakarta	40
7	Duta besar Indonesia	41
8	Nama suami Megawati	40
9	Gejala dan penyebab asma	1
10	Pemenang pertandingan piala Thomas jenis apapun asal Indonesia	8
11	Nama bos Manchester United	27
12	Laporan piala dunia	60
13	Nilai tukar rupiah terhadap dolar AS	74
14	Aktor aktris calon atau pemenang Oscar	3
15	Akibat kenaikan harga BBM	19
16	Susunan kabinet Timor Leste	1
17	Persidangan Tommy Soeharto	45
18	Kunjungan luar negeri Megawati	36
19	Masa jabatan Gus Dur sebagai presiden	3

Recently, in 2012, she got the third prize winner in Bintang Radio RRI Jember (a singing contest conducted by national radio of Indonesia at Jember). Her favorite artist is The Real Group, a world-acclaimed Swedish-based a capella group, which has significantly shaped her current music interest, and her dream is to able to employ music as part of therapy for people with mental disorders.

C.4 Kompas-Corpus

Kompas-corpus (11) is a set of newswire articles collected from a known Indonesian newspaper Kompas[4] published between January and June 2002. It consists of 3,000

[4]Kompas. URL: `http://www.kompas.com`

documents constructed by following the TREC format, thereby accompanied by a file of information needs and a file of relevance judgments. There are 20 topics chosen by a native speaker after reading each documents in order to represent the user information needs. Those topics are listed in Table C.6 as well as the total number of relevant documents for each topic. Out of 3,000, only 433 documents are assigned topic(s).

