Title	Mining Context-level Associations in Documents Collections using Passages
Author(s)	永井, 健太郎
Citation	
Issue Date	2005-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/532
Rights	
Description	Supervisor:Ho Tu Bao, 知識科学研究科, 修士

Mining Context-level Associations in Documents Collections using Passages

By Kentaro Nagai

A thesis submitted to
School of Knowledge Science,
Japan Advanced Institute of Science and Technology,
in partial fulfillment of the requirements
for the degree of
Master of Knowledge Science
Graduate Program in Knowledge Science

Written under the direction of
Professor Ho Tu Bao

March, 2004

Mining Context-level Associations in Documents Collections using Passages

By Kentaro Nagai (350047)

A thesis submitted to
School of Knowledge Science,
Japan Advanced Institute of Science and Technology,
in partial fulfillment of the requirements
for the degree of
Master of Knowledge Science
Graduate Program in Knowledge Science

Written under the direction of
Professor Ho Tu Bao

and approved by
Professor Ho Tu Bao
Professor Taketoshi Yoshida
Associate Professor Kenji Satou
Associate Professor Tsutomu Fujinami

February, 2004 (Submitted)

Acknowledgments

Just like all other research works, this works cannot be finished without the kind and continuous supports. First of all, I would like to thank my supervisor Professor Ho Tu Bao for his consistent encouragements for accomplishing this works. And I would like to thank Professor Mitsuru Ikeda for the supervision of my sub theme. I also like to thank two assistants in my lab: Dr. Kawasaki and Dr. Nam. Dr. Kawasaki has gave me many advises, especially on the tolerance rough sets. Dr. Nam has provided us the environment of PC-cluster to made it available all the time. I also would like to thank my colleagues. I had learned many things from their talks and discussions. Last but not the least, I would like to my family for the consistent support.

Abstract

The target of this work is to acquire text mined rules that are more easier to be interpreted by using passages as items, which represents the contexts. The result rules will be provided in sets of passages. In most of the past works of association rules mining, the items of rules are words or phrases. The problems are that single words are often ambiguous, and that they disregard the context the word is used. It is known that association rules mining has the problem of producing huge redundant rules. Producing rules that have ambiguous words might be making the problem worse. In this work, *context-level mining* using passage is proposed, in order to avoid producing those ambiguous rules.

We regard passages as excerpts of documents which represents some level of topics or contexts. At current state, paragraphs are used as passages. The representation of the passages is a set of keywords in paragraphs. To fulfill the lack of vocabulary, highly co-occurring words are added on the base of tolerance rough set model. For the problem of matching passages, we extended the distribution hypothesis, "the meanings of the words in the similar contexts tends to be similar" to "similar contexts share a certain amounts of similar or same words." The experiments are carried out on LA Times (TREC5) news articles collection in order to reveal the characteristics of this method. Qualitative analysis on individual rules shows that passage items has higher degree of freedom in representing items than single words or phrases. In addition, our approach tends to produce rules are more easier to be interpreted comparing to word-level rules based on the comparison of number of possible senses of words in rules they might take.

The results shows its effectiveness, but still requires improvements in order to reduce the problem of producing passages that are difficult to interpret. The computational complexcity of current algorithm is on the order of a square of total number of passages. Development effective mining algorithm also awaits for future work.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Mining from unstructured data is receiving increasing attention. According to the Oracle Corporation's report, 90% of the electronical available data is in unstructured or semi-structured format[Tre02].

In early years of KDD, targets were the databases, where data are in structured formats, but after the rise of the internet and the web, mining from the data that is not in structured (i.e. unstructured) format has become an important issue.

Rule mining in textual data is one of the approaches to give users an comprehensive information of large amounts of texts (in most cases) in the form of conjunctive rules. Representation of rules are compact and easy to be understood by users even they are not experts. Those rules could be later used in other application like text classification [CS96].

There are several related domains which also gives comprehensive information about a document or documents collections, representatively document clustering and text summarization. Document clustering, in general, is the task that groups documents that has similar contents. Results are provided in clusters of documents, often represented by keywords in clusters. The well known problem is that the interpretation of cluster of documents are difficult especially when a cluster contain many documents. Another approach, the text summarization, is more close to the natural language processing, which consists of relevant sentence selection and sentence reductions. The differences are that its target is single document but not documents collections [1], and that it reduce the documents based on contents, while the generation of rules gives a reduced information based on statistical inference. In the context of the text mining [Hea99], the former slightly is different from mining since the output is apparent from the content, while the latter's output is not always apparent even after reading the whole documents in collection. Rule mining approach do not suffers from the problem of interpretation of clusters and also reflects the nature of data mining than text summarization. However it has other problems. It is known to produce huge redundant (useless) rules, and especially for texts, produces uninterpretable rules [RB97].

[1] There is also a task so called *multiple texts summarization* but still not popular as one document summarization.

Rule mining in texts based on association rule mining has been conducted by many researchers. One of the initial work showed that associations mining on words or tagged texts produces huge amount of rules, mostly domain dependant compound names and also uninterpretable rules [RB97]. Other approaches has been proposed to reduce redundant rules, by mining only on the selected terms (the word "term" here means close to compound words) [FFK$^+$98], or pruning rules based on different interestingness measure [RT02], but in our observation most of the rules are syntactical relations, well known co-occurrences, or compound nouns.

We summarize the past works from several view points.

- Mining associations between all words in a document is not reasonable because it may produce associations between the word in beginning and one in the end of a document. [The04] shows the work of finding associations of words in a passage (window), producing relations which is more interpretable. *Combination of words from passages (or a context) makes sense than those from documents.*

- Extracting compounds words makes an improvement in that it reduces the number of irrelevant combinations and making the results easier to be interpreted [FFK$^+$98]. The reason is clear since it does not produce a redundant rules like "joint → venture", where the "joint venture" makes one meaning. In our observation, it is the choosing of *meaningful unit as items* that makes the rules to make sense.

- Mining the associations disregarding the importance of the items in context, will produce well known syntactic relations or compound words, because they do not correctly identify the importance of the words [RT02]. Evaluating *the correct importance of words* is important.

We also identified additional problems.

- The rules become hard to be interpreted because the words appear in rules are ambiguous. It is not until we go back to original texts (contexts) that we understand the meaning of the associations.

- The relevance of words differs document to document, but also context to context. This is true especially when the documents contain several subtopics.

Reflecting our observations, we propose a *context-level* associations mining, which mine rules using passages as items. With passage items, we expect to have rules that are more easier to be interpreted, and shows more not only the syntactical relations.

In order to mine on context-level, following problems must be solved: passage identification, passage representation, and passage matching. We must first extracts passages from documents. For the current state we selected the datasets with paragraphs markers and regarded those as passages. For the representation, set of keywords are used. For the last issue, we extended the distribution hypothesis: "the words in a similar contexts tends to be similar" to "two contexts share certain amount of words when the are similar."

Passages are identified as same contexts if the similarity of two passages is greater than a pre-defined threshold.

We carried out experiments in the framework described above. The objective of the experiments is to reveal its characteristics. The target data is the news articles from LA Times (TREC5). We mined the document collection and analysed the rules and carried out the quantitative analysis, comparing the number of possible senses the words in the rules might take. Qualitative analysis shows that rules that use passage items has higher degree of freedom in representing items, and to some extent, the contextual information. In addition, quantitative analysis shows that rules using passages will will reduce the ambiguity of words in rules.

The contribution of this paper is that we proposed and evaluated the usefulness of associations mining that incorporates the contextual information and showed its comprehensiveness over word-level rules.

Organization of the paper is as follows. The basic notions and previous works are explained in next chapter. In section Chapter.3, the context-level association mining is introduced. Chapter.4 gives the details of the implentations. In Chapter.5 results of the experimental evaluation on several datasets are shown. Remarks on results and future works are given in Chapter.6. The final chapter is the conclusion.

Chapter 2

Background

This chapter recalls the basic ideas of frequent itemset mining and its application to textual data. The use of passages and its the disciplines are also described.

2.1 Frequent Itemset Mining in Standard Data

The basic idea of *association rule mining* was introduced in [AIS93]. In short, the goal of association rule mining is to discover implication rules from the given transactional database. Rules will look like

$$X \rightarrow Y(s, c)$$

where X and Y is a set of items, and s and c are the values representing the joint probability of seeing X and Y, the conditional probability of seeing Y given X in the database, respectively.

The notion *frequent itemset mining* refers to a part of the *association rule mining* process. The problem of association rule mining consists of two parts: frequent itemset mining and rule discovery. The task of the first half is to find all itemsets that appear more than the support threshold. The task of second half is to prune the itemsets that have confidence value less than the threshold. Since, in many cases, confidence values will not change significantly by the moving items between antecedents and consequents, or simply of no interest, often the former half is focused. But note that the task of frequent itemset mining and rule discovery is separate process, and rule discovery can be applied after any frequent itemsets mining algorithms.

The following is the formal definition of the frequent itemset mining [AS94]. Let $\mathcal{I} = \{i_1, i_2, ..., i_m\}$ be a set of items. Let $\mathcal{D} = \{t_1, t_2, ..., t_n\}$ be a set of transactions, where each transaction is a set of items $t_i \subseteq \mathcal{I}$. Each transactions is assigned a unique identifier tid_i, and let $\mathcal{T}$ be the set of all *tid*s. The target is to find the sets of itemsets such that sets of itemsets $\mathcal{IS}_s = \{is_k \bigm| is_k \subseteq \mathcal{I}, support(is_k) > s\}$ where

$$support(is_i) = \frac{\bigl|\{t_i \mid is_i \subseteq t_i, t_i \in \mathcal{D}\}\bigr|}{\bigl|\mathcal{D}\bigr|}.$$

Rules of these kinds are especially useful for marketing basket analysis and studied excessively as a tool to find frequent patterns, correlation, and/or causal structures. Together with clustering, association rules mining is one of the most important tools for description task in KDD process.

The current study of association rule mining is focused on extensions of original association rules mining, where the target is on boolean sets and the results are all sets of items. One way of extension is to mining only certain kinds of sets, like maximal itemsets or closed itemsets. Other way of extension is to mine in other data types, like quantitative values [SA96], sequential data, graph data [WM03], and, the target of this work, textual data.

2.2 Frequent Itemset Mining in Textual Data

The extension of frequent itemsets mining to textual data has been studied by many researchers.

The difficulty of applying traditional frequent itemset mining to textual data is the unstructured nature of texts. To apply standard association rule mining to textual data, one must convert the textual data to structured format, which standard tools can process.

Many approaches has been proposed for this problem. As described in previous section, database is a set of transactions, and transactions is a set of items. In short, the problem is to convert texts in to sets of items and transations. Since the frequent itemset mining is a descriptive process [HMS01], the selection of transactions and items depends on the purpose of applications and, in fact, many researcher took different linguistic units/information as transactions and items for different purposes.

In the following sections, related works are introduced focusing on the selections of transactions and items. First the selection of transactions is discussed, and then the selection of items are discussed in three classes of our classification: word-level, term-level, and entity-level.

2.2.1 Selection of Transactions

For selection of transactions, The approaches can be classified into three main groups: documents [Raj97] [FFK$^+$98], windows [AHKV97], and passages [The04]. In traditional market basket analysis, one transaction represents one purchase or one event. In texts the event can be interpreted as the scope of co-occurrence. For example, if you chose documents as transactions, you are to find co-occurrences of words within documents, chosen passages as transactions, rules will be the co-occurrences within passsages. In the first example, you were to regard any pair of words as co-occurrence regardless of how close they appear: they might appear next to each other but they might be co-occurring in opening and closing of the documents. In many cases, this assumption is too strong. Thanaruk [The04] used passages as transactions to bring some level of proximity in counting the co-occurrences. The results indicates that words from same context are easier to understand than those from documents. Our work reflects this fact that a word set from context is much meaningful compared to those from documents.

2.2.2 Selection of Items (Word-level)

This approach uses all or most of occurrences of words as items. Initial work is done by Rajman et al. [RB97]. They used documents as transactions and words as items. Their experiments showed that it produces many redundant rules including rules that are un-interpretable. Ahonen et al. [AHKV97] and their successive works can be seen as word-level but their target is to mine phrases like *knowledge discovery in* → *databases* and applies sequential mining in the word that appear in a given window size. So the goal is different from ours.

Raghaven and Tsaparas [RT02] is another work that mines association between words, but considering the importance of co-occurrence pair by various sentence frequency based measures. Some of the mined relations are, “deutsch telekom”, “hong kong”, “chevron texaco”,”chateau empress frontenac”, “indigo reisman schwartz” and “del monte sunrype”. Although concidering of importance of words made an improvement, in our subjective view, the results are mostly compounds names, which might not be unexpected, interesting results.

As a summary of word-level selection of items, it can be seen that the results are “uninterpretable” [Raj97], “well-known syntactic structure” [RT02], well known co-occurrences, or compounds name. These information might be useful for further steps of text mining but it is still poor to give interesting information by itself. It tells us linguistically useful information, but will not tell us much information for real world use, which is more important in a task like topic detection and tracking.

2.2.3 Selection of Items (Term-level)

Second class is the ones which use terms as items, conducted by [RB97], [FFK$^+$98]. The word *term*, in Feldman et al.[FFK$^+$98]'s sense, is an adjacent of words that conform single meaningful unit like *net income* or *joint venture*. The process is straightforward. First they parse the documents to extract terms based on the co-occurrences, pruning unimportant co-occurrences, and then apply association rules mining to find associations between terms. They mined relation of joint ventures like “(america online inc)(bertsmann inc) → (joint venture).” The results are more interpretable and meaningful compared to the word-level ones. We see the reason of this results as a result of carefully choosing the meaningful items, pruning items that are not interesting in itself. This approach enables effective computation and reduces meaningless rules, but might be missing the chance to produce the rules which become interesting after the combination of not interesting items. This problem is addressed again in next section.

2.2.4 Selection of Items (Entity-level)

The last class is to use entities as items [NM01] [FAF$^+$99] [CC99]. Entities are the words or terms extracted by information extraction method. Information extraction is a method to extract entities like person name, location, company names from free texts, often guided by some kind of prior knowledge about the entities to extract. An example is to extract job type, required skills, ages, and locations from job listings. This class uses information extraction as a first step and then apply association rule mining or its variants to mine rules to find relationships between entities. The result rules are comparable to or better than term-level approaches in the sense of understandability or usefulness. This can be said because the entities are words or terms extracted with some level of intention. So it might be fair to assume that rules provide minimum

level of interestingness to users since they already filtered out the non-interesting items. In this term, this process can be seen as noise reduction or feature selection and these works are especially effective in reducing huge number of redundant rules by focusing on items of interest.

But there might be some limitations for this approach. Most limitation comes from the limitation of information extraction methods. Information extraction often requires templates or extraction rules. These prior knowledge is often differs from domain to domain, and therefore it can be said that this approach has some limitations of domain dependence[1]. Another questionable problem for this approach is that they might be dropping useful items in what they regarded as noise. As mentioned above, information extraction are often used with some level of intention. The kinds of the words to be extracted are often required to be predefined. In this way, relationships between known items could be found, but the relationships between un-awared items could not. This problem might not be significant in many cases, but we must be aware that we are losing the chances to find relationships between those unpredicted items.

Finally Table.2.1 shows the summary of the selection of transactions and items, together with subjective remarks about result rules.

	Transactions	Items	Subjective Remarks on Rules
Rajman 1997	Document	Word	Co-occurrence, Syntactic relation
Ahonen 1997	Window	Word	Phrase
Feldman 1998	Document	Term	Entity co-occurrences
Clifton 1999	Document	Entity	Entity relation with typed information
Thanaruk 2000	Passage	Word	Co-occurrence within similar context
Raghaven 2002	Document	Words (scored)	Compound Names

Table 2.1: Selection of Transations and Items of Related works

[1] Recent study of information extraction uses finite state transducers or ontologies-driven approaches to overcome this problem, so here we only refer to the template-driven information extraction.

2.3 Passages as Representation of Contexts

As an alternative solution to selection of items we propose contexts as items (represented by passages). The idea of using passages has a long history in other fields and we could see several reasons that passages could be reasonable solution to our problem. We briefly explain the use of passage in other domains and explain why we see passages as good units for items.

The notion *passage* is used in a similar way as *paragraph.* Paragraph is a excerpt of a document which usually represents one topic or an assertion. When the document is long, the document often consists of several subtopics and in some cases the human interest lies not in whole documents but in the passages.

The good example is the work of Salton et al. [SAB93] that consider the importance of words in passages in information retrieval. They break down the documents into passages and calculate the relevance of each passage to the query. The relevance of documents to query is determined by the sum of relevance of passages to the query. The results showed that their approach is useful especially for long documents. This can be seen that passages are *good unit of users' interests.*

Another use of passage can be seen in question and answering task. In question and answering task, retrieval of passage that possibly contains the answer is often the first step. The target of passage retrieval is to identify the place of answer. This facts is indicating that passages are the *units where the fact/evidence lie.*

The last example is from word sense disambiguation fields [MT94], where the passage is seen as a local context where the sense of words are consistent. In the task of word sense disambiguation, words' sense can be identified in two ways. one sense per discourse or one sense per collocation. This is on the distribution hypothesis: "the meanings of the words in similar contexts tends to be similar." It indicates the words' sense cannot always be identified uniquely in the document, but change passage to passage and the use of contexts in word sense disambiguation indicates that *provided with the surrounding words the words are more easier to be identified* in that the words' meaning is clear.

In short, we observed the advantages of using passages or contexts in followings reasons.

- Passages are units of users' interest
- Passages are units of facts/ evidence
- Passages are easier to be interpreted than words alone

In the next chapter, the methods of utilizing the passages and the issues concerning to using passages will be addressed.

Chapter 3

Methodology

In this chapter, the problem definition is given, issues are addressed and the methods are described.

Our approach is to use passages as items and mine associations between passages. In next we explain why using passages is promissing approach. And then the problem definition is given. We show the results of the preliminary experiments to show the difficulties when using passages.

3.1 Philosophical Reasons (Why Passages?)

Before going into details of methods. The reasons why using of passage is also promissing in rule mining is explained.

The first reason is that passages can be seen as meaningful units. As mentioned in previous chapter, Salton et al. [SAB93] showed that passages are good unit of users' interest, the use of passage in question and answering task shows the passage being used as units of fact or evidence. These results indicates that passages are the units of human interest. Mining using these units as items are promissing in that each items are of interest and hence we could expect the rules implicating the associations between these units also be of interest. As a result we expect the method to produce only the rules of users' interest, avoiding to produce non sense rules.

The second reason is that passages are more easier to interpret than words alone. As seen in previous chapter, past works are indicateing that contextual information are useful for disambiguating the word senses. Passages can be less ambiguous because it is possible to reduce the number of ambiguous words. We expect to get rules that are less ambiguous than the rules that consists of single words.

Finally, passages can be efficient in the sense that there is less risk of producing rules about words that appear in different contexts. We use only relevant words in passages. So the words that appear frequently but not with relevance will be filtered out. This will also avoid adding support from irrelevant contexts.

For these reasons, we expect to acquire rules more useful to users and telling much more about the contents.

3.2 Problem Definition

The goal of this work is to find associations between passages, in order to reveal how typical the set of passages are in the documents collections. As described in previous section, passages can be seen as good indicator of contexts. We regard passages (in any representation) as contexts. After the passages are extracted from the documents, the method is straight forward. Documents are regarded as bag-of-passages, that is, documents will be the transactions and the passages will be the items.

We identified three issues, we face when using passages on the framework of standard association rules mining: passage identification, passage representation, and passage matching.

The problem of passage identification is how to split the documents into set of passages. The most simple solution is to use the paragraph markers, since the paragraph is the boundary of the contexts given by the author. The problem is that those markers are not always provided. Another way is to use sliding windows. Users specify the number of words or sentences to split the documents. Windows could be non-overlapped or overlapped. The size of the windows might depend on the use, but in information retrieval, Kaszkiel and Zobel [KZ97] reported that window size of 50 with overlap gives the best performance. The method to automatically identify passage boundary has been a research problem and has long history. Some of the representative works are [Hea97] [MH91] [Koz93]. Choosing the passages with right granule of information is important and difficult problem. Since our target is in other place, and to minimize the effects of underlying algorithms, the documents collection with paragraph markers are chosen to simplify the problem.

Second problem is the representation of the passage. After finding the passages from documents, we must represent this excerpt of the document into machine-processable representation. The representation, in ideal, has the same information as the original, but is compact and easy to compute in later process. In domain like data mining, where the fast processing is required, relatively poor but highly efficient representation is preferred. Some of which are boolean sets taking full or part of the terms (also known as *bag-of-words* or index terms), term-weighting [Sal89] (set of terms with weights based on frequency or some kinds), or *latent semantic indexing* [DDL$^+$90]. For the simplicity, keywords (or index terms) is chosen. The problem of using keywords is that the vocabulary is limited. That is in keyword representation, passage might be represented with similar but different words, same concept can be expressed in different words in different passages. To avoid this problem, we enrich the keywords with words highly co-occurring with the keywords. This vocabulary enriching will be achieved using *tolerance rough set model.*

The last problem is the matching of passages. Passages differs the most from words in this aspect. Words are can be matched as same object by simply by comparing the symbol sequence (although it does not say that the sense is also matching), while passages cannot be said different simply because the sequence of the symbol does not match exactly. Passage of a context may be expressed using different words, in various length, and/or in different orders.

Although we could expect similar passages to share some features, but it is not too much to say that no two passages will have the exactly the same sets of words (even harder in the same order). So any two passages cannot be “same” but only “similar”, according to some similarity measure. In this paper we adopt several sets operation based measures.

To give the intuitive understanding of the idea, I start explaining from simple problems, the context-level mining with hard matching is described before the soft matching problem.

3.3 Hard Matching Scheme

We first give the formal statements for hard matching, before explaining the method based on soft matching. The following is the formal statement of the problem. This definition is based on the assumption that passages can *hard match* (*exact match*).

Definition Frequent Passage-sets Mining with Hard Matching
Let $\mathcal{P}$ be a set of all the passages[1]. Let the document collection $\mathcal{D} = \{d_1, d_2, ..., d_n\}$ be a set of documents, where each documents is a set of passages such that $d_i \subseteq \mathcal{P}$.

Given a user-defined threshold θ, a set of passages $X \subseteq \mathcal{P}$ is called *frequent* if $\mathit{support}(X) \geq \theta$, where

$$\mathit{support}(X) = \frac{|\{d_i | X \subseteq d_i\}|}{|\mathcal{D}|}$$

The results will be the sets of all the *frequent* passages $\mathcal{PS}_\theta$, i.e.

$$\mathcal{PS}_\theta = \{X | X \subseteq \mathcal{P}, \mathit{support}(X) \geq \theta\}$$

From here to end of this section, the results of hard matching with primitive settings are shown. Its potentials and problems of the methods are discussed.

3.3.1 Preliminary Results

LATFULL [2] dataset is used for this experiment. For details of other settings and preprocessings, refer to Chapter.5.

Followings are the major settings for the experiments:

- Passages are identified using SGML `<P>` tags. Headings are regarded as one passage.
- Passages are represented with top 3 keywords according to the term weighting scheme of $tf \cdot idf$ [Sal89]. Passages are denoted as $\{w_1, w_2, w_3\}$ where w_1,w_2, and w_3 are the selected keywords. The keywords in the passages are sorted in lexicographical order instead of term weights so as to remove redundancies. If the original passages has less than three words, they will be represented with 1 or 2 words.

[1] the representation of the passages are not restricted

[2] refer to Chapter.5 for details

- Passages are matched exactly. That is, two passages are said to be matched only if all 3 keywords in both passages are the same.
- No stemming applied. Stopwords are filtered.

Figure.3.1 shows the resulted rules.

```
#6526 : {juice,lemon,tablespoon} {fat,plain,yogurt} [support count:7]
#6617 : {manufacturing,semiconductor} {hiring,reducing,trend} [support
count:63]
#13731: {family,leave,sick} {handgun,possession,purchase}
{abortions,afford,woman} [support count:5]
```

Figure 3.1: Example of Rules with Hard Matching (2 year) selected manually by interest

The rules obtained are mostly from periodical articles like recipes, public statements, space for rent, etc. But we could see that the result rules catch some of the nature of the data. The discussion will be given in next section.

3.3.2 Discussions

Here we explain the rules in Figure.3.1. The rule #6526 is from articles of recipes. This rule can be interpreted as “low-fat (or fat-free) plain yogurt is used with tablespoon(s) of lemon juice.” The rule #6617 come from business trend forecasts. The antecedent has only 2 words. This is because the “semiconductor manufacturing” is the heading. The rule #13731 is from questionnaire for senators. This rule can be interpreted as “Women abortions problem are also asked in questionnaire with family sick leave, and purchasing and possession of handguns.”

Several problems hard matching becomes apparent from the results.

First, three keywords representation is poor in many senses. Passages might contain few words to several hundred words, but it is always represented in three words regardless of the original length. Since passages have variable length, fixed size keyword set might not appropriate. In later sections, some variants of selecting keywords are explained.

Another problem is that we failed to capture semantically similar but different in surface symbols. For example, (father-mother-child) and (father-mother-infant) can be seen a similar passage since “child” and “infant” are similar in sense and other terms are the same. Same kind of problem happens when there is passages like (tea-pot-price) and (coffee-pot-price). We failed to catch the possible topic (pot-price). This problems can be seen as the problem of lacking background knowledge but another reason is that in passages the expression of a contexts depends on specific term. Passages has much few words than documents and hence the variation of words used to express same meaning is also few. This problem cannot be solved simply by soft matching but also needs some way to fulfill for the lack of vocabulary problem. This problem is address again in Section.3.5.1.

Table.3.1 shows the number of generated rules depending on the support value. From this table, the system produces huge number of rules. Although this is a common problem in association rule mining, we would need to tackle this problem. This problem will be relaxed with soft matching since, soft matching will flock together the similar rules. This problem will not be discussed further until Chapter.6.

Min. Support	Number of Rules
0.01 %	321,070
0.05 %	1,094
0.10 %	634

Table 3.1: Number of rules generated on different threshold

3.4 Soft Matching Scheme

Reflecting the results of preliminary experiment, the problem had been re-defined. The support is redefined as *sim-support*, which is based on similarity. The definition of *sim-support* is defined as follows.