Appendix D

Main Class of the IRS

Listing D.1: The Main Class: IRS_TRSM

```
/*************************************************************
 * File    : IRS_TRSM.java
 * Note    : Main method
 * Purpose  : An ad hoc retrieval mode of an IRS
 *
 * Process  :
 *     0.  Indexing data source for thesaurus
 *     1.  Indexing data system
 *     2.  Text extraction
 *     3.  Generating relevant data
 *     4.  Searching ⟹ TFIDF–document VS TFIDF–query
 *     5.  Calculating the co–occurrence between terms
 *     6.  Generating thesaurus
 *     7.  Generating the upper representation
 *     8.  Re–weighting
 *     9.  Mapping phase
 *     10. Searching ⟹ TRSM–document VS TFIDF–query
 *     11. Searching ⟹ TRSM–document VS TRSM–query
 *     12. Searching ⟹ TRSM–MAP–document VS TFIDF–MAP–query
 *     13. Searching ⟹ TRSM–MAP–document VS TRSM–MAP–query
 *
 * Author : Gloria Virginia – May 2013
 *
 * It's powered by Lucene 3.1 and using CS stemmer of Indonesian
 *
 *************************************************************/
```

```
27
28
29 package virginia.java.main;
30
31 import java.util.*;
32 import java.util.Date;
33 import java.awt.*;
34 import javax.swing.*;
35 import java.lang.*;
36 import java.io.*;
37 import java.math.BigDecimal;
38 import java.math.MathContext;
39 import java.math.RoundingMode;
40 import lucene.index.*;
41 import virginia.java.roughsets.*;
42 import virginia.java.misc.*;
43 import virginia.java.roughsets.ToleranceClass_usingLucene.
       SimilarityMeasureType;
44 import lucene.index.LuceneIndexer_TREC_Format.FormatType;
45
46
47 public class IRS_TRSM{
48
49   //============================================== fields
50   private String newLine = System.getProperty("line.separator");
51
52   private String corpus, outFolder, locOutFile, runID;
53   private String locIndexDir_Thesaurus, locIndexDir_Data_System,
         locTREC_Data_Thesaurus, locTREC_Data_System, locTREC_Topic,
         locTREC_RJ, locLexicon, locStopword;
54
55   private int totalDataUsed, max_results, topK, totalIteration;
56
57   private ArrayList<Map.Entry<Int,Int>> arrmapTotalRetrievedDoc;
58   private ArrayList<Map.Entry<Integer,String>> arrmapRetrievedDoc;
59   private ArrayList<Map.Entry<Integer,String>> arrmapRetrievedScore;
60
61   private Map<Integer,Integer> mapTotalRetrievedDoc;
62   private Map<Integer,String> mapRetrievedDoc;
63   private Map<Integer,String> mapRetrievedScore;
64
```

```
65  private final int DECIMALS = 4;
66  private final MathContext MC = new MathContext(DECIMALS,
        RoundingMode.HALF_UP);
67  private final BigDecimal BDZERO = new BigDecimal(0, MC);
68  private final BigDecimal BDONE = new BigDecimal(1, MC);
69
70  private PrintToFile ptf;
71  private GetDurationOfProcess gdof;
72  private RetrievedData_inMap retData;
73
74  private Map<Integer,BigDecimal> mapMaxTolClass;
75  private Map<Integer,BigDecimal> mapMinTolClass;
76
77
78  //============================================================ main
79  public static void main (String[] args){
80    IRS_TRSM myMain = new IRS_TRSM();
81    myMain.theProcess();
82  }
83
84
85  //======================================================= constructor
86  public IRS_TRSM(){
87    max_results = 3000;
88    topK = 20;
89
90    totalDataUsed = 433;
91    corpus = "KOMPAS";          // KOMPAS or ICL or WORDS
92    outFolder = "outputFile/Thesaurus";
93
94    //... INPUT files
95    locIndexDir_Thesaurus = "index/index_Data_Source";
96    locIndexDir_Data_System = "index/index_Data_System";
97
98    locTREC_Data_Thesaurus = "data/TREC_"+corpus+"_DOC_"+
          totalDataUsed+".txt";
99    locTREC_Data_System = "data/TREC_"+corpus+"_DOC_"+totalDataUsed+
          ".txt";
100   locTREC_Topic = "data/TREC_"+corpus+"_TOPIC.txt";
101   locTREC_RJ = "data/TREC_"+corpus+"_RJ_"+totalDataUsed+".txt";
102
```

```
103     locLexicon = "data/Lexicon.txt";
104     locStopword = "src/lucene/stemmer/id/IndonesianStopWords.txt";
105
106     ptf = new PrintToFile();
107     gdof = new GetDurationOfProcess();
108
109     runID = "IRS_"+corpus+"_"+totalDataUsed;
110   }
111
112