Definition Sim-Support
Given a *similarity function* $\mathcal{S} : \mathcal{P} \times \mathcal{P} \rightarrow \{0, 1\}$, let $\mathit{simsets}_{\mathcal{S}}(p)$ be the sets of passages similar to passage set $p \subseteq \mathcal{P}$, given in

$$\mathit{simsets}_{\mathcal{S}}(p) = \{p_j \mid \mathcal{S}(p, p_j) = 1, p_j \in \mathcal{P}\}.$$

where $\mathcal{P}$ is the set of all passages. The value of *sim-support* for passage set p is,

$$\mathit{sim\text{-}support}(p) = \left|\{d_i \mid p_k \subseteq d_i, p_k \in \mathit{simsets}_{\mathcal{S}}(p), d_i \in \mathcal{D}\}\right|$$

More informally and analogously, we can be view each documents as set of queries in information retrieval sense. The frequent itemsets are the sets of queries that returns the same documents, and the number of the overlapping of the documents is the support threshold. Figure.3.2 gives an illustrative explanation of *sim-support*.

3.5 Passage Representation

The methods of representation and matching is explained in this section.

Bag-of-words are used for representation of passages, for its simplicity. All the words in passages can be used, but for efficient computation and for compact output to users, only the words that represents words are selected. This is usually done by means of term-weightings. The weights of the terms in a certain document (w_t^d) can be defined in many ways [TOK99].

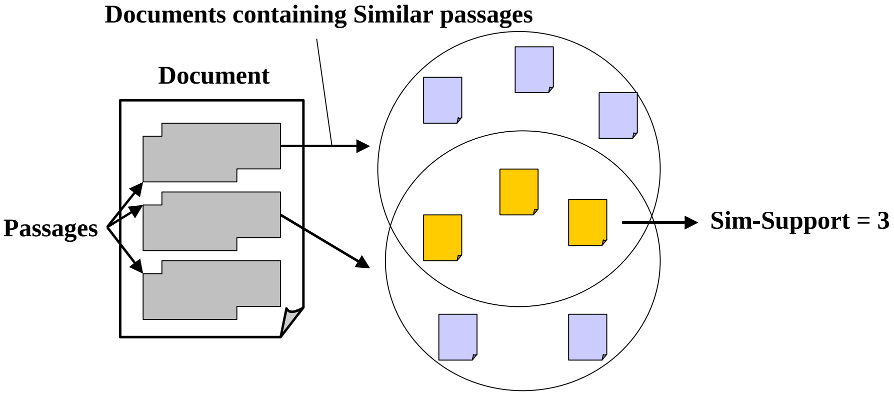


Figure 3.2: Illustrative Example *sim-support*

In this work, the most popular measure *term frequency - inversed document frequency* ($tf \cdot idf$) score is used for ranking and selecting the keywords. $tf \cdot idf$ score is given in following formula.

$$tf \cdot idf(w_i, d_j) = tf(w_i, d_j) \times log\frac{N_d}{df_{w_i}}$$

where $tf(w_i, d_j)$ is the frequency of the term w_i in document d_j, df_{w_i} is number of documents that contain w_i and N_d is the total number of document. Intuitively, the words that appear repeatedly, and the words which appear in specific documents will get high scores.

Three methods for selecting keywords are explained. For each methods, the number of keywords are adjusted so that at least three keywords are kept for those passage that has more than three words by adding the next candidate keywords in order to keep enough words to represent context.

1. Top N words.
   Select the top N ranked words, regardless of the length of the passages.

2. Top R ranked words.
   Select the top R % ranked words. The length of the passages are considered but the score is used just for ranking.

3. Top R % score.
   Select from the top ranked words until the cumulative score reaches the R % of the total score in the passage. Conciders the absolute score. Analogously, reducing the $(1 - R)$ % of energy.

In this work, the term-weights are not used after selecting the keywords. This is from three reasons. First, keywords consists of few words and term weight will not make significant differences. Second, we are comparing passages based on extended distribution hypothesis, which will not concider term weights. Third for the computational efficiency.

3.5.1 Tolerance Rough Set Model(TRSM)

In Section.3.3, we addressed the problem of lack of term variation when we use keywords to represent passages. To fulfill for the lack of vocabulary, we will use the *Tolerance Rough Set Model* (TRSM).

TRSM is one extension of original (equivalent) rough set model, a mathematical tool to deal with vagueness and uncertainty [Paw91]. Whilst the original rough set model is built on equivalence relation (reflexive, symmetric, and transitive), TRSM is based on the tolerance relations (reflexive and symmetric) [SS94].

The similar approach is to use a thesaurus. Actually the query expansion with thesaurus is well studied area in information retrieval [TOK99]. Using thesaurus is promising in that quality of the similar words in human built thesaurus is reliable and it also often gives the semantical information like hypernyms or hyponyms, but it also has drawbacks. First of all, thesaurus is not always available (especially for some special domains). Another drawback is that thesaurus usually does not contain wanted information like proper names or non-taxonomical relation like “pen and paper”. This problem is critical since many keywords often proper names. Finally in order to use thesaurus with right sense, we might first need to identify the sense the word is in use. We might require a words sense disambiguation task before expanding vocabulary with thesaurus.

TRSM does not have such problems, but generation of tolerance class (the sets of similar words) is based on the frequency, and therefore the similar words of low frequency will not be generated. In future, we might combine these two for improvement of this process.

3.5.2 Basic Notions of Rough Sets

The basic idea of rough sets is to approximate a set X in universe U by upper and lower approximations in an equivalence space $\mathcal{R} = (U, R)$ where $R \subseteq U \times U$ is an equivalence relation. Any two objects $x, y \in U$ are said to be *indiscernible* regarding R if xRy. The lower and upper approximations of $\mathcal{R}$ of any subset $X \subseteq U$ is

$$\mathcal{L}(\mathcal{R}, X) = \{x \in U \mid [x]_R \subseteq X\} \tag{3.1}$$

$$\mathcal{U}(\mathcal{R}, X) = \{x \in U \mid [x]_R \neq \emptyset\} \tag{3.2}$$

respectively, where $[x]_R$ is a class of objects indiscernible with x regarding the equivalence relation R. Figure.3.3 shows the illustrative example.

3.5.3 Tolerance Relation and Tolerance Spaces

Tolerance rough set model (TRSM) is a extension of rough sets that admits the overlapping classes based on tolerance relations [SS94].

Tolerance space is defined as 4 tuples: $\mathcal{R} = (U, I, \nu, P)$, where U is a universe of objects, $I : U \rightarrow 2^U$ is called an uncertainty function that identifies all the objects in tolerance class of an object that is similar. Uncertainty function can be any function that satisfies the condition $x \in I(x)$ and $y \in I(x)$ iff $x \in I(y)$ (reflexive and symmetric property).

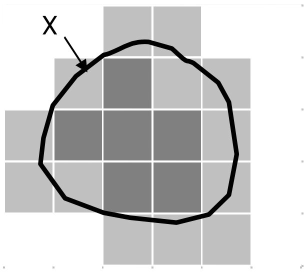


Figure 3.3: Illustrative Example of Upper and Lower Approximation of X. Dark gray is the lower approximation, and the light gray is the upper approximation.

$\nu : 2^U \times 2^U \longrightarrow [0,1]$ is called vague inclusion, which is the degree of inclusion of sets, used to determine whether tolerance class $I(x)$ is included in $X \subseteq U$. The only requirement of ν is that it is *monotonic* with respect to the second argument of nu, i.e. $\nu(X,Y) \leq \nu(X,Z)$ for any $X, Y, Z \in U$ and $Y \subseteq Z$. $P : I(x) \rightarrow 0, 1$ is called the structurality function, which determines that $I(x)$ is a structural subset or not. The lower and upper approximation is

$$\mathcal{L}(\mathcal{R}, X) = \{x \in U \mid P(I(x)) = 1 \wedge \nu(I(x), X) = 1\} \tag{3.3}$$
$$\mathcal{U}(\mathcal{R}, X) = \{x \in U \mid P(I(x)) = 1 \wedge \nu(I(x), X) > 0\} \tag{3.4}$$

respectively.

TRSM is observed to be more suitable for domain related to natural languages in that words' meaning is overlapping original can not express this overlapping. The Figure.3.4 is the figure adopted from [HN02], which shows part of meanings of the three words, *root*, *cause*, and *basis* in roget's thesaurus. From the figure, we could see the overlapping meanings of the words.

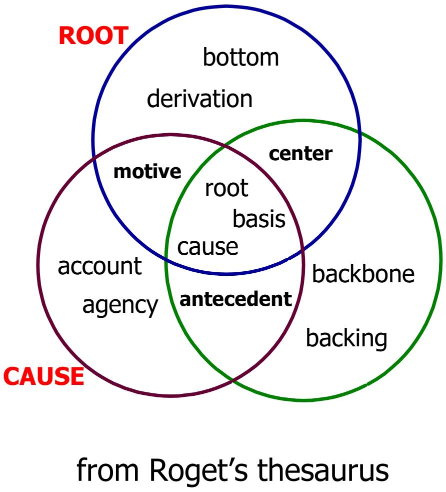


Figure 3.4: Overlapping Meanings of the Words

3.5.4 Tolerance Spaces for Documents

Now the explanation of TRSM in documents [BF98] is described as follows.

Let $\mathcal{T} = \{t_1, t_2, \ldots, t_N\}$ be a set of N distinct terms. Let $\mathcal{D} = \{d_1, d_2, \ldots, d_M\}$ be a set of M documents, where each document d_j is a set of terms such that $d_j \subseteq \mathcal{T}$. Each term t_i has a weight of importance $w_{ij} \in [0,1]$ [3] in d_j. $f_{d_j}(t_i)$ is the number of occurrences of term t_i in d_j, and $f_{\mathcal{D}}(t_i)$ is the number of documents in $\mathcal{D}$ which term t_i occurs. The weights w_{ij} will be calculated in the following equation.

$$w_{ij} = \begin{cases} (1 + \log(f_{d_j}(t_i))) \times \log \dfrac{M}{f_{\mathcal{D}}(t_i)} & \text{if } \ t_i \in d_j, \\ 0 & \text{if } \ t_i \notin d_j \end{cases} \tag{3.5}$$

The weights will be normalized by the size of the vector:

$$w_{ij} \leftarrow \frac{w_{ij}}{\sqrt{\sum_{t_{hj} \in d_j} (w_{hj})^2}}.$$

Denote by $f_{\mathcal{D}}(t_i, t_j)$ the number of documents in $\mathcal{D}$ in which two terms t_i and t_j co-occur. Given a threshold θ, the uncertainty function I is define as

$$I_\theta(t_i) = \{t_j \mid f_{\mathcal{D}}(t_i, t_j) \geq \theta\} \cup \{t_i\}. \tag{3.6}$$

It is clear that the function Equation.3.6 satisfies the reflexive and symmetric conditions. The vague inclusion function ν is defined

$$\nu(X, Y) \ = \ \frac{|X \cap Y|}{|X|} \tag{3.7}$$

This function is clearly monotonous with respect to the second argument. Using this function the membership function μ for $t_i \in \mathcal{T}, X \subseteq \mathcal{T}$ can be defined as

$$\mu(t_i, X) = \nu(I_\theta(t_i), X) \ = \ \frac{|I_\theta(t_i) \cap X|}{|I_\theta(t_i)|} \tag{3.8}$$

The term-weighting method defined by Equation.3.5 is extended to define weights for terms in the upper approximation $\mathcal{U}(\mathcal{R}, d_j)$ of d_j. It ensures that each term in the upper approximation of d_j but not in d_j has a weight smaller than the weight of any term in d_j.

$$w_{ij} = \begin{cases} (1 + \log(f_{d_j}(t_i))) \times \log \dfrac{M}{f_{\mathcal{D}}(t_i)} & t_i \in d_j, \\ \min_{t_{hj} \in d_j} w_{hj} \times \dfrac{\log(M/f_{\mathcal{D}}(t_i))}{1 + \log(M/f_{\mathcal{D}}(t_i))} & t_i \in \mathcal{U}(\mathcal{R}, d_j) \setminus d_j \\ 0 & t_i \notin \mathcal{U}(\mathcal{R}, d_j) \end{cases} \tag{3.9}$$

[3] only $w_{ij} \in \{0, 1\}$ for boolean model

With these definitions we can define a tolerance space as $\mathcal{R} = (\mathcal{T}, I, \nu, P)$ in which the *lower approximation* $\mathcal{L}(\mathcal{R}, X)$ and the *upper approximation* $\mathcal{U}(\mathcal{R}, X)$ in $\mathcal{R}$ of any subset $X \subseteq \mathcal{T}$ can be defined as

$$\mathcal{L}(\mathcal{R}, X) = \{t_i \in \mathcal{T} \mid \nu(I_\theta(t_i), X) = 1\} \tag{3.10}$$

$$\mathcal{U}(\mathcal{R}, X) = \{t_i \in \mathcal{T} \mid \nu(I_\theta(t_i), X) > 0\} \tag{3.11}$$

The vector length normalization is then applied to the upper approximation $\mathcal{U}(\mathcal{R}, d_j)$ of d_j. Note that the normalization is done when considering a given set of index terms.

3.6 Passage Matching

After representing the passages in either form, the passages is matched according to their similarity. Passage pair in which the similarity greater is than the user-defined threshold ϕ is regarded as matched.

There are several well known measures for calculation similarity between sets. Four of those measures are explained[Sal89]: inner product, Dice coefficient, cosine coefficient, and Jaccard coefficient.

Suppose we want to calculate the similarity of $X = \{x_1, x_2, \ldots, x_t\}$ and $Y = \{y_1, y_2, \ldots, y_t\}$ for boolean vector $sim(X, Y)$ and for real value vector $simw(X, Y)$ is defined as follows

- Inner product

$$sim(X,Y) = \big|X \cap Y\big|, \quad simw(X,Y) = \sum_{i=1}^{i} x_i \cdot y_i$$

- Dice coefficient

$$sim(X,Y) = \frac{2\big|X \cap Y\big|}{\big|X\big| + \big|Y\big|}, \quad simw(X,Y) = \frac{2\sum_{i=1}^{t} x_i \cdot y_i}{\sum_{i=1}^{t} x_i^2 + \sum_{i=1}^{t} y_i^2}$$

- Cosine coefficient

$$sim(X,Y) = \frac{\big|X \cap Y\big|}{\sqrt{\big|X\big|} \cdot \sqrt{\big|Y\big|}}, \quad simw(X,Y) = \frac{\sum_{i=1}^{t} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{i} x_i^2 \cdot \sum_{i=1}^{t} y_i^2}}$$

- Jaccard coefficient

$$sim(X,Y) = \frac{\big|X \cap Y\big|}{\big|X\big| + \big|Y\big| - \big|X \cap Y\big|}, \quad simw(X,Y) = \frac{\sum_{i=1}^{t} x_i \cdot y_i}{\sum_{i=1}^{t} x_i^2 + \sum_{i=1}^{t} y_i^2 - \sum_{i=1}^{t} x_i y_i}$$

We do not consider the weight of the keywords at this stage. Among these four measures, cosine coefficient, and two sets operations, *inclusion* and *subsumption* operation are adopted. Definition of *inclusion*, *subsumption*, and *overlap* is defined in next formulas

- Inclusion

$$sim(X,Y) = \frac{|X \cap Y|}{|X|} \tag{3.12}$$

- Subsumption

$$sim(X,Y) = \frac{|X \cap Y|}{|Y|} \tag{3.13}$$

- Overlap

$$sim(X,Y) = \frac{|X \cap Y|}{|min(X,Y)|} \tag{3.14}$$

Cosine coefficient is the angle between two vectors, which is widely used in information retrieval to calculate the distance between documents.

Inclusion and subsumption are the measures of how much one sets is subsumed by an another. Both measures are not symmetric. These measures are used when determining whether one set is subsumed by upper approximation tolerance class of another, instead of approximating both X and Y. We calculate this way in order to disregard the size of upper approximation. Overlap is similar measure but does not care the direction (symmetric). We prefer to use these sets based measures since we are only interested in matching keywords of passages, which is usually small sets of words. Another reason to prefer those measure is that that way we are strictly following the extended definition of distribution hypothesis, “two passages share certain amount of words if they refer to similar context.”

In later section, details of building similarity matrix between passages are addressed. For assymetric measure like inclusion all pairs must be calculated, while for symmetric measure only requires the half of the matrix calculation. This will be addressed again in later sections.

Chapter 4

Implementations

In this chapter the detail of the algorithms and the implementation is described based the approach and method given in previous chapter. The details of datasets appear in this chapter, can be found in Chapter.5.

4.1 System Overview

The overview of our system is as shown in Figure.4.1. Input is a collection of documents. Documents are broke down to passages. And then convert passage into transactional format. After then we apply association rule mining. It usually generates rule set of huge number. So in the final step system evaluates the rule to select or rank the most interesting rules. The output will be the selected or ranked rule set. At the current stage, the last step is not yet implemented.

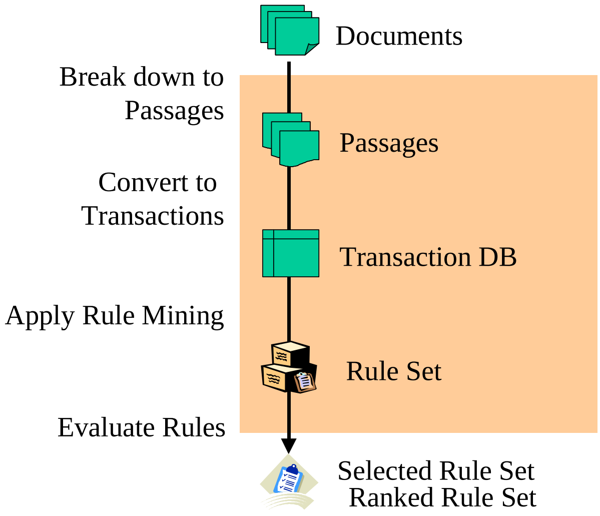


Figure 4.1: Overview of the System

The process is briefly described as follows.

1. Preprocessing
2. Keyword Selection
3. Vocabulary Enriching
4. Similarity Matrix Calculation
5. Association rules mining

Each steps will be explained in each section.

4.2 Preprocessing

There are four preprocessings: passage identification, sentence detection, stopwords, and stemming. All the programs are in this section was written in Perl.

Passage Identification

As mentioned before, at the current state, the datasets with paragraph markers are used. For dataset using La Times, `<P>` or `<DOC>` is used.

Sentence Detection

Sentence identification is process to detect the correct sentence boundary, carefully avoid the abbreviation periods. The modified version of *heuristic sentence boundary detection algorithm* in [MS99] is used. The basic algorithm is the same do additional processing like remove periods from abbreviations, etc. The code in Perl is in Appendix.B.

Stopwords Filtering

Stopwords filtering is a process to remove a functional words like "the" and "when", using a predefined list of stopwords. SMART system stopwords (571 words) [Cor99] are used in the experiments.

Stemming

Stemming is to reduce the transformational variants of words like past tense verb (-ed) or plural nouns (-s) to original forms. There are popular stemmers like Porter's stemmer [Por80] or Paice [Pai90] but on subjective observation that these stemmers cuts too much to the extent that might cause interpretation difficulty. These stemmers could reduce the adverb (-ly) or noun (-ion) to verbs but these might change the meaning of the words. So, inaddition to those standard stemmers, a simple stemmer has been developed to remove just -*ed*s and -*s*es. The Perl code is provided in Appendix.A.

4.3 Tolerance Class Generation

In this section, we show several variations for uncertainty function to acquire co-occurrences. We propose three measures in addition to the original frequency based measure: χ^2 test, pointwise mutual information, and weight pointwise mutual information [MS99]. Although the original co-occurrence measure is simple and fast, we seek for more accurate class generation, while the co-occurrence have the problem of taking co-occurrence that happen by chance. Comparison will be shown in Chapter.5

4.3.1 Frequency

The original original co-occurrence measure based uncertainty function was

$$I_\theta(t_i) = \{t_j \mid f_{\mathcal{D}}(t_i, t_j) \geq \theta\} \cup \{t_i\}.$$

where $f_{\mathcal{D}}(t_i, t_j)$ is the frequency of both t_i and t_j in documents (in our work, passages). The problem of frequency based approach is that

- It has the bias of selecting high frequent words. High co-occurrence might be reached only because both two words occur common in collections
- Selection of θ depends on the data and must be determined by some means.

To overcome these difficulties, we suggest statistical and information theoretic measures.

4.3.2 χ^2 test

The basic idea is to test the word pairs' independence (null hypothesis) with χ^2 test. The pairs are regarded as co-occurring when the null hypothesis is rejected with critical value.

The test is accomplished by 2 × 2 homogeneity test. It tests against the null hypothesis that two words are independent. If the $\chi \geq \chi_\alpha$ the null hypothesis of independence is rejected . The χ^2 statistics are given in following formula

$$\chi^2 = \sum_{\text{for each rols and cols}} \frac{(O - E)^2}{E}$$

O is the observed value, and E is the expected value which is the product of row sum and col sum, divided by the total sum.

$$E_{i,j} = (\text{marginal probability of row i}) \times (\text{marginal probability of col j}) \times (\text{total sum})$$

Figure.4.2 shows the contingency table of "corp" and "price" against "company".

The χ^2 value of "company" and "corp" has much greater value than critical value of $\alpha = 0.001$, so the "company" and "corp" pair are regarded as co-occurring pair, while "company" and "price" pair's χ^2 value is less than critical value and cannot be rejected, so the these pair will not be added in tolerance class. Note that "price" and "company" has

	corp	¬ corp	
company	785	9764	10549
¬ company	5624	109984	115608
	6489	119708	126157

$(\chi^2 = 133.11 \geq \chi^2_{0.001})$

	price	¬ price	
company	575	9974	10549
¬ company	5706	109902	115608
	6281	119878	126157

$(\chi^2 = 5.42 \leq \chi^2_{0.001})$

Figure 4.2: Contingency table of "corp" and "price" against "company"

high frequency value of 575 times, which is usually taken up as a frequent pair in original framework. It is known that χ^2 test will not work when E is too small (heuristically less than 5), so those cases are rejected in advance.

The advantages of χ^2 method is that it has statistical significance, and the threshold value need not determined datasets by datasets. χ^2 test is symmetric and satisfies the condition of uncertainty function.

4.3.3 Pointwise Mutual Information (PMI)

Pointwise mutual information is defined as

$$\mathcal{I} = log_2 \frac{p(x,y)}{p(x)p(y)}.$$

this is clearly symmetric. The idea is based on information theory. PMI indicates how much of uncertainty of seeing y will reduce if told that x is in the same document (or passages).

Pointwise mutual information has the bias towards giving high score to small occurrences. Explained in next formula for the perfect dependence, [MS99]

$$I(x,y) = log_2 \frac{p(xy)}{p(x)p(y)} = log_2 \frac{p(x)}{p(x)p(y)} = log_2 \frac{1}{p(y)}.$$

Informally, if the word appears in only one document in a corpus, knowing that word would gives all the information about the words in same document. So the mutual information, does not be suitable for sparse data like co-occurrence or n-grams and often the weighted version is used.

4.3.4 Weighted Pointwise Mutual Information (WPMI)

Weighted pointwise mutual information is the point wise mutual information value weighted by the frequency $(f_D(t_i,t_j)I(t_i,t_j))$. In this paper the following formula is used.

$$WI = log f_D(t_i,t_j) log_2 I(t_i,t_j).$$

We took the natural logarithm for frequency so that frequency will not dominates the pointwise mutual information. It relaxes the bias of pointwise mutual information the nature is not clear. This formula still holds the symmetric property.

Comparison of the methods will be shown in Chapter.5.

4.4 Building Similarity Matrix

Before the mining (or as a first iteration of mining), a similarity matrix between passages is built. In implementation, inverted indexing [FBY92] is used for efficiency. Basic idea of inverted indexing is to prepare a term to document (in this work, passage) file and retrieve the documents only related to current comparison, instead of comparing with all the documents. Figure.4.3 shows the illustrative example. The pseudo code for building similarity matrix with inverted indexing is described in Algorithm.1.

Passage ID	Term IDs
p_1	t_1, t_2
p_2	t_2, t_3, t_4
p_3	t_1, t_2, t_4
p_4	t_3, t_4

$\Rightarrow$

Term ID	Passage IDs
t_1	p_1, p_3
t_2	p_1, p_2, p_3
t_3	p_2, p_4
t_4	p_2, p_3, p_4

Figure 4.3: Inversion of Document(Passage)-Term Indexing to Term-Document Indexing

4.5 Mining Algorithm

In this section the mining algorithm is explained. First the related work [Nah04] is explained, and then our approach.

4.5.1 SoftApriori [Nah04]

This algorithm is based on APRIORI [AS94]. Although the data to be mined has different characteristics than ours, the framework of supporting soft matching is quite similar to ours.

In APRIORI , the process consists of two major parts: the item counting and the candidate generation. In the item counting step it counts the frequency of the candidates as they scan the database. In Nahm's framework [Nah04], if x *is similar to* y ($x \sim y$), the support of y in the database, will be added to x. Although they uses hash functions for fast lookup of similar items, they still has to load the similarity matrix during the process. In their case, there exists items that exactly the same, but in our framework it rarely happen.

Algorithm 1 Similarity Matrix Calculation

```
Function: $term2pidset(t)$
    Lookup term $t$ in term-passage index and return the set of passage IDs
Function: $pid2passage(pid)$
    Lookup passage ID $pid$ in passage-term index and return the passage (a set or a vector
    of terms) set(or vector)
Function: $sim(p1, p2)$
    Takes two passage $p1$ and $p2$ returns 1 if passages is similar, 0 otherwise
Procedure: $BuildMatrix(\mathcal{D}, M)$,
$\mathcal{D}$: the database,
$M$: an $N \times N$ matrix where $N$ is the total number of passages
    Set all elements in $M$ to 0
    for all passage $p_i \in \mathcal{D}$ do
        $pidset \leftarrow \emptyset$
        for all term $t \in p_i$ do
            $pidset_t \leftarrow term2pidset(t)$
            $pidset \leftarrow pidset \cup pidset_t$
        end for
        for all $pid \in pidset$ do
            passage $p_j \leftarrow pid2passage(pid)$
            if $sim(p_i, p_j) = 1$ then
                $M_{i,j} \leftarrow 1$
            end if
        end for
    end for
```

Considering this point, we propose a soft mining matching framework which is based on vertical mining. Unlike SOFTAPRIORI it does require the similarity matrix to be loaded during the mining process, by creating the small inverted files from similarity matrix,

4.5.2 Vertical Mining

Vertical mining make use of the inverted file format. The inverted file format is the same idea as described in Figure.4.3. Once finished creating the inverted file, the counting of the items is done by set intersections of transactions IDs.