113     //============================================== theProcess
114   private void theProcess(){
115     System.out.println("\n---------- S T A R T ----------\n");
116
117     //... (0) THESAURUS
118     //... genereate index for thesaurus
119     LuceneIndexer_TREC_Format li = new LuceneIndexer_TREC_Format();
120
121     if (corpus.equals("KOMPAS")) {
122       //... TSR = Kompas
123       li.generateIndex(locTREC_Data_Thesaurus, locIndexDir_Thesaurus
              , LuceneIndexer_TREC_Format.FormatType.FORMAT5);
124     } else {
125       //... TSR = ICL
126       li.generateIndex(locTREC_Data_Thesaurus, locIndexDir_Thesaurus
              , LuceneIndexer_TREC_Format.FormatType.FORMAT1);
127
128       //... TSR = Wiki
129       //li.generateIndex(locTREC_Data_Thesaurus,
              locIndexDir_Thesaurus, LuceneIndexer_TREC_Format.FormatType
              .FORMAT2);
130
131       //... TSR = ICL + Wiki
132       //li.generateIndex(locTREC_Data_System, locTREC_Data_Thesaurus
              , locIndexDir_Thesaurus, LuceneIndexer_TREC_Format.
              FormatType.FORMAT3);
133
134       //... TSR = ICL + ICL/WORDS
135       //li.generateIndex(locTREC_Data_System, locTREC_Data_Thesaurus
              , locIndexDir_Thesaurus, LuceneIndexer_TREC_Format.
              FormatType.FORMAT4);
```

```
136     }
137
138
139     //... (1) INDEXING
140     if (corpus.equals("KOMPAS")) {
141       li.generateIndex(locTREC_Data_System, locIndexDir_Data_System,
              LuceneIndexer_TREC_Format.FormatType.FORMAT5);
142     } else {
143       li.generateIndex(locTREC_Data_System, locIndexDir_Data_System,
              LuceneIndexer_TREC_Format.FormatType.FORMAT1);
144     }
145
146
147     //... (2) TEXT EXTRACTION
148     //... for thesaurus
149     GenerateTermsInfo_noDB gti_tsr = new GenerateTermsInfo_noDB(
            locIndexDir_Thesaurus);
150     gti_tsr.generate();
151
152     //... for data
153     GenerateTermsInfo_noDB gti = new GenerateTermsInfo_noDB(
            locIndexDir_Data_System);
154     gti.generate();
155     gti.calculateMinTFIDFeachDoc();
156
157     //... FIND KnownTerms between index term and lexicon
158     MappingTwoIndex mapping = new MappingTwoIndex();
159     mapping.findCOCterms(gti.getUniqueWords_asMap_TermID(), mapping.
            getMapOfString(locLexicon));
160
161
162     //... (3) GENERATING: RELEVANT DATA
163     RelevantData_inMap relData = new RelevantData_inMap(locTREC_RJ);
164     relData.generateData_inMap(gti.getDocuments_asMap(), gti.
            getDoc_asMap_IntStr());
165
166     Map<Integer, Integer> mapTotalRelevantDoc = relData.
            getTotalRelevantDoc();
167     Map<Integer, String> mapRelevantDoc = relData.getRelevantDoc();
168
169
```

```
170     //... (4) SEARCHING ==> TFIDF-document VS TFIDF-query
171     runSearching(gti.getTFIDFdoc_HashMap(), gti.
            getUniqueWords_asMap_TermID(), "TFIDF", gti.getDocFreq(), "
            TFIDFquery", gti.getDoc_asMap_IntStr(), gti.
            getIndexFile_DocIDinIntStr_inHashMap());
172
173
174     //... (5) CALCULATING SIMILARITY FOR TOLERANCE CLASS / THESAURUS
175     ToleranceClass_withSimilarityMeasure tc = new
            ToleranceClass_withSimilarityMeasure();
176
177     int startVal = 1, endVal = 100;
178
179     boolean COCbased = true;      //... COC based
180     //boolean COCbased = false;    //... NON-COC based
181
182     if (COCbased) {
183       tc.calculateSimilarity_COC(gti_tsr.getOCmatrix());
184
185       //... setting the endVal
186       if (tc.getMaxValue().intValue() > 100) {
187         endVal = 100;
188       } else {
189         endVal = tc.getMaxValue().intValue();
190       }
191     } else {
192       //int measureType = {(1,"COSINE"),(2,"JACCARD"),(3,"DICE")}
193       tc.calculateSimilarity_nonCOC_inHashMap(gti_tsr.
              getIndexFile_inHashMap(), gti_tsr.getTotalUniqueTerms(),
              gti_tsr.getTotalDocs(), gti_tsr.getUniqueWords(), 1);
194
195       //... setting the startVal
196       int intMinValue = tc.getMinValue().intValue();
197       String strTemp = String.valueOf(tc.getMinValue().doubleValue()