Vertical mining have benefits over APRIORI like algorithms.

1. No candidate generation. In APRIORI , the candidate generation of length l implies the $2^l - 2$ candidates to be generated.

2. Less I/O overhead. Vertical mining requires only one scan of databases.

Additionally for soft matching scheme,

3 Similarity matrix is required only in inversion of database, there is no need to load similarity matrix in mining steps. In APRIORI like approach, the similarity matrix must be loaded on memory until the end.

There are also drawbacks. Vertical mining is a memory over I/O methods, that is, it processes on memory to reduce to I/O. Vertical mining in essence the recursive process of set intersections, and therefore it is a depth first search.

4.5.3 Implementation Detail

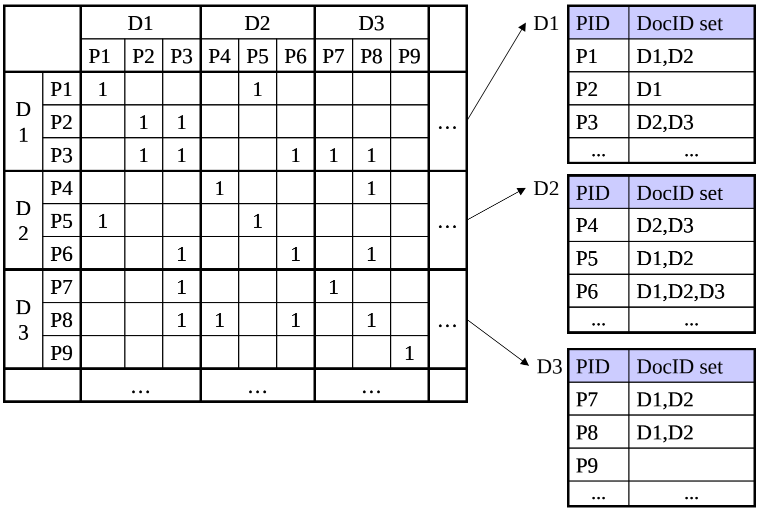
The implementation of the inversion program and the mining algorithm CHARM [ZH02] is done in C++. CHARM is the vertical mining based closed itemset mining algorithm. The implementation of CHARM is paralleled using MPI (Message Passing Interface) [Pac97], processing the small inverted files by dispatching them to other nodes, using producer and consumer protocol.

From similarity matrix, N inverted files are generated, where N is the total number of documents. Figure.4.3 shows the illustration. Algorithm.2 is the steps of converting similarity matrix into inverted files. Inverted files are separated document by document to make sure that it does not produce association between the similar items and similar items of the other. Similar passages is seen as a pseudo item that exists in other documents and used for support counting but it does not mean that passage really exists in those documents.

Algorithm 2 Similarity Matrix to Inverted Files

Procedure: $CreateInvertedFiles(M, P2D)$,
M: an $N \times N$ matrix where N is the total number of passages
$P2D$: passage id to document id mapping file

```
for all document $d_i \in \mathcal{D}$ do
    for all passage $p_j \in d_i$ do
        read similar passage id set $simset(p_j)$ of $p_j$ from matrix $M$
        write passage id of $p_j$
        for all passage $p_k \in simset(p_j)$ do
            write document id of $p_k$ using $P2D$
        end for
    end for
end for
```



		D1			D2			D3			
		P1	P2	P3	P4	P5	P6	P7	P8	P9	
D 1	P1	1				1					…
	P2		1	1							
	P3		1	1			1	1	1		
D 2	P4				1				1		…
	P5	1				1					
	P6			1			1		1		
D 3	P7			1				1			…
	P8			1	1		1		1		
	P9									1	
		…			…			…			

D1

PID	DocID set
P1	D1,D2
P2	D1
P3	D2,D3
...	...

D2

PID	DocID set
P4	D2,D3
P5	D1,D2
P6	D1,D2,D3
...	...

D3

PID	DocID set
P7	D1,D2
P8	D1,D2
P9	
...	...

Figure 4.4: Similarity Matrix to Small Inverted Files

Chapter 5

Experiments

In this chapter, we describe the experiment and show the results of the evaluation.

5.1 Datasets and Environment

The datasets are prepared from news articles of LA Times (TREC5) collections. The name in bold are the code name of the dataset referred elsewhere.

LATFULL News articles of LA Times from TREC5 (2 years)

LATM News articles of LA Times from TREC5 (1 month, Jan '89)

Table.5.1 shows the characteristics of the datasets.

Dataset	Number of Documents	Number of Passages	Number of Words
LATFULL	131,896	1,999,123	31,289.375
LATM	5024	75,413	1,174,194

Table 5.1: Characteristics of the datasets

LATFULL is only used in Section.3.3.1. In most experiments, LATM is used due to the computation time. The time complexity of creating similarity matrix is $O(N^2)$, where N is the number of passages in the documents collection.

Passages (paragraphs) in these documents are identified by SGML (Standard Generalized Markup Language) tags.

For the experiments, high spec PCs are used. In most experiments are run on Intel Xeon Dual 2.4GHz, 2GB memory, with RedHat Linux 7.3. Some parts of experiments are run on Pentium 4 2.2GHz, 1GB memory, with RedHat Linux 7.3.

5.2 Evaluation of Keywords Selections

In this section, the keywords selection methods is compared.

In previous section, three methods was introduced: top N words (wrN), top R % ranked words (wrR), and top R % score (srR). wr$N$ is used in preliminary experiments only (in Section.3.3.1). wr$N$ and sr$N$ are compared. Table.5.2 shows the statistics of wr and sr respectively, about the number of words (tokens) that appear in passages,

Input	Tokens	Mean	Variance	Median	Mode
original	1285822	17.76	147.30	16.00	3
wr80	834919	11.53	61.15	11.00	12
wr60	626287	8.65	33.09	8.00	3
wr40	424463	5.86	13.12	5.00	3
wr20	249246	3.44	1.96	3.00	3
sr80	734704	10.15	39.53	9.00	4
sr60	500289	6.91	14.89	6.00	4
sr40	338588	4.68	3.68	4.00	4
sr20	274495	3.79	0.58	4.00	4

Table 5.2: Statistics on number of tokens in passages for different threshold

The differences are not apparent from the statistics, but distribution reveals some differences. Figure.5.2 shows the frequencies (Y-axis) of the passages against the number of tokens (X-axis) they contain.

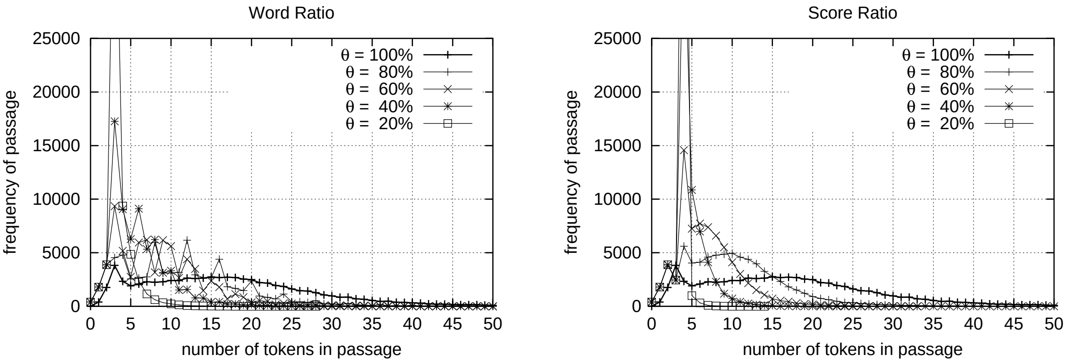


Figure 5.1: Frequency of passages against the number of tokens they contain
Word Ratio (left), Score Ratio (right) with different thresholds

The distributions of srR are reducing the number of words accordingly to the original distribution, (similar to the binominal distribution), while wrR makes the distribution quite unstable. In later experiments, we prefer to use $sr40$ for it keeps the original distribution.

5.2.1 Comparison on Different Measures for Tolerance Class Generation

In this section we compare the tolerance class generated from different methods. We first give the statistics. Table.5.3 gives the statistics.

Input	Classes	Mean	Variance	Median	Mode
co30	1374	17.14	1561.18	4.00	2
co50	745	10.72	479.88	3.00	2
co100	267	5.65	72.02	3.00	2
co150	136	3.94	20.29	2.00	2
co200	78	3.31	7.42	2.00	2
co250	52	3.00	3.62	2.00	2
co300	36	2.83	2.03	2.00	2
chi50	965	9.41	207.76	4.00	2
chi100	711	6.40	75.37	3.00	2
chi150	568	5.39	45.24	3.00	2
chi200	480	4.79	30.89	3.00	2
chi250	408	4.36	22.00	3.00	2
chi300	368	4.06	17.51	2.00	2
pmi3	26108	37.26	3682.14	13.00	2
pmi5	25413	15.49	313.14	9.00	2
pmi7	22155	7.29	40.16	5.00	2
pmi10	11957	3.38	4.33	3.00	2
pmi13	2501	2.43	0.76	2.00	2
pmi15	357	2.19	0.36	2.00	2
wpmi5	22774	16.14	731.16	7.00	2
wpmi7	18221	9.01	169.11	5.00	2
wpmi10	11162	5.44	34.59	3.00	2
wpmi15	5204	3.15	3.88	2.00	2
wpmi20	1936	2.43	0.89	2.00	2

Table 5.3: Comparison of Frequency, χ^2 test, PMI, and WPMI with different thresholds (co is the frequency)

First we see the statistics of number of class generated. From the statistics it is apparent that information theoretic methods generates the classes with large number of words. WPMI relaxes PMI but still has large number of words in a class. Comparing frequency and chi-square test, number of words decreases sharply with higher threshold for frequency, while that of chi-square test will not drop, even with higher threshold.

Figure.5.2.1 shows the frequency distribution (Y-axis) of the words contained (X-axis) in passage of the class generated with frequency, χ^2, PMI, WPMI, respectively. We could see how much words are in each classes. From the figures, we could observe that the

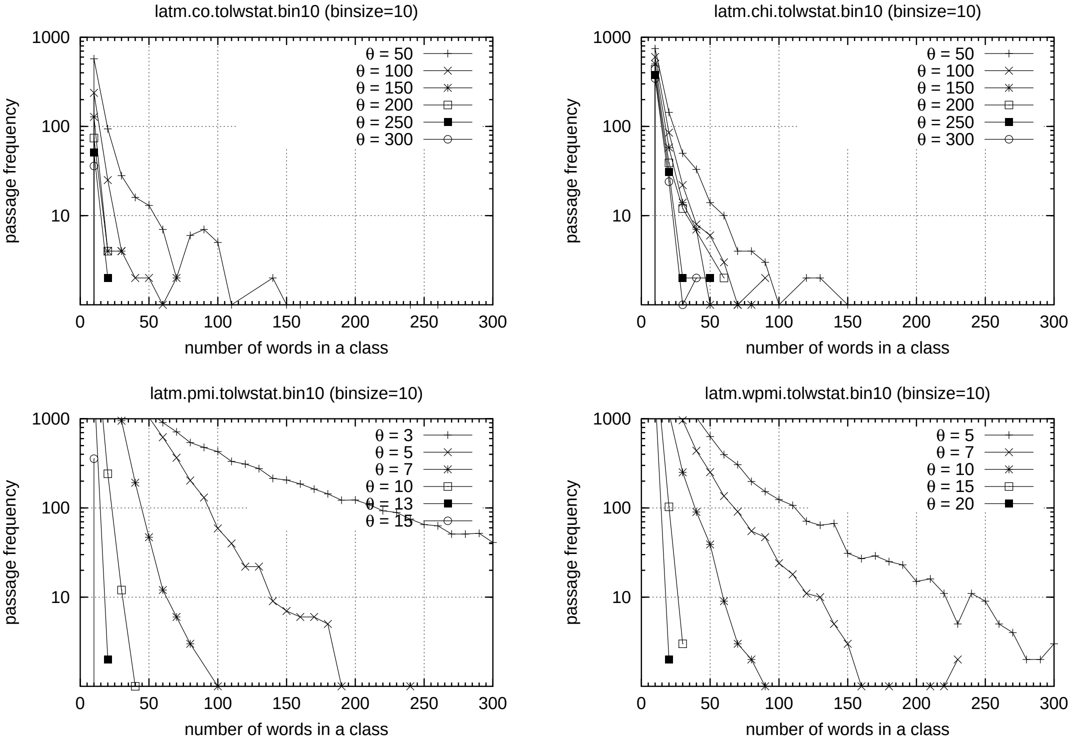


Figure 5.2: Comparison of different Tolerance Class Generation Methods
Number of words in each class (X-axis) against the passage frequency (Y-axis). Frequency (top left), χ^2 test (top right), PMI (bottom left), and WPMI(bottom right). Range of X is limited from 0 to 300, and from 0 to 1000 for Y.

frequency and the chi-square tests has similar distributions of number of the words in the class. PMI and WPMI generates class not only with large number of words but also with small number of words.

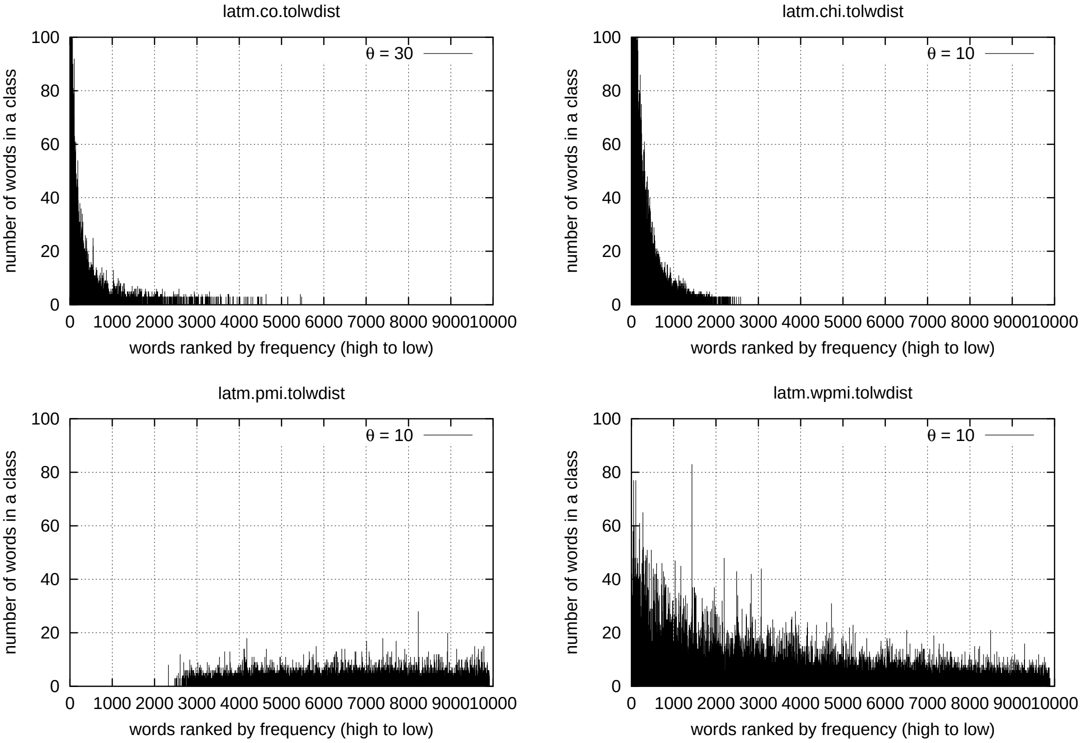


Figure 5.3: Comparison of Distribution of Tolerance class
The words in descending order of frequency (X-axis), number of words in tolerance class (Y-axis). Frequency (top left), χ^2 test (top right), PMI (bottom left), and WPMI(bottom right) Range of X is limited from 0 to 10000, and from 0 to 100 for Y.

Next we see the relation between the frequency of single word and the number of words in tolerance class to see the frequency bias. Figure.5.2.1 shows the distribution of number of words in tolerance class, against the words in descending order of frequency. Since we only want to know the distribution, the thresholds for each method are chosen arbitrary. From the figures, we could see that frequecy and the chi-square has bias towards high frequency words, but this is natural in that to become highly co-occurring pairs, each pair must be frequent. On the other hand, PMI has the bias to choose low frequency words and WPMI is somewhat biased to high frequency words but generates certain amount of words for both high and low frequent words.

Figure.5.4 shows an example of the class generated by frequency ($\theta = 200$), chi-square test ($\theta = 250$), and WPMI ($\theta = 10$). Same set of words with high frequencies are chosen. We could see that frequency based methods generated a class "time" with the word "people" and "game" which is co-occurring but is more by chance. "people" and "game" are also the high frequent words. On the other hand chi-square test does not

Classes from Frequency with $\theta = 200$

time: time state game people day play work make call back long

state: state time point game county san official california department united cal

game: game time state point play team high season coach player scored lead league conference basketball

play: play time point game team league

Classes from Chi-square test with $\theta = 250$

time: time spend

state: state official california department law united cal governor fullerton secretary deukmejian mexico florida northridge gov arizona utah fresno

game: game point play team high season coach player scored lead half night shot league minute conference won goal quarter win basketball led victory lost bowl guard average rebound playing miss forward winning cal athletic assist losing straight championship scoring halftime consecutive averaging streak touchdown loyola scorer

play: play point game team season coach player league minute conference basketball playing role tonight

Classes from WPMI with $\theta = 10$

time: time spend

state: state california law united cal governor fullerton secretary deukmejian mexico florida legislature northridge gov arizona utah fresno ohio oregon retarded equalization

game: game point play team season coach player scored lead shot league conference won goal quarter win basketball victory lost bowl guard average rebound playing forward winning cal athletic assist losing straight championship scoring halftime laker consecutive averaging streak touchdown loyola bruin scorer overtime dominguez rebounding nonconference nicholl unbeaten nonleague edmonton rebounder marmonte

play: play game team league basketball role tonight playwright fugard benson

Figure 5.4: Example of Tolerance Class of High Frequency words

put “people” or “game” in same class as “time.” The difference between chi-square test and WPMI is subtle in this comparison. The differences of these methods appear in low frequency words. For example, while chi-square produces tolerance class of “ohio” with {*ohio state*} only, WPMI produces {*ohio state cleveland michigan louisville ly burson columbus buckeye*}. The words in WPMI generated class are name of the states next to each other, or state symbols, or name of the city in ohio states. More thorough comparison awaits for future work, but it is showing that chi-square test or WPMI are effective in generating tolerance class from these qualitative view.

Further analysis is required, but for the time being we adopt the chi-square and WPMI is used for further experiment, for it has the statistical backgrounds and/or the generated class seems to be reasonable.

5.3 Comparison of Measures for Matching

In this section, we compare the measures for passage matchings.

First we compare the six matching measures: cosine, subsumption, inclusion, overlap, Dice, and Jaccard. The number of similar passages are compared to which measure matches the most and to see the distribution of number of passages. We made two setups. One is when passages contain many words (sr80), and another containing small number of words (wr40). Figure.5.3 shows the results of the former and Figure.5.3 shows the results of the latter. At this comparison no vocabulary enriching is applied.

Looking at Figure.5.3, we could see that inclusion and overlap tend to match with large number of passages. Dice and Jaccard matches with rather small number of passages. Cosine and subsumption has similar distribution, although they are different in that the former is symmetric and the latter is asymmetric. We could observe the similar situation in Figure.5.3 where words in passages are small. The shape of distribution will not change but inclusion and overlap reduce the number of matches largely. We could see that these two measure have the bias to match larger when original words have large number. For cosine and subsumption, the number of similar passages decrease to some extent but not as significant as inclusion and overlap. The most robust measures against number of words in passages are the Dice and Jaccard.

Next we compare the effect of vocabulary enriching. Effect of enriching is measured with two passage matching measures: cosine and subsumption. These two are selected because they have similar distribution with different characteristics, symmetric and asymmetric. For cosine measure, both of the passages to be matched is enriched, while only the first argument is enriched for the subsumption measure. The matching threshold is 50% for both measures. Method of enrichment is on the chi-square and the WPMI, thresholds for those are 150 and 15, respectively. Each comparison is tested on sr80 (large number of words) and sr40 (small number of words). Figure.5.3 shows the results.

First of all we can observe that vocabulary enriching is effective in increasing the number of similar passages regardless of number of words in passages. For enriching methods, although WPMI has larget number of words in tolerance class, chi-square test excels the WPMI in increasing the number of passages matched. It might be indicating that WPMI

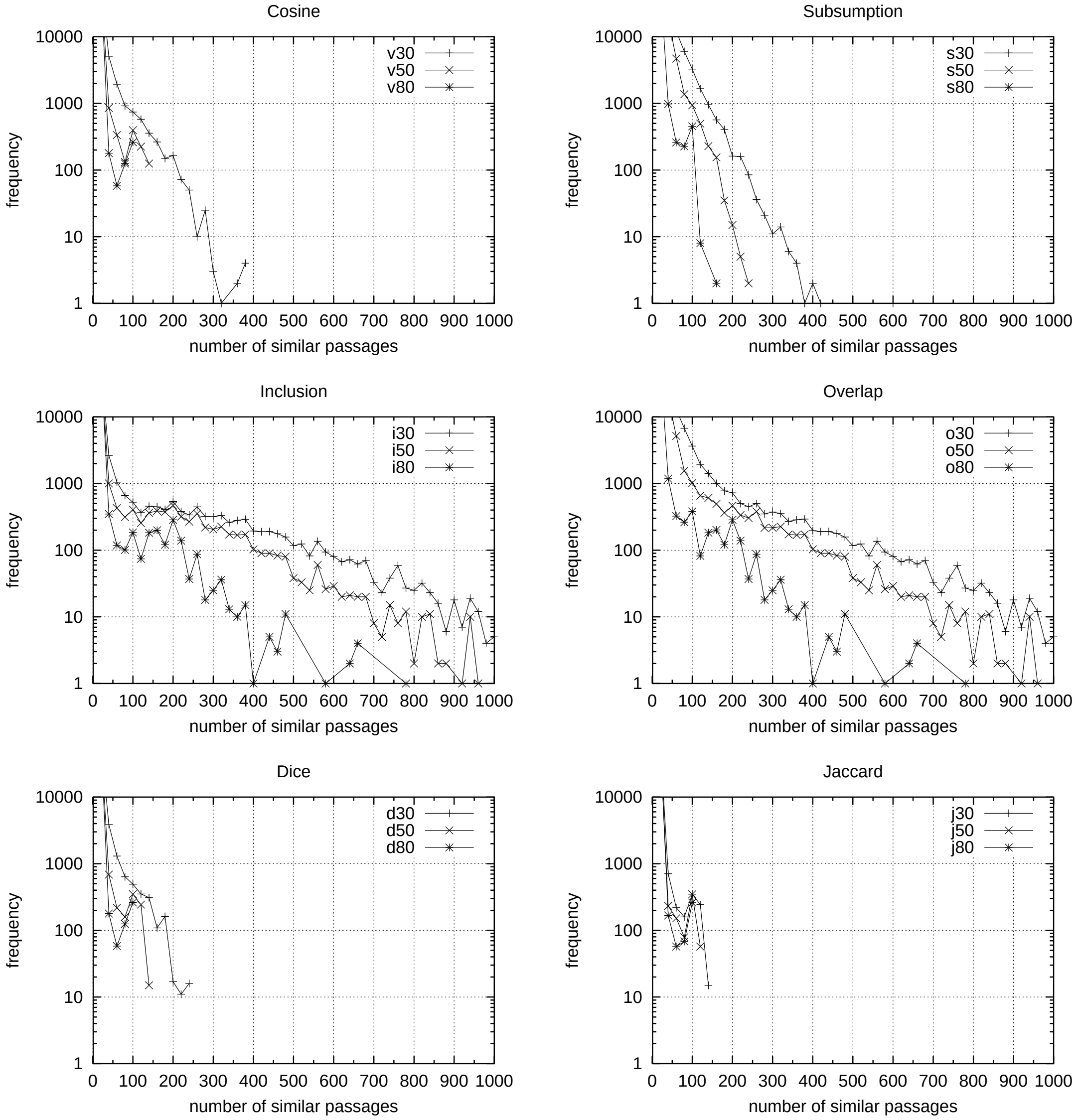


Figure 5.5: Comparison on Different Matching Methods on SR80. (passages contain large number of words.) The words in descending order of frequency (X-axis), number of words in tolerance class (Y-axis).

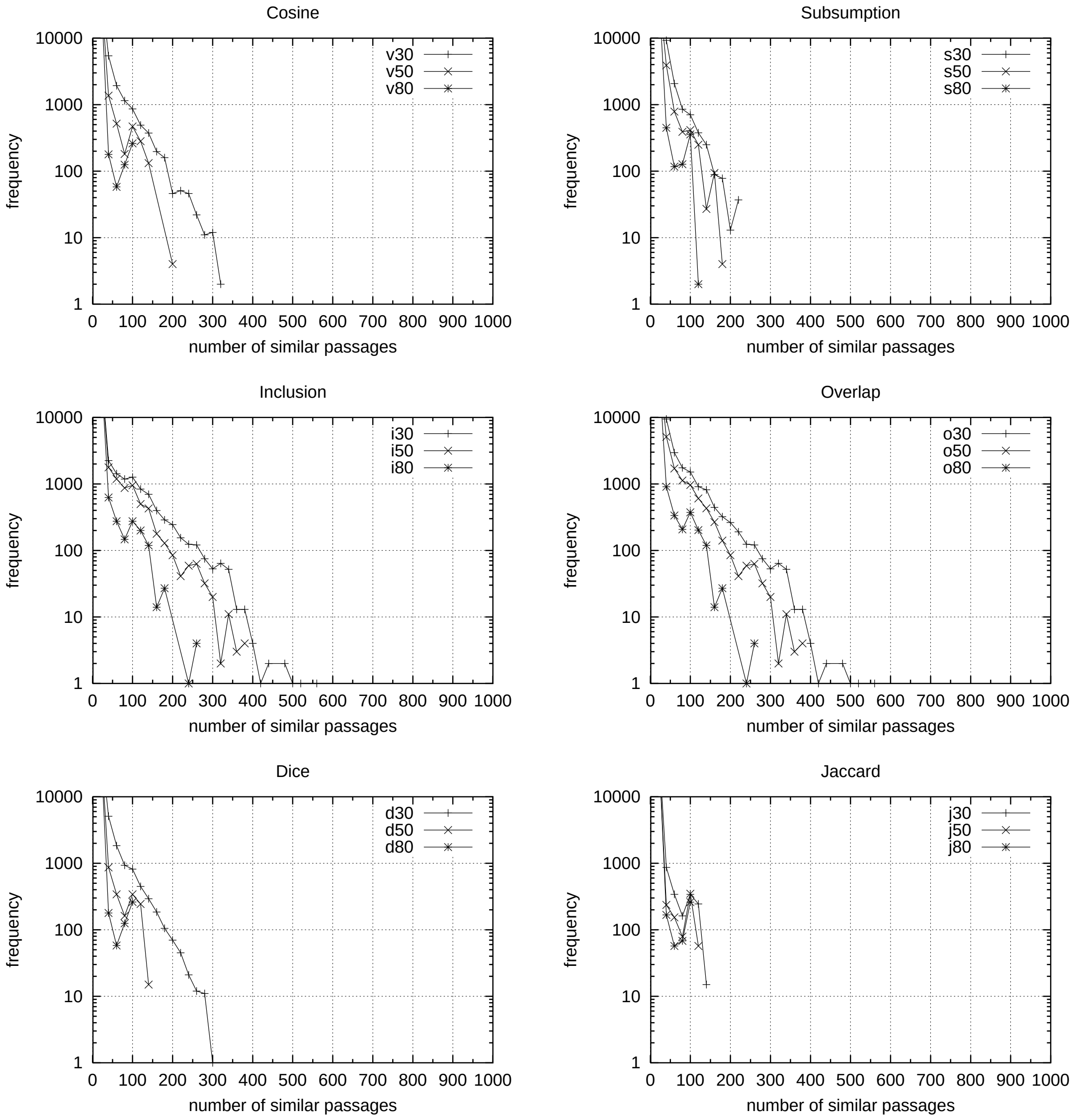


Figure 5.6: Comparison on Different Matching Methods on SR40. (passages contain small number of words) The words in descending order of frequency (X-axis), number of words in tolerance class (Y-axis).