              );
198       int index = strTemp.indexOf(".");
199       if (index > 0) {
200         String strValue = strTemp.substring((index+1), (index+3));
201         if (!strValue.isEmpty()) {
202           intMinValue = Integer.parseInt(strValue);
203           startVal = intMinValue;
```

```
204         }
205       }
206       endVal = 100;       //... setting the endVal
207     }
208
209
210     //... (6) GENERATING TOLERANCE CLASS / THESARURUS
211     mapMaxTolClass = new HashMap<Integer, BigDecimal>();
212     mapMinTolClass = new HashMap<Integer, BigDecimal>();
213
214     totalIteration = 0;
215     for (int tolVal = startVal; tolVal < (endVal+1); tolVal+=1) {
216       if (COCbased) {
217         tc.generateTOL_int_HashMap(tolVal,gti_tsr.getUniqueWords());
218       } else {
219         int intDivisor = 100;       //... to divide the tolVal; used
              for Cosine, Jaccard, Dice
220         tc.generateTOL_double_HashMap(tolVal, intDivisor, gti_tsr.
              getUniqueWords());
221       }
222
223
224       //... (7) GENERATING THE UPPER SETS OF DATA
225       RoughSets_noDB genRS = new RoughSets_noDB();
226       genRS.generateURdoc_HashMap(locIndexDir_Data_System, tc.
            getTOL_HashMap(), gti.getDoc_HashMap(), gti.
            getUniqueWords_asMap_TermID(), gti.getTotalDocs());
227
228       mapMaxTolClass.put(new Integer(tolVal), genRS.
            getMaxTolClassValue());
229       mapMinTolClass.put(new Integer(tolVal), genRS.
            getMinTolClassValue());
230
231
232       //... (8) RE-WEIGHTING
233       genRS.generateTRSMdoc(gti.getTFIDFmatrix(), gti.
            getMinTFIDFeachDoc(), gti.getDocFreq(), gti.getDoc_HashMap
            ());
234
235       genRS.computeTRSMdocWeight_HashMap(genRS.getTRSMdocRepr(), gti
            .getUniqueWords());
```

```
236         genRS.computeIndexFile_HashMap(genRS.getTRSMdocRepr(), gti.
                getUniqueWords(), gti.getDoc_asMap_IntStr());
237
238
239         //... (9) MAPPING PHASE
240         mapping.runMapping(genRS.getTRSMdocWeight_HashMap(), mapping.
                getKnownTerms());
241         mapping.computeIndexFile_HashMap(genRS.getTRSMdocRepr(),
                mapping.getKnownTerms(), gti.getDoc_asMap_IntStr());
242
243
244         //... (10) SEARCHING ==> TRSM-document VS TFIDF-query
245         runSearching(genRS.getTRSMdocWeight_HashMap(), gti.
                getUniqueWords_asMap_TermID(), "TRSM", gti.getDocFreq(),
                tolVal, "TFIDFquery", gti.getDoc_asMap_IntStr(), genRS.
                getIndexFile());
246
247
248         //... (11) SEARCHING ==> TRSM-document VS TRSM-query
249         runSearching(genRS.getTRSMdocWeight_HashMap(), tc.
                getTOL_HashMap(), gti.getUniqueWords_asMap_TermID(), gti.
                getDocFreq(), tolVal, "TRSM", "TRSMquery", gti.
                getDoc_asMap_IntStr(), genRS.getIndexFile());
250
251
252         //... (12) SEARCHING ==> TRSM-MAP-document VS TFIDF-MAP-query
253         runSearching(mapping.getWeightDoc_HashMap(), gti.
                getUniqueWords_asMap_TermID(), mapping.getKnownTerms(), "
                MAP", gti.getDocFreq(), tolVal, "TFIDFquery", gti.
                getDoc_asMap_IntStr(), mapping.getIndexFile());
254
255
256         //... (13) SEARCHING ==> TRSM-MAP-document VS TRSM-MAP-query
257         runSearching(mapping.getWeightDoc_HashMap(), tc.getTOL_HashMap
                (), mapping.getKnownTerms(), tolVal, "MAP", gti.
                getUniqueWords_asMap_TermID(), gti.getDocFreq(), "TRSMquery
                ", gti.getDoc_asMap_IntStr(), mapping.getIndexFile());
258
259         totalIteration++;
260     }
261
```

```
262     ptf.printMap_IntegerBD(mapMaxTolClass, outFolder+"/
            TolClass_MAXvalue_TolVal_"+startVal+"-"+endVal+".txt");
263     ptf.printMap_IntegerBD(mapMinTolClass, outFolder+"/
            TolClass_MINvalue_TolVal_"+startVal+"-"+endVal+".txt");
264
265     System.out.println("\n---------- F I N I S H ----------\n");
266     System.out.println("\n");
267   }
268
269
270   //=================================== runSearching - TFIDF
271   /**
272     * Searching for (TFIDF-document VS) TFIDF-query
273     */
274   private void runSearching(HashMap<Integer, HashMap<String,
          BigDecimal>> mapDoc, Map<String, Integer> mapUniqueTerm, String
           corpusType, int[] dfOfTerm, String queryType, Map<Integer,
          String> mapDocID, HashMap<String, HashMap<String, Integer>>
          indexFileData){
275
276     String locOutFileQuery = outFolder+"/Eval/"+queryType+"/
            QueryExpanded_"+corpus+"-"+totalDataUsed+"_"+corpusType;
277
278     //... RETRIEVED data ==> TRSM matrix
279     retData = new RetrievedData_inMap(locTREC_Topic, max_results,
            locOutFileQuery);
280     retData.generateData(mapDoc, mapUniqueTerm, dfOfTerm, mapDocID,
            indexFileData);
281
282     //... clear up the container
283     mapTotalRetrievedDoc = new HashMap<Integer, Integer>();
284     mapRetrievedDoc = new HashMap<Integer, String>();
285     mapRetrievedScore = new HashMap<Integer, String>();
286
287     //... fill in the container
288     mapTotalRetrievedDoc = retData.getTotalRetrievedDoc();
289     mapRetrievedDoc = retData.getRetrievedDoc();
290     mapRetrievedScore = retData.getRetrievedScore();
291     String[] arQueryUsed = retData.getTopicDesc();
292
293     //... print to file
```

```
294     ptf.print_RetRel_Docs_inMap(mapRetrievedDoc, outFolder+"/Eval/"+
            queryType+"/Retrieved_"+corpus+"-"+totalDataUsed+"_"+
            corpusType+"_Doc.txt");
295     ptf.print_RetRel_Docs_inMap(mapRetrievedScore, outFolder+"/Eval/
            "+queryType+"/Retrieved_"+corpus+"-"+totalDataUsed+"_"+
            corpusType+"_Score.txt");
296     ptf.print_RetRel_Total_inMap(mapTotalRetrievedDoc, outFolder+"/
            Eval/"+queryType+"/Retrieved_"+corpus+"-"+totalDataUsed+"_"+
            corpusType+"_TotalDoc.txt");
297     ptf.printString1Array(arQueryUsed, outFolder+"/Eval/"+queryType+
            "/QueryUsed_"+corpus+"-"+totalDataUsed+"_"+corpusType+".txt")
            ;
298
299     //... follow TREC format
300     ptf.print_TREC_result_inMap(mapRetrievedDoc, mapRetrievedScore,
            mapTotalRetrievedDoc, outFolder+"/Eval/"+queryType+"/
            TREC_EvalResult-"+totalDataUsed+"_"+corpusType+".txt",runID);
301   }
302
303
304   //========================== runSearching - TRSMdoc-TFIDFquery
305   /**
306     * Searching for (TRSM-document VS) TFIDF-query
307     */
308   private void runSearching(HashMap<Integer, HashMap<String,
          BigDecimal>> mapDoc, Map<String, Integer> mapUniqueTerm, String
           corpusType, int[] dfOfTerm, int tolVal, String queryType, Map<
          Integer, String> mapDocID, HashMap<String, HashMap<String,
          Integer>> indexFileData){
309
310     String locOutFileQuery = outFolder+"/Eval/"+queryType+"/
            QueryExpanded_"+corpus+"-"+totalDataUsed+"_Tol_"+tolVal+"_"+
            corpusType;
311
312     //... RETRIEVED data ==> TRSM matrix
313     retData = new RetrievedData_inMap(locTREC_Topic, max_results,
            locOutFileQuery);
314     retData.generateData(mapDoc, mapUniqueTerm, dfOfTerm, mapDocID,
            indexFileData);
315
316     printRetrievedData(tolVal, corpusType, queryType);
```

```
317   }
318
319
320   //============================== runSearching - TRSMdoc-TRSMquery
321   /**
322     * Searching for (TRSM-document VS) TRSM-query
323     */
324   private void runSearching(HashMap<Integer, HashMap<String,
          BigDecimal>> mapDoc, HashMap<String, HashMap<String, Integer>>
          TOLmap, Map<String, Integer> mapUniqueTerm, int[] dfOfTerm, int
           tolVal, String corpusType, String queryType, Map<Integer,
          String> mapDocID, HashMap<String, HashMap<String, Integer>>
          indexFileData){
325
326     String locOutFileQuery = outFolder+"/Eval/"+queryType+"/
            QueryExpanded_"+corpus+"-"+totalDataUsed+"_Tol_"+tolVal+"_"+
            corpusType;
327
328     //... RETRIEVED data ==========> TRSM matrix
329     retData = new RetrievedData_inMap(locTREC_Topic, max_results,
            locOutFileQuery);
330     retData.generateData(mapDoc, TOLmap, mapUniqueTerm, dfOfTerm,