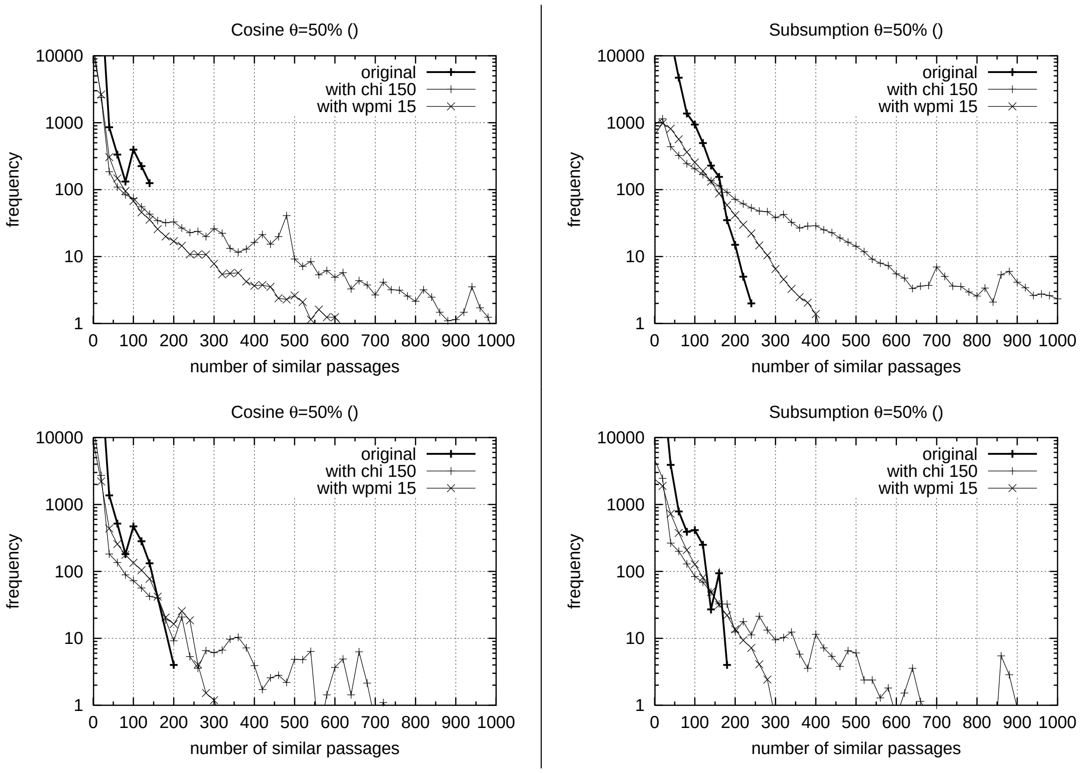


Figure 5.7: Comparison on Number of Similar Passages before and after vocabulary enriching. Number of similar passages (X-axis), and its frequency (Y-axis). First row is the comparison based on sr80, and for second row, on sr40. Left column is on cosine measures and right column is on subsumption measure.

are generating tolerance class of less important words, that is, the words with very low or very large frequencies. But note that we are just comparing the number of similar passages regardless of how they matched. Chances are that chi-square generated the tolerant classes with many not appropriate words. Not appropriate here mean, it is not replacable with the original word. With some level of overlook those kinds of errors, we could say the vocabulary enriching is working effectively.

5.4 Mined Rules

The some example of the results from LATM is shown in Figure.5.8. It shows the rules that could be interpreted. Although the stemming is making it somehow difficult but we could see the first set as items of “land dividing long beach”, “county life”, and “residents complaints”.

```
(land-long-beach-divid) (counti-life) (resid-complaint) [s:0.01]
(surpris-plai-move-expect) (hockei-player-talent-good) [s:0.01]
(school-roughli-kindergarten-colleg) (report-capitol)
                    (reduct-million-program-senior) [s:0.01]
```

Figure 5.8: Example of Rules with Soft Matching (1 month) selected manually by interest

5.5 Evaluation with Sense Entropy

In order to evaluate the interpretability of the rules, we defined a *sense entropy* of the rules. Many words has many senses, in the other words the word has the ambiguity (or uncertainty) about the sense the word my take. The idea of sense entropy is to measure how much ambiguity the word has by means of counting the possible combinations of sense they might take.

Definition Sense Entropy of Rules Let $W = \{w_1, w_2, \dots, w_k\}$ be the words, and let the rule r be a set of words ($r \subseteq W$). The sense entropy of a passage($H(p)$) is the sum of the entropy of words in the passage $H(p) = \sum_k H(w_k)$ The sense entropy of a rule ($H(r)$) is the sum of the entropy of words or passages in the rule $H(r) = \sum_k H(w_k)$ or $H(r) = \sum_k H(p_k)$

Definition Sense Entropy of Words The word w in sense s is denoted as w_s.h. S_w be the total number of senses of the word w. The sense entropy of a word w is defined as

$$H(w) = \sum_{k}^{S_w} p(w_k) log \frac{1}{p(w_k)}.$$

The probability of word w taking the sense s ($p(w_s)$) is calculated from following formula.

$$p(w_s) = \frac{c(w_s) + \dfrac{1}{S_w}}{\sum_k^{S_w} c(w_k) + 1} \tag{5.1}$$

where $c(w_s)$ is occurrence of word w taking the sense s, collected from some kinds of concordance database. If there is two or more part-of-speech, the entropy is calculated separately and average take the average. The entropy for unseen words is set to zero.

Equation.5.1 is the probability of the word w taking sense s with smoothing for unseen events, using Lidstone's law [MS99] using $\lambda = \frac{1}{S_w}$. The occurrence information $c(w_s)$ is taken from WordNet's frequency info [Fel98].

I chose `WordNet::SenseRelate` v0.01 [PBP03], as the word sense disambiguation tool. `WordNet::Similarity::lin` was used for the similarity measure.

We compared the next three settings.

1. No WSD: Sum of all word entropies
2. Local WSD: WSD applied using the words within the passages
3. Global WSD: WSD applied using whole word in passages

The evaluating has been done on few rules. Here I show one the rule as example.

```
{ england income form tax bank receive united deposited }
{ tax forgiven country pension united england adviser
                              taxe income foreign state } (support:46)
```

The Table.5.4 shows the entropy of the word "form" for noun. The total frequency of "form" in nouns is 272. This rule includes two passages, 19 words. the total entropy of "form" in nouns is 2.45, and 2.48 in verbs. So the final value of entropy for "form" in all part of speech is 2.47.

Please note that this result is tentative results, no statistical significance verified yet. It requires human validation of word senses correctly identified, we could not accomplish thorough evaluation on this method, but will be verified in future work.

Another approach is to reduce or rank the rules according to additional rule measure of rule interestingness. Our system has no rule evaluation module yet, but there have been many works on developing measures for rule interestingness [TK00]. Some works try to find rules using background knowledge to show only the unusual or unexpected rules. Especially [BMPG01] proposed a measure for text-mined rules, in which they utilized the lexical knowledge to evaluate the text-mined rules' semantic surprisingness. Similar thing can be thought in our work.

Another way to overcome this problem is to extend the successful works from standard data mining community. For example, mining of frequent closed item sets has been studied [PHM00], and is successful in removing redundant rules.

Sense ID	Freq.	Prob.	Entropy
1	96	0.35	0.53
2	76	0.27	0.51
3	40	0.14	0.40
4	23	0.08	0.30
5	19	0.07	0.27
6	8	0.03	0.15
7	3	0.01	0.07
8	2	0.01	0.05
9	2	0.01	0.05
10	1	0.004	0.03
11	1	0.004	0.03
12	1	0.004	0.03
13	0	0.0002	0.003
14	0	0.0002	0.003
15	0	0.0002	0.003
16	0	0.0002	0.003

Table 5.4: Frequency, Probability, and Entropy of each sense of “form” in nouns

	No WSD	Global WSD	Local WSD
Entropy allowing errors	13.08	**1.50**	4.45
Number of WSD errors	-	6	4
Entropy disallowing errors	13.08	10.34	**10.00**

Table 5.5: Comparison of Rule Entropies allow/disallowing error

Chapter 6

Discussions and Future Works

In this section, we discuss some of the problems that became clear after the experiment. We also give some possible improvement or refinement.

6.1 Passage Representation

In some example we still could see that some passages are still too short (ex. (place home) (surprise)), too long (ex.(baker secretary bank debtor call orient iii strategy treasury dealing designed previous commercial)), or even uninterpretable (ex.(district lik spent)). In the future work we need to work around to avoid the use of these passage or develop a better algorithm to find good keyword selection.

In my observation, representation and the use of contextual information, is receiving increasing attention. Some of the representative works are Topic Sensitive PageRank [Hav02], Context-sensitive Text Categorization [CS96], KeyGraph [YAC99], and Clustering Similar Contexts [PPK05]. Usually vectors of words are used as representation of contexts but this requires high dimension, large memory. If the contexts information could be represented in compact or fast comparing way, it can also be applied to many other domain.

Another approach for solution is that use more sophisticated passage identification algorithm, since even with the good keyword selection algorithm, we could imagine that we could not expect to have good keywords with length of 3 representing a passage that lasts for few pages. So the solution is the split the long passage into more concrete and precise passages.

6.2 Soft Passages Matching and Vocabulary Enriching

The target of our work is to reflect the contexts. We apply soft matching and vocabulary enriching in the process. Both are essential part of this work but there is also an problem. The loose use of these might break the contexts. For example, there is passage fujisawa advisor supreme vice die, by simple sets operations, we could drop the word "die", which might significantly change the context. We must be careful not to put the loose threshold or we might get a screwed results, or unrealistic support counts. For improvement for this task, soft matching which does not change the context is required. One way is to use the lower approximation of TRSM. Lower approximation is the core of the sets, so we could put some constraints in soft matching, that it can soft match as long as the lower approximation will not change.

Similar problem will happen when we enrich the passage with not precise replace of words. We need more analysis on the tolerant class generated.

6.3 Evaluation of Rule sets

Evaluation of rule sets is one of the important task in association rules mining. Traditional Evaluation are mostly on individual rules but, as to my knowledge there not much work to evaluate the quality of whole ruleset. For descriptioning task it is quite useful to know what kind of tendency the algorithm has in mining. Clustering, is also the description tool, has similar problem but it seems the issue of validating cluster has more attention than in association rules mining. We might be seeking for some improvements on this topic.

6.4 Resemblance with *prototypical document*

In initial step they find rules in documents but they did not got good results [RB97].After that they made improvements after the initial results. They changed to use terms instead of words, and they proposed of finding what they call a *prototypical documents*.

Their definition of *prototypical document* is given [RB98]:

Definition A *prototypical document* is informally defined as document corresponding to an information that occurs in a repetitive fashion. *i.e.* a document representing a class of similar documents in the textual base.

Example of such a result can be seen in Figure.6.1.

Resulting Cluster:
{due available management priced issuing combined denominations listed underwriting luxembourg payment_date paying} 45

Most frequent sequential decomposition:
(issuing due paying priced)(available denominations listed luxembourg) (pay-ment_date)(management underwriting combined) 41

Figure 6.1: Example of Prototypical Document [RB98]

After mining on the words level, they cluster the rules by the words rules share and splitting the words in a rule into set of items by looking into the original document and paragraph boundaries.

The *prototypical document* has quite similar output to the ones of ours. If just looking the output rule they may seem like the same, Although we did not have the chance to reproduce and confirm the differences, but in our approach, a word that has relatively lower frequency but has significance in a certain passage may appear in output rules, while words in prototypical documents are required to suffice support threshold.