            mapDocID, indexFileData);
331
332     printRetrievedData(tolVal, corpusType, queryType);
333   }
334
335
336   //============================== runSearching - MAPPINGdoc-TRSMquery
337   /**
338     * Searching for (TRSM-MAP-document VS) TRSM-MAP-query
339     */
340   private void runSearching(HashMap<Integer, HashMap<String,
          BigDecimal>> mapDoc, HashMap<String, HashMap<String, Integer>>
          TOLmap, String[] arrayKnownTerms, int tolVal, String corpusType
          , Map<String, Integer> mapUniqueTerm, int[] dfOfTerm, String
          queryType, Map<Integer, String> mapDocID, HashMap<String,
          HashMap<String, Integer>> indexFileData){
341
342     String locOutFileQuery = outFolder+"/Eval/"+queryType+"/
            QueryExpanded_"+corpus+"-"+totalDataUsed+"_Tol_"+tolVal+"_"+
```

```
            corpusType;
343
344         //... RETRIEVED data ==> TRSM matrix
345         retData = new RetrievedData_inMap(locTREC_Topic, max_results,
                locOutFileQuery);
346         retData.generateData(mapDoc, TOLmap, arrayKnownTerms,
                mapUniqueTerm, dfOfTerm, mapDocID, indexFileData);
347
348         printRetrievedData(tolVal, corpusType, queryType);
349     }
350
351
352     //========================= runSearching - MAPPINGdoc-TFIDFquery
353     /**
354       * Searching for (TRSM-MAP-document VS) TFIDF-MAP-query
355       */
356     private void runSearching(HashMap<Integer, HashMap<String,
            BigDecimal>> mapDoc, Map<String, Integer> mapUniqueTerm, String
            [] arKnownTerms, String corpusType, int[] dfOfTerm, int tolVal,
             String queryType, Map<Integer, String> mapDocID, HashMap<
            String, HashMap<String, Integer>> indexFileData){
357
358         String locOutFileQuery = outFolder+"/Eval/"+queryType+"/
                QueryExpanded_"+corpus+"-"+totalDataUsed+"_Tol_"+tolVal+"_"+
                corpusType;
359
360         //... RETRIEVED data ==> TRSM matrix
361         retData = new RetrievedData_inMap(locTREC_Topic, max_results,
                locOutFileQuery);
362         retData.generateData(mapDoc, mapUniqueTerm, arKnownTerms,
                dfOfTerm, mapDocID, indexFileData);
363
364         printRetrievedData(tolVal, corpusType, queryType);
365     }
366
367
368     //========================= printRetrievedData - TRSM & MAPPING query
369     /**
370       * Printing the "Retrieved Data"
371       */
```

```
372   private void printRetrievedData(int tolVal, String corpusType,
          String queryType) {
373
374     //... clear up the container
375     mapTotalRetrievedDoc = new HashMap<Integer, Integer>();
376     mapRetrievedDoc = new HashMap<Integer, String>();
377     mapRetrievedScore = new HashMap<Integer, String>();
378
379     //... fill in the container
380     mapTotalRetrievedDoc = retData.getTotalRetrievedDoc();
381     mapRetrievedDoc = retData.getRetrievedDoc();
382     mapRetrievedScore = retData.getRetrievedScore();
383     String[] arQueryUsed = retData.getTopicDesc();
384
385     //... print to file
386     ptf.print_RetRel_Docs_inMap(mapRetrievedDoc, outFolder+"/Eval/"+
          queryType+"/Retrieved_"+corpus+"-"+totalDataUsed+"_Tol_"+
          tolVal+"_"+corpusType+"_Doc.txt");
387     ptf.print_RetRel_Docs_inMap(mapRetrievedScore, outFolder+"/Eval/
          "+queryType+"/Retrieved_"+corpus+"-"+totalDataUsed+"_Tol_"+
          tolVal+"_"+corpusType+"_Score.txt");
388     ptf.print_RetRel_Total_inMap(mapTotalRetrievedDoc, outFolder+"/
          Eval/"+queryType+"/Retrieved_"+corpus+"-"+totalDataUsed+"
          _Tol_"+tolVal+"_"+corpusType+"_TotalDoc.txt");
389     ptf.printString1Array(arQueryUsed,outFolder+"/Eval/"+queryType+"
          /QueryUsed_"+corpus+"-"+totalDataUsed+"_"+corpusType+".txt");
390
391     //... follow TREC format
392     ptf.print_TREC_result_inMap(mapRetrievedDoc, mapRetrievedScore,
          mapTotalRetrievedDoc, outFolder+"/Eval/"+queryType+"/TREC_"+
          corpus+"_EvalResult-"+totalDataUsed+"_Tol_"+tolVal+"_"+
          corpusType+".txt", runID);
393   }
394 }
```