Incorporating Semantic Information

In experiment we used one kind of ontology WordNet, just for evaluating purpose. In future work we planning to use ontology for more precise vocabulary enriching or better soft matching. Anyway there are several issues to be tackled when using ontologies in soft matching. First the word sense must be disambiguated before mapping words to ontologies. Another problem is that similarity of words using certain ontologies can be defined in many measures [BH01].

Chapter 7

Conclusion

In this paper, mining associations between contexts have been proposed. The methodologies are described in details. Experimental evaluation showed that context-level has two advantages: (1) interpretation is easier than word-level representation in the means of number of possible combination of senses, and (2) use semantical units to improve usefulness of rules. The measure used in (1) is also one contribution of this paper. Although the measures for description tasks are often hard to define, this work is significant in that this is one of the few approaches that quantitatively measure the quality of text-mined rulesets (this is different from the interestingness of individual rules). From the experiments we showed that our approach gives rules that are more to understand. The current problem is mainly concerned to efficiency. Current algorithm takes time complexity and space complexity. Also the further analysis and evaluation is required in near future.

Bibliography

[AHKV97] Helena Ahonen, Oskari Heinonen, Mika Klemettinen, and A. Inkeri Verkamo. Applying data mining techniques in text analysis. Technical Report Report C-1997-23, Department of Computer Science, University of Helsinki, 1997.

[AIS93] Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Mining association rules between sets of items in large databases. In Peter Buneman and Sushil Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., 26–28 1993.

[AS94] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.

[BF98] Ho Tu Bao and Kaneme Funakoshi. Information retrieval using rough sets. *Journal of Japanese Society for Artificial Intelligence*, 13(3):424–433, 1998.

[BH01] A. Budanitsky and G. Hirst. Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures. In *Workshop on WordNet and Other Lexical Resources, in the North American Chapter of the Association for Computational Linguistics (NAACL-2001)*, 2001.

[BMPG01] Sugato Basu, Raymond J. Mooney, Krupakar V. Pasupuleti, and Joydeep Ghosh. Evaluating the novelty of text-mined rules using lexical knowledge. In *Knowledge Discovery and Data Mining*, pages 233–238, 2001.

[CC99] Chris Clifton and Robert Cooley. Topcat: Data mining for topic identification in a text corpus. In *Principles of Data Mining and Knowledge Discovery*, pages 174–183, 1999.

[Cor99] Cornell University. SMART system. `ftp://ftp.cs.cornell.edu/pub/smart/`, 1999.

[CS96] William W. Cohen and Yoram Singer. Context-sensitive learning methods for text categorization. In Hans-Peter Frei, Donna Harman, Peter Schäuble, and Ross Wilkinson, editors, *Proceedings of SIGIR-96, 19th ACM International*

Conference on Research and Development in Information Retrieval, pages 307–315, Zürich, CH, 1996. ACM Press, New York, US.

[DDL$^+$90] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[FAF$^+$99] Ronen Feldman, Yonatan Aumann, Moshe Fresko, Orly Lipshtat, Binyamin Rosenfeld, and Yonatan Schler. Text mining via information extraction. In *PKDD '99: Proceedings of the Third European Conference on Principles of Data Mining and Knowledge Discovery*, pages 165–173. Springer-Verlag, 1999.

[FBY92] William B. Frakes and Ricardo Baeza-Yates, editors. *Information retrieval : data structures & algorithms.* Prentice Hall, 1992.

[Fel98] C. Fellbaum, editor. *WordNet, an electronic lexical database.* MIT Press, 1998.

[FFK$^+$98] Ronen Feldman, Moshe Fresko, Yakkov Kinar, Yehuda Lindell, Orly Liphstat, Martin Rajman, Yonatan Schler, and Oren Zamir. Text mining at the term level. In *PKDD '98: Proceedings of the Second European Symposium on Principles of Data Mining and Knowledge Discovery*, pages 65–73. Springer-Verlag, 1998.

[Hav02] T. Haveliwala. Topic-sensitive pagerank. In *In Proceedings of the Eleventh International World Wide Web Conference, Honolulu, Hawaii, May 2002.*, 2002.

[Hea97] Marti A. Hearst. Texttiling: segmenting text into multi-paragraph subtopic passages. *Comput. Linguist.*, 23(1):33–64, 1997.

[Hea99] M. Hearst. Untangling text data mining. In *In the Proceedings of ACL'99: the 37th Annual Meeting of the Association for Computational Linguistics*, 1999.

[HMS01] David J. Hand, Heikki Mannila, and Padhraic Smyth. *Principles of Data Mining.* Bradford Books, August 2001.

[HN02] Tu Bao Ho and N. B. Nguyen. Nonhierarchical document clustering by a tolerance rough set model. *International Journal of Intelligent Systems*, 17(2):199–212, 2002.

[Koz93] Hideki Kozima. Text segmentation based on similarity between words. In *Proceedings of the 31st conference on Association for Computational Linguistics*, pages 286–288. Association for Computational Linguistics, 1993.

[KZ97] Marcin Kaszkiel and Justin Zobel. Passage retrieval revisited. In *SIGIR '97: Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 178–185. ACM Press, 1997.

[MH91] Jane Morris and Graeme Hirst. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Comput. Linguist.*, 17(1):21–48, 1991.

[MS99] Christopher D. Manning and Hinrich Schutze. *Foundations of Statistical Natural Language Processing.* The MIT Press, 1999.

[MT94] Okumura Manabu and Honda Takeo. Word sense disambiguation and text segmentation based on lexical cohesion. In *Proceedings of the 15th conference on Computational linguistics*, pages 755–761. Association for Computational Linguistics, 1994.

[Nah04] Un Yong Nahm. *Text Mining with Information Extraction.* PhD thesis, Department of Computer Sciences, University of Texas at Austin, 8 2004.

[NM01] Un Yong Nahm and Raymond J. Mooney. Mining soft-matching rules from textual data. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*, pages 979–984, 2001.

[Pac97] Peter S. Pacheco. *Parallel Programming with MPI.* Morgan Kaufmann Publishers, 1997.

[Pai90] C.D. Paice. Another stemmer. *SIGIR Forum*, 24(3):56–61, 1990.

[Paw91] Z Pawlak. *Rough sets: Theoretical Aspects of Reasoning about Data.* Kluwer Academic Publishers, 1991.

[PBP03] Ted Pedersen, Satanjeev Banerjee, and Siddharth Patwardhan. Maximizing semantic relatedness to perform word sense disambiguation. *(submitted)*, 2003.

[PHM00] J. Pei, J. Han, and R. Mao. Closet: An efficient algorithm for mining frequent closed itemsets. In *DMKD 2000*, pages 11–20, 2000.

[Por80] M.F. Porter. An algorithm for suffix stripping. *Program*, 14:130–137, 1980.

[PPK05] Ted Pedersen, Amruta Purandare, and Anagha Kulkarni. Name discrimination by clustering similar contexts. In *CICLing*, pages 226–237, 2005.

[Raj97] Martin Rajman. Text mining, knowledge extraction from unstructured textual data. In *Proceedings of EUROSTAT Conference*, 1997.

[RB97] M. Rajman and R. Besançon. Text mining: Natural language techniques and text mining applications. In *Proceedings of the seventh IFIP 2.6 Working Conference on Database Semantics (DS-7), Chapam & Hall IFIP Proceedings serie, (1997) Oct 7-10.*, 1997.

[RB98] M. Rajman and R. Besançon. Text mining - knowledge extraction from unstructured textual data. In *Proc. of 6th Conference of International Federation of Classification Societies (IFCS-98)*, pages 473–480, Roma (Italy), jul 1998.

[RT02] Prabhakar Raghavan and Panayiotis Tsaparas. Mining significant associations in large scale text corpora. In *IEEE International Conference on Data Mining (ICDM'02)*, 2002.

[SA96] Ramakrishnan Srikant and Rakesh Agrawal. Mining quantitative association rules in large relational tables. In H. V. Jagadish and Inderpal Singh Mumick, editors, *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, pages 1–12, Montreal, Quebec, Canada, 4–6 1996.

[SAB93] Gerard Salton, James Allan, and Chris Buckley. Approaches to passage retrieval in full text information systems. In *SIGIR*, pages 49–58, 1993.

[Sal89] Gerard Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer.* Addison-Wesley, 1989.

[SS94] A. Skowron and J. Stepaniuk. Generalized approximation spaces. In *The 3rd International Workshop on Rough Sets and Soft Computing*, pages 156–163, 1994.

[The04] Thanaruk Theeramunkong. Applying passage in web text mining. *International Journal of Intelligent Systems*, 19(1-2):149–158, 2004.

[TK00] P. Tan and V. Kumar. Interestingness measures for association patterns: A perspective. Technical Report TR00-036, Department of Computer Science, University of Minnesota, 2000.

[TOK99] Tokenobu TOKUNAGA. *Information Retrieval and Natural Language Processing*, volume 5 of *Computation and Language*. University of Tokyo Press, 1999.

[Tre02] Nathan Treloar. Text mining: Tools, techniques, and applications. In *Knowledge Technologies Conference*, 2002.

[WM03] Takashi Washio and Hiroshi Motoda. State of the art of graph-based data mining. *SIGKDD Explor. Newsl.*, 5(1):59–68, 2003.

[YAC99] Yukio OHSAWA Nels E.BERNSON Masahiko YACHIDA. Keygraph: Automatic indexing by segmenting and unifying co-occurrence graphs. *IEICE*, J82-D1(2):391–400, feb 1999.

[ZH02] M. J. Zaki and C.-J. Hsiao. Charm: An efficient algorithms for closed itemset mining. In *In R. Grossman, J. Han, V. Kumar, H. Mannila, and R. Motwani, editors, Proceedings of the Second SIAM International Conference on Data Mining*, 2002.

Appendix A

Perl Code for Stemming

```
package Stemmer;
require Exporter;
@ISA = qw(Exporter);
@EXPORT = qw(stem);
@EXPORT_OK = qw();

sub stem($)
{
    s/\’s$//;
    # s stemmer
    if(/s$/){
        s/([bcdghklmnpqrtvwy])s$/$1/; # ous x z
        s/sses$/ss/; #businesses
        s/([bcdefgklmnoprstuvwxyz]e)s$/$1/;
        s/hes$/h/;      # clothes
        s/ˆ(\w)ies/$1ie/; # lies
        s/([bcdfghklmnprstvz])ies$/$1y/; # companies # series
    }elsif(/ed$/){
        s/(\w)\1ed$/$1$1/; # scrubbed
        s/ˆ(\w)ied/$1ie/; # tied
        s/([rnfp])ied$/$1y/; # carried
        s/([bcdfgilmnprtuv])ated$/$1ate/;
        s/([o])ted$/$1te/;
        s/([cnprsl])ted$/$1t/; # resulted
        s/([cgsuvz])ed$/$1e/; # managed
        s/([hkowxy])ed$/$1/; # published
    }
    return $_;
}

1;
```

Appendix B

Perl Code for Sentence Detection

```perl
package Sentence;
require Exporter;
@ISA = qw(Exporter);
@EXPORT = qw(sentence_split);
@EXPORT_OK = qw();

sub sentence_split($)
{
    my $words = shift;
    #$words =~ s/[:;]/ \. /g;
    # remove consequtive periods to one
    $words =~ s/^\.(\s*\.)*//;
    # right?" My boss => right. My boss
    $words =~ s/[\?\!]([\"\']?\s*[A-Z])/\.$1/g;
    # ? and ! to .
    $words =~ s/[\?\!]/\./g;
    # "... done." He said => done. He said
    $words =~ s/\.([\"\'])\s*([^a-z])/$1 \. $2/g;
    # "... done." EOF => done. EOF
    $words =~ s/\.([\"\'])\s*$/$1 \./;
    # .) or ., => ) or ,
    $words =~ s/\.([,\)])/$1/g;
    # 14.99 => 14_99
    $words =~ s/\.(\d+)/_$1/g;
    # 2,3,4 => 234
    $words =~ s/,(\d+)/$1/g;
    # ^100. => 100
    $words =~ s/^(\d+)\./$1 /gi;
    # ^A. => A
    $words =~ s/^(\w)\./$1 /gi;
    # L.A.P.D. => LAPD
    $words =~ s/\.(\w+)/$1/g;
    # remove period that follows well known abbrev
    $words =~ s/(Prof|PP|pp|Mr|Mrs|Ms|St|
                 Jan|Feb|Mar|Apr|Jun|Jul|Aug|Sep|Oct|Nov|Dec|
                 Mon|Tue|Wed|Thu|Fri|Sat|Sun)\./$1/g; [ one line ]
    # \sA. => A
    $words =~ s/(\s[A-Z])\./ $1/g;
    # ^A. => A
    $words =~ s/^([A-Z])\./$1/g;
    # remove period that follows well known abbrev 2
    $words =~ s/(Jr|vs|Vs|etc|Us|US|us|LA|Co)\.(\s*[^A-Z])/$1$2/g;

    # separate periods from words
    $words =~ s/([^\.]{2,})\./$1 \./g;
    # remove redundant periods from EOF
    $words .= " ." if($words !~ /\.\s*$/);

    if(wantarray){
        # return in list of sentences
        return split(/\s*\.\s*/, $words);
    }else{
        # return in one string
        return $words;
    }
}
1;
```