References

[1] Stefan Büttcher, Charles L. A. Clarke, and Gordon V. Cormack. *Information Retrieval: Implementing and Evaluating Search Engine*. MIT Press, Cambridge, Massachusetts, 2010.

[2] Sholom M. Weiss, Nitin Indurkhya, Tong Zhang, and Fred J. Damerau. *Text Mining - Predictive Methods for Analyzing Unstructured Information*. Springer, New York, 2005.

[3] Halvor Eifring and Rolf Theil. *Linguistics for Students of Asian and African Languages*. 2005.

[4] Richard E. Grandy and Richard Warner. **Paul Grice**. `http://plato.stanford.edu/entries/grice/`, May 2006. Accessed 02-10-2012.

[5] J. R. Searle. *Intentionality: An Essay in the Philosophy of Mind*. Cambridge University Press, Cambridge, 1983.

[6] Herbert Paul Grice. *Studies in the Way of Words*. Harvard University Press, United States of America, 1989.

[7] Michael Haugh and Kasia M. Jaszczolt. *Speaker Intentions and Intentionality*, chapter 5, pages 87–112. Cambridge University Press, 2012.

[8] Mahmuda Akand. **Grice and Searle on Meaning**. *Copula - Journal of the Philosophy Department*, **XXVIII**:51–58, June 2011.

[9] Mirna Adriani and Ruli Manurung. **A Survey of Bahasa Indonesia NLP Research Conducted at the University of Indonesia**. In *Proceedings of the 2nd International MALINDO Workshop*, 2008.

[10] Jelita Asian. *Effective Techniques for Indonesian Text Retrieval*. PhD thesis, School of Computer Science and Information Technology, RMIT University, March 2007. Doctor of Philosophy Thesis.

[11] Jelita Asian, Hugh E. Williams, and Seyed M. M. Tahaghoghi. **A Testbed for Indonesian Text Retrieval**. In Peter Bruza, Alistair Moffat, and Andrew Turpin, editors, *ADCS*, pages 55–58. University of Melbourne, Department of Computer Science, 2004.

[12] James Sneddon. *The Indonesian Language: It's History and Role in Modern Society*. UNSW Press, 2003.

[13] Saori Kawasaki, Ngoc Binh Nguyen, and Tu Bao Ho. **Hierarchical Document Clustering Based on Tolerance Rough Set Model**. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, PKDD '00, pages 458–463, London, UK, 2000. Springer-Verlag.

[14] Tu Bao Ho and Ngoc Binh Nguyen. **Nonhierarchical Document Clustering Based on a Tolerance Rough Set Model**. *International Journal of Intelligent Systems*, **17**(2):199–212, February 2002.

[15] Hung Son Nguyen and Tu Bao Ho. *Rough Document Clustering and the Internet*, chapter 47, pages 987–1003. John Wiley & Sons Ltd., 2008.

[16] Yonghui Wu, Yuxin Ding, Xiaolong Wang, and Jun Xu. **On-line Hot Topic Recommendation Using Tolerance Rough Set Based Topic Clustering**. *Journal of Computers*, **5**:549–556, April 2010.

[17] Yi Gaoxiang, Hu Heping, Lu Zhengding, and Li Ruixuan. **A Novel Web Query Automatic Expansion Based on Rough Set**. *Wuhan University Journal of Natural Sciences*, **11**(5):1167–1171, 2006.

[18] Benjamin Martin Bly and David E. Rumelhart, editors. *Cognitive Science: Handbook of Perception and Cognition*. Academic Press, California, second edition, 1999.

[19] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, Inc., New Jersey, third edition, 2010.

[20] Ellen M. Voorhees and Donna Harman. **Overview of the Ninth Text REtrieval Conference (TREC-9)**. In *Proceedings of the Ninth Text REtrieval Conference (TREC-9*, pages 1–14. National Institute of Standards and Technology (NIST), 2000.

[21] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, 1999.

[22] Noam Chomsky. *Language and Mind*. Cambridge University Press, New York, third edition, 2006.

[23] G. W. Furnas, S. Deerwester, S. T. Dumais, T. K. Landauer, R. A. Harshman, L. A. Streeter, and K. E. Lochbaum. **Information Retrieval using a Singular Value Decomposition Model of Latent Semantic Structure**. In *Proceedings of the 11th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '88, pages 465–480, New York, NY, USA, 1988. ACM.

[24] David A. Grossman and Ophir Frieder. *Information Retrieval: Algorithms and Heuristics*. Springer, Netherlands, second edition, 2004.

[25] Evgeniy Gabrilovich and Shaul Markovitch. **Computing Semantic Relatedness using Wikipedia-Based Explicit Semantic Analysis**. In *Proceedings of the 20th International Joint Conference on Artificial intelligence*, IJCAI'07, pages 1606–1611, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[26] S. K. M. Wong, Wojciech Ziarko, and Patrick C. N. Wong. **Generalized Vector Spaces Model in Information Retrieval**. In *Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '85, pages 18–25, New York, NY, USA, 1985. ACM.

[27] S.Hoa Nguyen, Wojciech Świeboda, and Grzegorz Jaśkiewicz. **Extended Document Representation for Search Result Clustering**. In Robert Bembenik, Łukasz Skonieczny, Henryk Rybiński, and Marek

Niezgódka, editors, *Intelligent Tools for Building a Scientific Information Platform*, **390** of *Studies in Computational Intelligence*, pages 77–95. Springer Berlin Heidelberg, 2012.

[28] Sinh Hoa Nguyen, Grzegorz Jaśkiewicz, Wojciech Świeboda, and Hung Son Nguyen. **Enhancing Search Result Clustering with Semantic Indexing**. In *Proceedings of the Third Symposium on Information and Communication Technology*, SoICT '12, pages 71–80, New York, NY, USA, 2012. ACM.

[29] Marcin Szczuka, Andrzej Janusz, and Kamil Herba. **Semantic Clustering of Scientific Articles with Use of DBpedia Knowledge Base**. In Robert Bembenik, Lukasz Skonieczny, Henryk Rybiski, and Marek Niezgodka, editors, *Intelligent Tools for Building a Scientific Information Platform*, **390** of *Studies in Computational Intelligence*, pages 61–76. Springer Berlin Heidelberg, 2012.

[30] Zdzisław Pawlak. **Rough Sets**. *International Journal of Computer and Information Science*, **11**(5):341–356, 1982.

[31] Jan Komorowski, Zdzisław Pawlak, Lech Polkowski, and Andrzej Skowron. *Rough Sets: A Tutorial*, pages 3–98. Springer-Verlag, 1998.

[32] Zdzisław Pawlak. **Some Issues on Rough Sets**. In James F. Peters, Andrzej Skowron, Jerzy W. Grzymala-Busse, Bozena Kostek, Roman W. Swiniarski, and Marcin S. Szczuka, editors, *Transactions on Rough Sets I*, **3100** of *Lecture Notes in Computer Science*, pages 1–58. Springer, 2004.

[33] Andrzej Skowron and Jaroslaw Stepaniuk. **Tolerance Approximation Spaces**. *Fundam. Inf.*, **27**:245–253, August 1996.

[34] Ora Lassila and Deborah Mcguinness. **The Role of Frame-Based Representation on the Semantic Web**. Technical report, Knowledge System Laboratory, Standford University, 2001.

[35] Gloria Virginia and Hung Son Nguyen. **Lexicon-Based Document Representation**. *Fundamenta Informaticae*, **124**:27–45, 2013. To appear.

[36] Vinsensius Berlian Vega. *Information Retrieval for the Indonesian Language*. Master's thesis, National University of Singapore, 2001. Unpublished.

[37] Mirna Adriani, Jelita Asian, Bobby Nazief, Seyed Mohammad Mehdi Tahaghoghi, and Hugh E. Williams. **Stemming Indonesian: A confix-stripping approach**. *ACM Transactions on Asian Language Information Processing*, **6**:1–33, December 2007.

[38] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[39] Michael McCandless, Erik Hatcher, and Otis Gospodnetić. *Lucene in Action*. Manning Publications Co., 2010.

[40] Gloria Virginia and Hung Son Nguyen. **An Algorithm for Tolerance Value Generator in Tolerance Rough Sets Model**. In Manuel Gra na, Carlos Toro, Jorge Posada, Robert J. Howlett, and Lakhmi C. Jain, editors, *Advances in Knowledge-Based and Intelligent Information and Engineering Systems*, KES'12, pages 595–604, Netherlands, 2012. IOS Press.

[41] Gene. H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, third edition, 1996.

[42] Mirna Adriani and Bobby Nazief. **Confix-Stripping: Approach to Stemming Algorithm for Bahasa Indonesia**, 1996. Internal publication.

[43] Gamilia Obadi, Pavla Dráždilová, Lukáš Hlaváček, Jan Martinovič, and Václav Snášel. **A Tolerance Rough Set Based Overlapping Clustering for the DBLP Data**. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and International Conference on Intelligent Agent Technology - Workshops*, **3** of *WI-IAT '10*, pages 57–60. IEEE, 2010.

[44] Mark Troester. **Big Data Meets Big Data Analytics**. `http://www.sas.com/resources/whitepaper/wp_46345.pdf`, 2012. Copyright ©SAS Institute Inc.; Accessed 22-February-2013.

[45] Peter Ingwersen. *Information Retrieval Interaction*. Taylor Graham, London, first edition, 1992.

[46] Gerard Salton and Christopher Buckley. **Term-Weighting Approaches in Automatic Text Retrieval**. **24**(5):513–523, Aug 1988.

[47] Frank Manola and Eric Miller. **RDF Primer**. `http://www.w3.org/TR/2004/REC-rdf-primer-20040210/`, 2004. Copyright ©W3C; Accessed 12-January-2013.